Moving up the food chain: Supporting E-Commerce Applications on Databases

Anant Jhingran
IBM Almaden Research Center
anant@us.ibm.com

Abstract

Database systems have enjoyed a tremendous market because they have served many applications really well – transaction processing in the beginning, and then decision support. Today, with over 200% cumulative growth rate in certain segments of E-Commerce, it is clear that this new class of applications will be a strong driver for databases to grow, commercially, as well as from a Research perspective. This paper outlines some of the issues that I have learnt in dealing with E-Commerce applications that may well be the focus of some of the research in database systems over the course of next few years.

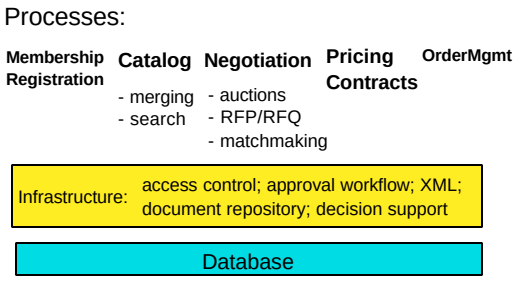
1. Introduction

A typical E-Commerce application is a three-tier application with the middle tier being where the bulk of the application logic runs. The lowest tier is a database, and the application uses the database for persistence and complex querying, but still does a lot of heavy data management lifting because the capabilities of the current commercial systems may not be up to par. (And sometimes, the issues of portability reduce it to using the lowest common denominator.)

Michael Stonebraker [Stonebraker 2000] argues that DBMS's of the future will have all the piece parts to be the middle tier where the application such as Ariba's ORMS and WebSphere Commerce Suite will be written. While I do not necessarily agree with his view of taking over the world, it is indeed true that DBMSs of the future can be much more supportive of these new applications.

A typical E-Commerce application is characterized by several building blocks, an example of which is IBM's WebSphere Commerce Suite, Marketplace Edition, which allows a company to set up an electronic marketplace. See [Jhingran 00] for some details on what these processes and infrastructure requirements are.

Typical Components of a B2B E-Commerce Application



In each subsystem, special issues arise with respect to database support. In general, it turns out that many of these applications tend to do data management tasks that would be better done by the database system of the future. We now describe some of the issues and pose research problems. I would also strongly recommend that people interested in database research pick up a copy of ACM SIG EC '00 to get a good idea of what are some of the application level issues facing E-Commerce researchers.

2. Document Management

It is apparent that B2B E-Commerce is conducted through document exchange among businesses, and EDI (Electronic Document Exchange) is a well respected, albeit old, standard in this space. It is also obvious that XML is the way to go for the documents of the future, and therefore we need to store and manage these documents. A great wealth of information about the emergence of XML as the lingua franca for B2B messaging can be found at www.xmledi.com.

A typical message flow in this would be: inbound message → router → repository → application. For the repository to be useful for the application, it must provide the following, above and beyond storage and retrieval of documents: 1. Search (find me all suppliers whose purchase orders have been received in the last ten days and who have not been invoiced since then) 2. Linking of documents (invoice and P.O., for example), 3. Signing and other authentication, 4. Data Mining (e.g.,

discovering that the suppliers in northeast tend to delay shipment of parts in the winter months) and 5. Combining data from different XML sources (such as SAP and a CAD drawing).

Current XML repositories, such as Xperanto [Carey 00] and Poet [www.poet.com] have addressed the "put" and the "get" capabilities of XML, and I believe that we will see a very fruitful next couple of years for research around the other issues mentioned above.

3. Application Model

Most of the current E-Commerce applications follow the EJB programming model. There is still a continental divide between this and the SQL (or Object QL) paradigm, and bridging this gap will allow many more functions to be pushed into the database. Issues of materialization of the objects (partial, in most case) in the application space in an efficient manner have not yet been addressed.

Compounding this programming model chasm is the performance of database systems in three tier applications. Many of these applications require path lengths to "materialized objects" which are far smaller than a fully buffered table can provide. Consequently, the applications tend to build their own object caches, and even use kernel extensions (such as IBM's AFPA cache, www.research.ibm.com/compsci/os/brochure.html) to achieve satisfactory performance. A promising area of research is such caches in a three tier application model fully supported through database replication.

Commercially, TimesTen with its main memory technology and FronTier caching [TimesTen 00] is trying to occupy this important technical front. However, it is important to realize that the application logic of a typical E-Commerce application has considerably greater path-length above the data management system, so even extreme efficiency within the data management system is likely to only yield limited results. What is required is that the future applications push considerably larger function into the database system – once that is achieved, many of these performance enhancing features that we can build, will begin to pay dividends.

