

# Spatial Database Applications: Are they for your project?

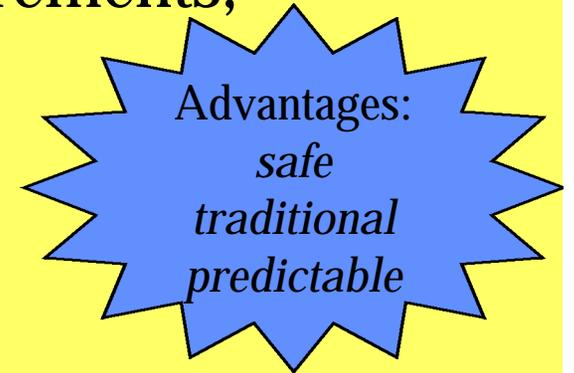
Jeanne Behnke/586

Dr. Kostas Kalpakis/Computer Science/UMBC

March 23, 2000

# Background

- To meet Science Data Processing requirements, projects at GSFC typically build:
  - data processing systems
  - data inventory systems
  - data analysis systems and so on....
- Many projects select the tools used to develop these systems based on:
  - legacy software systems and COTS packages
  - current hardware environments
  - examples used by projects nearby....
- *Problem:*
  - There is no room for radical change
  - Especially true in cost-constrained environments



# Meanwhile,

*back at the Silicon Valley & San Jose...*

- Database management systems have expanded to meet the commercial market
  - extended data types (Universal DBMS)
  - data mining (Red Brick Systems)
  - data warehousing (Platinum Systems)
  - spatial datatyping (complex query applications)
  - data mart applications (metadata parsing applications)
  - data warehouse tools (middleware, intelligent agents)
  - data visualization tools (web implementation)
- Software development techniques have changed radically to meet commercial sector needs
  - Not just client-server anymore!

# Are they for your project?

- Several questions arise, particularly for legacy databases:
  - How difficult is it to move to a different schema and what are the advantages?
    - Do the queries run faster?
    - Are resources optimized?
  - What are the disadvantages to use of spatial schemas?
  - How long would a conversion take?
    - Expertise? Cost?

# Astronomy Project

- Working with Tom McGlynn/668
  - Several big problems facing Astronomy
- Developed a study focus area
  - Spatial query performance
  - Scaling from thousands of rows to millions of rows

1. What objects appear in both the A2 catalog and the XX catalog?
2. What objects appear in A2, XX, and YY catalog?
3. How does my performance degrade when the number of joins is increased?
4. Are there ways to organize the catalog table so that queries on non-spatial fields would be optimized?
5. What can be derived from a single catalog? What can be derived from multiple catalogs?

# Experiment Setup

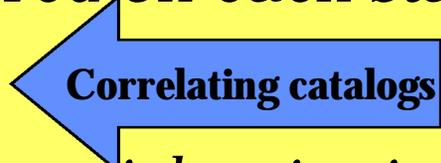
- USNO A2 (Monet Catalog) - 500 million objects on 11 CDs
  - Since we have three schema configurations to test, we were only able to test 140,000 objects
- SUN UltraSparc 60, 512 MB memory, Solaris 2.6 and 24 GB disk space
- INFORMIX Dynamic Server/Universal DataServer Option
  - Supports 2 flavors: relational and object-relational
  - SYBASE and ORACLE provide similar extensions

# Three Schemas Used

- Three schemas were developed using readily available indexes
  - *Relational only*
    - Btree, indexed on dec-RA and RA-dec
  - *Object-relational schema with commercial index*
    - Datatypes from the Geodetic DataBlade (product created for the EOSDIS project for geo-location)
  - *Object-relational schema with sample index*
    - Datatypes from Shapes2 R-tree datablade; sample supplied with Informix



# Four Types of Queries Tested

- **Spatial Window** - *Find all stars within a user-defined bounding rectangle*
- **Spatial Or-Windows** - *Find all the stars within at least one of two different bounding rectangles* 
- **Spatial Self-joins** - *Find all stars that are within a specified box centered on each star* 
- **Spatial Multi-join** queries: 
  - **Spatial chain-joins** - *Find a spatial region in chained tables*
  - **Star-join** - *Find a spatial region in “star” of tables*

# Queries

```
SELECT count(*), avg(field)
FROM <table>
WHERE dec >= x1 and RA >= y1
AND dec <= x2 and RA <=y2
```

Traditional relational

```
SELECT count(*), avg(field)
FROM <table>
WHERE inside (<GeoPoint>, <GeoBox>)
```

Spatial query based on  
Geodetic DataBlade

```
SELECT count(*), avg(field)
FROM <table>
WHERE within (<MyPoint>, <MyBox>)
```

