



Datenbanken: Self-Driving

Dr. Matthias Uflacker, Stefan Halfpap, Dr. Werner Sinzig

26. Juni 2019

- Einführung zu Unternehmensanwendungen (2 Vorlesungen)
- Einführung zu relationalen Datenbanken und Anfrageverarbeitung (2 Vorlesungen)
- Grundlagen des IT-gestützten Rechnungswesens und der Planung (3 Vorlesungen)
- Grundlagen von (spaltenorientierten) Hauptspeicherdatenbanken (5 Vorlesungen)
- **Trends in Hauptspeicherdatenbanken (4 Vorlesungen)**
- Klausur

- Basiert in großen Teilen auf der Arbeit von Jan Koßmann
<https://hpi.de/plattner/people/phd-students/jan-kossmann.html>

- Andy Pavlo
 - „Advanced Database Systems“ – Self-Driving Databases
<https://15721.courses.cs.cmu.edu/spring2019/slides/25-selfdriving.pdf>
 - Andy Pavlo „Make your Database Dream of Electric Sheep“ (Talk)
<http://www.cs.cmu.edu/~pavlo/slides/selfdriving-sfo2018.pdf>

- Datenbanken sind vielseitig einsetzbar
 - Nicht angepasst auf spezifische Anwendungsfälle
 - Aber optimale Performanz und Kosteneffizienz sind wünschenswert
- Datenbanken sind komplexe Systeme
 - Viele Komponenten mit verschiedenen Aufgaben, z.B. Queryoptimierung, Datenkompression
 - (Zunehmend) heterogene Hardware ermöglicht Optimierungen
 - Viele Konfigurations- und Optimierungsmöglichkeiten
 - “Every time database vendors don’t know how to do it, they introduce a new knob” Goetz Graefe
 - Physische Datenorganisation z.B. Indexauswahl

Optimale Konfigurationen hängen von Workload, Daten, Hardware, DBS-Version ab

Self-Driving Datenbanksysteme

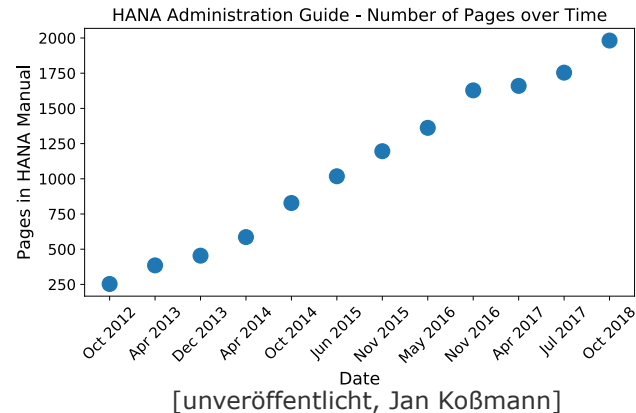
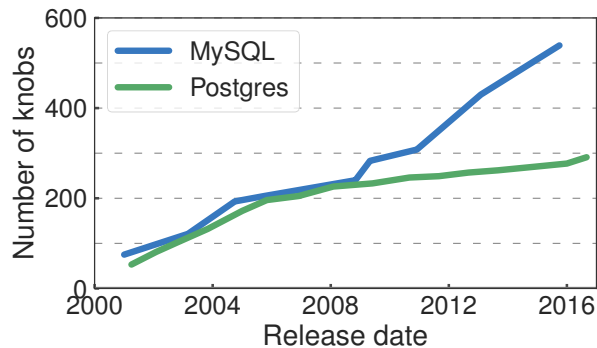
Motivation – DB-Administration: Status Quo

- Viele verschiedenartige Konfigurationsmöglichkeiten
 - Physische Ressourcen z.B. #Servers, #CPUs, Speichergrößen
 - Physische Datenorganisation z.B. Datenlayout, Hilfsstrukturen, Partitionierung, Replikation
 - Datenbanksettings (knobs) z.B. Threadanzahl, Puffergrößen, Mergezyklus
- Datenbankadministratoren konfigurieren und optimieren DB-Systeme oft manuell
 - Anspruchsvolle und zeitaufwendige Aufgabe
 - DB-Administration ist großer Teil der DB-TCO (Total Cost of Ownership)
 - Größe und Komplexität der DBS-Installation hat Menschliche Fähigkeiten überschritten