4. Catalog Management

This will, in my opinion, be the single most important area of database research in support of E-Commerce application. There are three specific areas that we believe are important in this context.

4.1 Catalog Integration

Already, commercial companies such as Cohera and Aspect Development (now acquired by i2) have realized the importance of this facet of E-commerce – if I have 80,000 suppliers (such as what Grainger supports in its catalog), how do I keep them in sync? In particular, since electronic catalogs are generally hierarchical (using some automatic or human defined category and taxonomy), how does one merge different catalogs into this hierarchy? [Hellerstein 00] gives a good overview of some of the technical challenges associated with this problem.

Since electronic marketplaces are dynamic (with suppliers coming and leaving), one major problem is how the catalog hierarchy is kept up-to-date? Consider a market for electronic marketplaces. This market has a catalog of semiconductor parts organized in a hierarchy **H**. When a new manufacturer joins this marketplace, it has its own hierarchy **h** for an overlapping (but not necessarily identical) set of electronic parts. It is assumed that in each hierarchy (**H** or **h**), with each product **P**, there is a text and/or <attribute, value> pair description. The problem for hierarchy merging then, is, the "recommendation" for each node **P** in **h**, the ranked order list of nodes in **H** that it best belongs to. [Agrawal Manuscript 00] describes an innovative approach that uses data mining technique of Naïve Bayes classification [Mitchell 97] to achieve this task with over 90% accuracy in a variety of industrial settings.

Other data cleansing, schema and metadata mapping techniques (such as Clio [Miller 2000]) are also clearly applicable, but we have only begun to scratch the surface.

4.2 Verticalized Schema

A typical catalog containing 100,000 Stock Keeping Units (or SKU's) may contain 10,000 different parts, and relational databases are just not good at managing so many tables, or in managing a universal table with so many nulls. Consequently, many E-Commerce applications deal with catalog represented in the form of <id, attribute name, attribute value>, which as might be suspected, is quite inefficient for finding shirts with blue collar and price < $30 (requiring at least a two way self-join as opposed to a simple index based selection). An interesting area of research is whether we can efficiently support the view of

10,000 tables from an application perspective, yet manage the database from a single verticalized table.

4.3 Search

A typical search through an electronic catalog takes one of two forms: browsing through the hierarchy; or getting to the product through specified search criteria. The former is relatively easy, except that the database can do a better job at prefetching once this browsing mode is known. The latter is interesting from the point of view that each subsequent query is related to the previous one, typically by addition or deletion of a clause. Information Retrieval systems have typically handled this well through efficient and-ing and or-ing of tuple-ids; relational systems have not.

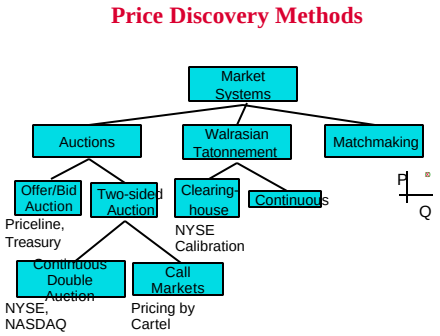
Another aspect of a search through an electronic catalog is the concept of "nearness" applied on a feasible region. For example, if I am looking for steel screws, my feasible region might be very narrow along the product specifications, but might be more flexible along the delivery date and price axis of the specification. The metric for nearness then depends on the specific query; typical R-Tree like data structures require fairly rigid distance metrics.

5. Matchmaking

Unlike fixed price catalog shopping, typical B2B E-Commerce is much more "negotiated." In addition, in a marketplace, there may be multiple buyers and sellers for the same product. In those environments, the following typically takes place: matchmaking → n:m negotiation → 1:1 negotiation.

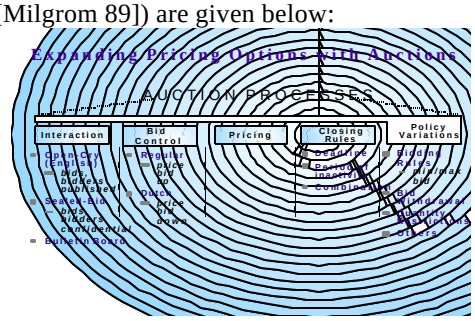
In matchmaking, among all buyers and sellers, the people who are eventually likely to match are pulled together (e.g., at NASDAQ, this would be the set of buyers and sellers interested in trading Microsoft). The subsequent step, of n:m negotation, tries to match individual buyer with individual sellers, using several algorithms such as double sided auctions (where the buyers bid among themselves to be the highest price bidder, and the sellers bid among themselves to be the lowest price asker) or Walrasian Tatonnment. When n = 1, this is typically called an auction, and when m = 1, it is called procurement (or RFP/RFQ process). The final step of 1:1 negotiation happens because often business dealings involve negotiations on very complex terms and conditions (T&C's) which are difficult to automate in the matchmaking or n:m negotiation phase.

