QE3D: Interactive Visualization and Exploration of Complex, Distributed Query Plans

Daniel Scheibli Christian Dinse Alexander Boehm
SAP SE, Walldorf, Germany
firstname.lastname@sap.com

ABSTRACT

QE3D is a novel query plan visualization tool that aims at providing an intuitive and holistic view of distributed query plans executed by the SAP HANA database management system. In this demonstration, we show how its interactive, three-dimensional plan representation helps to understand and quickly identify hotspots in complex, real-world scenarios.

Categories and Subject Descriptors

H.2.m [**Database Management**]: Miscellaneous

Keywords

SQL; SQL query analysis; parallel database; visualization

1. INTRODUCTION

The task of analyzing and tuning the performance of database queries is very common for any database management system. Consequentially, all major DBMS offer corresponding facilities that assist users and administrators in this process. Typically, the SQL-based textual `EXPLAIN PLAN` functionality is complemented by graphical tools that visualize the logical and/or physical query execution plans as an operator tree [4, 5, 6, 11]. These tools allow to reason about important aspects of the execution plan, such as the access paths and physical operators being used.

In the context of the SAP HANA in-memory database management system [2], we frequently face the limitations of these state-of-the-art tools, basically for two reasons:

First, business users (e.g. using SAP's business suite) have an increasing demand of running complex, analytical queries on their transactional data to get real-time insights without previous data extraction, transformation, or pre-aggregation [7]. As data is usually distributed across a large number of tables, the corresponding analytical queries quickly span dozens or even hundreds of tables [1, 10]. As a consequence, the query plans become very complex and easily include thousands of operators.

A second source of complexity is distribution: As the main memory capacity of a single host system is rather limited (with a capacity of a few terabytes only) compared to the size of most Business Warehouse (BW) installations [9], the HANA database management system is frequently deployed on a cluster of machines. In such a setup, individual tables are deployed to corresponding nodes, and large tables (e.g. the fact tables in BW) are horizontally partitioned over some or even all the available nodes.

Consequentially, the state-of-the-art tools suffer from the high number of tables and operators involved. Moreover, the support for the specific aspects of distributed query execution such as temporal interleaving of individual host activity, communication patterns, data volumes transferred and landscape-global parameters such as network utilization is usually very limited.

There are several attempts to improve on the state of the art, e.g. by combining multiple different analysis views such as plan graphs and (textual) profiler output into a single tool and allowing users to dynamically navigate query plans and switch between these perspectives. An example for this approach is the Stethoscope [3] tool. Interesting contributions were also made by the Vertica Query Analyzer team [13] who combined multiple different views (e.g. pie charts for cardinality estimates, Gantt charts for temporal operator interleaving) in the context of a single analysis tool, and added key performance indicators (such as intra-operator parallelism) to the tree-based operator view itself.

Even with this multitude of visualization options and improvements in place, the process of performance analysis for complex, distributed queries involves looking at various different diagram types in parallel, each of them focusing on a different aspect of the distributed query plan (e.g. network communication, physical operator interleaving, host utilization, temporal operator interleaving). Specifically, there is no holistic view to the entire query plan, that combines these aspects into a comprehensible visualization that enables the user to get an overview of the general structure and performance characteristics of a query.

2. QE3D

The goal of *Query Execution 3D* (*QE3D*) is to overcome the limitations of the state of the art of distributed query performance analysis tools outlined above. Specifically, it aims at providing the user with a high-level, intuitive understanding of key performance aspects of complex queries

SIGMOD'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.

ACM 978-1-4503-2758-9/15/05 ...$15.00.
http://dx.doi.org/10.1145/2723372.2735364.

