Geografische Analysen auf hadoop-Plattformen

Hans Viehmann
ORACLE Deutschland B.V. & Co. KG
NL Hamburg

Schlüsselworte
Hadoop, Oracle Big Data Spatial and Graph, Data Warehouse, Location Intelligence

Einleitung

Raumbezogene Daten sind ein wesentlicher Bestandteil zahlreicher Einsatzfeldern von Big Data Technologien. Egal, ob es um die Auswertung sozialer Medien auf Basis des Aufenthaltsorts, um die Analyse von beweglichen Objekten im „Internet of Things“ oder um den intelligenten Einsatz von Ressourcen in Smart Cities bzw. im Smart Grid der Energieversorger geht, immer müssen Standortdaten und deren Beziehungen untereinander berücksichtigt werden. Diese Positionsdaten kommen häufig in Form von geografischen Koordinaten vor, insbesondere, wenn sie von GPS-Sensoren geliefert werden. Teilweise verbergen sie sich aber auch implizit in Ortsnamen, Postleitzahlgebieten, oder anderen Standortbezeichnungen, wie etwa den Namen von Sehenswürdigkeiten. Daraus ergeben sich zwei wesentliche Anforderungen für Big Data-Plattformen, die mit raumbezogenen Daten umgehen können sollen. Einerseits müssen sie die für Geodaten erforderlichen Funktionalitäten – topologische Operatoren, mit denen die Lage von Objekten zueinander bestimmt wird, räumliche Indices wir R-Bäume, geometrische Funktionen, die die Länge eines Linienzugs bestimmen, usw. - unterstützen. Andererseits sollten sie in der Lage sein, Standortbezeichnungen in Textform zu interpretieren. Nachdem die Oracle Datenbank bereits seit über zwanzig Jahren die Verarbeitung von Geodaten unterstützt, gibt es seit Frühjahr 2015 mit Oracle Big Data Spatial and Graph ergänzend dazu ein Produkt im Oracle Portfolio, das auf dem Hadoop Framework aufsetzt und diese Anforderungen auf der Hadoop Plattform erfüllt. Diese Komponente soll im Folgenden kurz vorgestellt und anhand eines Beispiels illustriert werden.
Neben den Funktionalitäten zur Analyse von raumbezogenen Daten enthält Oracle Big Data Spatial and Graph auch eine Graph Database zur Verwaltung gewichteter Graphen („property graphs“) samt zugehöriger in-memory Analysefunktionen. Die Beschreibung dieser Komponente würde aber den Rahmen dieses Beitrags sprengen und wird daher in einem späteren Vortrag separat beschrieben.

Big Data Technologien

In den vergangenen Jahren sind neben konventionellen Data Warehouse Systemen zunehmend Plattformen zum Einsatz gekommen, die auf der Basis kostengünstiger Commodity Hardware Konzepte massiv paralleler Datenverarbeitung nutzen, um immer größere, variablere Datenmengen mit vertretbarem finanziellen Aufwand zu analysieren. Als aktuell vielversprechendster Ansatz hat sich dabei das Hadoop Framework herausgestellt. Bei Hadoop handelt es sich um ein Open Source Framework, das die verteilte Prozessierung auf Rechnerclustern mittels einfacher Programmiermodelle unterstützt. Es ist so aufgebaut, dass es von einem einzelnen Rechner bis zu Verbünden von tausenden Knoten skaliert. Jeder Knoten stellt in einer „shared nothing“ Architektur lokale Rechenkapazität und lokalen Speicher bereit, wobei letzterer auf einem eigenen Dateisystem, dem Hadoop File System (HDFS) basiert. Hohe Verfügbarkeit ist darin standardmäßig enthalten und wird über die Redundanz von Daten in der Anwendungsschicht sichergestellt. Diese Architektur kann sehr hohen Durchsatz erreichen, sie ist aber sicher nicht für alle Einsatzfälle mit raumbezogenen Daten optimal geeignet. Herkömmliche objekt-relationale Datenbanken, wie im Falle von Oracle Spatial and Graph, haben im Vergleich Stärken bei Transaktionalität, Tool-Unterstützung, Zugriffschutz, Manageability im Sinne von System Management, oder bei der Unterstützung von end-to-end

Prozessflüssen. Eine Gegenüberstellung technischer Entscheidungskriterien für diese beiden Arten von Technologien in Form eines Spider-Diagramms, aus dem die jeweiligen Stärken und Schwächen ersichtlich werden, zeigt Abb. 1.