Spatial query based on  
Shapes2 DataBlade

# Query and Resource Use Results

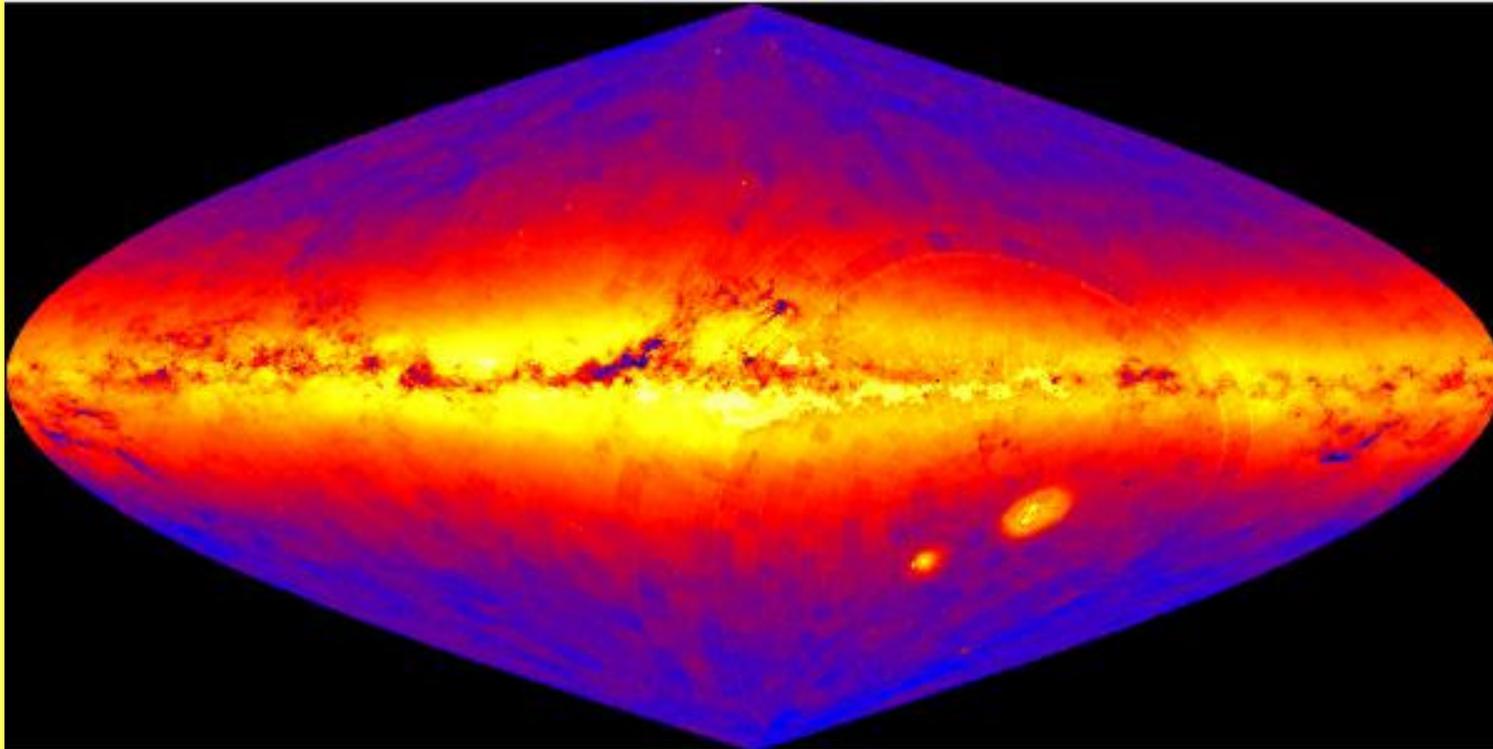


Image of the Monnet catalog found at:  
<http://www.nofs.navy.mil/projects/pmm/universe.html>

# Results from Loading Schemas

	140K rows (elapsed time to load in seconds)	500M rows (calculated load time in days)	B-tree Index (500M, est. days)	R-tree Index (500M, est. days)	Size in GB (500M objects)	Index Size in GB (500M objects)*
<b>Traditional (B)</b>	71.2	2.7	5.7		12.6	56 (8 ints)
<b>OR Schema (S)</b>	232.8	9.6	57.3	15.5	140	65.6, 77.3
<b>OR Schema (G)</b>	342.6	14.2	90.6	25.3	190.7	86.6, 104.8

- *Beware: Results are not for the faint of heart!*
