Technische Berichte des Hasso-Plattner-Instituts für Softwaresystemtechnik an der Universität Potsdam

Technische Berichte des Hasso-Plattner-Instituts für
Softwaresystemtechnik an der Universität Potsdam | 88

Christoph Meinel | Andreas Polze | Gerhard Oswald | Rolf Strotmann |
Ulrich Seibold | Bernhard Schulzki (Hrsg.)

HPI Future SOC Lab

Proceedings 2013

Universitätsverlag Potsdam

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de/ abrufbar.

Universitätsverlag Potsdam 2014
http://verlag.ub.uni-potsdam.de/

Am Neuen Palais 10, 14469 Potsdam
Tel.: +49 (0)331 977 2533 / Fax: 2292
E-Mail: verlag@uni-potsdam.de

Die Schriftenreihe **Technische Berichte des Hasso-Plattner-Instituts für Softwaresystemtechnik an der Universität Potsdam** wird herausgegeben von den Professoren des Hasso-Plattner-Instituts für Softwaresystemtechnik an der Universität Potsdam.

ISSN (print) 1613-5652
ISSN (online) 2191-1665

Druck: docupoint GmbH Magdeburg

ISBN 978-3-86956-282-7

Zugleich online veröffentlicht auf dem Publikationsserver der Universität Potsdam:
URL http://pub.ub.uni-potsdam.de/volltexte/2014/6819/
URN urn:nbn:de:kobv:517-opus-68195
http://nbn-resolving.de/urn:nbn:de:kobv:517-opus-68195

Contents

Configure in-memory database management systems for councils data to respond to citizen and business requirements on-demand using HANA technology

Ardavan Amini, Senior Lecturer
Birmingham City University
Technology, Engineering and Environment Faculty
International Academy of Enterprise Systems innovation
Ardavan.amini@bcu.ac.uk

Abstract

With innovation at the heart of organizations, and advancement in technology platforms such as in-memory data management, the organization are beginning to think how in-data-memory management can assist for the delivery of services and data to citizens and businesses based on demand.

1. Introduction

Birmingham City Council has possessed many different datasets gathered from many years. The datasets gathered are all from an open data platform, meaning that each person in the public can have access to this information. With the information they are allowed to do what ever they want to it and with it, whether it is for positive or negative uses.

Currently the data held is usually just a set of numbers, and to most of the general public, this information will not be useful information. There does not seem to be any purpose for these datasets, which is why this project has been created to tackle this problem. Due to this Birmingham City Council have decided to invest in new technologies in which to utilise these datasets, for a more effective purpose.

The data sets held are dull and lack variety of style and presentation. The project at hand will be to aggregate the information and display the data in a more elaborate form. These will be in the form of colourful dashboards and instantly customisable datasets.

Even though there are many different datasets available, the one that is being investigated for the project currently, are the ones concerning traffic within different locations. The reason for this selection will be highlighted below. So to summaries the main aim is to aggregate this data, cleanse the data and transform it into a modified form so that businesses and civilians of Birmingham will be able to utilise this information.

2. Case Study & Rational

With civilians being able to access on-demand information, they will be able to access traffic information instantly and for their routes that they are going to. This will allow civilians be more intuitive with there environment. For example users will be able to have live updates/feeds on mobile devices, showing where traffic volume is greatest or when transportation times are delayed instantly. This will mean fewer civilians will be late to there destinations, or allow users to forward plan better, just in case an emergency has occurred.

With all these live feeds of traffic and transportation information, users will be able to forecast future events. For example if there has been a large amount of snow, civilians will be to view previous occasions when the same occurred and see which transport services were not running or which roads were best avoided. Along with this information Birmingham city council, can use this data to improve services. Using the same example of snow, gritters could be sent out to prime locations to help improve road services where problems had occurred before. This could also be true for railway services and bus routes that had previously been terminated due to the weather.

With the access to live traffic information, it will allow more efficient traffic flow. Birmingham city council members will be able to monitor when set protocols need to be put in place in order to minimise the amount of traffic volume along certain roads. This may include a system of installing part time traffic lights that automatically activates if traffic flow is high. Or the reverse could also be applied. If many drivers were exceeding the speed limit along a certain road, then the council could implement a set of speed cameras or variable speed limit restrictions could be put in place to reduce or avoid future accidents.

With the constant live feeds of traffic data, this will allow integration with GPS devices to show the best route to take to avoid congestion. Even though this

technology already exists it is not always up to date and can cause lag, but with modern day 4G technologies especially, the download speed of data should be sufficient to be live times.
As mentioned before the council will be able to see where congestion is greatest. Even though immediate impacting protocols could be put in a place, other long-term improvements could be made. With the traffic information at the hands of the council, it will allow them to create new roads, or increase the number of lanes in high traffic volume areas.
Along with the point mentioned above. The fact that the United Kingdom and the EU are trying to reduce their carbon footprint, something will need to be done. To do this they are reduce the number of vehicles on the roads. With all the information about road layouts and congestion, the council will be able to entice civilians into taking trains or buses to set locations by providing new routes that go to high volume traffic places or build new stations to accommodate this factor. This will help reduce the volume of traffic along with reducing the carbon footprint.
By knowing the layout of the roads and where high volume of traffic occurs, this will cause increased delays for emergency services. Emergency services will be able to access live information on which route to take to avoid areas of high congestion, this will allow emergency vehicles to reach destinations more effectively.
Using the traffic datasets, it is possible to highlight accident black spots within different locations. By highlighting these key areas, the council will be able to improve roads, to avoid any further incidents or if not implement speed awareness protocols so that the number of accidents is reduced.

3 Key Outcomes and Success Criteria

The main key outcomes for the project at hand are to:

- Improve road structure
- Reduce congestion
- Improve public transport services
- Increase volume using public transport, in turn reduce number of cars on the road
- Reduce carbon footprint produced by cars
- Increase efficiency of emergency services ability to reach destinations
- Highlight areas that have been previously affected by harsh weather and make sure roads and rails are clear for the following years to avoid any accidents or congestion
- Avoid accident black spots by improving road layout or putting safety measures in place

4 Options/Costs

With the ever advances in mobile technologies, civilians will be able to access information quick and effectively via mobile devices. With the introduction of 4G technologies, users will be able to access information at lighting speeds. With this is mind; a lot of cloud-based applications could use via mobiles phones. Treating the phone as the tool for accessing the data. The cloud-based applications will be a one off low cost payment. The application will be able to access traffic updates, train times, and roads to avoid and many other different criteria could be set for users of the application.
As mentioned before there could be increased accuracy of GPS system integration. The traffic updates will be live and on demand. Even though this will depend on the device being use it should not be too high cost to implement and run.
Two other systems that could be implemented to present the data are using SAP HANA or SAP BI Crystal Reports. SAP Crystal report has been around for several years and allows users to view highly detailed/colourful dashboards. Users are able to set certain criteria based on there needs/demands. The only drawback of this is that the datasets used must be of legacy/historical data. This is where SAP HANA can be introduced. This is a cloud based platform that is able to access live/on demand datasets. SAP HANA processes the data to provide effective dashboards and gives users the information they need quickly and efficiently.
Before the Council would be able to make decisions to improve congestion, they will need to analyse and synthesize information. Hence our proposal is to use SAP HANA technology, with its widely known feature to aggregate data at an impressive speed, integrate it with the Councils SAP back end database in real time, and produce meaningful outputs such as dashboard, RSS feeds, etc.
There are other options out there but its not as good as HANA which uses in-memory technology.

5 Outline Plan

This is a highly basic plan but the main part for this project to aggregate the datasets that needs to be used. This dataset then needs to be cleansed to make sure that it does not contain any anomalies or mistakes. From this it will be then get transformed for use with certain applications or tools/options that can be used. Once this has been completed the product can be tested and once everything has been completed it can then be deployed for commercial use.

Service-Based 3D Rendering and Interactive 3D Visualization

Benjamin Hagedorn
Hasso-Plattner-Institut for
Software Systems Engineering
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
benjamin.hagedorn@hpi.uni-potsdam.de

Jürgen Döllner
Hasso-Plattner-Institut for
Software Systems Engineering
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
doellner@hpi.uni-potsdam.de

Abstract

This report describes the subject and preliminary results of our work in the context of the HPI Future SOC Lab. This work generally aims on exploiting high performance computing (HPC) capabilities for service-based 3D rendering and service-based, interactive 3D visualization. A major focus is on the application of HPC technologies for the creation, management, analysis, and visualization of and interaction with virtual 3D environments, especially with complex 3D city and landscape models.

1 Introduction

Virtual 3D city models represent a major type of virtual 3D environments. They can be defined as a digital, geo-referenced representation of spatial objects, structures and phenomena of a distinct geographical area; its components are specified by geometrical, topological, graphical and semantic data and in different levels of detail.

Virtual 3D city models are, e.g., composed of digital terrain models, aerial images, building models, vegetation models, and city furniture models. In general, virtual 3D city models serve as information models that can be used for 3D presentation, 3D analysis, and 3D simulation. Today, virtual 3D city models are used, e.g., for urban planning, mobile network planning, noise pollution mapping, disaster management or 3D car and pedestrian navigation.

In general, virtual 3D city models represent prominent media for the communication of complex spatial data and situations, as they seamlessly integrate heterogeneous spatial information in a common reference frame and also serve as an innovative, effective user interface. Based on this, virtual 3D city models, as integration platforms for spatial information, represent building blocks of today's and future information infrastructures.

1.1 Complexity of 3D city models

Virtual 3D city models are inherently complex in multiple dimensions, e.g., semantics, geometry, appearance, and storage. Three major complexities are described in the following.

Massive amounts of data: Virtual 3D city models typically include massive amounts of image data (e.g., aerial images and façade images) as well as massive amounts of geometry data (e.g., large number of simple building models as well as smaller number of buildings modeled in high detail). Vegetation models represent another source of massive data size; a single tree model could contain, e.g., approximately 150,000 polygons.

Distributed resources: In today's so called geospatial data infrastructures (GDIs), the different components (i.e., base data) of virtual 3D city models as well as functionalities to access, and process (e.g., analyze) virtual 3D city models can be distributed over the Internet. In specific use cases such as emergency response scenarios, they need to be identified, assembled, and accessed in an ad-hoc manner.

Heterogeneity: Virtual 3D city models are inherently heterogeneous, e.g., in syntax (file formats), schemas (description models), and semantics (conceptual models).

As an example, the virtual 3D city model of Berlin contains about 550,000 building models in moderate and/or high detail, textured with more than 3 million single (real-world) façade textures. The aerial image of Berlin (covering an area of around 850 km²) has a data size of 250 GB. Together with additional thematic data (public transport data, land value data, solar potential) the total size of the virtual 3D city model of Berlin is about 700 GB.

1.2 Service-based approach

The various complexities of virtual 3D city models have an impact on their creation, analysis, publishing, and usage. Our overall approach to tackle these complexities and to cope with these challenges is to design and develop a distributed 3D geovisualization

system as a technical framework for 3D geodata integration, analysis, and usage. For this, we apply and combine principles from Service-Oriented Computing (SOC), general principles from 3D visualization systems, and standards of the Open Geospatial Consortium (OGC).

To make complex 3D city models available even for small devices (e.g., smart phones, tablets), we have developed a client/server system that is based on server-side management and 3D rendering [1]: A portrayal server is hosting a 3D city model in a preprocessed form that is optimized for rendering, synthesizes images of 3D views of this data, and transfers these images to a client, which (in the simplest case) only displays these images. By this, the 3D client is decoupled from the complexity of the underlying 3D geodata. Also, we can optimize data structures, algorithms and rendering techniques with respect to specialized software and hardware for 3D geodata management and 3D rendering at the server-side.

Our project in the context of the HPI Future SOC Lab aims on research and development of how to exploit the Lab capabilities for the development and operation of such a distributed 3D visualization system, especially for 3D geodata preprocessing, analysis, and visualization. The capabilities of interest include the availability of many cores, large main-memory, GPGPU-based computing, and parallel rendering.

2 Project Work & Next Steps

During the last project phase, we continued our research and development on fundamental concepts and techniques in the area of service-based 3D rendering and service-based, interactive 3D visualization. The techniques developed so far rely on and take advantage of multi-core/multi-threading processing capabilities, the availability of large memory, and GPGPU systems as provided by the HPI Future SOC Lab. By exploiting these capabilities, we are able to accelerate processing, management and visualization of massive amounts of 3D geodata in a way that new applications in the area of 3D geovisualization become feasible.

Project work was done in three major areas – processing of massive 3D point clouds, processing of massive 3D city models, and assisted 3D camera control.

2.1 Processing massive 3D point clouds

We have continued our research on the processing, analysis, and visualization of massive 3D point cloud data based on multi-CPU and multi-GPU approaches including algorithms for the spatial organization of massive 3D point cloud data and for the computation of simplified representations of this data.

During last project phases, we could improve speed and quality of the spatial organization and the rasterization of 3D point clouds:

Spatial organization: Organization of massive point clouds is required to efficiently access and spatially analyze 3D points; quadtrees and octrees represent common structures for their organization. For this task we used a PARTREE algorithm, which creates several sub trees, which are combined to a single one.

Rasterization: Rastered 3D point clouds are a central component for visualization techniques and processing algorithms, as they allow efficient access to points within a specific bounding box. Rasterization transforms arbitrary distributed 3D points into a gridded, reduced, and consolidated representation; representative points are selected and missing points are computed and complemented. For this task we implemented a four-step process including: detection of the raster cell a point should be assigned to; ordering the points according to their raster cells; computing representative points for each cell; interpolating points for empty cells. A CUDA-based version of this algorithm was tested with the Future SOC Lab's TESLA system, which resulted in the rasterization of 30.5 million 3D points in 22 minutes in contrast to more than 5 hours on a single-threaded CPU version.

Since then, we have further improved our algorithms and have been working on segmentation algorithms, i.e., analysis algorithms, which allow for the classification of 3D points as part of, e.g., build infrastructure, vegetation, or terrain. Next steps include test and optimize these algorithms by help of the Future SOC Lab to make them ready to serve as a building block of an SOA-enabled 3D point cloud processing and analysis pipeline.

2.2 Processing massive 3D city models

In the field of 3D city model visualization and distribution, we continued our research on the processing of very large 3D city model data from CityGML data, which includes data extraction, geometry optimization and texture optimization. In a first iteration of adjusting our algorithms and testing those with HPC servers, we could significantly reduce the time to process very large sets of texture data (550,000 buildings with real façade textures) from more than a week on a standard desktop PC (2.8 GHz, 8 logical cores, 6 GB RAM) to less than 15 hours on the Future SOC Lab's RX600-S5-1 (48 logical cores, 256 GB RAM).

Also, we continued refactoring of the algorithms and processes used for texture optimization and consolidation, which allows us to generate and manage several texture trees in parallel. Next steps include testing and using this improved technique on the Future SOC Lab's HPC-servers to develop a fast data preprocessing tool chain, which is crucial for coping with continuous updates of the underlying data and

for ensuring that a 3D visualization is up to date as much as possible.

The data preprocessed this way forms the basis for the client/server-based rendering and visualization concept and system described above. Based on this technology, 3D geodata could be served in a scalable way to various platforms [2].

In the field of 3D city model processing, we are also planning to exploit the HPC capabilities for the integration of different massive data sources for 3D city models and additional relevant information, namely data from CityGML models and data from OpenStreetMap to create an information-rich 3D city model, which can be used as a spatial user interface to the original underlying data. Tasks will include to integrate data from CityGML files and OpenStreetMap to come up with an integrated, rich database, which should be as up to data as possible.

Additionally, we plan to flank this work by researching techniques for software-based and parallel rendering of massive 3D city models based on our previous work; we expect new insights in how to efficiently organize and preprocess massive 3D models (geometries, appearance information, thematic data) for cloud-enabled visualization and distribution of complex 3D models and for object-related information retrieval.

2.3 Assisted 3D camera control

In the last project phase, we also worked on algorithms and methods for service-based technologies for assisted interaction and camera control in massive virtual 3D city models. As a first step in such a process we have been developing heuristics, algorithms and a machine learning based process for generating and classifying so called "best views" on virtual 3D city models. Heuristics include geometrical and visual characteristics.

As next step, the algorithms and techniques developed need to be adapted and implemented for even better exploiting the Future SOC processing capabilities. This will allow us to compute a multi-dimensional "navigation space" for complete 3D city models very fast as well as to query in real-time user-specific and task-specific camera positions and paths.

3 Conclusions

This report briefly described the subject and preliminary results of our research and development in the context of the HPI Future SOC Lab as well as intended future work. Work and results were mainly in the areas of processing massive 3D point clouds, processing massive 3D city models, and assisted 3D camera control. Here, we could, e.g., reduce the time required to preprocess raw 3D geodata (CityGML data with geometry and textures; also massive 3D point clouds) and to make this data ready for visualization, analysis, and use. Also, we identified additional opportunities for optimizing these algorithms. More generally, our work leads to new opportunities for research and development on advanced and innovative technologies for the exploitation (e.g., analysis and visualization) of massive spatial 3D data sets.

Acknowledgements

We would like to thank our partner 3D Content Logistics for providing access to their 3D visualization platform as a basis for our work in this field.

References

[1] J. Döllner, B. Hagedorn, J. Klimke: Server-Based Rendering of Large 3D Scenes for Mobile Devices Using G-Buffer Cube Maps. 17th Int. Conference on 3D Web Technology, 2012.

[2] J. Klimke, B. Hagedorn, J. Döllner: Scalable Multi-Platform Distribution of 3D Content, 8th Int. 3D GeoInfo Conference, 2013. *(accepted)*

Smart Wind Farm Control

Patrick Böwe
University of Oldenburg
Department of Computing Science
Uhlhornsweg 84
D-26129 Oldenburg
patrick.boewe@uni-oldenburg.de

Ronja Queck
University of Oldenburg
Department of Computing Science
Uhlhornsweg 84
D-26129 Oldenburg
ronja.queck@uni-oldenburg.de

Michael Schumann
University of Oldenburg
Department of Computing Science
Uhlhornsweg 84
D-26129 Oldenburg
michael.schumann@uni-oldenburg.de

Deyan Stoyanov
University of Oldenburg
Department of Computing Science
Uhlhornsweg 84
D-26129 Oldenburg
deyan.stoyanov@uni-oldenburg.de

Benjamin Wagner vom Berg
University of Oldenburg
Department of Computing Science
Uhlhornsweg 84
D-26129 Oldenburg
benjamin.wagnervomberg@uni-oldenburg.de

Andreas Solsbach
University of Oldenburg
Department of Computing Science
Uhlhornsweg 84
D-26129 Oldenburg
andreas.solsbach@informatik.uni-oldenburg.de

Abstract

The amount of fossil energy resources is limited and the energy extraction is becoming increasingly expensive. In order to meet the energy demand in the long term, renewable energy technologies are promoted. Most notably, wind energy plays a central role in this context. The number of installed and operating wind turbines has risen rapidly over the past years. There are two types of wind farms – onshore (located on the main land) and offshore (located on the open sea). One of the key cost-factors of both types is maintenance. While onshore wind farms are relatively easy to maintain, offshore wind farms cause high maintenance costs. There is a variety of reasons for this: restricted means of transportation, dependency on meteorological conditions as well as a more complex supply chain.

The Smart Wind Farm project aims to broach the issue of increased maintenance efforts of offshore wind turbines and to show possible technical solutions. In addition to gaining knowledge about the topic of maintenance of offshore wind turbines, the focus is on the development of a supporting wind farm maintenance platform based on the in-memory system SAP HANA. The objective of which is to capture the whole data traffic of wind turbines as well as to detect error chains by using data mining methods and using them for a proactive maintenance.

1 Wind Farms

Against the backdrop of global challenges like climate change, growing energy demand, constantly rising prices for primary fossil fuels as well as the Fukushima nuclear disaster, renewable energies are providing an increasingly important contribution to the energy sector [1]. In the Federal Republic of Germany, the last nuclear power plants will be decommissioned by the end of 2022. Until then, the renewable energies will become the supporting pillar of the future energy supply, making up at least 35 percent of the energy mix. In the year 2050, 50 percent of the energy mix will be created by renewable energies [2]. Energy scenarios have shown that wind energy will play a central role in generating electricity in 2050. This requires a massive expansion of wind energy plants, and offshore wind parks in particular. Wind energy offers the most economic and effective potential for expanding renewable energies in the short and medium term [3]. Hence, the number of wind turbines has risen rapidly over the past years (see Figure 1).

Generally, the capacity of a wind turbine (WT) is defined by its rotor diameter. The rotor diameter determines the proportion of the wind flow which is available to the WT for conversion into electric energy. The energy of the wind flow rises to the third power to the wind speed, which increases with the

height above ground level. By building higher towers, the turbines can use increased wind speed and thus realize a higher return.

In order to evaluate and compare wind turbines, the annual energy supply is related to the rated output. This number is called full load hours and depends on the local conditions [5].

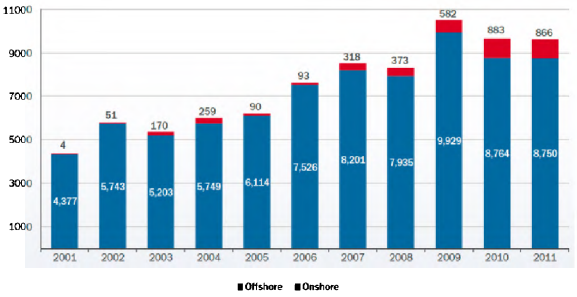


Figure 1: Annual onshore and offshore installations in MW [4]

A distinction is made between offshore and onshore wind farms. Onshore wind farms are located on the main land and can be subdivided into landscape categories. A normal onshore wind turbine produces 2 to 3 MW. The German average for a new WT is between 1552 and 1667 full load hours. All German WTs which were built before 2002 are qualified for repowering. Repowering means the replacement of older wind turbines with more modern multi-megawatt machines.

Offshore wind farms are located at sea. Modern offshore wind turbines produce around 5 MW. Because of increased average wind speed, the revenues are a lot higher than on the main land. Offshore wind farms can generate between 3,000 to 4,500 full load hours [5].

2 Maintenance of Wind Turbines

Wind turbines have a planned lifespan of 20 years. During this period, many main components have to be maintained or replaced.

The maintenance of offshore WTs is a lot more problematic than onshore WTs, because they can only be reached by ship and helicopter. Therefore, offshore maintenance causes costs which are six times higher than those on the mainland [1].

For ships, the wave height determines significantly the access of an offshore wind turbine. Usually, weather conditions with a wave height above 1.5 m are called “weather days”, because the WTs cannot be reached hazard-free. The annual number of “weather days” for different German offshore wind farms is shown in Figure 2 [5].

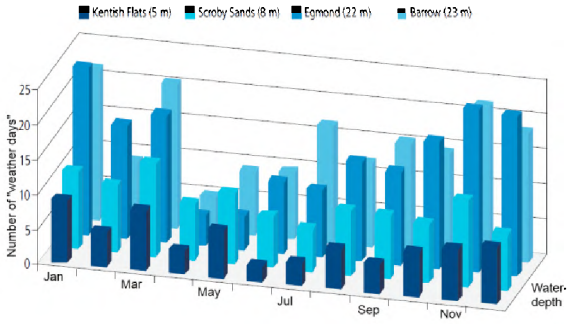


Figure 2: Accessibility of offshore wind farms [5]

In addition to the above-mentioned difficulties (restricted means of transportation and dependency on meteorological conditions), offshore maintenance also has a more complex supply chain. It is very important to ensure a reliable and cost-efficient parts supply. Since the topic of offshore wind farming is still very new, it has not been possible to standardize any maintenance concepts yet. The long-term reliability of wind turbines is still unknown. Hence, no spare part storage concept from other industries may be adopted [1].

3 Objective Target

The main objective target is to develop a wind farm maintenance platform based on new technologies and scientific approaches. The principal objective targets of this system can be separated into the following topics:

Proactive management system

The proactive management system intends to evaluate all relevant physical values, which are provided by the offshore wind park. Real-time monitoring and reporting should be based on a dataset containing 400 records per second per turbine. Particularly, the use of any averages for faster calculations as well as the reduction of storage space like in other systems has to be avoided. Furthermore, the system should provide an automated error detection and error classification unit.

Exact forecasts for maintenance periods

Focusing on the turbine maintenance, the target is to forecast lifetime estimation and breakdowns. Forecast reports and pre-alerting for all turbine components should be created automatically by the system. Based on different researches in this area, it is possible to develop algorithms which make the system able to generate these reports and alerting using weather data, resource data, operational data and maintenance history data.

On demand statistic functions for physical research

Resting on new database technologies, more complex analyses can be executed on a larger dataset. Finally,

the period of computation is shorter. Thus, faster responses are possible to improve the workflow in research, like developing algorithms or analyzing complex diagrams.

4 Reasons to Use Sap HANA for this Project

The main task of the proactive maintenance of offshore turbines is to calculate the average remaining life expectancy. As mentioned in chapter 3, the current analyses are performed on aggregated data, although more granular data could lead to more precise calculations. An advantage is that the 400 sensors of the wind turbine are already delivering data on a per second basis. In order to analyze this data set, which is increased by a factor of 600 compared to aggregated data on a 10 minute basis, SAP HANA has to be used.

By using SAP HANA, the creation of OLAP cubes can be omitted. Hence, more data can be analyzed and new analyses can be performed directly on the data. SAP HANA offers the possibility for the engineers to create new analyses, test them directly and subsequently continue their work with new findings.

SAP HANA and Business Intelligence Tools will be used to achieve the objective targets of the Smart Wind Farm Control project. The use of these information technologies for optimization and support of science researches makes this project unique.

5 New Insights

During the fourth quarter in 2012 and the first quarter in 2013 the project group started to develop the wind farm maintenance platform. First of all an architecture was created which illustrates all components of the platform as well as their relationships to each other, (see figure 3).

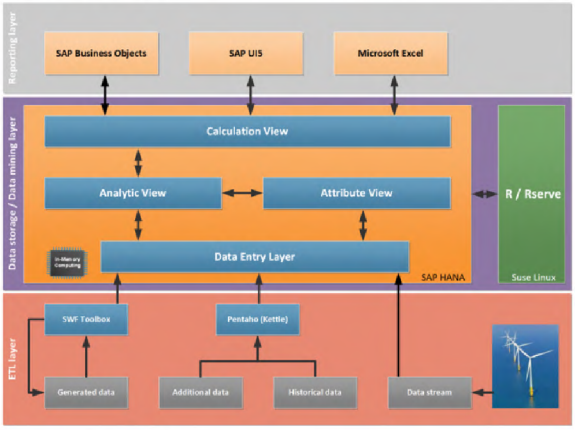


Figure 3: Architecture

The shown architecture is divided into three layers *ETL*, *Data storage and data mining* and *Reporting*. In the following three sections each layer and all performed activities will be outlined.

Extract, transform, load

The ETL (extract, transform, load) layer generally addresses data collection, data cleansing and transformation. Within the Smart Wind Farm Control project, the data is available in the form of historical and generated wind turbine data. In addition, data such as weather conditions or maintenance data can be collected. For a potential productive use, a continuous data stream of various wind turbines may be another data source. Being a native ETL software, Pentaho Data Integration CE (Kettle) is used to clean up the historical and supplementary data and transform them into the correct data model. Besides Kettle, a self-implemented SWF Toolbox is used to generate wind turbine data and to simulate possible data streams.

Data storage/Data mining

The data storage and data mining layer is divided into the components SAP HANA and R/Rserve. Within SAP HANA, a Data Entry Layer operates as an interface between the shown data sources and the database. The database model includes eight tables with various attributes, which are able to gather all kinds of upcoming data in the field of maintenance of offshore wind turbines. Furthermore, evaluation tools can access and output data using various views and SQL scripts in SAP HANA.

The project group has chosen R as a data mining tool. The SAP HANA Predictive Analytics Library would have met the requirements as well but was released too late during the project phase.

R was set up on a separate Suse Linux Server along with Rserve. The data mining results are send dynamically to an email address which can be changed inside a database table within SAP HANA.

Since real data was not provided until very late during the project phase, no actual data mining results could be found. But the project group was able to prove with easy examples that data mining would work together with SAP HANA.

Reporting

In the reporting layer, the data is presented to the end users in a graphical way. Microsoft Excel is used for fast and lightweight analysis and reporting in form of charts, tables and pivot tables. The result of the project group work is an Excel file with predefined SAP HANA connections and several reports.

Furthermore a web application based on SAP UI5 for analysis and reporting of wind turbines was developed. The web application is designed to provide an overview of the major functional areas – monitoring, log, reporting and data mining, (see figure 4).

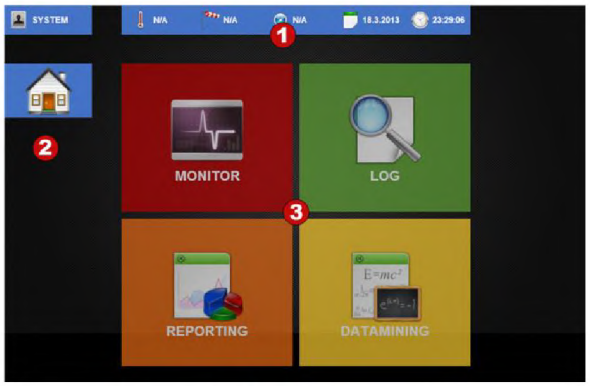


Figure 4: UI5-application

SAP Business Objects will be used in a later project stage in addition to SAP UI5.

Wind turbine data

Moreover the project group was able to constituted real wind turbine data from the project partners ForWind and Availon GmbH. So over 11 billion records per turbine from ForWind and about 150,000 records per turbine from Availon could be imported successfully into the SAP HANA database, (see figure 5).

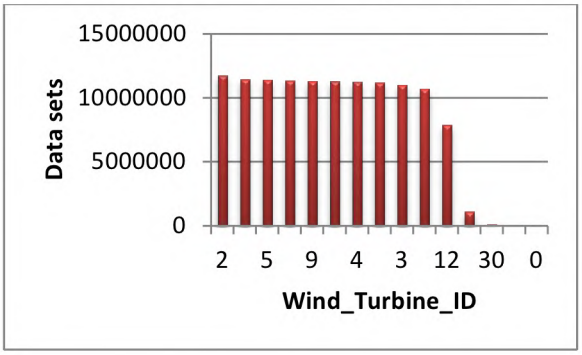


Figure 5: Data sets

The different amount of data is mainly due to the granularity at which the data was recorded. The ForWind data was measured on second basis in eight months, while the Availon data represents 10-minute aggregated values in twelve months.

6 Further Steps and Outlook

Looking forward a new team of students will take over this project and new or extending project goals will be defined. In addition more wind turbine data is expected by the Availon GmbH.

Possible tasks will be to gather more data mining results. As well as expanding the web application based on SAP UI5 and establishing SAP Business Objects as third reporting solution. The integration of more predefined reports and the development of the data mining function in SAP UI5 could be the main objectives.

Finally the current results could be used among other application fields, where the in-memory technology can create a benefit too.

References

[1] J. Westerholt: *Entwicklung eines Ersatzteilbevorratungskonzeptes für die Instandhaltung von Offshore Windenergieanlagen*. Diplomarbeit. Bremen, 2012.

[2] D. Böhme, W. Dürrschmidt, M. van Mark, F. Musiol, T. Nieder, T. Rüther, M. Walker, U. Zimmer, M. Memmler, S. Rother, S. Schneider, K. Merkel: *Erneuerbare Energien in Zahlen - Nationale und internationale Entwicklung*. Bundesministerium für Umwelt, Naturschutz und Reaktorsicherheit. 1. Auflage, Berlin, 2012.

[3] Bundesministerium für Wirtschaft und Technologie: *Energiekonzept für eine umweltschonende, zuverlässige und bezahlbare Energieversorgung*. Niestetal, 2010.

[4] J. Wilkes, J. Moccia, M. Dragan: Wind in power – 2011 European statistics. European wind energy association. 2012

[5] S. Pfaffel, V. Berkhout, S. Faulstich, P. Kühn, K. Linke, P. Lyding, R. Rothkegel: *Windenergie Report Deutschland 2011*. Fraunhofer Institut für Windenergie und Energiesystemtechnik. Kassel, 2012.

[6] SAP HANA http://www.sap.com/solutions/technology/in-memory-computing-platform/hana/overview/index.epx

Next Generation Operational Business Intelligence
exploring the example of the bake-off process

Alexander Gossmann
Research Group Information Systems University of Mannheim
agossman@mail.uni-mannheim.de

Abstract

Nowadays business decisions are driven by the need of having a holistic view on the value chain, throughout the strategic, tactical and operational level. Transferred to the retail domain, local store managers are focused on operational decision making, while top management requires a view on the business at a glance.
Both requirements rely on transactional data, whereas the analytic views on this data differ completely. Thus different data mining capabilities in the underlying software system are targeted, especially related to processing masses of transactional data.
The examined software system is a SAP HANA in-memory appliance, which satisfies the aforementioned divergent analytic capabilities, as will be shown in this work.

Introduction (Project Idea)

Operational Business Intelligence is becoming an increasingly important in the field of Business Intelligence, which traditionally was targeting primarily strategic and tactical decision making [1]. The main idea of this project is to show that reporting requirements of all organizational levels (operational and strategic) can be fulfilled by an agile, highly effective data layer, by processing directly operative data. The reason for such architecture is a dramatically decreased complexity in the domain of data warehousing, caused by the traditional ETL process [2]. This requires a powerful and flexible abstraction level of the data layer itself, as well as the appropriate processability of huge amounts of transactional data.
The SAP HANA appliance software is currently released in SPS 05. Important peripheral technologies have been integrated, such as the SAP UI5 Presentation Layer and the SAP Extended Application Services, a lightweight Application Layer. This project proves the tremendous possibilities offered by this architecture which allows a user centric development focus.
This report is organized in the following chapters. The first chapter provides a general overview of the explored use case. In the second chapter the used resources will be explained. The third and fourth chapters contain the current project status and the findings. This Document concludes with an outlook on the future work in the field.

1 Use Case

This project is observing a use case in the field of fast moving goods of a large discount food retail organization. Specifically, the so called bake-off environment is taken into account. Bake-off units reside in each store and are charged with pre-baked pastries based on the expected demand. The trade-off between product availability and loss hereby is extremely high.
From the management point of view, the following user group driven requirements exist: On the one hand, placing orders in the day to day business requires accurate and automated data processing, to increase the quality of the demand forecast. On the other hand, strategic decision makers need a flexible way to drill through the data on different aggregation levels, to achieve a fast reaction time to changing market conditions.
The observation period of two years is considered. The basic population consists of fine grained, minute wise data for thousands of bake-off units, providing all facts related to the bakery process.

1.1 Store Level Requirements

On the store level, the store manager will be supported with matters regarding daily operational demands. Primarily for order recommendations, a certain amount of historical data is taken into account to satisfy the appropriate statistical calculation on time series. Additionally location related and environmental information increases the accuracy of the forecasting model. Environmental variables, like historical weather and holidays, are considered in correlation with historical process data to improve the forecast model. Furthermore, forecasted weather data and upcoming holidays are taken into account for ex-ante data in order to improve the prediction. Model fitting and operational data analysis are being processed ad hoc and on demand by the appropriate store manager.

1.2 Corporate Level Requirements

On the corporate level a 'bird's eye view' is the starting point, where highly aggregated key figures indicate business success or problems. These measures

deliver information on a very high level, whereas the reasons for the appearance of these indicators can vary strongly. For accurate decisions it is tremendously important to drill down to the line level, to indicate the reasons for certain business patterns. As the strategic reporting is based on one common data foundation of operational data, navigation to the line level is implicated. It is important that the system is having user satisfying response times, allowing the exploration of a huge amount of data. The application provides the detection of certain patterns and correlations for a more complex classification. For example, the daily availability is analyzed based on certain thresholds, provided by minute wise real time data. To sum up, real-time enabled reporting on strategic level allows reactions on market changes to reach an unprecedented level of effectiveness.

2 Project set up

This chapter illustrates the used technology. After a listing of the architectural resources the appropriate implementation domains will be described in more detail.

2.1 Used Resources

As stated in the introduction, the used architecture is based on the SAP HANA Appliance Software SPS 05 [3].
The presentation layer is built upon the HTML5 based framework SAP UI5. The communication with the SAP HANA In-Memory database and user handling is established through SAP Extended Application Services (XS Engine). Data intensive calculations and data querying are handled by the appropriate APIs in the database, such as the calculation engine (CE), the SQL engine, the Application Function Library (AFL), and particularly the Predictive Analytics Library (PAL) [4].
For time series analysis the Rserve based R integration is used. The data load of CSV formatted transactional data, as well as the data replication and 3rd party are implemented in Java and imported through the JDBC API. The considered 3rd party data consists of weather data, as well as school and public holidays.

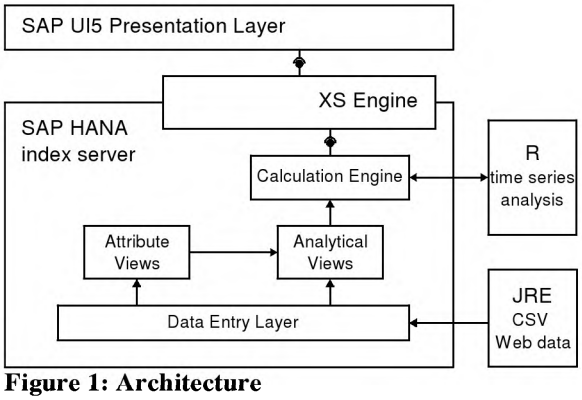


Figure 1: Architecture

The used architecture, as described in the following chapters, is summarized in Figure 1.

2.2 SAP Front End

As stated above the SAP UI5 constitutes the presentation layer. The Model View Controller pattern is being conducted for front end implementation. The SAP Extended Application Services serves as the controller layer. For web and mobile versions of the application two different view variants are implemented.
The entry point for a specific user group is the login screen, whereas the different management roles are distinguished by specific HANA user roles. The user groups are differentiated into the strategic, tactical, and operational management role. The strategic and tactical roles are showing the same reports, restricted by the related aggregation level. On the operational level, completely different reports are provided and are mainly focused on daily analysis. Additionally, order recommendations for the next three days are visualized. Each report relies on *one* associated calculation view, described later in the back end section. The selection parameter invoked by a user are handled by OData services, with the belonging data binding, or manually by SQL Script calls.

2.3 SAP HANA Back End

The HANA in-memory database is the core technology of this investigation. In the following section, the data model will be shortly discussed.
The data entry layer consists of two main fact tables. One fact table contains daily aggregated sales related key figures. The second fact table consists of minutely wise measures, derived from the bakery process. This fact table has an expected cardinality of approximately two billion records. In the current implementation this table has rounded 500 million records for the first testing runs. An appropriate partitioning policy, based on time related range partitioning is conducted here, in regards to the expected limit of 2 billion records per table. Several master data tables contain information about stores, regions, products, and holidays. Historical weather data is stored in an appropriate table, whereas weather forecasts will be stored separately and merged daily into the historical data table. All tables are implemented as column tables.
Upon this data entry layer several attribute views are implemented, building up the product, store and regional dimensions. The time dimension is based on the generated time table with minutely level of granularity (M_TIME_DIMENSION), provided by HANA standardly.
The two analytical views contain the fact tables, whereas the daily based fact table is additionally enhanced by the weather and holiday dimensions.

Based on this multidimensional data model eight calculation views are implemented, to satisfy user reporting scenarios about availability, loss, and sales on tactical and strategic level. Additionally one calculation view provides reporting needs on operational level, showing the relevant process information of the current and previous days.
For more sophisticated data mining on the strategic level, as well as data preprocessing of time series data, PAL is used [5]. Specifically the linear regression model function is used to draw trends of dynamically aggregated sales data over time. Further the anomaly detection function is used for outlier detection in daily sales data.

2.4 Peripheral technology

The load of historic and transactional data is handled by a proprietary Java import module, using the JDBC API. The reason for this implementation mainly relies on huge amount of heterogeneous CSV formatted files. Approximately two hundred thousand different types of CSV files have been imported into the HANA database. Therefore, a special bulk load strategy has been used, especially in spite of the insert properties of column oriented tables in the entry layer. Furthermore, historic weather data as well as weather forecast and holiday data is loaded via the JDBC interface of the import module.

Holidays

Both school and public holidays have been downloaded for the past two years, and until the year 2015 from the online portal 'Schulferien.org'. The data is available in the iCal format, and covers all dates for the different states of Germany. These files were loaded into the HANA index server, after conversion into CSV format, using the appropriate build in wizard.

Weather

The historical weather data has been imported from the web weather API 'wonderground.com'. For model training of the forecast module, the corresponding time interval values of daily, city wise consolidated store data was called from the API. This results in approximately one million JSON files (one file corresponds to one data record), generated by the REST interface, afterwards converted into CSV format and loaded via JDBC of the import module.

Forecast

The demand forecast requirements are primarily developed using the R environment. The appropriate time series are generated on demand, and invoked by a store manager who is responsible for one's store. As stated in the previous section, the time series are being preprocessed in advance by the PAL framework primarily for performance reasons.
The important outlier detection and handling have been additionally implemented in the R environment, as here more advanced algorithms are available in the R community. Furthermore, two different forecast models have been utilized for comparison reasons. The ARIMA (Auto Regressive Integrated Moving Average) model as well as the ANN (Artificial Neuronal Network) based model have been observed.

2.5 Development environment

The eclipse based HANA Studio is used as the main IDE for the development. In addition to the newly introduced SPS05 features, regarding the 'HANA development' perspective, the Java import module is implemented as well.
For usability reasons the following implementation strategy of the R environment has been utilized: each developer uses a local R runtime for coding R script and model testing. The appropriate time series data is supplied through the ODBC interface. After finalizing a model in R, it is transformed into the HANA environment using the RLANG extension in SQL Script [5].
All artifacts, including java classes, java script, UI5 artifacts and R script have been set under version control with git [6].

3 Findings

This chapter contains findings on technological as well as on the process level. The findings will be explained analogous to the outline of the previous chapter. In conclusion the outcome of this project will be summarized.

3.1 SAP Front End

Through the tight integration of the controller and model layer, the presentation layer profits of the advantage of a high abstraction level. The data binding feature of the OData services is especially beneficial for strategic and tactical reporting. Hereby flexible data navigation for the top management user is provided, by selecting free time intervals and breaking down into different products, regions, or stores. Never the less, the store management invokes an ad hoc data mining and forecasting capability by calling a SQL Script procedure through a Java Script DB connection call.
For parameterization of the calculation views the following limitations exist:

- exclusively input parameters are used, instead of variables for performance reasons
- for input parameters, no ranges are supported and graphical calculation views require additional filter expressions
- character based date parameters work with the OData interface (thus no type safety is provided, implicit cast)

The sap.viz library has been investigated deeply in context of data visualization. The main restriction has been experienced in the field of no supporting double scaled charts, which is a common and necessary

requirement. However, it is announced to be supported in the near future by SAP.

3.2 SAP HANA Back End

In the previous chapter (2.3) the data model has been explained. The biggest column based table contains two years of minutely based transactional data. It has been partitioned on a quarterly basis. The response times of the appropriate calculation view calls are absolutely satisfying. Nevertheless, the following main restrictions have been experienced which are listed by the appropriate domain:

Calculation Views

- graphical calculation views show best performance behavior, but limitations for complex key figure calculations
- for the usage of calculation engine (CE) calls in SQL script, only sequential data flow can be achieved and is therefore slower than graphical based calculation views
- CE syntax is more complicated and less expressive than SQL statements
- SQL statements in SQL script are many times slower than CE calls
- CE statements cover SQL functionality only partially and for complex calculations SQL statements could be indispensable, decreasing significantly the performance

Predictive Analytics Library

- usability of PAL functions is inconvenient and non-transparent
- restrictive parameterization policy
- very limited exception handling

The restriction in the design time usability especially in the case of PAL, compromises the performance experience of the data analysis.
The AFL framework is in a relatively early stage of maturity and in this project context, only few functions could be utilized. The major functionality in the area of time series analysis has been conducted in the R environment, as stated in the next section.

3.3 Forecast

The demand forecast for each store is calculated on demand. The appropriate time series is generated and sent, together with the belonging weather and holiday information to the R runtime. Hence the sent data frame to R contains daily related time series derivates of additional environmental data to the historic sales data for a certain pastry and store.

Time Series Preprocessing (Outlier Adjustment)

For comparison reasons the outlier detection is being performed with PAL and in a second scenario in R. In case of PAL preprocessing, the outliers are marked in the time series data frame provided to R. The used PAL function is PAL_ANOMALY_DETECTION with the following parameterization:

Table 1: PAL parameterization

Parameter	Value	Comment
THREAD_NUMBER	4	
GROUP_NUMBER	3	number of clusters k
OUTLIER_DEFINE	1	max distance to cluster center
INIT_TYPE	2	
DISTANCE_LEVEL	2	
MAX_ITERATION	100	

As depicted in Table 1, the used PAL function uses a k-means cluster algorithm, whereas GROUP_NUMBER corresponds to the number of associated clusters (k). Please note that this function detects always one tenth of the underlying number of lags in each time series as outliers. This could not be controlled by the parameter OUTLIER_PERCENTAGE, as expected and thus, limits this function enormously. In the R environment the k-means clustering for outlier detection is used as well. A straightforward approach of outlier handling is used. The majority of given outliers belongs to the class of additive outliers due to public holiday related store closing. The effect is even more significant, the longer a closing period is. Here the precedent open business date shows an abnormal high characteristic. Other outlier classes are by far less significant and cannot be assigned directly to events. Different outlier handling strategies have been tested and implemented, and will be investigated in further proceedings.

ARIMA based forecast

An automated ARIMA model has been implemented in R. The used package is mainly the package 'forecast' [7] available at CRAN (Comprehensive R Archive Network [8]). The automated ARIMA fitting algorithm 'auto.arima()'[9] has been utilized for this project purposes, which is based on the Hyndman et al algorithm [10]. Specifically seasonality, non-stationarity, and time series preprocessing (see outlier handling) required manually coded model adjustment. All additional predictor variables like holidays and weather information could be processed automatically, passed by the 'xreg' matrix parameter.

ANN based forecast

Alternatively to the ARIMA approach, an Artificial Neuronal Network model has been implemented and is especially for capturing automatically nonlinear time series shapes. As expected in the retail context, ANN is supposed to deliver more accurate forecast results [11]. In this use case the 'RSNNS' [12] (Stuttgarter Neural Nets Simulator [13]) package has been utilized. Similarly to the ARIMA model (see above), the independent variables, primarily the daily sales an all additional related variables are used for model fitting.

Summary Forecast

From the usability perspective, one typical forecast cycle takes about 3 seconds to show up the user the demand forecast for the next three days. Please note, that this reaction time includes the appropriate time series generation, the model fitting in the R environment, and finally the presentation of the results in the SAP UI5 front end. This is an acceptable response time, as it has to be done only once per day for one store. At this moment two years of daily training data is considered, longer time series would be preferable for better accuracy. This will require a linear growth in processing time.

3.4 Conclusion

The built prototype was expected to satisfy the reporting requirements of the different stakeholders of information consumption. Although the data mining capabilities differ throughout the organizational roles of managers, all human recipients expect short response times of a system. With the usage of the SAP HANA appliance software this challenging task could be achieved.

From the development perspective, previously known effectiveness could be achieved. As all reporting and predictive analytics requirements rely on only a few physical tables, the main effort consists in providing different views on this data. Even more complex measure calculations, like availability and some regression analysis, are processed on the fly. This is a completely new way of designing a reporting system. Compared to traditional ETL based data warehousing tools this saves a lot of manual effort in the loading process. However, this does not imply that the effort for implementing the business logic disappears, merely that the programming paradigm is straightforward. SAP constantly improves the appropriate API functionality (e.g. by introducing the 'HANA Development' perspective), whereas the programming framework is not matured yet.

The capability of providing demand forecasts based on long time series intervals for thousands of stores and different products particularly supports operational decision makers on the day to day business. This could not, or only very difficultly, be achieved with traditional disk based data warehouse approaches focused on aggregated measures. In this prototype, forecast algorithms are performed on demand. This makes sense, as the underlying models require readjustments with each new transaction. However, the time series growth will have a bad performance impact. For this reason, other model fitting strategies with serialization techniques shall be considered as well. A trade-off assessment of benefits of a longer time series versus forecast accuracy should be examined.

4 Outlook

The focus of the current work was set on the implementation of an analysis tool, processing masses of *historical* data. It is important to show in the next step, that this approach works with real-time enabled data provisioning. Specifically demand forecast has been valuated with historical data so far. In a productive scenario, weather forecast and upcoming holiday events will be considered in correlation with real-time data. Thus, especially the intraday recommendations and process monitoring will be focused for operational decision support.

For strategic decision makers, ad-hoc reports will be delivered upon real-time data. Additionally to the SAP UI 5 front end, realized in this project, front end tools from the SAP Business Objects portfolio will be evaluated. Especially the flexibility in real-time enabled data navigation will be focused in this scenario, known under the genus 'self-service business intelligence'.

References

[1] C. White: The Next Generation of Business Intelligence: *Operational BI. DM Review Magazine.* Sybase 2005

[2] H. Plattner: A common database approach for OLTP and OLAP using an in-memory column database. *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data.* ACM, 2009.

[3] SAP HANA Developer Guide. *help.sap.com/hana/hana_dev_en.pdf* , 21st of December 2012

[4] SAP HANA Predictive Analysis Library (PAL) Reference. *help.sap.com/hana/hana_dev_pal_en.pdf,* 23 of January 2013

[5] SAP HANA R Integration Guide. *help.sap.com/hana/hana_dev_r_emb_en.pdf,* 29th of November 2012

[6] *http://git-scm.com/*

[7] *http://cran.r-project.org/web/packages/forecast/forecast.pdf*

[8] *http://cran.r-project.org/*

[9] *http://otexts.com/fpp/8/7/*

[10] Hyndman, Rob J., and Yeasmin Khandakar. Automatic Time Series for Forecasting: The Forecast Package for R. No. 6/07. Monash University, Department of Econometrics and Business Statistics, 2007.

[11] Doganis, P., Alexandridis, A., Patrinos, P., & Sarimveis, H. (2006). Time series sales forecasting for short shelf-life food products based on artificial neural networks and evolutionary computing. *Journal of Food Engineering*, *75*(2), 196-204.

[12] *http://cran.r-project.org/web/packages/RSNNS/SNNS.pdf*

[13] *http://www.ra.cs.uni-tuebingen.de/SNNS/*

Realistic Tenant Traces for Enterprise DBaaS

Jan Schaffner [#1], Tim Januschowski [*2]

[#] *Hasso-Plattner-Institut*
University of Potsdam, Germany
[1] jan.schaffner@hpi.uni-potsdam.de
[*] *SAP Innovation Center*
Potsdam, Germany
[2] tim.januschowski@sap.com

***Abstract*—The benchmarking of databases is an involved topic to which cloud computing adds another level of complexity. For example, benchmarking of multi-tenant on-demand applications requires modeling tenants' dynamic behavior over time. This paper provides two methodologies for simulating realistic tenant behavior in enterprise on-demand scenarios. The methods cover the cases that no or not enough real-world tenant traces are available.**

I Introduction

Recently, Database-as-a-Service (DBaaS) has generated considerable attention in the literature, e.g. [1], [2], [3], [4]. Given the ever increasing demand for cloud services [5], this research area is likely to gain further momentum. However, traditional methodology from the database community, such as database performance benchmarking, e.g. [6], [7], [8], is only partially adequate for DBaaS. One explanation for the shortcoming of the existing methodology is the fact that DBaaS is a manifestation of a new database optimization problem [9]: the goal is no longer to execute a query as fast as possible or to execute as many queries as possible, but rather to execute a given number of queries while adhering to certain performance SLOs with as low cost as possible.

Work on general, multi-purpose benchmarks that is useful for DBaaS (and the new database optimization problem) only started recently [1]. Before that, researchers defined their own, problem-specific benchmarks, e.g. [3], [4], [10]. Common to and essential for all benchmarks in DBaaS is the incorporation of the dynamic behavior of tenants. How to generate realistic tenant behavior is, however, not considered in detail. The main goal of this paper is to help fill this gap. Our focus is on enterprise on-demand applications because these are highly regular in terms of user behavior. Also, load traces for these kinds of applications are typically difficult to obtain. Realistic tenant behavior is crucial for providing realistic answers to key questions in DBaaS such as cost-optimal or energy-efficient cluster operation and sizing.

The contributions of this paper are as follows. We present two methodologies for creating realistic tenant traces for enterprise on-demand applications, depending on the detail of information available. By tenant traces we mean relative request rates of tenants to a server cluster which we use to approximate tenant behavior. We assume all tenants to run the same (enterprise) application. In this paper, we provide

1) a generator for tenant traces that can be used if no tenant traces are available (Section III), and,
2) a bootstrapping methodology to enlarge a given set of traces (Section IV).

We use real-world tenant traces from one of SAP's productive on-demand services to develop our methodologies. Since we are not allowed to publish the traces, we describe their characteristics in detail. We also point out differences to private end-user cloud services (Section III).

To the best of our knowledge, realistic tenant behavior has not been described before in the literature in such detail, in particular not for the enterprise on-demand case. With a knowledge of realistic tenant behavior and sizes, tenant traces modeled following this paper can be used as input, for example, for MulTe [1]. The result would ultimately be a realistic benchmark for enterprise DBaaS which in turn is crucial for realistic experiments in DBaaS.

Before presenting our contributions in detail, we discuss related work.

II Related Work

Most relevant to this paper are generic multi-tenancy benchmarking frameworks because the tenant traces we describe here can directly be used as input to them. Such frameworks require modeling of single tenant behavior. Here, we describe both single and aggregated tenant behavior which suits benchmarks for multi-tenancy DBaaS particularly well. Aggregated workload modeling for clusters, e.g. [11], has attracted considerable attention, however, due to the aggregation such work is not directly useful for multi-tenancy benchmarks.

MulTe by Kiefer et al. [1] is a generic multi-tenancy benchmarking framework consisting of three steps: generation of tenants, execution of workloads, and, evaluation of the experiments. The tenant generation step relies on a set of parameters that the user of MulTe must specify. Once the tenants are generated, workloads based on well-accepted benchmarks such as TPC-H can be used to simulate the tenants' behavior. Kiefer et al. do not provide guidance on how to model tenants. Here, we provide such guidance by our analysis of enterprise on-demand tenant traces. Another generic benchmark is TPC-V [12], which is still under development.

Other works on DBaaS rely on the authors' custom-built multi-tenancy benchmarks which are similar in spirit to MulTe

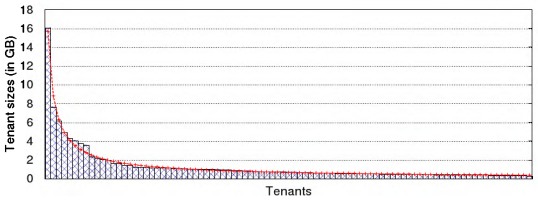


Fig. 1: Database sizes for the tenants in the trace

without being as general. For example, Yang et al. [4] use the TPC-W benchmark in their experiments for single-tenant behavior, but do not go into detail about the tenant modeling approach apart from the fact that they use a zipfian distribution to model database sizes. They do not further comment on the diurnal change in tenants' load. We provide details to both tenant behavior and tenant sizes. Curino et al. [3] use private end-consumer cloud services such as Second Life in their experiments. Here, we argue that such workloads differ considerably from enterprise workloads. Other experimental studies on DBaaS, e.g. [13], [14], [15], [2], use simplified tenant trace modeling. We argue in this paper that in enterprise on-demand applications complex, yet regular tenant behavior as well as different types of tenants can be observed. An incorporation of more realistic tenant behavior may lead to practically more relevant experimental results in particular with respect to enterprise on-demand scenarios.

III Tenant Traces: Observations and Generation

In the following, we describe real-world tenant traces, which provide good indications for tenants' behavioral changes, and tenant sizes from an enterprise on-demand service. We provide details for both aggregated and single tenant behavior. We point out differences to private end-user cloud usage for aggregate loads. Based on our observations, we provide a simple way to model tenant traces in order to generate realistic tenant traces.

We obtained tenant traces from a production multi-tenant, on-demand applications cluster, which uses SAP's in-memory column database [16]. These traces are the anonymized and (partially) normalized application server logs of 100 randomly selected tenants in Europe over a 4 months period. While the 100 tenants can be taken as representatives for the total set of tenants, the number of tenants is not great enough to allow proper benchmarking in multi-tenancy scenarios. Therefore, Section IV gives details on the generation of additional, realistic tenants.

A. Aggregated Tenant Traces

The left-hand side of Figure 2 shows a normalized view of the aggregate number of requests across all tenants over a five week period. For privacy reasons, we were only granted access to relative values normalized by the maximum request rate. We remark further that we only received traces of tenants that were already known to be active before the monitoring period. Sign-up rates could not be disclosed to us. We comment on this in more detail in Section V.

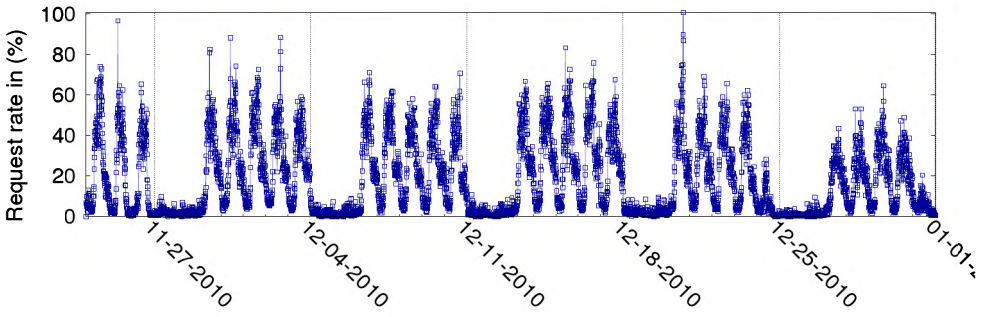
For the aggregated traces, one can clearly identify workdays and weekends. We observe that the application follows a strong 9–5 pattern: the number of active users per tenant and, correspondingly, request rates are elevated between 9:00a.m.–5:00p.m., with a drop around lunch time. We note several reasons for this pattern. First and foremost, enterprise on-demand services are used during these most common work times – and hardly at nights. Second, the server farm providing the on-demand service is geographically co-located with the users for several reasons: tight SLOs on response times demand a spatial proximity; and, enterprise on-demand users are conscious about data protection laws, i.e. they want their data to be held in a region where they know the legislation to be stable and protective towards enterprise data (often their own). The geographic co-location of servers and tenants means that no major shifts in the tenants' behaviors occur caused by different time zones which would smooth out the amplitude of the aggregated tenant request curve.

We contrast these tenant traces with the private end-user service Wikipedia [17]. We expect similar behavior for other private end-user service. However, one advantage of Wikipedia is free access to its cluster monitoring tools. Figure 2b shows an approximation of the request rate by considering the network throughput of Wikipedia's "Text Squid" cluster in Amsterdam which deals with article requests [18]. Patterns are similar for servers in other locations. We assume this network throughput rate to be proportional to the aggregated user request rate because users mainly use Wikipedia for reading articles. While absolute values are available, we chose to normalize the throughput rate to allow better comparison to the enterprise on-demand traces.

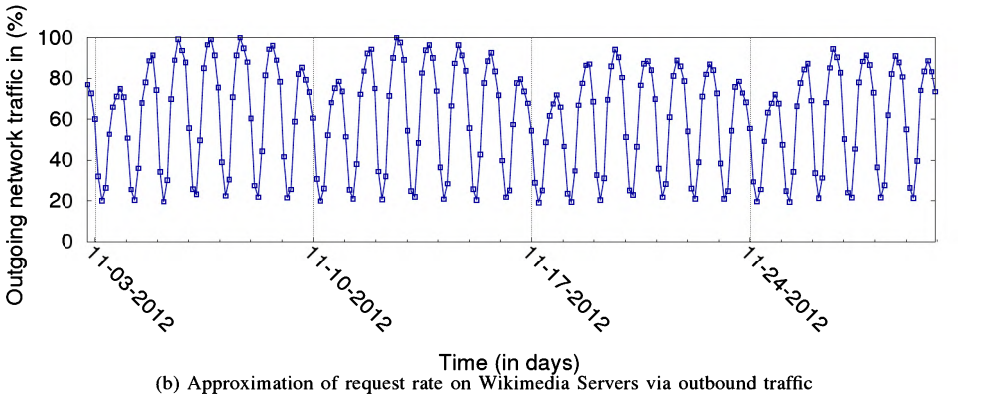
We note that an inactive/active pattern is visible in the server usage similar to Figure 2a. However, the amplitude of the pattern is much less pronounced and seems to follow the asleep/awake cycle. This may be due to several reasons. For example, Wikipedia has two server locations, one in Europe and one in the USA. The different time zones of the global community of Wikipedia users smooth out the amplitude. This could also partially explain the constant network load, even at night. It also seems plausible that users who are geographically co-located with the servers use Wikipedia both outside and inside of working hours, yet less so during day.

Another strong pattern in Figure 2a is weekends where very little tenant activity is visible. Saturdays are marked by dashed lines in Figure 2. This is in contrast to Figure 2b where the deviation between weekends and weekdays is much less pronounced.

Our analysis of the traces further reveals that holidays, such as Christmas, have the expected effect of lowered request rates for a limited period of time: the last week shown in Figure 2a is the week between Christmas and New Year's Day. Tenant activity in this week still conforms to the same periodic pattern as in the weeks before but exhibits a much lower amplitude. Also, as Christmas Eve in 2010 fell on a Friday, it comes as no surprise that the load on the second to last Friday of the surveyed period is considerably lower than the other Fridays.



(a) Five calendar weeks, including Christmas for SAP traces



(b) Approximation of request rate on Wikimedia Servers via outbound traffic

Fig. 2: Enterprise vs. private end-user aggregate, relative request rates

Again, these changes are much less pronounced (if visible at all) on private end-user cloud services.

Our tenant traces do not hint at the possibility of sudden load spikes as they occur in private end-user services, e.g. the load spike that Michael Jackson's death caused in Wikipedia [11]. One explanation may be the limited time for which we have tenant traces. However, we tend to think that enterprise on-demand may generally be more regular and less prone to outliers than private end-user service, simply because the number of end-users and the popularity of the services are less volatile.

B. Individual Tenant Traces

Next, we investigate single tenant traces in more detail. We note that most tenants exhibit a load pattern as shown in Figure 3a, which is similar to the aggregated request pattern in Figure 2a. However, there are also tenants in the trace which only use the system occasionally. Figure 3b shows an example.

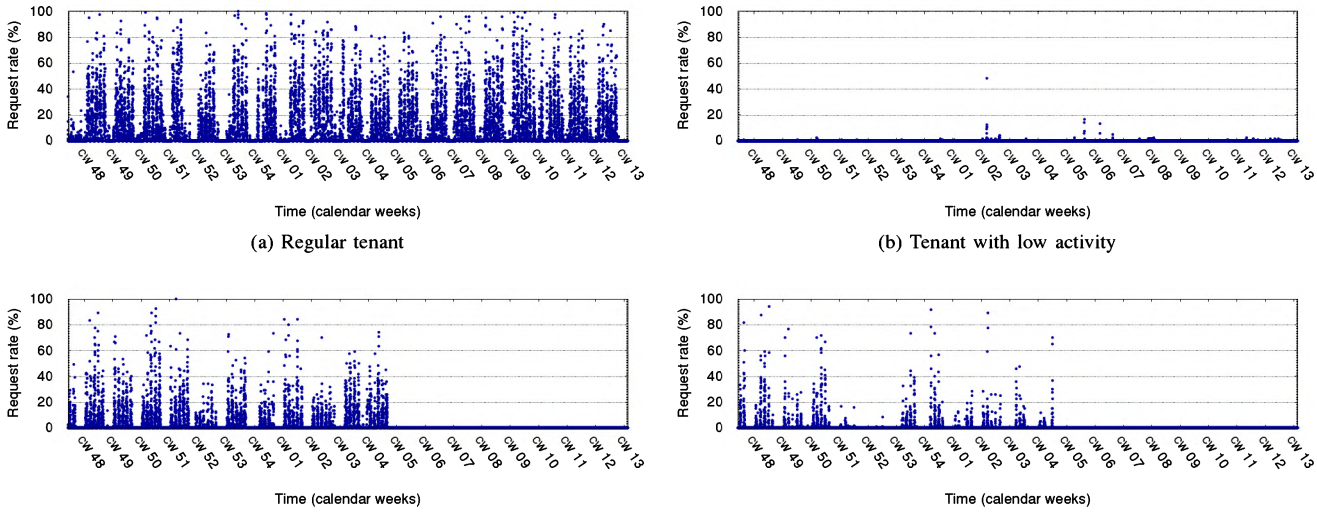
Investigating the log data revealed another interesting type of tenant: a non-negligible number of tenants appear suddenly, are active for 12 weeks, and then abruptly become inactive. Figure 3c depicts an exemplary trace. During this time these tenants behave similar to regular clients. Other short-lived tenants use the system actively for 2–3 weeks, then become inactive for a considerable amount of time (say, 2 weeks), and suddenly become active again for 6 weeks (Figure 3d). After investigating these tenants more carefully, we discovered that these were mainly demo and training systems. We believe demo systems to be another distinctive difference between enterprise on-demand and private on-demand services.

C. Tenant Sizes

In addition to the server logs, we also have access to the tenants' database size. The vertical bars in Figure 1 show their distribution. The tenant sizes follow a long-tail distribution rather than a self-similar distribution as is sometimes assumed [19]. The observed tenant distribution confirms Yang et al.'s [4] modeling of tenant traces via a zipfian distribution. We note that a few tenants are significantly larger than the rest, but the vast majority of tenants have approximately the same size. From additional statistics that we had access to, we note that the larger tenants also have significantly more active users and account for most load in the cluster.

D. Modeling Tenant Traces

In the following, we propose a model of the tenant traces which we base on the observation made above. Since our tenant traces show a strongly regular behavior both between weeks and days, it seems reasonable to focus on modeling a single working day and use this as the main building block for constructing an entire trace. We denote the request rate by $Q(t)$. For simplicity, we assume $t \in [1, \ldots, 144]$, i.e. each increment of t corresponds to a ten-minute interval which is the granularity on which we obtained the aggregated request rates. When modeling a single day d, we opted to fit a double



(a) Regular tenant

(b) Tenant with low activity

(c) Demo system used actively until end of trial

(d) Demo system used only at beginning and end of trial

Fig. 3: Different usage patterns for regular tenants and trial customers

Gaussian function. We use a Gaussian function due to the steep load increases/decreases for beginning and end of working day. The pronounced lunch-break drop for tenant request rates around noon motivates the use of a double Gaussian function as follows:

$$R_d(t) = a\exp(-\frac{(t-\mu_1)^2}{2\sigma_1^2}) + b\exp(-\frac{(t-\mu_2)^2}{2\sigma_2^2}) + c\,. \quad (1)$$

We parametrize (1) such that $m(R_d(t) - Q_d(t))$ is minimal. We denote the mean squared error by $m(\cdot)$. We opted for the simple model (1) because it fits the typical working day rhythm with a single lunch break. Future work could include other functions or more involved versions of (1), e.g. triple, asymmetric Gaussian functions [20].

We randomly chose a work day and fit (1) to it. Figure 4 shows the original trace (blue) and the least-squared fitted curve (green) with mean average absolute error (MAPE) of 43% and coefficients

$$a = 55.06,\ \mu_1 = 59.11,\ \sigma_1 = 8.90,$$
$$b = 75.24,\ \mu_2 = 88.84,\ \sigma_2 = 10.64,\ c = 11.90\,.$$

For modeling a working week w, we suggest shifting $R_d(t)$, i.e.

$$R_w(t) = \sum_{t\in[1,144\cdot5]} R_d(t \bmod 144)\,,$$

adjusting c appropriately. For weekends and public holidays indicator variables are needed to model the decrease in requests. In order to see how well (1) predicts tenant traces, we randomly chose 20 work days and calculated the MAPE between the aggregated tenant trace predicted by (1) and the actual aggregated load. The arithmetic average over the 20 MAPEs is 39% which clearly indicates how well (1) generalizes to other work days.

So far, we have modeled the aggregated tenant request rates. We obtain the request rate of a single tenant u by scaling $R_w(t)$, i.e. $r_{w,u}(t) = sR_w(t)$ by an appropriately chosen s. The distribution of tenant sizes provides a guideline on how to obtain such scale factors s by assuming requests to be proportional to database size. By adding random noise $e(t)$ to the working day pattern, we can create diversity among equally scaled tenant traces. We suggest using a normally distributed error, i.e.

$$e(t) = \frac{1}{\sigma_t\sqrt{2\pi}}\exp(-\frac{x^2}{2\sigma_t^2})\,. \quad (2)$$

such that $\sigma_t = 3$ for all t except $t \in \{\mu_i - 5, \mu_i + 5\}$ with $i \in \{1, 2\}$ where $\sigma_t = 10$ better captures the higher volatility in load. The orange line in Figure 4 shows an example of a tenant created by (1) and (2).