Self-Driving Datenbanksysteme

Motivation – Steigende Komplexität der DB-Administration

- Komplexe, schwankende Mixed Workloads
 - OLTP & OLAP auf der gleichen Datenbank
 - Saisonale (z.B. Quartalsabschluss, Mahnwesen) und unvorhersehbare (z.B. Hypes oder Krisen) Trends
- Steigende Anzahl der Einstellungsmöglichkeiten und DB-Features (z.B. neuer Indextyp)



Self-Driving Datenbanksysteme

Motivation – Cloudcomputing als zusätzlicher Treiber

- Übergang von On-Premise (vor Ort installiert) zu (On-Demand) Cloudcomputing
- Kosteneffizienz wichtige Eigenschaft fürs Cloud-Geschäft
 - Kunden können Kosten durch „Pay as you go“-Modelle sparen
 - Anbieter können Kosten durch Tuning der gehosteten Systeme sparen (aber begrenztes Wissen zu Workloads und Daten (privacy))

Selbstverwaltung und -optimierung des Datenbanksystems (ohne menschliche Interaktion)

- Datenbanksysteme haben Informationen über Workloads, Daten (inklusive Statistiken) und aktuelle Konfiguration
- Verwende Heuristiken, Machine Learning oder andere Optimierungstechniken um optimale/gute Datenbankkonfigurationen zu finden und anzuwenden

Self-Driving Datenbanksysteme

Bisherige Ansätze

- Forschung an automatischer Optimierung des Datenbankdesign begann in den 1970er Jahren
z.B. Hammer and Chan: „Index Selection in a Self-Adaptive Data Base Management System“ (1976)
- Fokus auf physische Datenorganisation
 - Indexauswahl
 - Partitionierung
 - Datenplatzierung
- Hier: Beispiele für Indexauswahl und Datenplatzierung

Self-Driving Datenbanksysteme

Bisherige Ansätze – Beispiel Indexauswahl

Chaudhuri and Narasayya: “An Efficient, Cost-Driven Index Selection Tool for Microsoft SQL Server” (1997)

<http://www.vldb.org/conf/1997/P146.PDF>

- Teil des AutoAdmin-Projekts: <https://www.microsoft.com/en-us/research/project/autoadmin/>
- Indexauswahlalgorithmus mit enger Kopplung an den Optimierer (, der später entscheidet, ob und welcher Index verwendet wird)
 - Daten- und anfragebasierter Ansatz
 - Indexkandidatenauswahl pro Query
 - Dann optimale Kombinationen (nicht skalierbar) und/oder Greedy Heuristik
 - Mehrattributindizes werden unterstützt
 - Verwendet „What-if-index-exists“-Aufrufe
 - Indexerstellung dauert und benötigt Speicherplatz
 - Idee: Schätze Kosten von Indizes – „was wären die Abfragekosten, wenn der Index existieren würde“

<https://github.com/HypoPG/hypopg>

Self-Driving Datenbanksysteme

Bisherige Ansätze – Beispiel Datenplatzierung

- Rabl and Jacobsen: "Query Centric Partitioning and Allocation for Partially Replicated Database Systems" (2017)
<https://www.redaktion.tu-berlin.de/fileadmin/fq131/Publikation/Papers/allocationCRC.pdf>
- Halfpap and Schlosser: "Workload-Driven Fragment Allocation for Partially Replicated Databases Using Linear Programming" (2019)
<https://www.redaktion.tu-berlin.de/fileadmin/fq131/Publikation/Papers/allocationCRC.pdf>

■ Datenplatzierung für partielle Replikation

- Daten- und anfragebasierter Ansatz
- Optimale Lösung (nicht skalierbar), Greedy Heuristik und Divide-and-Conquer Heuristik
- Optionen:
 - Robustheit gegen Ausfälle
 - Flexibilität gegen Workloadänderungen