- This winter, we decided to build our own DataBlade (Lightweight) based on R-tree index and use our space more efficiently. Results of the runs on the Lightweight aren't finished, but are much better than the commercially provided ones.

\*a,b= a is size of  
index on a point, b  
is index on a box

# Query Analysis

- Queries are performed on all 3 schemas and two choices of index clustering: dec-RA & RA-dec
- Queries performance measured with respect to:
  - Client elapsed time
  - Server CPU time
  - Number of buffer reads
  - Number of page reads
  - Per-table-size rankings were used with measurements to develop overall rankings

## Spatial Window Queries

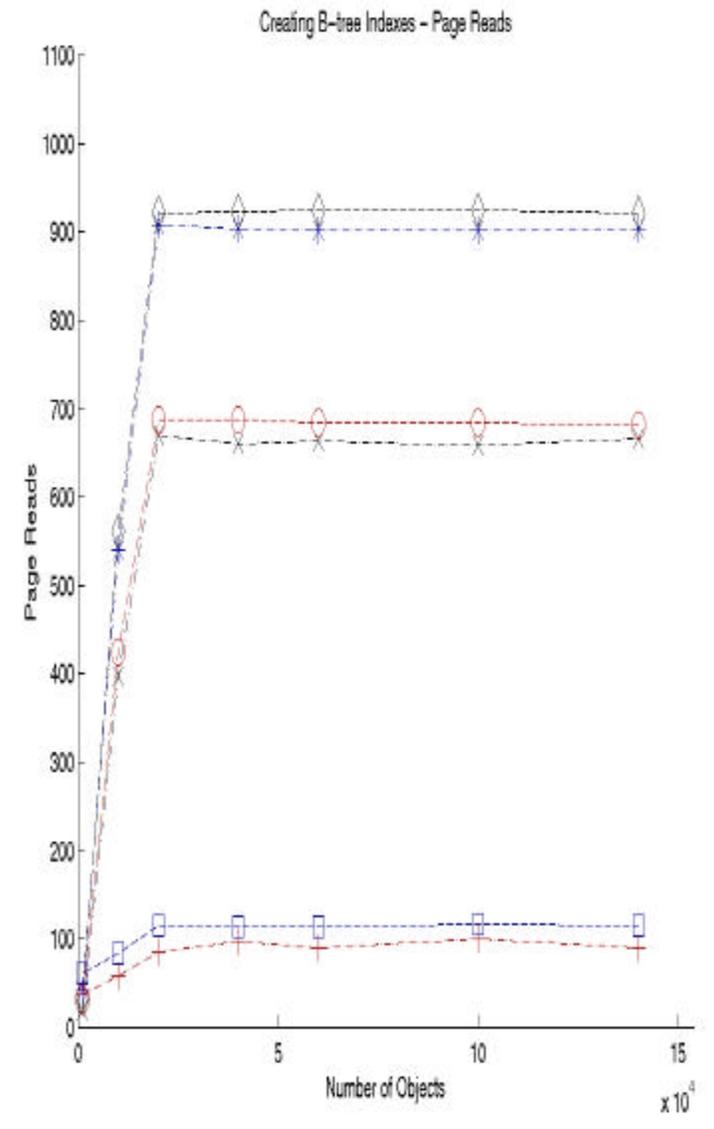
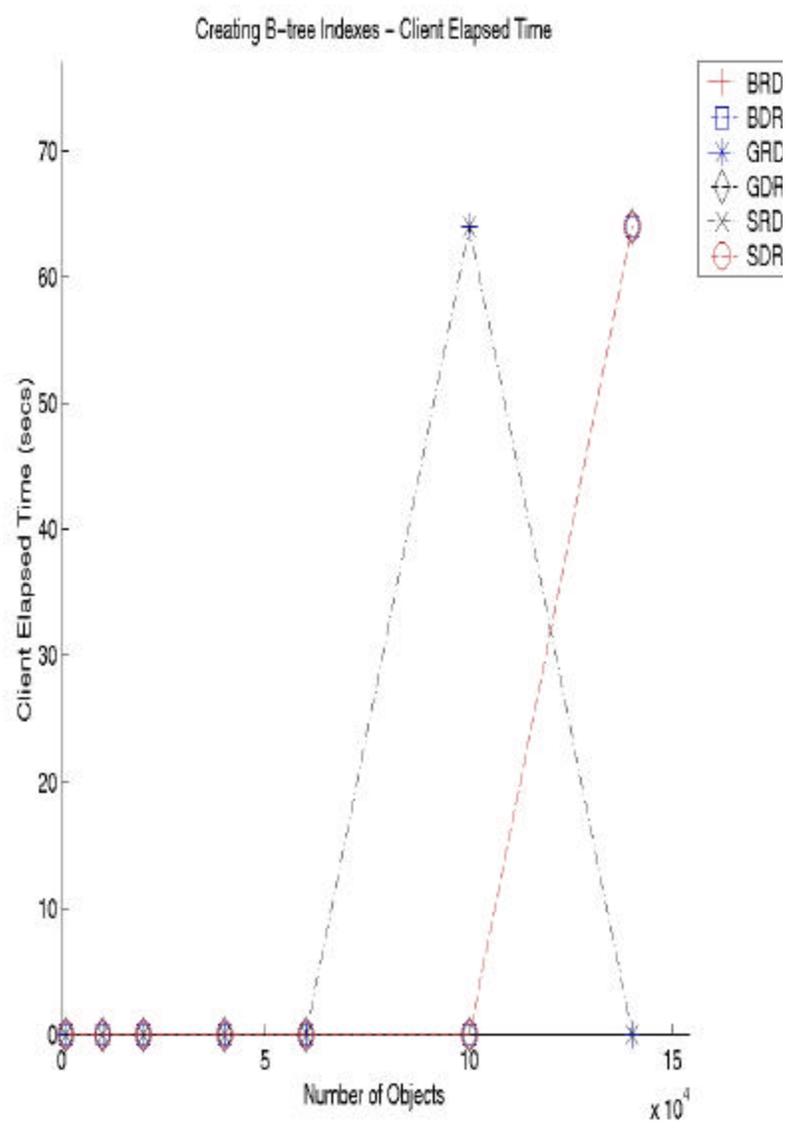
Rank	Client Elapsed Time	Buffer Reads	Page Reads
1	BDR BRD GDR GRD SDR SRD	GDR GRD SDR SRD	BRD
2		BRD	SRD
3		BDR	GDR
4			GRD
5			SDR
6			BDR