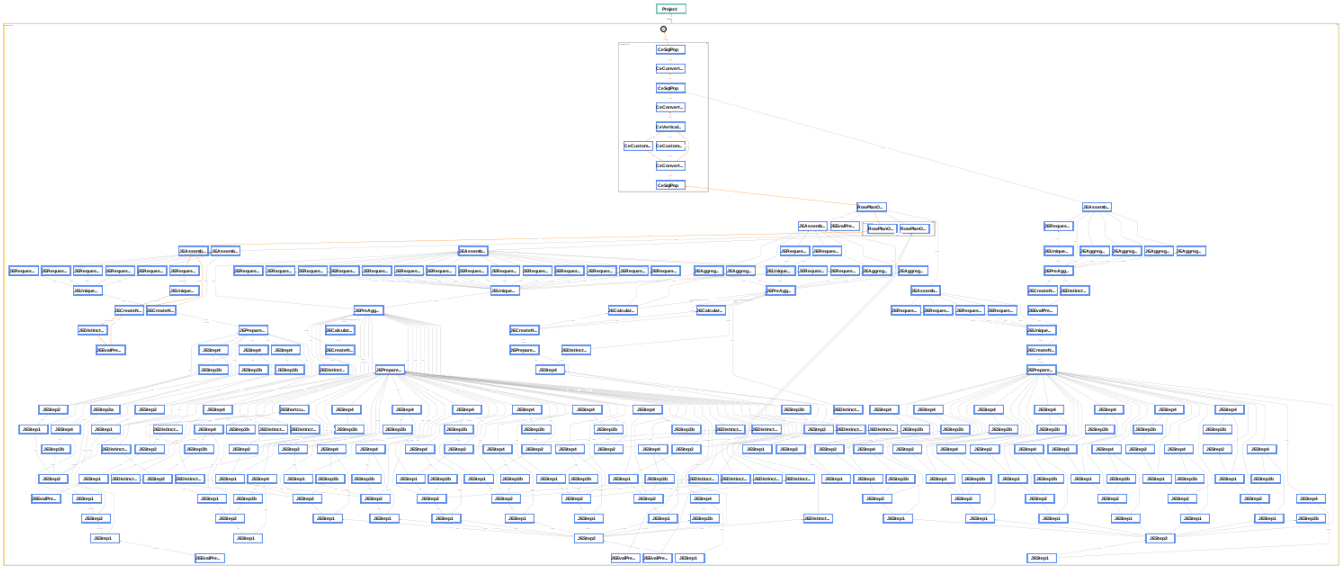

Figure 1: PlanViz visualizing a plan in graph view

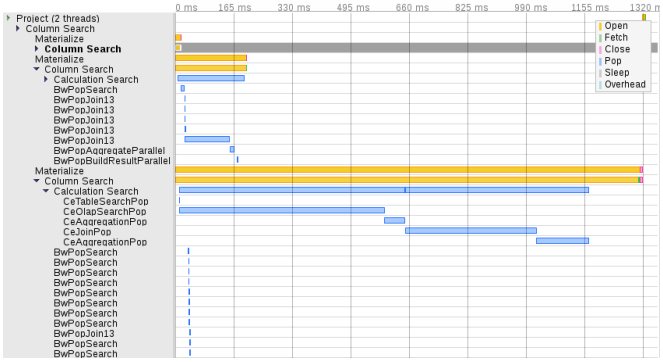

Figure 2: PlanViz visualizing a plan in timeline view

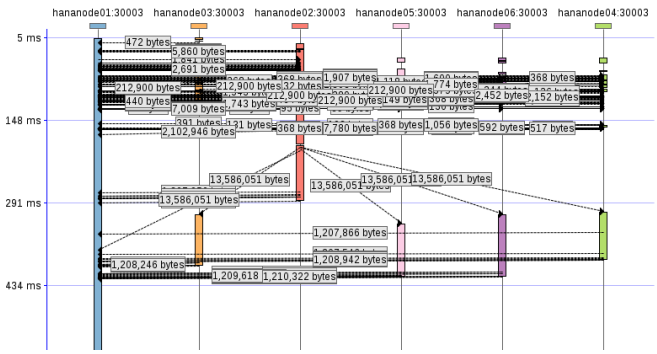

Figure 3: PlanViz visualizing the network traffic of a BW query running on 6 hosts

in order to identify specific starting points for a more detailed query analysis in the traditional tools.

QE3D is a Java application that is based on a stand-alone architecture. It is installed on a client machine and uses Processing [8] for implementing the interactive visualizations.

Users start by opening a XML encoded trace file that contains the collected details about the query execution. The data are parsed and the initial overview (see Figure 4) is shown. Now the user can navigate and explore the visualization using mouse and keyboard. Interactions include zooming, turning and rotating the visualization, accessing plan operator details or switching between different rendering alternatives.

