Wir haben doch keine Zeit – Tipps zum Oracle-Text-Index in Webcenter Content

Gunther Thielemann, Slix Gesellschaft für Computersysteme mbH

Mit OracleTextSearch (OTS) ermöglicht WebCenter Content (vormals Universal Content Management, UCM) die Suche nach Metadaten und Inhalten auch über große Bestände. Auch wenn mit der Version 11*g* die Performance gesteigert wurde, dauert der komplette Aufbau eines Index über mehrere Millionen Dokumente immer noch eine Weile. Der Artikel beschreibt, wie man diesen Prozess mit geringfügigen Änderungen weiter beschleunigen kann, indem man die Aufgaben geschickt auf Content Server und Datenbank verteilt.

Eine Anwendung, die schon seit einigen Jahren im Einsatz ist, sollte ein Upgrade von UCM 10*g* R3 auf UCM 11*g* erfahren. Die Anzahl der Dokumente war mittlerweile auf etwa vier Millionen angewachsen. Mit dem Upgrade war ein kompletter Neuaufbau der Search Collection verbunden. Da dieser Vorgang in der Vergangenheit mit geringerem Volumen nicht nur Tage, sondern Wochen in Anspruch genommen hatte, wurde nach einer Lösung gesucht, um den Zeitaufwand wesentlich zu reduzieren. Gleichzeitig sollten der Fortschritt besser überwacht und realistische Aussagen zur voraussichtlichen Fertigstellung ermöglicht werden.

Suche nach dem Flaschenhals

Die Dauer des Neuaufbaus ist neben Anzahl, Größe und Typ der Dokumente von der Umgebung und der verfügbaren Hardware abhängig. Die Untersuchung des vorhandenen Systems ergab, dass der Gesamtprozess nicht, wie etwa zu vermuten, durch die aufwändige Text-Extraktion oder zu geringen Durchsatz im Netzwerk, sondern durch die Synchronisation des Oracle-Text-Index gebremst wurde. Als hinderlich erwiesen sich die beim I/O-Verhalten der Datenbank festgestellten Latenzen. Da sich im Rahmen des Projekts daran nichts ändern ließ, mussten andere Lösungen gefunden werden.

Es wurde nach einer Möglichkeit gesucht, die Synchronisation des Index durch parallele Ausführung zu beschleunigen. Dass sich damit auch ohne Änderungen am I/O-Subsystem der Datenbank die Geschwindigkeit deutlich steigern ließ, wurde durch Tests bestätigt. Blieb nur noch die Frage, wie sich diese Erkenntnis für den Indexaufbau unter UCM nutzen ließe.

Die im Repository-Manager konfigurierbaren Parameter *(siehe Abbildung 1)* sind übersichtlich. „Content-Objekte pro Indizierungsbatch" legt die Anzahl der Dokumente je Stapel fest. Der zweite Wert bestimmt, in welchen Intervallen ein Checkpoint erfolgt. Hier empfiehlt Oracle, die Voreinstellung von 100.000 Dokumenten beizubehalten. Größere Werte führen auch nach unserer Erfahrung nicht zu merklichen Geschwindigkeitsvorteilen und wirken sich bei eventuellen Unterbrechungen kontraproduktiv aus. Eine Verringerung bedingt häufigere Checkpoints und kostet damit Zeit.

Wenn je Stapel mehr Objekte verarbeitet werden, kann das den Durchsatz durchaus steigern. Beliebig heraufsetzen kann man die Anzahl jedoch nicht. Um den Text zu extrahieren, werden alle Dokumente eines Stapels in den Hauptspeicher (der JVM) geladen; sie teilen sich diesen mit den Threads des Content Servers. Die roten Felder in *Abbildung 2* zeigen die zu durchlaufenden Schritte. In der vorgesehenen Abfolge wird nach Speichern von Metadaten und Text in der Datenbank die Prozedur „CTX_DDL.SYN_INDEX" aufgerufen und auf deren Abschluss gewartet.