For demo tenants, we also rely on $R_d(t)$. Here, a much smaller scale factor is appropriate and a variable that indicates the limited time window of activity. Similar to regular tenants, adding random noise leads to the desired variety in tenants.

For modeling sizes of tenants, we already remarked that they follow a zipfian distribution. We chose to fit

$$s(u) = a + bu^{-c}\,. \quad (3)$$

We assume that tenants are ordered by size non-increasingly. The position $u \in [1, \ldots, 100]$ in the sequence of sizes corresponds to the tenant. When we fitted (3), we obtained a MAPE of 7% for the following parameters

$$a = 0.015, b = 15.65, c = 0.83\,. \quad (4)$$

The red line in Figure 1 shows (3) with parameters (4).

Having discussed how to generate tenant traces synthetically based on our observations, we consider the case that one has access to a limited number of tenant traces and needs to generate more tenant traces.

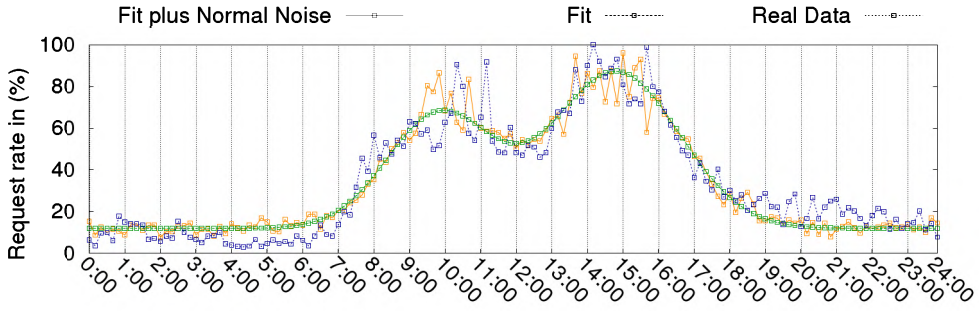


Fig. 4: Aggregated vs Generated Request Rates

IV Bootstrapping

As mentioned in Section I, it is difficult to obtain real-world tenant traces. While the previous chapter dealt with modeling tenant traces, we consider enlarging a set of given tenant traces in the following. We provide a straight-forward bootstrapping methodology with a single parameter, the *bootstrap window size*. For our use case, we show how to identify an appropriate value for this parameter.

In the server logs we obtained, a tenant trace comprises the request rates of a tenant across a 4-month-period aggregated in ten-minute intervals. We refer to a point in discretized time, such as the ten-minute interval in our case, as a *tick* in the tenant trace.

We were only granted access to a subset of the traces, 100 tenants in total. However, this random sample gives a realistic impression of the overall customer base. Unfortunately, the sample is not large enough to conduct meaningful experiments with realistically-sized server farms. For example, in October 2007, the SaaS CRM vendor RightNow had 3000 tenants distributed across 400 MySQL database instances [21]. To simulate a realistically sized DBaaS scenario, it is therefore necessary to increase the pool of tenant traces. Here, we suggest bootstrapping in order to enlarge the set of tenant traces such that

- the trace of a bootstrapped tenant shows the temporal characteristics of a working week,
- during a loaded workday, an active server holds 1–100 tenants,
- each tenant has a realistic size and trace volume, and,
- bootstrapped tenants resemble the sample tenants while being sufficiently distinct from them and each other.

In our methodology we follow [22]. Our methodology can naturally be combined with the generation of tenant traces as shown in the previous section. This may be interesting for example in the case where certain work day patterns are not present in the traces.

A. Bootstrapping Process

Our bootstrapping process produces for each original tenant trace, which we call a *parent* trace, a fixed number c of *child* traces. Figure 5 shows a schematic overview of the bootstrapping process. Apart from deciding on the trace pattern, we also need to assign a size to the child tenant. Based on our observations in Section III, we differentiate between regular tenants and demo tenants. Regular tenants all have comparably similar request rate patterns but highly varying sizes. Demo tenants have irregular usage patterns but exhibit smaller difference in their sizes. Therefore, it seems reasonable to choose size independently of trace pattern for regular tenants.

In more detail, our bootstrapping works as follows. From each parent tenant trace, we create c child tenants. The number of bootstrap copies c is a parameter of the process which we can choose. Each child tenant is then given a size drawn at random from a list of all parent tenant sizes (depending on the parent tenant being a demo or a regular tenant).

The process of bootstrapping a parent trace into a single child trace, shown in Figure 5, is as follows:

1) The parent trace is divided into equally sized subintervals of size W with the last subinterval of the trace potentially having size less than W. We call a subinterval a *window* in the following.
2) The indices of a window of the parent trace correspond directly to the indices of a window in the child trace. Window i spans $\{iW, iW+1, \ldots, iW+W-1\}$ in both parent and child traces.
3) For each parent window, W values are chosen at random with replacement and uniform probability. These values are placed into the corresponding child window in the order that they are chosen.

The quality of the bootstrapping processing depends on the choice of W. Section IV-B provides details.

The bootstrapping process results in c times many new tenants, each with a realistic size and a trace that is similar but not identical to its parent. For 100 parent tenant traces, we have $100 \cdot c$ many tenants. By drawing the child tenant size at random from all parent tenant sizes, we doubly ensure that the child is not too similar to its parent as size and request rate influence the load that a tenant puts onto a DBaaS cluster [10].

B. Choosing the bootstrap window size

In the following, we describe how to obtain a good value for W which determines the similarity between child and parent tenant. For a choice of a small W, we run the risk of all or most

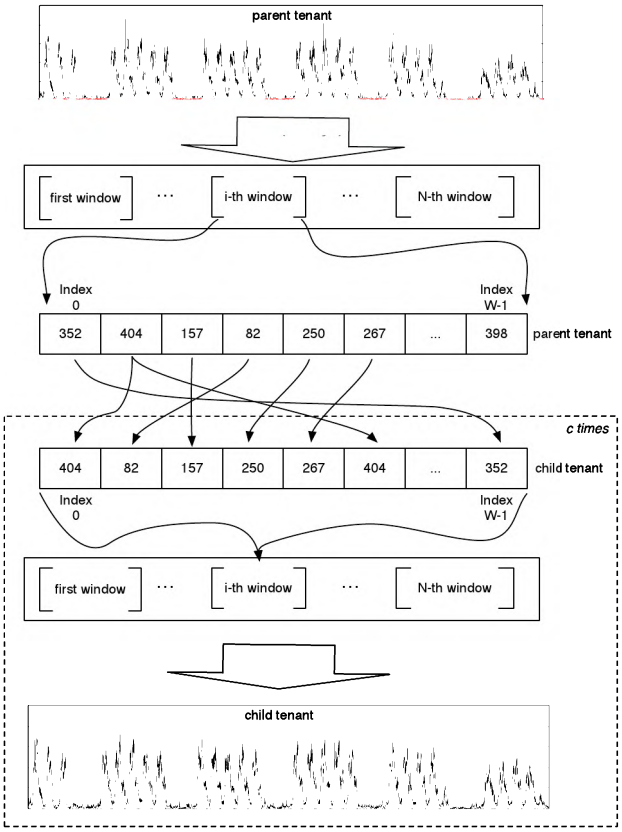


Fig. 5: The bootstrapping process for generating realistic tenant load traces.

c child traces of a single parent tenant trending in unison, as a load spike in the parent may occur in its children around the same ticks. Conversely, a large W causes irregularity and lack of natural smoothness in the children: for example, a window may fall on the boundary between night and workday, and the low-request at a tick during the night and high-request ticks during the workday become intermingled, giving the trace *unnatural spikes*. The goal is thus to set W such that the deviation between parent and children is high enough, while no unnatural spikes occur.

In the following we analyze both phenomena. While the methodology is generally applicable, we use our traces from the previous section as an example. For ease of exposition, we assume $c = 5$.

Deviation between Parent and Child. Figure 6a shows how the similarity between parent and child traces decreases as the size of the bootstrap window increases. Specifically, it shows the mean deviation in load across all ticks between the parent and its bootstrapped children, computed using the Root Means Squared Deviation (RMSD) for each parent/child pair. The figure depicts one of the largest tenants in our trace. Note that the increase of the RMSD flattens out beyond $W = 10$. Recall that the window size (depicted on the horizontal axis in Figure 6a) corresponds to numbers of ten-minute intervals for which request rates where aggregated in the traces.

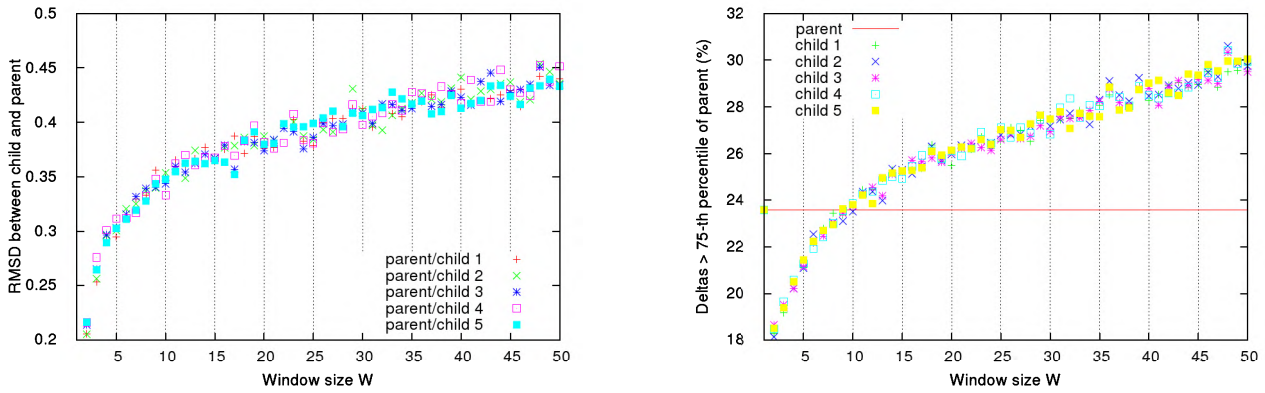
Controlling Unnatural Spikes. Figure 6b characterizes unnatural spikes in the bootstrapped children. When looking at the amount of change that a parent exhibits in request rates between two adjacent ticks, we define this change to be unnatural when it is larger than the 75-th percentile of all changes in request rates of the parent. Figure 6b, again focusing on a single tenant, shows the percentage of request changes larger than the 75-th percentile for the parent and its children with varying W. For the parent, obviously, this percentage is independent of the window size. For the children, the percentage of unnaturally high changes in requests is the same as for the parent when choosing $W = 1$. For $2 \leq W \leq 10$ it is lower than for the parent. This is because with such small window sizes there is always a relatively high chance for picking a value that has already been picked previously in the same window as the next value. For larger values of W, the children have higher changes in request between adjacent ticks than the parent, i.e. the children become more unnatural. This is because for large window sizes the chance increases that a vastly different value from the value picked for the current ten-minute interval is contained in the window and is picked as the value for the next tick at some point.

Based on the deviation and spike criteria, we suggest a bootstrapping window size of 10 as an appropriate value for W for our traces. This choice balances deviation of a parent and its children with unnatural changes between adjacent ticks.

V Conclusion and Future Work

The benchmarking of multi-tenant Database-as-a-Service scenarios is an involved topic on which research only recently started. Often tenant behavior is modeled after private end-user behavior or generated randomly. Here, we have made the point that there are significant differences between private end-user and enterprise on-demand services. As enterprise on-demand providers are more protective with respect to their customer base than public services such as Wikipedia, enterprise tenant behavior is more difficult to obtain. Restrictions and normalizations protect the interest of the involved parties rigorously. In this paper, we have shown ways to overcome restrictions which may occur due to protection of privacy and interests.

For usage in benchmarks for testing DBaaS, we note the following on our tenant traces. Our tenant traces are based on requests from real customers from small to mid-sizes enterprises. Although the exact nature of their requests is unknown to us, it is likely to be a so-called mixed workload. Therefore, one could approximate these real-world customer requests via a standard benchmark in a multi-tenancy benchmark which naturally introduces some inaccuracy. However, this is where the modularity of MulTe [1] would provide a benefit: one can simply experiment with different available benchmarks to approximate a range of enterprise on-demand scenarios. Second, companies often do not publicly disclose their exact server specifications or the degree of multi-tenancy. Another approximation is necessary here, for example via the little



(a) RMSD between parent and child increases with window size: the child is less similar to the parent.

(b) Difference in requests between adjacent ticks for children, compared to the 75th percentile of the parent.

Fig. 6: Choosing an appropriate value for the window size in the bootstrapping process denoted by W

information that is publicly available on multi-tenancy for enterprise on-demand [21].

One use of multi-tenancy benchmarks is to obtain a load metric that measures how much load the respective tenant puts onto a single server (also taking other tenants on the same server into account) [3], [15], [10], [13]. Since the degree of multi-tenancy cannot be disclosed publicly, an approximation is necessary. For example, the afore-mentioned SaaS provider RightNow maintains from 1 to 100 tenants in each MySQL database instance in their cloud service.

Our tenant traces only contain traces of tenants who were known to be active at the start of the recording period. Therefore, we cannot provide information on sign-up rates as well as tenant ramp-up behavior. While sign-up rates can be estimated from general market studies, ramp-up behavior would need to be modeled carefully and requires a more detailed study.

Acknowledgments

The authors further thank Megan Kercher, Tim Kraska, Malte Mues and Jasper Schulz for discussions on the topic and Camila J. Mazzoni and Gennadi Rabbinovic for careful reading.

References

[1] T. Kiefer et al., "MulTe: A Multi-Tenancy Database Benchmark Framework," in *TPC Technology Conference (TPCTC)*, 2012, pp. 11–18.

[2] S. Aulbach et al., "A comparison of flexible schemas for software as a service," in *SIGMOD Conference*, 2009, pp. 881–888.

[3] Carlo Curino et al., "Workload-aware database monitoring and consolidation," in *SIGMOD Conference*. ACM, 2011, pp. 313–324.

[4] Fan Yang et al., "A Scalable Data Platform for a Large Number of Small Applications," in *CIDR*. www.cidrdb.org, 2009.

[5] Gartner Research, "Gartner Says Worldwide Cloud Services Market to Surpass $109 Billion in 2012," http://www.gartner.com/it/page.jsp?id=2163616, September 2012.

[6] "TPC," http://www.tpc.org/.

[7] P. E. O'Neil et al., "The star schema benchmark and augmented fact table indexing." in *TPCTC*, 2009, pp. 237–252.

[8] A. Bog et al., "A mixed transaction processing and operational reporting benchmark," *Information Systems Frontiers*, vol. 13, no. 3, pp. 321–335, 2011.

[9] D. Florescu and D. Kossmann, "Rethinking cost and performance of database systems," *SIGMOD Record*, vol. 38, no. 1, pp. 43–48, 2009.

[10] J. Schaffner et al., "Predicting in-memory database performance for automating cluster management tasks," in *ICDE*, 2011, pp. 1264–1275.

[11] P. Bodík et al., "Characterizing, modeling, and generating workload spikes for stateful services," in *In SoCC'10: Proceedings of the 1st ACM symposium on Cloud computing*, 2010, pp. 241–252.

[12] P. Sethuraman and H. R. Taheri, "TPC-V: a benchmark for evaluating the performance of database applications in virtual environments," in *TPCTC*, 2010, pp. 121–135.

[13] Jennie Duggan et al., "Performance prediction for concurrent database workloads," in *SIGMOD Conference*. ACM, 2011, pp. 337–348.

[14] J. M. Milán-Franco et al., "Adaptive Middleware for Data Replication," in *Middleware*. Springer, 2004, pp. 175–194.

[15] W. Lang et al., "Towards Multi-tenant Performance SLOs," in *ICDE*. IEEE Computer Society, 2012, pp. 702–713.

[16] F. Färber et al., "SAP HANA database: data management for modern business applications," *SIGMOD Record*, vol. 40, no. 4, pp. 45–51, 2011.

[17] "Wikipedia," http://www.wikipedia.org/.

[18] "Ganglia:: Wikimedia Grid Report," https://meta.wikimedia.org/wiki/Wikimedia_servers.

[19] J. Gray et al., "Quickly Generating Billion-Record Synthetic Databases," in *SIGMOD Conference*. ACM Press, 1994, pp. 243–252.

[20] T. Kato et al., "Asymmetric gaussian and its application to pattern recognition," in *Structural, Syntactic, and Statistical Pattern Recognition*, 2002, pp. 405–413.

[21] M. Myer, "RightNow Architecture," in *HPTS*, 2007.

[22] Peter Bodík et al., "Characterizing, modeling, and generating workload spikes for stateful services," in *SoCC*. ACM, 2010, pp. 241–252.

Quasi-Real Time Individual Customer Based Forecasting of Energy Load Demand Using In Memory Computing – Project Report –

Witold Abramowicz Monika Kaczmarek Tomasz Rudny
Wioletta Sokolowska
Department of Information Systems
Faculty of Informatics and Electronic Economy
Poznan University of Economics
Al. Niepodleglosci 10, 61-875 Poznan
Poland
firstname.lastname@kie.ue.poznan.pl

Abstract

This report presents the individual customer based approach to energy demand forecasting using the computational power of SAP HANA. The research hypothesis was that demand forecasting can be done in quasi-real time, even if conforming to bottom-up approach, i.e., computing separate forecasts for each customer. The report provides information on the project idea, used HPI Future SOC Lab resources, findings as well as next steps envisioned.

1 Introduction

The European energy markets are facing many challenges – e.g., usage of renewable energy sources, application of smart metering and, consequently, the emergence of new players in the market (e.g., of prosumers). However, one of the biggest challenges the energy sector still needs to address, is how to accurately predict a short- and long-term value of energy demand, as well as the level of energy production from different sources. Erroneous forecasts entail costs, resulting, among others, from the need to purchase the additional energy capacity at the energy balancing market for a significantly higher price. In turn, the excess capacity in the system entails incurring fixed costs, resulting from the maintenance of overcapacity. In our project, we focused on a short term prognosis of energy demand. Taking into account a large number of influence factors and their uncertainty, "it is not possible to design an exact physical model for the energy demand" [4]. Therefore, usually both the long term and short term energy demand is calculated using statistical models (e.g., regression models, probabilistic models [7, 6, 3]), artificial intelligence tools (e.g., [8]) or hybrid approaches (e.g., [2, 5]). They all try to describe the influence of climate factors, operating conditions [4] and other variables, on the energy consumption. The quality of the demand forecast methods depends first of all, on the availability of historical consumption data, as well as on the knowledge on the important influence variables and their values. The second factor is the application of an appropriate forecast tool and method taking into account the data that we have, as well as existing limitations (e.g., computational).

Most of the existing approaches to energy demand prognosis focuses on characterising aggregated electricity system demand load profile. However, as shown by many researchers and practitioners, the energy load forecasting can be done more accurately, if forecasts are calculated for all customers separately and then combined via bottom-up strategy to produce the total load forecast [1, 3]. The reason for it is that the characteristics of aggregated and individual profiles are different (e.g., they have different shapes - the individual profiles show peaks in the morning and evenings and their shapes vary for different days of the week and part of the year). In fact, the individual electricity load profile is influenced by the number of factors, which may be divided in the following categories [3]: electricity demand variations between customers, seasonal electricity demand effects, intra-daily variations, and diurnal variations in electricity demand.

Also, additional improvement is expected, if different models and parameters can be tested quickly to provide the best model selection capability. Quasi real time load forecasting could lead to substantial savings for companies and also for the environment, up till now however, it was computationally difficult, as for a typical energy seller a number of residential customers exceeds hundreds of thousands.

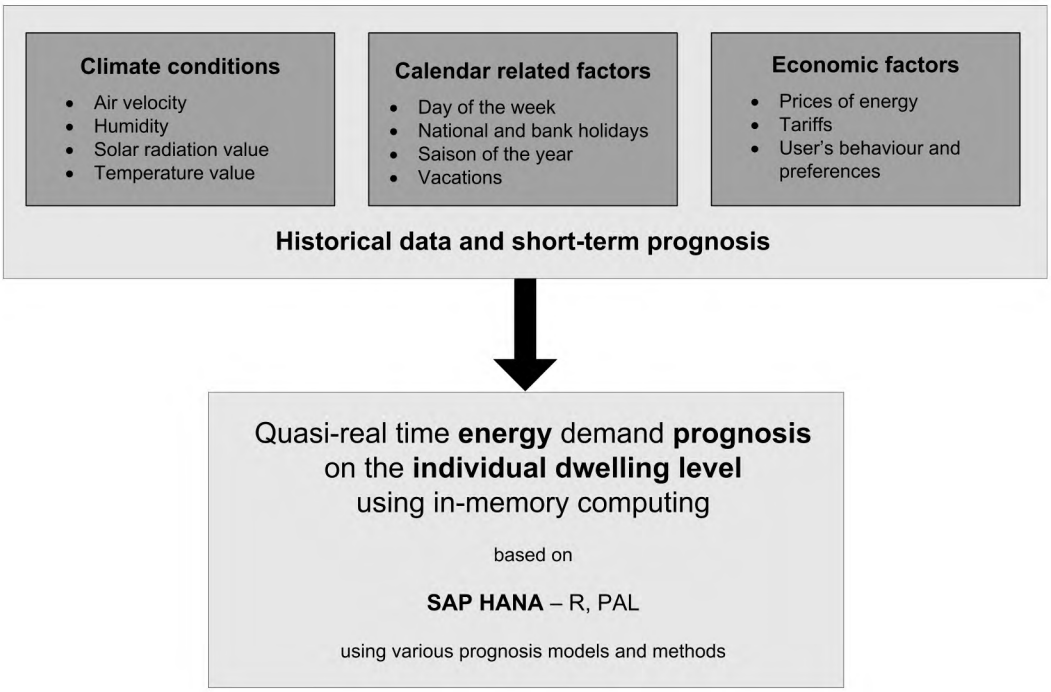


Figure 1: Project scenario

Within the conducted project, we focus on time series approach in order to predict the short-term energy demand value. We apply the time series method at an individual dwelling level (similarly as [3]), as is described in the next section of this report.

2 Project Aims

The main goal of the project was to design and implement quasi-real time load forecasting for a big number of individual customers profiles using in-memory computing. We formulated a hypothesis: forecasting accuracy (which is directly related to costs) can be substantially increased by individual forecasting on the level of each customer done in quasi real-time using SAP HANA parallelism, and a new insight into the analysed data can be gained (see 1).
The additional goal of the project was also to evaluate efficiency and performance of other computational strategies. Fast computations allow for comparisons of different forecasting models and choosing the best one on a spot. Also, the possibility for adding different exogenous variables to the models and being able to quickly analyse their significance opens up new applications of energy load forecasting.

3 Future SOC Lab Resources Used

During the project we used SAP HANA Studio and the assigned instance (12) combined with Rserve for predictive analyses. SAP HANA offers a way to incorporate the R code directly into the SQL Script. Table variables can be passed on as input parameters, while output parameters can receive values of R data frames. This makes it possible to conveniently use R packages and procedures directly from SAP HANA.
Throughout the project we uploaded the data into SAP HANA as column tables. Then, we rearranged the data for our experiments, creating auxiliary tables and columns. Because the data we obtained from a major Polish energy distributor consisted of only a limited number of times series, we decided to generate artificial data as randomization of the existing one. The randomization modified the values of the energy load demand by 10%. It was performed using the R script procedure.

```
CREATE PROCEDURE RANDOMIZE_DATA(IN x
"ENERGY_DEMAND_ALL", OUT y
"ENERGY_DEMAND_ALL_2")
LANGUAGE RLANG AS
BEGIN
dup <- data.frame(x)
dup2 <- data.frame(x)

for (i in 1:10000) {
dup$value <-
runif(dim(x)[1], 0.9 * x$value,
1.1 * x$value)
dup2 <- rbind(dup2, dup)
}

y <- as.data.frame(dup)

END;
```

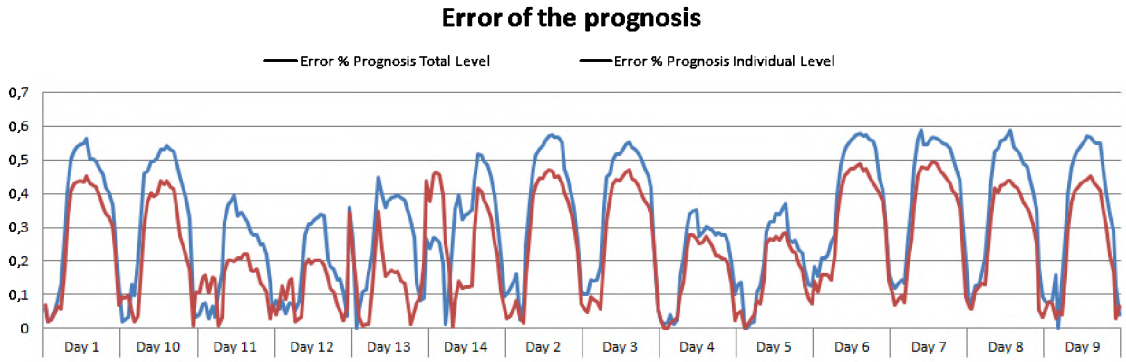


Figure 2: Prognosis error comparison - individual level and total level

Apart from the standard installation of SAP HANA combined with R service we used the new efficient Predictive Analysis Library (PAL), however, we did not yet run the same experiments on PAL.
The following experiments were run on SAP HANA using the aforementioned resources:

- Computing the summary forecast by summing all individual time series and calculating the (single) forecast over the summarized data,
- Calculating individual forecasts for all customers, then summarizing the forecasted values to compute the summary forecast,
- Comparing the forecasting error between different scenarios,
- Comparing different forecasting models - among others Holt-Winters exponential smoothing and ARIMA.

The experiments conducted so far focused on evaluating SAP HANA capabilities for time series forecasting, data manipulation and storage, as well as organizing the code.
We also had to impute the data as sometimes the value of the historical energy load demand was missing. The imputation code is presented below:

```
CREATE PROCEDURE IMPUTE_MISSING(IN x
"totals",OUT y "totals_imputed")
LANGUAGE RLANG AS BEGIN
missing <- is.na(x$value)
n.missing <- sum(missing)
x.obs <- x$value[!missing]
imputed <- x
imputed$value[missing] <-
sample(x.obs, n.missing, replace=TRUE)
y <- as.data.frame(imputed)
END;
```

4 Findings

As stated in the project proposal our aim is to evaluate SAP HANA capabilities in the area of fast massive time series computations. To this end we implemented a set of R procedures. The performed experiments proved that SAP HANA is an efficient tool due to its in-memory paradigm as well as parallel processing power. We were able to generate forecasts using various models based on a 180 days historical data with a 14 days horizon. The models tested included ARIMA and Holt-Winters exponential smoothing. The results show that SAP HANA can be efficiently used for this purpose.
The runtime for calculating summary forecast using the bottom-up approach varies around 600 ms as shown on the histogram (see fig. 2). It is not only possible calculate summary forecast in quasi-real time, but also to run different models and choose the best fitted one for each customer.
The code for forecasting can be nicely and neatly written in R as shown on the example of Holt-Winters exponential smoothing:

```
CREATE PROCEDURE FS_STEP(IN x "input2",
IN temp "temp", OUT y "totals_forecast")
LANGUAGE RLANG AS
BEGIN
ts1 <- ts(x$value, frequency=24)
m <- HoltWinters(ts1)
f <- predict(m, n.ahead=14*24)

forecast <- f[1:14*24]
```

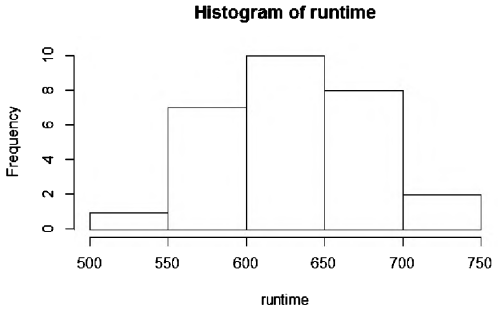


Figure 3: Runtime histogram of simple forecasting via bottom-up aproach

```
y <- as.data.frame(cbind(temp, forecast))
END;
```

Our proposed approach needs further works and polishing, but the initial hypothesis has been partially proven - SAP HANA is an efficient tool for energy market and time series forecasting (see fig. 3). However, the learning curve proved to be slightly steep, as mastering different specific details of SAP HANA scripting may not be easy.
Some of the obstacles we encountered included:

- problems with column store order of tables when manipulating data within SQL script procedures,
- problems with passing out data frames from R back to SAP HANA,
- problems with loop variables in SQL script.

We are convinced that these early problems can be overcome and this does not in any way diminish the value of SAP HANA. However, it was the lesson we had to learn.

5 Next Steps

The already conducted research using the Future SOC Lab resources allowed us to gain some insight into SAP HANA and the prognosis challenge and proved the benefits from the application of the in-memory computing offered by HANA. However, 5 months period turned out to be not enough time to test all our hypotheses and conduct all of the planned experiments. What we would like to do as next steps includes:

- building regression models to identify impact of exogenous variables,
- implement an automatic procedures to run all the tests and experiments,
- simulate special events as it is known that they should worsen to top-down forecast, while not affecting bottom-up approach,
- write selected algorithms in C langauge and launch them on the available NVidia box.

In addition, we started a collaboration with the local energy provider regarding the forecasting of the energy production from renewable sources. As we were able to obtain relevant data on energy generation, in addition, we would like to conduct research and run experiments to prove that SAP HANA and in-memory computing can be applied in this area as well.

References

[1] W. Charytoniuk, M.-S. S. Chen, P. Kotas, and P. Van Olinda. Demand forecasting in power distribution systems using nonparametric probability density estimation. *Power Systems, IEEE Transactions*, 14/4:1200–1206, 1999.

[2] M. S. Kiran, E. Ozceylan, M. Gunduz, and T. Paksoy. A novel hybrid approach based on particle swarm optimization and ant colony algorithm to forecast energy demand of turkey. *Energy Conversion and Management*, 53(1):75 – 83, 2012.

[3] F. McLoughlin, A. Duffy, and M. Conlon. Evaluation of time series techniques to characterise domestic electricity demand. *Energy*, 50(0):120 – 130, 2013.

[4] W. Schellong. *Energy Management Systems*, chapter Energy Demand Analysis and Forecast, pages 101–120. InTech, 2011.

[5] R. Shakouri, G.H.and Nadimi and F. Ghaderi. A hybrid tsk-fr model to study short-term variations of the electricity demand versus the temperature changes. *Expert Systems with Applications*, 36(2, Part 1):1765 – 1772, 2009.

[6] C. Wang, G. Grozev, and S. Seo. Decomposition and statistical analysis for regional electricity demand forecasting. *Energy*, 41(1):313 – 325, 2012. ¡ce:title¿23rd International Conference on Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy Systems, ECOS 2010¡/ce:title¿.

[7] J. Wang, X. Ma, J. Wu, and Y. Dong. Optimization models based on gm (1) and seasonal fluctuation for electricity demand forecasting. *International Journal of Electrical Power & Energy Systems*, 43(1):109 – 117, 2012.

[8] M. Zadeh, S. and A. Masoumi. Modeling residential electricity demand using neural network and econometrics approaches. In *International Conference on Computers and Industrial Engineering (CIE), 2010 40th*, pages 1 –6, july 2010.

A framework for comparing the performance of in-memory and traditional disk-based databases

Vassilena Banova
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
vassilena.banova@in.tum.de

Alexandru Danciu
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
danciu@in.tum.de

Helmut Krcmar
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
krcmar@in.tum.de

Abstract

Enterprises are facing an increasing amount of data, and rely more than ever on flexible and fast methods of data analysis. A proposed solution is the switch to in-memory database systems which promise huge performance increases. Unfortunately recent literature does not provide performance comparisons of main-memory and classical storage based database systems in enterprise use cases. A literature review is conducted identifying the scenario of discovering frequent item-sets as an appropriate use case to perform such a comparison. Requirements on a test environment are provided and a tool supporting those tests is implemented, encouraging the execution of OLAP, OLTP and mixed workload test-suites.

1 Introduction

Today's enterprises store and collect more data than ever. It is estimated that the fortune 500 enterprises store approximately seven to ten years of customer data [1]. This amount is told to increase even further in the coming years [2]. Amongst others this data is used in the context of Business Intelligence & Analytics"(BI&A) to gain insights on the market and make critical decisions in a timely manner based on up to date information [3]. But the huge amount of data makes it necessary to narrow the data down or to work with pre-calculated results what limits the flexibility and expressiveness of the analysis [4]. In an environment of fast changing market conditions and customer wishes this flexibility is crucial to recognize influences on the own business and react on undesirable developments [2].

A frequently used term in this context is "Big Data" what describes the processing and analysis of huge amounts of data with complex structure [3] and is sometimes referred to as a scaled BI&A [5]. The properties of Big Data, "high-volume, high-variety, high-velocity, and high-veracity" [6], often referred to as the three "HVs" [6], make it necessary for enterprises to come up with new requirements on storage, management, analysis and visualization of data [3]. One proposed solution to deal with this requirements is the use of in-memory databases, which describe database systems holding the entire dataset in main memory [7]. A study conducted amongst German IT-professionals in 2012, pointed out that every fifth enterprise thinks that meeting the challenges of the future data volume is only possible using in-memory technology. But the same study pointed out, that 60% of the enterprises are not able to make a clear decision on whether in-memory databases are relevant to them. They are especially concerned whether the proposed advantages of in-memory technology can really be leveraged in their enterprise [2].

To achieve a better understanding about the effects of in-memory databases and to be able to make a found decision on whether to invest into in-memory technology and migrate from traditional disk-based storage to main memory storage enterprises need tests to be executed, comparing those technologies in enterprise use case scenarios and providing insights on basic characteristics of the database systems. Those tests will help enterprises to determine the most suitable set up for their needs. This project aims on developing some basic test cases and the necessary toolset for executing this kind of tests.

2 Project Implementation

2.1 Project Description

The project aims at the development of a testing environment for the comparison of response times of in-memory based and classical disk storage databases in enterprise use case scenarios.

Two identical virtual machines, one with an in-memory database system, the other with a classical disk based storage database are used to verify, that the developed test cases can be executed on such environments. Therefore the same dataset is deployed to both virtual machines. Queries are generated and executed from a third, independent source where response time is measured, too. The target is to provide a test setup including enterprise use case scenarios being able to compare two database systems.

2.2 Project Phases

In a first phase a literature review on database performance and benchmarks was conducted to identify relevant stakeholders, use case scenarios and requirements for testing. During this literature review four major groups of stakeholders and their major topics of interest could be identified. The four groups, consisting of database developers, database administrators, business analysts and end-user/decision makers were characterized and the assumption was made, that the group of business analysts is the most appropriate target group and that scenarios in data mining such as the identification of frequent item-sets or the measurement of profits and losses generated by a specific sales-channel and its components are appropriate use case scenarios for testing. Furthermore the absence of tests comparing an in-memory database to a classical disk storage database in a comprehensible, reproducible way and on equal conditions could be approved.

In a second phase a framework for plan execution was developed simulating different circumstances for the virtual machines. The framework focuses on varying three components in the test environment - namely the number of threads executing one query in parallel, the size of the dataset and plans of execution, determining whether update and select statements are executed irrespective of each other or not. The number of threads and the different datasets represent the different amount of users and size of an enterprise. Different plans of execution allow a comparison of the databases for different areas of application. This increases the expandability of the tool to further application domains and aims on evaluating performance in mixed OLTP (Online Transaction Processing) and OLAP (Online analytical Processing) workloads.

In a third phase appropriate datasets were generated and queries for testing were developed. Furthermore a tool was implemented to execute the tests and measure response time on both virtual machines on equal conditions.

Data is extracted from the publicly available data generator of the TPC-DS benchmark, an industry standard benchmark for decision support systems (see: www.tpc.org), as it is available for free and approved to provide appropriate data for performance testing. For reasons of simplification only two of those tables were used. “catalog_sales” and “catalog_returns” which provide a fact table of sales through the catalog sales channel including items and orders and the profit and loss information of each line item. This information, in combination with the page number of the catalog, the item was ordered from, allows to execute the afore mentioned business use case scenarios. The tool for query execution is written in Java and uses JDBC to send the chosen queries to the databases and measure their response time.

3 Description of the Virtual Machines

The virtual machines provided by the HPI Future SOC Lab both had 64 GB main memory, a hard drive disk with a capacity of 100GB and a total of eight CPUs available. They are both located in the same network and run a Linux Kernel 2.6.32.59-0.7.

4 Test-framework and Toolset

4.1 Requirements

The test-framework is supposed to provide decision support for enterprises when deciding on switching to or adding an in-memory database to their IT landscape. Therefore the framework for testing has to simulate several environmental scenarios/enterprise characteristics and provide details about the influences on performance if those environmental factors change. As a test environment will never exactly match a real enterprise, metrics of comparison should be expressed in relative numbers and an easy, comprehensible way such as “On a level playing field, an increase of users by factor x performance loss of system A is factor z and performance loss of system B is factor y - B runs r times faster than A”.
Another important factor for generating an as equal as possible setup of the two databases, is to forbid every kind of tuning, such as additional indexes. Methods or elementary approaches as parts of the database management system can be used. Therefore tuning mechanisms such as adding further indexes to a table have to be avoided. Using a columnar table layout is appropriate. Basically the databases are supposed to be used in out-of-the-box versions as extra tuning might not be available for both database instances the same way and the test is not supposed to compare different tuning mechanisms.
Queries are supposed to be expressed in a way, that both database systems can handle the same expres-

sion as good as possible. Furthermore the queries have to be repeatable and comprehensible and have to be logged or generated in a way that reproduction is explicit.
The combination of queries and workload should try to emphasize the differences in the database systems and should therefore be designed to point out strengths and weaknesses of both instances.
The measurement of response time may not include the time necessary for query generation or the writing of result-sets. Therefore the machine running the test tool shall not be changed during tests and only the execution time of the SQL Statement is to be measured.

4.2 Design of tests

Environmental Factors
To meet the above mentioned requirements for simulating environmental factors the Queries are supposed to be executed by {1, 5, 10, 20, 40} Threads in parallel on a catalog_sales table of sizes {5GB, 10GB, 20GB, 40GB, 60GB, 80GB}. Those values were chosen as they can be computed in a relatively moderate time and are based on almost constant growth-rate. The high volumes of data are selected as they are close to (60GB) or even above (80GB) the available main memory (64GB).
Additionally to the catalog_sales table the catalog_returns table is used with a constant size of 0.5 GB.

Queries
One test consists of six queries which can be grouped into three categories. The first category consists of queries which perform very basic and simple SQL statements like selects or unions. The second group of queries are either inserting lines or updating tables and therefore represent OLTP workload. The third group includes rather expensive queries including joins and calculations.
It has to be mentioned, that the queries are not designed to perform their tasks in the most efficient way, but to generate workload on the database. The six queries are:

- Insert lines: a specified number of Lines {1, 50, 100, 150, 200, 250} is inserted into the catalog sales table at once. This query was chosen to evaluate the write performance for a high number of columns.
- Update one: This query updates one column in the catalog_sales table for a specific collection of items. This helps to evaluate update performance of single column and creates OLTP workload.
- Update six: This query updates six different columns of the catalog sales table for a predefined set of items. Here the update performance of multiple columns is tested.
- Select all: this statement simply returns the whole catalog_sales table and gives insight about the read performance of multiple columns.
- Select distinct: this statement returns the distinct values of item ids in the catalog_sales table and therefore provides insights about the read performance of a single column, what should be especially beneficial for a column store database.
- Frequent itemsets: This set of queries identifies Items which occur together in different orders a specific at least a predefined number of times. Through the implementation using an increasing number of self joins of the table catalog sales, join performance and the power of the internal query optimizer can be evaluated.
- Profit loss per catalog page: This query calculates based on the catalog_sales and catalog_returns table which page of a catalog caused which profit and what amount of loss. Here the performance of calculation operations such as sum() can be compared.

5 Test implementation

As mentioned before the tool suite was written in Java and uses JDBC to connect to the different instances. The architecture of the tool is rather simple, as one Class "Executer" executes the whole test suite by receiving Query-Strings of the "Generator" classes and starts the desired amount of threads "Runners".

Those threads can be executed in a variable and configurable order and amount, allowing to generate a mixed workload, too.

Response time is measured by storing the current system time before sending executing a statement and immediately after the result is received.

In the following examples of the queries are provided and some difficulties identified are described.

Insert lines: For this query the disk storage database and the in-memory database have to use different SQL-statements. For the classical database the typical "insert into table values (val1),(val2)…" statement is used. Unfortunately the tested in-memory database is not able to process such a query with more than one value. A workaround proposed, is to write the query as "insert into table (Select value1 from dummy union all Select value2 from dummy …). But this statement does not allow to insert more than around 250 lines at a time as the query is then told to be too complex to be processed. To avoid primary key constraints, each thread accesses its own file generated from one big TPC-DS catalog_sales data file. After one iteration, when all threads have finished, the data is deleted and the insert begins again.

The update one query is defined as "Update catalog_sales Set cs_warehouse = {a randomly generated integer of equal size for each Thread} where cs_item_sk = {predefined value} OR cs_item_sk =

…". The update six query is written accordingly but sets five more columns {cs_ship_mode_sk, cs_call_center_sk, cs_promo_sk, cs_ship_hdemo_sk, cs_ship_cdemo_sk} to new values. Those columns were chosen as they have no constraints and are not used by any other query.

"Select * from catalog_sales" is the statement executed by the select all statement and "Select distinct cs_item_sk from catalog_sales" is the query behind select distinct.

More interesting and causing a really heavy workload on the database is query set for identifying frequent itemsets.
The queries are generated by one thread executing queries in the style of "Select cs1.cs_item_sk as Item1 , cs2.cs_item_sk as Item2 from catalog_sales cs1 inner join catalog_sales cs2 on cs1.cs_order_number = cs2.cs_order_number where cs1.cs_item_sk < cs2.cs_item_sk AND group by cs1.cs_item_sk, cs2.cs_item_sk having count (cs1.cs_order_number) > {predefined value}" with an increasing amount of items, until no result is returned. Those queries are then stored in a list and executed one another by the "Runner". For each of these queries the time is locked. Unfortunately, the in-memory database stops when calculating itemsets with three items, while the classical database exceeds this number.

The profit loss per catalog page query is defined as:
"Select page_sk, sum (sales_price) as sales_price_sum, sum(profit) as profit_sum, sum(return_amt) as returns_sum , sum (net_loss) as loss_sum From (select cs_catalog_page_sk as page_sk, cs_sold_date_sk as
date_sk, cs_ext_sales_price as sales_price, cs_net_profit as profit, cast(0 as decimal(7,2)) as return_amt, cast(0 as decimal(7,2)) as net_loss from catalog_sales
union all select cr_catalog_page_sk as page_sk, cr_returned_date_sk as date_sk, cast(0 as decimal(7,2)) as sales_price, cast(0 as decimal(7,2)) as profit, cr_return_amount as return_amt, cr_net_loss as net_loss from catalog_returns) group by page_sk". Here a number of calculation operations are executed.

6 Next Steps and future Research

In a next step of the project, the queries for analyzing frequent item-sets have to be optimized to follow for example an a-priori algorithm or to leverage views. Furthermore the set of tests can be expanded, although a basis for a performance comparison of an in-memory database with a classical disk storage database is possible. The tests have to be executed and results have to be analyzed for what kind of enterprises the migration to an in-memory database offers the promised advantages and under which circumstances the switching cost exceed the delivered value.

The modular system of the testing tool allows it easily to expand the set of tests to further database systems or application domains. This makes it possible to compare multiple in-memory databases and compare the performance in other application domains as well.

References

[1] SAP AG, "The Global Information Technology Report 2012: Harnessing the Power of Big Data in Real Time through In-Memory Technology and Analytics", 2012.

[2] Frank Niemann, "In-Memory-Datenanalyse in Zeiten von Big Data," 2012.

[3] H. Chen, R. H. L. Chiang, and V. C. Storey, "Business Intelligence and Analytics: From Big Data to big Impact," in MIS Quarterly, pp. 1165–1188.

[4] SAP AG, SAP Solution Brief Business Analytics: SAP® In-Memory Appliance (SAP HANA™) The Next Wave of SAP® In-Memory Computing Technology, 2011.

[5] S. Rogers, "BIG DATA is Scaling BI and Analytics," Information Management (1521-2912), vol. 21, no. 5, pp. 14–18, 2011.

[6] M. Courtney, "Puzzling out Big Data," Engineering & Technology (17509637), vol. 7, no. 12, pp. 56–60, 2013.

[7] H. Garcia-Molina and K. Salem, "Main memory database systems: an overview," IEEE Transactions on Knowledge and Data Engineering, vol. 4, no. 6, pp. 509–516, 1992.

Future SOC Lab Autumn Term Project Activities Report:
Benchmarking for Efficient Cloud Operations

Dr. Ralph Kühne
Multitenancy Project Team
SAP Innovation Center
Potsdam Prof.-Dr.-Helmert-Str.
2-3 14482 Potsdam, Germany

Abstract

This project's overall focus was on the efficiency of cloud services provided in a multitenant environment. Several approaches to multitenancy have been implemented and evaluated using a multitenant-variant of the CBTR (Composite Benchmark for Transactions and Reporting) benchmark. Tenant workloads were varied in intensity and composition (analytical and transactional shares) and the effects on throughput and response time were measured. In the now-ending and for the project last lab term we completed a set of experiments based on the virtualization platform Xen. The overall results allowed interesting insights into the behavior of the approach implementations, but have been classified as company confidential and therefore must not be presented in this report.

1 Report

The idea behind the cloud is to offer computing as a service with seemingly endless capacity that can be added or removed on demand [1]. The cloud customer keeps the data inside the cloud infrastructure and has access to the performance of a data center to execute complex operations on it. Through the network, data can be accessed in an easy way with various devices.

Cloud computing turns the IT investments of companies that move into the cloud into operational expenditures that they pay for the consumed services to a cloud provider. Consequently, the risk of correctly dimensioning the infrastructure as well as the need to keep capital expenditures as well as administrative costs at viable levels is transferred to the cloud provider. Multitenancy, i.e. consolidating several customers onto the same infrastructure, is one method to achieve higher utilization and therefore a more efficient cloud operation [2].

This project investigated several ways of accommodating several customers on one server, such as private machine, shared machine, shared database process, and shared table [2]. These approaches were compared with respect to their effect on throughput and on response time using two server machines of the Future SOC Lab. One served as database server (32 cores) and the second one as client machine (64 cores) from which the client requests were submitted and where the performance was measured.

The experiments employed CBTRmt, our multitenancy extension of CBTR [3], which is a mixed-workload benchmark based on SAP's Order-to-Cash process comprising four analytical-type queries and nine transactional-type queries. Our extension allowed us to simulate a larger number of client organizations (for most experiments we selected 20 tenants) with various data sizes and request behavior in a highly concurrent manner.

In a first batch of experiments, we used analytical load, then added transactional load (20% and 80% share) and finally simulated geo-diverse tenant localizations, i.e. not all tenants were active at all times, but followed an overlapping diurnal load pattern with one group entering night time while the other group woke up. This was done in the two previous lab terms.

In the now-ending lab term we were able to complete our investigations for a further approach to realize multitenancy based on the virtualization platform Xen.

The results that we gathered allowed interesting insights into the approach implementations and their effect on system performance, but have been classified as company confidential and therefore must not be detailed in this public report.

2 Acknowledgements

The project members wish to thank André Pansani and Sascha Hellmann of SAP's FSOC lab team for their invaluable administrative support to setup the virtualized environment for the experiments that were conducted as part of the project in the now-ending lab term. An all-lab-terms thanks goes again to Bernhard Rabe of HPI.

3 References

[1] Peter Mell, Timothy Grance. *The NIST Definition of the Cloud Computing*. NIST Special Publication 800-145. 2011.

[2] Dean Jacobs, Stefan Aulbach. *Ruminations on Multi-Tenant Databases*. BTW 2007: 514-521

[3] Anja Bog, Hasso Plattner, Alexander Zeier. *A mixed transaction processing and operational reporting benchmark*. Springer Science + Business Media, LLC 2010.

Blog-Intelligence Extension with SAP HANA

Patrick Hennig
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
patrick.hennig@hpi.uni-potsdam.de

Philipp Berger
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
philipp.berger@hpi.uni-potsdam.de

Christoph Meinel
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
office-meinel@hpi.uni-potsdam.de

Abstract

Blog-Intelligence is a blog analysis framework, integrated into a web portal, with the objective to leverage content- and context-related structures and dynamics residing in the blogosphere and to make these findings available in an appropriate format to anyone interested. The portal has by now reached a mature functionality, however, requires ongoing optimization efforts in any of its three layers: data extraction, data analysis and data provision.

1 Introduction

With a wide circulation of 180 million weblogs worldwide, weblogs with good reason are one of the killer applications of the worldwide web. For users it is still too complicated to analyze the heavily linked blogosphere as a whole. Therefore, mining, analyzing, modeling and presenting this immense data collection is of central interest. This could enable the user to detect technical trends, political atmospheric pictures or news articles about a specific topic.
The basis of the Blog-Intelligence project is the big amount of data provided by all weblogs in the world. These data is gathered in the past and in the future by an intelligent crawler.
Blog-Intelligence already provides some basic analysis functionality for the crawled data. Through the improvement of the crawler and the consequent growing amount of data, the analysis gets into big performance issues. Since these performance problems, the analyses are only calculated in a weekly manner to reduce the run time of the analyses algorithms. Therefore the up-to-dateness of the results is not given any more and the web portal is only able to show already deprecated results.

2 Fields of application

With SAP HANA totally new opportunities are coming up. The fast execution of the analysis algorithms provides completely new and better interaction with the system for the end-user. Beside the advantage of exploring the blogosphere in real time, it is possible to provide analyses for the end-user calculated separately for each user with his interests. Furthermore, former time-consuming text and graph analysis algorithm can now get integrated into our framework because SAP HANA offers fast variants of these algorithms. This opens new perspectives onto the data and the blogosphere for the user.
For example, it is now possible to figure out, how and what is discussed about products or companies inside the blogosphere. Traditional providers limit these analyses to the biggest blogs worldwide. With Blog-Intelligence and SAP HANA it gets possible to calculate analyses over all weblogs worldwide.
In addition to the personalized illustration of the blogosphere, companies can figure out how their own weblogs perform and influence the blogosphere. Even the monitoring of competitors' social media influence is imaginable.

3 Used Future SOC Resources

Currently, we are running a small test machine that is embedded into the Future SOC network. This machine runs a SAP HANA instance that is used as test database for our current crawler development. The development state of the crawler becomes a stable version and the current version is able to crawl fast and more enriched data. Therefore, we observed the SAP

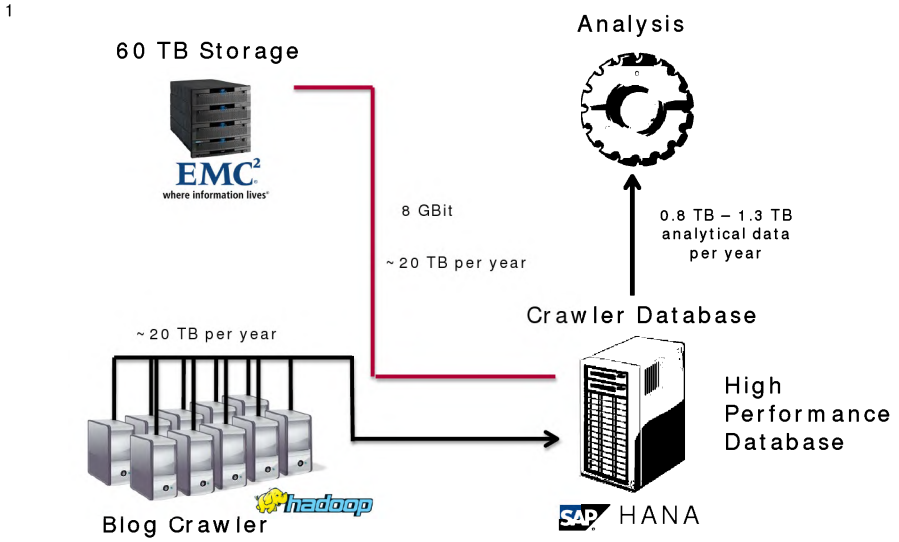


Figure 1: Technical Concept of Blog-Intelligence

HANA instance on our test machine to run out of resources. Hence, we revise our current setup and come up with a new Blog-Intelligence deployment described in the following section.

4 Blog-Intelligence deployment

Our overall technical setup is shown in figure 1. In order to increase the amount of data and to keep the data up-to-date, the weblogs are visited and revisited with the help of a crawler. The crawler is based on the MapReduce framework from Apache. The first structural analyses are done at crawling time like the language detection. The overall detection whether the web site is a blog, a news portal or a regular HTML page is also done by the crawler.
This crawler is executed in parallel and distributed on a Hadoop Cluster and saves the data directly into a SAP HANA instance. The complete extracted data, especially large objects like the original HTML content, is stored as well. Nevertheless, the large data objects get outsourced to a data storage provided by EMC with an overall capacity of 60TB. Although large objects get outsourced, an huge amount of data stays in-memory for the analytics component.

5 Computational Effort Estimation

In the worldwide web approximately 10 million highly active blogs exists with more than one post each week. An important part of these blogs is the news portal blogs, like "theguardian" with several new posts each day. Of the widespread 180 million weblogs these 10 million weblogs are the most important weblogs. Therefore, we want to store these weblogs inside a SAP HANA database to provide up-to-date real time analyses. As a result, we expect 0.8TB to 1.3 TB of compressed data for our analyses.

6 Next Steps

As mentioned before, the crawler implementation is nearly stable. Hence, we will start a permanent run using the Future SOC resources and create an adequate set of crawled weblog sites. Given this dataset in a running SAP HANA instance, we get able to the test in-memory data analysis algorithms of HANA. We expect to measure a significant performance improvement for the analytics. Thereby, we get able to evaluate real-time and user-centric analysis approaches for the blogosphere. Based on our findings about the blogosphere, we will adapt, develop and re-engineer our existing exploration interfaces for this new level of analytics.

References

[1] P. Berger, P. Hennig, J. Bross, and C. Meinel. Mapping the blogosphere–towards a universal and scalable blog-crawler. In *Privacy, Security, Risk and Trust (PASSAT), 2011 IEEE Third International Conference on and 2011 IEEE Third International Confernece on Social Computing (SocialCom)*, pages 672–677. IEEE, 2011.

[2] J. Bross, P. Hennig, P. Berger, and C. Meinel. Rss-crawler enhancement for blogosphere-mapping. *IJACSA Editorial*, page 51, 2010.

[3] J. Bross, M. Quasthoff, P. Berger, P. Hennig, and C. Meinel. Mapping the blogosphere with rss-feeds. In *Proceedings of the 2010 24th IEEE International Conference on Advanced Information Networking and Applications*, AINA '10, pages 453–460, Washington, DC, USA, 2010. IEEE Computer Society.

Generating A Unified Event Representation From Arbitrary Log Formats

Amir Azodi, David Jaeger, Feng Cheng, Christoph Meinel

Hasso-Plattner-Institut
University of Potsdam
14482, Potsdam, Germany

{amir.azodi, david.jaeger, feng.cheng, meinel}
@hpi.uni-potsdam.de

***Abstract*—Today, with the growing complexity of computer networks and companies dependence on such networks, securing them is more important than ever. Intrusion Detection systems have become a necessary tool to protect networks and privileged data from falling into the wrong hands. However IDS systems are limited by the amount of environmental information they process; including network and host information. Additionally the speed at which they receive such information can have a tremendous effect in the time needed to detect attacks. We have pinpointed three areas of Event Normalization, Alert Gathering and Event Correlation, where improvements can lead to better and faster results produced by the IDS.**

I Introduction

The current state of IDS systems does have some weaknesses. One of the biggest problems is the inability to detect some of the more complex attacks; due to lack of access to the information which would have led to a successful detection. An underlying reason for this problem is the limitations of the viewpoint of any single sensor in the network. An example of such a scenario is the access to log files. Log files can contain large amounts of information regarding any attempts to breach security of a system. This problem becomes more evident when application logs are considered. The possibility to correlate between events gathered from different logs and systems, provides a unique level of access to information needed to detect the more advanced and well hidden attacks [3].

The rest of this report is organized as follows.

- Review of HPI Security Analytics Lab: In this section an overview of the legacy Security Analytics Lab (SAL) is given as background information.
- Updates - Design and Architecture: This section outlines the changes that were made to the legacy SAL system in order to incorporate new features and possibilities for detecting attacks.
- Leveraging Live Environment Information for Instant Attack Detection: This section describes the gathering of information in order to generate a sound and complete network graph to be used in conjunction with a comprehensive vulnerability database for the generation of a live attack graph.
- Extracting Attack Activities from Logging Information: Discusses the methods behind extracting all possible log information and to normalize them into one single format.
- Results and Achievements: In this section particular attention is given to the deliverables and completed objectives of the new phase, so far.

II Review of HPI Security Analytics Lab

The SAL represents a system to encounter the challenges and provide a high level of security in a network. Different Log Gatherers and IDS Sensors provide a variety of data sources for the complex analysis on the In-Memory based platform. A multi-core-supporting architecture is the foundation for high performance as well as real-time and forensic analysis. Using efficient algorithms and various visualization techniques supports security operators with the challenging task of defending the network by identifying and preventing attacks [2].

The present deployment of SAL on the FutureSOC Lab has the following features listed below. This is a feature freeze instance and excludes the latest changes implemented into SAL. The latest changes will be added as part of the next deployment phase throughout the next six months.

- Detection of complex attack scenarios
- In-Memory based platform with up to 2 TB of main memory
- Multi-Core support with thousands of cores
- Correlation of events from a variety of data sources
- Utilization of environment information represented by attacks graphs
- Ranking of complex alert dependency graphs
- Visualization of attack scenarios and complex alert relations

III Updates: Design and Architecture

The new design follows the 3 level architecture or chain of command system. First there is a component named *Enforcer/Gatherer*, which is in charge of gathering and sending information relating to the host it is installed on. This information includes log files, system data such as installed software, available hardware and system status. At the second

level a component named *Agent* receives the information from the Enforcer and normalizes it into CEE [5] events or control messages before sending them to the Server. Finally the *Server* will make the information persistent and continues on to generating an attack graph and correlating events received in order to find attacks. Figure 1 illustrates this workflow.

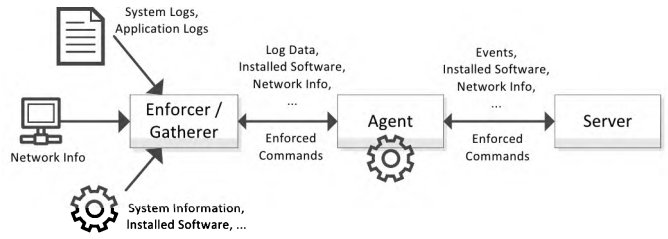


Fig. 1: Message workflow in the SAL

IV Leveraging Live Environment Information for Instant Attack Detection

The design proposed in this report allows for complete access to the host information of every host on which the Gatherer is deployed. This information includes, but is not limited to, log files (e.g. syslog), memory information, processor information, network information (e.g. hostname, IP addresses and MAC addresses) and application information (e.g. their CPE [6], which allows SAL to compile a list of current vulnerabilities present on any host).

Having imported all the information into SAL, a network graph can be created. Using a vulnerability database we can then construct a comprehensive attack graph to be used by SAL. In order to always have the most up-to-date attack graph, the SAL system continuously updates its underlying data structures, i.e. the network graph and the vulnerability database. Therefore at any moment an up-to-date attack graph can be compiled from the live information present in the underlying data structures.

V Supporting Intrusion Detection with Event Log Information

A common way for reporting intrusions are security alerts, which are generated by so called intrusion detection systems. Generally, these alerts are generated as a result of observed activities in an environment that are potentially harmful. However, an alert is not a guarantee that an actual intrusion has taken place. In addition, alerts are only limited to detected malicious activity, but they do not give information about the attackers activities before and after his intrusion attempt. An approach to solve the shortcomings of an intrusion detection with alerts only is the consideration of additional details about the attacker activities. One type of such information are event logs. They are produced by applications, monitoring systems, operating systems, etc. and are give an insight into all observed events in an environment. The next subsections show how these event logs can be gathered from different locations in an environment, how they are normalized into an automatically processable format [4].

A. Information Gathering

Event logs usually originate from a multitude of sources, such as IDSs, firewalls or applications, where each source produces a different log representation and output format. Examples for log representations are Syslog or the IDMEF specification and typical output formats are binary, CSV and XML. In order to read events from the log sources, the previously mentioned gatherers are employed. They can be attached to the log files on their host and are able to send the file content to their centrally connected agent.

B. Normalization

The received event logs are interpreted event by event, which makes it necessary to split the sequential event stream into single events. The SAL realizes this splitting with the help of regular expressions that are on the matching of event separators. Once the events have been extracted from the event stream, they are still in a format being specific to the generating event source. Therefore, each event is further transformed into a common and easily processable representation. In the concrete SAL implementation, the *Common Event Expression* (CEE) [5] has been chosen as the common representation of events, to which all custom event are normalized to. The CEE makes use of event containers that contains the properties of the event as key-value-pairs.

In order to normalize events to the CEE format, information from the raw events have to be extracted and put into the corresponding fields. This extraction of event properties from the raw events is performed with the support of regular expression, or more specifically with the group mechanisms in regular expressions. The idea is to first define the format of a single raw event in the expression and then surround structured information with regular expression groups, so that they can be matched and finally used as the properties of the event.

VI Results and Achievements

The mechanisms that have been described in the last two sections allow the SAL to react on live changes in the network, verify intrusions and uncover intrusions that could not be detected with the legacy SAL. In addition, it is possible to correlate the discovered intrusion attempts to the nodes in an attack graph. The described changes are mainly realized by interpreting dynamic and static environment information as well as using event logs instead of only alert logs.

VII Conclusion

As we work to secure networks from ever more sophisticated attacks, it is clear that the industry is in need of more intelligent protection mechanisms that actively protect our networks. As part of this project we aim to design intelligent solutions that consider a wide array of information relating a network and are therefore better able to detect intrusions. The initial results of our method show considerable promise. In the next phases of the project, we will focus finishing a POC implementation of our design principles and functional requirements. We also look to publish the results of our POC

implementation and the effectiveness of the system as a whole in subsequent reports aimed at the Future SOC Lab.

References

[1] Sebastian Roschke, Feng Cheng, Robert Schuppenies, and Christoph Meinel: Towards Unifying Vulnerability Information for Attack Graph Construction , in Proceedings of 12th Information Security Conference (ISC'09), Springer LNCS, Pisa, Italy, pp. 218-233 (September 2009)

[2] Christoph Meinel, Andreas Polze, Alexander Zeier, Gerhard Oswald, Dieter Herzog, Volker Smid, Doc DErrico and Zahid Hussain (Eds.): Proceedings of the Fall 2010 Future SOC Lab Day , Technical Report of Hasso-Plattner-Institute, Heft 42 (2011)

[3] Sebastian Roschke: Towards High Quality Security Event Correlation Using In-Memory and Multi-Core Processing , PhD Thesis, Hasso Plattner Institute at University Potsdam (May 2012)

[4] David Jaeger: Monitoring in Scenario-based Security Experiments , Master Thesis, Hasso Plattner Institute at University Potsdam (August 2012)

[5] The CEE Board: CEE Overview. Available from: http://cee.mitre.org/docs/overview.html

[6] Common Platform Enumeration:. Available from: http://cpe.mitre.org/

Detecting biogeographical barriers - testing and putting beta diversity on a map

Johannes Penner
Museum für Naturkunde, Leibniz-Institute for Evolution & Biodiversity Research, Department of Research, Herpetology Group
Invalidenstrasse 43
10115 Berlin, Germany
johannes.penner@mfn-berlin.de

Moritz Augustin
Technische Universität Berlin, Department of Software Engineering & Theoretical Computer Science, Neural Information Processing Group
Marchstr. 23
10587 Berlin, Germany
augustin@ni.tu-berlin.de

Abstract

Biogeographical regions are not only of scientific but also of conservation importance. The most common approach is to map species richness. However, beta diversity (species turnover) is better suited to identify where boundaries between regions are.

We calculated beta diversity for West African amphibians on a fine scale (30 arcseconds), using moving window and parallelization techniques. This would have been impossible with standard hard- and software. Not only was this the first time to produce such fine scaled maps but we systematically compared different indices and different moving window sizes.

Currently our results are only preliminary, the work plan has not been fulfilled yet and detailed analyses are pending. Furthermore we would like to ask for an extension of access possibilities and enhance our study significantly by switching from binary input data to proxies for abundance data.

1 Background

Amphibians form an integral part of biodiversity and are one of the most threatened vertebrate groups on the globe, with more than one third of all known species listed as threatened on the IUCN Red List [1, 2]. Biodiversity is declining in many areas and includes amphibians. The so called amphibian decline is a global problem and due to a number of reasons. The main ones are habitat destruction, alteration and fragmentation as well as pesticides, climate change, overharvesting and emerging diseases [3]. Causes are also often interacting. Thus, well-informed conservation decisions are needed, especially for areas which harbor the highest diversity but are poorly studied, i.e. the tropical regions. One base for such decisions is the answer to where and why species occur.

In general, amphibian diversity and biodiversity are measured on three different scales: genes, organisms and landscapes. Organismic diversity is the most common measure. Measured at one site it is called alpha diversity, the comparison between two sites is named beta diversity and gamma diversity is calculated across a landscape [4].

Besides the general interest where the highest species diversity is found, the distribution of diversity is important for conservation. One main aim is to set priority areas when efforts have to be concentrated. Another aim is to provide decision makes with the scientific sound arguments for areas in need of protection.

In general, a trait off exists between the conservation side which requests fine grained maps and the scientific need to cover large areas. Studies fulfilling both requirements, covering a large area with a high resolution, are scarce. Our study is one of the few attempts to provide both. We study West African amphibians, which are unique and face multiple threats; mainly severe habitat destruction and fragmentation (e.g. see [5]).

2 Previous work

We used Environmental Niche Modelling (ENM; also commonly called Species Distribution Modelling) to derive the occurrence of West African amphibians on a grid of 30x30 arcseconds (≈ 1km²). So far ENMs were constructed for 158 out of ca. 180 known species with more than 9000 occurrence records. The ENMs used 18 environmental parameters: ten climate (different temperature and precipitation

measures), five vegetation (different wavelengths from two satellites), two altitudinal (calculated from a digital elevation model and one "hydrology" parameters) (compare to [6]). A machine learning maximum entropy algorithm [7,8,9] compared environmental parameters at sites of occurrences against randomly sampled background data (see [10] for a statistical explanation of the algorithm). This resulted in map showing the distribution of modeled species richness covering the whole West African region (Penner *et al.* in prep.).

3 Project idea

Inspired by McKnight *et al.* [11] we wanted to investigate species turnover (≈ beta diversity) for the whole region. The main aim is to identify biogeographical regions and boundaries, which are reflected by relatively high species turnover. This is done by comparing species occurrences between neighboring grid cells.

To our knowledge this approach has never been tackled systematically, especially with real data, contrary to simulated data. No agreement exists for example which index should be used though some progress has been made. Therefore we will compare different indices and different moving window sizes.

The final programmed software will be open source published using a public domain license. It will allow the user to specify own indices and moving window sizes.

4 Used Future SOC Lab resources

The large scale of our analysis does not permit the use of available standard computer hard- and software. Applying a moving window approach we utilized the HPI Future SOC Lab Fujitsu RX600S5 machines, parallelizing data. Though it would be interesting to test how the Hewlett Packard DL980 G7 servers would speed up calculations.

5 Findings

Currently we programmed two binary indices Jaccard [12] and Mountford [13]. Beta diversity is calculated as the species turn over between one grid cell and its neighboring cells. A third index, Raup-Crick [14], will be implemented soon. All three indices use binary occurrence data and weigh presence and absence data differently (see [15]). Moving window sizes varied from 3 to 93 (Mountford) and 201 (Jaccard). Larger sizes will be explored but are very computing intensive.

Unique areas are consistently showing up regardless of the index and the moving window size (see poster presented at the Future SOC Lab day, 10th of April 2013). Preliminary analyses indicate that biogeographic barriers might correspond to major river systems in the region. However, at the current, stage proper conclusions are premature.