Visualizing query plans, our central approach has been to come up with visualization approaches that *scale*. The graph like representations of plans (see Figure 1) are, as an example, very powerful in outlining data flows and the relation between operators. However displaying hundreds or even thousands of operators, as we see in complex queries, shows this approach to be less effective because graph structure and wealth of detail become overwhelming. By raising the level of abstraction and for instance visualizing the activity during query processing as a basic line plot, we gain a visualization that works equally well across the before mentioned range of operators.

QE3D is built on this principle. In the following we will discuss the different components and aspects that comprise the visualization as shown in Figure 4.

2.1 Visualizing Host Interaction

QE3D is designed for the interactive visualization of distributed query plans. We assume that there are a number of database nodes $N_1, \ldots, N_n$ that are spread across a number of hosts. During the execution of a distributed query, the database nodes interact with each other over the network in order to coordinate and to ship intermediate results. While there are typical communication patterns like the ones between the coordinating node and the other database nodes, it is possible that any node N_i is communicating with any other node N_j. In state-of-the-art tools, such as HANA's Plan Visualizer (PlanViz) [11], these communications are typically shown in Sequence Diagram like

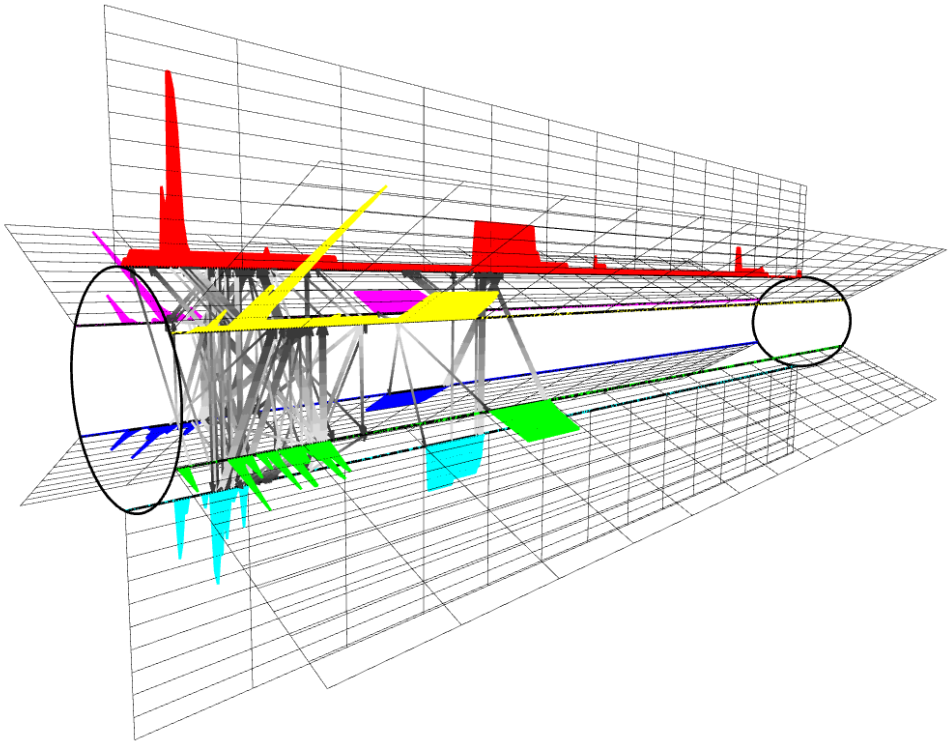

Figure 4: QE3D Visualizing a BW query running on 6 hosts

structures as shown in Figure 3. However this visualization is limited, because it is prone to overlapping information, e.g. when node N_1 transfers data to node N_3 and at the same time node N_2 is sending data to N_4. As a result transfers will intersect and otherwise recognized communication patterns might be missed.

To overcome this, we start with a *ring* on which all the involved database nodes are evenly placed in a circular manner. If there is an interaction between any two nodes, a connecting line segment is drawn between them. This visualization is known in the network monitoring domain, where tools like Etherman [12] provide similar representations. Unlike them, we know the time interval of interest, namely the total execution time of the query. As a result we extended the ring to a *cylinder* where the cylinder height represents time. We place the cylinder horizontally, such that the query starts on the left at time 0 and ends on the right hand side at time t_{total}. Now data transfers are drawn at the time when they take place. Here we provide three different modes of visualization. As an example, let us assume that node N_{source} is transferring data to N_{dest} and that the transfer starts at time t_{start} and ends at t_{end}. Let us further define that a point in the cylinder can be represented by the tuple (N, t).

