DISS. ETH NO. 20272

Elastic, Reliable, and Robust Storage and Query Processing with Crescando/RB

A dissertation submitted to
ETH ZURICH

for the degree of
Doctor of Sciences

presented by
PHILIPP THOMAS UNTERBRUNNER
M.Eng., Cornell University, Ithaca NY, USA
born 16 April 1982
citizen of Austria

accepted on the recommendation of
Prof. Dr. Donald Kossmann (ETH Zurich), examiner
Prof. Dr. Gustavo Alonso (ETH Zurich), co-examiner
Prof. Dr. Peter Druschel (MPI-SWS Saarbrücken), co-examiner
Prof. Dr. Martin Kersten (CWI Amsterdam), co-examiner

2012

Abstract

Recent advances in Utility Computing have given businesses of all sizes the ability to acquire computing resources in the Cloud at the tip of a button. Users of services such as Amazon EC2 or Rackspace can elastically scale their computing resources with growing or shrinking demand. Yet the elasticity of computational resources is of little use if the software itself is not elastic; that is, if the software cannot use the additional resources, or would become unavailable during reconfiguration.

The problem is particularly pronounced at the "bottom" of the software stack, the data tier. At this tier, potentially large amounts of shared application state need to be maintained in a consistent and continuously available way. To make matters worse, workloads of successful web products not only grow in terms of data size and throughput, but also in diversity as new services and functionality are added in response to customer demand. Consequently, the data tier should not only be elastic in terms of data size and processing power, but also be flexible and robust enough to adapt to continuously evolving workloads.

Traditional solutions; i.e., relational database management systems (RDBMS), do not meet these new requirements. Attesting the fact, the recent years have seen a proliferation of specialized "NoSQL" data stores. What individual specimen of this large class of systems have in common is that they radically simplify the programming model in one or more dimensions (query language, data model, consistency model) in order to achieve a higher degree of scalability, availability, and/or performance for specific workloads.

This dissertation presents such a system with a unique set of features: *Crescando/RB*. Crescando/RB implements a push-based, distributed, elastic, and fault-tolerant relational table that is robust to mixed (read-write), unpredictable, evolving workloads. It scales linearly on modern multi-core hardware, scales elastically with the number of machines (to a certain degree), guarantees strong consistency for overlapping multi-key and range queries as well as updates, tolerates permanent failure of any machine in the system, and does not require stable storage (disks). The system is not a purely academic exercise, but is used to solve a large, real-life use case in the context of the Amadeus airline reservation system.

Crescando/RB consists of three components, each of which has been designed and implemented from scratch: Crescando, Rubberband, and E-Cast.

Crescando is a push-based, relational storage engine that uses continuous, parallel, and massively shared scans rather than data indexes. This radical design makes Crescando significantly more robust to unpredictable, evolving workloads, and allows the engine to scale linearly on modern multi-core hardware.

Rubberband is a distribution framework for building elastic, reliable, shared-nothing data stores with strong consistency guarantees. Crescando/RB uses the Rubberband framework to partition and replicate the contents of independent Crescando instances.

Rubberband in turn is intimately based on E-Cast, a dynamic, uniform, causal total order multicast protocol for asynchronous networks with unreliable failure detectors. By taking advantage of E-Cast, Rubberband can guarantee sequential consistency for arbitrarily overlapping multi-key and multi-range operations, and tolerate permanent failure of any machine at any time. Unlike most systems with strong consistency guarantees, Rubberband does not rely on stable storage or perfect failure detectors, so it is easy to deploy in the Cloud. Furthermore, Rubberband is completely wait-free, so data stores built with Rubberband can achieve very high throughput and also remain fully available during reconfiguration (data repartitioning or machine replacement).

Crescando/RB combines the features and advantages of Crescando, Rubberband, and E-Cast into a single, comprehensive system. The experimental results show that Crescando/RB is highly elastic, reliable, and available, and easily meets the stringent performance and data freshness requirements of the Amadeus use case.

In describing the design and implementation of Crescando/RB, this dissertation contributes to our scientific understanding of how to make storage systems robust to unpredictable, evolving workloads, how to take full advantage of the increasing number of CPU cores on modern hardware platforms, and how to elastically and reliably scale to a large, dynamic set of machines without the loss of data consistency or availability.

Zusammenfassung

Jüngste Fortschritte im Bereich des "Utility Computing" haben Unternehmen jeder Größe die Möglichkeit gegeben, auf Knopfdruck Rechenkapazitäten in der "Cloud" anzufordern. Nutzer von Diensten wie etwa Amazon EC2 oder Rackspace können somit ihre Rechenkapazität elastisch dem wachsenden oder schrumpfenden Bedarf anpassen. Jedoch ist diese Elastitzität von Rechenkapazität von geringem Nutzen, sofern die Software selbst nicht elastisch ist, also sofern die Software die zusätzlichen Ressourcen nicht nutzen kann, oder während einer laufenden Rekonfiguration längerfristig nicht verfügbar wäre.