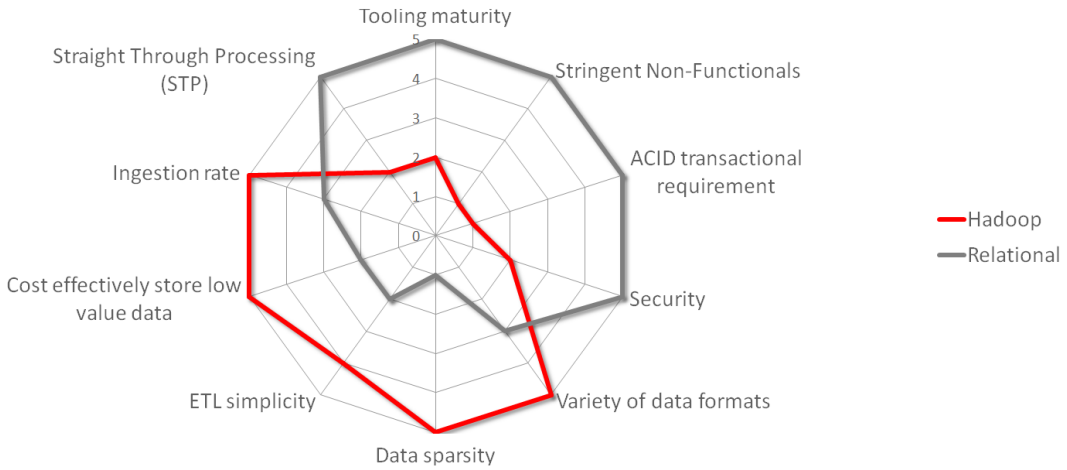

Abb. 1: Technische Entscheidungskriterien Hadoop Plattform vs. Datenbank

Datenverarbeitung mittels Hadoop

Wie bereits erwähnt, basiert die Verarbeitung auf der Hadoop Plattform darauf, dass die Daten auf viele Knoten verteilt sind. Ein Scheduler sorgt dafür, dass die Programmlogik passend auf die Knoten verteilt wird, wo sie unabhängig mit den jeweils lokalen Daten arbeitet, um den unnötigen Transfer von Daten zwischen Knoten zu vermeiden. Die Ergebnisse werden jeweils in Form von Key-Value Pairs an den nächsten Prozessschritt übergeben. Die eigentliche Abarbeitung der Logik erfolgt in wiederholenden Stufen, wobei Filterung und Projektionen (Map Phase), Umverteilung (Shuffle Phase) und Aggregation (Reduce Phase) sich abwechseln. Der Map- und Reduce Code wird in Java entwickelt oder aber generiert, wie beispielsweise auch im Falle des Oracle Data Integrator, der damit typische ETL Prozesse nativ auf Hadoop erzeugen, orchestrieren und koordinieren kann. Wesentlicher Unterschied zu herkömmlicher Verarbeitung ist, dass die Datenstrukturen nicht im voraus festgelegt werden müssen, sondern die Interpretation jeweils im Programmcode steckt. Man spricht in diesem Zusammenhang üblicherweise von “Schema-on-Read”. Entsprechend findet auch die Datenintegration in der Programmlogik statt, was beispielsweise auf die Nachvollziehbarkeit und die Datenqualität Auswirkungen hat.