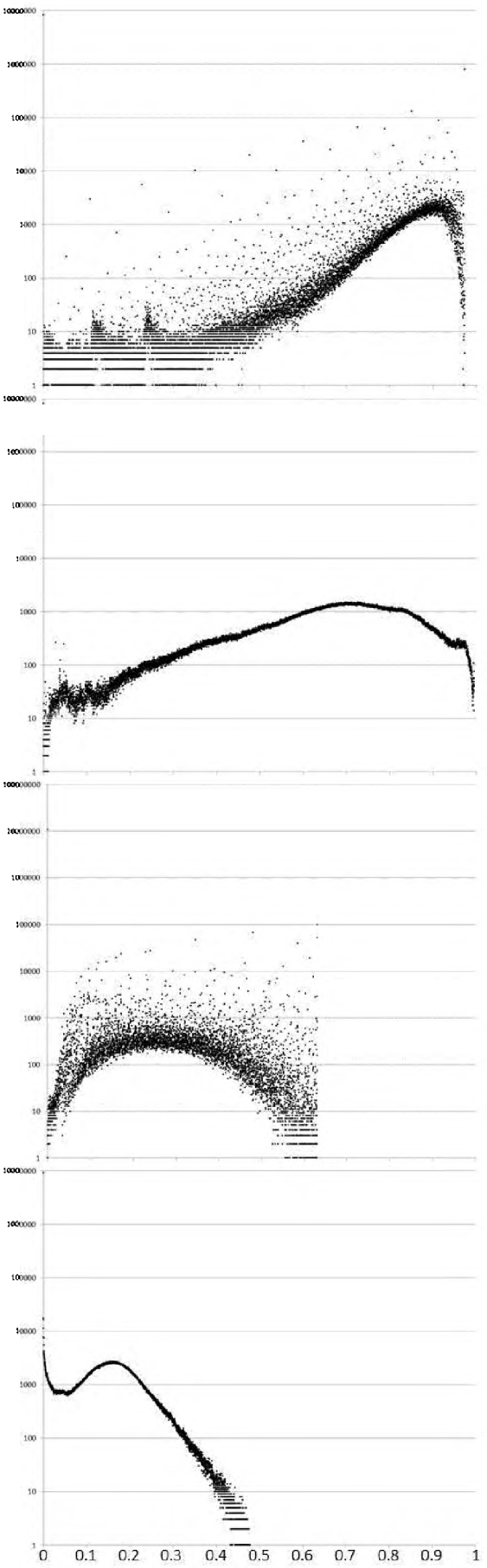


Figure 1 (from previous page): Example of frequency histograms of two different beta diversity indices with two different moving window sizes. From top to bottom (index name & moving window size): Jaccard 3x3, Jaccard 93x93, Mountford 3x3, Mountford 93x93.

Figure 1 shows four examples of frequency histograms for two indices (Jaccard & Mountford) and two different moving window sizes (3x3 & 93x93). It The non-metric properties of the Mountford index are clearly visible. It is obvious that both factors are important. More thorough analyses will be prepared.

6 Next steps

Our first runs were successful. Due to delays in the start of the project, analyses have not been completed yet and time and logistical constraints did not allow us to finish all intended runs. As mentioned above the third index awaits implementation.

In addition, we have discovered a solution to move the analyses even one large step further. So far we used binary data, resulting in relatively simple species richness maps. This was a result of the current work flow but has many theoretical disadvantages.

Therefore, we would like to use proxies for abundance. This would open a completely new window, incorporating such measures as heterogeneity and evenness in our analyses and potentially a much higher precision. Using "abundance" data, different indices are needed. Five potential candidates are: Shannon, Whittaker, Wilson & Shmida, NESS and C_{qN} which weigh evenness differently (e.g. see [16]).

Furthermore, using the likelihoods gained from the ENMs will enable us to calculate alpha diversity anew, also incorporating heterogeneity and evenness.

7 Extension request

So far the possibilities of the HPI Future SOC Lab not only enabled our work but also took it a huge step forward. As mentioned above not all analyses were finalized yet and they should be further improved. A further use of the HPI Future SOC Lab facilities would help us tremendously. Therefore we would be very grateful if an extension of our access to the HPI Future SOC Lab resources is granted. A follow-up proposal has been submitted.

8 Acknowledgements

We thank the innumerous colleagues who shared their data. We are also indebted to a number of people who assisted in the field, in the laboratory and in various other ways, the most important ones are: Mark-Oliver Rödel, Jakob Fahr, Matthias Herkt, Günther Barnikel, Mirjana Bevanda, Mareike Hirschfeld, Jenja Kronenbitter, Anja Pfahler, Meike Mohneke & K. Eduard Linsenmair. Various funding agencies enabled the work. The main part of it was initially funded by the German Ministry for Education and Research (BMBF): BIOTA West Africa, 01 LC 0617J.
We are very grateful to the HPI Future SOC Lab for granting us access to their computing capacities!

References

[1] S.N. Stuart, J.S. Chanson, N.A. Cox, B.E. Young, A.S.L. Rodrigues, D.L. Fischman, & R.W. Waller: Status and trends of amphibian declines and extinctions worldwide. *Science*, 306: 1783-1786, 2004.

[2] IUCN: *The IUCN Red List of threatened species*. Version 2012.2. http://www.iucnredlist.org. Last accessed 17th October 2012.

[3] S.N. Stuart, M. Hoffmann, J.S. Chanson, N.A. Cox, R.J. Berridge, P. Ramani & B.E. Young: *Threatened amphibians of the world*. Lynx Edicions, Barcelona, IUCN, Gland, Switzerland; and Conservation International, Arlington, TX, 2008.

[4] A.E. Magurran: *Measuring biological diversity*. Blackwell Publishers, Oxford, 2003.

[5] J. Penner, M. Wegmann, A. Hillers, M. Schmidt & M.-O. Rödel: A hotspot revisited – a biogeographical analysis of West African amphibians. *Diversity and Distributions*, 17: 1077-1088, 2011.

[6] S.J. Phillips, M. Dudík & R.E. Schapire: A maximum entropy approach to species distribution modeling. In: C. Brodley, Editor. *Proceedings of the Twenty-First International Conference on Machine Learning*. New York, ACM Press: 655-662, 2004.

[7] S.J. Phillips, R.P. Anderson & R.E. Schapire: Maximum entropy modeling of species geographic distributions. *Ecological Modelling*, 190: 231-259, 2006.

[8] S.J. Phillips & M. Dudík: Modeling of species distributions with Maxent: new extensions and a comprehensive evaluation. *Ecography*, 31: 161-175, 2008.

[9] J. Penner, G.B. Adum, M.T. McElroy, T. Doherty-Bone, M. Hirschfeld, L. Sandberger, C. Weldon, A.A. Cunningham, T. Ohst, E. Wombwell, D.M. Portik, D. Reid, A. Hillers, C. Ofori-Boateng, W. Oduro, J. Plötner, A. Ohler, A.D. Leaché & M.-O. Rödel: West Africa - A safe haven for frogs? A sub-continental assessment of the Chytrid fungus (*Batrachochytrium dendrobatidis*). *PLoS ONE*, 8: e56236, 2013.

[10] J. Elith, S.J. Phillips, T. Hastie, M. Dudík, Y.E. Chee & C.J. Yates: A statistical explanation of MaxEnt for ecologists. *Diversity and Distributions*, 17: 43-57, 2011.

[11] M.W. McKnight, P.S. White, R.I. McDonald, J.F. Lamoreux, W. Sechrest, R.S. Ridgley & S.N. Stuart: Putting beta-diversity on the map: broad scale congruence and coincidence in the extremes. *PLoS Biology*, 5: e272, 2007.

[12] P. Jaccard: Nouvelles recherches sur la distributionflorale. *Bulletin de la Societe Vaudoise Sciénces Naturelles*, 44: 223-270.

[13] M.D. Mountford: An index of similarity and its application to classification problems. In: P.W. Murphy, editor. *Progress in soil zoology*. Butterworth, London: 43-50, 1962.

[14] D. Raup, & R.E. Crick: Measurement of faunal similarity in paleontology. *Journal of Paleontology*, 53: 1213-1227, 1979

[15] P. Legendre & L. Legendre: *Numerical Ecology, 2nd Edition*. Elsevier Science B.V., Amsterdam, 1998

[16] A.E. Magurran & B.J. McGill: *Biological Diversity – frontiers in measurement and assessment*. University Press, Oxford, 2011.

Heterogeneous Software Pipelining for Memory-bound Kernels

Fahad Khalid
Hasso-Plattner-Institut
University of Potsdam
14482 Potsdam, Germany
fahad.khalid@hpi.uni-potsdam.de

Andreas Polze
Hasso-Plattner-Institut
University of Potsdam
14482 Potsdam, Germany
andreas.polze@hpi.uni-potsdam.de

Abstract

The use of accelerator architectures such as GPUs is now ubiquitous in high performance computing (HPC). For certain combinatorial algorithms, the size of output generated can be much larger than the size of the input dataset. This is especially true when combinatorial operations are performed element-wise. The massive increase in output size requires frequent transfer of result data from the GPU memory to the CPU memory. Since this transfer operation is much slower as compared to the speed of computing on the GPU, it becomes a performance bottleneck.

The heterogeneous software pipelining approach has been developed to deal with the above mentioned problem. In this approach, we execute the kernel in two parts, spanned over both the GPU and the CPU. The computation is performed in a pipelined fashion.

In this report, we consider a highly memory-bound combinatorial kernel from the domain of Computational Biology, and improve its performance on a heterogeneous platform using the heterogeneous software pipelining approach. The results show that our approach not only makes it possible to gain speedup over a serial implementation; it performs even better than a multi-threaded CPU-only SMP version.

1 Motivation

In recent years, research community in the High Performance Computing (HPC) and Scientific Computing sectors has witnessed an increasing application of Heterogeneous Computing [1] (where Heterogeneous Computing refers to the design of applications that can harness the power of both the CPUs and accelerators such as GPUs).

Many scientific applications comprise of computational kernels that benefit significantly from a processor's capability to perform a large number of arithmetic operations at high speeds. Such kernels are typically *compute-bound* i.e. the performance bottleneck lies in the complexity of arithmetic operations (e.g. dense matrix-matrix multiplication [3]). Accelerator architectures such as GPUs dedicate most of the chip area to arithmetic processing units. This makes it possible for properly tuned compute-bound kernels to perform at levels close to the peak performance offered by the underlying architecture. A significant number of algorithms have been successfully ported to GPUs, attaining up to 100x speedup over serial execution on a CPU [2]. Such speedup is limited to compute-bound kernels.

There is however another set of kernels, where the performance bottleneck is dictated by the frequency of memory access operations. Such kernels are termed *memory-bound*. These kernels typically have a low arithmetic intensity (i.e. a low compute to main-memory access ratio). A common example for such a kernel is Sparse Matrix-Vector Multiplication (SpMV) [4], which is the foundation for linear solvers. In addition, certain combinatorial algorithms are memory bound as well. As stated earlier, accelerator architectures favor compute-bound kernels. This raises the following questions:

Is it possible to effectively utilize Heterogeneous Computing for algorithms with low arithmetic intensity? Or must such algorithms be executed on CPU-only systems?

2 Heterogeneous Software Pipelining

2.1 Concept

The approach proposed in this section is applicable to memory-bound algorithms that can be split into two parts:

1. A part with *high arithmetic intensity*
2. A part with *very low arithmetic intensity*

For a given such algorithm, the part with high arithmetic intensity is implemented as a GPU kernel. The other part is implemented as a CPU kernel. This particular distribution of the kernels amongst the two types of processors is based on the processors' capabilities to process such kernels. GPUs can effectively utilize their massively parallel architecture to process kernels with high arithmetic intensity, while CPUs outperform GPUs on kernels with very low arithme-

tic intensity. By splitting the algorithm into two kernels and executing these on the most suitable processor architectures respectively, a significant gain in performance is expected over a homogeneous parallel implementation.

Once the kernels have been implemented, these are executed in the following steps:

1. The GPU kernel is executed, which computes some intermediate results.
2. As soon as the intermediate results are generated, two things happen concurrently. The results are transferred from the GPU memory to the CPU memory, and the GPU starts processing the next batch of input data.
3. After the transfer of the first batch of intermediate results from GPU memory to CPU memory is complete, the CPU kernel is executed, which processes these intermediate results.

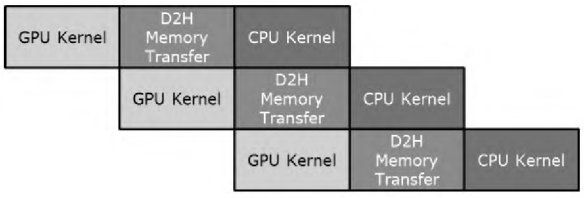


Figure 1: A 3-stage heterogeneous software pipeline.

The process is illustrated in Figure 1. The data processing loop is executed in a pipeline with 3-stages. All three stages are executed asynchronously, and therefore the waiting time between the different stages is minimized.

2.2 Results

The heterogeneous software pipelining approach was applied to a kernel from the domain of Computational Biology. This kernel is termed as *Combinatorial Candidate Generation*. The algorithm has already been successfully parallelized on both shared-memory [5] and distributed-memory [6] CPU based architectures. Here, a heterogeneous architecture based parallelization is presented. Three instances of the kernel were implemented: Serial, CPU-only parallel and heterogeneous software pipeline. Table 1 presents the execution times of the three different implementations, against four different datasets. The datasets represent real metabolic networks from the bacterium *E.Coli*.

As can be seen in the table, the CPU-only implementation outperforms the pipelined implementation on the first dataset, and the two implementations yield identical performance on the second dataset. This is due the fact that these datasets are very small, and the overhead incurred in pipelining over-shadows much of the performance gain. For larger datasets however, the heterogeneous pipelining approach yields a significant performance gain, resulting in a *6-fold* speedup over the serial implementation, and *1.8-fold* speedup over the CPU-only parallel implementation.

Table 1: Comparative results of three different implementations against four networks. Execution times (in seconds) are presented against each implementation and network; *m* is the number of metabolites, *r* is the number of reactions.

Network Size	#candidates	Time (s)		
		Serial	CPU-only	Pipelined
26m×38r	219743731	1.2	0.38	0.39
26m×40r	130992739	0.77	0.35	0.35
26m×31r	752482917	4.1	1.6	1.0
26m×43r	2616975505	14	5.5	2.5

3 Future Work

Following are some plausible directions for future work:

- *In-depth analysis of the heterogeneous software pipelining approach to investigate applicability to other kernels*: This will involve an empirical evaluation of the performance of different stages, finding the optimal parameter configuration to get the best possible performance, as well as finding the full set of limitations that may lead to improved solutions.
- *Investigation of arithmetic intensity as a useful metric to determine the most suitable processor architecture for a specific parallel kernel*: The objective here is to develop a quantitative model that given the arithmetic intensity of a kernel, provides a good estimate of performance expected on a given architecture. As per the current state-of-the-art, the decision of which architecture to choose is more a matter of expert opinion and empirical evaluation, than model based prediction. Once a suitable model is available, not only will it serve as a decision support tool, it might also provide further insight into the relative impact of certain architectural characteristics on the performance of memory-bound algorithms.

4 Publication

Based on the research described above, the following paper has been submitted for review:

- Fahad Khalid, Zoran Nikoloski, Peter Tröger, Andreas Polze. *Heterogeneous Combinatorial Candidate Generation*.

5 Conclusions

We have introduced a novel approach for managing arithmetic intensity of memory-bound combinatorial algorithms on heterogeneous architectures; especially for cases where transfer of data from the GPU memory to CPU memory forms the performance bottleneck. Our approach termed heterogeneous software pipelining has been successfully implemented for an application from the domain of Computational Biology. The results show a significant increase in performance as compared to serial and CPU-only parallel implementations.

In our collective work up till now, we have focused on improving the performance of the elementary mode enumeration problem from computational biology. To this end, we have made the following contributions:

- Improved the memory consumption behavior of the application on shared-memory SMP architectures.
- Developed the first ever heterogeneous implementation of the combinatorial candidate generation algorithm.

In addition, we have generalized our results to memory-bound combinatorial algorithms, for which the following novel method has been developed:

- Heterogeneous Software Pipelining

Based on our research and the above mentioned contributions, it can be seen that novel methods are needed for the effective utilization of heterogeneous resources when used for memory-bound algorithms.

References

[1] J. D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krüger, A. E. Lefohn, and T. J. Purcell, "A Survey of General-Purpose Computation on Graphics Hardware," Computer Graphics Forum, vol. 26, pp. 80-113, 2007.

[2] V. W. Lee, C. Kim, J. Chhugani, M. Deisher, D. Kim, A. D. Nguyen, N. Satish, M. Smelyanskiy, S. Chennupaty, P. Hammarlund, R. Singhal, and P. Dubey, "Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU," presented at the Proceedings of the 37th annual international symposium on Computer architecture, Saint-Malo, France, 2010.

[3] A. Buluç, J. R. Gilbert, and C. Budak, "Solving path problems on the GPU," Parallel Computing, vol. 36, pp. 241-253, 2010.

[4] J. D. Davis and E. S. Chung, "SpMV: A Memory-Bound Application on the GPU Stuck Between a Rock and a Hard Place," Microsoft Research Silicon Valley, Technical Report14 September 2012 2012.

[5] M. Terzer and J. Stelling, "Accelerating the Computation of Elementary Modes Using Pattern Trees," in Algorithms in Bioinformatics. vol. 4175, P. Bücher and B. Moret, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 333-343.

[6] D. Jevremović, C. T. Trinh, F. Srienc, C. P. Sosa, and D. Boley, "Parallelization of Nullspace Algorithm for the computation of metabolic pathways," Parallel Computing, vol. 37, pp. 261-278, 2011.

Using In-Memory Computing for Proactive Cloud Operations Future SOC Lab Report April 2013

Eyk Kny, Felix Salfner, Marcus Krug
SAP Innovation Center, Potsdam

Norman Höfler, Peter Tröger
Hasso-Plattner-Institut, Potsdam

Abstract

The operation of a complex IT landscape, such as a cloud computing data center, is a true operator challenge. An ever-growing number of servers, the heterogeneity of software, the necessary elastic load handling, energy consumption and other non-functional aspects have to be taken into account – continuously and in an adaptive fashion.
We introduce the concept of proactive cloud operations, were preventive maintenance activities (such as load migration) are automatically triggered when some part of the system is about to enter an erroneous state. The trigger is generated from a automated root cause analysis that relies on the in-memory computing of anomaly signals and structure-of-influence graphs for the different entities in the monitored IT landscape. We have implemented a prototype of proactive cloud operation with HANA technology in the FutureSOC lab. Our experiments show that an anomaly signal approach for proactive monitoring can achieve the necessary level of accuracy with real-world data.

1 Introduction

Cloud computing is currently one of the predominant trends in IT industry and research. The new paradigm changes the way how IT companies operate businesses as well as how end-users (private and corporate) perceive software and IT. Cloud computing moves the burden of IT infrastructure management and operation away from users to specialized providers that can guarantee fast, reliable, and secure operation of the provided service.
User expectations and scalability aspects turn the management of cloud computing infrastructures into a true challenge. An ever-growing number of servers, the heterogeneity of software, multiple interacting mechanisms to elastically react on changing load, the consideration of energy consumption and other non-functional aspects have to be taken into account. This requires intelligent management tools with a high level of automation, in order to deliver the guaranteed service level to the user while still making money.

Typical management tools rely on the availability of a continuous stream of monitoring data from the IT landscape. One of the key features in such monitoring is to aggregate low-level monitoring data to critical events being presented to the operation personnel. To achieve this goal, current monitoring systems rely on temporal correlation, spatial correlation, and infrastructure-centric rule-based analysis of the low-level data. These techniques usually have a snapshot-alike view on the system and therefore cannot detect problems that evolve in the system over a longer period of time.
In our project, we explore new approaches for computing correlations between monitoring signals, in order to identify the spreading of problems within a system. This report summarizes the second project phase and documents our results so far. In summary we have ...

- ... optimized the monitoring and computation infrastructure for anomaly detection, which was developed in the first phase.
- ... optimized the computation of signal correlation using SAP's in-memory database HANA.
- ... built a web-based front-end for interaction with structure-of-influence graphs stored in HANA.
- ... developed a new graphical representation for the correlation of hierarchical monitoring signals.
- ... investigated ways how to use structure-of-influence graphs for the prediction of upcoming failures.
- ... developed a mock version of the HANA-based software for quick algorithm comparison and signal generation testing.

2 Approach

The basic approach being applied since the first project phase is the algorithm by Oliner et al. [13]. It is based on the notion of anomaly signals, which are real values between zero and one that indicate how much a measurement signal deviates from "normal behavior".

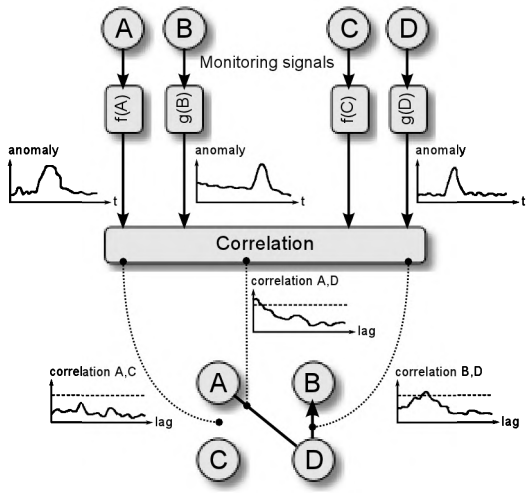


Figure 1: Creation of the Structure-Of-Influence Graph (SIG).

Anomaly signals are obtained from monitoring events by applying a transformation function that encodes local domain knowledge.

By computing the correlation between any pair of anomaly signals, the "spreading" of anomalies across the system can be quantified. More precisely, if there is a high correlation between the anomaly signals of two components, we can speculate that the abnormal states of both components are related. The time lag at which the correlation reaches its maximum can be used to determine the temporal interdependency between the two components, i.e., to identify the source (or initiator), and the target of the problem suffering from the anomaly of the source component. For further analysis, the interdependencies can be visualized in a *structure of influence graph (SIG)*. The nodes of the SIG are monitoring signals then. An edge is added to the graph if there is a significant correlation between the two signals. The edge is directed if the time lag of maximum correlation is larger than some threshold (see Figure 1).

The primary use case for SIGs is the root cause analysis for any kind of abnormal behavior spreading. Given a SIG, the initiating first anomaly can be detected and the component can undergo further manual inspection.

3 Applying In-Memory Computing

Investigating the root cause of a problem that occurred in a large-scale data center environment requires to analyze massive amounts of monitoring data, in order to compare the system behavior of the current faulty case with the "normal case". The operator must be enabled to either quickly drill down at specific points in time, or to get an overview on higher levels of infrastructure granularity. These requirements make proactive root cause analysis a very similar problem to business data exploration, which in itself makes in-memory database technology a key enabler for our project. We implemented the correlation algorithm with on-the-fly data aggregation on SAP's HANA in-memory database, which means that all computationally complex and data-intensive operations have been implemented in SAP's computational database language *SQLScript*.

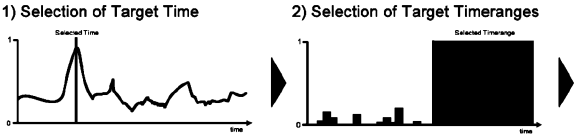


Figure 2: Time Frame Selection

Figure 2 shows how the administrator starts with a selection of some suspicious time frames in the overall data center monitoring interval.

Based on this time frame selection, the user can now chose the specific components ("entities") to be investigated (see Figure 3). This doesn't has to be done on one detail level, administrators can instead chose one granularity level, i.e. either racks, their contained servers, or the components ("entities") contained in these servers. Another set of options allows the filtering of anomaly signal values based on their severity.

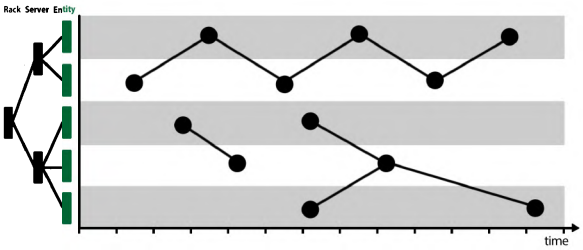


Figure 4: Visualization of Anomaly Spreading

Our implementation generates a on-the-fly representation of the anomaly data in this time frame, as shown in Figure 4. Since each entity (hard disk, processor, memory, ...) generates its own anomaly signal, we have one SIG node per entity. A time-delayed correlation of two anomaly signals is therefore shown as two connected dots. The X axis provides an indication of the anomaly spreading delay, which gives further information for the operator.

Our experimental investigation with customer data from the TACC HPC cluster shows satisfying true positive rates for the predictor, which makes the HANA-based prototype an interesting candidate for further exploitation with different classes of data.

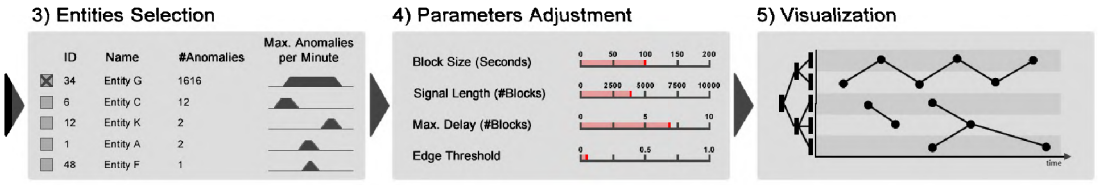


Figure 3: Anomaly Data Filtering

4 Mock Analysis Engine

In order to simplify the research on anomaly signal algorithms, we developed a new framework that allows the flexible testing of different analysis methods based on log file event traces.

The framework realizes the different steps as set of scalable, easy to modify Python mapping scripts. The goal here is to make it as fast and easy as possible to test new and improved future methods. To achieve this, the framework is built upon a plugin concept, which makes it very easy to integrate new log files, anomaly generation and analysis method scripts (see Figure 5). All test runs of this framework were performed with real-world data from the Computer Failure Data Repository (CFDR) on the FutureSOC RX600 resources.

The first step is a preprocessing step, in which a given set of log files is parsed. During this process, the monitoring entries are translated into a unified event data schema. Log parsing and analysis is parallelized under consideration of I/O issues. The user only has to specify the map function that converts a log line into the framework data model. The framework takes care of reading the log files in parallel, and writing the results to a local database.

The next step is the anomaly signal generation process, based on the extracted events. This step requires the implementation of the anomaly signal generator(s) by the researcher. Our framework allows running multiple signal generators at the same time, each generator can also run with several different configurations. The framework takes care of all the necessary routine work, such as reading the database with the preprocessed log entries into shared memory.

The last step is to analyze the generated anomaly signals and to generate the SIG. Again, the researcher has only to provide a mapping function here, while the framework is managing the data-parallel execution of the necessary computations.

Once the generation is finished, the result is analyzed and visualized so that the user can evaluate the quality of the anomaly signal generator, get new insights and improve it. The generation and analyzing steps are part of an iterative process and can therefore repeated several times.

5 Conclusions and Next Steps

Our project brings a new correlation-based monitoring data analysis method to in-memory database technology. Our proven prototype running on HANA in the FutureSOC lab shows first solid results and can be used by administration people through novel data representation approaches.

As part of the future work, we investigate how to fuse multiple, independent anomaly signals into one meta-signal. This allows for combining the advantages of different anomaly signal generation approaches into one data set.

A second major focus of the future work is the reduction of computational complexity using advanced preprocessing techniques in a new custom tool chain. We intend to make structure-of-influence graphs easier to consume by exploring new graphical representations in the near future. Furthermore, future work will focus on the exploration of simplification techniques as well as the application in new domains such as business data.

References

[1] M. Andrecut. Parallel GPU Implementation of Iterative PCA Algorithms. *Journal of Computational Biology*, 16:1593–1599, November 2009.

[2] Joshua Bowden. Application of the OpenCL API for Implementation of the NIPALS Algorithm for Principal Component Analysis of Large Data Sets. In *2010 Sixth IEEE International Conference on e-Science Workshops*, pages 25–30, Washington, DC, USA, 2010. IEEE Computer Society.

[3] Eric Brewer. Lessons from giant-scale services. *Internet Computing*, 5:46–55, July 2001.

[4] Chun-An Chen and Sun-Yuan Hsieh. (t,k)-Diagnosis for Component-Composition Graphs under the MM* Model. *IEEE Transactions on Computers*, 60:1704–1717, December 2011.

[5] Haifeng Chen, Guofei Jiang, Cristian Ungureanu, and Kenji Yoshihira. Failure detection and localization in component based systems by online tracking. In *eleventh ACM SIGKDD international conference on Knowledge discovery*

in data mining, pages 750–755, New York, NY, USA, 2005. ACM.

[6] Matthias Dehmer, Frank Emmert-Streib, and Jürgen Kilian. A similarity measure for graphs with low computational complexity. *Applied Mathematics and Computation*, 182:447–459, November 2006.

[7] Danny Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *Transactions on Visualization and Computer Graphics*, 12:741–748, 2006.

[8] Jon Maeng and Miroslaw Malek. A Comparison Method for Self-Diagnosis of Multiprocessor Systems. Technical report, 1981.

[9] M. Malek and J. Maeng. Partitioning of Large Multicomputer Systems for Efficient Fault Diagnosis. In *Digest of Papers of the 12th Fault-Tolerant Computing Symposium*, pages 341–348, New York, 1982. IEEE.

[10] Adam Oliner. *Using Influence to Understand Complex Systems*. PhD thesis, Stanford University, September 2011.

[11] Adam Oliner and Alex Aiken. A query language for understanding component interactions in production systems. In *24th ACM International Conference on Supercomputing*, pages 201–210. ACM, 2010.

[12] Adam Oliner and Alex Aiken. Online Detection of Multi-Component Interactions in Production Systems. In *Dependable Systems and Networks*, pages 49–60. IEEE, 2011.

[13] Adam J. Oliner, Ashutosh V. Kulkarni, and Alex Aiken. Using Correlated Surprise to Infer Shared Influence. In *Dependable Systems and Networks*, pages 191–200. IEEE Computer Society, 2010.

[14] Panagiotis Papadimitriou, Ali Dasdan, and Hector Garcia-Molina. Web Graph Similarity for Anomaly Detection. Technical Report 2008-1, January 2008.

[15] Spiros Papadimitriou, Jimeng Sun, and Christos Faloutsos. Streaming pattern discovery in multiple time-series. In *31st international conference on Very large data bases*, pages 697–708. VLDB Endowment, 2005.

[16] Yasushi Sakurai, Christos Faloutsos, and Spiros Papadimitriou. Fast Discovery of Group Lag Correlations in Streams. *ACM Trans. Knowl. Discov. Data*, 5, December 2010.

[17] Yasushi Sakurai, Spiros Papadimitriou, and Christos Faloutsos. BRAID: stream mining through group lag correlations. In *2005 ACM SIGMOD international conference on Management of data*, pages 599–610, New York, NY, USA, 2005. ACM.

[18] Lindsay Smith. A tutorial on Principal Components Analysis. 2002.

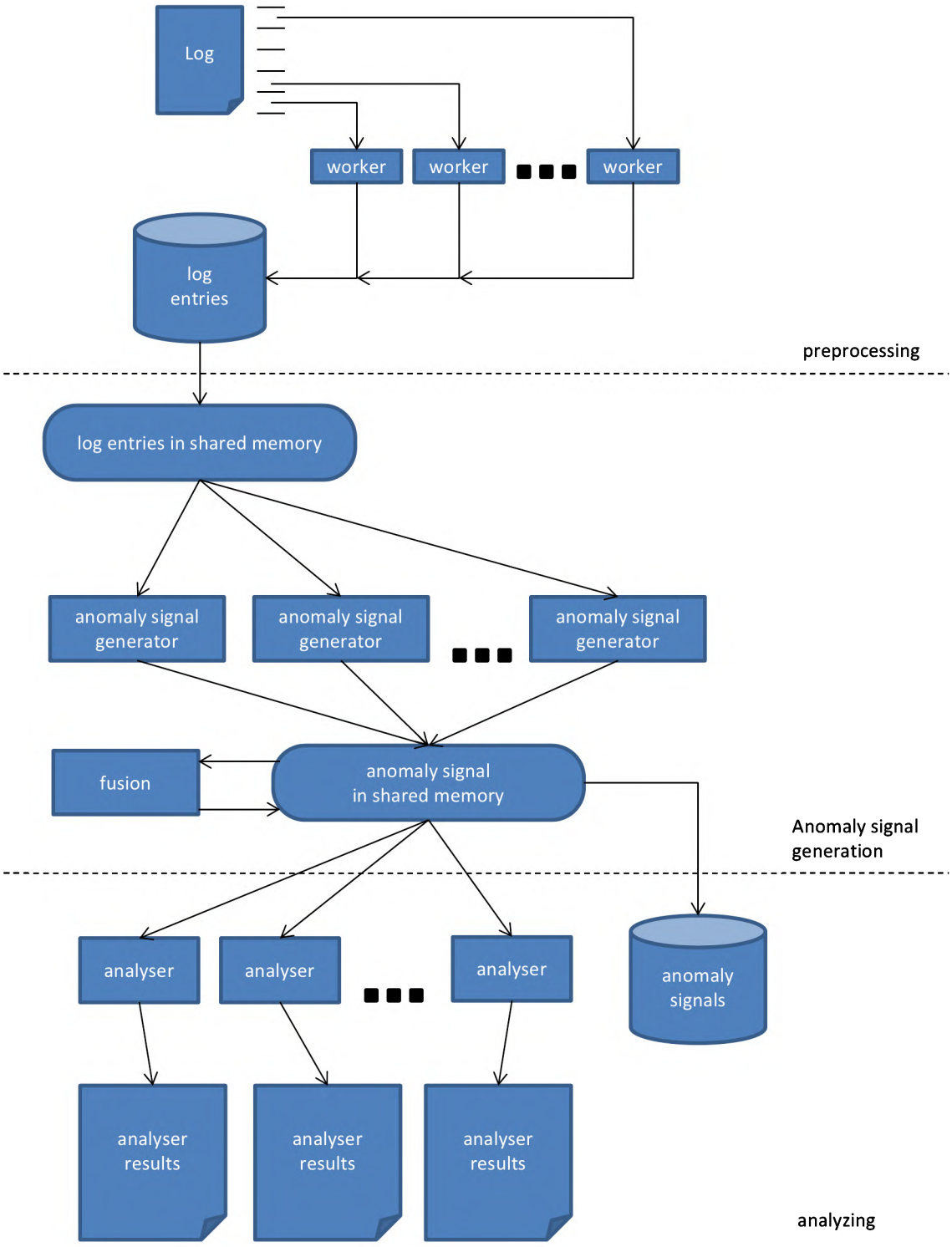


Figure 5. Architecture of the Mock Analysis Engine

Next Generation Sequencing: From Computational Challenges to Biological Insight

Cornelius Fischer
cfischer@molgen.mpg.de

Annabell Witzke
witzke@molgen.mpg.de

Sascha Sauer
Nutrigenomics and Gene Regulation
Otto-Warburg Laboratory
Max Planck Institute for Molecular Genetics, Ihnestr. 63-73, 14195 Berlin, Germany
sauer@molgen.mpg.de

Abstract

Next generation sequencing (NGS) is changing modern biology. One of the major bottlenecks in NGS applications consists of the computational analysis of experimental data. Using the Future SOC Lab resources we established and applied a computational pipeline for the analysis of sequencing data. We found that the provided resources worked very robust and fast to analyse large data sets derived from nucleic acids samples. This enabled us to move rapidly from raw NGS data to gaining initial biological insights.

1 Project idea

Our genome contains thousands of genes. The genes may be active or inactive. In an activated state the DNA of a gene is transcribed into messenger RNA (mRNA). In the subsequent process of translation the information encoded in the mRNA is used to generate proteins. Thus, the entirety of proteins in a cell determines how cells function. Therefore, investigating which genes are in an active or in an inactive state provides us with crucial information on how cells respond to environmental stimuli. Sequencing of mRNA molecules (RNA-seq) using NGS represents a powerful tool to obtain quantitative measurements of gene activity on a genome-wide scale. In the laboratory we exposed cultured cells to stress-inducing substances and took samples of cells at specific time points. The mRNA obtained from these samples was subsequently sequenced and the analysis of sequence information was carried out using the Future SOC Lab resources. This approach aimed to provide an initial framework for understanding and investigating cellular stress related transcriptional changes involved in metabolic and inammatory signal integration.

2 Used Future SOC Lab Resources

We used a Hewlett Packard DL980 G7 server that was equipped with eight 8-core Intel Xeon X6550 processors and 2048 GB of RAM running Ubuntu Server 12.04 LTS. This powerful system was perfectly suited for our approach. The only disadvantage regarding flexibility was the restriction to infrequent user time slots.

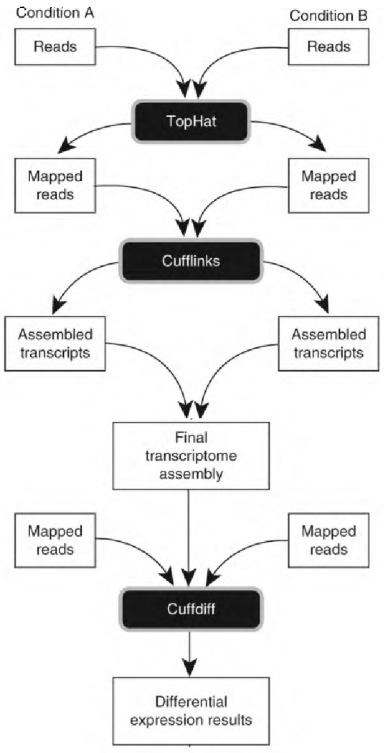


Figure 1: Overview of the used pipeline for computational analysis of RNA-seq data. Figure is modified from [2].

3 Methods and tools

To analyse RNA-seq data we essentially used the pipeline as described [2] and depicted in Figure 1. We used TopHat [1] to align sequence reads to the human reference genome February 2009 assembly (GRCh37/hg19). Then aligned read files were processed using Cufflinks [3] to assemble transcripts and to call expression values of human genes. Cuffdiff was applied to detect significantly differentially expressed genes. Abundances of transcripts were upper-quartile normalized and corrected for sequence bias.

4 Findings

The Future SOC Lab resources helped us to rapidly analyse data generated by NGS. Using the multi-core architecture we systematically tested several settings for sequencing tag mapping, filtering and gene expression estimation. Initially, we used the established pipeline to generate genome-wide density maps of RNA-seq reads (Figure 2). These maps enabled us to investigate gene expression changes of selected genes at different time points. For example, profiles in Figure 2 show that some active genes are constantly expressed at all different time points (Figure 2, *), whereas a flanking gene was activated only at time point 1 and time point 2 (Figure 2, **). We discovered that many hundreds of genes are differentially regulated under the experimental conditions. Once we have narrowed down our candidate genes to smaller modules (Figure 3), we will further investigate the significances of these genes by functional experimental approaches. This will help us to understand the interplay of transcriptional pathways in the context of cellular stress response.

5 Next steps

Further participation in the HPI Future SOC Lab would allow us to break new ground in data analysis and to extract biologically meaningful information from the generated NGS data. In particular, the provided capacities will enable us to integrate publicly available data from other research groups with our own data in future work. Therefore, the already established computational pipeline will be essential to identify and understand the mechanism of metabolic pathologies and inflammatory processes.

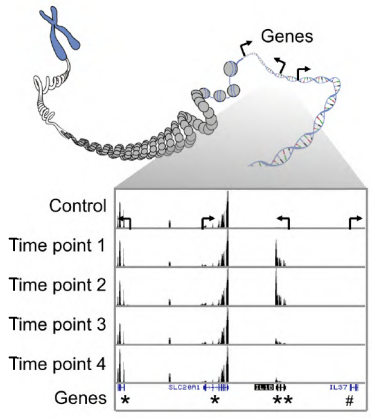


Figure 2: Density profiles generated from RNA-seq data help to investigate gene expression changes on a genome-wide scale (* constitutively active, ** conditionally activated, # inactive).

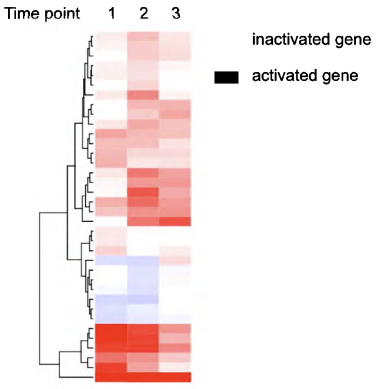


Figure 3: Heat map generated from RNA-seq data reflecting g ene e xpression of selected gene modules.

References

[1] C. Trapnell, L. Pachter, and S. L. Salzberg. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics (Oxford, England)*, 25(9):1105–1111, May 2009. PMID: 19289445.

[2] C. Trapnell, A. Roberts, L. Goff, G. Pertea, D. Kim, D. R. Kelley, H. Pimentel, S. L. Salzberg, J. L. Rinn, and L. Pachter. Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and cufflinks. *Nature protocols*, 7(3):562–578, Mar. 2012. PMID: 22383036.

[3] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature biotechnology*, 28(5):511–515, May 2010. PMID: 20436464.

Large-scale Schema Extraction and Analysis of Distributed Graph Data

Thomas Gottron, Malte Knauf
Institute for Web Science and Technologies
Universität Koblenz–Landau
56070 Koblenz, Germany
{gottron,mknauf}@uni-koblenz.de

Ansgar Scherp
Research Group on Data and Web Science
Universität Mannheim
68131 Mannheim, Germany
ansgar@informatik.uni-mannheim.de

Abstract

The Linked Open Data (LOD) cloud is a constantly growing distributed resource for interlinked semantic information. Schema-level indices such as SchemEX allow for a concise description of schema-related patterns on the LOD cloud. Extraction and construction of a SchemEX index can be implemented efficiently using a stream-based approach. In this project, we have implemented a large-scale information theoretic analysis of Linked Data based on SchemEX indices. The results provide interesting insights into correlation and redundancy of type and property information associated with entities modelled on the LOD cloud. The findings have an impact on the design of schema-level indices as well as applications which can benefit from the availability of patterns in schema information encoded in Linked Data.

1 Introduction

The Linked Open Data (LOD) movement fosters Tim-Berner Lee's vision of a web where data is published from various sources and interlinked to a huge knowledge graph. Given the intended generality of the concept of a web of data, there is no constraint on the schema of the data. Technology-wise, this is implemented by using RDF as flexible format for modelling data. In RDF the knowledge graph is represented by triples, which express a typed connection between two entities, link an entity to literal values or attach type information to an entity.

Schema information about entities is given in Linked Data in a twofold way: explicitly by providing the type of a resource and implicitly via the definition of its properties. These two manifestations of schema information are to a certain extent redundant, i. e., certain resource types entail typical properties and certain properties occur mainly in the context of particular types. For instance, we would expect a resource of type foaf:Person to have the properties foaf:name or foaf:age. Likewise, we can assume a resource with the property skos:prefLabel to be of type skos:Concept. An intriguing question is to which degree explicit and implicit schema information is correlated, i. e., to which extent the use of RDF types and properties appear together to describe resources. This leads to the overall question of how far the explicit schema information provided by RDF types coincides with the implicit schema information of the properties used in the LOD cloud and how consistent are the observed patterns and redundancies.

A fundamental prerequisite to answer this question is the availability of a reliable schema extracted from the LOD cloud that takes into account both explicit and implicit schema information. The SchemEX approach [7, 6, 4] is capable of extracting such a schema for very large amounts of RDF triples in an efficient manner and with high accuracy [3]. SchemEX produces a schema-level index, in which meta data about linked data resources is stored under the schema pattern they comply with. Thus, SchemEX provides the ideal technical basis to estimate the frequency of schema patterns on the LOD cloud. These frequencies, in turn, can be used to model a joint distribution over explicit and implicit schema information for a deeper information theoretic analysis

In this report, we briefly cover the idea of our information theoretic investigations in Section 2. We discuss the implementation using the resources of the HPI Future SOC Lab in Section 3 and discuss our findings in Section 4. We conclude with a brief summary on the potential impact of our analysis on applications and a look at future work. A more detailed discussion of metrics, experimental design and results is accepted as full paper at the Extended Semantic Web Conference 2013 [2].

2 Project Idea

The aim of this project was to model and analyse a joint distribution of explicit and implicit schema information over Linked Data. The joint distribution had to be estimated from real data. The estimates were computed on the basis of a SchemEX index, which provides schema information as well as frequency information about schema patterns in the payload meta

data stored in the index.
To be more specific, we estimated the probability of an entity on the LOD cloud to be described by a set of types t and a set of properties r by:

$$\hat{p}(t,r) = \frac{|x \in \text{SchemEX}(t,r)|}{N}$$

where N is the overall number of entities described in the analysed data set and $x \in \text{SchemEX}(t,r)$ denotes the set of entities indexed in SchemEX with the specified set of types and properties.
To model the joint distribution of type sets and property sets, we introduce two random variables T and R over the power sets $TS = \mathcal{P}(\textit{Classes})$ and $PS = \mathcal{P}(\textit{Properties})$ of all possible class type and property combinations, respectively. The joint distribution of this two random variables is then defined as:

$$P(T = t, R = r) = p(t,r)$$

On the basis of this joint distribution, we defined several metrics for answering our questions about information content and redundancy on the LOD cloud:

Normalized Entropy of the Marginal Distributions. To answer the question of how much information is encoded in the type sets or the property set of an entity, we consider the entropy [10] of the two marginal distributions of $P(T,R)$:

$$\begin{aligned} H(T) &= -\sum_{t \in TS} P(T=t) \cdot \lg\left(P(T=t)\right) \\ H(R) &= -\sum_{r \in PS} P(R=r) \cdot \lg\left(P(R=r)\right) \end{aligned}$$

For reasons of comparability, we further normalize these values making use of the maximal entropy:

$$\begin{aligned} H_0(T) &= \frac{H(T)}{\lg(|TS|)} \\ H_0(R) &= \frac{H(R)}{\lg(|PS|)} \end{aligned}$$

Expected Conditional Entropy. In order to respond to the question of how much information is still contained in the properties, once we know the types of an entity (or vice versa) implies a conditional probability and, thus, a conditional entropy. To further aggregate values over all possible conditions, we use the expected conditional entropy. The conditional entropy of the properties, given the types is defined as:

$$\begin{aligned} H(R|T) &= \sum_{t \in TS} P(T=t) H(R|T=t) \\ &= -\sum_{t \in TS} \sum_{r \in PS} p(t,r) \lg\left(\frac{p(t,r)}{P(T=t)}\right) \end{aligned}$$

The conditional entropy of the types, given the properties is defined analogously.

Normalized Mutual Information. To finally answer the question of how far explicit and implicit schema information is redundant, we make use of mutual information (MI) [1]. MI is a metric to capture the joint information conveyed by two random variables and, thereby, to which degree they can explain each other. The MI of explicit and implicit schema information of the LOD cloud is defined as:

$$I(T,R) = \sum_{r \in PS} \sum_{t \in TS} p(t,r) \lg \frac{p(t,r)}{P(T=t) \cdot P(R=r)}$$

A normalization of MI to the interval $[-1,1]$ is given in [11] and involves the entropy of the marginal distributions of T and R. This normalization serves as a direct measure for redundancy and is defined as:

$$I_0(T,R) = \frac{I(T,R)}{\min\left(H(T), H(R)\right)}$$

3 Implementation

For our empirical analysis, we use the different segments of the data set provided for the Billion Triple Challenge (BTC) 2012. The BTC data set has been crawled from the web in a typical web spider fashion and contains about 1.44 billion triples. It is divided into five segments according to the set of URLs used as seed for the crawling process: Rest, Datahub, DBPedia, Freebase and Timbl. Details about the different parts and the crawling strategies used for collecting the data are described on the BTC 2012 data set's website[1].
We consider especially the larger data segments particularly useful as they span different aspects of the LOD cloud. With *Datahub*, we have got a sample of several publicly available linked RDF data sources registered in a central location. *DBpedia* is interesting as it is one of the central and most connected resources in the LOD cloud extracted from the collaboratively curated Wikipedia. *Freebase*, instead, is also a collaborative knowledge base, but here the users directly operate on the structural data. The *Timbl* data set is a crawl starting at the FOAF profile of Tim Berners-Lee (thus, the name). Hence, it provides a snapshot from yet a different part of the LOD cloud, namely starting at small, manually maintained RDF files.
We used resources from the HPI cluster to be able to perform the memory intensive computation of the metrics in Section 2. We mainly made use of the Hewlett

[1] BTC 2012 data set: http://km.aifb.kit.edu/projects/btc-2012/

Packard DL980 G7 machines with 1 and 2 TB of RAM. While we used the machines also for the extraction of the schema-level index structures, the main motivation for making use of HPI lab resources was the information theoretic analysis. Estimating a joint distribution of type sets and property sets requires managing a relatively fine grained decomposition of the data sets. Essential functionalities for the computation of the information theoretic measures are a fast and flexible aggregation of the elements of this decomposition, which can easily be implemented in main memory.

4 Results

In this section we summarize the findings of our analysis. Table 1 gives an overview of the computed metrics on the five segments of the BTC 2012 data set. We will now go into the details of the single metrics.

Entropy in Type and Property Sets. We can observe the tendency that the property sets convey more information than type sets. This can be observed in the higher values of the normalized entropies. For instance, the normalized marginal entropy of the property sets has a value of 0.324 on the *DBpedia* data set, while the normalized marginal entropy of the type sets is 0.093. This observation provides a hint that on *DBpedia* the distribution into type sets is far more skewed than the distribution of property sets. Similar observations can be made for the data sets *Rest*, *Freebase* and *Timbl* as well, though to a lower extent. An exception is the *Datahub* data set, where the distribution of entities in type sets and property sets seems more comparable.

Expected Conditional Entropies. Looking at the expected conditional entropies reveals some interesting insights. Recall that the aggregation we chose for the conditional entropy provides us with the expected entropy, given a certain type set or property set. We can see in Table 1 that the entropy given a property set tends to be far lower than the one when given a type set. In conclusion: knowing the properties of an entity in these cases already tells us a lot about the entity, as the entropy of the conditional distribution can be expected to be quite low. On the contrary, when knowing the type of an entity, the entropy of the distribution of the property sets can be expected to be still relatively high (when compared to the entropy of the marginal distribution). We looked at the data more closely to investigate how often a given type set is already a clear indicator for the set of properties (and vice versa). The most extreme case is the *Freebase* data set, where for 80.89% of all entities it is sufficient to know the set of properties in order to conclude the set of types associated with this entity. Knowing, instead, the types of an entity conveys less information: only in 2.05% of the cases this is sufficient to precisely predict the set of properties of an entity. Again, and with the exception of *Datahub*, the other data sets exhibit a similar trend.

Mutual Information. Finally, the value of the normalized MI gives us insights on how much one information (either properties or types) explains the respective other. Also here, we observe a quite wide range from 0.635 on *DBpedia* to 0.881 on *Rest*. Accordingly, extracting only type or only property information from LOD can already explain a quite large share of the contained information. However, given our observations a significant part of the schema information is encoded also in the respective other part. The degree of this additional information depends on the part of the LOD cloud considered. As a rule of thumb, we hypothesise that collaborative approaches without a guideline for a schema (such as *DBpedia*) tend to be less redundant than data with a narrow domain (*Timbl*) or some weak schema structure (*Freebase*).

Discussion of the Results. The observations on the large data sets provide us with insights into the form and structure of schema information on the LOD cloud. First of all, the distribution of type sets and property sets tend to have a relatively high normalized entropy. We can conclude that the structure of the data is not dominated by a few combinations of types or properties. Accordingly for the extraction of schema information, we cannot reduce the schema to a small and fixed structure but need to consider a wider variety of type and property information. Otherwise the schema would loose too much information.

A second observation is the dependency between types and properties. The conditional entropy reveals that the properties of an entity usually tell much more about its type than the other way around. This observation is interesting for various applications. For instance, suggesting a data engineer the types of an entity based on the already modelled properties seems quite promising [8]. We assume that this observation can also be seen as an evidence that property information on the LOD cloud actually considers implicit or explicit agreements about the domain and range of the according property. However, this observation is not valid for the entire LOD cloud. Depending on the concrete setting and use case, a specific analysis might need to be run.

Finally, the observed MI values underline the variance of schema information in the LOD cloud. Ranges from 63.5% to 88.1% redundancy between the type sets and property sets have been observed. Thus, approaches building a schema only over one of these two types of schema information run at the risk of a significant loss of information.

Table 1. Result obtained for the various information theoretic measures on the five segments of the BTC 2012 data set.

Data set		Rest	Datahub	DBpedia	Freebase	Timbl
Number of Triples		22.3M	910.1M	198.1M	101.2M	204.8M
Type sets	$\|TS\|$	793	28,924	1,026,272	69,732	4,139
Property sets	$\|PS\|$	7,522	14,712	391,170	162,023	9,619
Entropy of type sets	$H(T)$	2.428	3.904	1.856	2.037	2.568
Normalized marginal entropy of type sets	$H_0(T)$	0.252	0.263	0.093	0.127	0.214
Entropy of property sets	$H(R)$	4.708	3.460	6.027	2.868	3.646
Normalized entropy of property sets	$H_0(R)$	0.366	0.250	0.324	0.166	0.276
Expected conditional entropy, given properties	$H(T\|R)$	0.289	1.319	0.688	0.286	0.386
Expected conditional entropy, given types	$H(R\|T)$	2.568	0.876	4.856	1.117	1.464
Joint entropy	$H(T,R)$	4.997	4.779	6.723	3.154	4.032
Mutual Information	$I(T,R)$	2.140	2.585	1.178	1.751	2.182
Normalized Mutual Information	$I_0(T,R)$	0.881	0.747	0.635	0.860	0.850

5 Conclusion

In this project, we have developed and applied a method and metrics for conducting in depth analysis of schema information on Linked Open Data. In particular, we have addressed the question of dependencies between the types of entities and their properties. Based on the five segments of the BTC 2012 data set we have computed various entropy metrics as well as mutual information. In conclusion, we observe a trend of a reasonably high redundancy between the types and properties attached to entities. As more detailed conclusion, we can derive that the properties of an entity are relatively indicative for the type of the entity. In the other direction, the indication is less strong. However, this observation is nor valid for all sources on the LOD cloud. In conclusion, if the application and data domain is not known, it is necessary to capture both: explicit and implicit schema information. For a detailed analysis of the results, please refer to our publication at the Extended Semantic Web Conference 2013 [2].

The findings will influence our works in several fields. In the context of search systems for LOD, we will consider options for compressing a SchemEX index structure by avoiding redundant information [5]. The detection of stable schema patterns is useful also in other application scenarios. In order to be able to construct an API for accessing LOD entities [9], it is necessary to identify such stable patterns. As mentioned above, also data engineers can benefit from redundancy [8]. Repetitive use of the same vocabulary indicates trends and common practices of how to combine types and properties to describe data well. Therefore, data engineering tools can support the engineer in the best choice of vocabulary by following typical patterns.

As future work, we plan to deepen these insights and incorporate the obtained deeper understanding into various applications. Therefore, we will look into the details of the conditional distributions for given type sets and property sets. In this way, we might identify which sets of types and properties allow for highly precise predictions of the respective other schema information.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 257859, ROBUST.

References

[1] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[2] T. Gottron, M. Knauf, S. Scheglmann, and A. Scherp. A Systematic Investigation of Explicit and Implicit Schema Information on the Linked Open Data Cloud. In *ESWC'13: Proceedings of the 10th Extended Semantic Web Conference*, 2013. to appear.

[3] T. Gottron and R. Pickhardt. A Detailed Analysis of the Quality of Stream-based Schema Construction on Linked Open Data. In *CSWS'12: Proceedings of the Chinese Semantic Web Symposium*, 2012.

[4] T. Gottron, A. Scherp, B. Krayer, and A. Peters. Get the Google Feeling: Supporting Users in Finding Relevant Sources of Linked Open Data at Web-Scale. In

Semantic Web Challenge, Submission to the Billion Triple Track, 2012.

[5] T. Gottron, A. Scherp, B. Krayer, and A. Peters. LODatio: Using a Schema-Based Index to Support Users in Finding Relevant Sources of Linked Data. In *K-CAP'13: Proceedings of the Conference on Knowledge Capture*, 2013. to appear.

[6] M. Konrath, T. Gottron, and A. Scherp. Schemex – web-scale indexed schema extraction of linked open data. In *Semantic Web Challenge, Submission to the Billion Triple Track*, 2011.

[7] M. Konrath, T. Gottron, S. Staab, and A. Scherp. SchemEX—Efficient Construction of a Data Catalogue by Stream-based Indexing of Linked Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 16(5):52 – 58, 2012. The Semantic Web Challenge 2011.

[8] J. Schaible, T. Gottron, S. Scheglmann, and A. Scherp. LOVER: Support for Modeling Data Using Linked Open Vocabularies. In *LWDM'13: 3rd International Workshop on Linked Web Data Management*, pages 89–92, 2013.

[9] S. Scheglmann, G. Gröner, S. Staab, and R. Lämmel. Incompleteness-aware programming with rdf data. In E. Viegas, K. Breitman, and J. Bishop, editors, *Proceedings of the 2013 Workshop on Data Driven Functional Programming, DDFP 2013, Rome, Italy, January 22, 2013*, pages 11–14. ACM, 2013.

[10] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 and 623–656, July and October 1948.

[11] Y. Y. Yao. Information-theoretic measures for knowledge discovery and data mining. In J. K. Karmeshu, editor, *Entropy Measures, Maximum Entropy Principle and Emerging Applications*, pages 115–136. Springer, Berlin, 2003.

Storage Class Memory Evaluation for SAP HANA

Ahmadshah Waizy
ahmadshah.waizy@ts.fujitsu.com
Dieter Kasper
dieter.kasper@ts.fujitsu.com
Konrad Büker
konrad.bueker@ts.fujitsu.com
Karsten Beins
karsten.beins@ts.fujitsu.com
Jürgen Schrage
juergen.schrage@ts.fujitsu.com

Fujitsu Technology Solutions GmbH
Heinz-Nixdorf-Ring 1
33106 Paderborn, Germany

Bernhard Höppner
bernhard.hoeppner@sap.com
Felix Salfner
felix.salfner@sap.com
Henning Schmitz
henning.schmitz@sap.com
Joos-Hendrik Böse
joos-hendrik.boese@sap.com

SAP Innovation Center Potsdam
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany

Abstract

Storage Class Memory (SCM) introduces a new storage technology, which combines word granular access by single CPU instructions with the advantage of being non-volatile and offering a high storage density. It is well suited to increase the memory capabilities of individual servers efficiently to deal with increasing memory demands of applications and can also be used to store data permanently.
SCM technology could potentially be utilized in many software applications. We concentrate on the in-memory computing database SAP HANA, which may particularly benefit from the introduction of SCM technology. Several possible applications of the new technology are analyzed. As SCM prototypes are not yet available physically we have developed a software emulator to simulate the effects of a physical presence of SCM technology on the speed of data access. This simulation is then used as the basis for extensive benchmarking of one field of application. The results of those benchmarks are presented in this document.
This project was executed at the Future SOC Lab of the Hasso-Plattner-Institute in Potsdam (Berlin), Germany. It utilized a Fujitsu RX600 server with 1 TB RAM, which is part of the lab's infrastructure.

1 Introduction

In existing architectures a clear distinction between memory and storage is made when developing software. Hard disk (HDD) or solid-state disk (SSD) based storage offers non-volatile characteristics and high storage density leading to low prices. By contrast, memory technologies like dynamic random access memory (DRAM) offer low latencies and byte granular access. This distinction leads main decision in software architectures. Introducing the SCM technology might be an inflection point for today's paradigms. SCM combines the characteristics of memory and storage, offering low latencies and byte granular access as well as high storage density and being non-volatile.

SAP's in-memory computing appliance SAP HANA introduces a technology that can be used to address problems in the domain of real-time processing of big data [1]. It may profit of the mentioned advantages in multiple ways. In general two different approaches are possible. The non-volatile characteristic of SCM offers new ways to ensure the durability of data whereas low latencies make SCM interesting for in-memory databases in order to work directly on the data utilizing a higher storage density compared to DRAM. Therefore it is particularly interesting to investigate how SAP HANA could benefit from the introduction of SCM and what adoptions might be needed to enable SAP HANA to use SCM.

This report summarizes the findings of the "Storage Class Memory Evaluation for SAP HANA" and documents its results. In summary we have

- Provided a description of the SCM technology characteristics and how it is simulated using an emulator.

- Conducted a capability study to determine some applications in which using SCM might be particularly rewarding.
- Benchmarked the characteristics of SCM by using I/O processing, SAP HANA based event stream processing, and a SAP specific benchmark from the domain of Enterprise Resource Planning applications based on SAP HANA, the so-called SD benchmark.
- Discussed the next steps this project should take as regards enabling SAP HANA to utilize SCM most efficiently.

2 Storage Class Memory Technology

Emerging device technologies including phase change-memory (PCM), spin-torque transfer RAM (STT-RAM) and memristors promise high-speed storage. These technologies collectively are termed storage-class memory (SCM) as data can be accessed through ordinary load/store instructions rather than through I/O requests. Hence, user-mode code can access data directly, so there is no need for the operating system to mediate every access.

The availability of large persistent memory combined with a small lowering of access speed seems to be within reach in the next 3-5 years. This is likely to trigger a new software design for applications, middleware and operating systems. Being prepared for this revolutionary step is the request of this research, with a special view on SAP HANA.

The software architectures as we know them (operating system as well as applications and data bases) are built on a foundation of fast volatile memory and slow persistent storage. The difference in latency between "fast" (memory) and "slow" (storage) lies in the order of magnitude of factor 10^6.

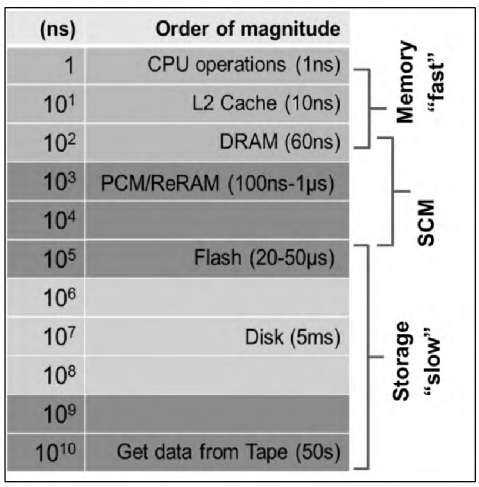


Figure 1: Access characteristics of different types of data storage media

With the introduction of flash memory the gap between "fast" and "slow" became somewhat smaller, see Figure 1. But still, flash memory acts as storage block device, because the granularity is given by physical limitations shown in Figure 2:

- Page_size: Size of the smallest atomic read or write operation (approx. 4k on enterprise Single-Level Cell, SLC and 32k on consumer Multi-Level Cell, MLC).
- Segment_size: Size of the smallest atomic erase operation (approx. 64k on enterprise SLC, 4m on consumer MLC).

Upcoming SCM technologies are technologies which will allow the CPU a more fine granular access by word as it is used today with DDR (Double Data Rate), see mark in Figure 2. This combination of real word granularity access by a single CPU instruction in combination with persistency and a large affordable capacity bigger than DRAM upturns the existing software architecture.

	persistent	CPU Instructions to access data	IO size (Bytes)	Read latency (ns)	Write latency (ns)	EUR / GB (10.2012)
DDR	N	1	1	60	60	8 - 44
SCM-Memory (PCM)	Y	1	1	300	700	unknown
SCM-Storage (PCM)	Y	> 10k	512	~ 5,000	~ 10,000	unknown
Flash SLC, PCIe	Y	> 10k	4096	20,000	50,000	6-8
Flash MLC	Y	> 10k	4096	100,000	250,000	3-4
HDD	Y	> 30k	512	5,000,000	5,000,000	0.30-0.60

Figure 2: Extended memory / storage hierarchy layers considering SCM

The characteristics of this new technology are essentially the fine granular access of data by Byte/Word, the accessibility by a single CPU instruction, the non-volatility, relatively low access times, the higher density than DRAM, and lower cost per bit.

3 Capability Study

Introducing Storage Class Memory as an inflection point for software architectures leads to the need for developing approaches to leverage the technology. Today's solutions make a clear distinction between the usage of non-volatile storage with a rather low performance but high data density and volatile memory with a high performance but low data density. This distinction has an impact on many decisions when designing software. With a view to SAP HANA this explains the usage of main memory for all actively used data in order to provide real time transactional and analytical operations. To ensure the durability of data, non-volatile storage is used to keep savepoints and log files. The following chapters focus on two main topics on how to use SCM in a future database application. In general we divide our attempts by the properties of SCM. At first we make use of the non-volatility characteristic of SCM by suggesting approaches for data persistence. Next we take a closer look onto the high data density and low latencies by introducing use cases for data aging and

a so-called Scale-In idea. We present general strategies on how to address the features of SCM and give first insights into their implementation efforts. We also evaluate those strategies by explaining their benefits and possible drawbacks.

3.1 Storage Class Memory for Data Persistence

The data persistence layer of a database is responsible for assuring that no data is lost if the system crashes and that the last consistent state is restored after the system has been repaired. Data persistence usually consists of two major parts, which are data checkpoints (sometimes also called savepoints or snapshots) and a transaction log that persists all committed changes to the database after the last checkpoint has been saved. In case of a database recovery, the last checkpoint is loaded into the database first. The transaction log is replayed thereafter so that the database is brought back into the last consistent state that was saved before the crash.

The need to save snapshots and transaction logs to persistent storage is a requirement also for in-memory databases. Although the data is kept in main memory, the database must be able to restore the last consistent state in case of, e.g., a power failure. SCM being non-volatile has - at least - the potential to survive power failures. However, writing to secondary storage such as flash disks, and shipping it to some redundant file server still offers a significantly higher level of fault tolerance. Therefore writing persistent data onto local SCM will not be sufficient from a fault-tolerance point of view. Fault-tolerant storage will be discussed in the next section. Nevertheless, SCM offers word-access to the data. We investigated how this could improve the data persistency layer of an in-memory database.

3.1.1 Transactional Logging

SCM offers increased access speed and the ability to support addressing of single data words, which is not offered by today's persistence layers - HDD and flash-based IO devices operate on a block-level only. We investigated whether this specific advantage could improve transactional logging of in-memory-databases significantly.

Access at word granularity level is beneficial primarily for scenarios with random data access patterns, i.e., data is either read or written in arbitrary patterns or data is read or written randomly to arbitrary positions of the device. Unfortunately, the logging layer of data persistency does not show any of these characteristics. Logging usually is a process where data is simply appended to the end of the log. During recovery, data is primarily read sequentially from storage. As SCM offers about equal throughput for large data transfers if compared to contemporary block-oriented storage, we do not see high potential in using SCM to significantly improve transactional logging. Only in the case of a high number of very short transactions SCM could outperform traditional solutions. However, massive changes concerning the persistence layer of an existing database would be necessary to implement a fine granular logging whereas advantages might only be feasible for a small subset of transactions.

3.1.2 Redundancy

The same arguments hold for the redundancy part of the data persistence layer of databases. Writing snapshots primarily forms a sequential data access pattern so that data word access does not provide significant benefit. This is at least true for contemporary database persistency layers, which are optimized for HDD-based storage tiers. To better exploit the performance of SCM efficiently for database persistence layers new data structures and approaches are required.

Some promising approach could be a more or less direct mapping of HANAs in-memory data structures to SCM. A first candidate for such a strategy could be the delta store, which is manipulated by every write operation. Using random word level access of SCM would allow keeping a complete copy of the delta store on SCM storage and recovering using a simple copy of the SCM structure to RAM. Therefore the need of transactional logging for the delta store would vanish and the persistence of the delta store could be achieved by memory copy instructions. Also this results in additional effort in redesigning the persistence layer but the performance increase is theoretically very promising.

3.2 Storage Class Memory for Data Aging and Scale-In

Companies face an enormously increasing amount of data being stored in their systems today. For new retail customers at SAP this could mean an average growth of 40GB per month in the first year [3]. Even when using an in-memory computing database like SAP HANA this fact leads to the problems of

- An increasing demand of memory, which leads to the infeasibility of keeping the entire data in main memory.
- A decreasing performance given by increasing processing times of database operations.

Figure 3 illustrates the correlation of database table fill levels and resulting database query response times. A decreasing performance can especially be observed for increasing fill levels regarding scan and aggregate operations where large datasets need to be processed. In the following we will discuss ideas on how to overcome the mentioned issues. Our main driving force is given by the fact that today's business store data of the last ten years on average, while only 20% are used actively [1].