Das Problem ist besonders eklatant auf der untersten Schicht der Softwarearchitektur, der Datenhaltungsschicht. Auf dieser Schicht müssen potenziell große Mengen an gemeinsam genutzten Anwendungsdaten in einer konsistenten und jederzeit verfügbaren Weise verwaltet werden. Erschwerend kommt hinzu, dass die Last erfolgreicher Anwendungen mit der Zeit nicht nur in in puncto Datenvolumen und Anfragendurchsatz steigt, sondern auch in puncto Diversität der Anfragen, da Dienste und Funktionalität auf Basis von Kundenwünschen hinzugefügt werden. Daher sollte die Datenhaltungsschicht nicht nur elastisch hinsichtlich des Datenvolumens und der Rechenkapazität sein, sondern auch flexibel und robust genug sein, um sich an eine fortwährend weiterentwickelnde Anfragenlast anzupassen.

Traditionelle Lösungen, das heißt relationale Datenbankverwaltungssysteme, sind nicht in der Lage diese neuen Anforderungen abzudecken. Diese Tatsache wird untermauert durch das Aufkommen zahlreicher spezialisierter "NoSQL" Systeme im Laufe der letzten Jahre. Die diversen Exemplare dieser breiten Klasse von Systemen eint, dass sie das Programmiermodell in einer oder mehreren Dimensionen (Abfragensprache, Datenmodell, Konsistenzmodell) radikal vereinfachen, um so einen höheren Grad an Skalierbarkeit, Verfügbarkeit oder Performanz hinsichtlich bestimmter Anfragen zu erreichen.

Diese Dissertation präsentiert ein solches System mit einer einzigartigen Kombination von Eigenschaften, *Crescando/RB*. Crescando/RB implementiert eine "push"-basierte, verteilte, elastische und fehlertolerante relationelle Tabelle, die robust gegenüber gemischten (Schreib- und Lesezugriff), unvorhersehbaren und veränderlichen Anfragenlasten

ist. Das System skaliert linear auf modernen Mehrkernprozessoren, skaliert elastisch mit der Anzahl der Maschinen (bis zu einem gewissen Grad), garantiert starke Konsistenz bezüglich Multischlüssel- und Bereichsaktualisierungsabfragen, toleriert bleibende Ausfälle beliebiger Maschinen im System, und benötigt keinen Permanentspeicher (Festplatten). Das System ist keine rein akademische Übung, sondern wird eingesetzt, um einen großen, realen Anwendungsfall im Kontext des Amadeus Flugreservierungssystems zu lösen.

Crescando/RB besteht aus drei Komponenten, die alle von Grund auf neu entwickelt wurden: Crescando, Rubberband und E-Cast.

Crescando ist eine "push"-basierte, relationale Datenverwaltungsengine, die kontinuierliche, parallele und massiv gemeinsam genutzte sequentielle Abfragen ("Scans") anstatt Datenindizes verwendet. Dieses radikale Design macht Crescando deutlich robuster gegenüber unvorhersehbaren, veränderlichen Anfragenlasten, und erlaubt Crescando linear auf Mehrkernprozessoren zu skalieren.

Rubberband ist ein verteiltes Softwareframework, welches die Erstellung elastischer, zuverlässiger, "shared-nothing" Datenverwaltungssysteme mit starken Konsistentgarantien ermöglicht. Crescando/RB nutzt das Rubberband Framework um Anwendungsdaten über mehrere unabhängige Instanzen von Crescando zu verteilen und zu replizieren.

Rubberband wiederum ist eng mit E-Cast verknüpft, einem dynamischen, uniformen, kausal-total geordnetem Multicastprotokoll für asynchrone Netzwerke mit unzuverlässigen Fehlerdetektoren. Durch die Verwendung von E-Cast kann Rubberband sequentielle Konsistenz für sich beliebig überlappende Multischlüssel- und Bereichsabfragen sowie Bereichsaktualisierungsabfragen garantieren, und bleibende Ausfälle beliebiger Maschinen zu jedem Zeitpunkt tolerieren. Im Gegensatz zu den meisten Systemen mit starken Konsistenzgarantieren verlässt sich Rubberband nicht auf Permanentspeicher oder perfekte Fehlerdetektoren, weshalb Rubberband sehr einfach in der "Cloud" eingesetzt werden kann. Des Weiteren ist Rubberband vollständig wartefrei. Dadurch können Datenverwaltungssysteme auf Basis von Rubberband sehr hohen Durchsatz erzielen, und sogar während Rekonfigurationen (Datenrepartitionierung oder Maschinenersatz) durchgehend verfügbar bleiben.

Crescando/RB kombiniert die Eigenschaften und Vorteile von Crescando, Rubberband und E-Cast in ein abgeschlossenes, funktionell vollständiges System. Die experimentellen Resultate zeigen, dass Crescando/RB höchst elastisch, verlässlich und verfügbar ist, und mit Leichtigkeit die strengen Performanz- und Datenaktualitätsanforderungen des Amadeus Anwendungsfalles erfüllt.

Anhand der Beschreibung des Designs und der Implementierung von Crescando/RB steuert diese Dissertation dem wissenschaftlichen Verständnis hinsichtlich der Fragen bei, wie ein

Datenverwaltungssystem robust gegenüber unvorhersehbaren, veränderlichen Anfragelasten gemacht werden kann, wie man die laufend zunehmende Anzahl an Prozessorkernen auf modernen Hardwareplattformen vollständig nutzen kann, und wie man ein System elastisch und zuverlässig, ohne den Verlust von Datenkonsistenz oder Verfügbarkeit, über eine grosse, dynamische Menge von Maschinen skaliert.