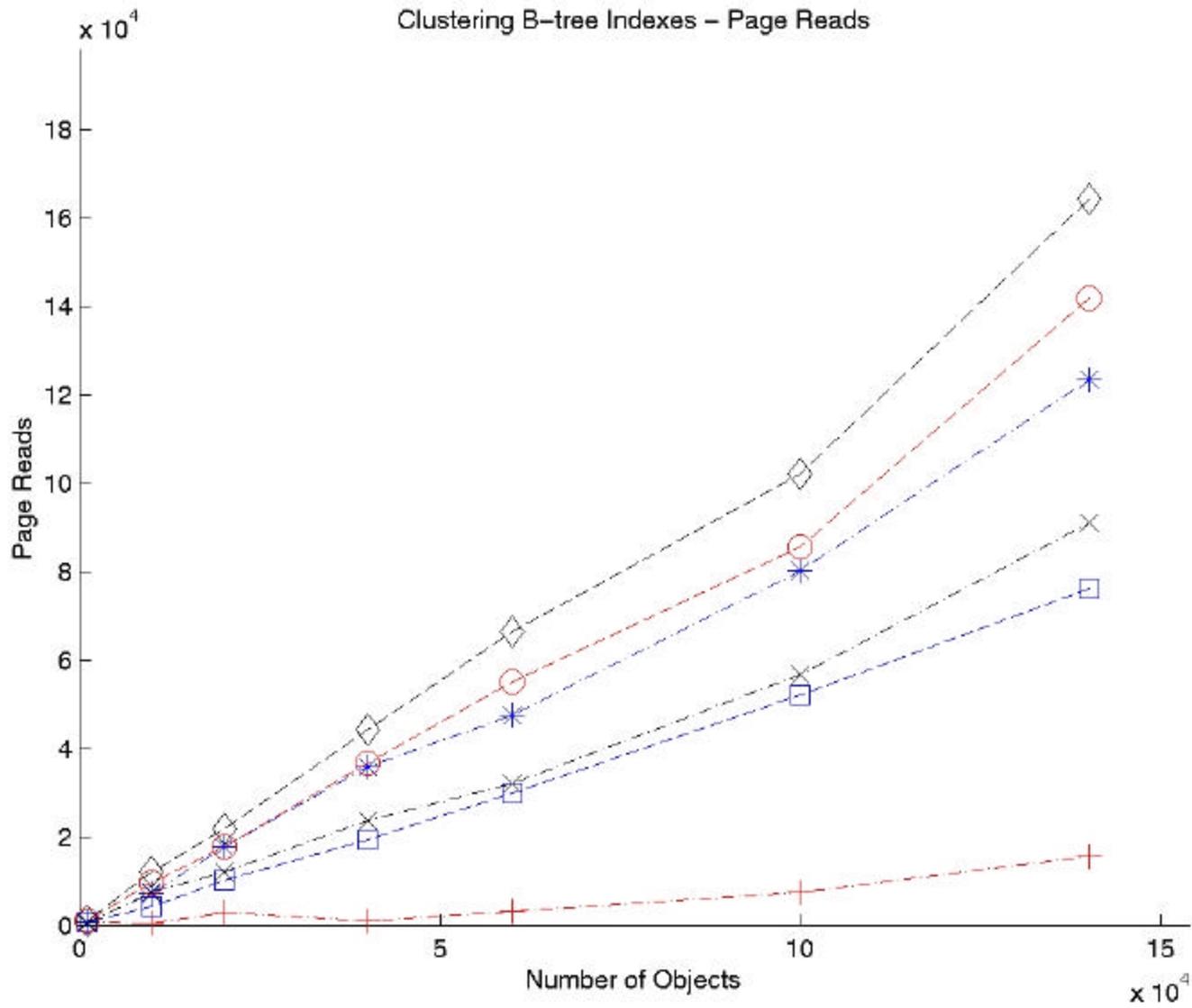
## Spatial Or-Window Queries

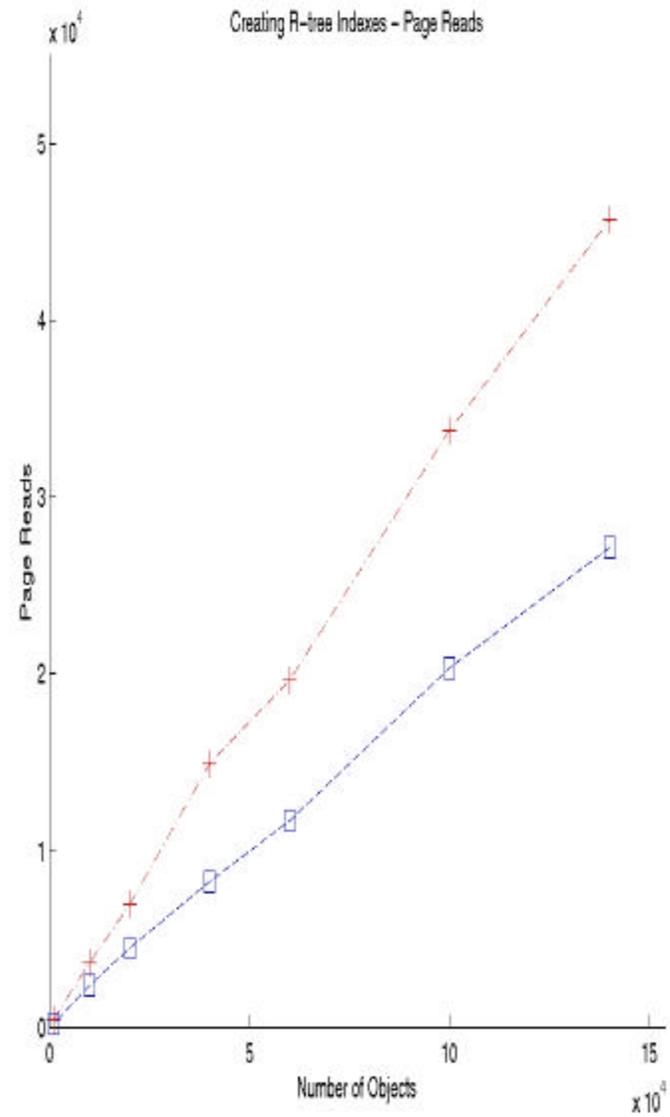
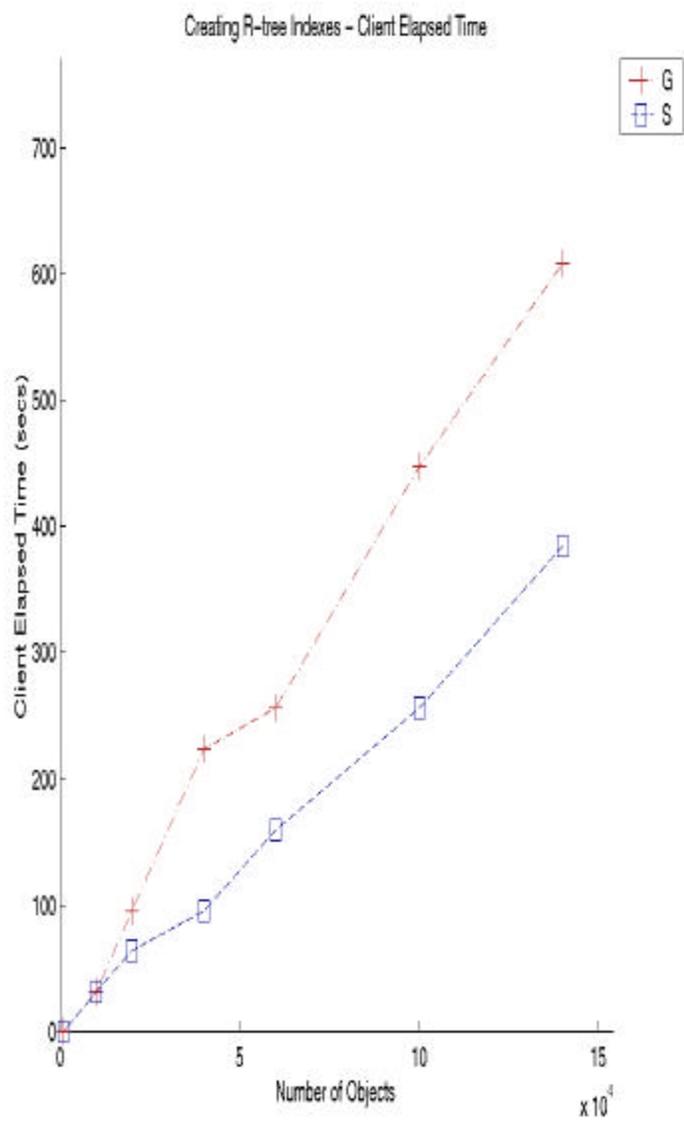
Rank	Client Elapsed Time	Buffer Reads	Page Reads
1	BDR BRD GDR GRD SDR SRD	GDR GRD SDR	SRD
2		SRD	BRD
3		BRD	BDR
4		BDR	GDR GRD
5			SDR

