Guy Lohman
(https://wp.sigmod.org/?author=20)

APRIL 10, 2014

IS QUERY OPTIMIZATION A "SOLVED" PROBLEM?

Databases (https://wp.sigmod.org/?cat=16)

*Is Query Optimization a "solved" problem? If not, are we attacking
should we identify the "right" problems to solve?*

I asked these same questions almost exactly 25 years ago, in an ex
Workshop on Database Query Optimization that was organized by t
Graefe at the Oregon Graduate Center [**Grae 89a**]. Remarkably an
of the issues and critical unsolved problems I identified in that brief
Researchers continue to attack the wrong problems, IMHO: ***they at***
can, i.e., that they have ideas for, rather than the ones that ***they s***
critical to successfully modeling the true cost of plans and choosing
importantly, that will avoid choosing a disastrous plan! At the risk o
to re-visit these issues, because I'm disappointed that few in the re
taken up my earlier challenge.

(http://wp.sigmod.org/wp-content/uploads/2014/04/divid

The root of all evil, the Achilles Heel of query optimization, is the es
intermediate results, known as cardinalities. Everything in cost estir
many rows will be processed, so the entire cost model is predicated
model. In my experience, the cost model may introduce errors of at
cardinality, but the cardinality model can quite easily introduce erro
magnitude! I'll give a real-world example in a moment. With such
"Why did the optimizer pick a bad plan?" Rather, the wonder is "Wh
pick a decent plan?"

"Well," you say, "we've seen lots of improvements in histograms, w statistics since 1989. Surely we do better now." There's been no sh true, but the wealth of such papers precisely illustrates my point. D that improve selectivity estimation for individual local predicates of 47 AND 63" by a few percent doesn't really matter, when other, mu introduced elsewhere in cardinality estimation dwarf those minor im engineering, folks. If I have to review one more such paper on impr predicates, I'll scream (and reject it)! It ***just doesn't matter***! Wha enough.

What still introduces the most error in cardinality estimation is (*a*) h parameter markers, (*b*) the selectivity of join predicates, and, even how we combine selectivities to estimate the cardinality. Amazingly have enjoyed the least research attention, or at least the fewest nu to solve them, unless I've missed some major contributions lately. I topics in turn, describing the fundamental causes of errors, and why reach disastrous proportions, illustrated by war stories from real cu

Host variables and parameter markers

Host variables, parameter markers, and special registers occur in S applications on top of the DBMS, not humans, invoke most queries, papers. Such applications typically get the constants used in predic blank" field on a web page, for example. The SQL predicate then lo :hv1 AND :hv2". At compile time, the optimizer has no clue what :h cannot look them up in those wonderful histograms that we all have make a wild guess on the average or likely values, which could be c execution to the next, or even due to skew. A war story illustrates t

One of our major ISVs retrofitted a table with a field that identified came from. It had 6 distinct values, but 99.99% of the rows had th subsystem, i.e., when there was only one. A predicate on this subsy to every query, with the value being passed as a host variable. Not priori, DB2's optimizer used the average value of 1/|distinct values| predicate's true selectivity was usually 0.9999 (not selective at all) was 0.0001 (extremely selective).

There has been some work on this so-called Parametric Query Optir
sometimes attacking the problem of other parameters unknown at (
number of buffer pages available) or limited to discrete values [**Ioa**
favorites is a fascinating empirical study by Reddy and Haritsa [**Re(**
several commercial query optimizers as the selectivity of multiple lo
It demonstrated (quite colorfully!) that regions in which a particular
be convex and may even be disconnected! Graefe suggested keepir
possible value of each host variable [**Grae 89b**], but with multiple l
number of possible values, Graefe's scheme quickly gets impractica
decide at run-time among the large cross-product of possible plans,
into regions having the same plan [**Stoy 08**].

