Perzentile mit Hive ermitteln

Ausgangspunkt

Dies ist eine Erweiterung zu dem Dokument „Perzentile mit Hadoop ermitteln“. Der ursprüngliche zweite Hadoop Job, der die Analyse und das Ermitteln der Perzentile übernommen hat, wird durch mehrere HQL-Queries (Hive Query Language) ersetzt. Die Queries verwenden die gleichen Input Daten, wie der Analyse Job.

Vorgehen

Um einen gleichberechtigten Vergleich des Analyse Jobs und der HQL-Query zu erhalten, arbeiten beide Methoden auf den gleichen Inputdaten, die von der Simulation geliefert werden. Da das Ausgabeformat der Simulation zu Beginn nicht für Hive angedacht war, ergeben sich einige redundante Werte beim Import der Simulationsergebnisse in Hive Tabellen. Um zwischen **Key** und **Value** zu unterscheiden, wird als Trennzeichen der Tabulator **‘\t‘** verwendet. Der **Key** Wert entspricht dem Simulationsergebnis. Der **Value** Wert enthält die **NODE_REF**, **DIRECTION** und die Zeilennummern (Vergleich mit Ausgabeformat Simulation). Diese befinden sich aber alle in einer Spalte, daher können die Informationen nicht mehr sinnvoll ausgewertet werden.

Was	Key Column	Value Column
Simulation Ausgabe	4.54863\t29249_0;3_59	
In Hive	4.54863	29249_0;3_59

Um dennoch eine sinnvolle Abfrage nach der **DIRECTION** anbieten zu können, werden die Daten in Partitionen geteilt. Die Tabelle wird wie folgt erstellt:

> hive > **create table simulation_res(key double, value string) partitioned by(direction string) row format delimited by ‘\t’ stored as textfile;**

Danach werden die Daten wie folgt importiert:

> hive > **load data inpath ‘/output/results/0.txt‘ into table simulation_res partition(direction=’0’);**

Die restlichen sieben Dateien mit Angabe der entsprechenden **DIRECTION** werden äquivalent importiert.

Der vorhergehende Analysejob fragt gleichzeitig alle **900** Perzentile für die acht **DIRECTION**s und alle Simulationen insgesamt ab. Dies ist mit Hive nicht möglich. Man muss alle **DIRECTION**s einzeln abfragen:

> hive > **select percentile_approx(key, array(0.01,0.02,0.03, [restliche Werte] ,0.97,0.98,0.99), SIMULATION_COUNT_DIRECTION) from simulation_res where simulation_res.direction = 0;**

Bei der Abfrage der Perzentile von allen Simulationsergebnissen entfällt die Where-Klausel:

> hive > **select percentile_approx(key, array(0.01,0.02,0.03, [restliche Werte] ,0.97,0.98,0.99), SIMULATION_COUNT_TOTAL) from simulation_res;**

In der HQL-Query muss auch die Anzahl der Simulationen angegeben werden. Die Informationen befinden sich in der Datei **total_simulation.txt**. Der übergebene Wert steuert die Approximationsgenauigkeit auf Kosten des Speichers. Wenn der Wert mindestens genauso groß wie die Anzahl an Reihen in einer Tabelle ist, gibt die **percentile_approx** Funktion die exakten Perzentile zurück. Bei dieser Aufgabe entspricht eine Reihe einer einzigen Simulation. Da wir keine approximierten Perzentile sondern die exakten abfragen wollen, müssen wir den Parameter übergeben. Der Standardwert ist mit 10.000 oft zu klein.

Die **percentile_approx** Funktion gibt nur die **Key**s der Tabelle für die Perzentile und nicht die kompletten Reihen mit den anderen Spalten zurück. Dafür ist eine extra Abfrage nötig, wo einer der zuvor ausgegebenen Schlüssel übergeben werden muss:

> hive > **select * from simulation_res where key = [KEY];**

Wie bereits erwähnt, entspricht bei der Aufgabe der **Key** dem Simulationsergebnis. Bei der obigen Abfrage erhält man den **Value** Wert, in dem sich die **NODE_REF** und **DIRECTION** befinden.