Den Aufruf so abzuwandeln, dass die Synchronisation parallel erfolgt, wäre na-

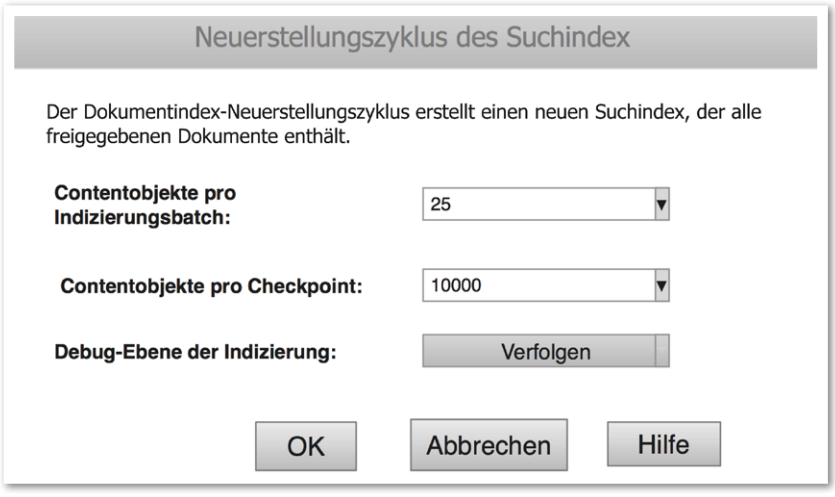

Abbildung 1: Parameter im Repository Manager

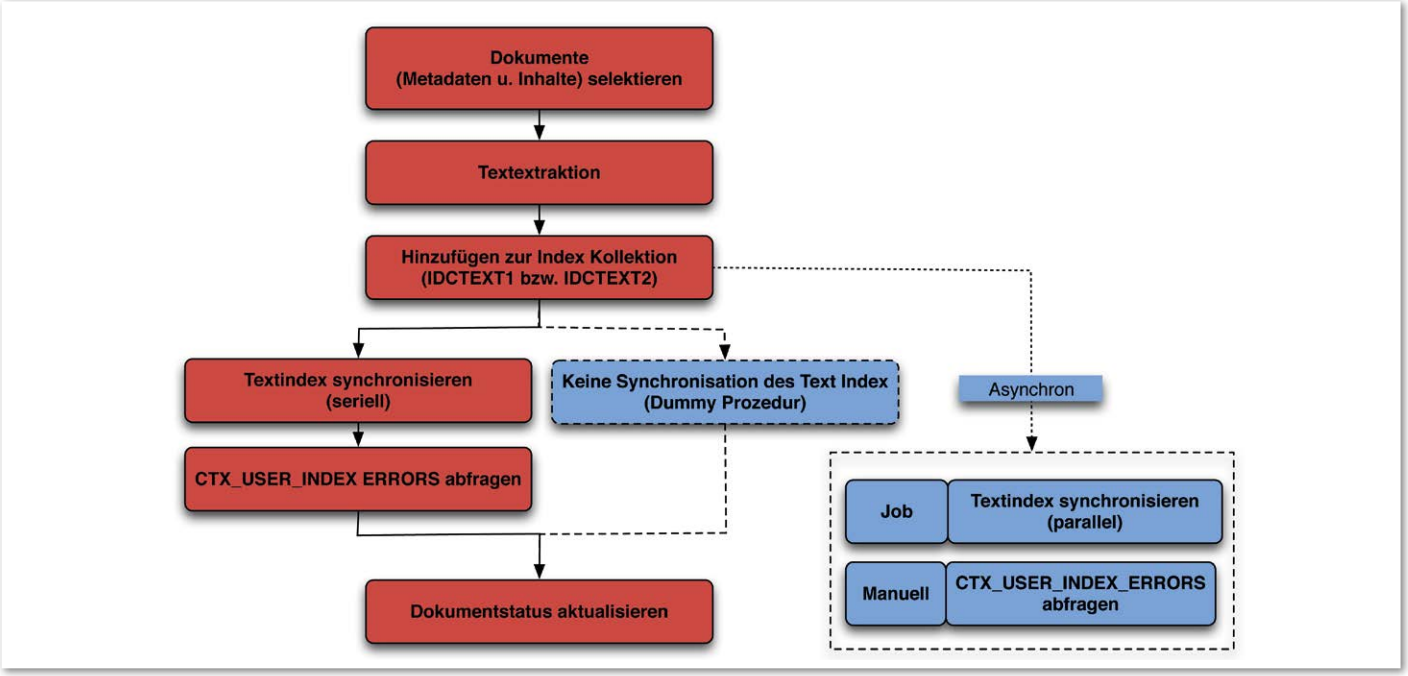

Abbildung 2: Ablauf-Indizierung

heliegend, doch dafür sind die Stapel einfach zu klein. Ein höherer Wirkungsgrad erfordert die parallele Verarbeitung von mehreren Tausend Objekten und damit die Veränderung des Ablaufs. Wie in *Abbildung 2* mit den blauen Feldern dargestellt, kann man diesen Schritt ausgliedern und asynchron durch einen Job in der Datenbank ausführen lassen. Dabei kontrolliert der Content Server die Synchronisation des Text-Index in der Datenbank nicht mehr, kann aber, ohne auf deren Abschluss zu warten, den nächsten Stapel in Angriff nehmen. Bei Testläufen traten in diesem Zusammenhang keine Probleme auf. Die Überwachung eines Datenbank-Jobs verursacht keinen außergewöhnlichen Aufwand.

```
<tr>
   <td>(ORACLE.CALLABLE)CoracleTextSyncIndex</td>
   <td>{call CTX_DDL.SYNC_INDEX(?, ?, NULL, ?, ?)}</td>
   <td>indexName varchar
       memory:50M varchar
       parallelDegree:1 int
       maxTime:15 int</td>
</tr>
```

Listing 1

```
<tr>
   <td>(ORACLE.CALLABLE)CoracleTextSyncIndex</td>
   <td>{call DEV_OCSSEARCH.DUMMY_SYNC
       (?, ?, NULL, ?, ?)}
   </td>
   <td>indexName varchar
       memory:50M varchar
       parallelDegree:1 int
       maxTime:15 int</td>

</tr>
```

Listing 2

Umsetzung

Die notwendigen Anpassungen sind mit geringem Aufwand zu implementieren. Welche Prozedur der Content Server bei der Synchronisation von OracleTextSearch aufruft, wird in der Datei „query.htm“ im Verzeichnis „<middleware_home>/Oracle_ECM1/ucm/idc/resources/core/tables“ bestimmt *(siehe Listing 1)*.

Für Tests ist es ausreichend, nach Erstellen einer Kopie die Anpassungen in der Originaldatei vorzunehmen. Später sollte man dafür eine Komponente verwenden. Die Änderungen *(wie in Listing 2)* und das Erstellen der Prozedur in der Datenbank gemäß *Listing 3* bewirken, dass sich der Content Server nach dem nächsten