Self-Driving Datenbanksysteme (Häufige) Grenzen bisheriger Ansätze

<https://15721.courses.cs.cmu.edu/spring2019/slides/25-selfdriving.pdf>

- Problem 1: **Benötigen menschliche Entscheidungen**

Mensch muss Zusatzinformation (z.B. Optimierungsziele) bereitstellen oder endgültige Entscheidung treffen, ob Empfehlungen angewendet werden

- Problem 2: **Reaktionäre Maßnahmen**

Maßnahmen beruhen (häufig) auf vergangenen Beobachtungen, antizipieren aber keine/selten zukünftige Trends/Probleme

- Problem 3: **Keine Integration**

DB-Komponenten und –Instanzen werden separat optimiert

Self-Driving Datenbanksysteme

Grenzen bisheriger Systeme – Automatisierungslevel basierend auf SAE J3016

<https://15721.courses.cs.cmu.edu/spring2019/slides/25-selfdriving.pdf>

- Level 1: **Manuell**: System macht nur, was Mensch einstellt; System gibt höchstens Warnungen
- Level 2: **Assistenz**: System gibt Vorschläge; der Mensch fragt und entscheidet
- Level 3: **Teilautomatisierung**: System und Mensch arbeiten gemeinsam am Systemmanagement; der Mensch leitet den Prozess
- Level 4: **Lokale Automatisierung/Optimierung**: Komponenten passen sich selbst an; aber keine Koordination/Integration
- Level 5: **Hochautomatisierung**: Mensch geben grobe Vorgaben/Hinweise; das System erkennt, wenn der Mensch eingreifen sollte
- Level 6: **Self-Driving/Vollautomatisierung**: Keine menschlichen Eingriffe

- Aufgaben:
 - Installation/Deployment
 - Konfiguration
 - Optimierung
- Ohne irgendeine menschliche Interaktion
 - Maßnahmen bestimmen
 - Zeitpunkt bestimmen
 - Von Maßnahmen für zukünftige Entscheidungen lernen
- Ganzheitliche Optimierung des Datenbanksystem (und nicht nur einzelne Komponenten)
- Proaktiv (und nicht nur reaktiv)

- Vorhersage von zukünftigen unbekanntem (probabilistischen) Workloads
- Komplexes Optimierungsproblem
 - Beeinflussung einzelner Optimierungsoptionen/-einstellungen → komplexes Modell
 - (Potenziell) Speicherintensiver, schwierig messbarer, ungenauer Modellinput
 - Verschiedene (teils mathematisch komplexe) Optimierungsansätze/-techniken
 - Verschiedene (teils widersprüchliche) Ziele (Kosten, Durchsatz, Latenz, Robustheit, Flexibilität für unbekannte Workloads (und Daten))
- Integration in die Datenbanksystemarchitektur

Self-Driving Datenbanksysteme (Eine Auswahl von aktuellen) Forschungsprojekte(n)

- Jan Koßmann: Self-Driving Hyrise

<https://hpi.de/plattner/people/phd-students/jan-kossmann.html>

- Carnegie Mellon University – Datenbankgruppe

- OtterTune – Tuning-as-a-Service für z.B. PostgreSQL, MySQL, ...

<https://db.cs.cmu.edu/papers/2017/p1009-van-aken.pdf>

- Self-Driving Database Peloton und Terrier

<http://www.cs.cmu.edu/~pavlo/papers/p42-pavlo-cidr17.pdf>

<https://github.com/cmu-db/terrier>

- Tim Kraska et al.: SageDB

<http://cidrdb.org/cidr2019/papers/p117-kraska-cidr19.pdf>

- Datenbankadministration ist zunehmend komplexe Aufgabe und wichtig für Cloudcomputing
- Idee von Self-Driving: Datenbankmanagementsystem installiert, konfiguriert und optimiert sich selbst und proaktiv ohne jegliche menschliche Interaktion
- „True autonomous DBMSs are achievable in the next decade.“

Andy Pavlo: Self-Driving Database Management Systems <https://15721.courses.cs.cmu.edu/spring2019/slides/25-selfdriving.pdf>