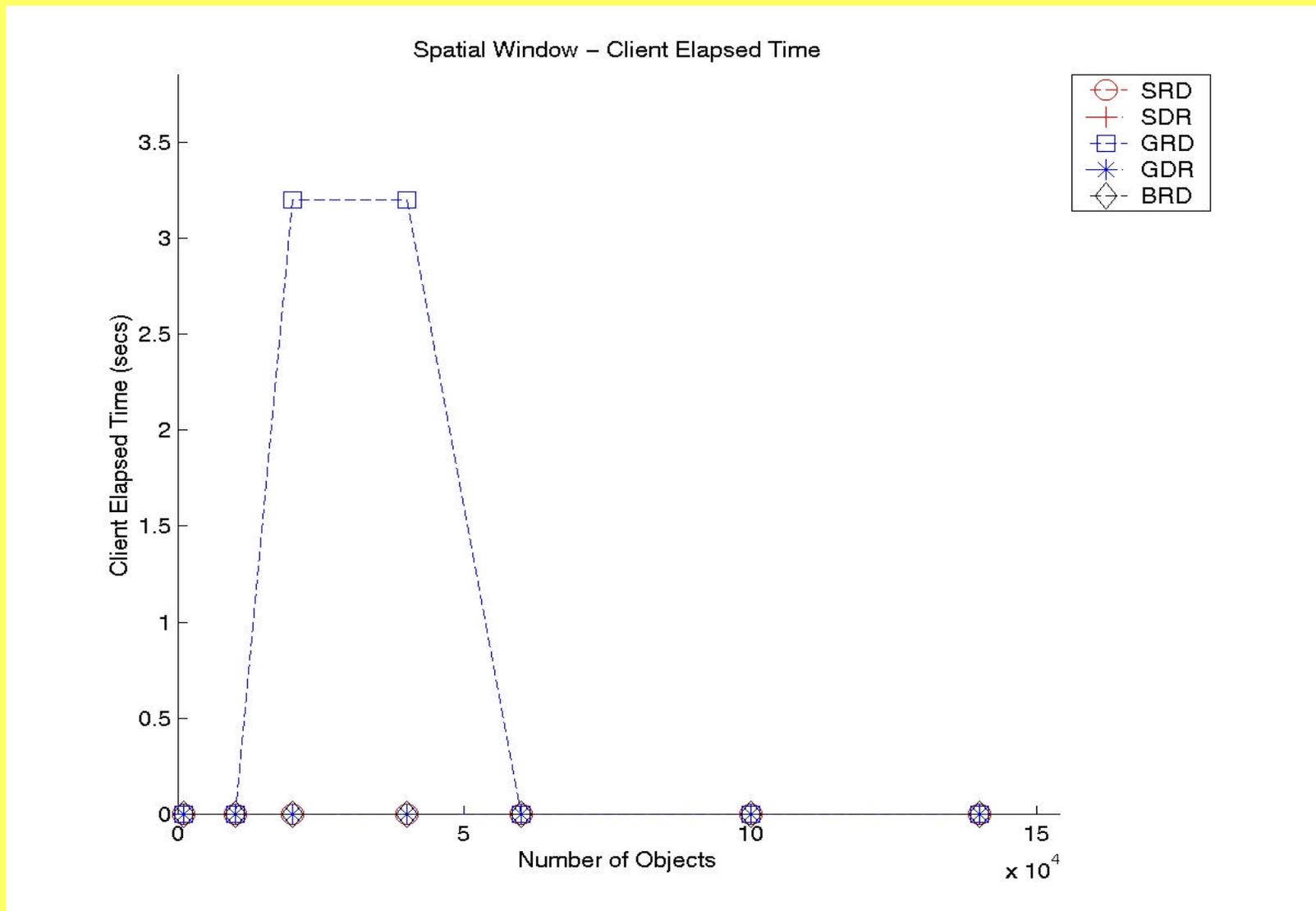
## Spatial Self-Join Queries

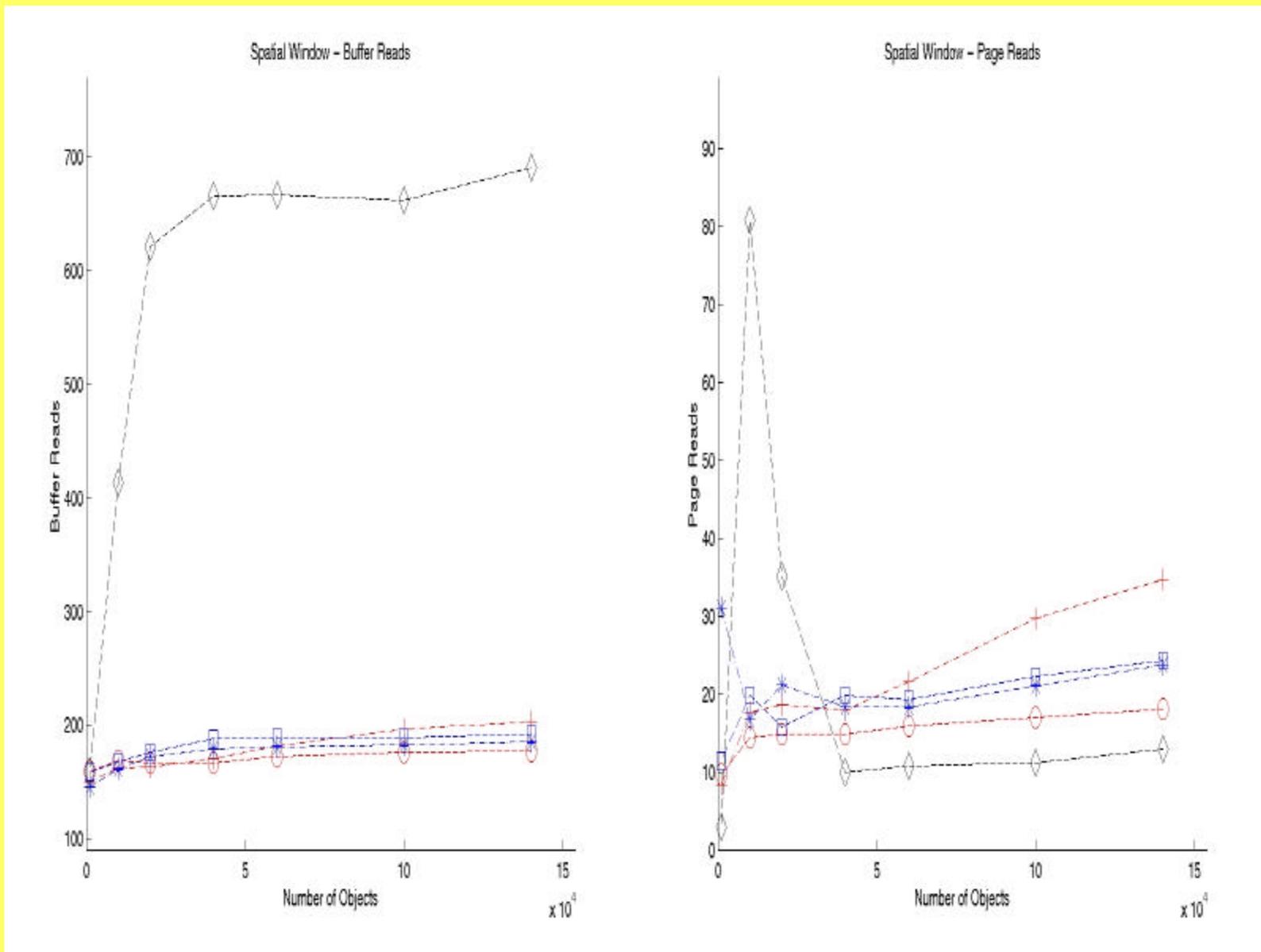
Rank	Client Elapsed Time	Buffer Reads	Page Reads
1	SRD	SDR	BRD
2	SDR	SRD	BDR SRD
3	GDR	GDR	GDR SDR
4	GRD	GRD	GRD
5	BRD	BRD	
6	BDR	BDR	

