REAL TIME BOM EXPLOSIONS WITH APACHE SOLR AND SPARK

Andreas Zitzelsberger
BILLS OF MATERIAL (BOMS) EXPLAINED
BOMS ARE NEEDED FOR...

- Production Planning
- Forecasting Demand
- Running Simulations
- Scenario-Based Planning
THE BIG PICTURE

Parts and abstract demands

Demand Resolver

Orders / Independent Demands

Production Planning

BOMs / Dependent demands

Analytics
# ORDERS / INDEPENDENT DEMANDS

## Order Data

<table>
<thead>
<tr>
<th>Id, Order Time, ...</th>
</tr>
</thead>
</table>

## Car Configuration

<table>
<thead>
<tr>
<th>TYPE</th>
<th>VARIANT</th>
<th>UPHOLSTERY</th>
<th>PAINT</th>
<th>OPTION_1, OPTION_2, ...</th>
</tr>
</thead>
</table>

... ...

... ...

### Predicted Orders

- Customer’s orders
- Stock cars

---

Image: Energy Solutions International Inc
### Parts and Abstract Demands

<table>
<thead>
<tr>
<th>Id</th>
<th>Description</th>
<th>Attributes</th>
<th>Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>TR_1</td>
<td>Automatic transmission, supplier A</td>
<td></td>
<td>TYPE_1 OR VAR_2 AND NOT OPT_2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>TR_2</td>
<td>Automatic transmission, supplier B</td>
<td></td>
<td>TYPE_2 AND VAR_1 OR OPT_5 OR OPT_6</td>
</tr>
<tr>
<td>DT_1</td>
<td>150hp, automatic transmission drive train</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LI_1</td>
<td>Xenon lights</td>
<td></td>
<td>OPT_3 OR VAR_9 OR ...</td>
</tr>
<tr>
<td>BT_1</td>
<td>Small Battery</td>
<td></td>
<td>8 &lt;= OPT_3<em>200 + VAR_9</em>5 + OPT_4*2 ... &lt;= 12</td>
</tr>
</tbody>
</table>

The constructive approach is impossible due to combinatorial explosion. The rules need to be evaluated during demand resolution.
WE NEED TO OVERCOME THREE CORE PROBLEMS...

Parts and abstract demands

Demand Resolver

Orders / Independent Demands

BOMs / Dependent demands

Production Planning

Analytics
1. HOW TO RESOLVE DEMANDS AS FAST AS POSSIBLE?

- Parts and abstract demands
- Demand Resolver
- BOMs / Dependent demands
- Orders / Independent Demands
2. HOW TO JOIN LARGE DATA SETS?

Parts and abstract demands

BOMs / Dependent demands

Analytics

Orders / Independent Demands
3. ANALYTICS ON LARGE DATA SETS?

Parts and abstract demands

BOMs / Dependent demands

Analytics

Orders / Independent Demands
1. HOW TO RESOLVE DEMANDS AS FAST AS POSSIBLE?

Parts and abstract demands — Demand Resolver — BOMs / Dependent demands — Orders / Independent Demands

On to problem #1...
THE SIMPLE SOLUTION...
A BETTER SOLUTION...
OR: ONLY CALCULATE WHAT IS STRICTLY NECESSARY

2 million parts
15 million abstract demands

~ 6000 possible demands
~ 3000 actual demands

Step 1: FILTER
Pre-filter demands with a fast index in $O(\log n)$. Yields a superset of the required parts.

Step 2: REFINE
Refine the superset by evaluating the usage terms in $O(n)$. 

Abstract Demands

Pre-Filter

Post-Filter

Actual demands
PRE-FILTER USING CANDIDATE AND A PROSCRIPTION SETS

- Consider \(((OPT_1) \text{ OR } (OPT_2)) \text{ AND } \neg(OPT_3 \text{ OR } OPT_4)\)

- Candidate set: \(C(t) = \{OPT_1, OPT_2\}\)
- Proscription set: \(P(t) = \{OPT_3, OPT_4\}\)

- Preselect by the criterium:
  - At least one condition from \(C(t)\) is satisfied
  - None of the conditions from \(P(t)\) are satisfied
EXAMPLE CANDIDATE AND PROSCRIPTION SETS

CANDIDATE SET

A part can only be used together with "white" or "blue" paint.

Entry: PART_1, C-PAINT:[white, blue]

Query: C-PAINT=blue

PROSCRIPTION SET

A part cannot be used with "yellow" paint.

Entry: PART_2, P-PAINT:[yellow]

Query: NOT P-PAINT=yellow
**SOLR IMPLEMENTATION**

<table>
<thead>
<tr>
<th>Solr Server</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Part</strong></td>
</tr>
<tr>
<td>PART_ID: “…”</td>
</tr>
<tr>
<td>// Part attributes</td>
</tr>
<tr>
<td>// …</td>
</tr>
<tr>
<td>TERMS: […]</td>
</tr>
<tr>
<td>TYPE_C: […]</td>
</tr>
<tr>
<td>TYPE_P : […]</td>
</tr>
<tr>
<td>VARIANT_C : […]</td>
</tr>
<tr>
<td>VARIANT_P : […]</td>
</tr>
<tr>
<td>OPTIONS_C : […]</td>
</tr>
<tr>
<td>OPTIONS_P : […]</td>
</tr>
</tbody>
</table>

**Post filter**

If evaluate(TERMS) accept
Else drop

Arrow:

- Bulk Solr query for N orders
- Bulk export of the search results
- Demand Resolver

Diagram:

- Dependent Demands (Solr)
WHY APACHE SOLR

- Really fast
- Powerful faceting
- Solr cloud
- Extensible

Most importantly, it just works
1. HOW TO RESOLVE DEMANDS AS FAST AS POSSIBLE?

