# Repeating History Beyond ARIES

25th International Conference on Very Large Data Bases
Edinburgh, September 1999

**C. MOHAN**

mohan@almaden.ibm.com
www.almaden.ibm.com/u/mohan/

**IBM Almaden Research Center**
**650 Harry Road, K01/B1**
**San Jose, CA 95120, USA**

# Agenda

- Introduction
- History of ARIES
- Family of Algorithms
- Transaction Management in Web Age
- Summary and Future

ARIES: Algorithms for Recovery and Isolation Exploiting Semantics

Acks: **THANKS** to all my current/past colleagues who worked with me, and implementors/enthusiasts of ARIES everywhere, especially IBM management

# Introduction

- Concurrency control and recovery (CC&R): Core functionality of any transaction system
- Books and paper collections on CC&R
- Real systems' internals much more complex than what they typically teach you!
- Goals of this talk
  - NOT a survey/tutorial on CC&R
  - History behind emergence/success of ARIES family
  - Brief introduction to some algorithms
  - Few points on real world directions - history repeats itself!

# FAQs

- Is ARIES a product/prototype/project name?
- Does IBM have patents on this technology?
- IBM let all this fancy technology be published?
- Do the algorithms have to be so complicated?
- Have you proved that they are correct?
- Where are the performance numbers with numerous graphs?

# ARIES History

- **Late-70s'** Sys R conclusion: Recovery with write-ahead logging better than with shadow paging BUT Sys R's CC&R paradigms later heavily influenced R&D
- **Mid-80s:** Basic CC&R believed to be a dead research topic by general DBMS community
- After R* project at IBM Almaden, Starburst was initiated to design a new extensible DBMS
- A few of us (non-Sys-R) decided to revisit Sys R legacy
  - Found major unsolved problems in the area of CC&R: Using write-ahead logging, efficient, fine granule locking with logical logging and flexible storage management
  - Important algorithms left undocumented: index CC&R, partial rollback handling, ...
  - Some significant original design flaws still remained in product version of Sys R: e.g., space reservation

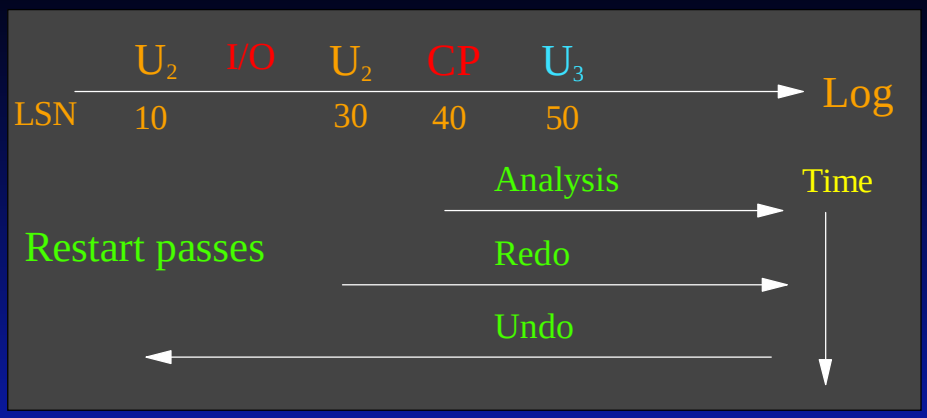
# ARIES History

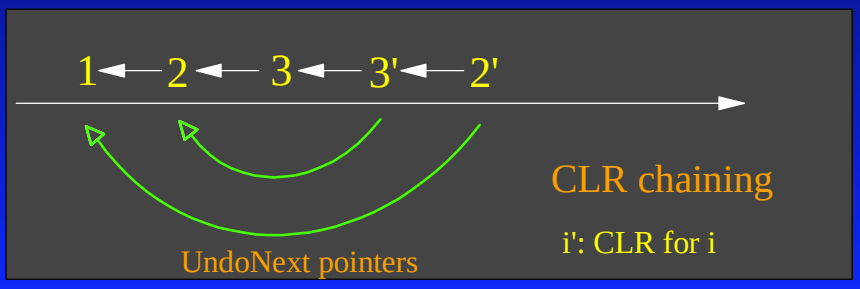
- Dug up some old unpublished Sys R memos
- Reverse engineered code of Sys R, SQL/DS, DB2/MVS, IMS: Many developers were long gone
- Consulted with developers of mainframe DB2 to learn why its recovery differed significantly from Sys R
- Synergy from researchers-developers interactions
  - Benefitted from accumulated prototype/product history
  - Greater appreciation of customer problems with resulting algorithms being much more realistic
  - Evolutionary, rather than revolutionary, solutions - very important for technology transfer
- Resulted in formation of Data Base Technology Institute (DBTI) to encourage interactions between IBM Research and DBMS product groups: Huge success!