Raumbezogene Daten auf Hadoop

Im einfachsten Fall lassen sich Geodaten mittels ein wenig linearer Algebra in MapReduce Programmierung verarbeiten, zumindest wenn es sich um Operationen auf einzelnen Geometrien handelt. So ließe sich etwa in Hadoop unter Ausnutzung der Parallelisierung jeweils der geometrische Schwerpunkt gegebener Flächen errechnen. Im Normalfall benötigt man aber komplexere topologische Operatoren, wie auch die Unterstützung unterschiedlicher Koordinatensysteme, oder geometrische Funktionen, wie etwa die Erzeugung eines Puffers um ein Polygon. Außerdem werden üblicherweise spezifische Dateiformate (etwa Shapefiles) und Notationen (GeoJSON) eingesetzt. Während MapReduce Algorithmen häufig darauf basieren, dass Datenbestände vollständig durchlaufen werden – ein Index also nicht erforderlich ist – profitieren bestimmte räumliche Operatoren doch von räumlichen Indices, wie sie etwa auch in der Datenbank vorkommen. Genau diese Funktionalitäten bietet Oracle Big Data Spatial and Graph, wobei mit diesem neuen Produkt nicht das Ziel verfolgt wird, die gesamte Verarbeitungslogik für Geodaten aus der Datenbank auf Hadoop zu portieren, sondern nur jene Teile bereitgestellt werden, mit denen typische Big Data Einsatzfälle unterstützt werden. Dies sind vor allem Funktionen, bei denen die massiv parallele

Prozessierung ihre Vorteile ausspielen kann und wird um Tools ergänzt, wie das Geo-Enrichment Framework, das speziell für Big Data use cases gedacht ist. Dieses Framework ermöglicht die Umsetzung von Ortsnamen und sonstigen Standortbezeichnungen in geografische Koordinaten, stellt also, technisch gesprochen, einen groben Geocoder bzw. Reverse Geocoder dar. Es ist vor allem zur Vorverarbeitung von standortbezogenen Daten in Textform, sowie zur Ergänzung mit räumlichen Hierarchien zur nachfolgenden Aggregation gedacht.
Um in der Lage zu sein, Datenbestände in HDFS in Kartenform zu visualisieren, ist zusätzlich der MapViewer Client, der bereits seit Jahren auch in der Fusion Middleware bzw. in Oracle BIEE enthalten ist, Bestandteil der Lizenz. Damit lassen sich auf Basis von HTML5 interaktiv Ausgangsdaten und Ergebnisse der Analyse auswerten.