Neustart beim vollständigen Neuaufbau nicht weiter mit der Synchronisation des Text-Index beschäftigt, sich beim Fast Index Rebuild aber genauso verhält wie vorher. *Listing 4* zeigt, wie der Job erstellt wird. Im diesem Beispiel wird die Synchronisation im Intervall von fünf Minuten gestartet. Der „IndexName“ ist abhängig vom aktiven Index im Content Server mit „FT_IDCTEXT1“ beziehungsweise „FT_IDCTEXT2“ anzugeben.

Um die Datenbank besser für Oracle Text optimieren zu können und Prozesse parallel auszuführen, ohne mit dem Content Server um die Ressourcen zu konkurrieren, wurde die Search Collection in eine separate Instanz verlagert *(siehe Abbildung 3)*.

Memory Parameter

Wie viel Hauptspeicher den Prozessen für die Synchronisierung des Text-Index zur Verfügung steht, lässt sich über einen optionalen Parameter beim Aufruf der Prozedur „CTX_DDL.SYNC_INDEX“ definieren. Ohne diese Angabe greift der voreingestellten Standard, normalerweise sind das 12 MB. Oracle empfiehlt die Anpassung der Voreinstellungen, sobald der Gesamtumfang des zu indizierenden Content 50 GB übersteigt.

Um einem Prozess mehr als 50 MB zuzuteilen, ist zuvor der Maximalwert „MAX_INDEX_MEMORY“ heraufzusetzen, wie in *Listing 5* gezeigt. Die absolute Obergrenze sind 2 GB. Bei der Dimensionierung ist zu berücksichtigen, dass dieser Speicher nicht Bestandteil der SGA ist und sich die Angaben auf jeden einzelnen Prozess beziehen.

Fortschrittskontrolle

Die im Ressource-Manager-Applet des Content Servers angebotene Fortschrittskontrolle ist für die Überwachung des automatischen Update-Zyklus sicher ausreichend, für den Neuaufbau eines großen Index liefert sie allenfalls eine Momentaufnahme. Wie in *Listing 6* dargestellt, kann man durch periodische Abfrage der Datei „<domain_home>/cs/search/rebuild /state.hda“ den Status fein granular in einer „.csv“-Datei protokollieren. Die so gesammelten Werte gestatten eine realistische Prognose der voraussichtlichen Fertigstellung und eignen sich auch für die Visualisierung des Verlaufs *(siehe Abbildung 4)*.