Version 5 of DB2 for OS/390 (shipped June 1997) developed a prac
optimization for host variables, parameter markers, and special reg
bind options REOPT(ALWAYS) and REOPT(ONCE). The latter re-opti
the statement is executed with actual values for the parameters, ar
values will be "typical", whereas the former forces re-optimization e
run. **Later (http://publib.boulder.ibm.com/infocenter/dziche**
topic=%2Fcom.ibm.db2z10.doc.comref%2Fsrc%2Ftpc%2Fd
a REOPT(AUTO) option was added to autonomically determine if re-
based upon the change in the estimated filter factors from the last

Selectivity of join predicates

The paucity of innovation in calculating join predicate selectivities is
extant systems still use the techniques pioneered by System R for c
a join as the inverse of the maximum of the two join-column cardin
essentially assumes that the domain of one column is a subset of th
assumption is valid for key domains having referential integrity con
overlap of the two sets may vary greatly depending upon the sema
domains. Furthermore, the common practice of pre-populating a dir
"Date" with 100 years of dates into the future can incorrectly bias t
fact table initially has only a few months of data. Statistics on join i
give much more precise selectivities for join predicates, if we were '
costs of maintenance and lock contention of updates to these join ir
indexes would not solve the **intersection problem** typical of star s

For example, suppose a fact table of Transactions has dimensions fo
and Dates of each transaction. Though current methods provide acc
for predicates local to each dimension, e.g., ProductName = 'Docke
Jose' and Date = '23-Feb-2013', it is impossible to determine the ef
these predicates on the fact table. Perhaps the San Jose store had a
Dockers that day that expired the next day, and a similar sale on so
day, so that the individual selectivities for each day, store, and proc
the actual sales of Dockers on the two days would be significantly d
interaction of these predicates, through join predicates in this case
doesn't address. This leads naturally to the final and most challengi

Correlation of columns

With few exceptions ([**Chri 83**], [**Vand 86**]), query optimizers sinc
selectivities as probabilities that a predicate on any given row will b
these individual selectivities together. Effectively, this assumes that
= Value2} = Prob{ColX = Value1} * Prob{ColY = Value2}, i.e., tha
probabilistically independent, if you recall your college probability. F
database industry, this assumption is often valid. ***However, occas***

My favorite example, which occurred in a customer database, is Ma
'Accord'. To simplify somewhat, suppose there are 10 Makes and 10
independence (and uniformity) assumption gives us a selectivity of
since only Honda makes Accords, by trademark law, the real selecti
under-estimate the cardinality by an order of magnitude. Such optir
worse than pessimistic over-estimation errors, because they cause
certain operations will be cheaper than they really are, causing nast
The only way to avoid such errors is for the database administrator
semantic relationship (a functional dependency, in this case) betwe
its consequences, and to collect ***column group statistics***, as **DB2
products now allow
(http://www.ibm.com/developerworks/data/library/techar
0612kapoor/)**.

To identify these landmines in the schema automatically, Stillger et
the LEarning Optimizer (LEO), which opportunistically and automati
actual cardinalities to optimizer estimates, to identify column combi
correlation errors. Ilyas et al. [**Ilya 04**] attacked the problem more

(CORrelation Detection by Sampling), searching somewhat exhaust
between any two columns in samples from the data before running
and colleagues [**Mark 05**], [**Mark 07**] have made ground-breaking
way to combine the selectivities of conjuncts in partial results.

All great progress on this problem, but ***none yet solves the probl***
predicates that can be inadvertently introduced by the query write
that "more is better", that providing more predicates helps the DBM
American as Apple Pie! Let me illustrate with one of my favorite wa

At a meeting of the International DB2 User's Group, a chief databas
major U.S. insurance company whom I'd helped with occasional bac
conduct a class on-site. I suggested it include an exercise on a real
After my class, she obliged me by presenting two 1-inch stacks of p
EXPLAIN of a plan for a query. I feared I was going to embarrass m
under the gun. The queries differed in only one predicate, she said,
seconds whereas the one with the extra predicate took over an hou