- In the first mode, we draw an arrow from (N_{source}, t_{start}) to (N_{dest}, t_{start}).
- In the second mode, we draw an arrow from (N_{source}, t_{start}) to (N_{dest}, t_{end}).
- In the third mode, we draw a rectangular plane where (N_{source}, t_{start}) and (N_{dest}, t_{end}) define two opposing corners of the plane.

Given the duration of a transfer can typically be used as a proxy for the transferred data volume, we can use the third mode to easily spot where in the query large data transfers are taking place. The disadvantage however is that the planes might overlap and even hide each other. Therefore the second mode is a good compromise as it comes with less overlap, but still provides visual feedback on the duration of the data transfer. Here short duration transfers are near perpendicular to the time axis (similar to the first mode), while long duration transfers show a more acute angle.

To further improve readability, we indicate the direction of the transfer by adding a gradient to the line or rectangle. Furher, we ensure that the coordinating node, where the client connects and where the query starts and ends, is placed at the cylinders 12 o'clock position.

2.2 Visualizing Host Activity

In addition to visualizing the data traffic between database nodes, QE3D is also visualizing the activity on each node. This is done by providing a plot area per individual node and placing it next to the nodes location on the cylinder. In case of node N_i, the baseline of the plot area is defined by $(N_i, 0)$ and (N_i, t_{total}). Consequently the X axis of the plot area and the timeline of the cylinder are aligned. The Y axis of the plot area is facing outward and represents the KPI data.

Each of the plot areas only show activities for their respective database node. The default KPI shown is the number of active operators, which is defined as the number of operators that are running at a given time. Even though the number of active operators is a good initial indicator to understand host activity, the visualization is not limited to this. It is also possible to add one or multiple additional system level KPIs like the CPU utilization, memory consumption or similar.