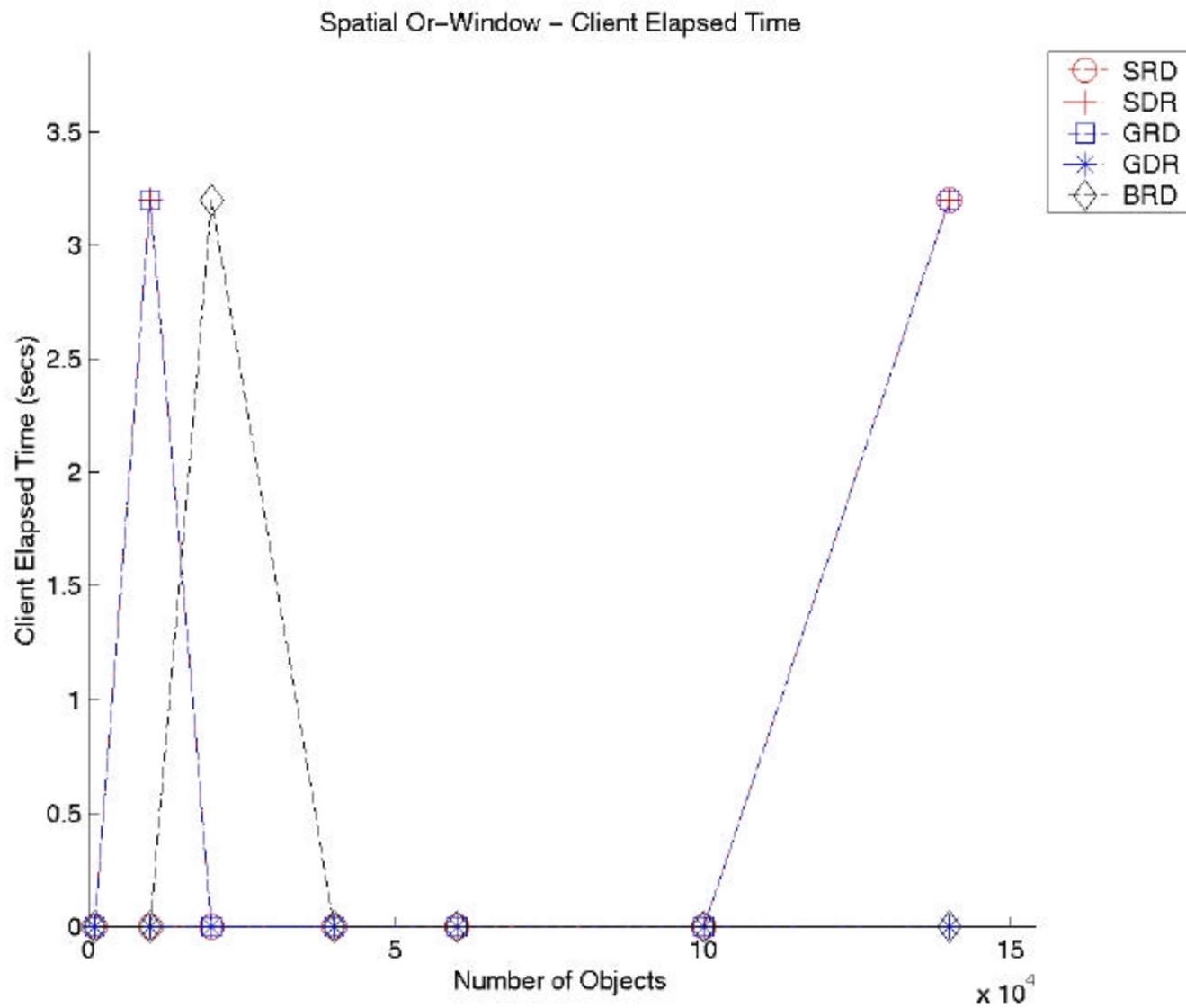


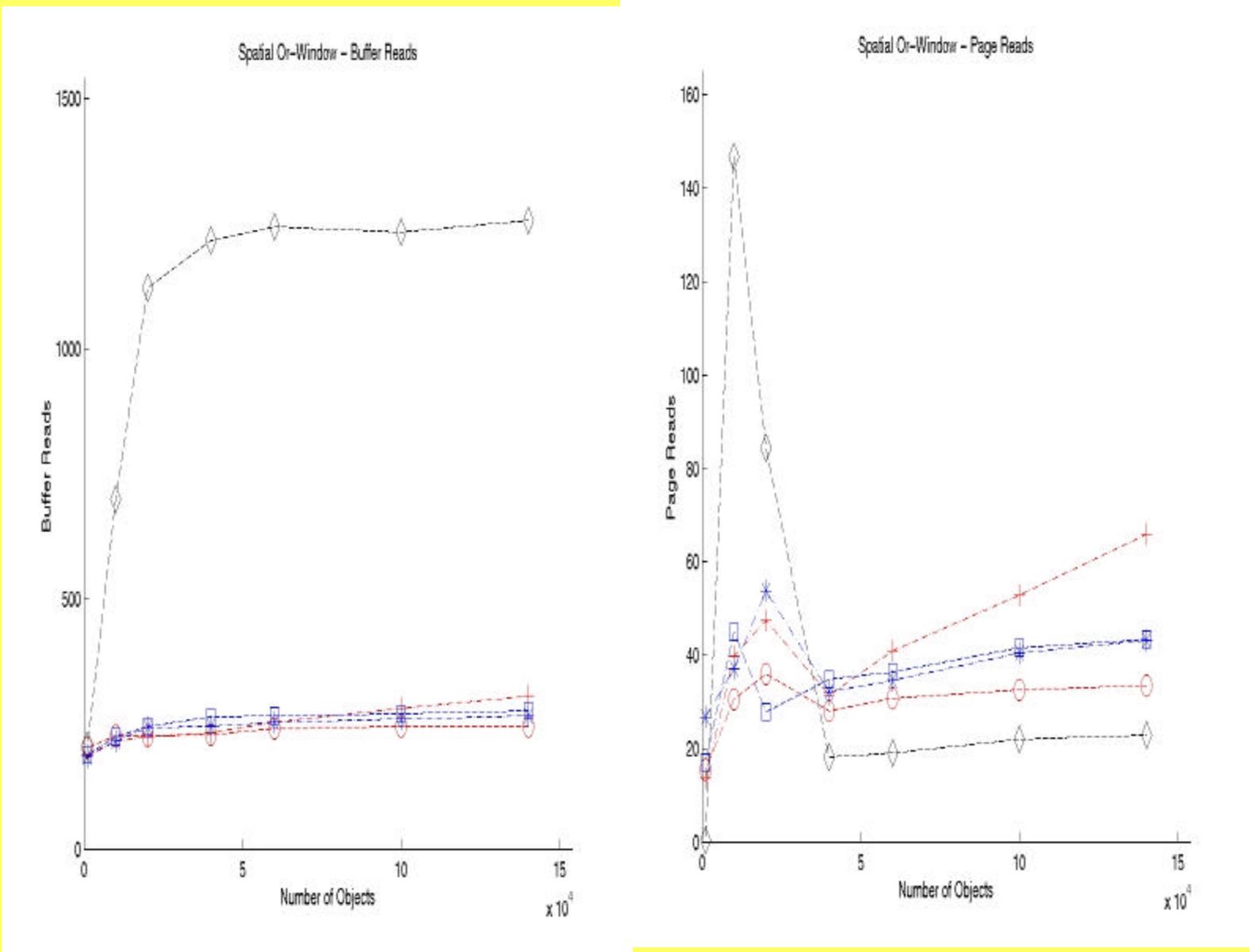