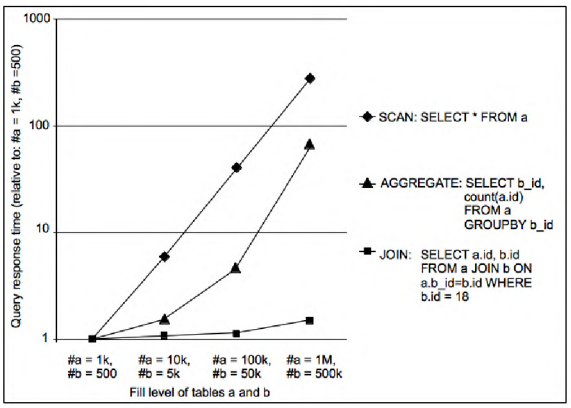


Figure 3: Database table fill levels and resulting query response times [1]

High data density allowing low costs per gigabyte and low read latencies make Storage Class Memory an ideal solution to face the problems described. Traditional cache strategies would not be suitable to utilize the characteristics of SCM as existing technologies flash-based solutions could be used. Therefore we are introducing an approach called data aging and show how it can benefit from using SCM. Data aging in general separates data storage into two types.

- Hot storage refers to main-memory, which is typically used to store data of in-memory database management systems.
- Cold storage offers less performance but higher data density. Today's attempts often use HDD or SSD based storage.

By aging we mean the movement of infrequently used data, which is also named as aged or cold data, from the hot into the cold storage. Cold data is only read and used for analytical processing. Consequently the amount of data in main memory (hot storage) decreases which leads to lower main memory consumption and overall increasing performance for hot data (data in hot storage). By contrast to traditional caching strategies cold data is directly read from cold storage for analytical processing. Therefore an additional effort for copying data between different memory layers becomes unnecessary. Data aging is also referred to as dynamic horizontal partitioning of database tables. Main efforts have to be put into the distinction between hot and cold data. Most data partitioning is defined in an application-specific way. Consequently developers give recommendations on the aging of data by defining associated criteria. Other approaches also take statistic based ideas into account where items are generally separated by their usage. Analyzing read and write access to data helps to induce a ranking as a basis for portioning.

SCM can ideally be used to introduce data aging for an in-memory computing based database management system such as SAP HANA because of low read latencies, which are perfectly situated for analytical processing on cold data. Introducing data aging combined with SCM offers an increasing performance for hot data and provides the ability to efficiently work on cold data without the need for copy operations between hot and cold data as they would be caused by caching strategies. This approach can also referred to as Scale-In. Existing architecture approaches respond to increasing memory requirements of software applications by introducing additional server nodes. This Scale-Out approach increases overall memory and compute power for the applications but also leads to several challenges. By contrast to the Scale-Out approach increasing memory capabilities of single nodes introduced by SCM and data aging make it possible to consider addressing even highest memory requirements with single node architectures. Approaches that aim at addressing big memory requirements by increasing the capabilities of individual nodes can be called Scale-In approaches.

4 Benchmarking

As Storage Class Memory is not available as hardware prototype today, the development of a software-based emulator became necessary to emulate the effects of a physical presence of SCM technology. The emulator aims at providing possible latency characteristics of SCM technology but is based on traditional DRAM. In our approach it is used as a block device in order to make it usable for SAP HANA to persist log files. The emulator is benchmarked by different approaches. In a first attempt the storage benchmark tool Flexible I/O is used to measure the general characteristics. In the later benchmarks SAP HANA is using the emulated SCM to store the transactional logs. Those logs are used in database management systems to record and persist data changes and ensure the durability of data. They need to be written to a non-volatile storage and are crucial to recover the most recent state of the database after a crash by replaying the recorded changes. Transactional logging is one of the bottlenecks for transactional throughput, which depends on the speed of data access. In today's appliances those logs are written to storage technologies like HDDs or SSDs, which offer a comparable high latency. Being non-volatile, offering a high data density and low latencies makes SCM fits the mentioned requirements. However, the significant lower write performance of SCM might be a critical drawback of the technology as the logging process mainly produces write operations and only a few read operations. Addressing SCM as a block device potentially decreases the performance even more. Those possible issues, which are influencing the feasibility of using SCM for SAP HANA's logs, are analyzed. The results of those benchmarks are illustrated and analyzed in the following chapters. We also describe our approach to emulate SCM and the general I/O characteristics of this solution.

4.1 Storage Class Memory Emulator

SCM is a step towards a fundamentally new memory hierarchy with deep implications across the software/hardware interface.

Our emulator is implemented as a block device driver for Linux that simulates the presence of a SCM installed in one of the DIMM slots on the motherboard.

The emulator is implemented as a kernel module for Linux that creates "/dev/scm0" when it is loaded and which acts as a block device with SCM latencies lying on top of the RAM device. Figure 4 shows the implementation concept. When the module is loaded, it benchmarks the performance of the systems RAM and computes the differences between the latencies of the RAM-disk and the simulated SCM.

All read/write operations into the RAM-disk will be emulated with latencies of the SCM.

Since the SCM technology will come with different read/write latency values, the emulator provides a command called "scmadm", which allows to individually configure the latency factors for read and write operations. So the emulator is capable of simulating a SCM with different latency values.

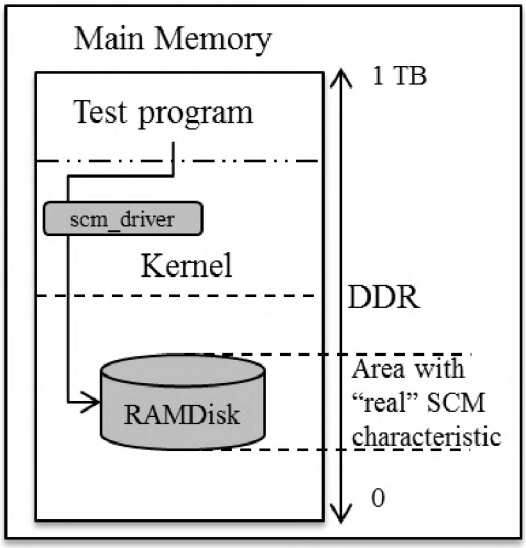


Figure 4: Emulator concept as a RAM-disk backed by a kernel block device driver

The emulator is a Linux RAM-disk driver which adds the wanted latency factors to SCM access. The emulated SCM latencies (delay factors for reading and writing) can be adjusted after the SCM driver is loaded by using the emulator command "scmadm". Figure 5 explains the implemented interfaces.

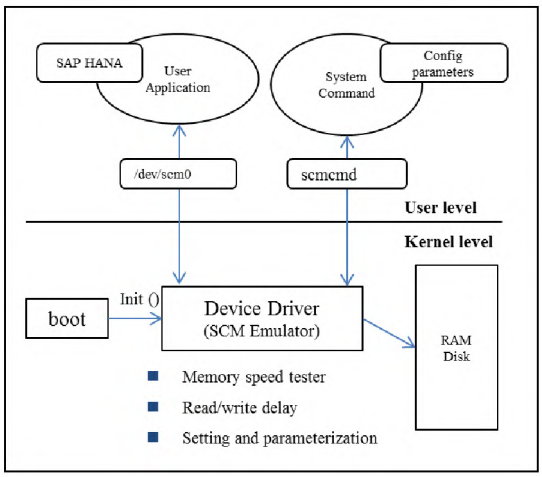


Figure 5: Emulator Interfaces

From a number of published SCM references (e.g. [4] to [11]) we extracted various delay times for writing and for reading operations. Figure 6 lists the latencies being predicted in publications. Our emulated SCM test cases are focusing on *SCM_Ref_1* and *SCM_Ref_2*. (PCRAM = Phase Change Random Access Memory).

Name	Mode	DRAM	PCRAM	ReRAM	STT-RAM
SCM_Ref_1	read	60 ns	**300 ns**	--	--
	write	60 ns	**700 ns**	--	--
SCM_Ref_2	read	60 ns	**300 ns**	--	--
	write	60 ns	**1400 ns**	--	--
Ref_3	read	55 ns	48 ns	100 ns	1.96 ns
	write	55 ns	150 ns	100 ns	7.76 ns
Ref_4	read	10-60 ns	48 ns	<10 ns	<10 ns
	write	10-60 ns	40-150 ns	~10 ns	12.5 ns
Ref_5	read	--	70 ns	7.2 ns	11 ns
	write	--	>180 ns	<7.2 ns	25 ns
Ref_6	read	--	10-100 ns	--	--
	write	--	100-1000 ns	--	--
Ref_7	read	--	--	1.773 - 426.8 ns	--
	write	--	--	100.6 - 518.2 ns	--
Ref_8	read	70 ns	206 ns	--	--
	write	70 ns	7100 ns	--	--

Figure 6: Expected read/write latencies extracted from publications

In order to compare the access behavior of a DRAM with the one of an emulated SCM, we carried out measurements with test pattern derived from the map depicted in Figure 7. In the map the contour of the triangle comprises the area of expected SCM latencies, once SCM is available.

The triangle is derived from the SCM characteristics given in Figure 6: Using the fact that reading will always be faster than writing, and assuming that SCM read/write will be always slower than DRAM read/write, and based on our given real system DRAM latencies as the starting point P5=(80ns;80ns), the contour of the triangle was estimated considering SCM_Ref_1 and SCM_Ref_2 to be positioned in the center area of the triangle.

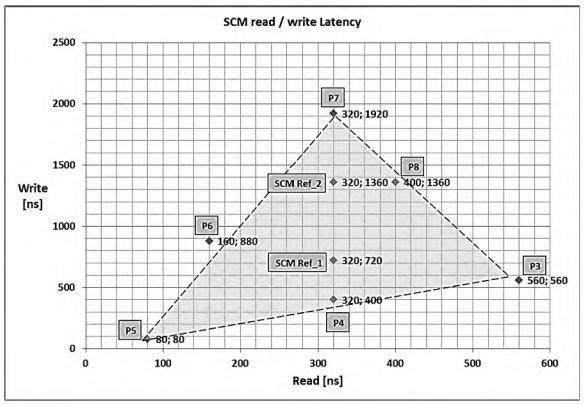


Figure 7: Expected range of SCM latencies considered towards emulation

From the triangle the test points listed in Figure 8 have been chosen.

SCM emulated test points		
Test point	read [ns]	write [ns]
SCM_Ref_1	320	720
SCM_Ref_2	320	1360
P3	560	560
P4	320	400
P5	80	80
P6	160	880
P7	320	1920
P8	400	1360

Figure 8: Chosen test points

In the emulator a test point can be applied by setting a specific read/write delay factor using the "scmadm" command. The following sections describe tests with the synthetic program "fio" and tests with a real SAP HANA which both have been carried out using the test point parameters listed in Figure 8.

4.2 Flexible I/O Tester (fio)

The "fio" is an I/O tool meant to be used for both, benchmarking and stress/hardware verification. It can work on block devices as well as on files. The "fio" accepts job descriptions in a text format.

We used the block device "fio" test according to the following description:

[global]
bs=4k
direct=1, numjobs=1
ioengine=libaio, iodepth=2
iodepth_batch=1, iodepth_batch_complete=0
rw=randwrite/randread, use_os_rand=1
randrepeat=0, time_based
runtime=30, [job]filename=/dev/scm0
size=1GB

Figure 9 presents reading-test results with read latencies of 80/160/320/400/560 ns. The order of the curves corresponds to the order of latencies used.

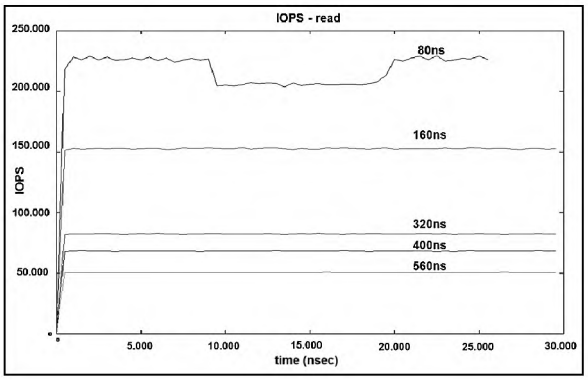


Figure 9: IOPS for different read latencies

Figure 10 presents writing-test results with write latencies of 80/400/560/720/880/1360/1920 ns. The order of the curves corresponds to the order of the latencies used.

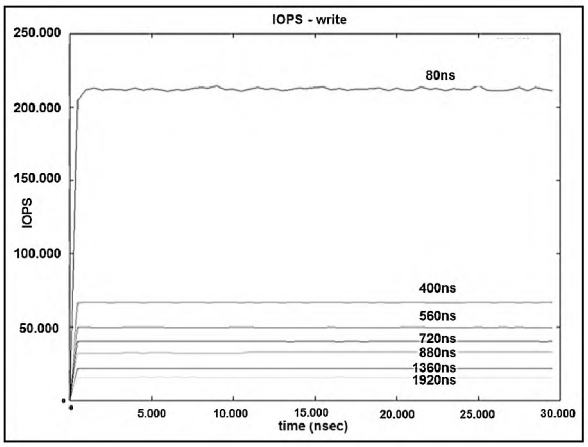


Figure 10: IOPS for different write latencies

From Figure 11 we can conclude that latency has an exponential-like impact on IOPS. Latencies larger than 750 ns bring IOPS dramatically down.

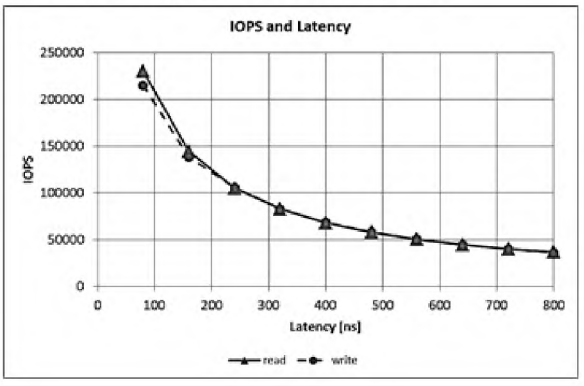


Figure 11: Impact on IOPS by different emulated latencies

4.3 SAP Business Suite on SAP HANA

SAP Business Suite on SAP HANA (SOH) offers an Enterprise Resource Planning System (ERP), which stores data using the in-memory computing database SAP HANA. The solution is used in productive environments and implements, as seen from the perspective of the database, typical Online Transaction Processing (OLTP) workloads. In order to benchmark the usage of Storage Class Memory as an archival medium for SAP HANA we decided to base a test on SOH. Figure 12 illustrates our general benchmark environment. We use a three-tier approach, which means that SOH and SAP HANA are running on separate servers. SAP HANA is using the emulated SCM in order to write the transactional logging.

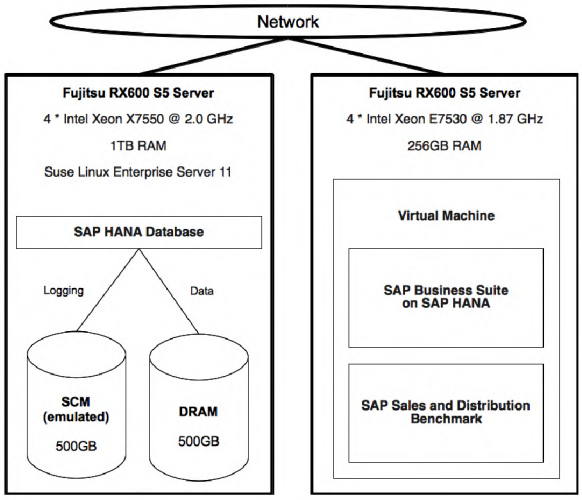


Figure 12: SAP Business Suite on SAP HANA based benchmark environment

In order to simulate a real world scenario based on SOH we used the SAP Standard Application Benchmarks. These can be applied to compare hardware configurations for IT solutions. The Sales and Distribution (SD) benchmark is appropriate to be used for SOH. It "[…] covers a sell-from-stock scenario, which includes the creation of a customer order with five line items and the corresponding delivery with subsequent goods movement and invoicing." [6]. It simulates an arbitrary number of concurrently working users by replaying dialog based working steps in the user interface. Thereby a business process is being replayed which creates a sales order, creates a delivery note for the order, displays the order, changes the delivery, posts a goods issue, lists orders and creates an invoice. The process is looped several times in order to ensure a minimum runtime of the benchmark. In our configuration we simulate 300 concurrently working users on the SOH. SCM is used to store the transactional logging written by SAP HANA. We analyze several SCM configurations as described in Figure 7.

Figure 13 illustrates the resulting request times of read and write operations for different SCM settings. By request times the time interval between the request of a read or write operation send by SOH to SAP HANA and the response of SAP HANA to SOH is described. Lower request times can lead to more fluent working processes for the users of SOH.

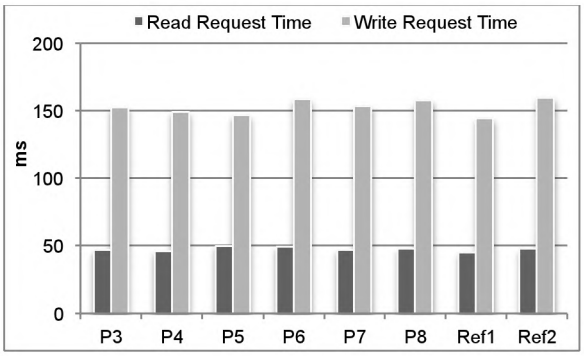


Figure 13: SAP Sales and Distribution Benchmark results showing the database read and write request times for different storage technologies and SCM emulator latencies.

The comparison of different SCM emulator latencies shows a clear independency of SCM latencies and database request times. This result might be explained by a rather small amount of write operations of the OLTP workload. Read operations will not directly benefit from the increasing logging performance. As a consequence any possible future latency characteristic of SCM would be acceptable to be used for traditional OLTP workload but it can be argued that those results will also be achieved by using today's SSD based solutions. In order to take a closer look on a more appropriate workload which is utilizing SCM we will introduce a completely insert based benchmark in the next chapter.

4.4 Event-Stream Processing

In traditional OLTP workloads more than 80% of all queries are read-only. Using Online Analytical Processing (OLAP) workloads this amount even increases to 90 % [2]. Hence measuring the insert performance of SAP HANA using the SCM emulator and a traditional OLTP or OLAP workload will not be an ideal use case to emphasize differences. Therefore other cases of application need to taken into account.

A more appropriate workload is introduced by the event-stream processing (ESP). In this approach an event-driven information systems is sending the incoming data to the SAP HANA database wherefrom it is going to be used for analytical operations. The data might be generated from sensors, which observe physical processes, or by another software system. Software related real life use cases would be high frequency trading or real time bidding. In order to make data available in the database as soon as it arises, the database has to provide high insert performance. For the following benchmark a very simply event streamer has been implement in C++ using an Open Database Connectivity (ODBC) connection

in order to send data to the SAP HANA database. As described before, SAP HANA is using the emulated SCM in order to store the transactional logs. The workload only consists of inserts.

As described in chapter 2, we concentrate on a selected range of possible future SCM latency characteristics. We also add results for logging to a hard disk, a SSD based solution using the PCI-Express interface and a RAM-disk. Logging to a hard disk indicates the lower bounds of database performance whereas logging to a RAM-disk, which is an emulated block device based on DRAM, equals the theoretically upper bound. SSD and PCI-Express based solutions support low write latencies, being non-volatile and are available on the market today. Using an identical workload, the benchmarks have been run for the different SCM emulator configurations and the mentioned storage technologies. The results can be found in Figure 14 and show the relation between different read and write factors of the technologies and the resulting number of inserts per second, which can be achieved by the database. The runtime of each benchmarks equals 15 minutes. We assumed a batch size of 6, describing the number of inserts being committed at once, and used 30 parallel connections to the database. The number of inserts per second is mainly influenced by the time needed of an insert statement, which again is influenced by the latency of the storage technology. Using a RAM-disk and the given benchmark setting, the insertion time for one statement is equal to one microsecond, which is describing the lower bound.

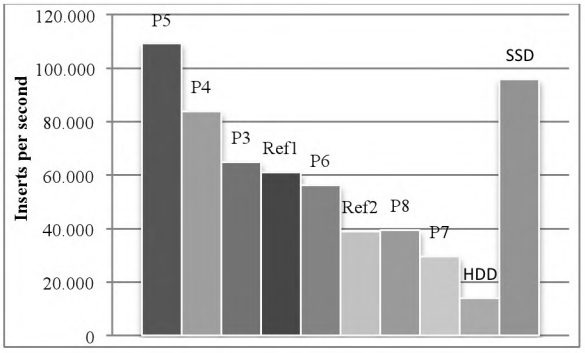


Figure 14: Event-stream processing benchmark results showing the number of inserts per second for different storage technologies and SCM emulator latencies.

Figure 14 gives a first insight into the performance of SAP HANA using different technologies to store logs. A visible declining of the number of insert per second can be seen for increasing latencies. As for the benchmarked SCM references characteristics vary in write and read latencies, the cause for the declining inserts per second does not completely get clear at this point. Therefore two more benchmark series have been taken into account, which are looking at changing write and read latencies separately. As shown in Figure 15 a linear decline for the number of inserts per second arises when constantly increasing the write latency of the SCM emulator. A similar result could not be shown for increasing read latencies and constant write latencies. For increasing read latencies the number of inserts per second stayed approximately constant. This leads to the assumption, that write performance is the main bottleneck.

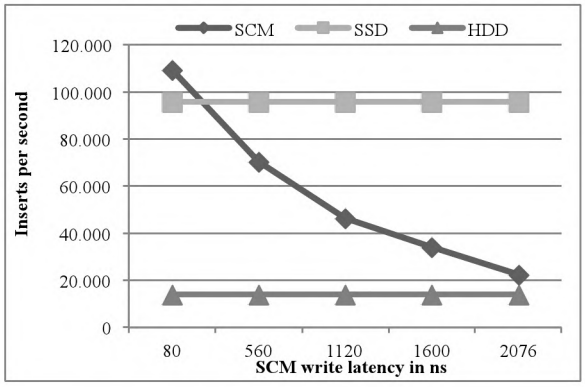


Figure 15: Event-stream processing benchmark for increasing SCM write latencies and constant read latencies of 80ns.

Looking at the number of write and read operation, as produced by the log process on the log device, the dependency of the write latency can be explained. Figure 16 shows the proportion of write and read operations. Summarizing the given results, the conclusion has to be made that SCM is not ideal to be used for storing the transactional logs of SAP HANA. The combination of a mainly write latency dependent operation and the low write performance of SCM lead to the measured results. A possible reason can be seen in the usage of a block addressed storage. Therefore further development needs to be done on behalf of the application in order to utilize the characteristics of SCM. Main improvements might be achieved by introducing fine granular addressing and memory based copy operations instead of CPU demanding storage write operations.

Read Operations	Write Operations
0,05 %	99,95 %

Figure 16: Percentage distribution of storage write and read operations within 15 minutes produced by the log process of SAP HANA

5 Conclusion and Next Steps

We developed a software based Storage Class Memory emulator to make the latency characteristics of SCM available today. Based on this we used several benchmarks to evaluate the usability of SCM as a block device in general as well as for the transactional logging of SAP HANA. Also we described different use cases based on SCM and SAP HANA.

Looking at the benchmark results we can emphasize that using SCM as a block device is beneficial for latencies lower than 150ns but the resulting performance can only slightly outperform today's solutions. Taking the word addressability of SCM into account might be crucial to achieve even better results. Therefore decisive changes have to be made to the existing application in order to utilize the technology. In a next step we will take a closer look at the performance of SAP HANA using SCM to store the in-memory data. Additionally, we will investigate if the technology is suitable for use cases like data aging.

References

[1] Hasso Plattner, Alexander Zeier: In-Memory Data Management: An Inflection Point for Enterprise Applications, Springer, 2011.

[2] Jens Krüger, Changkyu Kim, Martin Grund, Nadathur Satish, David Schwalb, Jatin Chhugani, Hasso Plattner, Pradeep Dubey, Alexander Zeier. Fast Updates on Read-Optimized Databases Using Multi-Core CPUs. Proceedings of the VLDB Endowment, 5 (1):61–72, 2011.

[3] SAP AG. Data Management Guide for SAP Business Suite, 2006.

[4] X. Dong, Y. Xie, "Modeling and Leveraging Emerging Non-Volatile Memories for Future Computer Designs", Submitted to ACM Student Research Competition, Department of Computer Science and Engineering, Pennsylvania State University, USA, 2011.

[5] S. Lee, B. Moon, C. Park, J.Y. Hwang, and K. Kim, "Accelerating In-Page Logging with Non-Volatile Memory", presented at IEEE Data Engineering, Bulletin, 2010, pp.41-47.

[6] C. Mohan, S. Bhattacharya, "Implications of Storage Class Memories (SCM) on Software Architectures", Presentation at workshop on the use of Emerging Storage and Memory Technologies (WEST), HPCA 2010, Bangalore, India, January 2010.

[7] D. A. Roberts, "Efficient Data Center Architectures Using Non-Volatile Memory and ReliabilityTechniques", Dissertation, Computer Science and Engineering, University of Michigan, USA, 2011.

[8] T. Perez, C. De Rose, "Non-Volatile Memory: Emerging Technologies And Their Impacts on Memory Systems", Technical Report Nº 060, Pontifícia Universidade Católica do Rio Grande do Sul; Faculdade de Informática Pós-Graduação em Ciência da Computação, Porto Alegere, Brasil, September 2010.

[9] Y. Xie, "Emerging NVM Memory Technologies", Presentation, Department of Computer Science and Engineering, Pennsylvania State University, USA, 2010.

[10] C. Xu, X. Dong e.a., "Design Implications of Memristor-Based RRAM Cross-Point Structures", Proceedings Design, Automation and Test in Europe Conference DATE 2011, Grenoble, France, March 2011.

[11] J. Grollier, "Memristors", Presentation at NANO-TEC Workshop 2 - Benchmarking of new Beyond CMOS device/design concepts, Athens, Greece, October 2011.

[12] SAP AG. SAP Standard Application Benchmarks. Sales and Distribution. http://www.sap.com /campaigns/benchmark/appbm_sd.epx. Accessed: 21/03/2013.

Adaptive Realtime KPI Analysis of ERP transaction data using In-Memory technology

Prof. Dr. Rainer Thome
Chair of Business Administration
and Business Computing
Joseph-Stangl Platz 2
97070 Wuerzburg
thome@wiinf.uni-wuerzburg.de

Dr. Andreas Hufgard
Dipl.-Kfm. Fabian Krüger
Dipl.-Kfm. Ralf Knauer
IBIS Labs
Mergentheimer Str. 76a
97082 Wuerzburg
hufgard@ibis-thome.de
fkrueger@ibis-thome.de
knauer@ibis-thome.de

Abstract

The adaptable real-time analysis provides an answer to the greater complexity and size of advanced ERP systems (big data), and increased market demands for speed and quality of decision-related data. In-memory storage enables direct access to documents or change data, and allows more flexibility when selecting by date, organization and many other characteristics. During this project we have been able to gain some basic knowledge about SAP HANA, transfer data and visualize it with the built-in XS Engine and SAPUI5.

1 Project idea

The goal of this project was to create a highly interactive analysis environment for process KPIs. We are interested in analyzing features like historic tables and executing statistical procedures directly on the database level. This means we focused on the software innovations of SAP HANA instead of maxing out the hardware.

1.1 Used resources at HPI

The team at HPI gave us access to a SAP HANA (SPS05) instance which is being hosted in Potsdam. This instance could be accessed via VPN using the SAP HANA Studio. The Studio installation was directly located on our clients which eliminates the need for any remote desktop. The connection works seamlessly, we didn't have problems at all.

Unfortunately it was not possible to use SAP System Landscape Transformation (SLT) to connect our SAP ERP to SAP HANA in this term. This is a consequence of legal reasons preventing the installation of the DMIS add-on in any non-HPI system.

The use of an R-Server for statistical assessments was not possible, too. The team of HPI is working on this but this feature will probably not be available until the next term.

1.2 Lessons learned

SAP HANA made a big step for developing applications with version SPS05. This is represented by the new, almost 400 pages long, developer guide, which is available at help.sap.com.

Now there is a repository for synchronizing developments with team members, a new development perspective in SAP HANA Studio, Projects to organize the development process and delivery units to transport those to other systems and customers.

Additionally SAP HANA now contains the XS Engine, allowing Server-Side Java Scripts, HTML5 GUIs (along with the SAPUI5 library) and OData support.

These new features made us redesign our project and replace Business Objects by SAPUI5 for the presentation layer. SAPUI5 does not require a separate license, is capable of mobile devices and keeps the system landscape lean.

2 Project related progress

The project aim is to create a fully functional prototype of an interactive real-time analysis environment. There are three separate activities to achieve this goal: You need a user interface or presentation layer, data to run the analysis on and last but not least the calculations which turn the data into KPIs to be displayed.

Datum	Beleganzahl
1.12.2012	354
2.12.2012	415
3.12.2012	529
4.12.2012	520
5.12.2012	462
6.12.2012	410
7.12.2012	483
8.12.2012	419
9.12.2012	485
10.12.2012	447

Figure 1: Table with SAPUI5

Additionally this data can be enhanced using statistical functions to find anomalies and display those to the user.

2.1 Presentation layer

Although we originally intended to use Business Objects, we will now use SAPUI5 for the presentation layer. SAPUI5 is a HTML5 based GUI suitable for desktops and mobile devices as well. There is no need for client software other than a standard web browser.

As far as we can evaluate, SAPUI5 offers anything we could possibly need in the presentation layer. It features a rich lineup of different charts (Line, Pie, Donut, Bar, Column, Bubble, Scatter, etc.), tables to display detailed data and many controls offering the user to interact with the data. A full reference of SAPUI5 can be accessed directly on the HANA Server at http://hana-2:8001/sap/ui5/1/sdk/.

The main advantages compared to SAP BO are:

- Development directly integrated in SAP HANA Studio by using the SAPUI5 eclipse plugin.
- No additional software component needed. This also reduces the complexity of the system landscape and makes it more attractive for customers.
- Since it is a JavaScript library there are a lot of debugging tools available. The JavaScript developer community is also very active making it easy to find answers if questions arise.
- Cross-Platform and cross device compatibility. HTML5 can be consumed on Windows, Mac, Linux, iOS, Android and probably any other operating system supporting state of the art web browser.

As shown in Figure 1 SAPUI5 can easily show data being exposed via OData in a nice looking, user friendly table. Figure 2 shows another example displaying the same data in a line chart.

2.2 Connection to SAP ERP

Since we could not connect any SAP ERP to our SAP

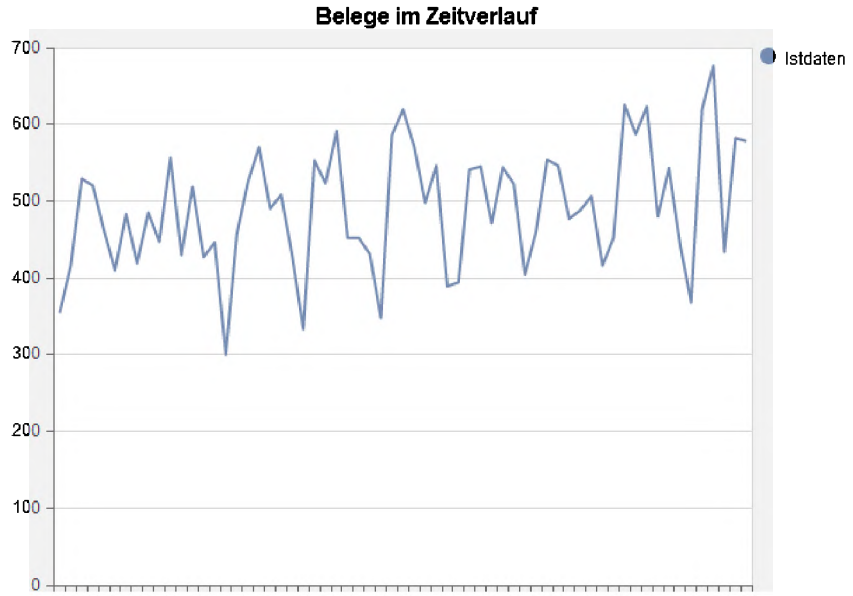


Figure 2: Line Chart with SAPUI5

HANA instance, we still had to rely on sample data.

This data has been extracted from an IDES System into .csv files and then imported into SAP HANA using the "Import from local file" feature in SAP HANA Studio. It is suitable for testing some queries but does not change over time. That's why we simulated a time series and used this data for the presentation layer. From a technical point of view it does not matter if the data is from a simulation or an SQL-Script actually calculating the real number of open orders per day. Once there is real-time data available the data source can be exchanged easily.

2.3 KPI Calculations

As presented in the last report, we already have some procedures calculating the open sales orders for any user-defined date in the past. The only change in KPI calculation is that we can now store our procedures in the repository using ".procedure" files. On Activation, these Design-Time files create the stored procedures (Run-Time). One of the main advantages of those Design-Time .procedure-Files is the ability to debug them. You can set breakpoints, run the procedure and the debugger will stop on every breakpoint, allowing you to see the contents of any variable. You can also control which parts of the code were actually executed and which were not when using if-clauses or loops.

Other than this technical change, due to the new release, there have been no changes to the KPI calculation.

2.4 Statistical integration

In the beginning we thought SAP HANA would natively support the statistical programming language R since you could define procedures and set the language to "RLang". Unfortunately it turns out that SAP HANA does not process these procedures itself but sends it to another server with an R installation as well as the plugin RServ. All the statistical computations are then executed on the R server and then sent back to SAP HANA. This is a massive drawback since a potential customer would need another server in his landscape, this server would also need a fast CPU and a lot of RAM, and it would have to be set up and maintained separately.

Nevertheless we developed an R procedure which computes the weekday of the date and executes a linear regression on the documents, explaining the number of documents as function of a constant, a general linear trend and the weekday. Using this regression results, R can even predict the next few days and calculate a lower and upper confidence limit.

Since the integration between SAP HANA and R was not possible this term we were able to test it locally. Using RStudio and the sample data as seen in figure 1 and 2 we get the output seen in figure 3.

The red line shows the actual data which ends on

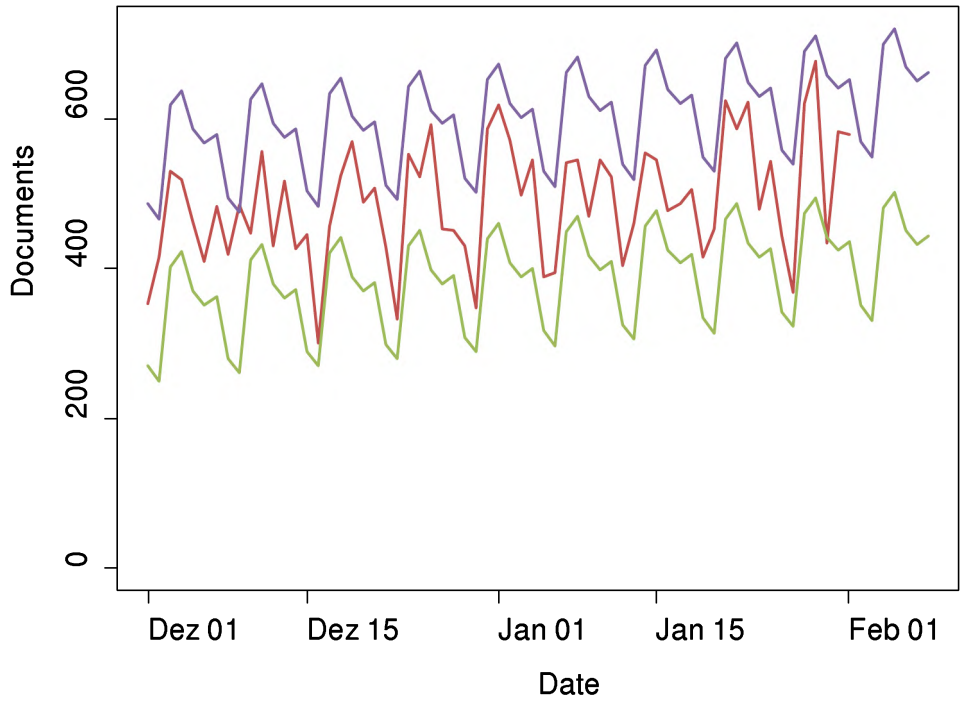


Figure 3: Dynamic Thresholds with R

February 1. Using this data R computes the purple (upper limit) and green (lower limit) lines.

The background of this is that some KPIs show natural fluctuation within a week. For example, open orders might be higher on Mondays since customers order products on the weekend but nobody is working to process them. On the other hand, open orders should dramatically reduce until Friday because otherwise the customers would have to wait the whole weekend. Therefore you need different thresholds depending on factors like the weekday.

Since R integration with SAP HANA involves some problems with extra hardware, we are currently analyzing the brand new PAL (Predictive Analysis Library) functionality from SPS05. This library also offers statistical functions but those can be executed natively by the SAP HANA instance itself. Maybe this can reduce the complexity of the application.

3 Further research

Until now we have already proven that SAP HANA has all capabilities to enable the interactive reporting we looked for. All of the layers (presentation, calculation, and statistics) are working as separate parts, but to make this application fully functional there are some tasks left:

- The data source is still missing. SAP now offers the whole SAP Business Suite on HANA. This eliminates the need for SLT or other replication tools. A high quality data source is needed for testing the time travel queries.
- Until now we only have looked into the open orders. There are similar ones like blocked, cancelled and completed orders and some which need to be calculated differently like 'changed orders'.
- We already eliminated Business Objects as an external dependency by using SAPUI5.
- R integration is still a problem but could possibly be replaced by PAL.
- All the different layers need to be connected to each other and integrated. If there are multiple GUI elements (like tables and charts), they should respond to selections, send those as filters to the HANA server and then update the other elements as well. On the other hand it would be nice to have some kind of push notification for the user if the current KPI value is out of the control limits.

Application Aware Placement and Scheduling for Multi-tenant Clouds

Lalith Suresh[1] and Marco Canini[2]

[1]Technische Universität Berlin, Germany
[2]Université Catholique de Louvain, Belgium
{lsuresh@inet.tu-berlin.de, marco.canini@uclouvain.be}

Abstract

In an Infrastructure-as-a-Service (IaaS) environment, it is paramount to perform intelligent allocation of shared resources. Placement *is the problem of choosing which virtual machine (VM) should run on which physical machine (PM), whereas* Scheduling *is the problem of sharing resources between multiple co-located VMs. An efficient placement and scheduling is one, that in addition to satisfying all constraints, increases the overall utilization of physical resources such as CPU, storage, or network. Determining an efficient placement and scheduling is a very challenging problem, especially in face of conflicting goals and partially available information about workloads.*
In order to reason about placement, we first tackle the problem of performance interference that may affect co-located VMs—when there is more demand by multiple VMs for a resource than is available at a given instant of time. We thus characterize the performance of Hadoop in a shared and virtualized setting.

1 Introduction

The focus of our study is multi-tenant, multi-purpose Infrastructure-as-a-Service (IaaS) cloud data centers, wherein servers are virtualized, with multiple tenants deploying applications atop shared infrastructure. Utilizing compute, storage and network resources efficiently is a crucial objective in these environments as virtual machines (VMs) are collocated on physical servers and contend for a set of shared resources on the physical server and network.
However, it is well-known that virtualized computing environments can suffer from *performance interference* (see e.g., [4]). More precisely, due to imperfect performance isolation by the underlying hypervisors and lack of fine-grained bandwidth assurances across tenants within the network, the performance of a VM may suffer due to collocated VMs. Such performance interference can lead to unpredictable delays for the cloud tenant, which would ultimately manifest in failed service-level agreements (SLAs) at the application level and in loss of business revenue.

To mitigate performance interference, it is thus important to balance the different workloads of various tenants across the data center. While VMs from different tenants may be collocated on the same physical host, different tenants in the data center may have varying demands for resources depending on the applications that they are running. For instance, a tenant running a Hadoop [3] deployment on the cloud may have a greater demand for disk I/O bandwidth than a tenant managing a multi-tier website. Depending on the resource demands of different applications being run by different tenants, VMs may experience varying degrees of performance interference due to contention for shared resources. Thus, the performance degradation experienced by different VMs is sensitive to the specific placement of VMs in the data center [7].

Since this performance degradation depends on the applications being run within the VMs, understanding how applications behave and where their bottlenecks are may allow an operator to perform smarter placement of VMs within the data center. This project takes a fresh look at the fundamental problem of application placement in the private cloud environment. Application placement is the task of deciding which VM runs on which physical server. There have been proposals to address application placement from both theoretical and systems standpoints. For instance, theoretical approaches model the environment as a multi-dimensional bin packing problem [8] or provide schedules based on statistical models [1]. However, these models are too coarse to accurately account for performance interference, which require fine-grained experimentation to observe. On the other hand, [5, 6, 2, 9] propose techniques to identify performance interference by observing low-level system

metrics on individual VMs and then making placement decisions to minimize it or allocating more resources to compensate for interference. However, these approaches do not exploit the specific characteristics and goals of the applications.
In fact, in today's cloud ecosystem, most applications are distributed across more than one VM. With service-oriented architectures becoming the norm, systems are decomposed into multiple, loosely coupled, communicating clusters of components. An individual logical component in an application could by itself be a self-healing system comprised of multiple nodes, which have built-in mechanisms to compensate for stragglers or slower nodes (e.g, MapReduce's scheduler). Taking these aspects into consideration, we argue that performance interference should be reasoned about at the granularity of *logical components of the application* using application-specific metrics, and not at the granularity of the individual VMs using system-level metrics. This project aims to demonstrate this fact experimentally.
In order to design *application- and interference-aware approaches* to plan application placement in multi-tenant clouds, we first try to characterize the performance of different systems under multi-tenant environments. This information can then be used to effectively place applications in a cloud, as show in the methodology illustrated in Figure 1. In the findings presented below, we present performance results of running Hadoop in shared environments.

2 Usage of Future SOC Lab

HPI FSOC Lab provided us access to a state-of-the art, 1000-core computing cluster. The cluster consists of 25 nodes, each equipped with 40 cores at 2.40 GHz, 1 TB RAM, 3.6 TB SSD storage and 10 Gbps Ethernet. To use this cluster for our experimental study, we needed it to be configured as a virtualized cloud environment, and we obtained simultaneous, dedicated access to all nodes.
We ran experiments of the Hadoop cluster under multiple configurations and co-location scenarios.

- Baseline: every VM of the Hadoop cluster is running on a separate physical machine.
- Co-located Datanodes: two Hadoop Datanode VMs from the same cluster are co-located each physical machines.
- Co-located Clusters: a VM each from two different Hadoop clusters are co-located on the same physical machine.

Through our runs, we vary the number of reducers used by Hadoop, the Hadoop scheduler being used (Capacity and Fair schedulers), and we enable/disable speculative execution (wherein tasks are speculatively cloned to account for stragglers). We use the following notations to describe the combinations of these parameters. *cpt* and *cpf* represent the use of the Capacity Scheduler along with speculative execution enabled and disabled, respectively. *fst* and *fsf* represent the use of the Fair Scheduler along with speculative execution enabled and disabled, respectively. The workload we run is the TeraSort benchmark that is packaged with the Hadoop distribution, with four jobs submitted in parallel.

3 Findings

In this section, we describe some of our findings.
Figure 2 represents measurements of the difference between the maximum and minimum job completion times when submitting four TeraSort jobs to the cluster at the same time, under varying scenarios. We use this as an indication of fairness in the system. We find that the baseline and co-located Datanodes scenarios provide identical fairness when using different schedulers and regardless of whether speculative execution is enabled or disabled. However, in the co-located clusters scenario, we observe that due to performance interference from each cluster on the other, the system is unable to guarantee fairness in job completion times even when using the Hadoop Fair Scheduler. Figure 3 describes the same set of results but when using 10 reducers instead of 1. Since Hadoop can now distribute the reduce tasks over multiple nodes, the skew in job completion times is minimized, but still remains significant in the co-located clusters scenario.
One challenge we faced in obtaining more results and measurements was that the specifications of the physical machines being used deviated significantly from those used in typical Infrastructure-as-a-Service deployments, where servers have a different CPU cores to I/O bandwidth ratio.

4 Summary and Future Work

Our studies indicate that performance interference not only affects the job completion times of Hadoop directly, but also leads to interference in the Hadoop schedulers, affecting the guarantees they are expected to provide (such as fairness). We plan to use these measurements in order to prepare measurement-driven interference models which can in turn be used as input to placement algorithms which can be used to map VMs to physical machines in cloud environments. Our study will also extend to other systems such as key-value stores and the interactions between these systems.

5 Acknowledgements

We would like to thank the system administrators of the Future SOC-Lab infrastructure for their support, in

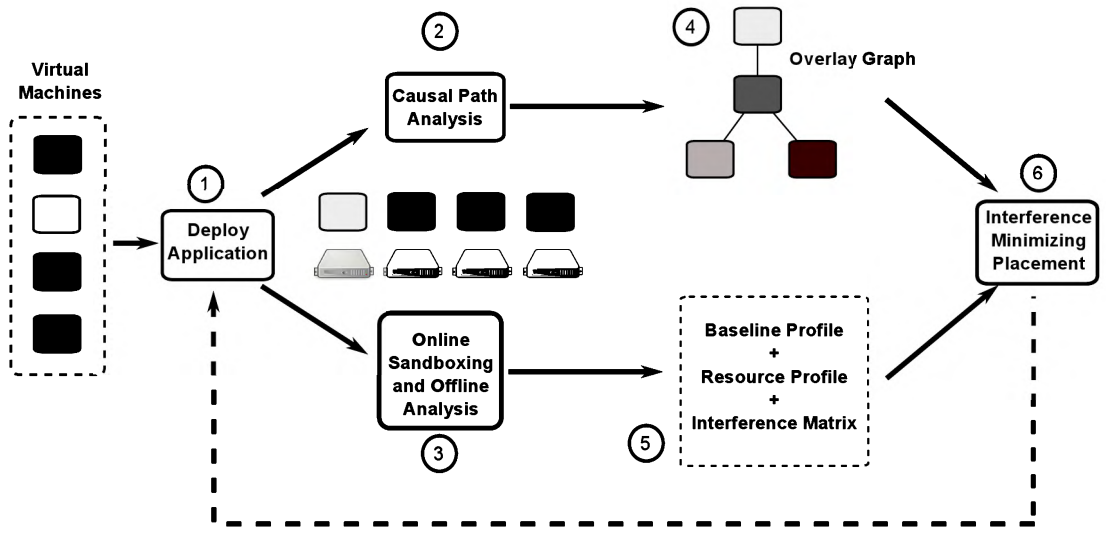


Figure 1: At a high level, our workflow combines online and offline measurements of applications with specific information about the application's internals in order to perform interference minimizing placement.

particular with setting up OpenStack on the machines. We also thank Zubair Khwaja Sediqi for help with the setup of the measurement code and workloads.

References

[1] R. C. Chiang and H. H. Huang. Tracon: interference-aware scheduling for data-intensive applications in virtualized environments. In *Proc. ACM SC*, 2011.

[2] D. Novaković et al. Deepdive: Transparently identifying and managing performance interference in virtualized environments. In *Proc. USENIX ATC*, 2013.

[3] A. Hadoop. `http://hadoop.apache.org/`, last accessed Dec 20th, 2013.

[4] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu. An analysis of performance interference effects in virtual environments. In *Performance Analysis of Systems & Software, 2007. ISPASS 2007. IEEE International Symposium on*, pages 200–209. IEEE, 2007.

[5] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu. An analysis of performance interference effects in virtual environments. In *Proc. IEEE ISPASS*, 2007.

[6] R. Nathuji, A. Kansal, and A. Ghaffarkhah. Q-clouds: managing performance interference effects for qos-aware clouds. In *Proc. EuroSys*, 2010.

[7] A. Roytman, A. Kansal, S. Govindan, J. Liu, and S. Nath. Algorithm design for performance aware vm consolidation.

[8] B. Urgaonkar, A. Rosenberg, and P. Shenoy. Application placement on a cluster of servers. *Int. J. Found. Comput. Sci.*, 2007.

[9] Y. Xu, Z. Musgrave, B. Noble, and M. Bailey. Bobtail: avoiding long tails in the cloud. In *Proc. NSDI*, 2013.

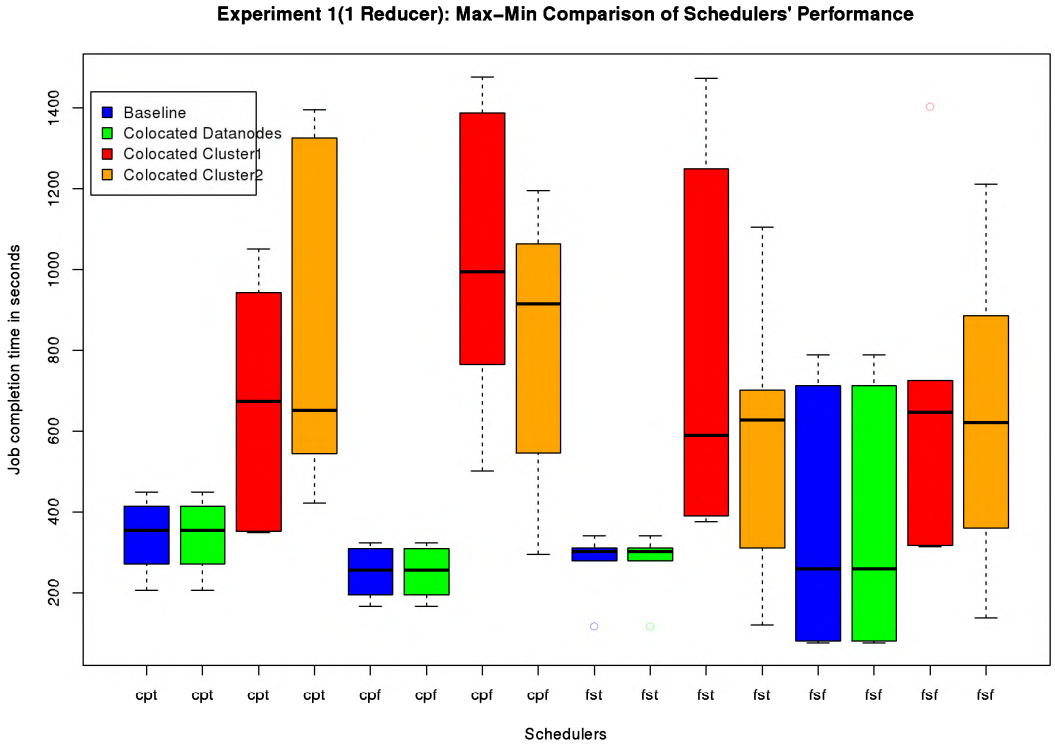


Figure 2: When using the capacity and fair scheduler for Hadoop, the systems are unable to guarantee fairness of job completion times when running as co-located clusters due to performance interference.

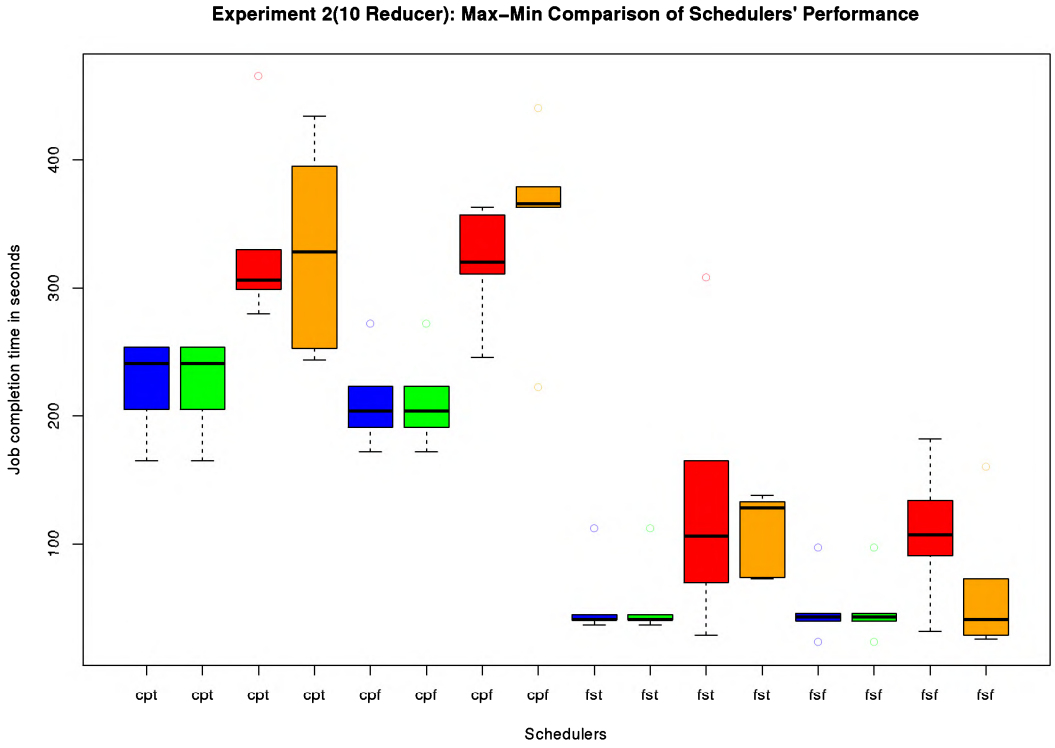


Figure 3: As with the single reducer case, the systems are unable to guarantee fairness of job completion times when running as co-located clusters. However, using more reducers reduces the skew in job completion times.

A framework for comparing the performance of in-memory and traditional disk-based databases

Vassilena Banova
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
vassilena.banova@in.tum.de

Alexandru Danciu
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
danciu@in.tum.de

Helmut Krcmar
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
krcmar@in.tum.de

Abstract

Enterprises are facing an increasing amount of data, and rely more than ever on flexible and fast methods of data analysis. A proposed solution is the switch to in-memory database systems which promise huge performance increases. Unfortunately recent literature does not provide performance comparisons of main-memory and classical storage based database systems in enterprise use cases. A literature review is conducted identifying the scenario of discovering frequent item-sets as an appropriate use case to perform such a comparison. Requirements on a test environment are provided and a tool supporting those tests is implemented, encouraging the execution of OLAP, OLTP and mixed workload test-suites.

1 Introduction

Today's enterprises store and collect more data than ever. It is estimated that the fortune 500 enterprises store approximately seven to ten years of customer data [1]. This amount is told to increase even further in the coming years [2]. Amongst others this data is used in the context of Business Intelligence & Analytics"(BI&A) to gain insights on the market and make critical decisions in a timely manner based on up to date information [3]. But the huge amount of data makes it necessary to narrow the data down or to work with pre-calculated results what limits the flexibility and expressiveness of the analysis [4]. In an environment of fast changing market conditions and customer wishes this flexibility is crucial to recognize influences on the own business and react on undesirable developments [2].

A frequently used term in this context is "Big Data" what describes the processing and analysis of huge amounts of data with complex structure [3] and is sometimes referred to as a scaled BI&A [5]. The properties of Big Data, "high-volume, high-variety, high-velocity, and high-veracity"[6], often referred to as the three "HVs" [6], make it necessary for enterprises to come up with new requirements on storage, management, analysis and visualization of data [3]. One proposed solution to deal with these requirements is the use of in-memory databases, which describe database systems holding the entire dataset in main memory [7]. A study conducted amongst German IT-professionals in 2012, pointed out that every fifth enterprise thinks that meeting the challenges of the future data volume is only possible using in-memory technology. But the same study pointed out, that 60% of the enterprises are not able to make a clear decision on whether in-memory databases are relevant to them. They are especially concerned whether the proposed advantages of in-memory technology can really be leveraged in their enterprise [2].

To achieve a better understanding about the effects of in-memory databases and to be able to make a sound decision on whether to invest into in-memory technology and migrate from traditional disk-based storage to main memory storage, enterprises need tests to be executed, comparing those technologies in enterprise use cases and providing insights on basic characteristics of the database systems. Those tests will help enterprises to determine the most suitable set up for their needs. This project aims on developing some basic test cases and the necessary toolset for executing this kind of tests.

2 Project Implementation

2.1 Project Description

The project aims at the development of a testing environment for the comparison of response times of in-memory based and classical disk storage databases in enterprise use case scenarios.

Two identical virtual machines, one with an in-memory database system, the other with a classical disk based storage database are used to verify, that the developed test cases can be executed on such environments. Therefore the same dataset is deployed to both virtual machines. Queries are generated and executed from a third, independent, source where response time is measured. The target is to provide a test setup including enterprise use cases being able to compare two database systems.

2.2 Project Phases

In a first phase a literature review on database performance and benchmarks was conducted to identify relevant stakeholders, use cases and requirements for testing. During this literature review four major groups of stakeholders and their major topics of interest could be identified. The four groups, consisting of database developers, database administrators, business analysts and end-user/decision makers were characterized and the assumption was made, that the group of business analysts is the most appro-priate target group and that scenarios in data mining such as the identification of frequent item-sets or the measurement of profits and losses generated by a specific sales-channel and its components are appropriate use cases for testing. Furthermore the absence of tests comparing an in-memory database to a classical disk storage database in a comprehensible, reproducible way and on equal conditions could be approved.

In a second phase a framework for plan execution was developed simulating different circumstances for the virtual machines. The framework focuses on varying three components in the test environment - namely the number of threads executing one query in parallel, the size of the dataset and plans of execution, determining whether update and select statements are executed irrespective of each other or not. The number of threads and the different datasets represent the different amount of users and size of an enterprise. Different plans of execution allow a comparison of the databases for different areas of application. This increases the expandability of the tool to further application domains and aims on evaluating performance in mixed OLTP (Online Transaction Processing) and OLAP (Online analytical Processing) workloads.

In a third phase appropriate datasets were generated and queries for testing were developed. Furthermore a tool was implemented to execute the tests and measure response time on both virtual machines on equal conditions.

Data is extracted from the publicly available data generator of the TPC-DS benchmark, an industry standard benchmark for decision support systems (see: www.tpc.org), as it is available for free and approved to provide appropriate data for performance testing. For reasons of simplification only two of those tables were used. "catalog_sales" and "catalog_returns" which provide a fact table of sales through the catalog sales channel including items and orders and the profit and loss information of each line item. This information, in combination with the page number of the catalog, the item was ordered from, allows to execute the aforementioned business use cases. The tool for query execution is written in Java and uses JDBC to send the chosen queries to the databases and measure their response time.

3 Description of the Virtual Machines

The virtual machines provided by the HPI Future SOC Lab both had 64 GB main memory, a hard drive disk with a capacity of 100 GB and a total of eight CPUs available. They are both located in the same network and run a Linux Kernel 2.6.32.59-0.7.

4 Test-framework and Toolset

4.1 Requirements

The test-framework is supposed to provide decision support for enterprises when deciding on switching to or adding an in-memory database to their IT landscape. Therefore the framework for testing has to simulate several environmental scenarios/enterprise characteristics and provide details about the influences on performance if those environmental factors change. As a test environment will never exactly match a real enterprise, metrics of comparison should be expressed in relative numbers and an easy, comprehensible way such as "On a level playing field, an increase of users by factor x performance loss of system A is factor z and performance loss of system B is factor y - B runs r times faster than A".
Another important factor for generating an as equal as possible setup of the two databases, is to forbid every kind of tuning, such as additional indexes. Methods or elementary approaches as parts of the database management system can be used. Therefore tuning mechanisms such as adding further indexes to a table have to be avoided. Using a columnar table layout is appropriate. Basically the databases are supposed to be used in out-of-the-box versions as extra tuning might not be available for both database instances the same way and the test is not supposed to compare different tuning mechanisms.
Queries are supposed to be expressed in a way, that both database systems can handle the same expres-

sion as good as possible. Furthermore the queries have to be repeatable and comprehensible and have to be logged or generated in a way that reproduction is explicit.
The combination of queries and workload should try to emphasize the differences in the database systems and should therefore be designed to point out strengths and weaknesses of both instances.
The measurement of response time may not include the time necessary for query generation or the writing of result-sets. Therefore the machine running the test tool shall not be changed during tests and only the execution time of the SQL Statement is to be measured.

4.2 Design of tests

Environmental Factors
To meet the above mentioned requirements for simulating environmental factors the Queries are supposed to be executed by {1, 5, 10, 20, 40} Threads in parallel on a catalog_sales table of sizes {5GB, 10GB, 20GB, 40GB, 60GB, 80GB}. Those values were chosen as they can be computed in a relatively moderate time and are based on almost constant growth-rate. The high volumes of data are selected as they are close to (60GB) or even above (80GB) the available main memory (64GB).
In addition to the catalog_sales table the catalog_returns table is used with a constant size of 0.5 GB.

Queries
One test consists of six queries which can be grouped into three categories. The first category consists of queries which perform very basic and simple SQL statements like selects or unions. The second group of queries are either inserting lines or updating tables and therefore represent OLTP workload. The third group includes rather expensive queries including joins and calculations.
It has to be mentioned, that the queries are not designed to perform their tasks in the most efficient way, but to generate workload on the database. The six queries are:

- Insert lines: a specified number of Lines {1, 50, 100, 150, 200, 250} is inserted into the catalog sales table at once. This query was chosen to evaluate the write performance for a high number of columns.
- Update one: this query updates one column in the catalog_sales table for a specific collection of items. This helps to evaluate update performance of single column and creates OLTP workload.
- Update six: this query updates six different columns of the catalog sales table for a predefined set of items. Here the update performance of multiple columns is tested.
- Select all: this statement simply returns the whole catalog_sales table and gives insight about the read performance of multiple columns.
- Select distinct: this statement returns the distinct values of item ids in the catalog_sales table and therefore provides insights about the read performance of a single column, what should be especially beneficial for a column store database.
- Frequent item sets: this set of queries identifies Items which occur together in different orders a specific at least a predefined number of times. Through the implementation using an increasing number of self joins of the table catalog sales, join performance and the power of the internal query optimizer can be evaluated.
- Profit loss per catalog page: this query calculates based on the catalog_sales and catalog_returns table which page of a catalog caused which profit and what amount of loss. Here the performance of calculation operations such as sum() can be compared.

5 Test implementation

As mentioned before the tool suite was written in Java and uses JDBC to connect to the different instances. The architecture of the tool is rather simple, as one Class "Executer" executes the whole test suite by receiving Query-Strings of the "Generator" classes and starts the desired amount of threads "Runners".

Those threads can be executed in a variable and configurable order and amount, allowing to generate a mixed workload, too.

Response time is measured by storing the current system time before executing a statement and immediately after the result is received.

In the following, examples of the queries are provided and some difficulties identified are described.

Insert lines: For this query the disk storage database and the in-memory database have to use different SQL-statements. For the classical database the typical "insert into table values (val1),(val2)..." statement is used. Unfortunately the tested in-memory database is not able to process such a query with more than one value. A workaround proposed, is to write the query as "insert into table (Select value1 from dummy union all Select value2 from dummy ...). But this statement does not allow to insert more than around 250 lines at a time as the query is then told to be too complex to be processed. To avoid primary key constraints, each thread accesses its own file generated from one big TPC-DS catalog_sales data file. After one iteration, when all threads have finished, the data is deleted and the insert begins again.

The update one query is defined as "Update catalog_sales Set cs_warehouse = {a randomly generated integer of equal size for each Thread} where cs_item_sk = {predefined value} OR cs_item_sk =

…". The update six query is written accordingly but sets five more columns {cs_ship_mode_sk, cs_call_center_sk, cs_promo_sk, cs_ship_hdemo_sk, cs_ship_cdemo_sk} to new values. Those columns were chosen as they have no constraints and are not used by any other query.

"Select * from catalog_sales" is the statement executed by the select all statement and "Select distinct cs_item_sk from catalog_sales" is the query behind select distinct.

More interesting and causing a really heavy workload on the database is query set for identifying frequent itemsets.
The queries are generated by one thread executing queries in the style of "Select cs1.cs_item_sk as Item1, cs2.cs_item_sk as Item2 from catalog_sales cs1 inner join catalog_sales cs2 on cs1.cs_order_number = cs2.cs_order_number where cs1.cs_item_sk < cs2.cs_item_sk AND group by cs1.cs_item_sk, cs2.cs_item_sk having count (cs1.cs_order_number) > {predefined value}" with an increasing amount of items, until no result is returned. Those queries are then stored in a list and executed one another by the "Runner". For each of these queries the time is locked. Unfortunately, the in-memory database stops when calculating itemsets with three items, while the classical database exceeds this number.

The profit loss per catalog page query is defined as:
"Select page_sk, sum (sales_price) as sales_price_sum, sum(profit) as profit_sum, sum(return_amt) as returns_sum , sum (net_loss) as loss_sum From (select cs_catalog_page_sk as page_sk, cs_sold_date_sk as
date_sk, cs_ext_sales_price as sales_price, cs_net_profit as profit, cast(0 as decimal(7,2)) as return_amt, cast(0 as decimal(7,2)) as net_loss from catalog_sales
union all select cr_catalog_page_sk as page_sk, cr_returned_date_sk as date_sk, cast(0 as decimal(7,2)) as sales_price, cast(0 as decimal(7,2)) as profit, cr_return_amount as return_amt, cr_net_loss as net_loss from catalog_returns) group by page_sk". Here a number of calculation operations are executed.

6 Project Outcome

6.1 Number of threads

The analysis shows that the DDB handles an increasing number of threads better than the MMDB. The query execution time increases slower with the DDB.

The impact of an increasing number of threads depends for both databases on the data set size. With a 10 GB data set, an increasing number of threads has more impact on the MMDB than on the DDB. With a 20 GB data set, the DDB is more susceptible to an increasing number of threads.

Overall, the performed test showed that DBB performs better with increasing number of threads. The impact the number of threads has on the time duration a query rises with increasing data sets. However, when increasing the size of the data set after 20 GB the impact seems to be constant when increasing the number of threads.

Regarding the predictability of time duration for queries the DDB shows better results than the MMDB irrespective of the size of the data set.

6.2 Size of dataset

With the 5 GB data set the MMDB outperformed the DBB, particularly for complex queries and a high number of threads.

For the 10 GB data set the DBB was superior to the MMDB for all types of queries.

For the 20 GB data set the MMDB excels again, for almost all types of queries.

The DBB performs best for ad-hoc queries, while the MMDB is better for complex data mining queries.

An advantage of the MMDB for incremental sizes of data sets is to be expected as here the gains of main memory and column based storage have increased impact. A verification with a 40 GB data set was for reasons of time limitation not possible.

It can be seen that across all analyses the use of MQT is a lot more efficient than the use of a view operating a MMDB.

The analysis of different types of queries could not show any benefit using one or the other database.

6.3 Summary

Table 1: Assumptions and their fulfillment

ID	Assumption	Evaluation
E 1.1	The impact on the average speed of the queries for increasing number of threads is smaller on a MMDB than on a DDB	◔
E 1.2	With increasing size of the data set the ability to operate many parallel threads has a greater impact	◑
E 2.1	The time duration of a query execution is better predictable on MMDB than DDB	○
E 3.1	The average query is faster on a MMDB than a DDB	◑
E 3.2	For increasing size of data the speed benefit increases	●
E 4.1	Query properties can be matched to database categories to enlarge or diminish the advantages / downsides	◔

The crucial factors appear to be the size of the dataset and, yet less significant, the number of threads.

Overall, the MMDB did not meet the assumptions made at the beginning of the tests (s. Table 1). Especially the expected advantages of parallel execution of queries with multi threads as well as the assumed better predictability of the time duration of a query execution were not evident. Only the assumption that with increasing data sets the speed benefits would also increase was confirmed by the performed tests.

7 Conclusion

A main memory database brings advantages with increasing amount of data. Especially complex data mining queries are most likely to benefit from the properties of a MMDB.

For iterative OLAP queries, the number of threads is the key factor on whether a MMDB is beneficial.

Another finding is the less precise predictability for MMDB compared to a disk based database.

References

[1] SAP AG, "The Global Information Technology Report 2012: Harnessing the Power of Big Data in Real Time through In-Memory Technology and Analytics", 2012.

[2] Frank Niemann, "In-Memory-Datenanalyse in Zeiten von Big Data," 2012.

[3] H. Chen, R. H. L. Chiang, and V. C. Storey, "Business Intelligence and Analytics: From Big Data to big Impact," in MIS Quarterly, pp. 1165–1188.

[4] SAP AG, SAP Solution Brief Business Analytics: SAP® In-Memory Appliance (SAP HANA™) The Next Wave of SAP® In-Memory Computing Technology, 2011.

[5] S. Rogers, "BIG DATA is Scaling BI and Analytics," Information Management (1521-2912), vol. 21, no. 5, pp. 14–18, 2011.

[6] M. Courtney, "Puzzling out Big Data," Engineering & Technology (17509637), vol. 7, no. 12, pp. 56–60, 2013.

[7] H. Garcia-Molina and K. Salem, "Main memory database systems: an overview," IEEE Transactions on Knowledge and Data Engineering, vol. 4, no. 6, pp. 509–516, 1992.