Ergebnis

Mit dem beschriebenen Vorgehen war der Neuaufbau schon nach einigen Tagen abgeschlossen. Gegenüber einer Dauer von mehr als fünf Wochen zuvor wurde damit eine Verbesserung um eine Größenordnung erreicht. Der Durchsatz war über den gesamten Zeitraum nahezu konstant. In Tranchen mit mehreren Tausend Dokumenten ließ sich zum einen die Synchronisierung des Text-Index gut parallelisieren und zum anderen wurde der Index nicht so stark fragmentiert. Da keine Anforderung bestand, dass neue und geänderte Dokumente in Echtzeit zur Suche zur Verfügung stehen sollten, wurde dieses Verfahren auch über den Neuaufbau hinaus für den periodischen Aktualisierungszyklus beibehalten.

```
create or replace PROCEDURE DUMMY_SYNC (indexName in VARCHar2, parti-
tion in varchar2, memory in varchar2, parallelDegree in number, maxTime
in number)
-- Platzhalter für sync Index
-- Sync Index wird von Job ausgeführt, nicht hier
-- es sei denn, es handelt sich um den Shadow Index
AS
BEGIN
IF upper(indexName) LIKE 'RIO%' THEN
CTXSYS.CTX_DDL.SYNC_INDEX(indexName,memory,null,parallelDegree,maxti
me);
else
 NULL;
end if;
END DUMMY_SYNC;
/
```

Listing 3

```
BEGIN
  DBMS_SCHEDULER.CREATE_JOB (
      job_name => '"DEV_OCSSEARCH"."'',
      job_type => 'PLSQL_BLOCK',
      job_action => 'begin ctx_ddl.sync_index(,'FT_IDCTEXT1''); end;'
      number_of_arguments => 0,
      start_date => TO_TIMESTAMP_TZ(,2014-10-13 10:00' EUROPE/
BERLIN','YYYY-MM-DD HH24:MI TZR'),
      repeat_interval => 'FREQ=MINUTELY; INTERVAL=5',
      end_date => NULL,
      enabled => TRUE,
      auto_drop => FALSE,
      comments => 'Synchronisieren Index');
END;
/
```

Listing 4

```
begin
ctxsys.ctx_adm.set_parameter ('max_index_memory','1024M');
end;
/

begin
ctxsys.ctx_adm.set_parameter ('default_index_memory','256M');
end;
/
```

Listing 5

```
LOG_DIR=/var/tmp
while [ true ]
do
IDXDOC=$(cat $CS/search/rebuild/state.hda | grep totalAddIndex | cut -d
= -f 2)
IDXTIME=$(date "+%d.%m.%Y %H:%M:%S")
echo "$IDXTIME;$IDXDOC;" >> $LOG_DIR/ots_progress.csv
sleep 60
done
```