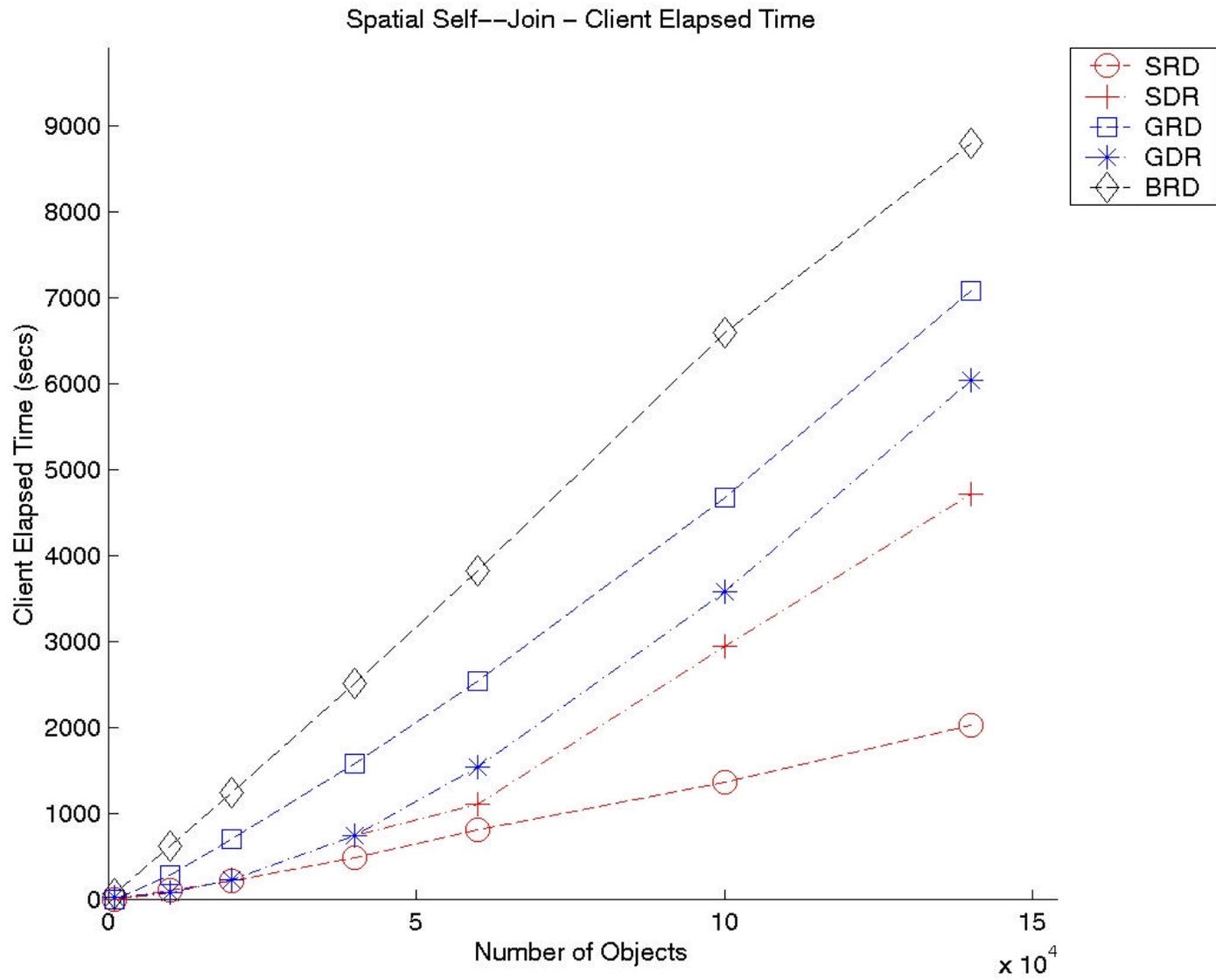


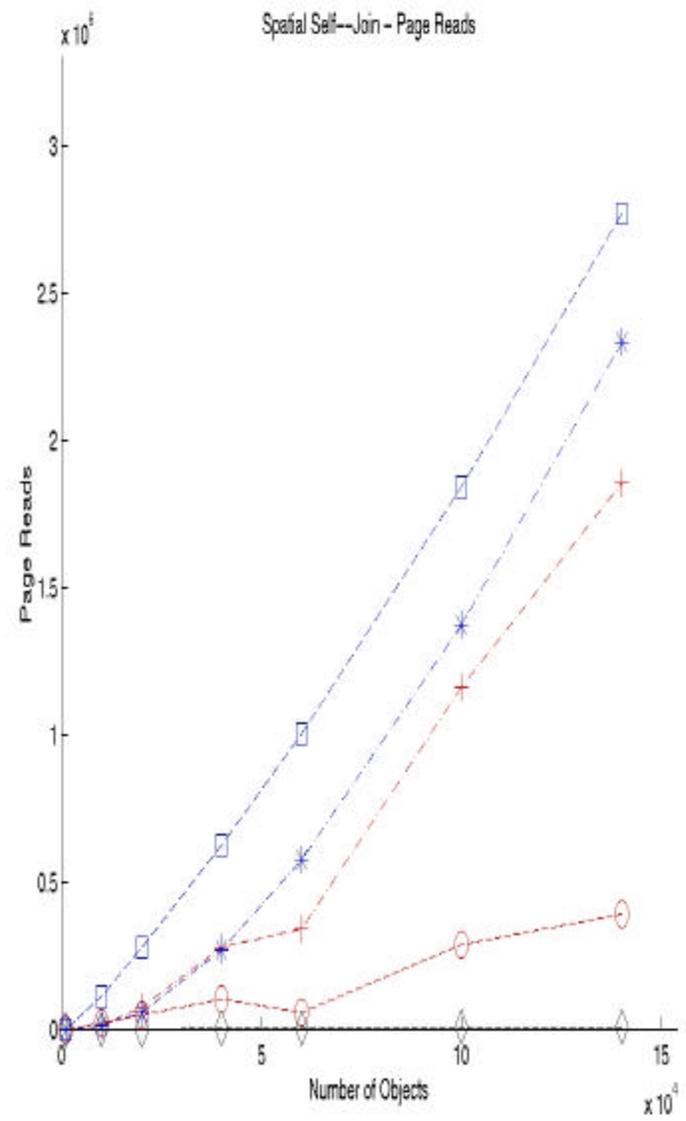