Full Text processing using SAP HANA

(Author)
Jevgenij Jakunschin
University of Wismar
JevJaku@gmail.com

(Supervisor)
Prof. Dr.-Ing. Antje Düsterhöft
University of Wismar
Antje.Duesterhoeft@hs-wismar.de

Abstract

This project seeks to create an evaluation of full text search and related features of the SAP HANA database and other No-SQL and in-memory databases.

1 Introduction

In today's environment of rapidly evolving technologies and database systems, with the boom of NO-SQL databases and the rising problem of the "big data" issue, new projects are rapidly created, merged, changed and even aborted.

Most new systems are very bare bone, offering little possibilities and description of their capabilities for specific tasks, e.g. full text search processing.

This project, as part of a master thesis, seeks to test different systems, primary SAP HANA, on their possibilities, performance, stability, scalability and precision in big data full text tasks. It also aims to compare the algorithmic base and provide a graphical representation of the results.

During the course of this project, it will be also extended with information on the performance of different modern NO-SQL and in-memory databases in order to provide an in-depth comparison of different systems, their performance and features.

2 Project state

A large amount of full text data is required in order to pursue the goals of the project.

A big database of books and articles (initially in *.txt format) with irregular file sizes has been prepared in order to perform such tests.

The books have been collected, using the Project Gutenberg (http://www.gutenberg.org/) database. Only books in .txt format and (mostly) mostly English books have been selected for the tests. Overall a database of 11.531 files, with a total size of 4.34 gigabytes has been collected and prepared for the tests.

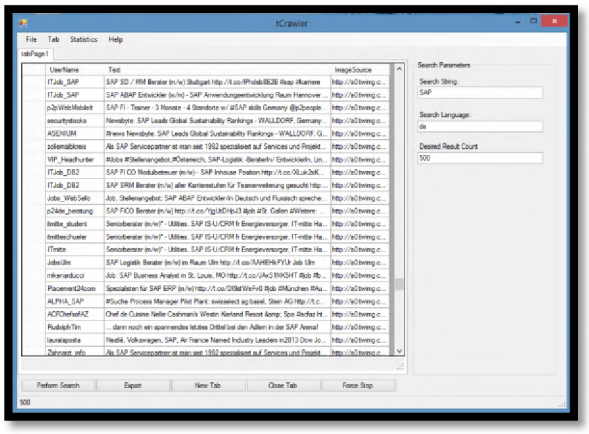


Figure 1: twitter crawler

Afterwards these files have been run through a special application to remove special signs, HTML encoding, restrict line length and fix some other formatting problems that might be encountered during the tests. The program also merges files into Microsoft Excel Comma Separated Values (CSV) format to prepare them for the FTP upload into the database.

The application also provides the possibility to split and merge files, in order to test the optimal table length and storage strategy.

The database already provides sufficient material in order to make performance tests.

However, the project also contains a small application to quickly gather full text data from gather from a small twitter crawler for special tests, like fuzzy search efficiency, performance and precision. Initially the data from both applications is (initially) stored in a 3-column CSV file, consisting out of an ID-key value, a line-value for splitting long texts and the text field.

Additional fields for evaluations and tests are added from the SAP HANA studio during the progress.

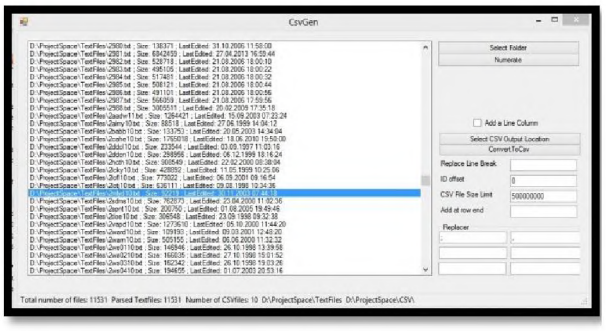


Figure 2: Gutenberg->Csv Converter

In addition, a "troublemaker" application has been designed to create, recorded edits in different files in order to test the fuzzy search possibilities and precision of a database.

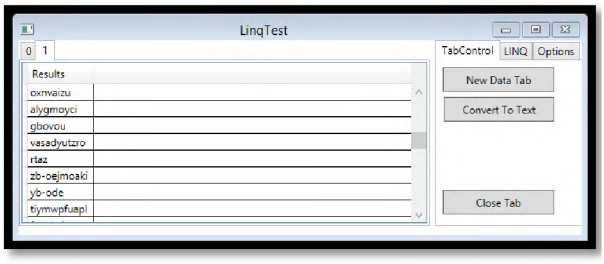


Figure 3: "Troublemaker" application

2.1 Running tests

The text data from the resources above is uploaded to the SAP HANA database using the SFTP protocol and is then imported from the SAP HANA Studio. The text fields are indexed with the "text" (or "short text") type and the tables are being prepared for the following tests:

The results are going to be evaluated statistically upon the following criteria:

- Supported features
- Indexing options and their performance, features and limits (short text and text)
- Performance (full text search, insert statements, fuzzy search upload etc.,)
- Stability and accessibility
- Scalability (depending on cell size, table size, document size and search complexity)
- Comparison to SQL-Standards
- Other statistics

In addition the following full text specific criteria are being tested:

- Fuzzy Search possibilities
- Fuzzy Search precision, false positives, false negatives, optimal factor
- Syntax analysis options
- Language options
- Indexing options
- Full text search limits

Over the course of the project, different databases will undergo the same processes to create a statistics database.

Currently the project is done with the phase of data collecting, converting and manipulating and the first tests and test data with SAP HANA are being prepared and uploaded.

The database will be tested on the features described above, by performing different queries and measuring and comparing the results.

Part of the tests also includes searches with different databases sizes and interpolating and approximating the response times in order to predict the scalability of the project with full-text searches with an automated indexing.

A graphical interface is designed in order to quickly automate and represent different findings for a database in chart form and compare its performance against other databases with similar setup.

The charts will also represent the performance, precision and scalability relation in comparison to different table sizes, text indexes and text lengths.

2.2 Next Steps

As the SAP HANA part of the project is pretty young, a multitude of steps is still to come.

Other databases are going to be tested in a similar manner, as described previously.

The project also seeks to test databases with different text languages in order to evaluate the syntax analysis options.

The troublemaker application will be used in order to test the capabilities of different databases setups to work with flawed data, using algorithms like fuzzy search.

Finally a big part of the project is to test different full text storage and indexing strategies in order to minimize the performance loss, coming from non-optimal storage and scheme approaches.

2.3 Conclusions

The project is in the state of gathering it's most vital data right now and will be ready to present it's findings soon.

Since the project is in the critical phase, where it is ready to perform the required tests and draw conclusions, I would also like to ask for an extension.

Raising the power of Ensemble Techniques

Christoph Engels, Christoph Friedrich, David Müller
University of Applied Sciences and Arts Dortmund, Department of Computer Science
Emil-Figge-Str. 42, D-44227 Dortmund
christoph.engels@fh-dortmund.de, christoph.friedrich@fh-dortmund.de,
david.mueller@fh-dortmund.de

Abstract

Ensemble methods (like Random Forests, Quantile Forests, Gradient Boosting Machines and variants) have demonstrated their outstanding behavior in the domain of data mining techniques. Some outstanding characteristics are presented in [5, 9]. This project aims to raise these potentials in the powerful HANA envi-ronment. In principle there are two alternatives for reaching this objective: Using the function primitives of HANA PAL to build an ensemble or transferring a subset data sample to an R server.

1 Project idea

Predictive statistical data mining has evolved further over the recent years and remains a steady field of active research. The latest research results provide new data mining methods which lead to better results in model identification and behave more robustly especially in the domain of **Predictive Analytics**. Most an-alytic business applications lead to improved financial outcomes directly, for instance demand prediction, fraud detection and churn prediction [2, 3, 7, 8, 10, 11]. Even small improvements in prediction quality lead to enhanced financial effects. Therefore the application of new sophisticated predictive data mining techniques enable business processes to leverage hidden potentials and should be considered seriously.

Especially for classification tasks Ensemble Methods (like Random Forests) show powerful behavior [5, 6] which includes that:

- they exhibt an excellent accuracy
- they scale up and are parallel by design
- they are able to handle
    - thousands of variables
    - many valued categorials
    - extensive missing values
    - badly unbalanced data sets
- they give an internal unbiased estimate of test set error as primitives are added to ensemble
- they can hardly overfit
- they provide a variable importance
- They enable an easy approach for outlier detection

We have tried to transfer these techniques to the HANA environment where two options in R [1] or HANA PAL [12] seem to be feasible (Figure 1).

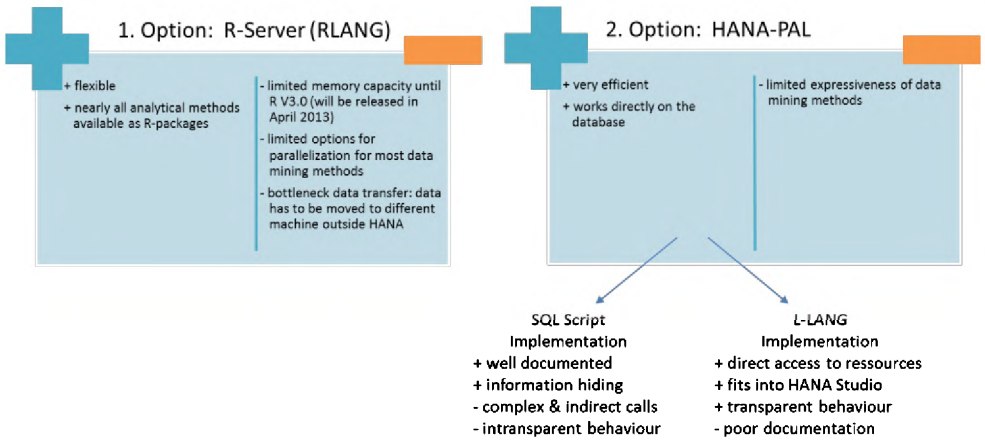


Figure 1: Comparison of Analytic Options

Because the second option makes use of the computing power of the HANA hardware and minimizes data transfer from the database to the analytic function we are in favor of option 2. (following the paradigm: "bring the analytic function to the data"!). The project will try to implement option 2 and option 1 (which seems to be straightforward) and will compare both results by application to example classification problems.

2 Used Future SOC Lab resources

The project has needed a HANA environment (HW and SW) with the latest PAL distribution available and an R installation optionally.

An exclusive access was necessary for performance measurements at the end of the project for approx. five nights.

3 Findings

3.1 Analysis of standalone weak learners

The basis of ensembles are weak learners in form of decision trees in our case. On the PAL side we use the C4.5 decision tree algorithm from the library with variations of learning parameters while on the R side we apply the C5.0 implementation with default settings. The PAL C4.5 shows some restrictions in terms of flexibility w.r.t. an ensemble usage: Pruning is mandatory and randomization on node level is not possible yet. It turns out (see Annex) that the PAL C4.5 implementation compared to a similar R approach shows significant longer training times by a magnitude. The reason is given by the fact that this implementation is relatively new and programmers have not taken the full potential of the HANA architecture [4]. The accuracies of both approaches on different datasets are similar with some variations of the results due to training parameter variations.

3.2 Analysis of ensembles

Ensembles are created by using weak learners and the principles of randomization, boosting and bagging. In our case we have to apply some simplifications due to the limited flexibility of the PAL C4.5 decision tree implementation: Randomization can be done on a tree level instead of a node level only and pruning of trees has to be accepted which is omitted in standard ensemble methods. Additionally sampling with replacement is hard to implement and is done without replacement here instead. Two datasets are used (OPTICAL_REC and KRKOPT). While the results in OPTICAL_REC are comparable in accuracy, there is a large deviation in the KRKOPT dataset with poor performance in the case of two attributes per tree in accordance with default setting (#attributes per split = sqrt(#attributes)), see Annex. It seems that these default settings on the node level cannot be transferred on tree level for datasets in general. If we increase the number of attributes per tree the results are improving. It is remarkable that the PAL ensemble method (RF) in the OPTICAL_REC dataset shows significantly superior accuracy than the PAL decision tree (DT).

4 Next steps

In the recent research and project period we have investigated the behavior of both options followed by a comparison. It could be shown that there are some open issues concerning the one to one mapping of the R-server approach to HANA-PAL:

- Performance issues
- Mandatory Pruning of Decision Trees
- Missing Randomization of Input Attributes
- Sampling with Replacement

Due to these unexpected observations during the project the extension of ensembles by using Stochastic Gradient Boosting [16] could not be achieved within the first project period. Furthermore it turns out that the provision of ensembles will need dedicated SQLScript programming skills which makes the application of PAL methods for "normal" users difficult. Additionally we have experienced limited expressiveness of the SQLScript wrapper call technique.

5 Extension

The main objective of this project is to address the remaining methodic work regarding ensemble methods by implementing Stochastic Gradient Boosting and to simplify the allocation of ensemble methods by providing a graphical user interface.

In order to deal with the situation from the last spring period we propose the following activities:

5.1 Activity "Migration"

Migration of ensemble technique to native L-LANG. At the moment L-LANG shows some restrictions like limited reusability and no direct support for parallelism. This restrictions will be resolved probably within next weeks and month, so this project is able to make use of these improvements.

5.2 Activity "HANA Studio Embedding"

Embedding the ensemble functionality in the HANA Studio Workflow Tool as L-LANG node due to the agreement with [4].

5.3 Activity "Stochastic Gradient Boosting"

Add the Stochastic Gradient Boosting method [13] into the ensemble creation. Because the weak learners within the ensemble will be learned sequentially, the algorithm is able to use the information gained during

the training of the initial learners for an improved data selection for the remaining learners.

5.4 Conclusions

The implementation of HANA PAL functionality still undergoes a steady extension. As a consequence the comparability of different approaches and options in implementing Ensemble Techniques remains difficult. Nevertheless we expect a richer and more complete set of functions especially in the PAL context in the near future such that remaining differences between PAL and R will vanish over time.

References

[1] J. Aswani; J. Doerpmund: "Advanced Analytics with R and SAP HANA", Slideshare. Retrieved 2012-03-14 (2012).

[2] R. E. Banfield; R.E., el. al.: "A Comparison of Decision Tree Ensemble Creation Techniques", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29, No. 1 (2007).

[3] S. Benkner, A. Arbona, G. Berti, A. Chiarini, R. Dunlop, G. Engelbrecht, A. F. Frangi, C. M. Friedrich, S. Hanser, P. Hasselmeyer, R. D. Hose, J. Iavindrasana, M. Köhler, L. Lo Iacono, G. Lonsdale, R. Meyer, B. Moore, H. Rajasekaran, P. E. Summers, A. Wöhrer und S. Wood: „@neurIST Infrastructure for Advanced Disease Management through Integration of Heterogeneous Data, Computing, and Complex Processing Services", DOI:10.1109/TITB.2010.2049268, IEEE Transactions on Information Technology in BioMedicine, 14(6), Seite 1365 - 1377, (2010).

[4] J.-H. Böse, SAP Innovation Center Potsdam, personal communication, Aug. 2013.

[5] L. Breiman: „RF / tools – A Class of Two-eyed Algorithms", SIAM Workshop http://www.stat.berkeley.edu/~breiman/siamtalk2003.pdf , (2003).

[6] L. Breiman: "Random Forests", http://www.stat.berkeley.edu/~breiman/random-forests-rev.pdf, (1999).

[7] C. Engels; W. Konen: „Adaptive Hierarchical Forecasting". Proceedings of the IEEE-IDACCS 2007 Conference, Dortmund (2007).

[8] C. Engels: „Basiswissen Business Intelligence.", W3L Verlag, Witten (2009).

[9] J. Friedman: „Computational Statistics & Data Analysis", Volume 38, Issue 4, 28 February 2002, Pages 367–378, http://dx.doi.org/10.1016/S0167-9473(01)00065-2, (2002).

[10] M. Hülsmann; D. Borscheid, C. M. Friedrich und D. Reith: "General Sales Forecast Models for Automobile Markets and their Analysis", Transactions on Machine Learning and Data Mining, Volume 5, Number 2, Seite 65-86, 2012.

[11] G. Üstünkar; S. Özögür-Akyüz; G. W. Weber; C. M. Friedrich und Y. A. Son, „Selection of Representative SNP Sets for Genome-Wide Association Studies: A Metaheuristic Approach", DOI:10.1007/s11590-011-0419-7, Optimization Letters, Volume 6(6), Seite 1207-1218, (2012).

[12] PAL, https://help.sap.com/hana/hana_dev_pal_en.pdf (2013).

[13] G. Ridgeway: „Generalized Boosted Models: A guide to the gbm package". http://cran.r-project.org/web/packages/gbm/vignettes/gbm.pdf, (2007).

Annex: Detailed results

The following tables summarize the results of our experiments with decision trees and ensembles over decision trees in terms of performance and crossvalidated accuracy (DT means Decision Tree, RF abbreviates the ensemble method Random Forest. Datasets can be found at http://archive.ics.uci.edu/ml/datasets.html.

Dataset	DT / RF	PAL / R	rows used	Inputrows Training	Parameter Training	Performance (overall) for RF on PAL	Performance (Training) (R C5.0 / PAL C4.5)	SIZE DT R C4.5 / PAL C4.5)	Inputrows Testing	Parameter Testing	Performance (Testing) (R C50 / PAL C45)	Accuracy (R C50 CROSS / PAL C45 CROSS)
Dataset IRIS (150rows)	**DT**	PAL	150 (all)	94	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2	--	95 ms	12	56	THREAD = 16	34 ms	Accuracy = 0,94 ++ = 0,98 -- = 0,91
		R	150 (all)	94	--	--	13 ms	7	56	--	6ms	Accuracy = 0,92 ++ = 0,98 -- = 0,86
Dataset KRKOPT (28056rows)	**DT**	PAL	28.056 (all)	1.7675	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2	--	7,08 sec	23.237	10.381	THREAD = 16	2,23 sec	Accuracy = 0.585 ++ = 0.595 -- = 0.576
		R	28.056 (all)	1.7675	--	--	1.13 sec	6.017	10.381	--	332 ms	Accuracy = 0,63 ++ = 0,635 -- = 0,621
	RF	PAL	28.056 (all)	Pool 1.7675 random	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2 tree = 100 Attribute per Tree = 2	18 min	--	-	10.381	THREAD = 16	--	Accuracy = 0,272
		R	28.056 (all)	Pool 1.7675 random	tree = 100	--	14,17 sec	--	10.381	--	231 ms	Accuracy = 0,68

Dataset	DT / RF	PAL / R	rows used	Inputrows Training	Parameter Training	Performance (overall) for RF on PAL	Performance (Training) (R C5.0 / PAL C4.5)	SIZE DT R C4.5 / PAL C4.5)	Inputrows Testing	Parameter Testing	Performance (Testing) (R C50 / PAL C45)	Accuracy (R C50 CROSS / PAL C45 CROSS)
Dataset COVERTYPE (581.012rows)	**DT**	PAL	581.012 (all)	366.037	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2 with ranges	--	3:54 min	185.368	214.974	THREAD = 16	10.1 sec	Accuracy = 0.693 ++ = 0,694 -- = 0,692
		R	581.012 (all)	366.037	without ranges	--	2:18 min	(C5.0, not C4.5) 10.266	214.974	--	27,5 sec	(C5.0, not C4.5) Accuracy = 0.936 ++ = 0,937 -- = 0,935
Dataset Pokerhand (1.025.010rows)	**DT**	PAL	1.025.010 (all)	645.756	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2	--	1:18 min	387.382	379.254	THREAD = 16	22 sec	Accuracy = 0.928 ++ = 0.929 -- = 0.927
		R	1.025.010 (all)	645.756	--	--	81.284 sec	(C5.0, not C4.5) 27.517	379.254	--	29.86 sec	(C5.0, not C4.5) Accuracy = 0.842 ++ = 0,858 -- = 0,83
		R	1.025.010 (all)	Pool 645.756 random	tree = 100	--	158 min	--	379.254	--	35 sec.	Accuracy: 84,54
Dataset CONNECT4 (67.557rows)	**DT**	PAL	67.557 (all)	42.560	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2	--	18.41 sec	21.778	24.997	THREAD = 16	896 ms	Accuracy = 0,777 ++ = 0,789 -- = 0,76
		R	67.557 (all)	42.560	--	--	12.437 sec	4.420	24.997	--	5.144 sec	Accuracy = 0,794 ++ = 0,8 -- = 0,783
Dataset OPTICAL_REC (5.620rows)	**DT**	PAL	5.620 (all)	3.540	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2	--	3.7 sec	7.258	2.080	THREAD = 16	562 ms	Accuracy = 0.61 ++ = 0.62 -- = 0.60
		R	5.620 (all)	3.540	--	--	898 ms	289	2.080	--	327 ms	Accuracy = 0,892 ++ = 0,9 -- = 0,877
	RF	PAL	5.620 (all)	Pool 3.540 random	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2 tree = 100 Attribute per Tree = 8	29 min	--	--	2.080	THREAD = 16	--	Accuracy = 0,91
		R	5.620 (all)	Pool 3.540 random	tree = 100	--	6.7 sec	--	2.080	--	74 ms	Accuracy = 0,975
Dataset GLASS (214rows)	**DT**	PAL	214 (all)	135	THREAD = 16 SPLIT MODEL = 1 PMML EXPORT = 2 MIN_REC = 2	--	160 ms	102	79	THREAD = 16	99 ms	Accuracy = 0.642 ++ = 0.684 -- = 0.595
		R	214 (all)	135	--	--	21 ms	37	79	--	11 ms	Accuracy = 0.697 ++ = 0,768 -- = 0,618

Integration of a VEE-Framework within a Smart Gateway into SAP HANA

Jan-Patrick Weiß
University of Oldenburg
Department of Computer Science
Uhlhornsweg 84
D-26129 Oldenburg
jan-patrick.weiss@uni-oldenburg.de

Benjamin Reinecke
University of Oldenburg
Department of Computer Science
Uhlhornsweg 84
D-26129 Oldenburg
benjamin.reinecke@uni-oldenburg.de

Marco Lucht
University of Oldenburg
Department of Computer Science
Uhlhornsweg 84
D-26129 Oldenburg
marco.lucht@uni-oldenburg.de

Jad Asswad
University of Oldenburg
Department of Computer Science
Uhlhornsweg 84
D-26129 Oldenburg
jad.asswad@uni-oldenburg.de

Jan Hendrik Wege
University of Oldenburg
Department of Computer Science
Uhlhornsweg 84
D-26129 Oldenburg
jan-hendrik.wege@uni-oldenburg.de

Christoph Walther
University of Oldenburg
Department of Computer Science
Uhlhornsweg 84
D-26129 Oldenburg
christoph.walther@uni-oldenburg.de

Abstract

The energy market is changing by legislation so by 2020 smart meters will be installed in most households. Therefore the IT systems of energy companies have to adjust to deal with the growing amount of measured data. With smart meters it is possible to measure the consumption every 3 minutes and shorter instead of measuring only one annual value. This near real time data consumption allows companies to create new business use cases. They can for example predict the usage of their grids and close short time contracts with suppliers to meet the demands. Furthermore there are new possibilities for analysis and assurance of data quality. In order to assure a defined integrity, possible missing or false values of the available measuring data have to be corrected. The project deals with these problems of data quality and handling of these massive data. To ensure integrity and the best quality of data, a VEE-Framework (Validation, Estimation, Editing) will take a place. The VEE-Framework shall be implemented within a Smart Gateway and integrated into SAP HANA to benefit from the efficiency of in-memory-database technology.

1 Introduction

This document shows the main idea of the project "Integration of a VEE-Framework within a Smart Gateway into SAP HANA". At first the business context of smart metering in modern households is described. After that an explanation of the VEE-Framework will demonstrate how smart meter data can be used and how data is being precessed. In the part "objective target" the goals of the project are described. Furthermore the reasons to use SAP HANA for this project are explained. At the end new insights and further steps and outlook are shown.

2 Smart Metering

Smart Metering describes a method to count, document and analyze the consumption (electricity, gas, water etc.) by using electrical meters. The measurement data can be recorded in individual time inter-

vals (minutes, hours, days etc.) and is automatically submitted to the measurement data responsible, which will submitted to the supplier, which can use this information e.g. for monitoring or billing processes. Smart meters enable two-way communication between the meter and the suppliers' system. Furthermore they can be used to gather information for remote reporting. When using additional technical functions, the consumer is able to monitor his consumption and to make constant analyzes. Energy hogs can be identified and the use of energy-intensive devices can be switched to time frames, where energy is less expensive. That may lead to decrease energy costs. Furthermore smart meters can be part of a smart grid.

3 VEE-Framework - Validation, Estimation, Editing

Metering data from different sources is a very extensive process which needs to provide a large amount of functionality and robustness, not only hardware but software based. However, potential errors can never be barred and have to be treated respectively. Therefore a process, called **V**alidation, **E**stimation and **E**diting (VEE) is introduced. The main objective is to offer a set of methods which allow a partly automated approach of specific error handling based on predefines rules. This allows business users to identify erroneous and potentially problematic data before it is used by downstream applications. It is important to state that all changes that are made to the considered data have to be logged in order to reconstruct them in the future.

In a first step, the source dataset needs to be validated in order to determine inconsistencies, e.g. missing values, duplicates or spikes. Therefore the necessary meter data has to be identified, collected and versionized. Afterwards, several tests like Time Tollerance-, Sum- and Spike Checks are peformed and evaluated on the specific values. The various test types are identified and selected automatically via configurable parameters within the meter read. Depending on the test results, the individual values need to be marked as failed and subsequently further processed.

During estimation phase, all previously marked values shall be adapted and corrected in order to correspond to their specific requirements. Therefore several appropriate replacement values are generated by choosing a specific set of customer based methods and applying them to the read meter data. These rules can be assigned to individual customers as well as to groups of various customers and are furthermore exchangeable in any order. It is important that those values which have been estimated or treated in any way are marked accordingly in order to identify them for later use.

The third step of VEE process implies the actual editing of the available meter data. This includes writing the individual replacement value into the data set as well as setting the status parameter to its designated value (see figure 1).

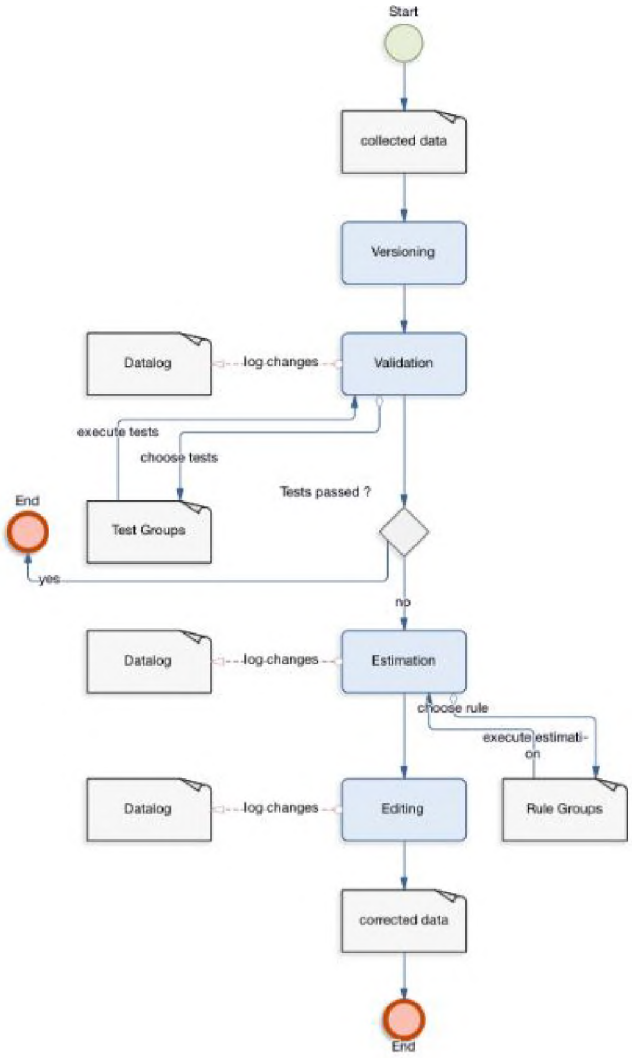


Figure 1: VEE process

Due to the customizable sets of rules, the automatic data validation and estimation and the consecutive logging functionalities, the VEE process guarantees an accurate and quick verification of read meter data which is a necessary assumption for various business use cases.

4 Objective Target

The project group is aiming to develop an SAP-HANA appliance which fits into the existing systems used for measuring customers' energy consumption and creating the customers' invoices. Therefore the following steps shall be taken:

1. When meter-values for a certain customer are requested by SAP e.g. for creating an invoice, SAP HANA loads the data (from the PI-Server or the CX4U Smart-Gateway).

2. The loaded consumption values from the gas meters are checked for faulty or even missing values (either within the Smart Gateway or in SAP HANA).

3. If such a value is found, it is being replaced by a calculated value which was calculated by a certain pattern fitted to the customer.

4. The generated value is logged to a file, so that the invoice stays transparent.

5. Both, calculated and measured values are taken to SAP so that the invoice can be created.

As the calculation method for faulty and missing values depends on the customer and can be changed, the calculation in SAP HANA is just temporary and the values are only passed to the system which requested them and will be deleted afterwards.

5 Reasons to use SAP HANA for this project

SAP HANA is used to optain high performance in processing large amounts of heterogeneous data. By connecting households to smart meters, the massive data volumes increase, since there is not as yet, an annual settlement, but a possibility to measure every 15 minutes, or at even shorter intervals, energy consumption data. Every Smart Meter generates 768 bytes of data per day - with a base of 15,000 meters in the used demonstration system, we are talking about approximately 330 gigabytes of data in an accounting period (normally a month) [1].
In order to forecast electricity consumption for a day, we have to process nearly 11 gigabytes of data. With the available data; analysis, forecasting and quality of results can be increased. In order to generate settlements it is necessary to have correct and complete values. SAP HANA supports this process with statistical functions to find and eliminate missing and non-valid data. This functionality is also called VEE (Validation, Estimation, Editing). When using SAP HANA for processes such as analysis or prediction calculations, it is necessary to integrate the functionalities of SAP HANA with the VEE-Framework within the Smart Gateway.

6 New Insights

Implementing the VEE Framework within SAP HANA and connecting SAP HANA with Smart Gateway leads us to new insights through benefiting from real time processing of the in-memory computing technology. SAP HANA with its simplicity and easy to handle interfaces and functions presents a full database platform in addition to its analytical and transactional capabilities within the attribute, the analytic and the calculation views and other artifacts.

However, despite the capability of HANA to interconnect with other SAP or non-SAP systems or databases, there is always latency in loading and exporting the metering data into and from SAP HANA. These two operations are not automated and time consuming, which is very crucial to our process. In order to handle the data transfer the project group has to provide a solution that transfers the metering data fast, efficiently and accurately.

7 Further Steps and Outlook

Currently the project group cooperates with CX4U Consulting and analyses the data which is provided by the product *SmartGateway*. It is important to understand how the interfaces of the *SmartGateway* work to get the necessary data into SAP HANA. The next steps are to show - within a proof of concept - in which way the VEE-Framework can be implemented into SAP HANA, and in which way SAP HANA can be connected or intgrated with Smart Gateway. There should be several alternatives and the best one should be shown in a prototype.

References

[1] C. Kuhlmann. *E-Mail, 06.08.2013*. CX4U AG, Oldenburg.

Next Generation Operational Business Intelligence
exploring the example of the bake-off process

Alexander Gossmann
Research Group Information Systems University of Mannheim
agossman@mail.uni-mannheim.de

Abstract

Large retail organizations have to plan customer demands accurately, to achieve customer satisfaction and loyalty. The primary objective is to avoid out-of-shelf situations. On the other hand, losses of perished goods, especially in case of fresh food, have to be minimized. The handling of the trade-off between availability and loss can be dramatically improved by a real-time analytic system. The challenge is to analyze large amounts of data (big data), typically derived from the transactions in the retail process, enhanced by external data, like weather and holidays. Different management groups require specific information with short response times at reasonable costs. Transferred to the retail domain, local store managers are focused on operational decision making, while top management requires a view on the business at a glance.
Both requirements rely on transactional data, whereas the analytic views on this data differ completely. Thus different data mining capabilities in the underlying software system are targeted, especially related to processing masses of transactional data.
The examined software system is a SAP HANA in-memory appliance, which satisfies the aforementioned divergent analytic capabilities, as will be shown in this work.

Introduction (Project Idea)

Operational Business Intelligence is becoming an increasingly important in the field of Business Intelligence, which traditionally was targeting primarily strategic and tactical decision making [1]. The main idea of this project is to show that reporting requirements of all organizational levels (operational and strategic) can be fulfilled by an agile, highly effective data layer, by processing directly operative data. The reason for such architecture is a dramatically decreased complexity in the domain of data warehousing, caused by the traditional ETL process [2]. This requires a powerful and flexible abstraction level of the data layer itself, as well as the appropriate processability of huge amounts of transactional data.
The SAP HANA appliance software is currently released in SPS 05. Important peripheral technologies have been integrated, such as the SAP UI5 Presentation Layer and the SAP Extended Application Services, a lightweight Application Layer. This project proves the tremendous possibilities offered by this architecture which allows a user centric development focus.
This report is organized in the following chapters. The first chapter provides a general overview of the explored use case. In the second chapter the used resources will be explained. The third and fourth chapters contain the current project status and the findings. This Document concludes with an outlook on the future work in the field.

1 Use Case

This project is observing a use case in the field of fast moving goods of a large discount food retail organization. Specifically, the so called bake-off environment is taken into account. Bake-off units reside in each store and are charged with pre-baked pastries based on the expected demand. The trade-off between product availability and loss hereby is extremely high.
From the management point of view, the following user group driven requirements exist: On the one hand, placing orders in the day to day business requires accurate and automated data processing, to increase the quality of the demand forecast. On the other hand, strategic decision makers need a flexible way to drill through the data on different aggregation levels, to achieve a fast reaction time to changing market conditions.
The observation period of two years is considered. The basic population consists of fine grained, minute wise data for thousands of bake-off units, providing all facts related to the bakery process.

1.1 Store Level Requirements

On the store level, the store manager will be supported with matters regarding daily operational demands. Primarily for order recommendations, a certain amount of historical data is taken into account to satisfy the appropriate statistical calculation on time series. Additionally location related and environmental information increases the accuracy of the forecasting model. Environmental variables, like historical weather and holidays, are considered in correlation with historical process data to improve the forecast model. Furthermore, forecasted weather data and upcoming holidays are taken into account for ex-ante

data in order to improve the prediction. Model fitting and operational data analysis are being processed ad hoc and on demand by the appropriate store manager.

1.2 Corporate Level Requirements

On the corporate level a 'bird's eye view' is the starting point, where highly aggregated key figures indicate business success or problems. These measures deliver information on a very high level, whereas the reasons for the appearance of these indicators can vary strongly. For accurate decisions it is tremendously important to drill down to the line level, to indicate the reasons for certain business patterns. As the strategic reporting is based on one common data foundation of operational data, navigation to the line level is implicated. It is important that the system is having user satisfying response times, allowing the exploration of a huge amount of data. The application provides the detection of certain patterns and correlations for a more complex classification. For example, the daily availability is analyzed based on certain thresholds, provided by minute wise real time data. To sum up, real-time enabled reporting on strategic level allows reactions on market changes to reach an unprecedented level of effectiveness.

2 Project set up

This chapter illustrates the used technology. After a listing of the architectural resources the appropriate implementation domains will be described in more detail.

2.1 Used Resources

As stated in the introduction, the used architecture is based on the SAP HANA Appliance Software SPS 05 [3].
The presentation layer is built upon the HTML5 based framework SAP UI5. The communication with the SAP HANA In-Memory database and user handling is established through SAP Extended Application Services (XS Engine). Data intensive calculations and data querying are handled by the appropriate APIs in the database, such as the calculation engine (CE), the SQL engine, the Application Function Library (AFL), and particularly the Predictive Analytics Library (PAL) [4].
For time series analysis the Rserve based R integration is used. The data load of CSV formatted transactional data, as well as the data replication and 3rd party are implemented in Java and imported through the JDBC API. The considered 3rd party data consists of weather data, as well as school and public holidays.

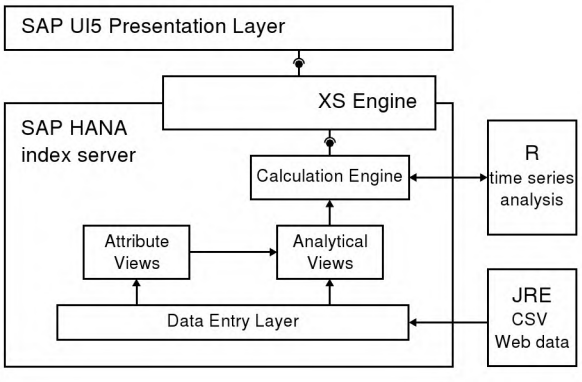


Figure 1: Architecture

The used architecture, as described in the following chapters, is summarized in Figure 1.

2.2 SAP Front End

As stated above the SAP UI5 constitutes the presentation layer. The Model View Controller pattern is being conducted for front end implementation. For web and mobile versions of the application two different view variants are implemented.
The entry point for a specific user group is the login screen, whereas the different management roles are distinguished by specific HANA user roles. The user groups are differentiated into the strategic, tactical, and operational management role. The strategic and tactical roles are showing the same reports, restricted by the related aggregation level. On the operational level, completely different reports are provided and are mainly focused on daily analysis. Additionally, order recommendations for the next three days are visualized. Each report relies on *one* associated calculation view, described later in the back end section. The selection parameter invoked by a user are handled by OData services, with the belonging data binding, or manually by SQL Script calls.

2.3 SAP HANA Back End

The HANA in-memory database is the core technology of this investigation. In the following section, the data model will be shortly discussed.
The data entry layer consists of two main fact tables. One fact table contains daily aggregated sales related key figures. The second fact table consists of minutely wise measures, derived from the bakery process. This fact table has an expected cardinality of approximately two billion records. In the current implementation this table has rounded 500 million records for the first testing runs. An appropriate partitioning policy, based on time related range partitioning is conducted here, in regards to the expected limit of 2 billion records per table. Several master data tables contain information about stores, regions, products, and holidays. Historical weather data is stored in an appropriate table, whereas weather forecasts will be stored separately and merged daily into the historical data table. All tables are implemented as column tables.

Upon this data entry layer several attribute views are implemented, building up the product, store and regional dimensions. The time dimension is based on the generated time table with minutely level of granularity (M_TIME_DIMENSION), provided by HANA standardly.
The two analytical views contain the fact tables, whereas the daily based fact table is additionally enhanced by the weather and holiday dimensions.
Based on this multidimensional data model eight calculation views are implemented, to satisfy user reporting scenarios about availability, loss, and sales on tactical and strategic level. Additionally one calculation view provides reporting needs on operational level, showing the relevant process information of the current and previous days.
For more sophisticated data mining on the strategic level, as well as data preprocessing of time series data, PAL is used [5]. Specifically the linear regression model function is used to draw trends of dynamically aggregated sales data over time. Further the anomaly detection function is used for outlier detection in daily sales data.

2.4 Peripheral technology

The load of historic and transactional data is handled by a proprietary Java import module, using the JDBC API. The reason for this implementation mainly relies on huge amount of heterogeneous CSV formatted files. Approximately two hundred thousand different types of CSV files have been imported into the HANA database. Therefore, a special bulk load strategy has been used, especially in spite of the insert properties of column oriented tables in the entry layer. Furthermore, historic weather data as well as weather forecast and holiday data is loaded via the JDBC interface of the import module.

Holidays

Both school and public holidays have been downloaded for the past two years, and until the year 2015 from the online portal 'Schulferien.org'. The data is available in the iCal format, and covers all dates for the different states of Germany. These files were loaded into the HANA index server, after conversion into CSV format, using the appropriate build in wizard.

Weather

The historical weather data has been imported from the web weather API 'wonderground.com'. For model training of the forecast module, the corresponding time interval values of daily, city wise consolidated store data was called from the API. This results in approximately one million JSON files (one file corresponds to one data record), generated by the REST interface, afterwards converted into CSV format and loaded via JDBC of the import module.

Forecast

The demand forecast requirements are primarily developed using the R environment. The appropriate time series are generated on demand, and invoked by a store manager who is responsible for one's store. As stated in the previous section, the time series are being preprocessed in advance by the PAL framework primarily for performance reasons.
The important outlier detection and handling have been additionally implemented in the R environment, as here more advanced algorithms are available in the R community. Furthermore, two different forecast models have been utilized for comparison reasons. The ARIMA (Auto Regressive Integrated Moving Average) model as well as the ANN (Artificial Neuronal Network) based model have been observed.

2.5 Development environment

The eclipse based HANA Studio is used as the main IDE for the development. In addition to the newly introduced SPS05 features, regarding the 'HANA development' perspective, the Java import module is implemented as well.
For usability reasons the following implementation strategy of the R environment has been utilized: Each developer uses a local R runtime for coding R script and model testing. The appropriate time series data is supplied through the ODBC interface. After finalizing a model in R, it is transformed into the HANA environment using the RLANG extension in SQL Script [5].
All artifacts, including java classes, java script, UI5 artifacts and R script have been set under version control with git [6].
The prototype has been completely redesigned regarding the SAP HANA components. The SAP HANA repository has been used for this purpose, to store all relevant design time artifacts like:

- hdb tables
- hdb roles
- procedures

3 Findings

This chapter contains findings on technological as well as on the process level. The findings will be explained analogous to the outline of the previous chapter. In conclusion the outcome of this project will be summarized.

3.1 SAP Front End

Through the tight integration of the controller and model layer, the presentation layer profits of the advantage of a high abstraction level. The data binding feature of the OData services is especially beneficial for strategic and tactical reporting. Hereby flexible data navigation for the top management user is provided, by selecting free time intervals and break-

ing down into different products, regions, or stores. Never the less, the store management invokes an ad hoc data mining and forecasting capability by calling a SQL Script procedure through a Java Script DB connection call.
For parameterization of the calculation views the following limitations exist:

- exclusively input parameters are used, instead of variables for performance reasons
- for input parameters, no ranges are supported and graphical calculation views require additional filter expressions
- character based date parameters work with the OData interface (thus no type safety is provided, implicit cast)

3.2 SAP HANA Back End

In the previous chapter (2.3) the data model has been explained. The biggest column based table contains two years of minutely based transactional data. It has been partitioned by regions. The response times of the appropriate calculation view calls are absolutely satisfying . Nevertheless, the following main restrictions have been experienced which are listed by the appropriate domain:

Predictive Analytics Library

- usability of PAL functions is inconvenient and non-transparent
- restrictive parameterization policy
- very limited exception handling

The restriction in the design time usability especially in the case of PAL, compromises the performance experience of the data analysis.
The AFL framework is in a relatively early stage of maturity and in this project context, only few functions could be utilized. The major functionality in the area of time series analysis has been conducted in the R environment, as stated in the next section.

3.3 Forecast

The demand forecast for each store is calculated on demand. The appropriate time series is generated and sent, together with the belonging weather and holiday information to the R runtime. Hence the sent data frame to R contains daily related time series derivates of additional environmental data to the historic sales data for a certain pastry and store.

Time Series Preprocessing (Outlier Adjustment)

For comparison reasons the outlier detection is being performed with PAL and in a second scenario in R. In case of PAL preprocessing, the outliers are marked in the time series data frame provided to R. The used PAL function is PAL_ANOMALY_DETECTION with the following parameterization:

Parameter	Value	Comment
THREAD_NUMBER	4	
GROUP_NUMBER	3	number of clusters k
OUTLIER_DEFINE	1	max distance to cluster center
INIT_TYPE	2	
DISTANCE_LEVEL	2	
MAX_ITERATION	100	

Table 1: PAL parameterization

As depicted in Table 1, the used PAL function uses a k-means cluster algorithm, whereas GROUP_NUMBER corresponds to the number of associated clusters (k). Please note that this function detects always one tenth of the underlying number of lags in each time series as outliers. This could not be controlled by the parameter OUTLIER_PERCENTAGE, as expected and thus, limits this function enormously. In the R environment the k-means clustering for outlier detection is used as well. A straightforward approach of outlier handling is used. The majority of given outliers belongs to the class of additive outliers due to public holiday related store closing. The effect is even more significant, the longer a closing period is. Here the precedent open business date shows an abnormal high characteristic. Other outlier classes are by far less significant and cannot be assigned directly to events. Different outlier handling strategies have been tested and implemented, and will be investigated in further proceedings.

ARIMA based forecast

An automated ARIMA model has been implemented in R. The used package is mainly the package 'forecast' [7] available at CRAN (Comprehensive R Archive Network [8]). The automated ARIMA fitting algorithm 'auto.arima()' [9] has been utilized for this project purposes, which is based on the Hyndman et al algorithm [10]. Specifically seasonality, non-stationarity, and time series preprocessing (see outlier handling) required manually coded model adjustment. All additional predictor variables like holidays and weather information could be processed automatically, passed by the 'xreg' matrix parameter.

ANN based forecast

Alternatively to the ARIMA approach, an Artificial Neuronal Network model has been implemented and is especially for capturing automatically nonlinear time series shapes. As expected in the retail context, ANN is supposed to deliver more accurate forecast results [11]. In this use case the 'RSNNS' [12] (Stuttgarter Neural Nets Simulator [13]) package has been utilized. Similarly to the ARIMA model (see above), the independent variables, primarily the daily sales an all additional related variables are used for model fitting.

Summary Forecast with R

One major design change has been made; the R runtime has been transferred to another server. As the previous solutions has been running together with the HANA instance on a virtual machine with 64 cores the parallelized ARIMA based forecast used all available resources on the Linux server. This is not recommended by SAP, as it could harm the processes of the HANA instance itself. Thus, an R runtime on a separated server is obligatory. It can be stated that the performance behaves nearly inversely proportional to the number of cores for the ARIMA algorithm, proposed above. The performance of retrieving, serialization and deserialization of the data frames is nearly neglectable (in the area of ms). Nevertheless, different loading and presentation strategies are required, to provide user acceptance in response times. For instance, asynchronous XSJS calls could be performed, to avoid persisting trained models. This is especially true for ANN algorithms which are only tenuous parallelizable.

3.4 Conclusion

The built prototype was expected to satisfy the reporting requirements of the different stakeholders of information consumption. Although the data analysis capabilities differ throughout the organizational roles of managers, all human recipients expect short response times of a system. With the usage of the SAP HANA appliance software this challenging task could be achieved.

From the development perspective, previously not known effectiveness could be achieved. As all reporting and predictive analytics requirements rely on only a few physical tables, the main effort consists in providing different views on this data. Even more complex measure calculations, like availability and some regression analysis, are processed on the fly. This is a completely new way of designing a reporting system. Compared to traditional ETL based data warehousing tools this saves a lot of manual effort in the loading process. However, this does not imply that the effort for implementing the business logic disappears, merely that the programming paradigm is straightforward. SAP constantly improves the appropriate API functionality (e.g. by introducing the 'HANA Development' perspective), whereas the programming framework is not matured yet.

The capability of providing demand forecasts based on long time series intervals for thousands of stores and different products particularly supports operational decision makers on the day to day business. This could not, or only very difficultly, be achieved with traditional disk based data warehouse approaches focused on aggregated measures. In this prototype, forecast algorithms are performed on demand. This makes sense, as the underlying models require readjustments with each new transaction.

4 Outlook

The focus of the current work was set on the implementation of an analysis tool, processing masses of data in real-time for bake-off use case. To provide further information on the applicability of this approach, in the fresh food industry, an additional use case will be observed. Fresh cakes have a sell-by date of only one day in this case. Thus, the restriction is even more rigid, compared to bake-off, as unbaked pastries can remain frozen for a longer period.

Both, the reliability of the demand forecast as well as the reusability of SAP HANA components will be observed.

References

[1] C. White: The Next Generation of Business Intelligence: *Operational BI. DM Review Magazine.* Sybase 2005.

[2] H. Plattner: A common database approach for OLTP and OLAP using an in-memory column database. *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data.* ACM, 2009.

[3] SAP HANA Developer Guide. *help.sap.com/hana/hana_dev_en.pdf* , 21st of December 2012.

[4] SAP HANA Predictive Analysis Library (PAL) Reference. *help.sap.com/hana/hana_dev_pal_en.pdf,* 23 of January 2013.

[5] SAP HANA R Integration Guide. *help.sap.com/hana/hana_dev_r_emb_en.pdf,* 29th of November 2012.

[6] *http://git-scm.com/.*

[7] *http://cran.r-project.org/web/packages/forecast/forecast.pdf.*

[8] *http://cran.r-project.org/.*

[9] *http://otexts.com/fpp/8/7/.*

[10] Hyndman, Rob J., and Yeasmin Khandakar. Automatic Time Series for Forecasting: The Forecast Package for R. No. 6/07. Monash University, Department of Econometrics and Business Statistics, 2007.

[11] Doganis, P., Alexandridis, A., Patrinos, P., & Sarimveis, H. (2006). Time series sales forecasting for short shelf-life food products based on artificial neural networks and evolutionary computing. *Journal of Food Engineering*, *75*(2), 196-204.

[12] *http://cran.r-project.org/web/packages/RSNNS/RSNNS.pdf.*

[13] *http://www.ra.cs.uni-tuebingen.de/SNNS/.*

Using SAP ERP and SAP BW on SAP Hana

A mixed workload case study

Galina Koleva
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
galina.koleva@in.tum.de

Stephan Gradl
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
stephan.gradl@in.tum.de

Helmut Krcmar
Technische Universität München
Chair for Information Systems
Boltzmannstr. 3, 85748 Garching, Germany
krcmar@in.tum.de

1 Abstract

In-memory databases promise to be able to cope with a mixed workload in comparison to relational databases. The aim of this project is to perform a load test to compare the performance of an ERP and BW system running on relational and in-memory databases. To achieve this, a reference process has been identified and implemented using the Rational Performance Tester (RPT) to simulate virtual users that generate load on the system In an experimental environment a performance test of an SAP ERP system using a relational database has already been conducted. Due to open issues concerning the heterogeneous system copy, the performance test of the SAP ERP on HANA system will be addressed within the next project phase.

2 Introduction

According to a study from the consulting company IDC from 2012, more than three quarter of IT responsible persons expect a yearly data growth up to 25 percent within the next two years. Thirteen percent even expect data growth of 25-50 percent [1]. Business users are used to have information "at their fingertips" [2], comparable to Google searches. However, most businesses lack the technical possibility of providing information in real-time [2]. This results in the question, how huge data volumes can be processed in future. One possibility to cope with this challenge is the use of in-memory databases. They promise processing of huge real time data volumes within seconds. This speed advantage can bear a fundamental competitive advantage [2].

An in-memory database system (also referred as main-memory database system) stores data primary within the main-memory. No I/O accesses to the hard drive are necessary which results in performance boost ([3, 4, 5, 6, 7]). This can bear a fundamen-tal advantage regarding the separation of BW and ERP systems.

Enterprise resource planning (ERP) systems form the backbone for the execution, controlling and management of business processes in today's companies and are therefore vital for companies' business processes. In order to prevent the performance of OLTP-queries in those systems from being throttled by complex and time consuming OLAP processes, those applications have been separated into different kinds of systems like SAP ERP and SAP BW systems. In-memory databases promise the possibility of using both types of applications on the same database.

3 Project Implementation

3.1 Project Goal

The main purpose of our research project is to evaluate the mixed workload on SAP HANA based on existing SAP ERP and BW case studies. Therefore, a performance comparison between an ERP and BW

system running on an in-memory database (HANA) and on a relational database (DB2) was conducted.

3.2 Project Phases

The project was structured in three phases. The first phase consisted of the development of a performance framework to be able to conduct the experiment. It contains the database related KPIs to measure performance, as well as reference process that is conducted by agents within the SAP systems. The agents were simulated by the Rational Performance Tester (RPT) from IBM.

The next phase was the allocation of the ERP and BW systems on an in-memory database, as well as on a traditional database. Therefore a heterogeneous system-copy containing the GBI-dataset was conducted.

The third step was the execution of the performance test and the comparison of the results.

3.3 First Phase: Creating a framework containing KPIs and a reference processes

Meanwhile, the performance measurement on the standard ERP system was prepared, as the study contains a comparison of SAP ERP on HANA to SAP ERP on a relational database (in our case DB2). The first step was the identification of key performance indicators (KPI) for the comparison of the ERP system on two different databases. Users of transactional systems like ERP systems perceive the response time as most critical. Response time is defined as the time that collapses between an input to the system until its reaction. Due to the different hardware and resource settings, we analyzed the components of the response time with respect to their hardware dependency and their significance for the description of database performance. As a result we created the measurement infrastructure to record overall database time, kind and number of database queries, number and amount of enqueue actions. The amount of available measurement data for each step in the workload is shown in the first illustration. It is an excerpt from transaction STAD within the SAP system containing measured performance values.

Analysis of time in work process

CPU time	350 ms	Number	Roll ins	0
RFC+CPIC time	0 ms		Roll outs	0
			Enqueues	7
Total time in workprocs	1.582 ms			
		Load time	Program	13 ms
Response time	1.582 ms		Screen	0 ms
			CUA interf.	0 ms
Wait for work process	0 ms			
Processing time	326 ms	Roll time	Out	0 ms
Load time	13 ms		In	0 ms
Generating time	0 ms		Wait	0 ms
Roll (in+wait) time	0 ms			
Database request time	1.242 ms	Frontend	No.roundtrips	0
Enqueue time	0 ms		GUI time	0 ms
			Net time	0 ms

Analysis of ABAP/4 database requests (only explicitly by application)

Connection	DEFAULT	Request time	1.242 ms
Database requests total	1.011	Commit time	17 ms
DB Proc. Calls	0	DB Proc. Time	0 ms

Type of ABAP request	Database rows	Requests	Requests to buffer	Database calls	Request time (ms)	Avg.time / row (ms)
Total	2.850	1.011	401	336	1.242	0,4
Direct read	12	335	229		6	0,5
Sequential read	2.713	652	172	211	1.208	0,4
Update	3	3		3	1	0,3
Delete	0	0		0	0	0,0
Insert	122	21		122	27	0,2

Figure 1: excerpt of available statistical KPIs within an SAP system

The illustration presents the components of the response time and provides a detailed view on the queries processed by the database and the supporting buffers. These queries are divided in 5 types:

- Direct read: A direct read access to a single tuple (ABAP command „select single“)
- Sequential Read: read access to a number of tuples
- Update: Modifying access to an existing tuple
- Delete: Deletion of an existing tuple
- Insert: Insertion of a new tuple in a database table.

Both for each type and the overall sum the following data is provided:

- Database rows: number of accessed tuples
- Requests: number of SQL requests to the database
- Requests to buffer: number of SQL request that has been answered by the corresponding buffers
- Database calls: number of executed operations (Select, Insert, Update und Delete)
- Request time (ms): duration of request
- Avg.time / row (ms): average request duration per affected tuple.

Out of all available numbers, the ones related to the database underneath are selected. The value „Request time (ms)“ consists only of requests to the database. As the database response time is by magnitudes greater the duration of request processed by the buffers is not included in this sum. In addition the number and the execution time of so called „Stored Procedures“ is shown using the field „DB Proc Calls“. Finally the time needed for committing database transactions is provided by the field „Commit time“.

The second required preparation for the experiment is the selection of a reference process, which should be conducted by many agents to perform the performance test. A reference process created within the SAP University Competence Center has been chosen, as it is used by more than 400 universities worldwide. The second reason is, because it contains different modules, like Production planning and control (PP), Controlling (CO) and Materials Management (MM). It furthermore contains the Material Requirements Planning Run (MRP), which is a relatively complex run and requires a huge data amount. The MRP run calculates the minimum lowest possible material and product levels in store at any point in time based on the bill of material, the stock level and the incoming orders. We expected significant response differences for this run. An overview of the total process is provided by figure 2:

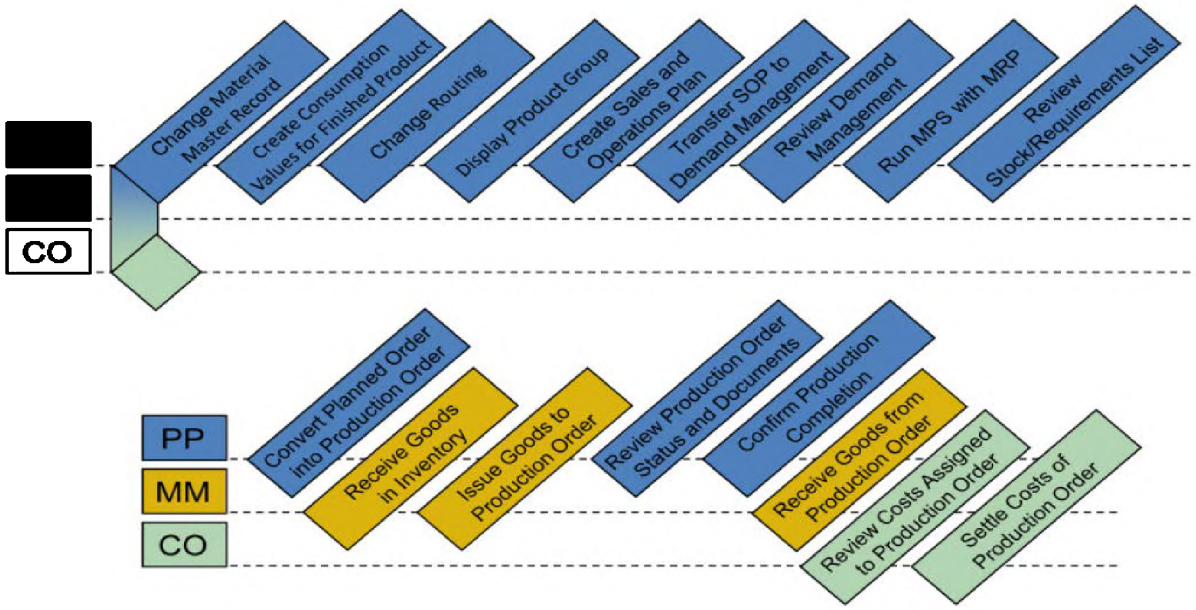


Figure 2: Chosen reference process

This mentioned reference process is conducted by 500 virtual concurrent users. Therefore, the Rational Performance Tester is used for recording, testing, and executing workload with 500 virtual users (VU) on SAP ERP/BW on HANA and SAP ERP/BW on DB2.

After the workload has been recorded and the variables have been defined, the recording has to be tested. This is done to reduce the number of errors while mass execution. If tests have been conducted repeatedly with prospected results, schedules for operation and users groups can be created. This can emulate the capacity that is needed for the huge amount of VUs. SAP batch input test can be added as well to the schedule to simulate workload and capacity peaks. Simultaneously the virtual test resources can be minimized to forecast different system behavior.

Furthermore performance schedules can combine and repeat several SAP performance and batch input tests or assign them to different user groups. Global test preferences, like resource monitoring of certain parameters, can simulate reflection periods of VUs or configure performance requirements. User groups may consist either of host systems themselves or of previously configured RAC agents. Not only the total or relative amount of test participants of user groups is easily measurable with parameters, but also the possibility to order the chronology of test runs is given. In this case, test can be repeated with loops in three different ways: finite, infinite or time-controlled.

Creating implementation profiles leads to an efficient realization of the performance schedule. In this case it is possible to configure, if the SAP GUI is inserted on the host system during testing. Each virtual user interacts with an SAP GUI of an own instance according to a user guide so that the agents can pull initially the time-scheduled processes of the workbench of the host system. The SAP servers provide and record the results for response times. Test points and each of their results are recorded as well. The following graphic shows the architecture and the components offering these functionalities

After the test run, results are visualized in a detailed report in the development environment. So results like parameters of the resource monitors, performance requirements or statistics to batch input test are only listed if they had been configured before. The tool also supports the measurement of the already mentioned performance measures.

As there are still some issues with the data import, the test on ERP on HANA with the GBI dataset has not been conducted yet on the ERP/BW on HANA system.

company with a use case similar to IDES. The upgrade and export of this system showed neither errors nor further problems. However, during the import phase on the HANA database severe errors in four

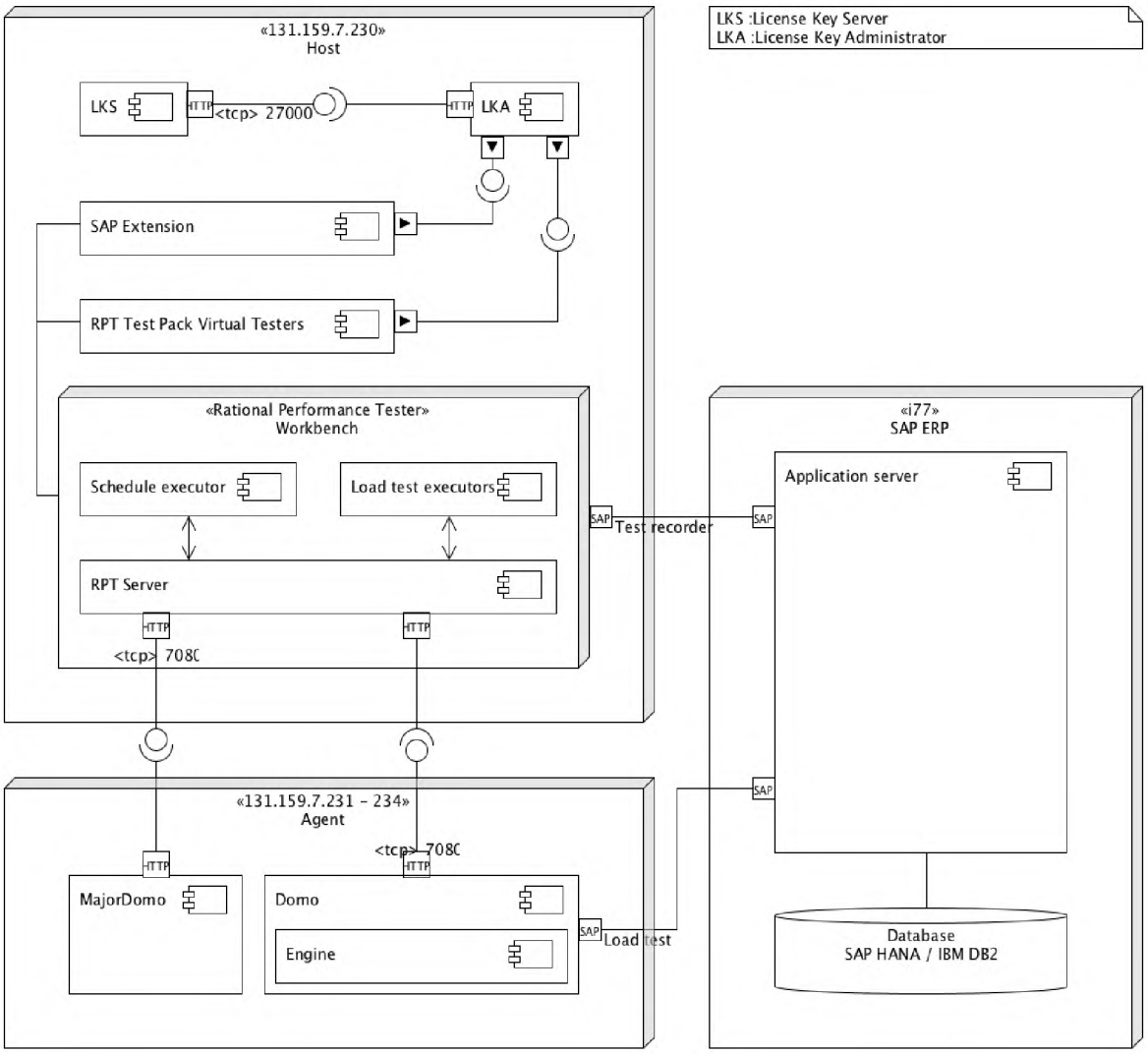


Figure 3: Overview of the Rational Performance Tester Architecture

3.4 Second Phase: Allocation of the Systems

The first step was the migration of an existing SAP ERP on HANA and an SAP BW on HANA system including the necessary datasets for the intended workload. This was planned to be done using the heterogeneous SAP system copy procedure. One prerequisite for doing so is an upgrade to SAP enhancement package 6 (EhP 6). Due to problems with the upgrade of SAP IDES systems, we decided to focus on the SAP ERP system with Global Bike Inc. (GBI) dataset. GBI is an exemplary dataset, of a

packages occurred. The log files are being analyzed further at this point, but no solution to this problem has been found yet.

4 Conclusion

The goal of this study was to test the load performance of an ERP/BW system on HANA in comparison to the same system running on a relational database. The first step was to create a reference frame-

work to identify which KPIs were only related to database behavior, as well as to create a reference process which will be conducted by many virtual users on the different systems. The reference framework contains the KPIs: request time (ms), DB Proc Calls, Enqueue Time and Commit Time. As a reference process for the ERP system a typical production planning process was selected, as it includes different modules, as well as the MRP run. The MRP run is a complex calculation of required material at different point in times and needs a huge data amount.

The next step was the system copy to migrate the SAP ERP on HANA. Therefore, the ERP software had to be updated to the latest enhancement package (EhP) first. Errors regarding the upgrade and the export phase were resolved successfully. However, there are still some issues with the import phase of the ERP system with an exemplary dataset (GBI). As these errors are not resolved yet, we applied for a further project within the HPI Future SOC Lab to be able to reach our project goals.

5 References

[1] IDC (2012): IDC-Studie: Big Data in Deutschland 2012 - Unternehmen stehen noch ganz am Anfang. http://www.idc.de/press/presse_idc-studie_big_data2012.jsp, accessed at 01.02.2013.

[2] Plattner, H.; Zeier, A. (2011): In-Memory Data Management: An Inflection Point for Enterprise Applications, Springer 2011.

[3] Loos, P.; Lechtenbörger, J.; Vossen, G.; Zeier, A.; Krüger, J.; Müller, J.; Lehner, W.; Kossmann, D.; Fabian, B.; Günther, O. (2011): In-Memory-Datenmanagement in betrieblichen Anwendungssystemen. In: Wirtschaftsinformatik, Vol. 6 (2011) No. 53, pp. 383-390.

[4] Loos, P.; Strohmeier, S.; Piller, G.; Schütte, R. (2012): Kommentare zu „In-Memory-Datenmanagement in betrieblichen Anwendungssystemen". In: Wirtschaftsinformatik, Vol. 54 (2012) No. 4, pp. 209-213.

[5] Stonebraker, M. (2011): Stonebraker on Data Warehouses. In: Communications of the ACM, Vol. 54 (2011) No. 5, pp. 10-11.

[6] Kemper, A.; Neumann, T. (2010): HyPer: HYbrid OLTP&OLAP High PERformance Database System. Institute of Computer Science, 2010.

[7] Strohmeier, S. (2012): Hauptspeicherdatenbanken in der betrieblichen Informationsversorgung - Technische Innovation und fachliche Stagnation? In: Wirtschaftsinformatik, Vol. 54 (2012) No. 4, pp. 209-210.

Using Thread-Level Speculation to Enhance JavaScript Performance in Web Applications

Jan Kasper Martinsen and Håkan Grahn
School of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden
{Jan.Kasper.Martinsen,Hakan.Grahn}@bth.se

Anders Isberg
Sony Mobile Communications AB
SE-221 88 Lund, Sweden
Anders.Isberg@sonymobile.com

Abstract

We have implemented Thread-Level Speculation in the Squirrelfish JavaScript engine, which is part of the WebKit browser environment. We were able to speed up the execution time by a factor of up to 8.4 times compared to the sequential version for 15 very popular web applications on an 8-core computer, without making any modifications to the JavaScript source code. The results also show that we probably could take advantage of an even higher number of cores.

1 Introduction

JavaScript [3] is a dynamically typed, object-based scripting language where the JavaScript code is compiled to bytecode instructions at runtime which in turn are interpreted in a JavaScript engine [2, 14, 9].
One of the most common uses for JavaScript is for interactivity to the client side of web applications. Modern web applications are complex networks of code running on multiple servers and in the client-side web browser. Our intention with this work is to speed up the client-side execution.
Several JavaScript optimization techniques and benchmarks have been suggested, but these benchmarks have been reported as unrepresentative for JavaScript execution in web applications [11, 12, 5]. One important result of this, is that the popular optimization technique just-in-time compilation (JIT) where the JavaScript code first is compiled, then executed as native code *decrease* the execution time for a set of JavaScript benchmarks, while it at the same time *increases* JavaScript execution time for popular web applications [6].
JavaScript is a sequential programming language and cannot take advantage of multicore processors. Fortuna et al. [1] showed that there exists significant potential parallelism in many web applications with a potential speedup up to 45 times compared to the sequential execution. However, they did not implement support for parallel execution in any JavaScript engine.
Web Workers [13] allows parallel execution of tasks in web applications, but it is the programmer's responsibility to extract and express the parallelism. In addition, Web Workers aim is to increase the responsivity of a web application, rather to improve the execution speed of JavaScript execution.
To hide the details of the underlying hardware, one approach is to dynamically extract parallelism from a sequential program using Thread-Level Speculation (TLS). The performance potential of TLS has been shown for applications with static loops, statically typed languages, in Java bytecode environments, and there have been some initial attempts for JavaScript [4].
In [8] we demonstrated an up to 8 times speedup over sequential execution time for 15 very popular web applications [10] on a dual quadcore computer with a TLS enabled Squirrelfish [14] used in the WebKit browser environment. The execution and behaviour of a web application is dependent not only on the JavaScript engine itself, but also on the interaction between JavaScript and the web browser such as manipulation of the Document Object Model (DOM) tree. However, we deliberately focused on the JavaScript aspect.
As we mentioned earlier, Fortuna et al. [1] showed that there exists a 45 times speedup for web applications, therefore we are interested in how well our findings could scale with a larger number of cores.