Funktionalität zur räumlichen Analyse von Vektordaten
In seinem Java API unterstützt Oracle Big Data Spatial and Graph alle räumlichen Datentypen in Vektorform, die bereits für die Datenbank vorhanden waren. Dies umfasst neben den gängigen Datentypen, wie Punkte, Linien, Linienzüge samt Kreisbögen und Flächen bzw. zusammengesetzte Flächen in 2D auch alle Objekte in 3D, sowie Splines (NURBS), usw.. Dazu gibt es eine Bibliothek von Operatoren und Funktionen darauf, die etwa Objekte bearbeiten oder topologische Beziehungen ermitteln kann. Damit lässt sich beispielsweise feststellen, ob ein Punkt innerhalb einer Fläche liegt, ein Linienzug einen gegebenen Abstand zu einer Fläche (Buffer) schneidet, oder ob sich zwei Flächen in drei Dimensionen berühren. Dabei ist es unerheblich, ob die Koordinaten in einem kartesischen Koordinatensystem vorliegen, oder ob es sich um geografische Koordinaten handelt, weil Funktionen zur Transformation zwischen Koordinatensystemen enthalten sind.
Die Ausgangsdaten können in beliebigen Datenformaten in HDFS vorliegen; seitens des Frameworks gibt es hier keine Vorgabe. Es muss allerdings jeweils eine RecordReader Klasse bereitgestellt werden, die je Datensatz den Schlüsselwert und die raumbezogenen Daten extrahiert und an das API übergibt. Für gängige Formate wie Shapefiles oder GeoJSON wird dieser RecordReader bereits mit ausgeliefert. Die Ergebnisse der Auswertungen werden in Form von GeoJSON in HDFS abgelegt, wo sie weiter verarbeitet, oder vom MapViewer Client ausgelesen und visualisiert werden können.
Die grundlegende Aufteilung des Verarbeitungs-Frameworks ist in Abb. 2 dargestellt. Aufbauend auf dem Java API lassen sich eigene Map und Reduce Klassen entwickeln; man kann aber auch das bestehende Framework oder mit ausgelieferte Templates als Basis für die eigene Lösung verwenden.
Die Vorgehensweise ist einfach. Angenommen, man hat eine Datei mit Linienzügen in Form von GeoJSON und möchte je Geometrie einen Puffer gegebener Breite berechnen. In dem Falle muss nur die Ausgangsdatei in HDFS kopiert werden, in einem Map Prozess ausgelesen und jeweils der Buffer mit JGeometry.buffer() berechnet werden. Im Zuge des Reduce Prozesses werden die Ergebnisse als GeoJSON in HDFS zurückgeschrieben. Da GeoJSON direkt unterstützt wird, ist nicht einmal ein eigener RecordReader notwendig.
Handelt es sich bei den Ausgangsdaten nicht um geografische Daten in Koordinatenform, sondern um Ortsangaben in Textform, muss noch die Geocodierung vorgeschaltet werden. Hierzu kann man das mitgelieferte Framework direkt verwenden; eigene MapReduce Programmierung ist nicht erforderlich. Die Ortsangabe als String wird an einen Service (`MVSuggest`) übergeben, der im Hintergrund – seinerseits auf Basis von MapReduce – den passendsten Punkt im Referenzdatenbestand ermittelt. Standardmäßig wird hierzu der GeoNames Datensatz zugrunde gelegt (s. www.geonames.org), dieser kann aber ebensogut ersetzt, oder mit eigenen Daten angereichert werden. Neben den eigentlichen Koordinaten liefert `MVSuggest` auch die zugehörige Hierarchie administrativer Grenzen, womit sich in der Folge die Ergebnisse nach räumlichen Kriterien aggregieren lassen. Ebenso wie der eigentliche Referenzdatenbestand sind auch sie konfigurierbar. Daten aus dem Geonames Referenzdatenbestand sind anhand eines Beispiels in Abb. 3 dargestellt.

Abb. 3: Inhalte aus dem Geonames.org Referenzdatenbestand

Man erkennt zu der Ortsbezeichnung "Nürnberger Burg" neben den zugehörigen geografischen Koordinaten auch alternative Namen ("Kaiserburg", "Nuremberg Castle"), die gefunden würden, sowie die Hierarchie von Gemeinde – Kreis – Regierungsbezirk – Bundesland – Land. Die grafische Aufbereitung hier stammt im übrigen von der GeoNames Website und ist nicht Bestandteil des Produkts.
Zur Anschauung wird neben Beispielcode auch eine vollständige Beispielanwendung mit Oracle Big Data Spatial and Graph ausgeliefert. Sie umfasst eine Konsole, über die MapReduce Jobs gestartet, räumliche Indizierung ausgeführt, oder Ergebnisse in Kartenform dargestellt und interaktiv analysiert werden können. In Abb. 4 ist ein Screenshot dargestellt.