Some of the n:m negotiation models are shown below.



Two-sided continuous double auctions are simple forms of two queues, and can be maintained by inserts and deletes from one or two tables. However, to-date I have not seen an effective performance evaluation of the various options (such as pure database implementation; main memory heap-based implementation, or something in between), and TPC-W benchmark is still focused on retailing environment. In a similar vein, while there have been several works on pushing down mining functions into the database, I expect to see more work on pushing down other matching functions, such as Walrasian Tatonnment, which currently are done entirely outside the database.

In addition, some of the variations in auctions (one of the most studied methods in economics, and my belief that every computer science student must read [Milgrom 89]) are given below:



Auctions have other performance problems, and often lead to some interesting deadlock situations.

An interesting problem in matchmaking is what I have called the "mating dance in electronic marketplaces." A Buyer (or a seller) would not like to reveal information about his trading position to the parties that he would end up not doing a trade with -- this concept of information

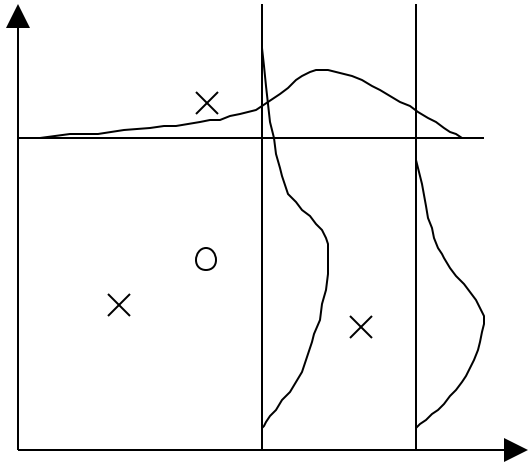
hiding is very critical in Electronic Marketplaces where a trading position is either a key indicator of the seller's or buyer's strategy, or where the details of the position can be used to extract a better deal by the opposing parties.

Of course, not revealing any information leads a party to be not matched with anyone; and that is clearly counterproductive. So the issue is the tradeoff between revelation of information and the speed (or accuracy) of match. Already, Electronic Marketplaces are being built to this model. As an example, consider ViMP, a Virtual Insurance MarketPlace being built by IBM Zurich. In such a marketplace, there is a hierarchical revelation of information. A set of dialogs might go like this:

- Customer: "I am a male aged 50, and I need term life insurance"
- 20 out of 50 insurance companies respond saying "yes, we offer"
- Customer: "BTW, I am a smoker"
- Another 10 drop out
- Customer: "I had a heart attack last year"
- ….

This is a classic case of "need to know" -- participants reveal information only when it is needed. The classical matching problems come in a wide variety. Typically, they involve bipartite graphs (in our case, a bipartite graph G of N nodes, with edges between B buyers and N-B = S sellers). We could do several weighted matching problems; however intuition tells us that the right matching is a "stable marriage" matching [Gusfield and Irving].

From a database aspect, the primary question is: what are the data structures (such as R-Trees) that will lead to a good match in the presence of incomplete information? Figuratively, when the positions of the buyers and sellers (marked with O and X respectively) are in a two dimensional space, then the unknown values of a seller (either x or y or both unknown) can be represented by a probability curve, as shown in the following figure.



Every participant can declare one's position using (for simplicity's sake) either fully specified position (i.e. both x and y known), or one dimension unspecified. All the participants know that the probability distribution for the unknown dimension. If the buyer (represented by a O) has one shot of determining with whom to conduct the negotiations, which of the X's does he choose? What data structures will allow this choice to be made correctly (albeit probabilistically)? While there have been works dealing with unknown dimensions for spatial indexes, their extensions to this matching problem is, to the best of my knowledge, still open.