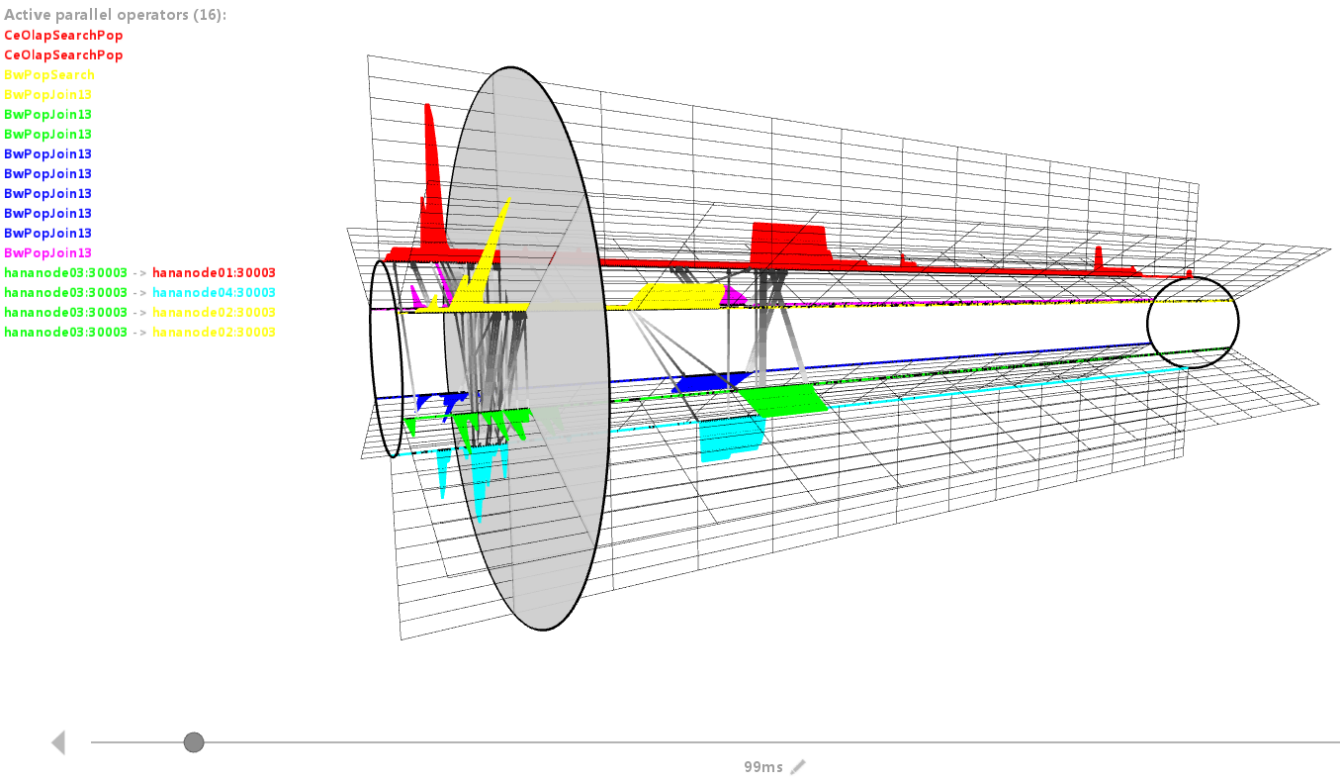

Figure 5: QE3D with selector

As an optimization, we segmented the query runtime into a configurable number of evenly spaced time intervals. For each time interval the KPI is calculated and the result is shown. Given the limits in screen resolution this is possible and it helps us to ensures similar frame rates for visualizing query plans of different complexities.

2.3 Detail Selection

Once an interesting aspect of the plan has been identified, it is important to learn more about that particular part. In some cases it is enough to further zoom into the plan, but at some point additional operator details are needed.

For this we introduced the *selector* which is shows as a disk that can be moved along the time axis (see Figure 5). Moving the selector is done by dragging a slider at the bottom of the screen. If dragging is done close to the slider at the bottom of the screen, the movement in time is fast. The further away the dragging is done, the slower the movement in time will be. With this interaction pattern it is possible to do effective millisecond level positioning inside a query that might run for minutes.

Once a specific point in time has been selected with the selector, the then active operators are listed with their names on the left hand side of the screen. To further improve the mapping of operators to database nodes, they share the same color with the nodes they are executed on.

Selecting a single operator is done by moving the mouse over it. This will open the operator details window shown in Figure 6 that contains information like start and stop times, resource consumption, payloads, the name of the processed database objects and other information.

3. DEMONSTRATION

Our demonstration is built on a number of complex, distributed database queries that we collected from both internal testing systems and production scenarios deployed by our customers running the HANA database management system. First, we quickly recap the state of the art of query analysis and visualization by showing the query structure in the existing visual explain facilities that HANA provides [11]. These include a tree-based plan graph (Figure 1) and Gantt-charts for temporal operator interleaving (Figure 2) and network communication (Figure 3). Already for a rather small cluster with only six active machines, the high complexity of the query plans make these visualization techniques impractical. As a result plan analysis is very tedious.

As a next step, we load the same query plans into the QE3D tool. We explain the various dimensions that are visualized by the tool (as discussed in Section 2 in detail) and show how the individual parts of complex plans (e.g. semi-join reduction and result assembly) translate into interaction patterns between the hosts that are part of the ring-based visualization. Next, we explain how both CPU and network bottlenecks can be easily identified using the various visualization modes of network communication, and the node-specific display of parallelism and workload characteristics.

Additionally, we demonstrate the replay functionality that animates the progression of the query plans in a given speed (with the original total execution time as default).

The last step of our demonstration is to introduce the selector feature discussed in Section 2.3. We show how it can be used to retrieve very fine-granular information from the

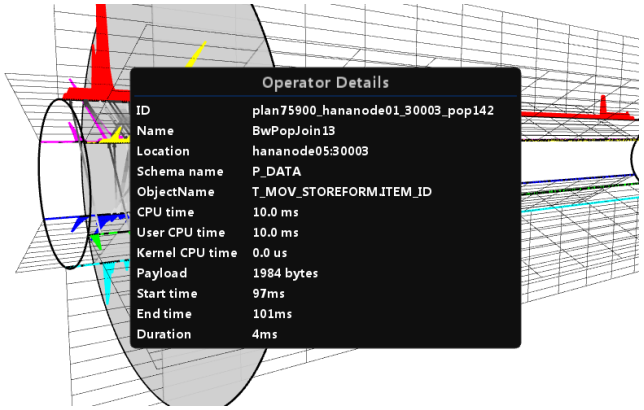

Figure 6: QE3D with selector and operator details overlay

complex query plans, such as operator-specific cardinality information or CPU consumption.