2 Current Results

In Figure 1 we have measured the speedup with Thread-Level Speculation(TLS) and Just-in-time compilation (JIT) on a set of very popular web applications. The results indicate that:

- *TLS always decreases the execution time*
- *The Squirrelfish JIT based engine increases the execution time* for 10 out of 15 cases (similar to the results in [7])

- *The Google V8 JIT engine increases the execution time* for 8 out of 15 cases

The results in Table 1 show that the maximum number of threads is higher for 14 out 15 use cases, than the number of cores in order dual quad core computer. Therefore there could be a potential to improve execution speed further with a larger number of cores.

Both the Thread-Level Speculation execution time results in Figure 1 and Table 1 follows the postulation of Fortuna et al. [1], that there exists a significant potential for parallelism in web applications. There are numerous reasons for this; first that, JavaScript in web applications makes many function calls. These function calls are often events called from the web application, and they are often anonymous function. These functions are often independent to one another (i.e., do not write or read to the same global variable and return any values), which makes them very suitable for being executed in parallel. Thread-Level Speculation is one technique for doing this, which is suggested in [8].

Our Thread-Level Speculation experiments have been performed on a a dual quad core computer, so there is a limit to how much extracted parallelism this computer can take do. However, interestingly in web applications, we are not lacking of parallelism to take advantage of (Table 1 shows many speculations and few rollbacks). What we are currently missing, is a computer with even more cores. Another interesting feature is that this parallel potential is perhaps not the normal applications, where parallelism is used.

In this respect, the Hasso-Plattner institute gave us access to a computer with 48 cores. We are in the progress of understanding if an increased number of available cores can be used to take advantage of the parallelism of web application, or if there are other factors (of for instance more architectural nature or the execution characteristics in web applications) that prevents us from taking advantage of the large number of cores.

3 Future research

There is a potential to increase the speedup with Thread-Level Speculation, and it would be very interesting to see on a larger number of cores, as the measurements suggests that we could take advantage for more cores. One extension we will do in future studies is that we want to extends the number of use cases for the web application, to make them both larger and study their use with a larger number of users.

Therefore we hope to continue to have access to Hasso-Plattner Institute's wide selection of computers, to future understand the factors of execution of these use cases on a large number of cores, and collaborate with Hasso-Plattner for future and even more interesting results.

References

[1] E. Fortuna, O. Anderson, L. Ceze, and S. Eggers. A limit study of javascript parallelism. In *2010 IEEE Int'l Symp. on Workload Characterization (IISWC)*, pages 1–10, Dec. 2010.

[2] Google. V8 JavaScript Engine, 2012. `http://code.google.com/p/v8/`.

[3] JavaScript. `http://en.wikipedia.org/wiki/JavaScript`, 2010.

[4] J. K. Martinsen and H. Grahn. An alternative optimization technique for JavaScript engines. In *Third Swedish Workshop on Multi-Core Computing (MCC-10)*, pages 155–160, November 2010.

[5] J. K. Martinsen and H. Grahn. A methodology for evaluating JavaScript execution behavior in interactive web applications. In *Proc. of the 9th ACS/IEEE Int'l Conf. On Computer Systems And Applications*, pages 241–248, December 2011.

[6] J. K. Martinsen, H. Grahn, and A. Isberg. A comparative evaluation of JavaScript execution behavior. In *Proc. of the 11th Int'l Conf. on Web Engineering (ICWE 2011)*, pages 399–402, June 2011.

[7] J. K. Martinsen, H. Grahn, and A. Isberg. Evaluating four aspects of JavaScript execution behavior in benchmarks and web applications. Research Report 2011:03, Blekinge Institute of Technology, July 2011.

[8] J. K. Martinsen, H. Grahn, and A. Isberg. Using speculation to enhance javascript performance in web applications. *IEEE Internet Computing*, 17(2):10–19, 2013.

[9] Mozilla. SpiderMonkey – Mozilla Developer Network, 2012. `https://developer.mozilla.org/en/SpiderMonkey/`.

[10] G. R. Notess. Alexa: Web archive, advisor, and statistician. *Online*, 22(3):29–30, 1998.

[11] P. Ratanaworabhan, B. Livshits, and B. G. Zorn. JSMeter: Comparing the behavior of JavaScript benchmarks with real web applications. In *WebApps'10: Proc. of the 2010 USENIX Conf. on Web Application Development*, pages 3–3, 2010.

[12] G. Richards, S. Lebresne, B. Burg, and J. Vitek. An analysis of the dynamic behavior of JavaScript programs. In *PLDI '10: Proc. of the 2010 ACM SIGPLAN Conf. on Programming Language Design and Implementation*, pages 1–12, 2010.

[13] W3C. Web Workers — W3C Working Draft 01 September 2011, Sep. 2011. `http://www.w3.org/TR/workers/`.

[14] WebKit. The WebKit open source project, 2012. `http://www.webkit.org/`.

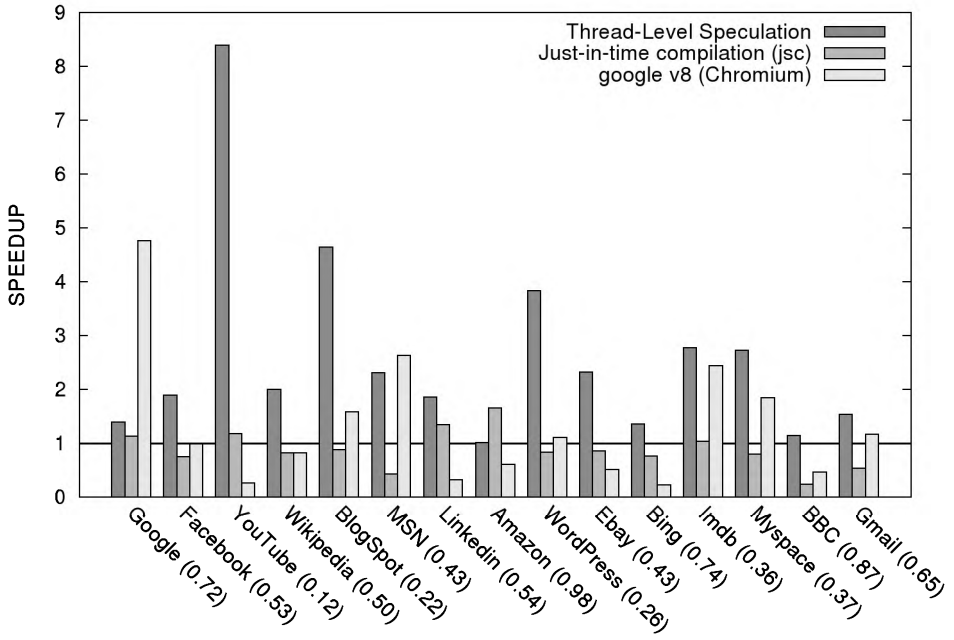


Figure 1: JavaScript execution speedup over the interpreted sequential JavaScript engine for the 15 web applications when TLS is enabled, when just-in-time compilation is enabled and for the Google V8 JIT engine in the chromium web browser. We have written the execution time of TLS relative to the sequential execution time in parenthesis after the name of the web applications.

Table 1: Number of speculations, number of rollbacks, relationship between rollbacks and speculations, maximum number of threads, maximum nested speculation depth, average depth for recursive search when deleting values associated with previous speculations, and average memory usage before each rollback (in megabytes).

Application	Number of speculations	Number of rollbacks	Speculations / Rollbacks	Maximum number of threads	Max speculation depth	Average depth	Memory usage (MB)
Amazon	10768	267	40.31	83	23	8.0	14.1
BBC	6392	154	41.51	117	14	5.12	33.0
Bing	303	18	16.83	30	7	2.22	1.4
Blogspot	778	15	51.87	16	14	2.16	1.6
Ebay	7140	101	70.69	63	15	5.33	27.0
Facebook	968	51	18.98	27	22	9.16	7.1
Gmail	1193	19	62.79	34	10	2.68	1.95
Google	1282	36	35.61	40	10	3.9	5.5
Imdb	5300	156	33.97	54	24	6.85	17.8
LinkedIn	1815	51	35.59	36	11	2.27	7.1
MSN	12012	133	90.32	191	24	5.85	20.1
Myspace	3679	93	39.56	39	14	5.54	17.4
Wordpress	5852	63	92.89	63	99	4.55	9.7
Wikipedia	12	0	*undefined*	8	4	0	1.1
YouTube	7349	25	293.96	407	13	5.44	17.1

Forecasting of Energy Load Demand and Energy Production from Renewable Sources using In-Memory Computing – Project Report –

Witold Abramowicz Tomasz Rudny
Monika Kaczmarek
Department of Information Systems
Faculty of Informatics and Electronic Economy
Poznan University of Economics
Al. Niepodleglosci 10, 61-875 Poznan
Poland
firstname.lastname@kie.ue.poznan.pl

Abstract

This report presents the short overview of the initiatives undertaken in the field of forecasting of energy load demand and energy production using the computational power of SAP HANA. One of the research hypothesis that we focused on, was that the forecasting of demand can be done in quasi-real time, even if conforming to bottom-up approach, i.e., computing separate forecasts for each customer. The report provides information on the project main objectives, used HPI Future SOC Lab resources, findings as well as next steps envisioned.

1 Introduction

The vast amount of data that organizations should gather, store and process, entails a set of new requirements towards the analytical solutions used by organizations. These requirements have become drivers for the development of the in-memory computing paradigm [4], which enables the creation of applications running advanced queries and performing complex transactions on very large sets of data in a much faster and scalable way than the traditional solutions.
The main aim of our work is to examine the analytical possibilities of the in-memory computing solution, on the example of SAP HANA, and their possible applications. In order to do that we apply SAP HANA and its components to the challenge of forecasting of the energy demand and production in the energy sector.
Within this report we focus on the forecasting of energy load demand. Forecasting energy demand in real time as well as creation of models featuring a low prediction error is extremely difficult [7]. Therefore, such a scenario requires a solution that supports fast operations on a very large number of data and is equipped with adequate analytical modules. These requirements correspond to SAP HANA.
As shown by many researchers and practitioners energy load forecasting can be done more accurately, if forecasts are calculated for all customers separately and then combined via bottom-up strategy to produce the total load forecast [2, 3]. Also, additional improvement is expected, if different models and parameters can be tested quickly to provide the best model selection capability. Quasi real time load forecasting could lead to substantial savings for companies and also for the environment, up till now however, it was computationally difficult, as for a typical energy seller a number of residential customers exceeds hundreds of thousands.

2 Project Aims

The goal of the presented project was to design and implement quasi real-time load forecasting for a huge number of individual customers profiles using in memory computing. We formulated a hypothesis: forecasting accuracy (which is directly related to costs) can be substantially increased by individual forecasting on the level of each customer done in quasi real-time using SAP HANA parallelism, and a new insight into the analysed data can be gained.
Throughout the project we aimed at building an analytical system using SAP HANA for support of Energy Load Demand Forecasting. Every energy selling company must accurately predict the load demand in order to prepare - either by own generation resources or by buying on the long and short term market - the supply. Our aim was to evaluate whether SAP HANA analytical capabilities can augment the classical methods of forecasting. On the other hand, we wanted

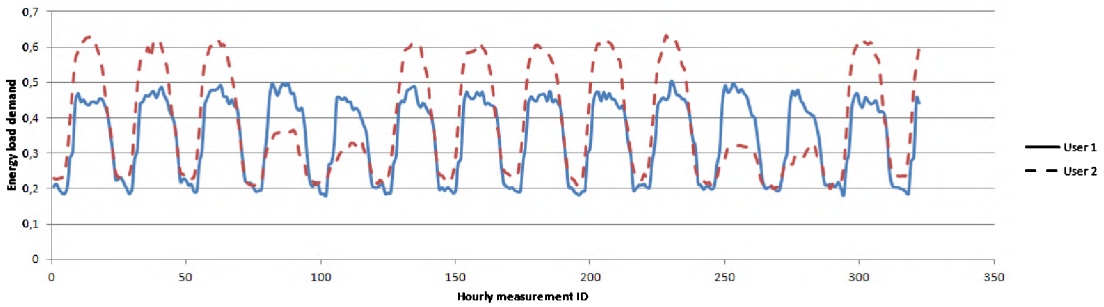


Figure 1: Comparison of the energy load demand of two exemplary users

to check if the outstanding processing power of SAP HANA can lead to new concepts and methods of forecasting. Here, we relied on the bottom-up approach, which has rarely been used till now, also because of the lack of computation power strong enough to harness thousands of time series.

Apart of forecasting our aim was to check how SAP HANA can support daily business analytic task of energy companies for example contract preparation. One of the most important tasks involved is finding a customer profile most similar to the prospect customer, thus fitting the tariff. But since comparisons of demand profile are very expensive in terms of computations, here we saw another place for potential use of SAP HANA.

To achieve our aims we organized the work into the following steps:

1. Designing and setting-up of a database optimized for storage of massive historical load data as well as predictive analysis.

2. Cleansing the data and remove missing values by imputation,

3. Implementing different forecasting strategies based on the bottom-up approach.

4. Testing different forecasting models and exogenous variables.

5. Calculating many forecasts for the same time series data using different parameters and models.

6. Comparing the models to choose the best settings.

We hoped that with the parallel processing and SAP HANA libraries to calculate load demand forecasts, by keeping data in-memory, which will eliminate the costly data transfer from hard drives into memory, which has been one of the key inhibitions of classical analytical software, we will be able to prove that new approach to demand forecasting is now possible.

3 Future SOC Lab Resources Used

During the project we used SAP HANA Studio and the assigned instance (12) combined with Rserve for predictive analyses. SAP HANA offers a way to incorporate the R code directly into the SQL Script [6, 1]. Table variables can be passed on as input parameters, while output parameters can receive values of R data frames. This makes it possible to conveniently use R packages and procedures directly from SAP HANA.

Throughout the project we uploaded the data into SAP HANA as column tables. Then, we rearranged the data for our experiments, creating auxiliary tables and columns. Because the data we obtained from a major Polish energy distributor consisted of only a limited number of times series, we decided to generate artificial data as randomization of the existing one. The randomization modified the values of the energy load demand by 10%. It was performed using the R script procedure.

The experiments and analysis of individual user profiles have shown that indeed, the characteristics of aggregated and individual profiles are different, e.g., they have different shapes, and in addition, that the profiles of individual users also differ among each other. As shown on Figure 1 - both individual profiles show peaks in the morning and evenings, but the shape of the second user vary for different days of the week and parts of the year whereas the first user exhibits a more or less constant energy load demand irrespectively of the day of a week. The individual electricity load profile is influenced by the number of factors, which may be divided in the following categories [3]: (1) electricity demand variations between customers, (2) seasonal electricity demand effects, (3) intra-daily variations, and (4) diurnal variations in electricity demand. The influence of those factors is specific for groups of customers (having similar profiles).

Within the phase of our project addressed by this report, we focused on using SAP HANA Predictive Analysis Library (PAL) [5] both to build univariate time series forecasting models and to build more advanced regression models (see Figure 2).

The following experiments were run on SAP HANA

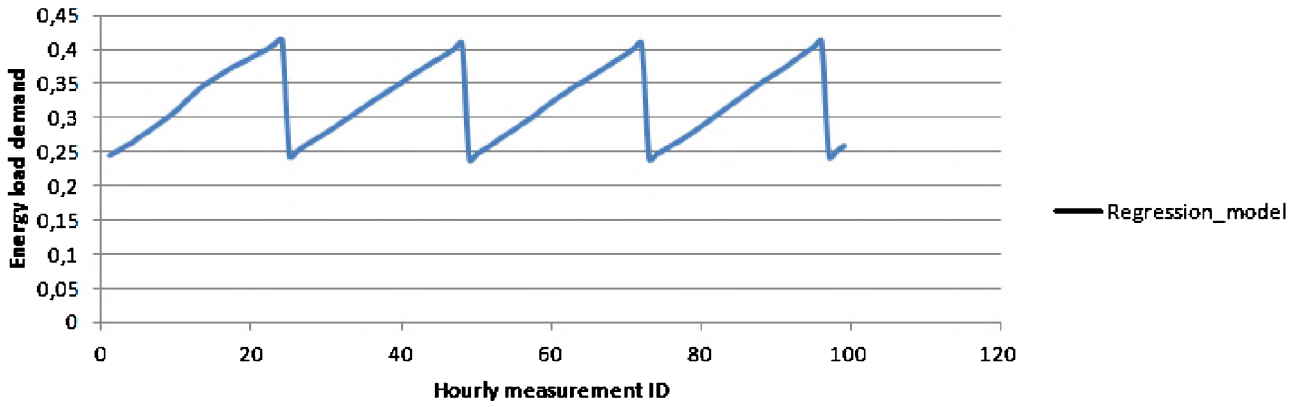


Figure 2: Prognosis of energy load demand using the regression model

using the aforementioned resources:

- Computing the summary forecast by summing all individual time series and calculating the (single) forecast over the summarized data,
- Calculating individual forecasts for all customers, then summarizing the forecasted values to compute the summary forecast,
- Comparing the forecasting error between different scenarios,
- Comparing different forecasting models - among others Holt-Winters exponential smoothing (single and double) and linear regression.

The experiments conducted so far focused on evaluating SAP HANA capabilities for time series forecasting, data manipulation and storage, as well as organizing the code.
We also had to impute the data as sometimes the value of the historical energy load demand was missing. The imputation code is presented below:

```
CREATE PROCEDURE IMPUTE_MISSING(IN x
"totals",OUT y "totals_imputed")
LANGUAGE RLANG AS BEGIN
missing <- is.na(x$value)
n.missing <- sum(missing)
x.obs <- x$value[!missing]
imputed <- x
imputed$value[missing] <-
sample(x.obs, n.missing, replace=TRUE)
y <- as.data.frame(imputed)
END;
```

As already indicated, within the first steps of the research we assigned our focus to the conducting experiments using R. We were able to generate forecasts using various models based on 180 days historical data with a 14 days horizon. The models tested included ARIMA and Holt-Winters exponential smoothing. The code for forecasting can be nicely and neatly written in R as shown on the example of Holt-Winters exponential smoothing:

```
CREATE PROCEDURE FS_STEP(IN x "input2",
IN temp "temp", OUT y "totals_forecast")
LANGUAGE RLANG AS
BEGIN
ts1 <- ts(x$value, frequency=24)
m <- HoltWinters(ts1)
f <- predict(m, n.ahead=14*24)

forecast <- f[1:14*24]

y <- as.data.frame(cbind(temp, forecast))
END;
```

In turn, data preparation in PAL was straightforward like in this example:

```
DROP table TEMP_REG_DATA_01;
CREATE column table
TEMP_REG_DATA_01("ID" INTEGER,
"VALUE" DOUBLE, "WDAY" INTEGER,
"H" INTEGER, "TEMPERATURE" DOUBLE,
"FORECAST" DOUBLE);

INSERT INTO TEMP_REG_DATA_01
(SELECT DISTINCT
(DAYOFYEAR("data")-1)*24+"h"
AS "ID", "value" AS "VALUE",
WEEKDAY("data") AS "WDAY",  "h" AS "H",
"temperature" AS "TEMPERATURE",
"forecast" AS "FORECAST"
FROM "eda2" WHERE "uid" = 1 ORDER BY "ID");
```

Much to our surprise, the setting up of PAL engine for linear regression did not require special efforts and can be done in few simple, standard steps which can be then reused when needed.

```
DROP TYPE PAL_T_RG_DATA;
CREATE TYPE PAL_T_RG_DATA AS TABLE
("ID" INTEGER, "VALUE" DOUBLE,
"WDAY" INTEGER, "H" INTEGER,
"TEMPERATURE" DOUBLE);
```

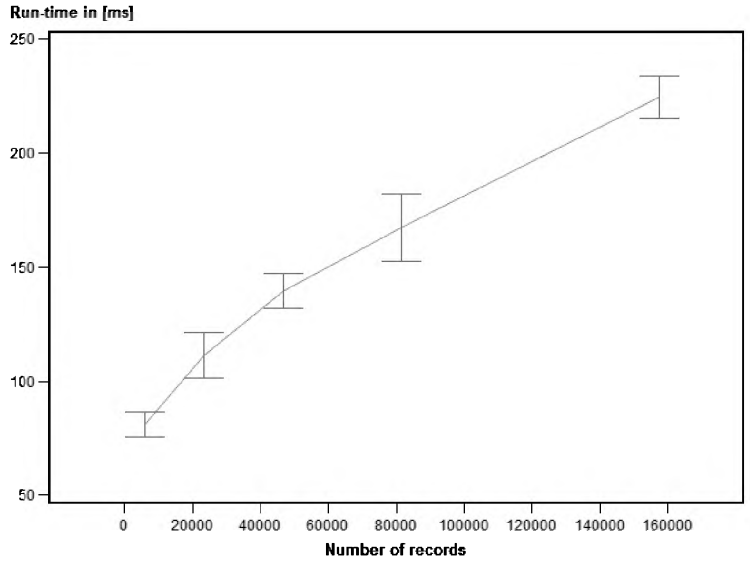


Figure 3: Scaling of PAL forecasting procedure

```
DROP TYPE PAL_T_RG_PARAMS;
CREATE TYPE PAL_T_RG_PARAMS AS TABLE
("NAME" VARCHAR(60), "INTARGS" INTEGER,
"DOUBLEARGS" DOUBLE,
"STRINGARGS" VARCHAR (100));
DROP TYPE PAL_T_RG_COEFF;
CREATE TYPE PAL_T_RG_COEFF AS TABLE
("ID" INTEGER, "AI" DOUBLE);
DROP TYPE PAL_T_RG_FITTED;
CREATE TYPE PAL_T_RG_FITTED AS TABLE
("ID" INTEGER, "FITTED" DOUBLE);
DROP TYPE PAL_T_RG_SIGNIFICANCE;
CREATE TYPE PAL_T_RG_SIGNIFICANCE AS TABLE
("NAME" VARCHAR(50), "VALUE" DOUBLE);
DROP TYPE PAL_T_RG_PMML;
CREATE TYPE PAL_T_RG_PMML AS TABLE
("ID" INTEGER, "PMML" VARCHAR(5000));

DROP TABLE PAL_RG_SIGNATURE;
CREATE COLUMN TABLE PAL_RG_SIGNATURE
("ID" INTEGER, "TYPENAME" VARCHAR(100),
"DIRECTION" VARCHAR(100));
INSERT INTO PAL_RG_SIGNATURE
VALUES (1, 'PAL_T_RG_DATA', 'in');
INSERT INTO PAL_RG_SIGNATURE
VALUES (2, 'PAL_T_RG_PARAMS', 'in');
INSERT INTO PAL_RG_SIGNATURE
VALUES (3, 'PAL_T_RG_COEFF', 'out');
INSERT INTO PAL_RG_SIGNATURE
VALUES (4, 'PAL_T_RG_FITTED', 'out');
INSERT INTO PAL_RG_SIGNATURE
VALUES (5, 'PAL_T_RG_SIGNIFICANCE', 'out');
INSERT INTO PAL_RG_SIGNATURE
VALUES (6, 'PAL_T_RG_PMML', 'out');

CALL SYSTEM.AFL_WRAPPER_GENERATOR
('PAL_RG3', 'AFLPAL', 'LRREGRESSION',
PAL_RG_SIGNATURE);
```

Summing up, using PAL turns out to be efficient and easy alternative to R-serve.

4 Findings

We were pretty much satisfied with the capabilities of PAL. It proved to be a user-friendly toolbox with very efficient procedures. The average time of training the regression model (on the 180 days time series data) is 84 ms, which is a superb result in terms of performance.

As hoped, the SAP HANA Predictive Analysis Library (PAL) offers better performance than R (which was discussed in the previous report). Especially promising is the scalability of the PAL code as shown in Fig. 3.

The average time for comparing demand profiles (180 days long, hourly granularity) is 34 ms. This means that if typically we would like to compare one profile of a prospective customer with 1000 profiles from the database - which is usually far too many as only business tariffs need to be tailored to the specific customers - this would be completed in about 3.5 s. It is a time perfectly acceptable by the end user and something which was out of reach till now. It proves that SAP HANA opens up new avenues for business cases. This leads to the following proposed new business cases for analytical users and account managers:

1. An analyst can try hundreds of different forecasting models, with different sets of explanatory variables in a short time, and then choose the best fitting model. Moreover, this can be done for each time series separately. The performance achieved in our tests shows this can be done to generate daily forecasts.

2. A new prospective customer comes requesting contract terms and conditions. An analyst can quickly find the most similar customer to him and

then customize the tariff based on what is already known for the existing similar customer. This can lead to substantial savings and better tariff optimization.

3. Demand forecasts can be generated for individual customers which will play an important role when smart grids based on renewable sources become more popular.

5 Conclusions and Next Steps

The goal of the presented project was to design and implement quasi real-time load forecasting for a huge number of individual customers profiles using in memory computing. We proved a hypothesis: forecasting accuracy can be substantially increased by individual forecasting on the level of each customer done in quasi real-time using SAP HANA parallelism, and a new insight into the analysed data can be gained.
Within the project we evaluated efficiency and performance of various computational strategies. Fast computations allowed for comparisons of different forecasting models and choosing the best one on a spot. Also, the possibility for adding different exogenous variables to the models and being able to quickly analyse their significance opened up new applications of energy load forecasting.
An additional intermediate result achieved is the possibility to cluster customers for the purpose of pricing and marketing campaign management. Thanks to SAP HANA huge computational powers it became possible to compare models in quasi-real time, select the most significant variables and provide new ways of analysis. The achieved results include a significant speed-up of the forecasting process and accuracy improvement.
With the analytical and computational experiments conducted we are now ready to put all those blocks together and utilize the reporting capabilities of SAP HANA to build a prototype of a system which business users could use to support their daily activities. So the next steps would be: Building a prototype of a system supporting business users in energy demand forecasting.

References

[1] Y. Aragon. *Séries temporelles avec R. Méthodes et cas*. Springer, Collection Pratique R, 1st edition, 2011.

[2] W. Charytoniuk, M.-S. S. Chen, P. Kotas, and P. Van Olinda. Demand forecasting in power distribution systems using nonparametric probability density estimation. *Power Systems, IEEE Transactions*, 14/4:1200–1206, 1999.

[3] F. McLoughlin, A. Duffy, and M. Conlon. Evaluation of time series techniques to characterise domestic electricity demand. *Energy*, 50(0):120 – 130, 2013.

[4] H. Plattner and A. Zeier. *In-Memory Data Management: An Inflection Point for Enterprise Applications*. Springer, Berlin Heidelberg, 2011.

[5] SAP. SAP HANA Predictive Analysis Library (PAL) Reference. 2012.

[6] SAP. SAP HANA R Integration Guide. 2013.

[7] R. Weron. *Modeling and Forecasting Electricity Loads and Prices: A Statistical Approach*. John Wiley and Sons Ltd., 2006.

Normalisation of Log Messages

Report for Project "Towards an Integrated Platform for Simulating, Monitoring, and Analytics of SAP Software" in 2013 Spring at HPI FutureSoC Lab

Andrey Sapegin, David Jaeger, Amir Azodi, Feng Cheng, Christoph Meinel
Hasso-Plattner-Institut (HPI), University of Potsdam
P.O.Box 900460, 14440 Potsdam, Germany
{andrey.sapegin, david.jaeger, amir.azodi, feng.cheng, christoph.meinel}@hpi.uni-potsdam.de

***Abstract*—The differences in log file formats employed in a variety of services and applications remain to be a problem for security analysts and developers of intrusion detection systems. The proposed solution, i.e., the usage of common log formats, has a limited utilization within existing solutions for security management. In our paper, we reveal the reasons for this limitation. We show disadvantages of existing common log formats for normalisation of security events. To deal with it we have created a new log format that fits for intrusion detection purposes and can be extended easily. Taking previous work into account, we would like to propose a new format as an extension to existing common log formats, rather than a standalone specification.**

***Keywords*—log normalisation, intrusion detection, common log format**

I Introduction

Nowadays, every application has its own log file format. Such a variety of formats complicates log analysis [1] and causes problems for system administrators, developers of intrusion detection systems (IDS) [2] and security analysts [3].

Therefore, several solutions [4] were introduced with the aim to solve this problem: CEE [3], CEF [5], IODEF [6].

However, there are many developers who avoid using common log formats in their projects. This statement particularly applies to intrusion detection systems. Many software vendors and open source projects still use their own log format for IDSs [7, 8, 9]. The reasons for this situation hide in the IDS architecture. To operate rapidly, such systems need to analyse huge amount of logs from hundreds of devices and services on the fly. T herefore, t he l og f ormat s hould b e lightweight and able to encapsulate all network and host logs from large datasets.

Log file standards that are aiming to handle all possible log output of any service, do not fit for these purposes. Moreover, in some cases, the variety of standardised fields, o ffered by such a common log file f ormat, d o n ot p rovide a suitable schema for normalisation and correlation of security events. During the development of our own IDS —Security Analytics Lab [10] — we have faced the same challenges and came to the same solution: a log format that is specially designed for IDSs. However, the object log format we present in this paper noticeably differs from the existing solutions.

Our goal was to experiment with a flexible lightweight format to optimise attack detection. First, we analysed a variety of large log files and came up with a list of common fields that can be found in the logged events. Then, this list was reduced to the fields that we considered most important for the security analysis of occurred events.

As a result, the developed log format facilitates the detection of attacks and simplifies their correlation. We demonstrate the ability of our format to simplify attack detection within a case study. Besides this particular use case, the proposed log format retains to be adjustable and could be utilised for generic normalisation concepts.

The remainder of the paper has the following structure: Section II provides overview of the Future SOC Lab project, Section III describes the data set used, Section IV evaluates and compares existing log formats, Section V provides details on the log format we developed, Section VI tells about results of a case study. We discuss our results and conclude in the Section VII.

II HPI Future SOC Lab project

The research result described in this report is part of our HPI Future SOC Lab project "Towards an Integrated Platform for Simulating, Monitoring, and Analytics of SAP Software". Within this project we expect to design and implement a multi-purpose platform with complex software, e.g., SAP software. This platform will be used as a base for vulnerability analysis research in simulated enterprise environment. To carry it out, we are testing the known and modern attack approaches and also attract external attackers using self-organised honeypot system. Each component of the platform should be connected to the centralised monitoring and security management system, which collects data, that could be used for different research directions also after the project ends. The project was planned in three phases:

- design, development and deployment of SAL-monitored SAP platform
- research, design and implementation of Attacks and Honeypots of SAP software
- creation of use and attack scenarios, scientific analysis of results and data collected

The first phase concentrates on the preparation of the platform, research and selection of tools and technologies. We select typical distributions of SAP software, install, configure them and prepare the connection to the Security Information and Event Management System (SIEM). At the same step we prepare other components of the system and develop the platform architecture. The second step is focused on the

vulnerability analysis of software deployed on the platform. We select tools for penetration testing, analyse known and discover unknown vulnerabilities. Meanwhile we explore existing attacking approaches and research for new attacks. For the last phase we propose more detailed analysis of collected data, improvement of the platform and security management system, as well as the documentation, white papers with technical details and vulnerability reports. In the end we plan to deliver the stable prototype of the platform, including honeypot system, monitoring and intrusion detection systems, as well as actual SAP software, for which we will report all found previously unknown vulnerabilities. All tested attack tools and approaches will be documented and presented in the repository together with the documentation of project concepts and technical reports.

We have carried out the first phase of the project in the spring and summer 2013. During this phase we have designed a test bed for vulnerability analysis with SAP software and HPI Security Analytics Lab (HPI-SAL) — our own developed SIEM, based on SAP HANA in-memory database [10]. The use of SIEM implies the ability to collect, process and correlate log messages from all monitored software. Variety of different log formats and message types oblige the intrusion detection system to be able to normalise all the events being produced into the same format. The normalisation process is one of the most challenging tasks for the modern intrusion detection systems. HPI Security Analytics Lab should not only convert the messages in the same format, but also recognise (and put in the different fields) the meaningful parts of log messages, extracting the semantics of the event details for posterior correlation. To deal with this problem we have developed our own log format for the normalisation of any types of log messages. The proposed format has a number of advantages — we describe them in this report — if compared with existing solutions.

Another challenging task we have faced during the first phase is a vulnerability modelling. To design different platform components, such as honeypot or develop the penetration testing strategy, it is extremely important to gather information about vulnerable software, collect details and classify newly discovered vulnerabilities. We have solved this problem and used the separate database that acts as an integrated storage for all existing vulnerability information: the HPI Vulnerability Database (HPI-VDB) [11]. HPI-VDB is a repository containing structured information about almost 60000 vulnerabilities, including the description of pre- and post-conditions for every vulnerability. We use these data to conduct research on information quality, new services and classification of vulnerabilities. During the first phase we were working on the further development of HPI-VDB and connection to HPI Security Analytics Lab. Our goal is to embed the HPI-VDB into the platform and integrate it into the framework for the vulnerability analysis of SAP Software. During the next phase we will use it as information source while discovering, evaluating and classifying vulnerabilities of selected software.

The prototype of the platform described above will allow us to perform vulnerability analysis and penetration testing of SAP software, as well as discover new attack vectors through the honeypot system, which are the next steps in the project.

III Data

To prove our concepts, we utilised data from "Scan of the Month" Honeynet [12] Challenge: "Scan 34 - Analyze real honeynet logs for attacks and activity" [13]. The Honeynet Project aims to help security specialists to sharpen their forensics and attack detection skills. To achieve it, the project shares real attack data collected from honeynets from all over the world. The "Scan of the Month" challenge provides archives with log files from different hardware and software systems for the postmortem analysis. The "Scan 34" challenge contains log files being collected between 30 January and 31 March 2005. We chose this particular challenge because it implies analysis of multiple log files for different services and includes various heterogeneous attack traces. We describe the log files in Table I.

The "Scan 34" challenge also provides the description on the attacks and important events in the log files. See the details of the 15 most important events in Table II.

In total, four attacks led to the server's compromisation. Two of four attackers used the vulnerability of 'awstats.pl' installed after the server reboot, probably by a system administrator. Another two attackers successfully brute-forced the SSH password. The first attacker has also installed an IRC bot right after the intrusion. Other events listed in the table were not harmful for the server; they represent suspicious activity that did not result into a successful intrusion.

IV Existing log formats for normalisation of security events

Before we decided to use our own log format, we have evaluated existing solutions, trying to estimate if they could fit for our purposes of normalisation of security events. Please see the overview of proposed log formats in Table III.

The selected formats have different structure, size and even usage purpose. CEE is a generic format for logging any types of events, IDMEF was developed for exchange of security messages. IODEF is specially designed for computer security incidents, as well as CEF developed as a part of intrusion detection system. Although the described formats have different purposes, each of them could be used for log normalisation.

However, not every part of the log message could be normalised to the standard fields defined in every format. To deal with it, each format allows to use extra fields or extend its structure. Therefore we decided to check how many changes we need to fully normalise log messages we have into each of 4 formats. The Table IV shows, that some parts of log messages have no corresponding field in the log format. This small observation reveals a deeper problem, if investigated. The reviewed log formats offer limited number of options to parse the textual event description precisely. For example, using the IDMEF, messages like "authentication failure" and "Relaying denied. IP name lookup failed" are supposed to be written into the Classification class as whole. Such single field available in the log format does not allow to effectively normalise descriptions like "BLEEDING-EDGE WORM Mydoom.ah/i Infection IRC Activity [Classification: A Network Trojan was detected]". CEE format demonstrates a similar issue, as all

TABLE I: LOG FILES AVAILABLE TO SOLVE THE CHALLENGE

Server	Service	Date	Total number of lines
n/a	HTTP Server	Jan 30 - Mar 16	3925
bridge	iptables firewall	Feb 25 - Mar 31	179752
bastion	snort IDS	Feb 25 - Mar 31	69039
combo	syslog	Jan 30 - Mar 17	7620

TABLE II: SECURITY RELATED EVENTS THAT OCCURRED DURING THE MONITORING FOR THE CHALLENGE

N	Event	Date	Details
1	Reboot	Feb 11, 2005	n/a
2	Software installation	Feb 25, 2005	AWSTATS installed
3	Server compromised	Feb 26, 2005	Code injection through awstats.pl
4	Server compromised	Mar 04, 2005	Code injection through awstats.pl
5	Server compromised	Mar 06, 2005	ssh brute-force successful
6	Server compromised	Mar 13, 2005	ssh brute-force successful
7	Software installation	Feb 26, 2005	IRC bot installed by an attacker
8	ICMP alert	n/a	ICMP Destination Unreachable
9	Slammer worm	n/a	Worm propagation attempt
10	IIS attacks	n/a	WebDAV search access, cmd.exe access, etc.
11	SMTP scan	n/a	POLICY SMTP relaying denied
12	Typot trojan	n/a	trojan traffic
13	RPC scan	n/a	RPC portmap status request
14	Port scan	n/a	NMAP -sA (ACK scan)
15	Slapper worm	n/a	Worm propagation attempt

descriptions and other details are supposed to be stored in the "message" field. Since the information still could be stored in such generic fields, we do not mention it in the table as parts without corresponding field.

Another common issue affecting all log formats implies complications while normalising the event details like user name, rule/action applied, method (e.g., GET) and response (404 or 550). In other words, the existing formats offer a limited number of fields to specify, what has happened. CEF deals with it better than other formats, due to the presence of keys like 'act' (Action mentioned in the event), 'app' (Application protocol), 'request', 'requestMethod' and others.

Finally, all formats have too few – or do not have any – fields for expression of properties specific for intrusion detection, like relation between events, security metrics or classifiers such as CVE and CWE identifiers, object and subject of the event. These properties usually could not be extracted from the log lines itself, but are expected to be filled by IDS.

Thus, not only missing fields from the table should be added to each of formats, but also the fields to improve parsing of message parts and make possible to store intrusion detection properties. That's why we decided to select and re-engineer one of the formats. However, the structure, number of fields/attributes and possible values or contents are different for all formats, which make them hard to compare with each other. Therefore, we chose the following additional criteria to examine log formats:

- *scalability*. There should be an ability to add custom fields, if some log formats could not be normalised into standard ones.
- *light weight*. The log format should have reasonable number of fields/attributes to avoid redundancy and fit for high-speed normalisation purposes.
- *multilevel schema*. Intrusion detection implies correlation of the normalised logs. The hierarchical or other connected structure is preferable as it will allow to categorise different fields into classes.

Applying the first criterion, all selected formats are scalable to some extent. IDMEF [14] and IODEF contain an *Additional Data* class, CEF supports *Custom Dictionary Extensions* and CEE allows to create custom events and fields.

Analysing the second criterion, i.e., the number of fields/attributes, the CEE format is much more compact than IODEF, IDMEF and CEF. We also would like to notice, that IDMEF is designed for data exchange for intrusion detection and security management systems only. So we could expect a lot of additional attributes being added to IDMEF to adopt it for normalisation purposes.

Finally the last criterion is examined. The CEF has a flat structure; IODEF and IDMEF have similar multi-level schemas with connections between different classes; and CEE has a clear object-based hierarchical structure.

Based on the criteria defined, we selected CEE to be extended as the log format that fits better for our purposes.

V OBJECT LOG FORMAT

As mentioned in the previous Section, the decision to develop our own log format for IDS came with the attempt to utilise the CEE format in our experimental intrusion detection system [10].

We started with modifying the CEE format and adding custom fields to cover all possible log message variants from

TABLE III: EXISTING STANDARDS FOR LOG FORMATS

Format name	Organisation	Size estimation	Format structure
CEE (Common Event Expression)	MITRE	58 fields, 7 objects	two levels, hierarchical, object-based
IDMEF (Intrusion Detection Message Exchange Format)	IETF	118 elements, 5 core classes, 53 attributes	multi-level, class-based
IODEF (Incident Object Description Exchange Format)	IETF	53 elements, 19 top-level classes, 83 attributes	multi-level, class-based
CEF (ArcSight Common Event Format)	HP	104 keys	one level, key-value pairs

TABLE IV: PARTS OF LOG MESSAGES WITHOUT CORRESPONDING FIELD IN THE LOG FORMAT.

Log line	CEE	IDMEF	IODEF	CEF
81.181.146.13 - - [15/Mar/2005:05:06:53 -0500] "GET //cgi-bin/awstats/awstats.pl? configdir= —%20id%20— HTTP/1.1" 404 1050 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)	HTTP/1.0, GET, 404	Mozilla/4.0, 404	81.181.146.13, GET, //cgi-bin/awstats/ awstats.pl, 404, Mozilla/4.0	-
Mar 15 13:38:03 combo sshd(pam_unix)[14490]: authentication failure; logname= uid=0 euid=0 tty=NODEVssh ruser= rhost=202.68.93.5.dts.net.nz user=root	-	-	14490, 202.68.93.5 .dts.net.nz, user=root	14490
Mar 1 20:45:12 bastion snort: [1:2001439:3] BLEEDING-EDGE WORM Mydoom.ah/i Infection IRC Activity [Classification: A Network Trojan was detected] [Priority: 1]: TCP 11.11.79.67:2568 -> 129.27.9.248:6667	TCP	-	Priority: 1, 11.11.79.67, 129.27.9.248	Priority: 1
Mar 24 19:46:50 bridge kernel: INBOUND ICMP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=63.197.49.61 DST=11.11.79.100 LEN=32 TOS=0x00 PREC=0x00 TTL=111 ID=1053 PROTO=ICMP TYPE=8 CODE=0 ID=512 SEQ=29421	ICMP, eth0, eth1, br0	-	SRC = 63.197.49.61, DST = 11.11.79.100, eth0, eth1, br0	br0
Feb 1 10:08:32 combo sendmail[32433]: j11F8FP0032433: ruleset=check_rcpt, arg1 = <china9988@21cn.com>, relay=[61.73.94.162], reject=550 5.7.1 <china9988@21cn.com>... Relaying denied. IP name lookup failed [61.73.94.162]	ruleset = check_rcpt, 550, china9988@ 21cn.com	ruleset = check_rcpt, 550, china9988@ 21cn.com	check_rcpt, relay= [61.73.94.162], 550, 61.73.94.162	32433, ruleset = check_rcpt

the Honeynet challenge. However, this draft log format still had several problems. First, we have used less than half of fields (24 of 58) defined in the standard. Compared to the number of custom fields added – 59 – and the fact that standard CEE fields do not always present the key properties of the log message, it becomes hard to argue for using the resulting format inside the IDS system. Second, the object-field hierarchy defined in the Field Dictionary contains only one abstraction level. Taking into account overlapping semantics (e.g., *appname* and *app.name* fields) in the CEE notation, this relatively flat structure contains many similar fields and adds a lot of confusion for a developer. To handle with our goals, we made changes to the CEE structure as well. We extended the hierarchical structure of CEE to three levels to achieve flexibility and improve clarity.

We present the proposed format in Figure 1. The first level (marked with bold) describes global parameters or classes of parameters, such as *network* or *original_event*. On the second level of our hierarchy (written in normal font) we describe the most significant properties. And on the third level (marked with italics) we place specific information such as network protocol fields.

Compared to the formats examined in the Section IV, the proposed format offers multiple fields to store event details describing what has happened (*tag* and *application classes*), as well as fields, which are highly relevant for intrusion detection (mainly, classes *related_ids*, *relation* and *security*).

Let's consider the example from the Table IV to show how to parse real log data into the proposed format:

```
81.181.146.13 - - [15/Mar/2005:05:06:53 -0500] ''GET //
    cgi-bin/awstats/awstats.pl? configdir= |\%20id\%20|
     HTTP/1.1'' 404 1050 ''-'' "Mozilla/4.0 (compatible
    ; MSIE 6.0; Windows 98
```

The log line listed above is taken from the 'access_log' file on the HTTP server. Using it, we now describe the most significant elements and how the information from our example log entry should be distributed over them.

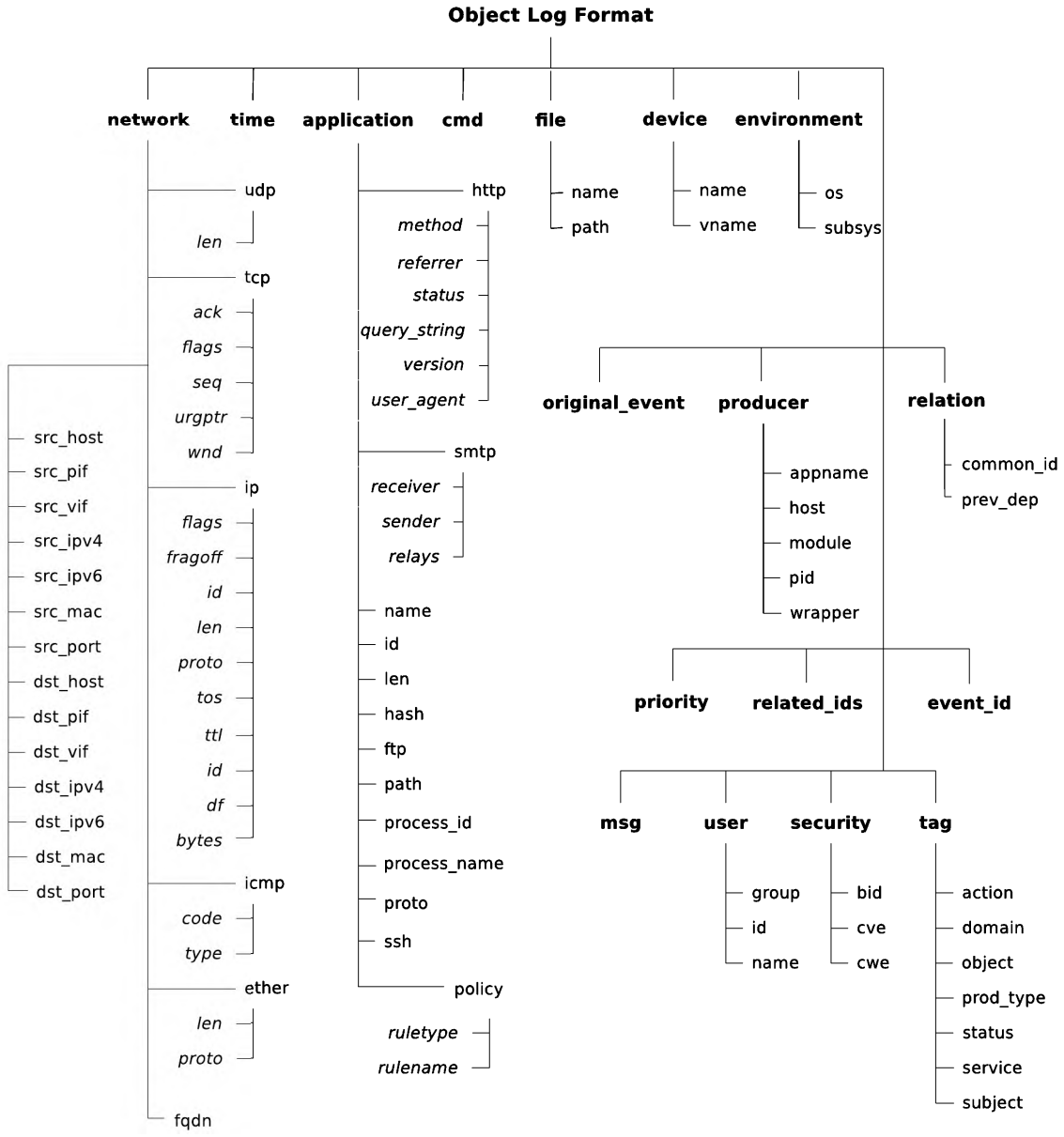


Fig. 1: Tree of properties in our log format

- **network** This class covers all properties around the lower network layers, i.e., the link, network and transport layer of the TCP/IP protocol stack. This basically covers most information about the source and destination endpoints of network events, including their MAC address, IP address and ports. Additionally, to the endpoints, information on protocol fields in captured network packets is organised in subclasses, such as *ether*, *ip*, *icmp*, *tcp* and *udp*. These details can then be used to analyse the exact workings of a network communication. It should be noted, that since we are parsing security events and not only the network packets, this class could provide details for multiple network packets associated with an event, even if they were reported in different logs.

  From the log line sample, we extract only the IP address—218.1.111.50—to fill the 'network.src_ipv4' field.

- **application** This class covers all properties around the applications and services involved in the event. The application can represent different kind of involved applications in an event. In a network connection, this application can be the client application, which initiates a connection or sends a packet to the server, or the service application, which is the target of a sent packet or initiated connection. To cover the various special characteristics in service communications, the *application* class also allows to further specify parameters in the context of HTTP, FTP and more. In a host-based event, this application is usually the application

that initiates an action, like an application that writes a file. The semantics of the specified application can be obtained from the *subject* and *object* field of the *tag* class.
Analysing the example log line, it is possible to notice, that the application is a user agent, so the 'application.http.user_agent' field should be filled with "Mozilla/4.0 (compatible; MSIE 4.01; Windows 95)" line and 'application.http.query_string' field – with the following line:
"GET /scripts/..%c1%9c../winnt/system32/cmd.exe?/c +dir HTTP/1.0".

- **producer** This class gives information on the application that observed and eventually persisted an event. It should describe the first application that persisted the events and should not be changed to one of the intermediate processing applications.
In our example, a producer is a HTTP server, namely "Apache HTTP Server", which should be written into the 'producer.appname'. If we would know some details about the host from the other log lines, we could also fill the 'producer.host' field, e.g., with *http_server*.

- **file** This class describes the files that were involved in an event. A file can appear in different contexts, i.e., mainly as a data source and target of access operations, such as read and write. In the case of an FTP or HTTP connection, this parameter could give information on the accessed resource. Similar to the *application* class, the concrete context for one event is defined by the *subject* and *object* field of the *tag* class.
Considering the sample log line, the 'file.name' field will contain 'cmd.exe' and 'file.path' – */scripts/..%c1%9c../winnt/system32/cmd.exe'*.

- **original_event** This field keeps the original log as found in the log source. This is usually a string of the log line or the fields of that where found log database. The whole sample log line should be filled into the 'original_event' in our case.

- **relation** This class serves the cases when several events are related to each other. In the case of multiple events sharing a common identifier for correlation, this identifier is stored in the *common_id* field. The *prev_dep* property indicates if the current event depends on a previous one.
For the example considered, these fields will be empty, because the log event is standalone and not related to any other events.

- **tag** This class provides abstract information on top of the message details, mainly to categorise and tag events. As this information is not always directly represented in the log data, this is required to be set by the user. In some cases the action (and other fields as well) cannot be explicitly identified with a single term. Therefore, we propose all fields within this class as multi-value.
E.g., the 'tag.action' could be {*get,access*}, 'tag.subject' – *host* or *user_client*, 'tag.object' – *file* or *web_document*, 'tag.prod_type' – *web_server* and 'tag.service' could be *web*.

- **security** This class provides links to identifier of vulnerability, related to the logged event. For example, 'security.cve' could be "CVE-2010-4369".

- **event_id** This field contains a unique internal id of the event. This should be unique among all generated events in a management system. For our example, we used an SQL database for automatic id generation.

The developed format structure is easy to present, extend and map into database relations. These features simplify the developer's tasks and clarify the semantics of fields with similar names. For example, now the former *appname* and *app.name* fields are easier distinguishable by using *producer.appname* and *application.name* as names. Finally, with 107 fields used, we were able to effectively normalise every log message from the dataset. After the normalisation step, most of the attacks could be discovered using simple search queries, as shown in the next section.

VI The Role of the Common Log Format in Attack Detection

As mentioned earlier, we normalise files before searching for attacks. This pre-processing step includes parsing of the logs into described log format with regular expressions and inserting them into the SQL database.

The proposed log format allows to detect attacks described in the Table II using simple queries, without the usage of a correlation engine or other advanced intrusion detection techniques. The challenge winners on the other part have used self-written scripts and manual analysis [15, 16]. We now provide several use cases to demonstrate the benefits of the proposed log format. E.g., to check for SMTP scans, we use the following query:

```
select * from event where application_protocol = 'smtp'
    and tag_status = 'failure'
```

All 220 log lines returned correspond to event 11 from Table II. These lines include both *sendmail* messages from the mail server ("combo") and *snort* alerts from the logging server ("bastion").

Next, to detect the code injection attempts, we suggest another simple SQL query:

```
SELECT * FROM event WHERE application_cmd LIKE '%_%\%3b%
    ' ESCAPE '\' AND application_protocol = 'http'
    ORDER BY time
```

We search for a semicolon in the URL being processed by any HTTP Server mentioned in the log files[1]. The result contains 20 logs, all of them related to the code injection cases through awstats.pl. 15 logs correspond to events 3 and 7 from Table II, two other lines to event 4. The remaining 3 lines were related to the awstats.pl code injection attempts on 2 and 12 March 2005, not being mentioned in the official challenge results[2].

[1] In almost all cases, the semicolon in the URL indicates a malicious event like code injection.

[2] these events were found by the challenge winner [15].

TABLE V: HTTP PATTERNS.

HTTP request header element	Log Format Object	Patterns
Host	Network.fqdn & File.path	"eval(", "concat", "union!+!select", "(null)", "base64_", "/localhost", "/pingserver", "/config.", "/wwwroot", "/makefile", "crossdomain.", "proc/self/environ", "etc/passwd", ".exe", ".sql", ".ini", "/.bash", "/.svn", "/.tar", " ", "¡", "¿", "/=", "...", "+++", "/&&"
Content-Location	Application. http. query_string	"?", ":", "[", "]", "../", "127.0.0.1", "loopback", "%0a", "%0d", "%22", "%27", "%3b", "%3c", "%3e", "%00", "%2e%2e", "%25", "union", "input_file", "execute", "mosconfig", "environ", "scanner", "path=.", "mod=."
User-Agent	Application. http. user_agent	"binlar", "casper", "cmswor", "diavol", "dotbot", "finder", "flicky", "jakarta", "libwww", "nutch", "planet", "purebot", "pycurl", "skygrid", "sucker", "turnit", "vikspi", "zmeu"

Obviously, many attacks could not be captured with such simple queries. But the log format structure allows a developer to easily create more sophisticated checks. Please see Table V for sample patterns of malicious HTTP events. The proposed object hierarchy of the log format allows specifying object-specific patterns (in this example – patterns for different elements of HTTP request header). This flexibility simplifies creation of rules for matching of malicious events. Now, if we check the normalised log messages with all the patterns selected, we could match 740 HTTP events related to line 10 from Table II, as well as 35 redirects and 2 cases of calls to the 'libwww' library.

VII DISCUSSION

We have shown how the common log format, if thoroughly developed with the regard for specific usage conditions (intrusion detection in our case), could facilitate a lot of operations, including search for attack patterns and correlation of events from different servers. However, the structure of existing log format standards could differ from the one imposed by specific use conditions. On the one hand, common log formats being developed nowadays (CEE [3] and IODEF [6]) try to handle all possible use cases. Such unified approach often needs a manual adoption to be a best fit for a specific use case. On the the other hand, log formats, specially established for intrusion detection sometimes have a limited scope. IDMEF [14] defines the format for inter-communication only (between intrusion detection, response and management systems). And CEF [5] has a flat hierarchy, which makes it less flexible in comparison with object-based log formats like CEE [3].

Within our proposed format we try to utilise the strong sides of both approaches and design the flexible object log format that fits the specific purposes, but could be easily extended for generic use cases. However, we do not intend to create a replacement for standards proposed by MITRE [17], IETF [18] and others. Rather, we hope that existed standards could be more flexible to be used for specific purposes and be able to combine extensible structure, light weight and multifarious capabilities, such as search and correlation facilities for intrusion detection systems.

REFERENCES

[1] D. Casey, "Turning log files into a security asset," *Network Security*, vol. 2008, no. 2, pp. 4–7, 2008.

[2] L. Yang, P. Manadhata, W. Horne, P. Rao, and V. Ganapathy, "Fast submatch extraction using obdds," in *Proceedings of the eighth ACM/IEEE symposium on Architectures for networking and communications systems*, ser. ANCS '12. New York, NY, USA: ACM, 2012, pp. 163–174. [Online]. Available: http://doi.acm.org/10. 1145/2396556.2396594.

[3] "Common Event Expression White Paper," June 2008.

[4] G. T. McGuire, "The state of security automation standards - 2011," http://www.mitre.org/work/tech_papers/2011/11_3822/, November 2011.

[5] ArcSight, "Common Event Format," http://mita-tac.wikispaces.com/file/view/CEF+White+Paper+071709.pdf, July 2009.

[6] R. Danyliw, J. Meijer, and Y. Demchenko, "The incident object description exchange format," IETF RFC5070, December 2007.

[7] "BlackStratus LOG Storm," http://www.blackstratus.com/enterprise/supported-technologies/log-storm/.

[8] "Sawmill log analysis tool," http://www.sawmill.net/.

[9] "Open Source Host-based Intrusion Detection System," http://www.ossec.net/.

[10] S. Roschke, F. Cheng, and C. Meinel, "An extensible and virtualization-compatible ids management architecture," in *Proceedings of the 2009 Fifth International Conference on Information Assurance and Security - Volume 02*, ser. IAS '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 130–134. [Online]. Available: http://dx.doi.org/10.1109/IAS.2009.151.

[11] "HPI Vulnerability Database," https://www.hpi-vdb.de/.

[12] "The Honeynet Project," http://honeynet.org/.

[13] A. Chuvakin, "Scan 34 - analyze real honeynet logs for attacks and activity," http://old.honeynet.org/scans/scan34/, February 2005.

[14] H. Debar, D. Curry, and B. Feinstein, "The Intrusion Detection Message Exchange Format (IDMEF)," IETF RFC4765, March 2007.

[15] M. Richard, M. Ligh, A. Magnusson, S. Seale, and K. Standridge, "Project honeynet scan of the month 34," http://old.honeynet.org/scans/scan34/sols/1/index.html, May 2005.

[16] C. Kronberg and A. Freeworld, "Analysis of the logfiles given in SotM34," http://old.honeynet.org/scans/scan34/sols/2/proc.pdf, 2005.

[17] "The MITRE Corporation," http://www.mitre.org/.

[18] "Internet Engineering Task Force," http://www.ietf.org/.

Counting Polyominoes in the Limit

Gill Barequet
The Technion—Israel Institute of Technology
Department of Computer Science
Haifa 32000, Israel
barequet@cs.technion.ac.il

Günter Rote
Freie Universität Berlin
Institut für Informatik
Takustraße 9, 14195 Berlin, Germany
rote@inf.fu-berlin.de

Abstract

We improved the lower bound on the growth constant λ of polyominoes to 4.00253.

1 Problem Statement

A polyomino (or "animal") is an edge-connected set of squares on the two-dimensional square lattice, see Figure 1. The research of polyominoes is motivated by problems arising in statistical physics, e.g., computing the mean cluster density in percolation processes, in particular those of fluid flow in random media, and modeling the collapse of branched polymers at high temperature. There is no closed formula for the number A_n of polyominoes of size n. The numbers A_n have been computed up to $n = 56$ (also with the help of supercomputers [3]). We investigate one of the fundamental problems related to polyominoes, namely, computing their asymptotic growth rate $\lambda = \lim_{n\to\infty} A_{n+1}/A_n$. This limit λ is known as Klarners constant. Determining the exact value of λ (or even good bounds) is an extremely hard problem in enumerative combinatorics.

Figure 1: The 12 pentominoes (n=5), figuring as part of a game

2 Project Idea

For attacking this problem, we count polyominoes on a *twisted cylinder*, based on our earlier work [1], see Figure 2. This is a wrap-around spiral-like structure which can be built by adding one square at a time in a uniform way; therefore the incremental buildup of polyominoes can be modeled very conveniently. This is visualized in a recent video by Gill Barequet and Mira Shalah [2].

The number of polyominoes on the twisted cylinder is a lower bound on the number of polyominoes in the plane, in which we are actually interested. The larger the width W of the cylinder, the better the bound, but the more computation is needed. In 2004, we had developed a serial computer program which required extremely high computing resources by the standards

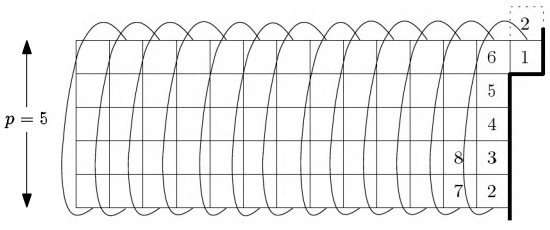


Figure 2: A twisted cylinder of perimeter (width) $p = W = 5$

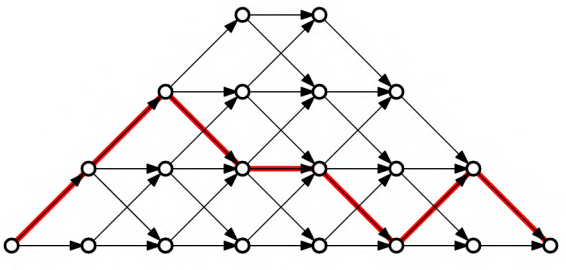


Figure 3: A Motzkin path

of that time, both in terms of running time and main memory, going up to width $W = 22$ with 32 GiBytes of main memory. Before the start of this project, the bounds on Klarner's constant were 3.985 from below and 4.6 from above. The bound 3.985 comes from the twisted cylinder of width $W = 23$. We set out to push the lower bound above 4.

3 Used Future SOC Lab resources

We used a Hewlett Packard DL980 G7 computer. It has 8 processors nodes with a total of 128 cores, and 2048 GiB RAM in total. Each node is an Intel Xeon X7560 Processor (8C, 2.26 GHz) with 24 MB cache, Intel64 architecture. Hyperthreading is used to allow 16 processes to run on 8 "physical cores", sharing certain resources.

4 Findings

The basic program is an iteration which computes a series of vectors y.
Each iteration constructs a new iterate vector y^{new} from the previous iterate y^{old} in a simple loop:

for $s := 1, \ldots, M$:
$(*)$ $\quad y^{\text{new}}[s] := y^{\text{new}}[\mathit{succ0}[s]] + y^{\text{old}}[\mathit{succ1}[s]]$;

The pointer $\mathit{succ0}[s]$ may be null, and in this case the corresponding entry is not used.
The vectors are indexed from 1 to M. Each entry represents a *state*, which corresponds to some combinatorial information about a partially built polyomino. Successor states correspond to adding an occupied (*succ1*) or a free (*succ0*) cell. States are encoded by so-called Motzkin paths, paths in the directed acyclic network shown in Figure 3, and these paths can be bijectively mapped to numbers between 1 and M. (The number M of Motzkin paths is called a Motzkin number.)
In the iteration $(*)$, the vector y^{new} depends on itself, but this is not a problem because $\mathit{succ0}[s]$, if it is non-null, is always less than s. In fact, the states can be partitioned into groups $G_0, G_1, \ldots, G_W$ such that $\mathit{succ0}[s]$ of an entry $s \in G_i$, if it is non-null, belongs to G_{i+1}. Thus, all entries s in a group G_i can be treated in parallel, in a straightforward manner. The groups are processed sequentially in the order $G_W, G_{W-1}, \ldots, G_1, G_0$.
The pointer arrays $\mathit{succ0}[]$ and $\mathit{succ1}[]$ are computed beforehand.
We first parallelized the existing program for using 128 parallel processor cores. The programs are written in C, and we used the OpenMP compiler directives for parallelization. We handled the distribution of the loops over the processors explicitly.
By May 1, 2013, we succeeded to run the iteration for $W = 26$ in 3 hours using about 1 TeraByte of main memory, establishing a new record lower bound of 3.9989. The number of states for this case is $M = 73.007{,}772.801$.
Getting to $W = 27$ and thus pushing the lower bound on λ above 4.0 was only possible by engineering tricks that saved memory at the expense of run-time, since otherwise the memory requirement would increase by a factor of 3 with each increase of W, and we have "only" 2 TeraBytes available.
We used the following measures.

1. Eliminating a certain fraction of "unreachable states" (about 11%). These states don't contribute to the limit. Elimination of these states required a change in the mapping between "reachable" Motzkin paths and integers.

2. "Bit-streaming" of the *succ*-arrays instead of storing them in fixed 4-byte or 8-byte words. (These arrays are accessed only sequentially.)

3. Storing only one iterate vector instead of two for groups $G_2, G_3, \ldots$.

4. Complete elimination of the *succ0*-arrays; instead, these pointers are recomputed each time they are needed. This required a streamlined program to accelerate the successor computation.

5. Streamlining the conversion between Motzkin paths and integers, using precomputed and stored tables. We took care that each processor has access to its own copy of the table, and the table is small enough to fit in the level-1 cache.

6. Writing checkpoints of intermediate results to disk.

On May 20, we succeeded to run the iteration for $W = 27$, using about 36 hours of runtime in total, distributed over several runs and writing intermediate results to disk. The lower bound on λ broke through the mythical barrier of 4.0 after 120 iterations. After 290 iterations, the lower bound converged to 4.00254, establishing the current record.
With hindsight, it would not have been necessary to write and read checkpoints to disk; on the other hand, it was good to have the data on disk for running an independent check.

Certification of the results. The proof of this bound consists essentially of an array of floating-point numbers, which is about 450 GBytes long. We wrote two independent programs for checking this "proof" based on programs for the successor computation written by different people. These programs worked purely sequentially and took about 20 hours each. They carry out one iteration $(*)$ and compute an interval for λ.
The result is calculated from the data by at most 26 additions of positive numbers plus 1 division, all in single-precision `float`. No denormalized numbers occur. So the relative error caused by the floating-point operations can be estimated, and we obtain a certified lower bound of 4.00253176 on λ.
In addition, we checked whether the conversions routines between Motzkin paths and integers work really as bijections, and whether the two versions of successor computation yield the same result. These checking programs do not need much memory and take about 10–20 hours each (sequentially). We also ran them on our own local workstations, which were slightly faster.

Some benchmarks. The 8 processor nodes are organized in hierarchical groups, leading to non-uniform memory access (NUMA): The following table lists the "distances" for accessing memory on one node from another node (or from the node itself)

node	0	1	2	3	4	5	6	7
0:	10	14	23	23	27	27	27	27
1:	14	10	23	23	27	27	27	27
2:	23	23	10	14	27	27	27	27
3:	23	23	14	10	27	27	27	27
4:	27	27	27	27	10	14	23	23
5:	27	27	27	27	14	10	23	23
6:	27	27	27	27	23	23	10	14
7:	27	27	27	27	23	23	14	10

Each loop of the program was partitioned into 128 equal-size chunks that were assigned to individual processors. We made preliminary experiments with various options for assigning the chunks to processors on the 8 nodes: (a) Round-robin (b) 16 consecutive chunks are assigned to one node. An iteration $(*)$ involves a write access to the entry s in $y^{\text{new}}[s]$ on the local node, and up to two read accesses to entries on (possibly) different nodes. Without a thorough investigation of the access pattern it is hard to predict which method would give advantages. It turned out that method (b) seemed to be slightly faster; this was the method that was used for the final runs.
After the result was obtained, we made some test runs with reduced numbers of processors, to see how the program scales and the check whether parallelism is really used. The running time for one iteration ($W = 27$) is shown in the following table:

number of processors	128	64	32	16
one iteration (minutes)	6	15	27	57
computation of *succ0* (min.)	11	13	29	54

This shows a close-to-linear speedup for the iterations. This seems to indicate a good balance between local memory accesses and arithmetic on the one hand and and global accesses (reading the entries of y) on the other hand. The initial computation of *succ0*-successors, which is done only once, has a less impressive speedup. The time for initialization of the vector y took always about 6 minutes, independent of the number of processors. This involves the effective page allocation (assignment of address ranges to processor nodes and to physical memory).

5 Next steps

With some more effort, it would be feasible to run $W = 28$. This would require (a) to eliminate also the storage for the *succ1*-successors and compute them from scratch, (b) to eliminate all groups $G_1, \ldots, G_W$ and keep only group G_0, (c) a customized floating-point storage format for the numbers y. (With a total of 2 GiBytes, we can only afford 27 bits per entry.) This would probably increase the bound to about 4.0065.