# Basic ARIES Algorithm

- Every page has a Log Sequence Number (PageLSN)
- Buffer manager tracks dirty pages using RecLSNs
- Log ALL updates on per page basis, including updates performed during rollbacks - latter with redo-only CLRS (Compensation Log Records)
- Regularly checkpoint transaction table and RecLSNs
- On restart after system failure
  - ► Analyze log from most recent checkpoint to end to update checkpointed info
  - ► Repeat history (i.e., redo missing updates) from min(RecLSNs) to end of log
  - ► Undo in-flight transactions

# Basic ARIES



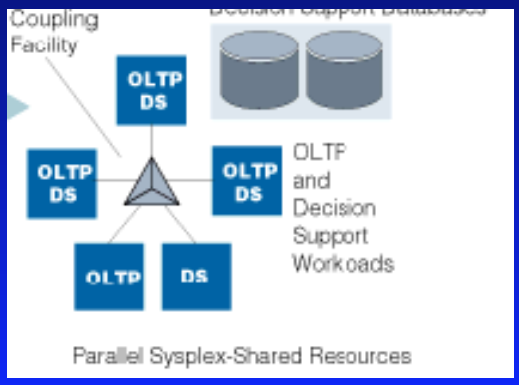



# Paradigm Shifts Compared to Sys R

- Normal processing
  - Logging all changes, including space management and indexes
  - Tracking page state using LSN
  - Using short-duration latches rather than longer duration locks for physical consistency
- Rollbacks: Writing CLRs to describe updates
- Restart recovery
  - Redoing all transactions' changes, including losers' (repeating history)
  - Redo pass before undo pass

# The ARIES Family of Algorithms

- Index management
  - Very limited knowledge/understanding all around
  - Locking on key values: ARIES/KVL
  - Recovery and locking on data identifiers: ARIES/IM
- Nested transactions: ARIES/NT (VLDB89 paper)
- Clustering of S/390 systems with shared disks and coupling facility with even more failure possibilities: ARIES for SD



## The ARIES Family of Algorithms

- Client server systems with client caching of data beyond transaction end and server managing data and log disks: ARIES/CSA
- Simple technique exploiting recovery info for locking optimizations: Commit_LSN
- Managing messages with high concurrency: ARIES MQ
- Linear hashing locking and recovery: ARIES/LHS
- Fast restart by processing new transactions during restart recovery
- Fast (parallel, incremental) database backup and restore
- Online index build (i.e., while table is being updated)

## Domino and Transactions

- New feature of R5: Result of joint work between Dominotes project at IBM Almaden and Iris
- Logging optional at DB granularity
- Implicitly each API call treated as a transaction
- Single log per server
- Extensions to ARIES to permit LSN-based recovery, and to handle unlogged updates to attachments, and for switching between logged and unlogged DB modes

# Domino Recovery Complications

- Original design of storage management not done with recovery in mind
- Too many persistent structures
- A single file with different data structures: B+-tree, hash access method, bit maps, summary buckets, non-summary buckets, attachments' storage, lists, arrays, ...
  - Some are paginated, others are byte streams
  - Some pages have headers and trailers, others don't
  - Pages are of varying sizes
- Structures get allocated, deallocated, migrated
- How to handle situations where user overwrites an existing database file with an older/newer replica?

# ARIES/Domino Extensions

- Could not afford/tolerate complete redesign of on-disk formats to conform to ARIES requirements
- Use of traditional DBMS as persistent layer also ruled out due to complexity, Notes API semantics + flexible data model, ...