We conclude our demonstration by handing over the control of the notebooks running the tool to our audience: As Q3ED provides a very intuitive interface that allows mouse-based navigation, rotation and zooming within the plan visualization, the attendees can directly try the tool hands-on and use it for interactive analysis of complex query plans. Using this, to the best of our knowledge, never before shown visualization approach, we are hopeful, that attendees will be able to identify hotspots and bottlenecks in complex query plans, they have never seen before, in a matter of just a few seconds.

4. CONCLUSION

Our demonstration shows QE3D, a novel tool for the visualization and interactive exploration of complex, distributed query plans executed by the SAP HANA database management system. In contrast to state-of-the-art tools that are based on two-dimensional visualization such as operator trees and Gantt charts, QE3D provides a three-dimensional view of complex query plans in a distributed environment. QE3D allows users to get a quick and intuitive understanding of the query-specific aspects of distributed query execution (such as network communication, plan parallelism and operator interleaving) and to easily identify bottlenecks.

As the visualization and techniques demonstrated in this paper are not specific to SAP HANA, we are confident that similar techniques can also help to improve performance analysis and supportability for other database management systems that need to handle complex query workloads.

5. ACKNOWLEDGMENTS

We are grateful to our Business Warehouse and HANA Performance Analyzer colleagues for their input and feedback on the QE3D visualization.

6. REFERENCES

[1] N. Dieu, A. Dragusanu, F. Fabret, F. Llirbat, and E. Simon. 1,000 tables inside the from. *PVLDB*, 2(2):1450–1461, 2009.

[2] F. Färber, S. K. Cha, J. Primsch, C. Bornhövd, S. Sigg, and W. Lehner. SAP HANA database: data management for modern business applications. *SIGMOD Rec.*, 40(4):45–51, Jan. 2012.

[3] M. Gawade and M. L. Kersten. Stethoscope: A platform for interactive visual analysis of query execution plans. *PVLDB*, 5(12):1926–1929, 2012.

[4] IBM Corporation. DB2 Version 9 for Linux, UNIX, and Windows: Visual Explain overview. http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.admin.doc/doc/c0005135.htm. Retrieved November 16, 2014.

[5] Microsoft. Displaying Graphical Execution Plans (SQL Server Management Studio). http://msdn.microsoft.com/en-us/library/ms178071.aspx. Retrieved November 16, 2014.

[6] Oracle Corporation. MySQL Workbench Performance Tools: Visual Explain Plan. http://dev.mysql.com/doc/workbench/en//wb-performance-explain.html. Retrieved November 16, 2014.

[7] H. Plattner. A Common Database Approach for OLTP and OLAP Using an In-Memory Column Database. In *Proc. SIGMOD*, pages 1–2, 2009.

[8] C. Reas and B. Fry. *Processing: A Programming Handbook for Visual Designers and Artists*. The MIT Press, 2007.

[9] SAP SE. SAP EHP1 for SAP NetWeaver Business Warehouse 7.3, powered by SAP HANA. http://help.sap.com/nw731bwhana. Retrieved November 16, 2014.

[10] SAP SE. SAP Fiori for SAP Business Suite. http://help.sap.com/fiori. Retrieved November 16, 2014.

[11] SAP SE. SAP HANA Troubleshooting and Performance Analysis Guide: Analyzing SQL Execution with the Plan Visualizer. http://help.sap.com/hana/SAP_HANA_Troubleshooting_and_Performance_Analysis_Guide_en.pdf. Retrieved November 16, 2014.

[12] M. Schulze, G. Benko, and C. Farrell. Homebrew network monitoring: A prelude to network management. Technical report, Curtin University of Technology, Perth, West Australia, 1993.

[13] A. Simitsis, K. Wilkinson, J. Blais, and J. Walsh. VQA: vertica query analyzer. In C. E. Dyreson, F. Li, and M. T. Özsu, editors, *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 701–704. ACM, 2014.