I instinctively examined first the cardinality estimates for the result
slower one had a cardinality estimate 7 orders of magnitude less th
asked what column the extra predicate was on, my host explained t
key constructed of the first four letters of the policy-holder's last na
initials, the zip code, and last four digits of his/her Social Security N
query have predicates on all those columns? Of course! And how m
table? Ten million. Bingo! I explained that that predicate was compl
others, and its selectivity, $1/10^7$, when multiplied by the others, un
cardinality by 7 orders of magnitude, wreaking havoc with the plan.
minutes to figure this out, and I was immediately dubbed a "genius
straightforward: the added predicate might help the run-time, espe
on that column, but it totally threw off the optimizer, which couldn't
that redundancy without LEO or CORDS.

So c'mon, folks, let's attack problems that really matter, thos
optimizer disasters, and stop polishing the round ball.

Disclaimer: The postings on this site are my own and don't necessa
positions, strategies or opinions.

References

[Chri 83] S. Christodoulakis, "Estimating record selectivities", Info. Systems 8,2 (1

[Grae 89a] G. Graefe, editor, Workshop on Database Query Optimization, CSE Tec
Graduate Center, Portland, OR, 30 May 1989.

[Grae 89b] G. Graefe, "The stability of query evaluation plans and dynamic query
of ACM-SIGMOD, Portland, OR (1989).

[Ioan 97] Y. Ioannidis et al., "Parametric query optimization", VLDB Journal, 6(2),

[Ilya 04] I. F. Ilyas, V. Markl, P. J. Haas, P. Brown, A. Aboulnaga: CORDS: Automatic
Soft Functional Dependencies. SIGMOD Conference 2004: 647-658.

[Mark 05] V. Markl, N. Megiddo, M. Kutsch, T. Minh Tran, P. J. Haas, U. Srivastava:
Selectivity of Conjuncts of Predicates. VLDB 2005: 373-384.

[Mark 07] V. Markl, P. J. Haas, M. Kutsch, N. Megiddo, U. Srivastava, T. Minh Tran:
estimation via maximum entropy. VLDB J. 16(1): 55-76 (2007)

[Redd 05] N. Reddy and J. R. Haritsa. "Analyzing plan diagrams of database query

[Seli 79] P.G. Selinger, M.M. Astrahan, D.D. Chamberlin, R.A. Lorie, and T.G. Price, "
Relational Database Management System", Procs. Of ACM-SIGMOD (1979), pp. 23

[Stil 01] M. Stillger, G. M. Lohman, V. Markl, M. Kandil: LEO – DB2's LEarning Opti

[Stoy 08] J. Stoyanovich, K. A. Ross, J. Rao, W. Fan, V. Markl, G. Lohman: ReoptSMA
Cache. **Technical report (http://www.cs.columbia.edu/~wfan/PAPERS/Tech**

[Vald 87] P. Valuriez, "Join indices", ACM Trans. on Database Systems 12, 2 (June

[Vand 86] B.T. Vander Zanden, H.M. Taylor, and D. Bitton, "Estimating block acces
correlated, Procs. of the 12th Intl. Conf. on Very Large Data Bases (Kyoto, Sept. 1

Blogger’s Profile:

Dr. Guy M. Lohman (http://researcher.ibm.com/researcher/view.php?persor *Disruptive Information Management Architectures in the Advanced Information M Research Division’s Almaden Research Center in San Jose, California, where he has currently manages the Blink research project, which contributed* ***BLU Acceleratio******(http://www.ibm.com/software/data/db2/linux-unix-windows/db2-blu-acc*** *and Windows (LUW) 10.5 (GA’d 2013) and the query engine of the IBM Smart Anal V1.1 and the* ***Informix Warehouse Accelerator (http://www.ibm.com/softwar advanced-enterprise-edition)*** *products (2007-2010). Dr. Lohman was the archit DB2 LUW and was responsible for its development from 1992 to 1997 (versions 2 – prototyping of Visual Explain, efficient sampling, the DB2 Index Advisor, and optim DB2. Dr. Lohman was elected to the* ***IBM Academy of Technology (http://www.ibm.com/ibm/academy/index.html)*** *in 2002 and made an IBM M the General Chair for* ***ACM’s 2013 Symposium on Cloud Computing (http://ww*** *General Co-Chair of the* ***2015 IEEE International Conference on Data Engineer (http://www.icde2015.kr)****. Previously, he was on the editorial boards of the “Ver “Distributed and Parallel Databases”. He is the author of over 75 papers in the refe has been awarded 39 U.S. patents. His current research interests involve disruptive Business Intelligence, advanced data analytics, query optimization, self-managing management appliances, database compression, and autonomic problem determin*

Systems & Databases: Let’s Break Down the Walls (https://wp.sigmod.org/?p=1009)

Exploratory search: New name for an old hat? (https://wp.sigmod.org/?p=1183)

8 Comments

James Parker on April 17, 2014

I found this an interesting and useful paper, given that this is not my area of expertise (main-memory DBMS). However, I wonder to what extent query optimization ought to also consider the working set of rows already in memory when determining a query plan as well. Such an approach might even improve performance when paging of the underlying in-VM rowss is possible, by keeping them in the working set(s) of the DBMS process(es).

Stephen Dillon (http://www.linkedin.com/in/stephendillon/) on April 20, 2014

James,

I too was contemplating the role of an in-memory DBMS (IMDBS) and these query optimization points. Just how much of an impact will they contribute to a system such as VoltDB? This is actually something I've had discussions with other architects about to address their own proprietary main memory DBMS.

My initial thoughts are that a commercial IMDBS may be impacted less, than a traditional RDBMS, by such optimizations when you consider the significant impact main-memory operations already provide queries. By commercial, I really mean one that is professionally developed and properly addresses latching, locking, durability, et al. That does not mean proper or better optimization is not needed for an IMDBS. Nobody has proven these optimizations yet with an IMDBS to my knowledge. I just believe that as compared to a traditional RDBMS, an IMDBS may not benefit as much. I also believe that these modern main-memory DBs such as memSQL or VoltDB allow "some" bad code to hide under the guise of fast execution plans that have benefited from zero disk reads. Now for someone's proprietary IMDBS well these optimizations could certainly provide more benefits considering how less efficient it is to a commercial product.

Ruslan Fomkin (http://se.linkedin.com/in/fomkin/) on April 25, 2014

I haven’t tested, but in my opinion the query optimization has similar effect on in-memory and disk-based databases. I see two reason for this:
(1) Data in disk-based database are cached in main-memory. Thus no disk access in both DBMS types. The main difference is cardinality units: pages vs cache lines.
(2) The main bottleneck for query execution is memory wall nowadays. Thus suboptimal query plans in IMDBMS can still perform by an order of magnitude worse then optimal one.

James Parker on April 26, 2014

Stephen,

I think you misunderstood me; I was not thinking directly of main memory (AKA in-memory) DBMSs, but rather about the memory hierarchy (memory vs. “disk”) and where specific data resides at the time of a query. In a traditional DBMS, some data is known to be in memory, including which rows from which tables — a query optimizer might favor joins using tables with most or all rows in memory over those which are not, as a part of query optimization. Of course the effort must take into account paging, if it may occur; however paging can often be managed programmatically, e.g., by “locking” pages in memory.