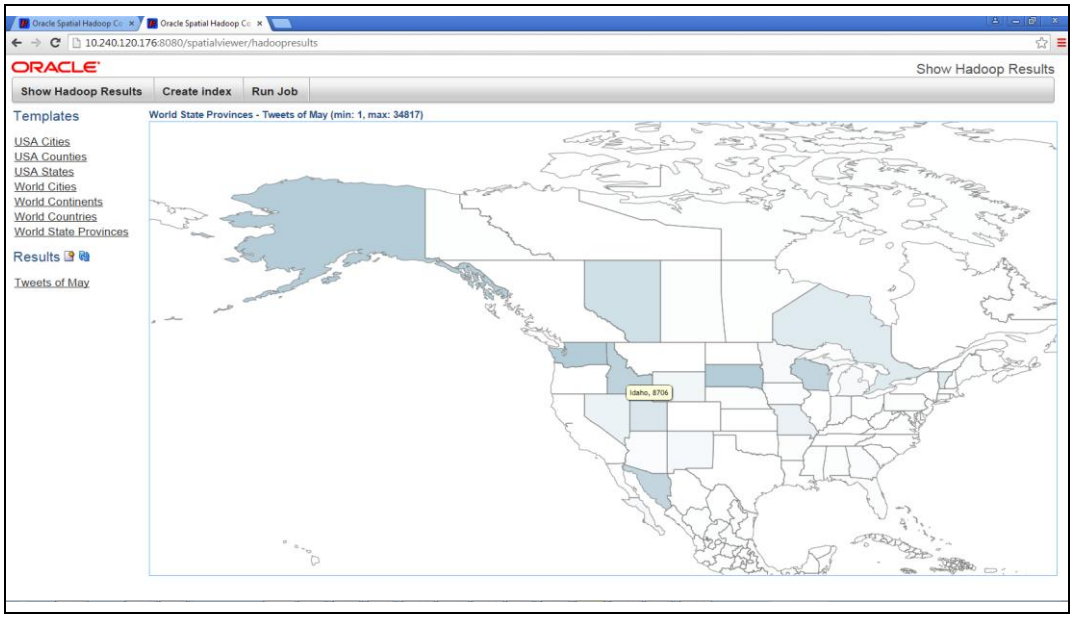

Abb. 4:Screenshot der Beispielanwendung für Vektordaten

Verarbeitung von Rasterdaten
Neben der Verarbeitung von Vektordaten lassen sich auf der Hadoop Plattform auch sehr performant parallele Operationen auf Rasterdaten, also beispielsweise Satellitenbildern oder Orthophotos, durchführen. Diese Daten liegen üblicherweise in einem der gängigen Ausgangsformate (GeoTIFF, etc.) vor und können mit einem im Lieferumfang von Oracle Big Data Spatial and Graph enthaltenen Loader einzeln oder in Gruppen in HDFS übertragen werden. Der Loader basiert auf der Geospatial Data Abstraction Library (GDAL) und unterstützt darüber mehr als 100 verschiedene Datenformate. Auf den Daten in HDFS können mittels Oracle Big Data Spatial and Graph dann Operationen, wie die Erzeugung von Bildausschnitten, die Umprojektion in andere Koordinatensysteme, oder die

Zusammensetzung zu großen Mosaiken aus mehreren ggf. überlappenden Aufnahmen durchgeführt werden. Außerdem lassen sich Algorithmen zur Bildverarbeitung auf diese Rasterdaten anwenden. Zu diesem Zweck werden die Originaldateien in Blöcke zerlegt und auf HDFS blocks verteilt.
Für Algorithmen, die zur Berechnung eines neuen Pixels auch die benachbarten Punkte benötigen, lassen sich die Bilder auch mit einem oder mehreren Pixeln Überlapp auf die Blöcke verteilen. Dies erlaubt die nachfolgende Parallelisierung bei gleichzeitig geringstmöglichem Transfer von Datenblöcken zwischen den Rechnerknoten. Danach lässt sich die Prozessierung mittels MapReduce wie gewohnt durchführen. Am Ende werden die Ergebnisse dann zusammengefügt und in einer einzigen Datei abgelegt.
Wie im Falle der Vektordaten wird eine kleine Beispielanwendung mit ausgeliefert. Sie ist als Vorlage für eigene Lösungen gedacht und unterstützt die Ladeprozesse, verwaltet einen Katalog aller Rasterdaten im System und kann genutzt werden, um die oben beschriebenen MapReduce Jobs zu starten. Auch der MapViewer Client ist integriert, um die Ergebnisse zu visualisieren und eine interaktive Möglichkeit zu schaffen, die Datenbestände in HDFS zu durchsuchen.