Parts and abstract demands → Demand Resolver → Customers / Independent Demands

Orders / Independent Demands

1043x406

1583x736

69x935

823x695

1392x717

349x612

508x805

1217x705

1551x805

1920.0x1080.0

872x323

1032x515
2. HOW TO JOIN TWO LARGE DATA SETS?

Parts and abstract demands

BOMs / Dependent demands

Orders / Independent Demands

let’s continue with problem #2...
FORWARD AND BACKWARD JOINS ON PARTS

Given:
Order criteria (TYPE= XY12)
and parts criteria (PART_ID = 4711 OR 4712)

Task:
Find all orders that satisfy the order criteria
AND contain at least one part, that satisfies the part criteria
Return all orders and the parts for each order that satisfy the part criteria.
Nobody wants to join 7 million x 21 billion x 2 million records.

Nobody wants to store 21 billion records.
PARTIAL DENORMALIZATION TO THE RESCUE

Orders

<table>
<thead>
<tr>
<th>Ids</th>
<th>Attributes</th>
<th>Part Ids</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>...</td>
<td>1, 3, 786,...</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td>2, 6, 786,...</td>
</tr>
<tr>
<td>3</td>
<td>...</td>
<td>3, 75, 95,...</td>
</tr>
</tbody>
</table>

Parts

<table>
<thead>
<tr>
<th>Ids</th>
<th>Attributes</th>
<th>Order Ids</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>...</td>
<td>54, 23321, ..</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td>1, 4, 8652, ..</td>
</tr>
<tr>
<td>3</td>
<td>...</td>
<td>864, 23452, ..</td>
</tr>
</tbody>
</table>

<10 Mio. records instead of 21 billion
SIMPLE, BUT NOT EASY: CUSTOM JOIN LOGIC IN SOLR

**Step 1**
Pre-filter the parts using an index search

**Step 2**
Pre-filter the orders using an index search

**Step 3**
Compute the intersection of both supersets

Final Result

- Custom Search Components
- Solr Search Components
**WE ONLY EXECUTE THE JOIN IF WE MUST**

<table>
<thead>
<tr>
<th></th>
<th>No filter</th>
<th>Filter by order attributes</th>
<th>Filter by part attributes</th>
<th>Filter by order and part attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facets on parts</td>
<td>Index</td>
<td>JOIN</td>
<td>Index</td>
<td>JOIN</td>
</tr>
<tr>
<td>Facets on orders</td>
<td>Index</td>
<td>Index</td>
<td>JOIN</td>
<td>JOIN</td>
</tr>
<tr>
<td>List parts</td>
<td>JOIN</td>
<td>JOIN</td>
<td>JOIN</td>
<td>JOIN</td>
</tr>
<tr>
<td>List orders</td>
<td>Index</td>
<td>Index</td>
<td>JOIN</td>
<td>JOIN</td>
</tr>
</tbody>
</table>

Result sets may be unwieldingly large. Early termination for interactive queries if the result grows too large.
2. HOW TO JOIN TWO LARGE DATA SETS?

Parts and abstract demands

Orders / Independent Demands

BOMs / Dependent demands

Analytics
3. ANALYTICS ON VERY LARGE DATA SETS?

What about problem #3?
WHY APACHE SPARK

- Fast
- High-Level functional API
- Well integrated
- Batch semantics
- Works well with Solr (If you use your own connector...)

Spark
BULK EXPORT WITH A SHARED OFF-HEAP CACHE

Lucene Segment

Lucene Segment

DocId Order DocId Part

Web interface

Shared Off-Heap Cache

Business Facade Object

data

http://chronicle.software/products/chronicle-map/
3. ANALYTICS ON VERY LARGE DATASETS?
LESSONS LEARNED

https://en.wikipedia.org/wiki/Melk_Abbey
OPTIMIZE LATE, BUT DO OPTIMIZE
**SOME OF OUR OPTIMIZATIONS**

- **Bulk RequestHandler**
  - Solving thousands of orders with one request

- **Search components with custom JOIN algorithm**
  - Be able to store data effectively using our own JOIN implementation.

- **Binary DocValue support**
  - Speed up the access to persisted data dramatically using binary doc values.

- **Computing BOM explosions**

- **Boolean interpreter as postfilter**
  - Enable Solr with custom post filters to filter documents using stored bool expressions.

- **Mass data binary response format**
  - Use the standard Solr binary codec with an optimized data-model that reduce the amount of data by a factor of 8.
LOW LEVEL OPTIMIZATIONS CAN YIELD GREAT RETURNS

Development of the processing time Demand Calculation Service PoC

Profiling result and the some improvements to reduce the query time.
DON'T TRUST PAPER, DO POCs

WE'LL TELL THEM IT'S JUST FOR THE PROOF OF CONCEPT

THEN WE'LL PUT IT INTO PRODUCTION
IN CONCLUSION
THANK YOU

@andreasz82
andreas.zitzelsberger@qaware.de