Für die Abfrage einer Reihe für eine spezifische **DIRECTION** muss die Where-Klausel erweitert werden:

> hive > **select * from simulation_res where key = [KEY] and simulation_res.direction = [DIRECTION]**

Auswertung

Die Benchmarks wurden wieder auf dem gleichen Cluster der **Friedrich-Schiller-Universität Jena** ausgeführt. Die Ergebnisse befinden sich in der Excel Tabelle.

Die Simulation hat insgesamt **100.000.000** Ergebnisse geliefert. Alle acht Dateien wurden wie beim Vorgehen beschrieben in eine Hive Tabelle mit acht Partitionen importiert. Damit sind die Zeiten mit den Zeiten der Analyse von Job 2 bis Job 13 vergleichbar. Diese Jobs brauchten im Schnitt **91** Sekunden, um alle **900** Perzentile zu ermitteln.

Hive braucht für eine einzige **percentile_approx** Query im Schnitt nur rund **70** Sekunden. Allerdings müssen anschließend noch die Werte der gesamten Reihe abgefragt werden, was durchschnittlich weitere **56,69** Sekunden in Anspruch nimmt. Erst dann hat man die kompletten Informationen mit **NODE_REF** und **DIRECTION**. Eine **percentile_approx** Query über alle Daten benötigt **183** Sekunden und eine anschließende Abfrage der Reihe im Schnitt weitere **78** Sekunden.

Es wird deutlich, dass Hive wesentlich mehr Zeit als der Analyse Job benötigt, um die gleichen Daten zu ermitteln. Schon allein eine **percentile_approx** Query über alle Daten benötigt die doppelte Dauer.

Bei der Durchführung mit Hive ist ein weiteres Problem aufgetreten. Die Funktion **percentile_approx** liefert keine genauen sondern approximierte Perzentile, obwohl der Approximationsparameter groß genug gewählt wurde (vgl. **Result** Spalte in der Excel Tabelle). Erwartet wird ein Array mit Werten, die die gleiche Anzahl an Nachkommastellen wie die **Key** Werte der Reihe haben. Jedoch beginnt beispielsweise das Ergebnis für die **percentile_approx** Query für die **DIRECTION** 90 mit einem korrekten und einem approximierten Wert:

> [-2.32566,-2.053359090909091, …]

Laut Dokumentation müssten exakte **Key** Werte für die Perzentile ermittelt werden. Der Approximationsparameter wurde mehrmals mit der Anzahl an Reihen in der Tabelle überprüft. Eine mögliche Ursache dafür könnte sein, dass der Wert schlichtweg zu groß ist. Mit kleinen Testtabellen funktionierte der Parameter wie in der Dokumentation beschrieben.

Fazit

Mit Hive ist es sehr einfach möglich, Perzentile von bestehenden Daten zu ermitteln, ohne das Eingabeformat anpassen zu müssen. Mit Hilfe von Partitionen können Daten gut aufgeteilt werden. Bei Abfragen, die sich nur auf eine Partition beziehen, liest Hive auch nur die entsprechenden Dateien ein, was einen großen Performancegewinn mit sich bringt. Hive skaliert sehr gut, was man an der Anzahl an erstellen Mapper Instanzen bei mehr Inputdateien und der gesamten CPU Zeit im Vergleich zur Dauer erkennen kann.

Allerdings müssen bei Hive mehrere Abfragen gestartet werden, um die gleichen Ergebnisse wie beim Analyse Job zu erhalten. Approximierte Perzentile sind keine Hilfe, wenn man eine konkrete Reihe abfragen möchte, um die entsprechende **NODE_REF** und **DIRECTION** für ein Simulationsergebnis zu erfahren.

Ausblick

Da die HQL-Query mit den bestehenden Daten arbeiten muss, besteht noch einiges Optimierungspotential. Die Simulation muss nicht mehr die Zeilennummern ausgeben, was viel Speicher sparen kann und somit die IO Zeit für die Ausgabe stark reduziert. Die Ausgabe der Simulation muss nicht sortiert sein, daher können viele Reducer bei der Simulation parallel arbeiten. Wie stark dieser Performancegewinn gegenüber den Zeitverlust der HQL-Abfragen überwiegt, muss erprobt werden.