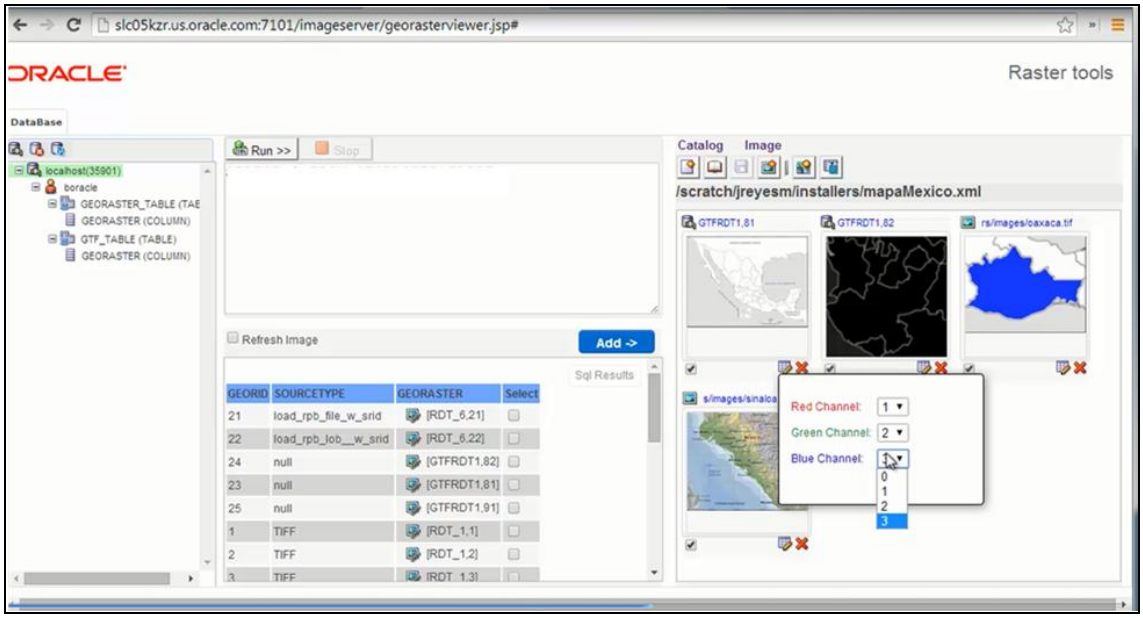

Abb. 5:Screenshot der Image Server Beispielanwendung

Gesamtarchitektur
Hat man die entsprechenden Bibliotheken und Frameworks zur Verfügung, lassen sich auf der Hadoop Plattform sehr große, variable Datenmengen massiv parallel verarbeiten. Bislang werden diese Systeme im Zusammenhang mit raumbezogenen Daten meist zur Vorverarbeitung eingesetzt, um die Ausgangsdaten so weit zu bereinigen und verdichten, dass sie auf konventionellen Datenbank-Plattformen, also insbesondere in Oracle Spatial and Graph, der weiteren Analyse, Visualisierung und vor allem auch Auslieferung zugeführt werden. Beide Plattformen ergänzen sich, indem die geringerwertigen Massendaten in der kostengünstigeren Hadoop Umgebung prozessiert werden und dann die höherwertigen, aufbereiteten Datenbestände dort genutzt werden, wo bereits jahrelang vor-integrierte Tools und Prozesse etabliert sind und auch Expertise wesentlich einfacher zu bekommen ist.

Kontaktadresse:

Hans Viehmann
ORACLE Deutschland B.V. & Co. KG
Kühnehöfe 5
D-22761 Hamburg

Telefon:	+49 (0) 40-89091-173
Fax:	+49 (0) 40-89091-250
E-Mail	hans.viehmann@oracle.com
Internet:	www.oracle.com/goto/spatial