References

[1] G. Barequet, M. Moffie, A. Ribó, and G. Rote. Counting polyominoes on twisted cylinders. *INTEGERS: The Electronic Journal of Combinatorial Number Theory*, 6 (#A22):37 pages, Sept. 2006.

[2] G. Barequet and M. Shalah. Polyominoes on twisted cylinders (video). In *Proceedings of the 29th Annual Symposium on Computational Geometry, Rio de Janeiro (SoCG '13)*, pages 339–340. Association for Computing Machinery, 2013. `http://www.computational-geometry.org/SoCG-videos/socg13video/`.

[3] I. Jensen. Counting polyominoes: A parallel implementation for cluster computing,. In *Proc. Int. Conf. on Computational Science, part III, Melbourne, Australia and St. Petersburg, Russia,*, volume 2659 of *Lecture Notes in Computer Science*, pages 203–212. Springer, 2003.

Evaluation of Visual Concept Classifiers

Peter Retzlaff
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
peter.retzlaff@student.hpi.uni-potsdam.de

Christian Hentschel
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
christian.hentschel@hpi.uni-potsdam.de

Harald Sack
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
harald.sack@hpi.uni-potsdam.de

Abstract

Visual Concept Detection aims to classify images based on their content. Bag-of-(Visual-)Words features that quantize and count local gradient distributions in images similar to counting words in texts have proven to be meaningful image representations. In combination with supervised machine learning approaches such as AdaBoost, models for nearly every visual content can be learned by sufficient labeled training data. Similar to text classification the classification of an image depends on a weighted combination of the visual words from the vocabulary and the classifier is expected to learn these weights in order to obtain best results. Hence, the learned weights or importances of each visual word are a strong indicator for the robustness and reasonableness of the overall concept model. This work visualizes learned concept models by colorizing the image pixels using the importance value of the corresponding local visual word using a heatmap-like representation. Thereby we explicitly show sources of misclassification and thus help to understand and improve varying results for different concept classes. Due to the high-dimensional nature of the employed features training the concept models is an expensive process when regarding both – memory and computational resources. Multi-core architectures can help to significantly reduce the time required to train a model for a given visual concept and availability of large main memory guarantees that training can be preformed without latency due to disk I/O.

1 Introduction

Given a set of images and a set of concepts, the task of visual concept classification is to assign one or more of the concepts to each of the images. The concepts may describe the image contents, like 'car', 'person' or 'landscape', as well as more abstract characteristics, e.g. whether it is over- or underexposed. This task is commonly solved by describing the images as vectors of local and global features, such as color and gradient histograms. Supervised machine learning techniques are subsequently used to learn how to separate positive from negative examples of each concept class.

One approach that is commonly used to represent images is the Bag-of-(Visual-)Words model (BoW), which extends an idea from text retrieval to visual classification [3, 8]. In text classification systems, each text can be represented by a histogram of the frequencies of each possible word from a given vocabulary. That is, a text is characterized by the number of occurrences of each possible word, independently of the word order. Similarly, we describe an image as a frequency distribution of *visual words*. While the notion of a word in natural languages is clear, visual words are less easy to describe. Typically local image characteristics are used to represent visual words such as intensity changes, i.e. the gradients in an image region.

By representing images as histograms of visual words, they are mapped into a high dimensional vector space. Classification of images can then be considered as a supervised learning problem trying to separate vectors corresponding to positive from those corresponding to negative example images of a given concept class. Each vector dimension represents a specific visual word and gaining insight into the influence of each dimension to the overall classification result may enable us to gain an understanding of the concept model, that is learned. Thereby, sources of misclassification can be better explained and, finally, classification results may be improved by using these insights.

Therefore, the goal of this project is to evaluate means to visualize the influence of visual words on the overall classification result, in terms of a heatmap. This can be achieved by first evaluating the importance of each feature to the trained concept model and then assigning to each pixel of the image an importance value based on the visual words it contributes to.
This paper is structured as follows: In Section 2 we give some background on supervised machine learning in general and ensemble methods in particular that are used within this work to learn a concept model. Next, we present implementation details of the proposed approach in Section 3 and show where and how Future SOC Lab resources were used in order to increase the training performance. Section 4 provides a short analysis of our results and finally Section 5 summarizes the paper and gives a brief outlook of the work we are planning to do in the future.

2 Visual Concept Modeling

As discussed in the introduction, we perform visual concept classification by applying supervised machine learning techniques to Bag-of-Visual-Words feature vectors. In this section we therefore briefly introduce Bag-of-Visual-Words feature extraction and describe the machine learning approach chosen to train a visual concept model. Different from standard approaches for Bag-of-Visual-Words classification [3, 8] we do not use kernel-based Support Vector Machines since these do not allow for deriving feature relevance from a learned concept model.

2.1 Bag of Visual Words Extraction

Following [3, 8] we use local histograms of gradients to describe local properties of images that form a visual word space. The Scale-Invariant Feature Transform (SIFT, [5]) is the most widely used local gradient descriptor for Bag-of-Visual-Words representations. SIFT computes orientation invariant histograms of gradients at selected regions of interest (i.e. keypoints) in an image. While this approach is especially useful when matching image regions, in image classification one usually strives for a holistic image representation and therefore computes SIFT descriptors at densely sampled keypoints. We quantize local features by clustering a random subset of SIFT vectors extracted from the training image using k-means clustering. The derived centroids form our vocabulary, which is fixed to a size of $k = 4000$ visual words, a common size in related research. Finally, for each image a nearest neighbor search is performed to assign each keypoint's SIFT-descriptor to the most similar visual word in the vocabulary. The frequency distributions of the visual words within an image is represented as a histogram consisting of k bins, one for each visual word. This histogram is then normalized forming a vector of k real numbers, which represents the given image.

2.2 Supervised machine learning

Given the set of possible instances X, the set of possible labels Y and a set of labeled examples $\{(x_1, y_1), .., (x_n, y_n)\}$ with $x_i \in X$ and $y_i \in Y$, the task of supervised machine learning is to infer a function $h : X \to Y$, that correctly maps the x_i to y_i. The function h maps from the space of all possible inputs X to a finite set of outputs Y and is often called the *hypothesis* of the machine learning algorithm.
For the task at hand, the elements of X would be abstract representations of the given images such as Bag-of-Visual-Words representation and the elements of Y correspond to the concept labels (e.g. 'landscape'), which shall be learned. Typically in image classification, the problem of detecting $|Y|$ different concepts is treated as a binary classification problem where the task is to separate images into two groups on the basis of whether they depict a specific concept or not.
For a set of labeled examples

$$S = \{(x_1, y_1), .., (x_n, y_n)\}$$

and a hypothesis h we define the (training) error rate ϵ_h of h to be

$$\epsilon_h = \frac{1}{|S|} \cdot \sum_{i=1}^{|S|} I(h(x_i) \neq y_i)$$

where $I(\cdot)$ is the indicator function. That is, the error rate of hypothesis h is the fraction of samples, that are incorrectly labeled by h.
We distinguish between weak and strong learning algorithms. A strong learning algorithm is one, that can achieve an arbitrarily low error rate with arbitrarily high probability, given sufficient time and training samples. In contrast, weak learning algorithms are only required to perform better than random guessing. That is, their error rate has to be smaller than 0.5 in case of a binary classification task.

2.3 Boosting and bagging algorithms

Schapire [7] proved that it is possible to combine multiple weak learners into a single strong learner. That is, given a sufficient number of learning algorithms, that perform only slightly better than a random guess, we can create an algorithm that predicts the data with an arbitrarily small error rate.
This finding justifies the use of a number of learning algorithms known as *ensemble methods*. Ensemble methods try to achieve high predictive performance by combining multiple learners, that are diversified either by training them on different (random) subsets of data (*bootstrap aggregating* or *bagging*, see [1]) or by iteratively building models that emphasize samples which

where mis-classified by previous models (*boosting*, see [7]). Two prominent examples of each of these classes of algorithms are *AdaBoost* [4] by Freund and Shapire and *Random Forests* [2].

AdaBoost is an iterative algorithm, which iterates over training a weak learner using the set of (weighted) training examples $\{(x_1, y_1)...(x_n, y_n)\}$. It is common to choose the sample weights $\{w_1, .., w_n\}$ to be uniformly distributed for the first iteration. After each iteration the weights are updated proportionally to the error rate of the current weak learner, assigning larger weights to samples, which have previously been mislabeled. Thus, intuitively speaking, samples with higher weights are more 'difficult' to classify than others and the weak learner is forced to 'concentrate' more on classifying them correctly. After a defined number of iterations T, the algorithm stops and outputs its final hypothesis, which corresponds to a weighted sum of all the weak learner's predictions. Obviously the choice of the weak learning algorithm has some impact on the outcome of AdaBoost. Typically decision stumps[1] are used, as they are very easy to implement and efficiently computable.

Another well known algorithm that combines many simple decision trees into a strong learning algorithm is the Random Forest algorithm. It trains each weak learner on a randomly selected subset of the training data and at each node chooses a random subsample of the features to determine the decision for that node from. For the classification task, the final output of a Random Forest is the mode[2] of the outputs of all its trees. The weak learners in the case of Random Forests are full-grown decision trees.

While decision stumps as well as decision trees provide ad hoc methods for computing individual feature importance, they lack enough expressiveness to correctly represent complex problems such as visual concept modeling. However, following [7], we use AdaBoost and Random Forests that allow for complex problem space representations as well as for feature importance estimation simply by computing the weighted sum of importances of the individual weak learners. This makes boosting and bagging algorithms an ideal choice for this project, as they achieve better classification performance than the underlying, simpler algorithms, while still allowing us to easily assess the importance of single Bag-of-Visual-Words features for classification.

The computation of an individual weak learner is typically quite fast. However, we need to train several thousand weak learners for each AdaBoost classifier. Though the run-time scales linearly with the number of weak learners, one training session may need several hours to complete. Considering the fact, that we need to train classifiers for many different parameter combinations (see Section 3.2) and as many different concept classes as possible, this project would soon become infeasible without utilization of the Future SOC's parallel architecture.

Bagging methods such as Random Forests provide a straightforward parallelization strategy, as all the weak learners operate completely independent from each other on different subsets of the data. With boosting algorithms, in particular AdaBoost, it is not that simple, as this is an inherently iterative algorithm and each iteration depends on the preceding one. Nevertheless, we can still benefit greatly from parallel architectures simply by optimizing different parameters of AdaBoost concurrently through parallel grid search (see Section 3.2).

[1] decision trees with only one or very few decision nodes

[2] the classification value, that appears most often

3 Implementation and Future SOC Lab resources

This section describes the implementation aspects of our approach focusing on the exploitation of the Future SOC SMP multicore architecture. We used a machine equipped with 24 processor cores and 64 GB of main memory. Development and testing was done on a local machine and the Future SOC was used to perform computational expensive and long-running tasks on the complete dataset. This includes feature extraction, training and grid search, and visualization. All of these tasks were parallelized and benefit largely from a multi-core architecture.

For training and testing purposes we use the publicly available ImageCLEF 2011 dataset for visual concept classification [6], which contains 8.000 training samples, each manually labeled with a subset of 99 image categories. We evaluate our classifier on a set of 10.000 images, also provided by ImageCLEF.

3.1 Feature Extraction

The feature extraction process includes several expensive computations, namely the computation of many SIFT descriptors, the clustering of a large number of high-dimensional vectors and nearest-neighbor search in their vector space. Therefore, in order to increase the performance of Bag-of-Visual-Words extraction, each step of the process runs concurrently, making use of the available Future SOC multicore node. Extraction of SIFT features on the 18.000 training and testing images took approximately 1 hour compared to one full day when computed on a single core (linear scaling is assumed here, since images are scaled to equal size and thus the number of SIFT features extracted per image is the same). Generating the prototypes of visual words using k-means clustering can also be parallelized since it consists of mainly independent distance computations between training vectors. In our implementation clustering runs for 1 hour compared to an approximate 3-4 weeks of a respective serial implementation. Finally, nearest-neighbor search for com-

puting the Bag-of-Visual-Words histograms of each image can be parallelized over all images and therefore again provides a linear speed-up of factor 24 (1 hour compared to 24 hours).

3.2 Concept Training and Grid Search

In our first implementation we focused on training of concept models using AdaBoost. There are a number of parameters that influence the performance of AdaBoost, namely the number of iterations T, the maximum depth of each decision stump and the optimization criterion of the decision stumps. It is good practice to optimize these parameters by performing a grid search and by using $n-$fold cross validation. We selected reasonable parameter ranges for each parameter and created a parameter grid by generating all possible combinations of the respective values. The training set is split into 3 stratified subsets where 2 folds are used for training the AdaBoost strong learner and the remaining fold is used to test the derived hypothesis. By iterating over all possible folds, we make sure to use the whole training set as training data while avoiding testing on training data at the same time (3-fold cross validation).
We used extensive grid search to find a parameter combination that shows best results on the training set. While AdaBoost itself is hard to parallelize, we utilized the Future SOC's parallel architecture to evaluate as many parameter combinations in parallel as possible. As the models for each parameter combination can be learned independently of each other, this greatly reduces the time needed for grid search and thus enabled us to explore a much larger space of combinations in the given time. Hence, grid search was parallelized by delegating the training for each parameter combination to a dedicated core. This lead to an AdaBoost classifier, that shows promising results of 73.56% average precision on the test set for one of the concepts ('landscape').

3.3 Visualization

In order to visualize the impact of the various visual words on the classification result, the importance of each feature for the obtained AdaBoost model needs to be computed. This is done by computing the importance of the feature for each weak learner, which corresponds to the information gain that is achieved when this feature is used in a decision node. These weak importance values are then weighted by their corresponding weak learner's weight and added together to get the feature importance for the overall AdaBoost classifier. Each feature in a visual word vector corresponds to a visual word in the aforementioned vocabulary. Hence, since each local SIFT vector in an image is directly mapped to the most similar visual word we can directly relate the visual word importance to all corresponding SIFT vectors and thus to the respective keypoints where these features have been extracted. By assigning each keypoint in an image the importance value of its corresponding visual word we create an importance map of all pixels in the image for the overall classification result depending on the computed concept model. This is done by setting the importance of a single pixel to be the weighted sum of the importances of all keypoints which are near that pixel. The rationale for this approach is that visual words were formed by computing SIFT descriptors around keypoints and therefore using pixel information in an area around that keypoint. Therefore, each pixel in that area has some kind of influence on the resulting visual word and thus on its importance for classification. As SIFT weighs the influence of pixel information on the final descriptor by applying a Gaussian, so do we to reduce the importance of pixels with increasing distance to their corresponding keypoints. We visualize the importance map by rendering a heatmap (see Fig. 1).

Again, we rely on the Future SOC parallel architecture to reduce the time needed for this process, as multiple images can be processed in parallel. For the computation of the heatmaps a near-linear speed-up can be achieved. However, disk I/O represents a bottleneck for the parallelization of this stage, as the resulting images need to be written to disk.

4 Results

At the time of this writing, we extracted Bag-of-Visual-Words features for the complete ImageCLEF dataset. We trained, evaluated and visualized an AdaBoost classifier for the image concept 'landscape', which is a scene related concept meaning that the important features will most likely be spread over the image plane as a whole. Some of the resulting images can be seen in Fig. 1.

The visualization shows no obvious pattern in examples that were classified as positives, although occasionally an emphasis on horizontal edges can be seen, which may indicate that the horizon line is an important indicator for images depicting landscapes. This finding is consistent with our initial assumption. However, on many negatively classified images, large areas of homogeneous and very important features can be seen (see Fig. 1(d)). This seems to imply that those large image regions have high discriminative value in the sense that the existence of such a region is an indicator for a 'non-landscape'.

This result suggests, that the learned model is less of a representation of what is typical for landscapes and more a characterization of everything that is not. However, a more thorough analysis of this and other concepts is needed to verify this assumption.

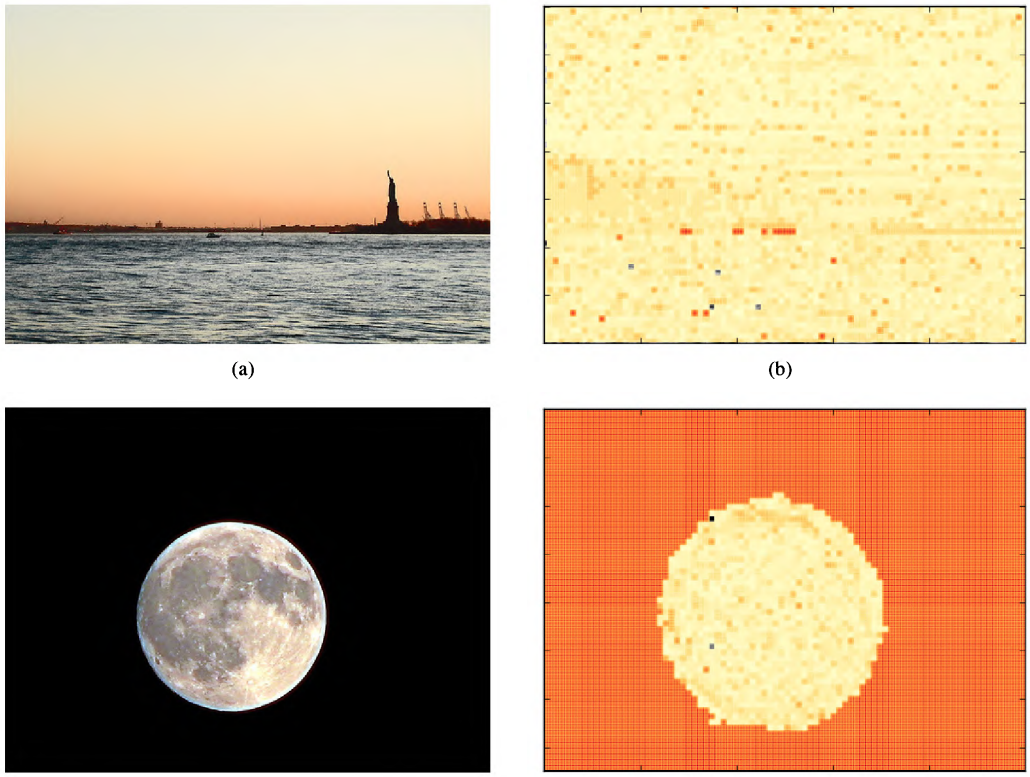


Figure 1: Original image and resulting heatmap for a positive (top row) and a negative (bottom row) sample of the image class 'landscape'.

5 Future Work

As a next step we need to verify the assumption that there are features with high discriminative power, that are exclusively seen in either positive or negative sample images. Furthermore, we intend to give evidence to which class of features has the highest impact on the overall classification.

Also, we hope to gain some insights into the process of visual concept classification by training the classifier on further concepts and comparing the results with each other. This may lead to the discovery of similarities and differences in the way different concepts are represented in the corresponding Bag-of-Visual-Words model and eventually enable us to improve classification results by using these insights to create more meaningful visual words.

The entire analysis shall also be performed with other machine learning methods like the aforementioned Random Forests or linear SVMs, to show if there are indeed visual words that characterize a given concept and are therefore used in its model representation, independent of the learning algorithm.

For each combination of algorithm and learned concept, we need to re-run the grid search, training and visualization steps. Thus, our work will still benefit greatly from parallel architectures like the FutureSOC.

References

[1] L. Breiman. Bagging predictors. In *Machine Learning*, pages 123–140, 1996.

[2] L. Breiman. Random forests. *Machine Learning*, 45, 2001.

[3] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, C. Bray, and D. Maupertuis. Visual Categorization with Bags of Keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[4] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1997.

[5] D. G. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, 2004.

[6] S. Nowak, K. Nagel, and J. Liebetrau. The clef 2011 photo annotation and concept-based retrieval tasks. In *CLEF (Notebook Papers/Labs/Workshop)*, 2011.

[7] R. E. Schapire. The strength of weak learnability, 1990.

[8] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, number Iccv, pages 1470–1477. IEEE, 2003.

Landscape Virtualization Management tools at the HPI Future SOC Lab

Rami Akkad, André Deienno Pansani, Felix Salfner
SAP Innovation Center
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam
{rami.akkad, andre.pansani, felix.salfner}@sap.com

Abstract

On-Demand services can only be provided in a profitable way if they are operated at minimum cost. Therefore, automated monitoring and task management are indispensable. Today, On-Demand providers use custom-made monitoring and management solutions. It is the goal of this project to assess the SAP LVM and HP Converged Cloud maintenance tools used at the FutureSOC Lab. We provide a classification of functionality (e.g. mass operations, reporting, monitoring, provisioning of services) supported by practical experiences from managing Future SOC Lab resources with up to 16 SAP HANA instances, a 1000 core cluster, storage solutions, etc.

1 Introduction

The "HPI Future SOC Lab" is a cooperation of the Hasso-Plattner-Institut (HPI) and industrial partners. Its mission is to enable and promote exchange and interaction between the research community and the industrial partners. The HPI Future SOC Lab provides researchers with free of charge access to a complete infrastructure of state of the art hard- and software. This infrastructure includes components, which might be too expensive for an ordinary research environment, such as servers with up to 64 cores and a 1000 core cluster. The offerings address researchers particularly from but not limited to the areas of computer science and business information systems. Main areas of research include cloud computing, parallelization, and In-Memory technologies.

In total 131 projects from 34 institutes were conducted using the HPI FSOC Lab infrastructure since its launch in 2010.

Given the diverse and huge infrastructure the Future SOC Lab comprises of, automated monitoring and task management are indispensable to plan ahead and reduce costs.

2 Used Resources

SAP LVM is a management tool that enables the SAP basis administrator to automate SAP system operations including end-to-end SAP system copy/refresh operations. The HP Converged Cloud powered by Openstack [6] offers a portfolio of comprehensive cloud solutions to build, operate, and consume IT services that are open and enterprise-class across private, managed, and public clouds.
We compared SAP NetWeaver Landscape Virtualization Management (SAP LVM) Enterprise Edition version 2.0 and the HP Converged Cloud with HP CloudSystem Matrix version 7.2 Update 1. We classified functionalities within both maintenance tools based on white papers and other information provided by SAP and HP, and on experiences running those tools inside the Future SOC Lab infrastructure.
Furthermore we compared CPU load and memory utilization monitoring of both tools.

3 Feature Comparison and Assessment

We compared three categories which are

1. supported virtualization providers,
2. supported storage systems and
3. operations automation.

3.1 Supported virtualization providers

We compared SAP LVM and HP Converged Cloud regarding their support of virtualization technologies of the following providers: Microsoft Hyper-V, VMware, KVM, Amazon, IBM and Novell.
Both SAP LVM as well as HP Converged Cloud support VMware vCenter (LVM VIM API 2.5 onwards) [3, 4]. Only HP Converged Cloud supports Microsoft Hyper-V [3, 4] and the KVM API [3, 4]. However, only SAP LVM supports the Amazon AWS API [4], IBM Hardware Management Console (for Power - HMC V7R3.5.0 and for System z - HMC 2.10.1), IBM

Systems Director 6.2.1 with VM Control 2.3.1 [4] and Novell PlateSpin Orchestrate 2.0.2 [4].

3.2 Operations Automation

Both the assessed tools have a different focus. The HP Converged Cloud focuses in infrastructure provisioning and operation, whereas SAP LVM focuses in SAP application provisioning and operation.

The HP Converged Cloud is used to design provisioning templates (e.g. SLES, Microsoft Windows, VMware ESXi, Microsoft Hyper-V, etc.), create services, assign costs to services, create organizational units and assign virtual and/or physical resources to them (e.g. blades, networks, storage, provisioning templates, etc.) which the organization can independently use by deploying services on them. The HP Converged Cloud also supports automated backup and on-the-fly recovery in cases of hardware malfunction. It is also able to perform mass operations on deployed services, e.g. starting and stopping multiple servers.

SAP LVM simplifies and automates the management and operations of SAP systems running on traditional, virtual or cloud infrastructures. It enables administrators to perform mass operations on SAP systems, e.g. starting and stopping an SAP ERP or SAP HANA system. SAP LVM also supports cloning, copying, relocation, live, migration and scaling of SAP systems.

4 Monitoring Capabilities

Even though the HP Converged Cloud and SAP LVM were not developed as monitoring tools, they have some monitoring capabilities.

The HP Converged Cloud is able to monitor compatible systems in regard to availability, performance and more. For instance, the HP Converged Cloud is able to detect hardware malfunction of a connected blade and can automatically provide a failover blade on the fly to avoid or reduce downtime. It also records disk usage, CPU load, network usage and memory utilization.

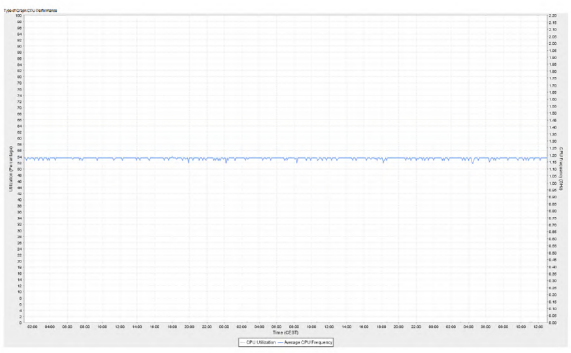


Figure 1: CPU load visualized by the HP Converged Cloud

SAP LVM monitors host metrics like CPU and memory usages. Furthermore, SAP LVM is able to monitor a variety of SAP application specific metrics, e.g. database or front-end response time, number of users, ABAP dialog work processes, Java metrics, state and availability of SAP applications. The navigation through the SAP LVM monitoring reports is user friendly. There is even the possibility to include multiple systems into one single report to compare the same metric within different systems, which makes a lot of sense in case SAP applications run multiple hosts.

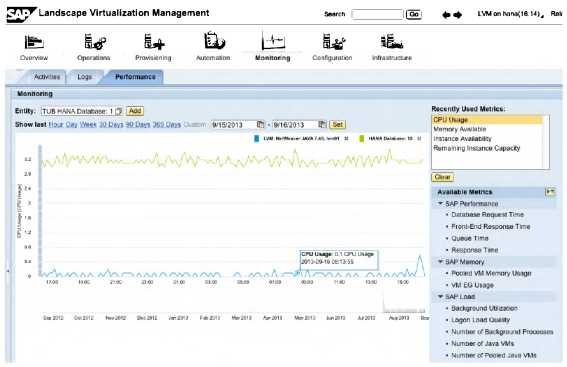


Figure 2: CPU load visualized by SAP LVM

5 Conclusion

In this report we assessed the HP Converged Cloud private cloud solution and the SAP Landscape Virtualization Management software. SAP LVM provides deeper functionalities in regard to SAP application operation, whereas the HP Converged Cloud acts as a private cloud solution which can be used on-premise.

6 References

[1] SAP Netweaver Landscape Virtualization Management Software Integration With EMC, EMC Corporation, 2012.

[2] SAP Netweaver Landscape Virtualization Management: An Overview of Use Cases, SAP AG, 2012.

[3] HP CloudSystem Matrix 7.2 Update 1 Compatibility Chart v7.2.1.0, Hewlett-Packard Development Company, L.P., 2013.

[4] SAP Note 1783702 SAP NetWeaver Landscape Virtualization Management 2.0, SAP AG, 2013.

[5] SAP NetWeaver Landscape Virtualization Management Software Overview, SAP AG, 2012.

[6] HP Converged Cloud Solution Brief, Hewlett-Packard Development Company, L.P., 2013.

Implementation of a module for

Predictive Analysis Library (PAL)

Prof. Dr. Ali Reza Samanpour
Fachhochschule Südwestfalen
Lindenstraße 53
59872 Meschde
samanpour.ali-reza@fh-swf.de

André Ruegenberg
Fachhochschule Südwestfalen
Lindenstraße 53
59872 Meschede
ruegenberg.andre@fh-swf.de

Abstract

This report describes the implementation of a prediction-system, that consists of one or many differently structured neural networks (feed forward networks, radial basic function networks, etc.), which are consolidated via a gating network and are overall trained to minimize the fitness function.

Such a multi expert system is able to analyze heterogeneously constructed data spaces very well, because different experts can individually concentrate on special areas of the data space and one particular prediction system cannot include the data space's entire topology to, for example, sufficiently approximate a time series in its whole.

1 Introduction

Why neural networks?

There are categories of problems, which cannot be covered by inflexible algorithms, e.g. the determination of stock prices. The solution to such a problem depends on a vast variety of factors and the human brain can roughly estimate it, but due to the limitations of algorithms a computer cannot.

Computers can perform numerical calculations with great speed, but they do not adapt intelligently to new problems. This brings us to the question of learning: How does the human brain learn to solve problems?

If we oppose the human brain to a computer, we discover that theoretically the computer should be more effective, because its transistors switch faster than the neurons in the human brain. However in the human brain most of the data is constantly being processed, while most of the data in computers is merely stored. Therefore the human brain works close to its theoretic maximum capacity most of the time, while a computer processes data serially, which means an exponential difference in switching performance. In addition our brain, as a biologic neural network, is able to adapt to problems while running, to restructure and therefore learn, compensate errors and generate solutions.

Thus a neural network does not have to be programmed according to specific problems, it can find plausible solutions for problems by using examples of similar problems. This also means a bigger error tolerance with fuzzy input data. Error tolerance again stands in close relation to the human brain, which is even so error tolerant, that we can read different handwritings, although particular letters may be unreadable.

On the contrary our "oh-so-modern" technology is not automatically error tolerant. For example a computer with a defective hard drive controller cannot assign the job to its graphic board with the aim of keeping the system running in opposition to have it fail completely.

It definitely is a disadvantage that in the divided, error tolerant neural network, it is not easy to see what it knows or where its errors are. Furthermore new knowledge can only be put into such a system using a learning process, in which several errors can occur and which is not always easy to facilitate.

2 Net Modelling

Now let us sum up the outstanding characteristics of the human brain that we can try to implement into our technology:

- Compliance
- Ability to generalize and associate
- Error tolerance

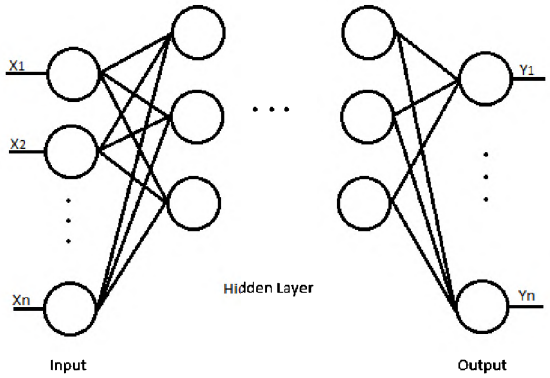


Figure 1: Artificial neural network (own illustration)

But it has to be mentioned that a neural network, which is able to solve any given problem does not exist. There are many different types of networks, learning methods and cases in which they can be used. Every network is bound to its original problem to a certain degree and is therefore able to solve similar problems, but can never replace its natural architecture model.

3 Application Example

We want to teach a neural network a mathematical equation having two input dimensions: z = f(x,y).

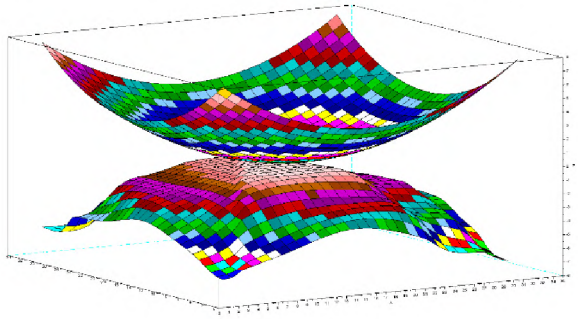


Figure 2: Top training data of example of a 3-dimensional function -, below the "learned" output (own illustration) mirrored

For successful learning it is essential to have plausible data, with which the network can be trained. A training example consists of an exemplary input and a proper output.

4 Training

Now the question is, how this information can be transferred into the network. The Input of a technical neuron has many components, like a biological neuron also can have many input signals, basically a vector. It can only have one output signal, which however can be applied to many neurons in the next layer. This means, that somewhere inside the neuron the many input signals have to be merged to one scalar. Many scalar outputs again build a vector input for another neuron. As in the biology the input data is being merged and, while transiting to the next neuron, preprocessed with emphases – this means it gets multiplied by a factor assigning a certain priority to the input data. The priorities that are assigned to the input data are variable, which gives a great dynamic to the network. Therefore the knowledge is mostly being stored and presented including these priorities (p). In consequence a neurons' output is calculated like this:

$$y = f\left(\sum_i w_i * x_i\right)$$

Now the question is, how this information can be transferred into the network. The Input of a technical neuron has many components, like a biological neuron also can have many input signals, basically a vector. It can only have one output signal, which however can be applied to many neurons in the next layer. This means, that somewhere inside the neuron the many input signals have to be merged to one scalar. Many scalar outputs again build a vector input for another neuron. As in the biology the input data is being merged and, while transiting to the next neuron, preprocessed with emphases – this means it gets multiplied by a factor assigning a certain priority to the input data. The priorities that are assigned to the input data are variable, which gives a great dynamic to the network. Therefore the knowledge is mostly being stored and presented including these priorities (p). In consequence a neurons' output is calculated like this:

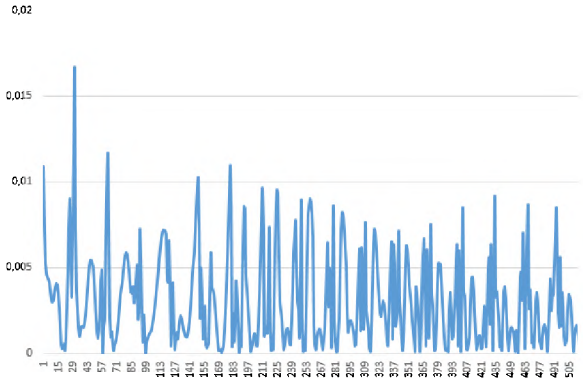


Figure 3: Network error of individual records to Fig. 2

5 Conclusion

Through the complete virtualization for in-memory-environments, in which SAP HANA runs, it is possible to operate many different projects simultaneously and in a significantly more efficient manner.

From the explained approach we can derive an evaluation of complex, interconnected data to support decisions. This is an important aspect in which lies a great technologic challenge. Classic programming methods face their limitations in today's complex online world. The challenge is, to see complex connections in massive amounts of data and create precise prognoses from it. The combination of learning systems in combination with the HANA technology, which can process hugest amounts of data in real time, is superior to other systems. With this combination it is possible to identify relevant connections in data masses and provide precise prognoses.

6 Perspective

The data areas, which count as basis for economically relevant processes in the real world, are usually multidimensional and not linear to a maximum degree.

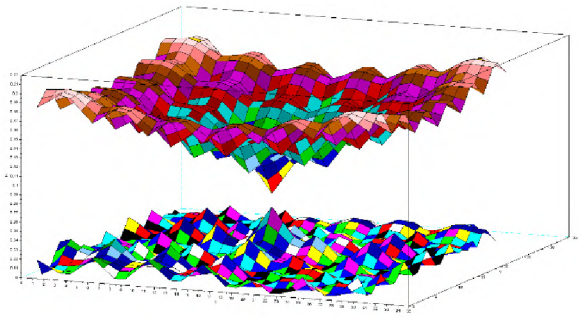


Figure 4: Ackley function - top training data, below mirrored the learned Network Edition

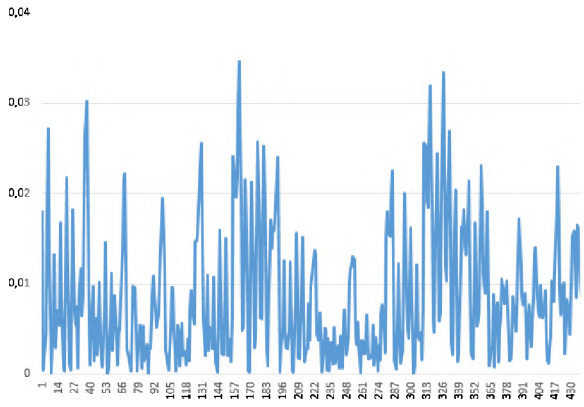


Figure 5: Network error

For this reason the fitness functions are equipped with different extreme values. These are difficult to avoid if using the classic optimizing methods. Here it can be helpful to use evolutionary algorithms, which ignore the extreme values and find significant local/global optimum values, which lead to considerably better results in the prediction systems. The error of the prediction due to approximation of not linear time series can be exponentially optimized. Using evolutionary strategies as optimizing methods leads to robust solutions. Through this direct method to optimize parameters it is possible to analyze flexible fitness functions, which no longer need a calculation of gradients or multivariate functions.

The time series used for testing can consist of artificially created data, which is deterministic, but chaotic and not linear. Therefore in further steps real data from real business processes should be used.

References

[1] Raúl Rojas: Theorie der Neuronalen Netze. *Springer Verlag,* 1994, ISBN 978-3540563532

[2] Amit Konar: Computational Intelligence, *Springer Verlag,* 2005, ISBN 3-504-20898-4

[3] Bruce Ratner: Statistical an Machine-Lerarning Data Mining, *CRC-Press,* 2012, ISBN 978-1439860915

[4] Heinrich Braun: Neuronale Netze: Optimierung durch Lernen und Evolution, *Springer Verlag,* ISBN 987-3642645358

Next Generation Sequencing: From Computational Challenges to Biological Insight

Cornelius Fischer
cfischer@molgen.mpg.de

Annabell Witzke
witzke@molgen.mpg.de

Dr. Sascha Sauer
Max Planck Institut for Molecular Genetics, Nutrigenomics and Gene Regulation
Otto-Warburg-Laboratory
Ihnestrasse 63-73, 14195 Berlin
sauer@molgen.mpg.de

Abstract

In this project age-associated and inflammatory pathologies shall be studied using approaches such as chromatin immunoprecipitation (ChIP-seq) and transcriptome profiling (RNA-seq) coupled to next generation sequencing (NGS) technologies. While NGS enables profound analyses of biological processes, the main bottleneck remains in the computational analysis and interpretation of the generated data. Using Future SOC Lab resources we established a computational pipeline for RNA-seq (Fall 2012) and ChIP-seq (Spring 2013) data analysis. The utilized DL980 G7 server worked robustly and facilitating fast analysis of large data sets.

1 Project idea

In the laboratory we exposed cultured cells for a defined period of time to a mild stress-inducing agent. DNA obtained from these experiments was subjected to chromatin immunoprecipitation (ChIP). Combining ChIP with next generation sequencing (NGS) technology enables thorough determination of treatment-induced changes (for any protein of interest) in the DNA-protein interaction on a genome-wide scale. In the past years a basic ChIP-seq analyses pipeline was established, which can be altered in multiple ways and thus accustomed to individual needs. Using Future SOC Lab resources we were able to establish and make use of a highly flexible and ultra-fast ChIP-seq analysis pipeline. This approach provided the initial framework for the analyses of future NGS-generated samples. Hence, further participation in the HPI Future SOC Lab would enable us to investigate stress-induced genome-wide changes involved in age-associated pathologies and inflammatory processes.

2 Used Resources

Having collaborated with the HPI Future SOC Lab for the third consecutive term, we were able to work on the same hardware environment; most changes were made on the software layer.

2.1 Hardware

As in the previous terms, we worked on a Hewlett Packard DL980 G7 server equipped with eight 8-core Intel Xeon X7560 processors and 2048 GB of DDR3-1066 main memory. Since the Intel Xeon X7560 provides Hyper-threading, 128 logical processors were available for data analysis. This proved to be a very powerful system, perfectly suiting our purposes.

2.2 Software

Ubuntu Server 12.04.2 LTS was used as an operating system. Due to recent software updates concerning ChIP-Seq analysis, new versions of Bowtie 2 (aligner) and MACS (peak caller) were utilized.

3. Methods and tools

For ChIP-Seq analysis, the pipeline depicted in figure 1 was used. In brief, Illumina HiSeq 2500 output was aligned to the human reference genome February 2009 assembly (GRCh37/hg19) [2] using Bowtie 1 [4] (version 1.0.0) or Bowtie 2 [3] (version 2.1.0). After converting aligned reads from ASCII-based SAM to binary BAM format, output files were sorted and indexed using SAMtools [5] (version 0.1.19). For reasons of additional condensation and subsequent visualization in the UCSC Genome Browser, sorted BAM files were converted to binary bigWig format via the ASCII-based wiggle track

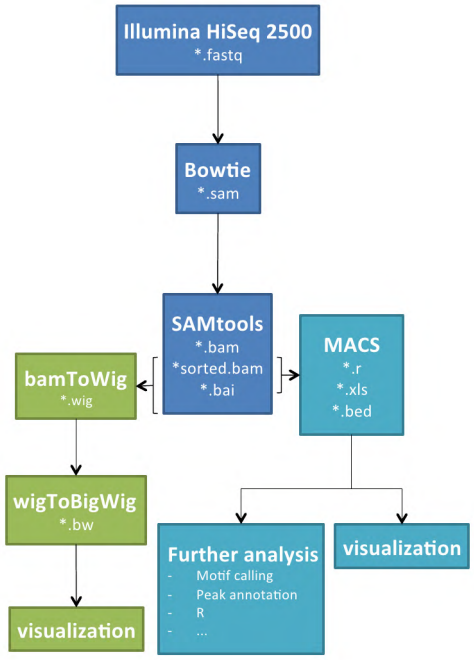


Figure 1: ChIP-Seq analysis pipeline.

format using publicly available bamToWig (RSeqC package [7], version 2.3.7) and wigToBigWig (UCSC Genome Browser [1]) utilities. Besides visualizing alignment results, transcription factor binding sites and significant enriched ChIP regions were identified and evaluated using MACS [8] (version 1.4 or 2.0).

4 Findings

To exploit the resources of the provided machine, the major challenge was retrieving ideal launch parameters for all tools in order to make use of parallel resources.

4.1 Bowtie

While Bowtie achieved almost linear speedup up to utilizing approximately 16 threads per instance, speedup dropped to sub-linear level in the range of 16 to 32 threads per instance and significant slowdown was observed for larger number of threads per instance. Since we had to process 16 samples in total, we decided to launch 16 instances of Bowtie in parallel, each using 8 threads. In order to reduce disk access, the output of Bowtie was piped into SAMtools, thus converting SAM output to BAM format on the fly. While reducing overall I/O load, this resulted in an additional thread next to the 8 threads of Bowtie itself. Furthermore, not all samples contained an equal number of reads, thus resulting in some instances of Bowtie to finish earlier than others. Generating a total number of 144 threads turned out to deliver ideal overall performance due to keeping all CPUs busy as long as possible.

4.2 Conversation steps

Since Bowtie is the only tool explicitly making use of multithreading, launching multiple instances of SAMtools, bamToWig and wigToBigWig was the only way of leveraging parallelism for the conversation steps. As a consequence of handling large data volumes on persistent storage, the conversation tasks were mainly I/O limited and the NFS-based home directories turned out to be the main bottleneck. Since SAM files are approximately 3 to 4 times as large as their binary BAM counterpart and wiggle files are roughly 5 to 6 times as large as their binary bigWig counterpart, binary representations were preferred in order to minimize I/O load. Furthermore, keeping as much data as possible in main memory by using pipes between process calls, as well as moving conversation tasks to local disk storage enabled us running up to 16 conversation pipelines in parallel without any performance penalties.

4.3 MACS

With the peak caller heavily relying on the Numpy library [6], our hope was to indirectly achieve speedup by using a version of Numpy built against the Intel Math Kernel Library. Unfortunately, we were unable to measure any speedup compared to the vanilla build of Numpy. As a result, we had to resort to launching multiple instances of MACS in parallel.

5 Outlook

Hitherto, the majority of experiments have been conducted using pooled cell populations. Single cell analyses can provide unique opportunities to discover exclusive characteristics of a diseased cell state and to develop new compounds for prevention and treatment. The analysis of a cell population on single-cell level represents a huge challenge on an experimental and computational level. While aligning 16 samples took between 4 and 6 hours on a 8-core workstation (2x Intel Xeon E5462, 16GB DDR2-800 main memory), aligning and converting to BAM was achieved in less than 20 minutes on the resources provided by the HPI Future SOC Lab. In the future, we need to process a larger number of samples, including 48 ChIP-Seq samples and approximately 384 single cell RNA-Seq samples. Further participation in the HPI Future SOC Lab enables us to process all those samples in a feasible amount of time. Furthermore, short processing times allow for new levels of interactive, flexible and up-to-date data analyses.

References

[1] W. J. Kent, C. W. Sugnet, T. S. Furey, K. M. Roskin, T. H. Pringle, A. M. Zahler, and D. Haussler. The human genome browser at ucsc. *Genome research*, 12(6):996–1006, 2002.

[2] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001.

[3] B. Langmead and S. L. Salzberg. Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357–359, 2012.

[4] B. Langmead, C. Trapnell, M. Pop, S. L. Salzberg, et al. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biol*, 10(3):R25, 2009.

[5] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, et al. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[6] T. E. Oliphant. Python for scientific computing. *Computing in Science & Engineering*, 9(3):10–20, 2007.

[7] L. Wang, S. Wang, and W. Li. Rseqc: quality control of rna-seq experiments. *Bioinformatics*, 28(16):2184–2185, 2012.

[8] Y. Zhang, T. Liu, C. A. Meyer, J. Eeckhoute, D. S. Johnson, B. E. Bernstein, C. Nusbaum, R. M. Myers, M. Brown, W. Li, et al. Model-based analysis of chip-seq (macs). *Genome Biol*, 9(9):R137, 2008.

Detecting biogeographical barriers - testing and putting beta diversity on a map

Johannes Penner
Museum für Naturkunde, Leibniz-Institute for Evolution & Biodiversity Research, Department of Research, Herpetology Group
Invalidenstrasse 43
10115 Berlin, Germany
johannes.penner@mfn-berlin.de

Moritz Augustin
Technische Universität Berlin, Department of Software Engineering & Theoretical Computer Science, Neural Information Processing Group
Marchstr. 23
10587 Berlin, Germany
augustin@ni.tu-berlin.de

Abstract

Biogeographical regions are not only of scientific but also of conservation importance. The most common approach is to map species richness. However, beta diversity (species turnover) is better suited to identify where boundaries between regions are.

We calculated beta diversity for West African amphibians on a fine scale (30 arcseconds), using moving window and parallelization techniques. This would have been impossible with standard hard- and software. Not only was this the first time to produce such fine scaled maps but we systematically compared different indices and different moving window sizes.

Currently our results are only preliminary, the work plan has not been fulfilled yet and detailed analyses are pending. Furthermore we would like to ask for an extension of access possibilities and enhance our study significantly by switching from binary input data to proxies for abundance data.

1 Background

Amphibians form an integral part of biodiversity and are one of the most threatened vertebrate groups on the globe, with more than one third of all known species listed as threatened on the IUCN Red List [1, 2]. Biodiversity is declining in many areas and includes amphibians. The so called amphibian decline is a global problem and due to a number of reasons. The main ones are habitat destruction, alteration and fragmentation as well as pesticides, climate change, overharvesting and emerging diseases [3]. Causes are also often interacting. Thus, well-informed conservation decisions are needed, especially for areas which harbor the highest diversity but are poorly studied, i.e. the tropical regions. One base for such decisions is the answer to where and why species occur.

In general, amphibian diversity and biodiversity are measured on three different scales: genes, organisms and landscapes. Organismic diversity is the most common measure. Measured at one site it is called alpha diversity, the comparison between two sites is named beta diversity and gamma diversity is calculated across a landscape [4].

Besides the general interest where the highest species diversity is found, the distribution of diversity is important for conservation. One main aim is to set priority areas when efforts have to be concentrated. Another aim is to provide decision makers with the scientific sound arguments for areas in need of protection.

In general, a trade-off exists between the conservation side which requests fine grained maps and the scientific need to cover large areas. Studies fulfilling both requirements, covering a large area with a high resolution, are scarce. Our study is one of the few attempts to provide both. We study West African amphibians, which are unique and face multiple threats; mainly severe habitat destruction and fragmentation (e.g. see [5]).

2 Previous work

We used Environmental Niche Modelling (ENM; also commonly called Species Distribution Modelling) to derive the occurrence of West African amphibians on a grid of 30x30 arcseconds (≈ 1km²). So far ENMs were constructed for 158 out of ca. 180 known species with more than 9000 occurrence records. The ENMs used 18 environmental parameters: ten climate (different temperature and precipitation

measures), five vegetation (different wavelengths from two satellites), two altitudinal (calculated from a digital elevation model and one "hydrology" parameters) (compare to [6]). A machine learning maximum entropy algorithm [7, 8, 9] compared environmental parameters at sites of occurrences against randomly sampled background data (see [10] for a statistical explanation of the algorithm). This resulted in a map showing the distribution of modeled species richness covering the whole West African region (Penner *et al.* in prep.).

3 Project idea

Inspired by McKnight *et al.* [11] we wanted to investigate species turnover (≈ beta diversity) for the whole region. The main aim is to identify biogeographical regions and boundaries, which are reflected by relatively high species turnover. This is done by comparing species occurrences between neighboring grid cells.

To our knowledge this approach has never been tackled systematically, especially with real data, contrary to simulated data. No agreement exists for example which index should be used though some progress has been made. Therefore we will compare different indices and different moving window sizes.

Another aspect of biological diversity is species richness (≈ alpha diversity). We will compute different species richness measures and compare the model results with established maps used in biological conservation projects.

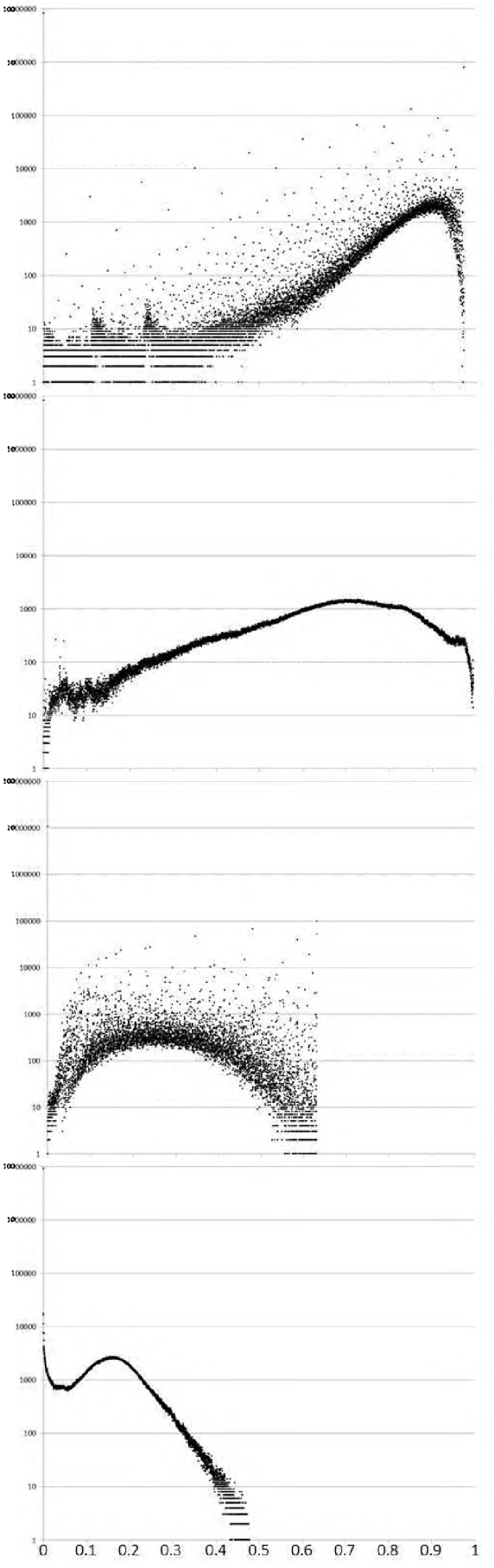
The final programmed software will be open source published using a public domain license. It will allow the user to specify own indices and moving window sizes.

4 Used Future SOC Lab resources

The large scale of our analysis does not permit the use of available standard computer hard- and software. Applying a moving window approach we utilized the HPI Future SOC Lab Fujitsu RX600S5 machines, parallelizing data. Though it would be interesting to test how the Hewlett Packard DL980 G7 servers would speed up calculations.

5 Findings

Currently we programmed two binary indices Jaccard [12] and Mountford [13]. Beta diversity is calculated as the species turn over between one grid cell and its neighboring cells. A third index, Raup-Crick [14], is currently being implemented. All three indices use binary occurrence data and weigh presence and absence data differently (see [15]). Moving window sizes varied from 3 to 93 (Mountford) and 201 (Jaccard). Larger sizes will be explored but are computationally very intensive.



Unique areas are consistently showing up regardless of the index and the moving window size. Preliminary analyses indicate that biogeographic barriers might correspond to major river systems in the region. However, at the current, stage proper conclusions are premature.

Figure 1 (from previous page): Example of frequency histograms of two different beta diversity indices with two different moving window sizes. From top to bottom (index name & moving window size): Jaccard 3x3, Jaccard 93x93, Mountford 3x3, Mountford 93x93.

Figure 1 shows four examples of frequency histograms for two indices (Jaccard & Mountford) and two different moving window sizes (3x3 & 93x93). The non-metric properties of the Mountford index are clearly visible. It is obvious that both factors are important. More thorough analyses will be prepared.

Recently, we have additionally computed alpha diversity both for binary data (species occurence count) and abundance data (exponential Shannon entropy). The preliminary results reveal significant areas of high species richness which have not been found by previous coarse-grained diversity estimation. However, more detailed investigations are necessary to make robust conclusions.

6 Next steps

Our previous runs were successful. Due to delays in the start of the project, analyses have not been completed yet and time and logistical constraints did not allow us to finish all intended runs. As mentioned above the third index awaits implementation.
In addition, we have discovered a solution to move the analyses even one large step further. So far we used binary data, resulting in relatively simple species richness maps. This was a result of the current work flow but has many theoretical disadvantages.

Therefore, we would like to use proxies for abundance. This would open a completely new window, incorporating such measures as heterogeneity and evenness in our analyses and potentially a much higher precision. Using "abundance" data, different indices are needed. Five potential candidates are: Shannon, Whittaker, Wilson & Shmida, NESS and C_{qN} which weigh evenness differently (e.g. see [16]).

Furthermore, using the likelihoods gained from the ENMs enables us to calculate alpha diversity anew, also incorporating heterogeneity and evenness. Although the preliminary results regarding species richness look very promising a general conclusion is not yet possible. Therefore we want to investigate this measure in more detail and relate it to previous findings.

7 Extension request

So far the possibilities of the HPI Future SOC Lab not only enabled our work but also took it a huge step forward. As mentioned above not all analyses were finalized yet and they should be further improved. A further use of the HPI Future SOC Lab facilities would help us tremendously. Therefore we would be very grateful if an extension of our access to the HPI Future SOC Lab resources is granted. A follow-up proposal has been submitted.

8 Acknowledgements

We thank the innumerous colleagues who shared their data. We are also indebted to a number of people who assisted in the field, in the laboratory and in various other ways, the most important ones are: Mark-Oliver Rödel, Jakob Fahr, Matthias Herkt, Günther Barnikel, Mirjana Bevanda, Mareike Hirschfeld, Jenja Kronenbitter, Anja Pfahler, Meike Mohneke & K. Eduard Linsenmair. Various funding agencies enabled the work. The main part of it was initially funded by the German Ministry for Education and Research (BMBF): BIOTA West Africa, 01 LC 0617J.
We are very grateful to the HPI Future SOC Lab for granting us access to their computing capacities!

References

[1] S.N. Stuart, , J.S. Chanson, N.A. Cox, B.E. Young, A.S.L. Rodrigues, D.L. Fischman, & R.W. Waller: Status and trends of amphibian declines and extinctions worldwide. *Science*, 306: 1783-1786, 2004.

[2] IUCN: *The IUCN Red List of threatened species*. Version 2012.2. http://www.iucnredlist.org. Last accessed 17th October 2012.

[3] S.N. Stuart, M. Hoffmann, J.S. Chanson, N.A. Cox, R.J. Berridge, P. Ramani & B.E. Young: *Threatened amphibians of the world*. Lynx Edicions, Barcelona, IUCN, Gland, Switzerland; and Conservation International, Arlington, TX, 2008.

[4] A.E. Magurran: *Measuring biological diversity*. Blackwell Publishers, Oxford, 2003.

[5] J. Penner, M. Wegmann, A. Hillers, M. Schmidt & M.-O. Rödel: A hotspot revisited – a biogeographical analysis of West African amphibians. *Diversity and Distributions*, 17: 1077-1088, 2011.

[6] S.J. Phillips, M. Dudík & R.E. Schapire: A maximum entropy approach to species distribution modeling. In: C. Brodley, Editor. *Proceedings of the Twenty-First International Conference on Machine Learning*. New York, ACM Press: 655-662, 2004

[7] S.J. Phillips, R.P. Anderson & R.E. Schapire: Maximum entropy modeling of species geographic distributions. *Ecological Modelling*, 190: 231-259, 2006.

[8] S.J. Phillips & M. Dudík: Modeling of species distributions with Maxent: new extensions and a comprehensive evaluation. *Ecography*, 31: 161-175, 2008.

[9] J. Penner, G.B. Adum, M.T. McElroy, T. Doherty-Bone, M. Hirschfeld, L. Sandberger, C. Weldon, A.A. Cunningham, T. Ohst, E. Wombwell, D.M. Portik, D. Reid, A. Hillers, C. Ofori-Boateng, W. Oduro, J. Plötner, A. Ohler, A.D. Leaché & M.-O. Rödel: West Africa - A safe haven for frogs? A sub-continental assessment of the Chytrid fungus (*Batrachochytrium dendrobatidis*). *PLoS ONE*, 8: e56236, 2013.

[10] J. Elith, S.J. Phillips, T. Hastie, M. Dudík, Y.E. Chee & C.J. Yates: A statistical explanation of MaxEnt for ecologists. *Diversity and Distributions*, 17: 43-57, 2011.

[11] M.W. McKnight, P.S. White, R.I. McDonald, J.F. Lamoreux, W. Sechrest, R.S. Ridgley & S.N. Stuart: Putting beta-diversity on the map: broad scale congruence and coincidence in the extremes. *PLoS Biology*, 5: e272, 2007.

[12] P. Jaccard: Nouvelles recherches sur la distributionflorale. *Bulletin de la Societe Vaudoise Sciénces Naturelles*, 44: 223-270.

[13] M.D. Mountford: An index of similarity and its application to classification problems. In: P.W. Murphy, editor. *Progress in soil zoology*. Butterworth, London: 43-50, 1962.

[14] D. Raup, & R.E. Crick: Measurement of faunal similarity in paleontology. *Journal of Paleontology*, 53: 1213-1227, 1979

[15] P. Legendre & L. Legendre: *Numerical Ecology, 2nd Edition*. Elsevier Science B.V., Amsterdam, 1998

[16] A.E. Magurran & B.J. McGill: *Biological Diversity – frontiers in measurement and assessment*. University Press, Oxford, 2011.

Maximum Resource Utilization Framework and the Performance vs. Productivity Tradeoff in Hybrid Parallel Computing

Fahad Khalid
Hasso-Plattner-Institut
University of Potsdam
14482 Potsdam, Germany
fahad.khalid@hpi.uni-potsdam.de

Andreas Polze
Hasso-Plattner-Institut
University of Potsdam
14482 Potsdam, Germany
andreas.polze@hpi.uni-potsdam.de

Abstract

In our previous reports, we have shown the necessity and effectiveness of the pipeline pattern for implementing combinatorial algorithms on hybrid parallel architectures. It was argued that the need for pipelining in such problems stems from the fact that the output size grows exponentially as a function of the input size. In this report we present further progress in our research in two major directions: 1) maximum resource utilization, and 2) using design patterns to balance the productivity vs. performance tradeoff.

Since the underlying assumption is the use of both CPU and GPU simultaneously for computation, maximum resource utilization implies that not only the GPU, but all threads on the CPU are also fully utilized for processing. We present an approach that processes the computational kernel simultaneously on the CPU and the GPU, keeping all resources occupied with productive work.

The above mentioned contributions have been applied to a problem from the domain of computational biology. Here, we present a generalization of the approach that makes it applicable to a broader set of problem classes. We argue that by identifying patterns in the algorithm, it is feasible to decide at design time which parts of the algorithm should be implemented on which architecture; i.e., CPU or GPU. We further argue that utilizing this knowledge can simplify the process of developing applications for hybrid parallel architectures.

1 Introduction

Hybrid computing architectures can be defined as those that in addition to the general purpose CPU processors, employ accelerators (or co–processors) for performance improvement. Employing accelerators can yield significant performance gains [1]; and consequently, hybrid architectures are now commonly used in HPC. According to the Top500 [2] list of the fastest supercomputers (published in June 2013), the top two positions are held by supercomputers based on hybrid architectures.

Graphical Processing Unit (GPU) is the most commonly used general purpose accelerator. Over the past few years, a multitude of computational kernels have been ported to GPUs with successful performance results. However, there are issues pertaining to developing codes for hybrid architectures. We focus on two such issues in this report.

First, much of the research is focused on porting and optimizing parallel algorithms for execution on GPUs. A general assumption in such an approach is that the entire kernel of interest will execute on the GPU only. In the subsections to follow, we show that if the kernel can be executed simultaneously on both the CPU and GPU, a significant performance improvement can be gained.

Second, GPU acceleration comes at a cost. GPUs are built on a massively parallel architecture with a large number of compute cores. However, as compared to the CPU, each GPU core supports a rather simplistic feature set. Moreover, GPU caches are much smaller than that of the CPU. Differences like these limit the types of kernels that can gain significant performance improvement by a straight forward port to the GPU. For a large number of computational problems, a straight forward port to the GPU does not result in substantial performance improvement.

Even though productivity tools for GPU programming have improved markedly over time, the range of features provided by these tools is far narrower than those available for programming CPUs. Much of the code optimizations are still left to the programmer, which increases both the development and maintenance effort. This leads to the fact that even though it is possible to accelerate computational kernels using GPUs, it results in a considerable reduction in productivity.

2 Maximum Resource Utilization

2.1 Concept

In this Section, we present our approach for the efficient and effective utilization of all processing resources available in a hybrid parallel architecture [3]. The method presented here is specific to embarrassingly parallel problems.

In a hybrid parallel environment, execution starts with the CPU, and is delegated to the GPU at certain points in the control flow. If the computational kernel in question were to be executed on the GPU only, the domain decomposition step would constitute distributing the input data over the available thread blocks. This is true for datasets small enough that fit in the GPU memory. For larger datasets, an additional step would partition data before the data is moved from the CPU to the GPU memory. Each partition would then be processed by a corresponding iteration of the hybrid pipeline [3].

In our maximum resource utilization approach, a geometric decomposition step is introduced before partitioning is performed for pipeline execution. At this stage, the input data is decomposed into two parts: one for execution on the CPU and the other for execution on the GPU. The decomposition is performed based on the relative capability of the two processing devices. E.g., if the GPU is considerably more powerful than the CPU, only a small part of the input will be processed by the CPU, while most of the data will be processed by the hybrid pipeline.

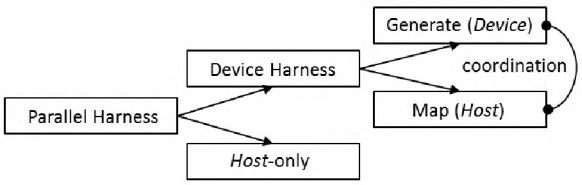


Figure 1: Framework for Maximum Resource Utilization [3].

As shown in Figure 1, the *Maximum Resource Utilization* framework spans the hybrid pipeline. The geometric decomposition is performed by the *Parallel Harness*. The *Parallel Harness* then forks two threads, i.e., the *Device Harness* and the *Host-only* code. The *Device Harness* orchestrates the hybrid pipeline by receiving as input the (generally larger) part of the input dataset. The *Host-only* module is an OpenMP parallel implementation of the computational kernel. This module can utilize all available hardware threads for processing the kernel on the CPU. The *Host-only* module and the *Device Harness* execute in parallel, thereby maximizing resource utilization across the hybrid parallel architecture.

2.2 Results

The hybrid pipeline approach was applied to a kernel from the domain of Computational Biology. This kernel is termed as *Combinatorial Candidate Generation*. The algorithm has already been successfully parallelized on both shared-memory [4] and distributed-memory [5] CPU based architectures. Here, a hybrid parallel implementation is presented that incorporates the *Maximum Resource Utilization* framework. Three instances of the kernel were implemented: Serial, CPU-only parallel and hybrid pipeline. Figure 2Performance comparison between the three implementation against four E.Coli datasets. presents the execution times of the three different implementations, against four different datasets. The datasets represent real metabolic networks from the bacterium *E.Coli*.

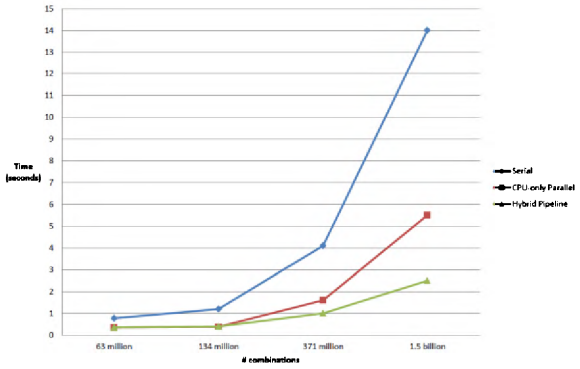


Figure 2: Performance comparison between the three implementation against four E.Coli datasets.

As can be seen in Figure 2, the CPU-only implementation outperforms the pipelined implementation on the first dataset, and the two implementations yield identical performance on the second dataset. This is due the fact that these datasets are very small, and the overhead incurred in pipelining over-shadows much of the performance gain. For larger datasets however, the *Maximum Resource Utilization* approach yields a significant performance gain, resulting in a *6-fold* speedup over the serial implementation, and *1.8-fold* speedup over the CPU-only parallel implementation.

Figure 2 provides us with four data points only. This is because the real biological datasets tend to suffer from combinatorial explosion beyond these four data points. In order to gain further statistically valid evidence of performance improvements gained by using the *Maximum Resource Utilization* framework, the code was executed against nine artificially created datasets. These datasets require the same computational procedure. However, they make it possible to control the problem size and avoid combinatorial explosion. The results from the artificially generated datasets are presented in Figure 3.

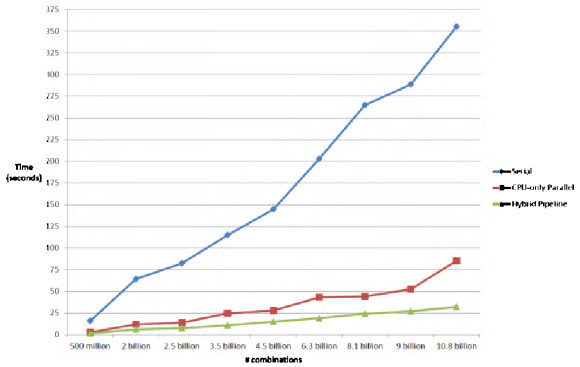


Figure 3: Performance comparison between the three implementations against artificially generated datasets.

3 Design Patterns for Performance vs. Productivity Tradeoff

This Section presents our analysis toward the generalization of the approaches we have developed during the course of our FutureSOC project. We emphasize the need for methods that make it possible to improve the *performance vs. productivity* tradeoff; i.e., we need methods that make it possible to leverage the performance potential of hybrid architectures while simplifying the development process.

3.1 Existing Solutions

A major improvement in the programmability of NVIDIA GPUs was made possible by the development of the CUDA [6] programming model. Later on, the OpenCL [7] standard was developed in order to provide CUDA like capabilities for vendor neutral accelerator programming. Over time, these advancements led to the following two major approaches for improving the effective use of hybrid architectures:

1. Accelerated Domain Specific Libraries
2. Compiler based code generation

In terms of effective utilization of the underlying hybrid architecture, each of the above mentioned solution approaches has associated drawbacks. Following is a brief summary:

The domain specific libraries are vital for scientific computing. However, these serve a very specific purpose and cannot be used as building blocks for problems in other domains. There is room for development of libraries that provide mechanisms for coordination between GPUs and CPUs, and simplify the process of implementing a much larger set of problems on hybrid architectures.

As for code generation, not all kinds of computations are suitable for GPU architectures. Therefore, simply automating the process of code generation does not guarantee a significant performance gain. For certain computational kernels, even optimized GPU code results in wasted clock cycles. Therefore, it is important that code generation is complemented with libraries that make it possible to write cross-device optimized code.

3.2 Design Patterns and Algorithm Decomposition

The design patterns introduced by the Gang of Four [8] are considered as building blocks for a robust object oriented design. These patterns provide abstractions from the domain, and make it possible for design ideas to be used irrespective of the domain. These however are not sufficient for parallel programming. In order to fill this gap, Design Patterns for Parallel Programming [9] were explicitly proposed, and have gained considerable popularity over the years.