Listing 6

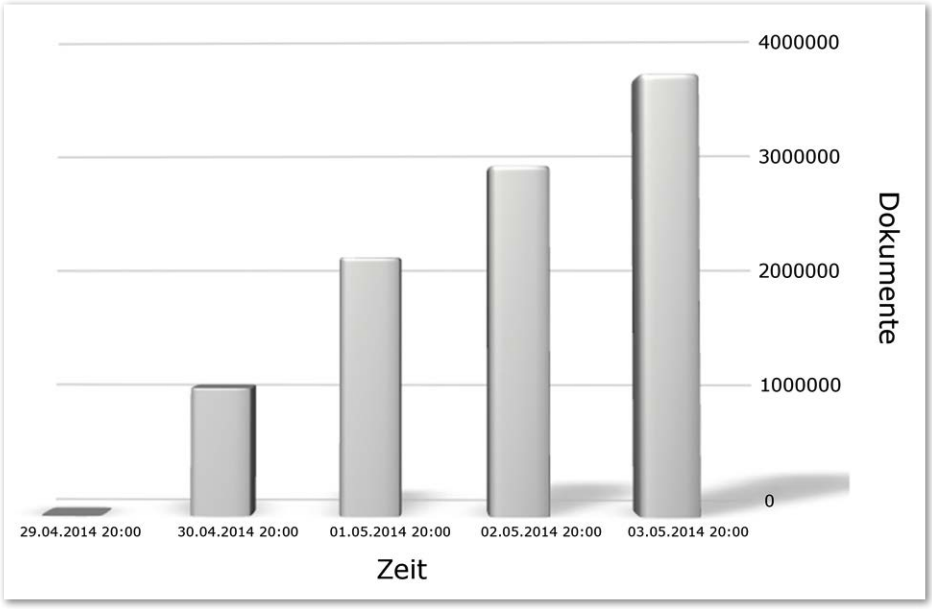

Abbildung 4: Diagramm Index-Neuaufbau

Fazit

Ein kompletter Neuaufbau der Index-Kollektion für Oracle WebCenter Content kann sich als eine durchaus zeitaufwändige Angelegenheit erweisen und bei großem Bestand schon mal mehrere Tage in Anspruch nehmen. Da der bedeutend schlankere und schnellere Fast Index Rebuild zwar viele Fälle abdeckt, aber eben nicht alle, ist diese Prozedur nicht nur einmalig beim Urknall durchzuführen, sondern hin und wieder unumgänglich.

Mit dem hier vorgestellten Lösungsansatz lassen sich der Zeitaufwand deutlich reduzieren und der Fortschritt besser überwachen. Dass die Synchronisierung des Text-Index asynchron über einen Datenbank-Job erfolgt und nicht mehr der Kontrolle des Content Servers unterliegt, hat sich nicht als spürbar nachteilig erwiesen.

Gunther Thielemann
gunther.thielemann@slix.de

DOAG DevCamp – Software Upcycling

Entwickler im Oracle-Umfeld sollten sich den 29. und 30. April 2015 rot im Kalender anstreichen: Unter dem diesjährigen Motto „Software Upcycling" lädt das DOAG DevCamp in Frankfurt erneut zu einer Runde unter Gleichgesinnten. Die Teilnehmer treffen sich zu zwei interaktiven Tage mit spannenden Sessions und einer Abendveranstaltung zum Networken in der Jahrhunderthalle, um in lockerer Atmosphäre ihr Wissen auszutauschen.

Bei diesem Veranstaltungskonzept der DOAG heißt es: Ärmel hochkrempeln und mitmachen. Jeder Einzelne ist gefragt, denn es gibt weder Programm noch Speaker. Eine feste Agenda? Fehlanzeige! Auch Beamer und Standard-Präsentationen sind tabu. Stattdessen bringen die Teilnehmer ihre Ideen und Fragen am Veranstaltungsmorgen einfach mit, stimmen dann die Themen untereinander ab und diskutieren sie anschließend innerhalb der Sessions in Gruppen. Das Ergebnis? Interaktion und lebendiger Wissensaustausch.

Aus Teilnehmern werden Teilgeber

Im Mittelpunkt der Veranstaltung steht das Thema „Software Upcycling – Modernisierung in der Softwareentwicklung". Der Schwerpunkt wird auf Architektur-Themen liegen. Alte Relikte aus längst vergangenen Zeiten, wie Oracle Forms oder COBOL, sollen abgestaubt werden und einen neuen Farbanstrich erhalten. Aber auch ADF, APEX, PL/SQL, agile Softwareentwicklung und Mobile werden thematisiert, sofern die Teilnehmer es möchten. Und das alles in lockerer Atmosphäre: Alle Teilnehmer sind an den Veranstaltungstagen „per du", um echten Team Spirit zu erzeugen und den direkten Austausch zu fördern.

Weiterhin gilt: Eine Session beginnt, wenn sie beginnt, und endet, wenn sie endet. Das freie Format sorgt für eine große Dynamik und Interaktion. Auch das Wechseln zwischen den einzelnen Sessions ist möglich, und sogar gewünscht: der Teilnehmer ist dann wie eine Biene, die für Befruchtungen zwischen den Sessions sorgt. Findet man einmal keine der laufenden Sessions interessant, bieten sich auch am Buffet noch viele Gelegenheiten, um Informationen zu teilen, Kontakte herzustellen und Ideen zu entwickeln.

Das neue Veranstaltungsformat des DOAG DevCamp wurde bereits im letzten Jahr erfolgreich getestet: Rund 70 Teilnehmer kamen in die Allianz Arena nach München und ließen sich von der einmaligen interaktiven und offenen Atmosphäre anstecken und inspirieren. Der hohe Informationsgehalt der Sessions und der rege Wissensaustausch kamen bei den Teilnehmern besonders gut an.

Mehr Informationen zur Veranstaltung unter *http://barcamp.doag.org*