6. Decision Support

Databases in general have become very good at supporting data mining and OLAP kinds of queries against standard customer and point-of-sales data. However, in B2C environments, the dimensionality of the problem is significantly higher, and there are several research papers dealing with personalization in this domain. However, B2B environments (supply chain and e-marketplaces) remain significantly understudied. Typical decision support questions that need to be answered in this environment are:

- How are my suppliers responding?
- What is a good strategy for negotiating prices?
- What is a good strategy for setting prices (i.e. how much will the market bear, or what are the competitors doing)?
- What are the futures for a commodity X?
- What external factors are influencing the market?

Except for the first (where traditional OLAP style queries work well), we need to invent new algorithms (borrowing from financial and competitive business intelligence domain) and use

databases to effectively answer these. Especially in e-markets, where a good marketplace has a global view of the entire world's trade in some commodities, the size of the database over which we have to do some of these new inferences could be terabytes.

A big problem that will need solutions over the next few years is what I call the one petabyte problem, where the one petabyte is composed of 1000 *different* one terabyte databases, as opposed to one large one petabyte database. This is by virtue of the fact that suppliers are loathe to lose control over their data, and decision support across the supply chain (assuming that a large enterprise has O(1000) suppliers) will involve running distributed queries across these databases, with the associated schema and access control challenges.

7. Personalization and Privacy

Of course, B2C commerce is all about personalization. In particular, data management (especially data mining) research has moved well into this area of "collaborative filtering" and other forms of model building [Resnick 97, Sarwar 2000] and I will not belabor the obvious here.

Rakesh Agrawal has convinced me that the next battleground for this in Research is the so-called "Hipporactic Databases" [Agrawal KDD 00, Agrawal SIGMOD 00] which preserve the privacy of the individual, while still serving the marketing needs of the enterprise. Statistical database research (e.g., [Adam 89]) provides a good foundation for this kind of research.

8. Conclusions

We are at an interesting time in the database research field. Applications such as E-Commerce are beginning to capture the mindshare of the market, and are beginning to use databases in somewhat simplistic terms. We have to "grow up" to support more of their requirements (by working hand-in-hand with them) so that we continue to be excessively relevant well into the 21st century.

9. Acknowledgments

I have benefited from discussions with several researchers including Rakesh Agrawal, Hamid Pirahesh, Manoj Kumar, Vibby Gottemukkala, Chung-Sheng Li and Raghu Ramkrishnan, and my thanks to all of them for helping shape my thoughts.

10. References

[Adam 89] Adam, N. and Wartman, J., "Security Control Methods for Statistical Databases," ACM Computing Surveys, 21(4), 1989.

[Agrawal SIGMOD 00] Agrawal, R. and Srikant, R., "Privacy Preserving Data Mining," *Proc. SIGMOD* 2000.

[Agrawal KDD 00] Agrawal R., "Hippocratic Databases," *Invited Talk, ACM KDD*, 2000.

[Agrawal Manuscript 00] Agrawal, R., and Srikant, R., "Merging Product Hierarchies in an E-Marketplace," *manuscript in preparation*, 2000.

[Carey 00] Carey M. et al., "XPERANTO: Middleware for Publishing Object-Relational Data as XML Documents," *Proceedings VLDB*, 2000.

[Gusfield 97] Gusfield, D. and Irving R., "The Stable Marriage Problem: Structure and Algorithms," *M.I.T Press*, 1989.

[Jhingran 00] Jhingran A., "Anatomy of a Real E-Commerce Application," *Proc. SIGMOD* 2000.

[Hellerstein 00] Hellerstein, J., "Technical Requirements for Production-Level B2B E-Catalogs," *Cohera Corp. White Paper*, 2000.

[Milgrom 89] Milgrom P., "Auctions and bidding: a primer," *Journal of Economic Perspectives*, 3(3), 1989.

[Miller 00] Miller, R.J. et al., "Schema Mapping as Query Discovery," *Proc. VLDB*, 2000.

[Mitchell 97] Mitchell, T., "Machine Learning," *Chapter 6, McGraw Hil*l, 1997.

[TimesTen 00] Neimat, M-A. et al., "High-Performance and Scalability Through Application Tier In-Memory Data Management", *Proceedings VLDB,* 2000.

[Resnick 97] Resnick, P. and Varian H., "Recommender Systems," *CACM*, 40(3), 1997.

[Sarwar 00] Sarwar, B. et al., "Analysis of Recommendation Algorithms for E-Commerce," *Proceedings ACM EC '00*, 2000.

[Stonebraker 00] Stonebraker, M., "Future of OR DBMS's," *Invited Talk, IBM Almaden Research Center*, 2000.