The design patterns for parallel programming can be used to identify the data and control flow in a computational kernel. This makes it possible to see whether a certain kernel would be suitable for a given architecture.

3.3 Outlook

It is conjectured here that looking at the problem from the point of view of design patterns in parallel computing can lead to effective utilization of hybrid architectures. Not only does it highlight possibilities for performance improvement, it also improves programmer productivity.

For new codes, the patterns and associated constraints can be used to decide how the algorithm can be effectively distributed across the architecture. This fits into the existing design strategy for parallel programs. The method presented in this report is just an added step to an established process.

In the authors' research, for existing codes, an automated process will be introduced. A tool will be developed for the analysis of existing sources. This tool will provide recommendations on how the algorithm should be decomposed. An automated tool will require the representation of patterns and constraints in a formal language. Also, an efficient implementation of the hybrid pipeline model will require optimization of the control flow graph. Therefore, much mathematical formalism will be used in future work.

4 Conclusions

With this report, we declare the project *"Parallelization of Elementary Flux Mode enumeration for large–scale metabolic network"* successfully concluded. Due to the possibility of utilizing high-end machines at the FutureSOC Lab, this project has made novel contributions to the fields of *Parallel Computing* and *Computational Biology*. The project has culminated in an original publication [3].

The major contributions made by this project are:

- Development of the *Hybrid Pipelining* approach, with application to a real-world problem from the domain of computational biology.
- Development of the *Maximum Resource Utilization* framework, with application to a real-world problem from the domain of computational biology.
- Insights into generalization of the approach for applicability to the problem of managing the *Performance vs. Productivity* tradeoff in Hybrid Parallel Computing.

Moreover, this project is an example of a successful interdisciplinary collaboration between the *Hasso Plattner Institute for Software Systems Engineering* and the *Max Planck Institute of Molecular Plant Physiology*.

References

[1] Victor W. Lee, Changkyu Kim, Jatin Chhugani, Michael Deisher, Daehyun Kim, Anthony D. Nguyen, Nadathur Satish, Mikhail Smelyanskiy, Srinivas Chennupaty, Per Hammarlund, Ronak Singhal, and Pradeep Dubey. "Debunking the 100x gpu vs. cpu myth: an evaluation of throughput computing on cpu and gpu". In *Proceedings of the 37th annual international symposium on Computer architecture*, pages 451–460. ACM, 2010.

[2] Erich Strohmaier. "TOP500 - TOP500 supercomputer". In *SC*, page 18. ACM Press, 2006.

[3] Fahad Khalid, Zoran Nikoloski, Peter Tröger, and Andreas Polze. "Heterogeneous combi-natorial candidate generation". In *Felix Wolf, Bernd Mohr, and Dieter Mey, editors, Euro-Par 2013 Par-allel Processing, volume 8097 of Lecture Notes in Computer Science*, pages 751–762. Springer Berlin Heidelberg, 2013.

[4] M. Terzer and J. Stelling, "Accelerating the Computation of Elementary Modes Using Pattern Trees." In *Algorithms in Bioinformatics*. vol. 4175, P. Bücher and B. Moret, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 333-343.

[5] D. Jevremović, C. T. Trinh, F. Srienc, C. P. Sosa, and D. Boley, "Parallelization of Nullspace Algorithm for the computation of metabolic pathways". *Parallel Computing*, vol. 37, pp. 261-278, 2011.

[6] NVIDIA. *CUDA C programming guide. Design Guide*. PG-02829-001_v5.0, October 2012.

[7] Khronos OpenCL Working Group. *The OpenCL specification*. Standard specification, De-cember 2011.

[8] Erich Gamma, Richard Helm, Ralph John-son, and John Vlissides. *Design Patterns*. Addison-Wesley, 1995.

[9] Kurt Keutzer, Berna L. Massingill, Timothy G. Mattson, and Beverly A. Sanders. *A design pat-tern language for engineering (parallel) software: merging the PLPP and OPL projects*. 2010.

Batch method for efficient resource sharing in real-time multi-GPU systems

Uri Verner
Technion - Israel Institute of Technology
Haifa 32000, Israel
uriv@cs.technion.ac.il

Abstract

The performance of many GPU-based systems depends heavily on the effective bandwidth for transferring data between the processors. For real-time systems, the importance of data transfer rates may be even higher due to non-deterministic transfer times that limit the ability to satisfy response time requirements.

We analyze the bandwidth of two multi-GPU system as a step towards developing an execution method for data transfers and distributed computations between the host (CPUs) and multiple devices (GPUs).

Our experiments show that the bandwidth between a CPU and a GPU, and between GPUs, in a multi-GPU system is influenced by concurrent data transfers. Our results show that different multi-GPU systems can have different bandwidth distribution behavior.

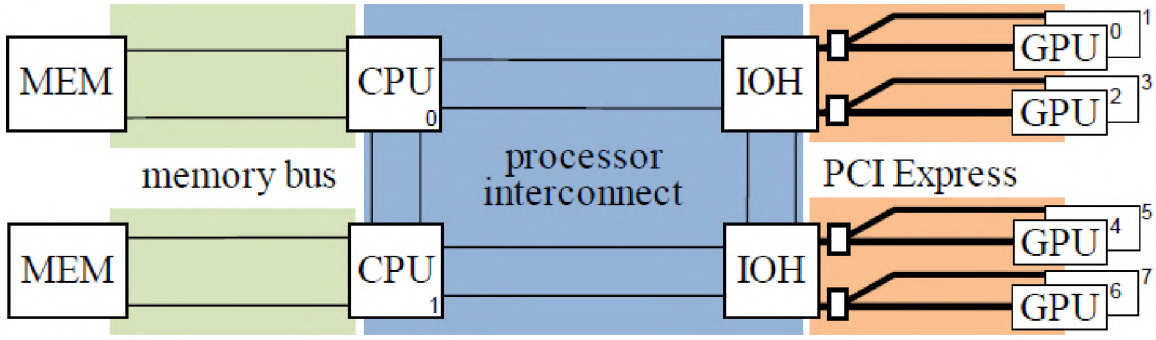


Figure 1: System architecture of a multi-GPU system

1 Introduction

In high throughput data processing systems, the computational load is distributed across multiple interconnected processors. To efficiently execute data-parallel parts of the computation, such systems often make use of discrete GPUs and other compute accelerators. The data is collaboratively processed by the CPUs and accelerators, and transferred between their local memories as required by the algorithm.

For real-time systems, where the worst-case execution time is important, the common methods for resource sharing include bandwidth allocation and time division. The bandwidth allocation method calls to divide the resource into a number of smaller portions, and to assign each of them to a task. For example, if the bandwidth of a bus is 8MB/s, it can be distributed among four tasks so that each task is assigned 2MB/s. The time division method calls to split the time domain into time-slots and to assign them to the tasks according to a given priority scheme. For example, the 8MB/s bus can be assigned for a period of time to task A and later to task B. Each of these methods allows the system to make the communication time deterministic at the expense of underutilizing resources such as the bus bandwidth, since a resource that is assigned to a task but not fully utilized is not used by the other tasks. Underutilization of the resources could increase power consumption or even prevent the system from meeting real-time deadlines.

In this project we analyze the bandwidth of two multi-GPU system as a step towards developing an execution method for data transfers and distributed computations between the host (CPUs) and multiple devices (GPUs).

Our experiments show that the bandwidth between a CPU and a GPU, and between GPUs, in a multi-GPU system is influenced by concurrent data transfers. Our results show that different multi-GPU systems can have different bandwidth distribution behavior.

2 System architecture

In this work, we will focus on a single-node multi-GPU system that includes one or more CPUs and a set of discrete GPUs, each of which has a local memory module and serves as a computational device. Figure 1 illustrates a possible architecture of such a system. The interconnect provides connectivity among the different components such as memories, processors, GPUs, etc., and consists of several communication domains, depicted in the figure by background blocks. Each domain consists of components that use the same architecture and protocol for communication. The domains are bridged by components that belong to several domains.

As the figure shows, the system consists of memory modules (MEM), CPUs, GPUs, and I/O Hubs (IOH). The I/O hubs bridge between the processor interconnect and PCI Express, and provide connectivity between PCI Express devices and the main memory modules; in some architectures, the I/O hubs are integrated into the CPUs. In addition to GPUs, other external devices may be connected via PCI Express; examples include compute accelerators such as Intel's Xeon Phi and high-throughput network cards. In this work, we focus on GPU-based systems, but the proposed method can be extended to be used with other devices as well.

Each CPU accesses its local memory directly via the memory bus, while distant CPUs and I/O Hubs (IOH) access it indirectly via the processor interconnect, which is a set of high bandwidth full-duplex point-to-point links. Examples of such interconnects include Intel QuickPath Interconnect (QPI) and HyperTransport.

Data transfers from and to GPU memory are executed by DMA controllers (a.k.a. DMA engines). GPUs that have two DMA controllers support bi-directional data transfer, while GPUs that have a single DMA can transfer data only in one direction at a time. GPUs from the same PCI Express domain can exchange data directly, while most systems do not support direct data transfer between GPUs in different domains due to chipset limitations. A common solution for transferring data between GPUs that reside in different domains, is to stage the data in CPU memory, using two DMA controllers.

3 Evaluation

To evaluate the batch method, we consider two applications running on two multi-GPU systems. We compare the batch method with two other bandwidth distribution methods: bandwidth allocation and time division.

For each system, we provide the system specification, find and analyze the effective bandwidth, and compare the performance using each of the methods in test case applications.

We describe two existing systems and usetwo existing techniques to analyze their effective bandwidth.

3.1 Nehalem multi-GPU system

This system is a Tyan FT72B7015 server featuring the following components:

1. Two 4-core Intel Xeon 5620 CPUs at 2.4GHz, based on the Nehalem micro-architecture
2. An Intel 5520/ICH10R chipset with a QPI processor interconnect at 4.8GT/s (9.6GB/s) and two I/O hubs, each with two PCIe 2.0 ports
3. A total of 24GB RAM in two modules
4. Four NVIDIA Tesla C2050 GPUs, each with 3GB of GDDR 5 memory

The system runs Ubuntu 10.04 x64 Linux with CUDA SDK 5.0.

Analyzing effective bandwidth

The results show that the bus bandwidth is asymmetric; the effective H2D PCIe bandwidth is between 25% and 50% higher than the D2H bandwidth. We also see that the effective bandwidth to the remote GPUs is 20% lower. Since the QPI bus has higher bandwidth than the PCIe bus, this indicates that the latency of the extra hop over QPI translates into throughput degradation. The H2D aggregate bandwidth scales up for two GPUs by 57% for local GPUs (saturates the QPI bus), and by 33% for remote GPUs. In contrast, the D2H bandwidth for two GPUs does not scale. For four GPUs, the H2D bandwidth does not scale further, but the aggregate D2H bandwidth lines up with the bandwidth of the local GPUs. We ascribe the reduced scaling to chipset limitations, except where the QPI bus was saturated. Since the GPUs only have one DMA engine, they are not able to get more bandwidth from bi-directional transfer, yet we see that for the local GPUs a bi-directional transfer is faster than an H2D transfer followed by a D2H.

3.2 Sandy Bridge multi-GPU system

This system features the following components:

3.2 Sandy Bridge multi-GPU system

This system features the following components:

1. Two 6-core Intel Xeon E5-2667 CPUs at 2.9GHz based on the Sandy Bridge micro-architecture
2. An Intel C602 chipset with QPI at 8GT/s (16GB/s) and two PCIe3.0 ports in each CPU 64GB of RAM in two modules
3. Two NVIDIA Tesla K10 cards, each with two GPU modules, connected by a PCIe switch, that include a GPU and 4GB of GDDR5 memory.

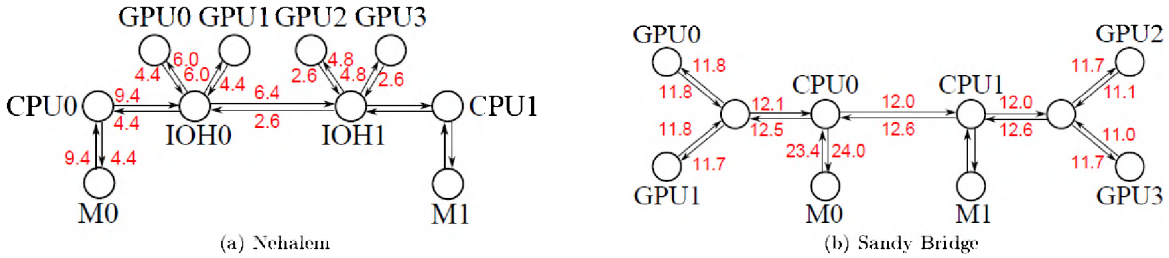


Figure 2: Topology graph showing effective bandwidth in GB/s

The system runs Red Hat Ent. 6.2 Linux with CUDA SDK 5.0.

Analyzing effective bandwidth

The Sandy Bridge system has higher bandwidth than the Nehalem system as it uses PCIe3.0. Moreover, the bandwidth in Sandy Bridge is symmetric and is similar for local and remote GPUs, unlike in the Nehalem system. The aggregate bandwidth to the GPU pairs is only slightly higher than for the individual GPUs; this is expected, as the GPUs share a PCIe bus. However, moving further to four GPUs, the bandwidth scales almost perfectly. For a single GPU, bi-directional transfers increase the bandwidth by 32%-72% over uni-directional transfers, while for all four GPUs, the increase in bandwidth is only 10%.

Since the GPUs only have one DMA engine, they are not able to get more bandwidth from bi-directional transfer, yet we see that for the local GPUs a bi-directional transfer is faster than an H2D transfer followed by a D2H.

4 Conclusions

We have shown that the throughput of a stream of data in a multi-GPU system depends on the bandwidth consumed by concurrent data transfers and that the bandwidth distribution is not trivial.

Using our measurements and analysis we develop a method that schedules data transfers in a predictable way by computing the effective bandwidth of each data transfer, taking concurrent traffic into account.

SAP HANA in a Hybrid Main Memory Environment

Ahmadshah Waizy
ahmadshah.waizy@ts.fujitsu.com
Ralf Liesegang
ralf.liesegang@ts.fujitsu.com
Konrad Büker
konrad.bueker@ts.fujitsu.com
Dieter Kasper
dieter.kasper@ts.fujitsu.com
Jürgen Schrage
juergen.schrage@ts.fujitsu.com

Fujitsu Technology Solutions GmbH
Heinz-Nixdorf-Ring 1
33106 Paderborn, Germany

Bernhard Höppner
bernhard.hoeppner@sap.com
Ole Lilienthal
ole.lilienthal@sap.com
Henning Schmitz
henning.schmitz@sap.com

SAP Innovation Center Potsdam
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany

Abstract

Hybrid main memory environments refer to the introduction of a further memory level to today's hardware architectures, which offers a fine-granular distinction of data and introduces new possibilities for applications. Storage Class Memory (SCM) as a feasible memory sublayer introduces a new class of memory technologies, which is well suited to increase the memory capabilities of individual servers efficiently by combining byte granular access, low latencies, high data density and non-volatility. We concentrate on potential applications of a hybrid main memory environment in the area of in-memory computing databases. We focus on evaluating SAP HANA concerning the usage of SCM for analytical data processing. As SCM prototypes are not yet available physically, a hybrid main memory environment is integrated into Linux by emulating the speed of data access as expected in future SCM technology. This second memory layer has been made available in SAP HANA and is used as the database component for storing relational data in-memory. A common benchmark comparing multiple SCM configurations was executed and the results are presented in this document.
This project was executed at the Future SOC Lab of the Hasso-Plattner-Institute in Potsdam, Germany, and utilized infrastructure available in the lab.

1 Introduction

The clear distinction of memory and storage in today's software architectures will potentially be blurred by the introduction of Storage Class Memory (SCM). SCM brings together the characteristics of available storage solutions like Hard Disks (HDD) or Solid-State Drives (SSD) with existing memory technologies such as Dynamic Random Access Memory (DRAM). This upcoming class of memory combines non-volatility and low power consumption, allowing low access latencies and still offers high data density. A classification of SCM latencies compared with existing storage and memory solutions can be found in Figure 1. SCM can be implemented in hardware by for example Phase-change Random Access Memory (PCRAM) and Resistive Random Access Memory (ReRAM) which are both not expected to be available before 2016. Bridge technologies overcoming this period in time will be available as of 2014 and will be sufficient to cover specific characteristics of SCM.

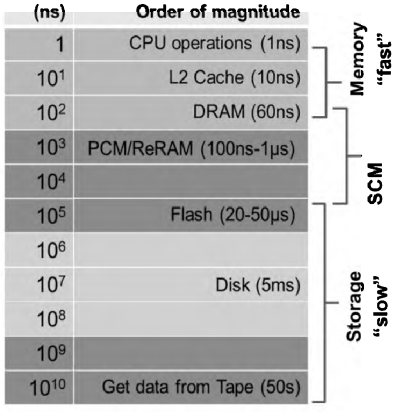


Figure 1: Latency characteristics of different types of data storage media

An additional but decisive characteristic of SCM is the accessibility at the level of byte granularity. In general it has to be distinguished between SCM-Storage (SCM-S) and SCM-Memory (SCM-M). SCM-S refers to an implementation of SCM only allowing access at block level making more CPU instructions for I/O operations and additional layers as file systems necessary. SCM-M offers the full advantages of SCM by also enabling load and store operations on the level of bytes. In a previous project we have analyzed the capabilities of SCM-S for the transactional logging of SAP HANA. Transactional logging can become a bottleneck for database systems as logs need to be persisted to disk latest when a transaction finishes. Our studies showed that logging is only causing a small overhead in SAP HANA, which led to the result that only very specific use cases can benefit from the introduction of SCM-S. In the area of Event-Stream-Processing a gain in the maximum number of inserts per second into a single table by using SCM-S instead of available SSD based solutions could be gained for SCM-S write latencies lower than 150ns per cache line. For SAP HANA based Enterprise Resource Planning (ERP) systems, such as SAP Business Suite on HANA, an advantage in comparison with available storage solutions could not be measured. We have also compared DRAM based RAM-Disk storage with SCM-S leading to the final result that the overhead of using block devices is limiting the performance of low latency offering devices in a way that an extensive gain in performance can not be expected. The results of this previous project and further insights into SCM technology can be found in [1].

This paper concentrates on the abilities a byte addressable SCM-M technology will be offering to SAP HANA. As in the previous project we rely on the emulation of SCM-M characteristic latencies as solutions are not available physically today. In this paper we will be focusing on single server systems and scale-up scenarios. The SCM characteristic of high data density will increase the maximum amount of memory in a single system while still offering fast access speed and will therefore be suitable to reduce the total cost of ownership (TCO) of a database management system (DBMS) infrastructure. The characteristic of non-volatility remains important as it reduces the power consumption of systems by making permanent memory refreshes, as known from DRAM, obsolete. A potential scale-up application for SCM-M in an in-memory computing DBMS might be the partitioning of data into hot, warm and cold stores, known as data aging. As SCM latencies will be slightly higher than DRAM latencies, SCM-M is especially suitable for warm and cold data. It is open whether SCM latencies will be sufficiently low to directly work on warm and cold data without constantly moving data back and forth between DRAM and SCM. The combination of DRAM and SCM in a single system is referred to as hybrid main memory environment.

This report summarizes the evaluation of "SAP HANA in a Hybrid Main Memory environment" and documents the results. Within this report we

- Provide an overview of the implementation of a hybrid main memory environment and describe the emulation and integration of SCM-M.
- Outline use cases of SCM-M for SAP HANA and described the integration of hybrid main memory by giving insights into the necessary adoptions.
- Summarize the results of quantifying the characteristics of SCM-M by using the TPC-H benchmark with different SCM-M latency configurations to compare DRAM and SCM-M based analytical database processing.
- Discuss the next steps to fully leverage the strengths of SCM-M in SAP HANA by outlining a fine-granular implementation for hybrid main memory.

2 Hybrid Main Memory

Main memory refers to physical memory that is internal to the computer. Main memory is distinguished by the time needed to access contained data and the total amount of memory available to the CPUs. *Access time* is defined as the total time needed from addressing data in main memory until the data contained under the wanted address can be read. The time for reading and writing data to and from main memory can be different.

Today operating systems only support homogeneous types of main memory (SDRAM). Therefore, the operating system LINUX for instance provides only one interface to programs for allocating virtual memory. The mapping of physical memory to the virtual memory is done by the first memory access exception (*page fault)*. The page fault handler tries to allocate the first free and available physical page for the faulty address. This algorithm does not support the allocation of different types (SDRAM or SCM) of physical memory depending on the (faulty) virtual memory. That means, today's memory allocation interface does not provide support for any special type of physical memory allocation. It is not possible to predict where the memory will be allocated.

This project, however, provides an additional interface to allocate two different types of main memory in a way that the application is able to choose which

data to store in what type (SDRAM or SCM) of physical memory area. – We call that mechanism *Hybrid Main Memory Support*.

From our *Hybrid Main Memory Support*, the application, SAP HANA, knows two different types of memory: the one that will contain the entire SAP HANA database and the other one that contains all other program data, e.g. text, bss, data and stack (Hybrid Main Memory). Therefore, SAP HANA will use two different memory allocation calls (malloc and scmalloc) to put the data into these different areas. Figure 2 shows the physical memory split into two parts that is used by SAP HANA.

The SCM emulating area of the physical memory (behaving like SCM) is used to store the database, and the remaining rest of the physical memory (behaving like standard SDRAM) is used for application and kernel memory, respectively.

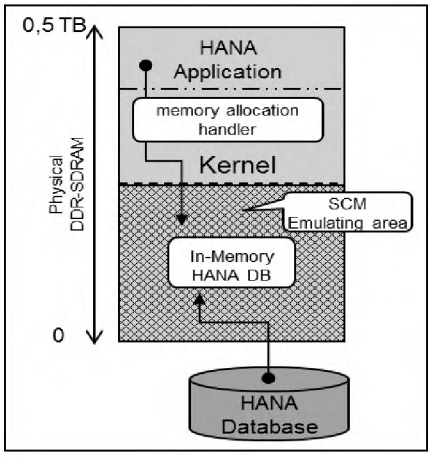


Figure 2: Split and usage of the Physical Main Memory

2.1 Configuration of a Hybrid Main Memory

The main memory is divided into two parts, the emulated SCM part and the standard SDRAM part (Hybrid Main Memory).

Figure 3 shows the hybrid main memory configuration that was used. The DIMMs attached to CPU 3 and CPU 4 are used for the SCM emulating area. The other part is for standard system use. CPU 3 and CPU 4 are switched off to ensure that no applications are assigned to these CPUs and memory access into the SCM emulating area is never local (fast access).

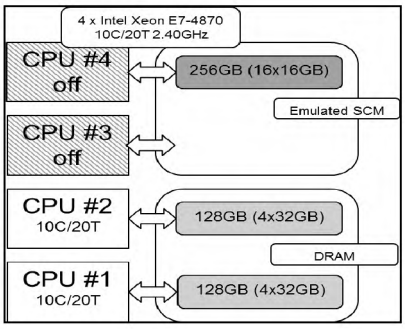


Figure 3: Physical Main Memory Configuration

2.2 Emulating SCM latencies

Latency refers to delays in transmitting data between the CPU and SDRAM. SDRAM latency is often measured in memory bus clock cycles. However, the CPU operates faster than the memory, so the CPU must wait while the proper segment of memory is located and read before the data can be sent back. This also adds to the total SDRAM latency.

During *power-on self-test* (POST) computers automatically configure the hardware that is currently present. *Serial presence detect* (SPD) is a memory hardware feature which lets the computer know what memory is present, and what timings to use to access the memory DIMMs specified.

The SPD data of the DIMMs attached to CPU 3 and CPU 4 are writeable and can be changed to return different latency values during POST (Figure 4).

We used this technique to emulate different SCM latencies.

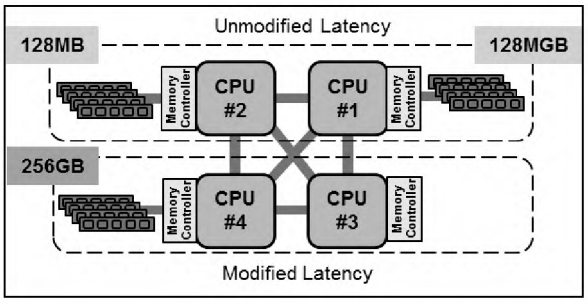


Figure 4: CPU/Memory Configuration with adjustable Latencies

The latency definitions are shown in Figure 5. Since the SCM latencies would be multiples of DRAM latencies, our SCM emulation changes the DRAM latency value (tRL) by different factors.

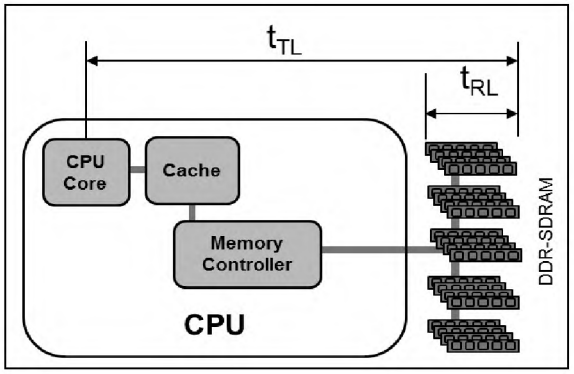


Figure 5: Memory Access Latencies

2.3 Enhancements to Virtual Memory Management to Support Hybrid Main Memory

Today's operating systems (OS) provide only one way to allocate virtual memory for the application: the *malloc* function. There is no way to instruct *malloc* to use particular pages of physical memory. The OS only knows the location of the physical memory (ccNUMA); besides this there is no differentiation between the physical memory pages. Figure 6 shows the current standard implementation of allocating memory pages for an application.

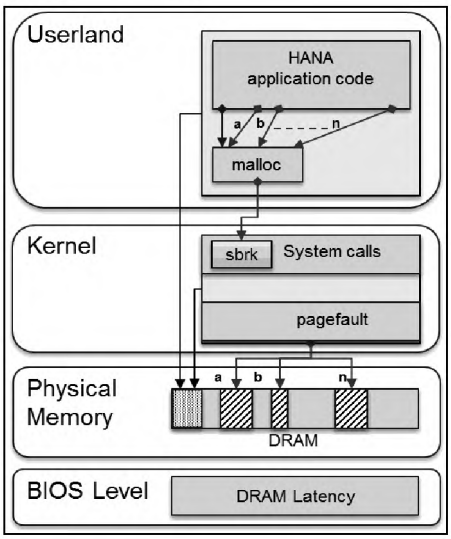


Figure 6: How memory allocation works today

We have implemented the following memory allocation functions to support hybrid main memory allocation for SAP HANA. By using these additional functions the application is enabled to allocate SCM emulating memory. The essential requirement of hybrid main memory management is to provide ways to dynamically allocate portions of desired memory (SDRAM or SCM) to programs at their request, and freeing it for reuse when no longer needed.

List of implemented additional functions:

- *void *scmalloc(size_t);*
- *int posix_scmemalign(void**,size_t, size_t);*
- *void scmfree(void *);*
- *void *scmrealloc(void *, size_t);*

The function *scmalloc* and the other provided functions implement the same, thread-safe API as *malloc* and similar functions.

Using these functions (*malloc* or *scmalloc*) an application is able to choose where the physical memory is to be allocated (SDRAM or emulated SCM).

Figure 7 shows the hybrid main memory allocation implementation. The physical pages of the SCM emulating memory are managed by a kernel driver (*rscmmap*). Memory requested by the function *scmalloc* will be allocated by *rscmmap* from the SCM emulating physical pages and will be mapped into the application so that the application addresses SCM emulating memory directly in a byte/word-like manner. The SCM area is reserved upon booting and is no longer available to the normal OS paging system. This means that the physical memory is allocated into different pools, where each pool represents pages of memory of a certain type, which are managed by the Linux paging or *rscmmap*.

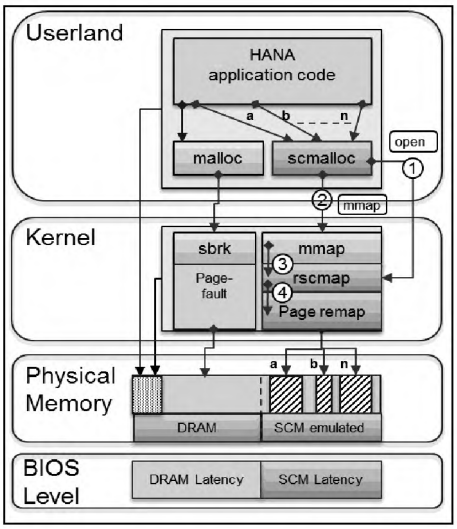


Figure 7: How the system is extended to support hybrid main memory

3 SAP HANA

Since 2010 SAP is offering its in-memory computing appliance SAP HANA, which introduces a technology that can be used to address problems in the domain of real-time processing of big data [2, 3]. Besides fundamental aspects like the organization of tables in columns, dictionary encoding and many more, the utilization of main memory as the only data store can be seen as the most significant decision in architectural design. Thus SAP HANA profits from shifting the old performance bottleneck of hard disks to the latency of RAM. Accordingly RAM becomes

the key factor when scaling the size of SAP HANA systems. Although the feasible amount of RAM in a single server has increased over the past years, it's still limiting compared to hard disks. As announced in May 2013 [4] HANA Hawk is today's largest prototype system offering 12TB of main memory in a single server. Even though this is a big improvement for a single server, databases for business systems like Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems can exceed this limitation. Customers facing larger amounts of data will be given the opportunity to scale-out their HANA systems. A shared-nothing approach offers the ability to independently spread data across multiple servers increasing the maximum amount of data as well as the computation power of the whole landscape. Nonetheless increasing the size of the data stores comes with a considerable amount of economic effort. Apart from that, new memory technologies such as SCM offer affordable alternatives introducing a higher data density compared to DRAM while still offering much higher performance than NAND Flash based solutions such as SSDs. Introducing these solutions to today's systems provides the opportunity to increase the maximum amount of memory in single server, thus offering new scale-up scenarios.

This paragraph concentrates on the capabilities of upcoming memory solutions like SCM, mainly focusing on low latencies and high data density.

3.1 Hybrid Main Memory for SAP HANA

Today's hardware architectures typically comprise three levels of caches, main memory and storage with all of them fulfilling specific tasks. As shown in Figure 1, upcoming hardware technologies addressed by the phrase SCM will change this architecte by introducing another level of memory. Blurring the differences between storage and memory will certainly involve the adoption of existing software architectures. As we concentrate on the latency and data density characteristics of SCM, it is more likely to describe SCM as a descendant of main memory and introduce the concept of so called hybrid main memory architectures. As described in paragraph 2, we use a main memory based emulation of SCM and introduce a second kind of memory based on this. For application developers SCM is accessible similar to main memory by using the appropriated commands *scmalloc, scmrealloc, posix_scmemalign and scmfree*. Introducing a second level of memory by preserving the availability of main memory makes it possible to distinct parts of the system which need to run on fast DRAM and others which can reside on slower but cheaper solutions.

For a database management system such as SAP HANA this distinctions leads to the question how to split data into partitions which is part of the scientific area of data aging. In general data aging partitions data into hot, warm and cold data where data is assigned to these groups by classification algorithms. Data classification can be achieved by statistical and manual approaches. Manual classification relies on application specific input while statistical approaches monitor the usage of attributes [5], [6]. Besides data aging hybrid main memory environments are applicable to any big data use case. A multi-level memory environment could be used as another caching architecture where SCM serves as big data store and traditional RAM as a subsequent layer. This approach involves additional effort copying data between the different memories and relies on the quality of the implemented caching algorithms.

In this paper we concentrate on the question whether it is desirable to directly operate on SCM instead of shifting data between memory layers. This analysis can answer the question of whether or not to directly work with cold or even warm data in SCM. Therefore the impact of slightly higher latencies on the performance of database operations is the key indicator of our study. The aspect of a hybrid architecture is utilized by only applying SCM to the Index Server of SAP HANA, which contains the actual data stores and engines for data processing. All other components of the database and the whole operating system will be using normal RAM.

3.2 Adoption of SAP HANA

First insights into the correlation between memory latencies and database performance can be gained by adopting SAP HANA in order to support the usage of the emulated SCM. HANA consists of the four main components Name Server, Index Server, Statistics Server, and Preprocessor Server. A basis library for memory management is shared between all components offering an efficient memory pool based management. "This pool of allocated memory is preallocated from the operating system over time, up to a predefined global allocation limit, and is then efficiently used as needed by the SAP HANA database code." [7] This pool allocation using normal main memory is executed for all HANA components except of the Index Server.

We have introduced another allocator in HANA allowing the allocation of memory in emulated SCM. As described in paragraph 2.3 memory is also allocated in chunks and then used as needed by SAP HANA. The allocation of memory is based on mapping the emulated SCM devices into the virtual address space of the appropriate application. This allocator is integrated into the basis library of SAP HANA and can be enabled for distinct components by changing environment variables. By specifying appropriate hook functions the behavior of the GNU C Library function *malloc, realloc* and *free* as well as the posix C function *posix_memalign* is redirected to the newly implement SCM allocator. Therefore any allocation of the appropriate database component will be using SCM instead of traditional RAM. We have activated the SCM allocator for the Index Server

only, in order to just influence the actual date stores. As data processing also takes place in the Index Server a general negative effect onto the whole data processing has to be faced. This implementation will be sufficient to give first insights into the correlation of memory latencies and database performance but will certainly give an incomplete view compared with a productive and fine-granular implementation. For future implementations a clear distinction of data in main memory and SCM will be necessary. This outlook is also discussed in the final paragraph of this paper. Today's implemented integration is shown in Figure 8, which provides a general overview of the adopted SAP HANA system.

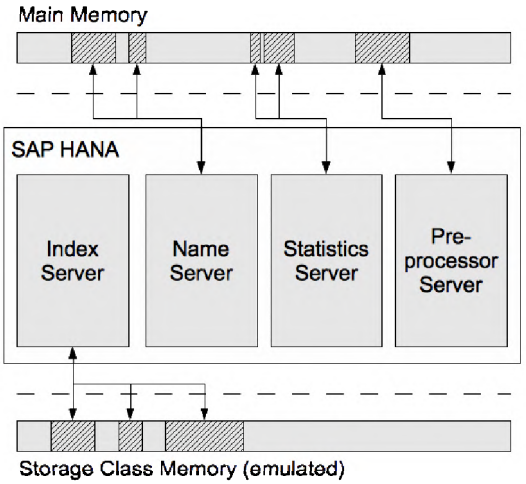


Figure 8: Hybrid Main Memory in SAP HANA (abstraction)

4 Benchmarking

Scale-up use cases such as data aging or caching rely on the characteristics and the impact of newly introduced memory technologies. Depending on the performance of SCM, different classifications for aged data or caching algorithms will be indicated. In the area of data aging it is particularly interesting whether computing can be done directly on the memory where the cold data is stored or not. The purpose of our benchmarks is to give insights into how the slightly lower latency of upcoming SCM will influence the computation power of SAP HANA. Besides using a synthetic memory test, which is measuring the latencies of different SCM configurations, we use TPC-H as a common database benchmark. TPC-H is especially suited for benchmarking as it offers a typical Online Analytical Processing (OLAP) workload. Cold or warm data is first and foremost used for analytical purpose, for example in an ERP system where data which is older than a certain threshold will potentially only be used as part of statistical analysis.

The following paragraph will be giving insights into the synthetic latencies of the emulated SCM. In the following we will give a short introduction into TPC-H and how the database benchmarks have been executed. Finally we present the results of the HANA based benchmarks and analyze the impact on the use case of data aging.

4.1 Synthetic Memory Test

The changes made to DIMMs' SPD data are measured with a special memory test program, which measures the overall system latency on average. This value is used to calculate the real DRAM latency which is the result of the modifications.

The measurement program checks the CPU cache sizes and ensures that the next read will point to an address which is guaranteed to reside outside the cache – avoiding cache hits.

Our test results are listed in Figure 9.

Average latency system: The time needed to fetch data from DIMM, via system bus, memory controllers, CPU cache and different HW buffers, into the application is shown in Figure 5.

Average latency real DRAM: The DIMM latency – changed by SPD data. The latency of unmodified DIMM [8] used in this project is 36ns.

	Test cases			
	0	1	2	3
Operation mode	Standard	emulated	emulated	emulated
Average latency system (t_{TL}) [ns]	188,5	234,7	250,2	265,6
Average latency real DRAM (t_{RL}) [ns]	36,0	82,2	97,7	113,1
Latency factor system	1,0	1,2	1,3	1,4
Latency factor real DRAM	1,0	2,3	2,7	3,1

Figure 9: Test Results

4.2 TPC-H Benchmark

The TPC (Transaction Processing Performance Council) offer a wide range of standardized database benchmarks covering different workloads. The area of Online Transactional Processing (OLTP) is covered by the TPC-C and TPC-E benchmarks. The TPC-H benchmark on the other hand offers a standardized OLAP workload, which is typical for a business analysis application. The specification of TPC-H can be found in [9]. As shown in Figure 10 the benchmark is designed for Decision Support Systems (DSS), which can be used for business analysis. By contrast to the philosophy of SAP HANA, different levels of database management systems are addressed by the TPC-H benchmark covering OLAP and OLTP separately. On the other hand the partitioning of data into hot and cold store as assumed by data aging fits this separation well. Cold data could be moved from the hot store to a different data store offering higher data density, such as SCM. In the assumptions of TPC-H this cold store is similar to the DSS database as data is moved into it from a different location and only analytical processing will be done on it later.

Therefore TPC-H is well suited to evaluate whether memory such as SCM will be suitable to directly work on cold data without copying it back to main memory.

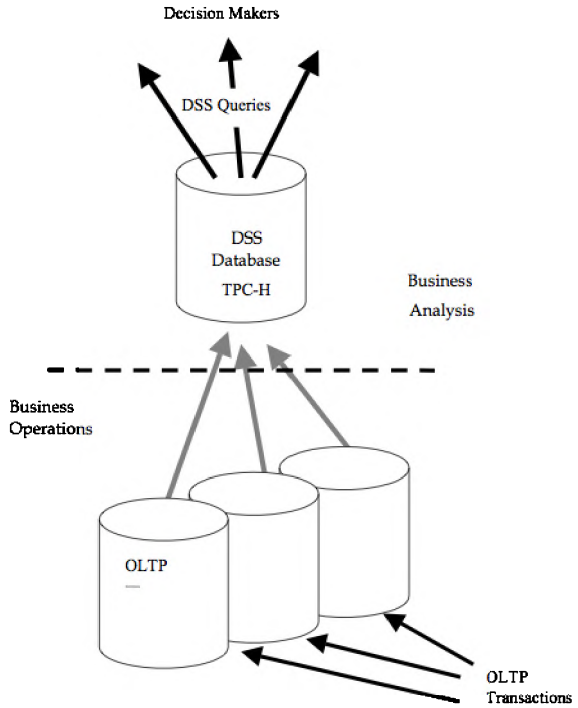


Figure 10: TPC-H Business Environment [9]

As most benchmarks TPC-H consists of two parts, dataset and workload. The schema of the dataset comprises eight tables with some tables having one-to-many relationships between each other. Each table has a predefined size which is depending on the scaling factor of the TPC-H benchmark. Using the available generator application the dataset can be created according to the TPC-H specification. Scaling factors out of a predefined set can be selected where scaling factors are about equal to the size of the dataset in gigabytes. According to the available configuration of the hybrid main memory environment and its implementation the highest scaling factor possible in our setup is 100GB. Besides the general dataset additional tables for so-called refresh queries have to be generated. The whole dataset is finally inserted into the database which is then prepared for the benchmark.

The workload of TPC-H, portraying the activity of a wholesale supplier, consists of 22 analytical queries Qi where 1<=i<=22 plus two refresh queries RF1 and RF2. A query set is a sequential execution of all 22 queries. A query stream equals the execution of a single query set by a single emulated user. There are two types of refresh queries where RF1 adds new sales information into the database and RF2 and removes old sales information. A pair of refresh functions consists of one execution of RF1 followed by one execution of RF2. A refresh stream is defined as the sequential execution of an integral number of pairs of refresh functions. A full TPC-H performance test consists of one power test followed by one throughput test where for our purposes it is sufficient to focus on the TPC-H Power. A power test equals the sequential execution of RF1 followed by all 22 Queries Qi followed by RF2. Therefore it measures the raw query execution power of the system using only a single user. The execution time in seconds for each query and refresh function is recorded. The TPC-H Power metric, which is the query per hour rate, is used to compare DBMS and is defined as

$$Power = \frac{3600 * SF}{\sqrt[24]{\prod_{i=1}^{i=22} QI(i,0) * \prod_{j=1}^{j=2} RI(j,0)}}$$

4.3 TPC-H using Hybrid Main Memory

The correlation of SCM latencies and the performance of SAP HANA will be the key indicator to evaluate the usage of SCM for cold data. The impact of latencies on database performance will be giving insights into the question if cold data can be directly accessed from a slightly slower but decisively larger storage. We have used the TPC-H Power metric to compare the number of queries per second for different latencies of the emulated SCM. Figure 11 is visualizing the results of the TPC-H Power benchmark in seconds for every refresh query and each of the three different SCM configurations one to three and the initial DRAM latency, which are shown in Figure 9. Besides the total runtime of each benchmark execution we also added the TPC-H Power metric which quantifies the query per hour rate. Additionally, factors comparing latency configuration one to three with the initial configuration zero are provided, which allows the comparability of the configurations.

Query	0	1	2	3
R1	7,30s	10,00s	11,30s	12,00s
1	13,90s	14,80s	14,90s	14,80s
2	6,90s	7,80s	8,50s	8,40s
3	21,60s	24,10s	25,90s	22,60s
4	80,20s	87,80s	94,20s	94,90s
5	44,70s	51,50s	57,00s	54,60s
6	1,30s	1,70s	1,90s	1,90s
7	6,90s	8,10s	9,00s	8,90s
8	41,80s	51,80s	61,10s	67,50s
9	21,40s	25,70s	29,50s	32,10s
10	31,60s	50,70s	42,80s	38,50s
11	1,70s	1,70s	1,90s	1,80s
12	23,10s	21,80s	24,40s	23,50s
13	40,60s	47,10s	52,50s	58,30s
14	1,70s	2,40s	3,10s	4,10s
15	6,70s	7,00s	7,20s	5,70s
16	38,60s	39,10s	40,90s	39,80s
17	31,10s	36,40s	40,10s	40,70s
18	105,40s	118,90s	129,60s	118,60s
19	0,90s	1,30s	1,70s	2,50s
20	26,30s	29,70s	33,30s	36,10s
21	56,40s	59,70s	63,50s	63,70s
22	11,50s	15,90s	17,00s	11,70s
R2	71,90s	78,50s	84,60s	84,10s
Total Runtime	693,50s	793,50s	856,00s	847,00s
Runtime Factor	1,000	1,144	1,234	1,221
TPC-H Power	23260	19831	18065	18061
Power Factor	1,000	0,853	0,777	0,776

Figure 11: TPC-H Power results in seconds for all 22 queries and four different configurations of emulated SCM

The benchmark results of the configurations zero to two show increasing runtimes and decreasing TPC-H Power results. Configuration three seems to be deviating from this behavior. We will be focusing on results one to two first and analyze the special configuration three separately later. The results in Figure 11 outline the correlation of memory latencies and SAP HANA performance. The coefficient of determination of TPC-H Power factors and real DRAM latency factors (compare Figure 9) shows this strong correlation of latencies and performance:

$$R^2 = 99{,}06\%$$

This result has been expected as increasing the memory latency has slowed down the whole data store of the in-memory database. On the other hand it is even more interesting to take a closer look onto the resulting regression line in Figure 12.

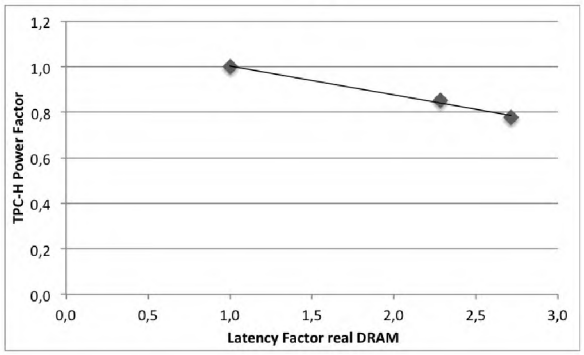


Figure 12: Latency Factor for DRAM (real) and TPC-H Power Factor including regression line

The regression line shows a gentle negative slope where the linear function looks as follows:

$$Y = 1{,}13 - 0{,}13 * X$$

Therefore the increase of memory latencies X is only decreasing the TPC-H Power Factor Y by 0,13. Based on this observation a slightly slower memory such as SCM will affect SAP HANA less critically as assumed. However, this statement has to be qualified in an environment with a fine-granular implementation, e. g. on tuple level, in which the resulting impact of latencies on performance will potentially decrease further. Nonetheless it has to be admitted that due to limitations in current hardware only three different latency configurations could be analyzed. Also the implementation of SCM is not optimized which has a negative impact on results compared to a productive SAP HANA environment.

The neglected results of configuration three might help to understand the results discussed before. Comparing the results of configuration two with the results of configuration three shows a decrease of total benchmark runtime even though the latency of the emulated SCM increased by 15,4ns. This latency configuration is achieved by setting the memory interleaving of the system from none to 2-way. N-way memory interleaving describes the allocation of memory to banks in turns where memory location *i* is allocated to bank number *i mod n*. The approach of interleaving has a positive effect on contiguous read and write operations to memory as each memory bank is accessed in turns. This results in an increase of memory throughput. On the other hand interleaving has a negative effect on memory latency as it can also be seen in Figure 9 when comparing configuration two to configuration three. As SAP HANA uses a cache aware memory organization, the memory access pattern can be compared to a contiguous operation, which is making memory throughput more important as latency. As memory configuration three only differs from memory configuration two with regards to the interleaving, memory latency increased on the one side but memory throughput did as well. Based on this the lower runtime for configuration three compared with configuration two becomes clear. On the other hand the TPC-H Power metric for configuration three is still lower than configuration two. This can be explained by taking a closer look on the TPC-H Power metric, which is using the geometric mean. When using the geometric mean instead of the arithmetic mean the metric is not dominated by more challenging queries, which are simply running longer because of the different workload. With this explanation it becomes clear that configuration three as it introduces a special case.

This analysis also gives an explanation to understand the gentle negative slope shown in Figure 12, which can be interpreted based on the cache aware memory organization of SAP HANA. As memory latency is only increased on the level of tRL (compare Figure 5), contiguous read and write operations will suffer less from introducing SCM as throughput does not decrease in the same way. Therefore working on SCM will affect the performance of cache aware applications less. This makes direct access to cold data in SCM even more viable. Furthermore a more fine-granular implementation of hybrid main memory in SAP HANA will improve the performance when working with SCM even more because only explicitly selected data will reside in SCM whereas the rest of SAP HANA's Index Server runs on DRAM.

5 Conclusion and Next Steps

In this paper we have taken a closer look at a hybrid main memory environment offering two different memory layers DRAM and SCM-M to SAP HANA. Our purpose was the evaluation of using SCM-M as a fast but TCO reducing data store for an in-memory computing DBMS. We focused on the questions how OLAP queries are affected by working with SCM-M and whether this upcoming technology will be suitable for directly working on aged data.

Our findings show that the higher latency of SCM compared to DRAM is strongly correlated to the OLAP performance of SAP HANA. On the other

hand, as memory throughput is not decreasing equally with increasing memory latency, the performance of SAP HANA is not decreasing equally to the increase of memory latency as well. These relations make SCM viable for directly storing cold data without copying it back and forth between DRAM and SCM.

The integration of hybrid main memory outlined in this paper still offers no fine-granular distinction of data in DRAM and SCM-M based data stores. In a future project we are planning to improve the implementation of the hybrid main memory integration by making two memory layers available in single SAP HANA Index Server. This will allow a clear partitioning of data into fast but rather small DRAM and slightly slower but cheaper and bigger SCM-M. This will make new scale-up scenarios possible.

References

[1] Ahmadshah Waizy, Dieter Kasper, Konrad Büker, Karsten Beins, Jürgen Schrage, Bernhard Höppner, Felix Salfner, Henning Schmitz, Joos-Hendrik Böse: Storage Class Memory Evaluation for SAP HANA. Fujitsu Technology Solutions and SAP AG. 2013.

[2] Hasso Platter, Alexander Zeier: In-Memory Data Management: An Inflection Point for Enterprise Applications. Springer. 2011.

[3] Jens Krüger, Changkyu Kim, Martin Grund, Nadathur Satish, David Schwalb, Jatin Chhugani, Hasso Plattner, Pradeep Dubey, Alexander Zeier. Fast Updates on Read-Optimized Databases Using Multi-Core CPUs. Proceedings of the VLDB Endowment, 5 (1):61–72. 2011.

[4] Hewlett-Packard Development Company, L.P.. http://www8.hp.com/us/en/hp-news/press-release.html?id=1411830#.UiSowhag06E. Accessed: 02/09/2013.

[5] Martin Grund, Jens Krueger, Juergen Mueller, Alexander Zeier, Hasso Plattner: Dynamic Partitioning for Enterprise Applications. Hasso Plattner Institute. 2011.

[6] Justin J. Levandoski, Per-Ake Larson, Radu Stoica: Identifying Hot and Cold Data in Main-Memory Databases. Microsoft Research. 2013.

[7] SAP AG. SAP BW on HANA Cookbook. https://cookbook.experiencesaphana.com/bw/operating-bw-on-hana/hana-database-administration/monitoring-landscape/memory-usage/. Accessed: 02/09/2013.

[8] Samsung. Semiconductor DRAM Specification. http://www.samsung.com/global/business/semiconductor/product/computing-dram/detail?productId=7426&iaId=692. Accessed: 12/09/2013.

[9] Transaction Processing Performance Council (TPC). TPC BENCHMARK H Revision 2.16.0. 2013.

Towards real-time IT service management systems: In-situ analysis of events and incidents using SAP HANA

Thorsten Pröhl
Technische Universität Berlin
Straße des 17. Juni 135
10623 Berlin
t.proehl@tu-berlin.de

Rüdiger Zarnekow
Technische Universität Berlin
Straße des 17. Juni 135
10623 Berlin
ruediger.zarnekow@tu-berlin.de

Abstract

IT service management (ITSM) is the paradigm shift from a traditional IT department to a customer and service orientation. Therefore, a mixture of IT, processes, and people is necessary to implement, manage, and deliver high quality IT services that address business needs. Furthermore the live analysis of big data becomes relevant for ITSM. Especially the event and incident management have to cope with mass data. This paper investigates the real-time identification possibilities of events and (major) incidents.

1 Introduction

Information technology service management (ITSM) is of increasing importance to information technology (IT) organizations around the world, while information systems (IS) play a crucial function in private and public sector organizations [1]. ITSM is the paradigm shift from a traditional IT department to a customer and service orientation. The need to align the IT strategy with the business strategy, increasing the transparency of IT processes, and quality while reducing the cost of IT services are some of the reasons why the information technology infrastructure library (ITIL), as the de-facto standard of IT service management, is introduced.

This research project will investigate the real-time identification possibilities of events and (major) incidents. Therefore, the authors build up a "live" system based on a big database and develop respectively use pattern matching techniques in order to treat the event and (major) incident issues. Finally, the in-situ monitoring of service level agreements (SLAs) will be performed. In order to realize live analysis of big data the in-memory technology is necessary.

In-memory computing is the storage of information in the main random access memory (RAM) instead of using complicated relational databases operating on comparatively slow disk drives. In-memory computing enables high-speed computing and enhances the performance of operations.

Big data and business analytics within the information systems are a growing research field [2]. This research field can be divided into five areas: Big data analysis, text analysis, network analysis, web analysis, and mobile analysis. Most research focuses on mobile analysis, e.g. mobile apps controlled by sensors can measure positions or activities in social networks [2]. Nonetheless, big data analysis and business analytics in ITSM have not been examined so far.

2 IT service management

The IT infrastructure library is the de-facto standard of IT service management. ITIL V3 was published in 2007, and in 2011, an update was introduced. The following section will give a brief introduction to this extensive update. The ITIL 2011 Edition contains no change in the basic concept of the ITIL service lifecycle; however, there are many improvements in terms of general consistence and clarity. For example, the stages are now consistently called phases and some new, extended, and updated processes are presented. Moreover, many concepts are improved and an alignment between different processes is achieved [3].

In the service strategy phase of the ITIL framework, the strategy for the provision of services for the business is developed. Here will be determined which types of services are allocated to which business unit [3].

During the service design phase services are developed according to defined requirements. Here, it is possible that the processes change existing services and as the case may be improve them [3].

The setup and roll-out of the services happens during service transition. This phase is responsible for im-

portant coordination tasks regarding the changes of services [3].

Through service operation an efficient and effective delivery of services is ensured. In this context, user requests are answered and problems get solved [3].

Continual service improvement (CSI) is used to improve the effectiveness and efficiency of services and processes with the support of quality management methods [3].

These phases and corresponding processes are shown in figure 1.

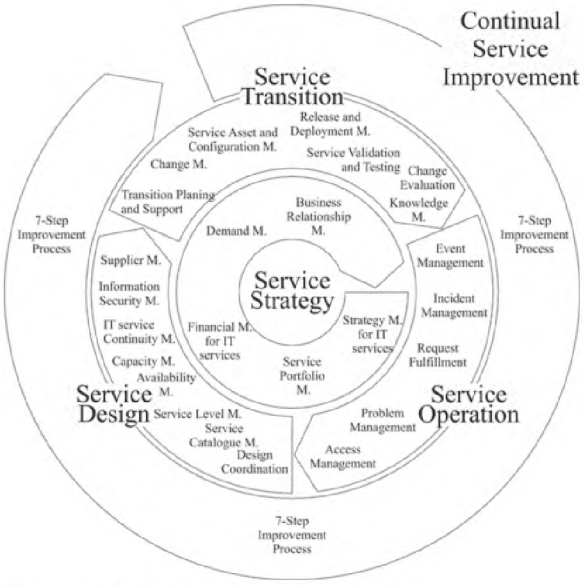


Figure 1: ITIL 2011 edition service lifecycle referring to [4]

Key topics of ITIL are event and incident management, which describe how to manage the whole event and incident lifecycle. According to ITIL an event is defined as "a change of state that has significance for the management of an IT service […]. The term is also used to mean an alert or notification created by any IT service, configuration item or monitoring tool. […] often lead to incidents being logged. In addition, IT organizations are dealing with a huge number of incidents in their daily business. ITIL describes incidents as "an unplanned interruption to an IT service or reduction in the quality of an IT service". Furthermore, major incidents are described as "the highest category of impact. A major incident results in significant disruption to the business." Thus, it appears that the identification and resolution of these major incidents is a matter of special importance.

3 Big data

In the field of business information systems, there are various conference and journal articles which examine big data regarding different aspects and thereby pick up on various specific topics. The topics can be divided into four groups: technical data provisioning, technical data evaluation, subject-specific data provisioning, and subject-specific data analysis. At the moment, the majority of articles deals with technical issues (NoSQL, in-memory, RCFile, Apache Hadoop) whereas subject-oriented data provisioning has virtually so far not been discussed.

The domain business intelligence and analytics (BI&A) of the business information system area can be divided into three evolution stages similar to Web 1.0-3.0 [5]. In the context of BI&A 1.0, DBMS-based topics, data warehousing, data mining, ETL, and OLAP have been treated. This was done primarily with structured data. By BI&A 2.0, we understand analyses and techniques that can be applied to unstructured data. Social media/network analyses, web analyses as well as information retrieval and extraction are some examples of BI&A 2.0. Since the end of 2011, the analyses and evaluation of data generated by mobile devices and sensors is summarized under BI&A 3.0. Real-time analysis based on the evaluation of local and context data represent as well as cover important issues besides the visualization of big data and reveal previously unknown cause-effect relationships.

On a technical level, the analysis of large location data is becoming increasingly efficient with the dissemination of new column-, document-, and graph-oriented databases as well as new paradigms (e.g. MapReduce). At the same time, these approaches are facing new challenges, for example in terms of time-series studies of location data or the pattern recognition within longitudinal analysis. Therefore, procedures and recommendations for action should be identified as part of this project.

4 Applying in-memory computing for ITSM processes

This research examines the potentials using in-memory computing for big data in incident and event management processes. Within ITSM, event and incident management have to handle with large data and therefore are predestined for applying in-memory computing to enhance the processes.

Event management (figure 2) monitors configuration items (CIs) and IT services. In addition, filtering of occurring events takes place.

Potentials for in-memory computing are within the event notification and detection process. Event notifications can be proprietary, and it is not unusual that only certain management tools can be used to detect these events. Most of the configuration items (CIs) generate event notifications. Once an event notification has been generated, it will be detected. The generation and detection of event notifications can be accelerated by keeping relevant data in the RAM available for short-term analysis.

Second, in-memory computing can enhance the correlation and filtering of first and second level events. Event correlation is a technique for making sense of a large number of events and detecting important events in big data. Live analysis and identification of correlations in real-time are enabled when data is stored in the RAM.

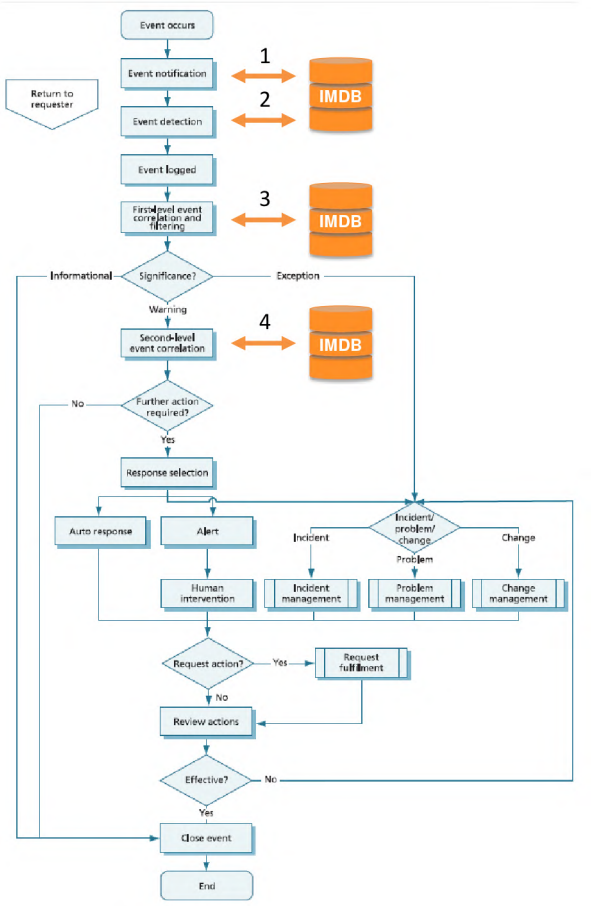


Figure 2: ITIL 2011 edition event management process [3]

In incident management (figure 3), the lifecycle of all incidents is managed. This process has the primary goal of restoring IT services to business units as quickly as possible. Important to understand, the source of all the incidents is an incident classification. In-memory computing can be related to the incident categorization. In order to gather incidents, fields for category and subcategory exist, which allow a classification. These categories can be used by the system to create automatic assignment rules or notifications. For instance, an important category for incidents is their incident state. Based on this state the service desk can track how much work has been done, and identify the next process steps. While the user enters categories, these can be compared and possible categories can be recommended in real-time.

The organization can be affected differently by the incidents. In order to handle the incidents a priority based on the impact and urgency is defined. Enterprises can make use of this in order to manage a lot of incidents and to minimize the incident impact on the business. In large companies, prioritization of incidents requires many computing resources due to the large number of requests. With in-memory computing, a dynamic prioritization can be realized. The assignment of the priorities is based on dynamic prioritization and live analysis of the data. In order to identify viewpoints underlying a text span a so called sentiment analysis can be used [6]. Given that users create incidents and comment on these, those comments could be analyzed via sentiment analysis. The real-time results could be used to help prioritize incidents. Another possibility to make use of the fast data processing of the in-memory database could be pattern-matching to identify major incidents.

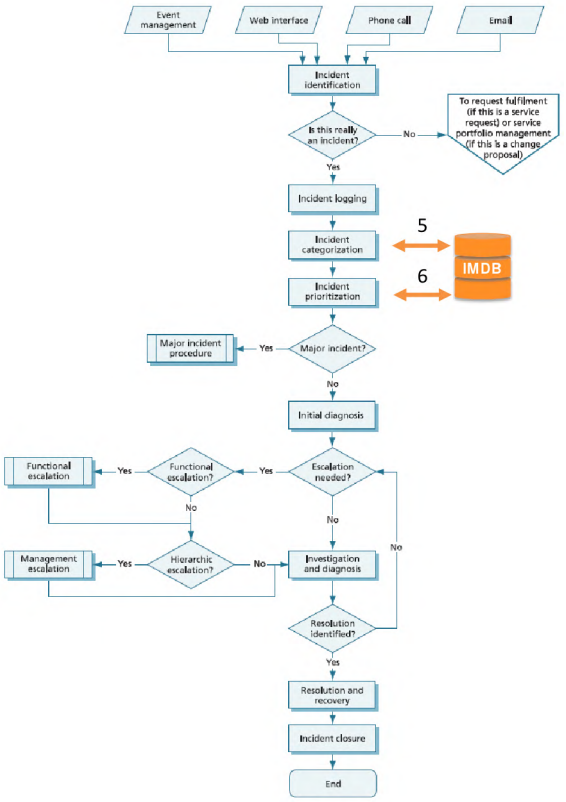


Figure 3: ITIL 2011 edition incident management process [3]

Figure 4 presents a high level view of the underlying architecture. Within the project scope, we implement monitoring nodes based on Nagios or Icinga in order to monitor and control IT resources like databases, web and ftp servers. Furthermore, these nodes observe virtual machine (VM) hosts and single VMs. It is necessary to distinguish between agent-based and agentless monitoring; our nodes support both modes of operation. These monitoring systems collect event and sensor data from different sources in order to transfer these via VPN to our HANA instance, where a continual load into corresponding database tables

happens. On the opposite side, a dashboard shows events, created incidents, and compliance of contracted SLAs. This proof of concept has different users, which have different needs for information and therefore various dashboard views. This architecture will be implemented in the future project step.

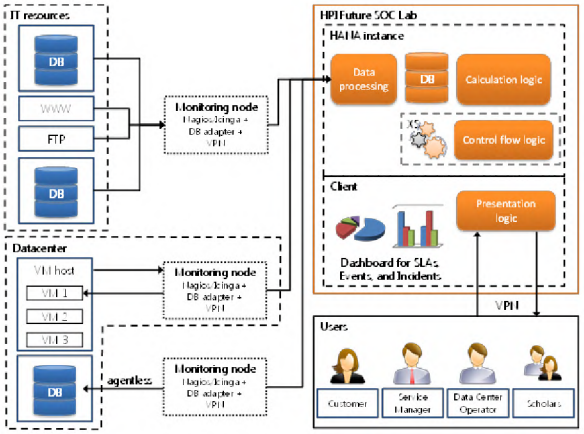


Figure 4. IT service management system: high level view of underlying architecture

5 Outlook

In the domain of ITSM, there are various fields of application for big data analyses and operation of in-memory technology. Real-time analysis of alerts, events, and incidents leads to proactive actions and processes. Incidents could be predicted, rapidly identified, and resolved in a timely manner. Moreover, serious issues, major incidents, could be recognized among a crowd of "normal" incidents. In order to recognize these kinds of incidents, text analysis techniques and dependency resolution of IT services can be applied. In contrast to the traditional resolution and visualization of configuration items (CIs), now the resolution of service dependencies is possible. The demand for this kind of analyses is increasing due to the complexity of services and data center structures. Cloud services and far-reaching value chains lead up to service dependency analyses. In addition, the ITIL IT security process ("information security management") is encouraged by user behavior and data analyses. Near-term determination of capacity and availability issues support capacity and availability management. On top of this, in-memory technologies offer new possibilities regarding the observance of service level agreements (SLAs). Usually, SLA reports are generated once a month, now, real-time SLAs are conceivable, whereas predictive SLAs will support service managers as they are able to start corrective actions in a timely manner. Dashboards visualize both kinds of SLAs in an appealing way, therefore service managers don't have to wait for recurring reports.

References

[1] Marrone, M. and L.M. Kolbe, "Impact of IT Service Management Frameworks on the IT Organization", Business & Information Systems Engineering, 3(1):5–18, 2011.

[2] Chen, H., Roger H. L. Chiang, R.H.L., Veda C. Storey. *Business Intelligence and Analytics: from Big Data to Big Impact*, MIS Quarterly Special Issue: Business Intelligence Research, 36(4):1165-1188, December 2012.

[3] Best Management Practice (BMP): ITIL Lifecycle Suite 2011: One single source for: Service Strategy, Service Design, Service Transition, Service Operation, and Continual Service Improvement, TSO, London, 2011.

[4] Buchsein, V. and Günther, M.: IT-Management mit ITIL®V3 - Strategien, Kennzahlen, Umsetzung; Vieweg+Teubner, 2008.

[5] Guo, S., Xiong, J., Wang, W., and Mastiff, R.L.: A MapReduce-based System for Time-Based Big Data Analytics. 2012 IEEE International Conference on Cluster Computing (CLUSTER), pages 72-80, September 2012.

[6] Pang, B. and Lee, L.: "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts". Proceedings of the Association for Computational Linguistics (ACL), pages 271–278, 2004.

Aktuelle Technische Berichte des Hasso-Plattner-Instituts

Band	ISBN	Titel	Autoren / Redaktion
87	978-3-86956-281-0	**Cloud Security Mechanisms**	Christian Neuhaus, Andreas Polze (Hrsg.)
86	978-3-86956-280-3	**Batch Regions**	Luise Pufahl, Andreas Meyer, Mathias Weske
85	978-3-86956-276-6	**HPI Future SOC Lab: Proceedings 2012**	Christoph Meinel, Andreas Polze, Gerhard Oswald, Rolf Strotmann, Ulrich Seibold, Bernhard Schulzki (Hrsg.)
84	978-3-86956-274-2	**Anbieter von Cloud Speicherdiensten im Überblick**	Christoph Meinel, Maxim Schnjakin, Tobias Metzke, Markus Freitag
83	978-3-86956-273-5	**Proceedings of the 7th Ph.D. Retreat of the HPI Research School on Service-oriented Systems Engineering**	Christoph Meinel, Hasso Plattner, Jürgen Döllner, Mathias Weske, Andreas Polze, Robert Hirschfeld, Felix Naumann, Holger Giese, Patrick Baudisch (Hrsg.)
82	978-3-86956-266-7	**Extending a Java Virtual Machine to Dynamic Object-oriented Languages**	Tobias Pape, Arian Treffer, Robert Hirschfeld
81	978-3-86956-265-0	**Babelsberg: Specifying and Solving Constraints on Object Behavior**	Tim Felgentreff, Alan Borning, Robert Hirschfeld
80	978-3-86956-264-3	**openHPI: The MOOC Offer at Hasso Plattner Institute**	Christoph Meinel, Christian Willems
79	978-3-86956-259-9	**openHPI: Das MOOC-Angebot des Hasso-Plattner-Instituts**	Christoph Meinel, Christian Willems
78	978-3-86956-258-2	**Repairing Event Logs Using Stochastic Process Models**	Andreas Rogge-Solti, Ronny S. Mans, Wil M. P. van der Aalst, Mathias Weske
77	978-3-86956-257-5	**Business Process Architectures with Multiplicities: Transformation and Correctness**	Rami-Habib Eid-Sabbagh, Marcin Hewelt, Mathias Weske
76	978-3-86956-256-8	**Proceedings of the 6th Ph.D. Retreat of the HPI Research School on Service-oriented Systems Engineering**	Hrsg. von den Professoren des HPI
75	978-3-86956-246-9	**Modeling and Verifying Dynamic Evolving Service-Oriented Architectures**	Holger Giese, Basil Becker
74	978-3-86956-245-2	**Modeling and Enacting Complex Data Dependencies in Business Processes**	Andreas Meyer, Luise Pufahl, Dirk Fahland, Mathias Weske
73	978-3-86956-241-4	**Enriching Raw Events to Enable Process Intelligence**	Nico Herzberg, Mathias Weske
72	978-3-86956-232-2	**Explorative Authoring of ActiveWeb Content in a Mobile Environment**	Conrad Calmez, Hubert Hesse, Benjamin Siegmund, Sebastian Stamm, Astrid Thomschke, Robert Hirschfeld, Dan Ingalls, Jens Lincke
71	978-3-86956-231-5	**Vereinfachung der Entwicklung von Geschäftsanwendungen durch Konsolidierung von Programmier-konzepten und -technologien**	Lenoi Berov, Johannes Henning, Toni Mattis, Patrick Rein, Robin Schreiber, Eric Seckler, Bastian Steinert, Robert Hirschfeld