The similar issue in a main memory DBMS would be where in the CPU cache hierarchy data resides; this is generally of much lower utility, since machine architectures generally do not expose ways to manage these caches programmatically (although I have speculated as to how hardware and OS architects might provide useful features toward this end, and would almost certainly need to be combined with processor scheduling and interrupt handling).

spaghettidba (http://spaghettidba.com) on April 18, 2014

Thanks for the interesting writing.
The recently released SQL Server 2014 incorporates a new cardinality estimator that (partially) addresses the predicate independance issue. The new algorithm used is know as "exponential backoff" and it helps reducing the estimation error with multiple predicates.
You can read about it here: **http://www.sqlperformance.com/2014/01/sql-plan/cardinality-estimation-for-multiple-predicates (http://www.sqlperformance.com/2014/01/sql-plan/cardinality-estimation-for-multiple-predicates)**

Lothar Flatz on April 18, 2014

After 15 years of tuning in the field I have come accross all of these issues.
This I have my own examples i.e. Correlation of columns: mobile phone service provider. Primary key of table units consist of customer number and phone number. Selectivity of join predicates: Road assistence search for service centers. Selectivity differs largely depending on country of accident. Wrote a paper on it. Hope it gets accepted. 🙂

Alexandr Savinov (http://conceptoriented.org/savinov) on April 18, 2014

Thanks for the interesting post and relevant works. One problem in query optimization I am currently dealing with is optimizing column operations for analytical queries. It is highly important for a novel system for analytical data integration (ConceptMix – **http://conceptoriented.com (http://conceptoriented.com)**) where all operations with data are described in terms of an algebra of functions (a function represents a column).

Essentially, all manipulations are reduced to manipulating functions (and inverse functions) and the problem is to translate queries from the concept-oriented expression language to these functional operations.

Radim Baca on April 22, 2014

Thank you for an inspirational article. I have a question regarding the “redundant predicates problem”: can the CORDS method solve the problem? From my perspective this problem is just a result of high correlation of attributes, which is addressed by CORDS. Thanks

Comments are closed

Search … Search

Categories

Select Category

Recent Comments

Atsuyuki Morishima on **Imagine all the People and AI in the Future of Work (https://wp.sigmod.org/?p=2931#comment-54638)**

Zakaria Maamar on **Imagine all the People and AI in the Future of Work (https://wp.sigmod.org/?p=2931#comment-54635)**

Yuriy on The Rise of Natural Language Interfaces to Databases (https://wp.sigmod.org/?p=2897#comment-54545)

william e winkler (https://www.census.gov/srd/csrmreports/byyear.html) on Data cleaning is a machine learning problem that needs data systems help! (https://wp.sigmod.org/?p=2288#comment-54417)

Gio Wiederhold (http://i.stanford.edu/~gio) on A Tribute to José Alfredo Blakeley (https://wp.sigmod.org/?p=2261#comment-54399)

Archives

April 2016 (https://wp.sigmod.org/?m=201604)

February 2016 (https://wp.sigmod.org/?m=201602)

November 2015 (https://wp.sigmod.org/?m=201511)

September 2015 (https://wp.sigmod.org/?m=201509)

July 2015 (https://wp.sigmod.org/?m=201507)

June 2015 (https://wp.sigmod.org/?m=201506)

March 2015 (https://wp.sigmod.org/?m=201503)

February 2015 (https://wp.sigmod.org/?m=201502)

December 2014 (https://wp.sigmod.org/?m=201412)

November 2014 (https://wp.sigmod.org/?m=201411)

August 2014 (https://wp.sigmod.org/?m=201408)

June 2014 (https://wp.sigmod.org/?m=201406)

April 2014 (https://wp.sigmod.org/?m=201404)

February 2014 (https://wp.sigmod.org/?m=201402)

December 2013 (https://wp.sigmod.org/?m=201312)

October 2013 (https://wp.sigmod.org/?m=201310)

September 2013 (https://wp.sigmod.org/?m=201309)

June 2013 (https://wp.sigmod.org/?m=201306)

May 2013 (https://wp.sigmod.org/?m=201305)

March 2013 (https://wp.sigmod.org/?m=201303)

December 2012 (https://wp.sigmod.org/?m=201212)

November 2012 (https://wp.sigmod.org/?m=201211)

August 2012 (https://wp.sigmod.org/?m=201208)

Blog editor: Georgia Koutrika

ACM SIGMOD Blog