Evolution was needed rather than revolution!!

- Preanalysis of log to identify data structure allocations/deallocations and logging of migrated data
- Deal with direct I/Os that bypass buffer pool and use of OS file caching
- Eliminates "fixup" at restart after failure and allows fuzzy backups

# ARIES Impact

Numerous IBM products and prototypes


- DB2 RDBMS workstation and mainframe versions
- Starburst extensible DBMS
- MQSeries/390 transactional messaging & queuing product
- Lotus Domino/Notes

Lotus .domino

- ADSM backup and restore product on numerous platforms
- Encina transaction monitor
- QuickSilver distributed operating system

# ARIES Impact

- Other company products
  - Microsoft SQL Server and NT File System
  - $O_2$ object-oriented DBMS
  - ???
- Research impact and prototypes
  - >150 citations by others to original ARIES paper
  - Extensions, formalization of subsets, ...
  - Gamma database machine, Exodus extensible DBMS, Shore persistent object system, Paradise GIS
  - Predator object-relational DBMS
  - Cosmos, KAIST, Korea
  - Pjama - persistent Java
  - ???

# Transactions in Web Age

- At first, RAS (reliability, availability, serviceability) used to be a concern only for high-end (i.e., mainframe) customers - banking, insurance, ...
- Niche market of fault-tolerant vendors
- Then, open systems (i.e., Unix) got more serious about RAS - still, mainly in large enterprises
- With web, even small and medium businesses need good RAS, high performance and security - change of culture **in progress**
  - Whole businesses totally dependent on IT infrastructure
  - Very different expectations on such market-facing systems
  - Easy for new players to emerge and attain quick popularity but they can also very quickly go down with inadequately designed IT systems
  - Infamous outages in Amazon, eBay, E*Trade, e.Schwab, ...

# Themes

- Product developers innovating more and more
- Transactional messaging-queuing systems in widespread usage (e.g., MQSeries)
  - Publish-subscribe a major growth area (e.g., Tibco) with hardly any research attention
- E-businesses fuel growth of service providers: ISPs (AOL), ASPs (Corio), CSPs (Exodus)
- Replication within and across system types (relational, groupware, ...)
- Remote site back up and disaster recovery
- Support for clusters - load balancing and fail over
- Online reorg necessary for 24X7 availability
- Pervasive computing devices with persistent data and mobile computing

# Web-based Systems

- Led to 3 tier distributed computing with thin clients
- Renewed interest in mainframe systems for network-centric computing
- Systems management and predictable performance are important
  - ► Guaranteed quality of service with workload managers
  - ► Graceful degradation of service with unexpected loads
- Groupware gets transaction and web-server functionality; links with traditional TP systems (e.g., Lotus Domino/Notes and CICS)
- Web enabled workflow management systems (WFMSs)
- E-commerce and supply chain management demand interoperable WFMSs

# Summary and Future

- Concurrency control, recovery and storage management must be considered together
- Internet may be hot and flashy, but solid infrastructure and age-old concerns still crucial - Java, CORBA and Windows systems not yet industrial strength enough
- Simple solutions, shortcuts ultimately return to haunt!
- No web presence better than one working badly
- TPC-W to be released
- Tech transfer takes time/patience, especially to existing products - evolutionary solutions more likely to succeed
- Open systems do not warrant ignoring the mainframe past, not even IMS technologies

## Summary and Future

- Advanced transaction concepts will become real via workflow systems
- Main memory DBMSs finally appearing, even though IMS Fast Path had support decades ago!
- Some firms realizing going from producing shrink wrapped software to mission critical software not easy!
- Emergence of service providers will impact usability and manageability aspects of transaction systems

## Summary and Future

- Much more of index CC&R should be taught
- In research, avoid tendency to "go with the flow"
  - Tackle ignored problems
  - Focus more on practical AND intellectually challenging problems (e.g., parallel, concurrent utilities with self tuning)
- Benefited enormously working for a company with numerous transaction systems and great colleagues
- Questioning senior colleagues' conclusions/advice and perseverance ultimately pay off!
- Industry should publish more
  - Industrial tracks in SIGMOD and VLDB are a great start
  - At least company internal publications essential for "debugging" algorithms