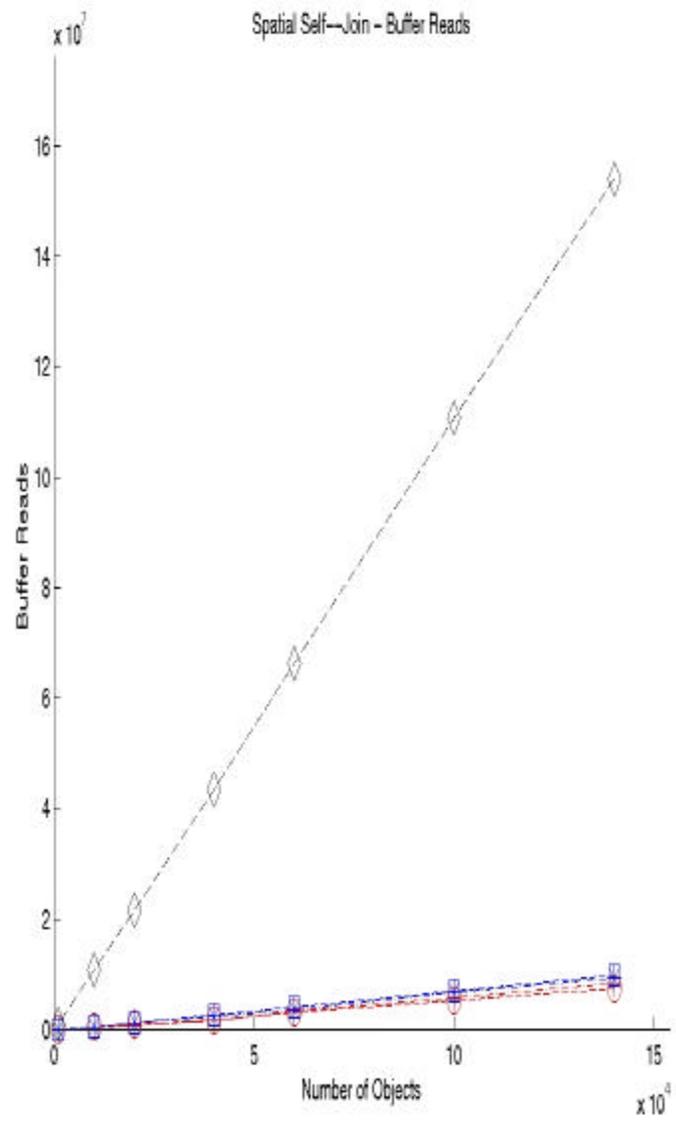


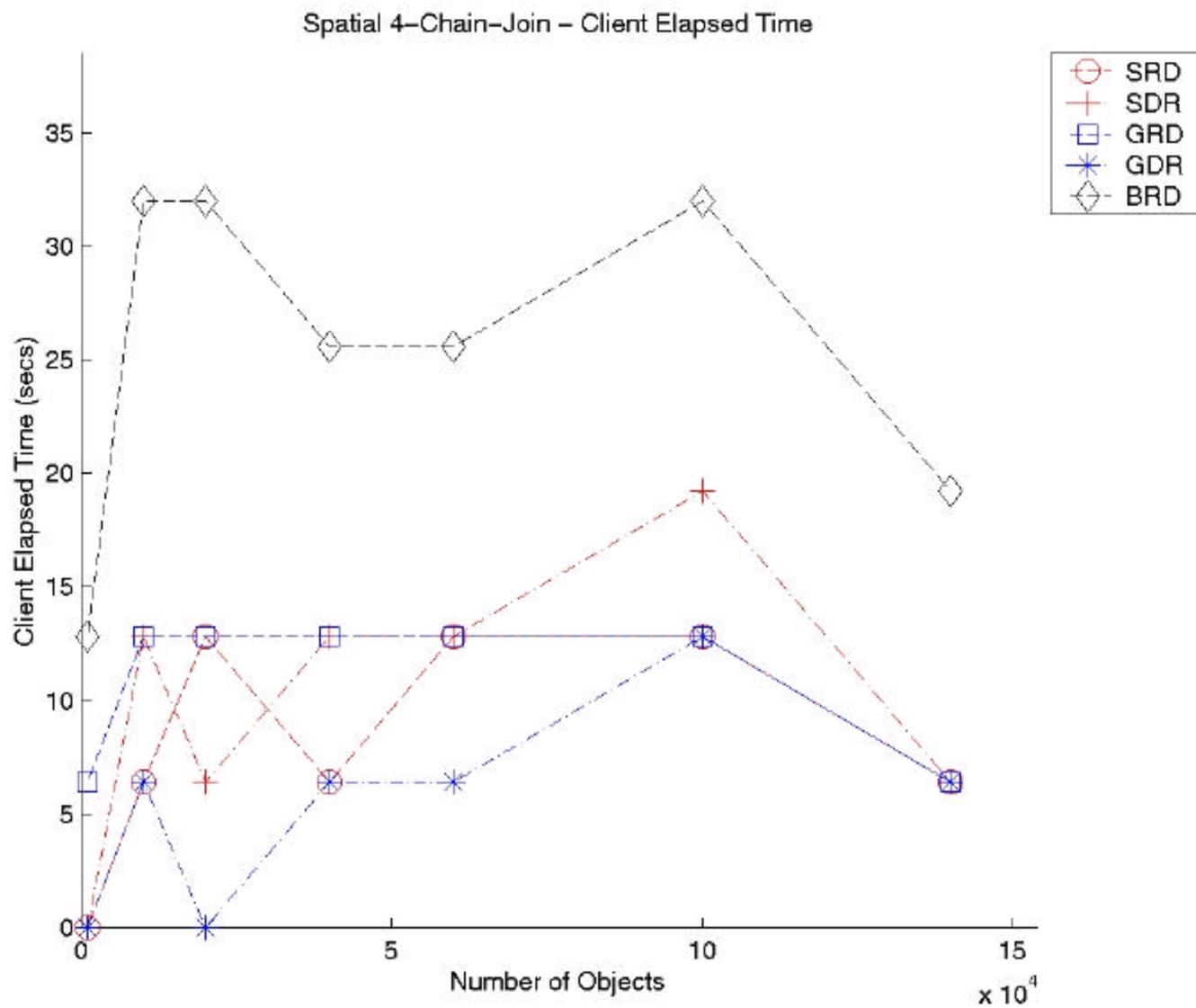


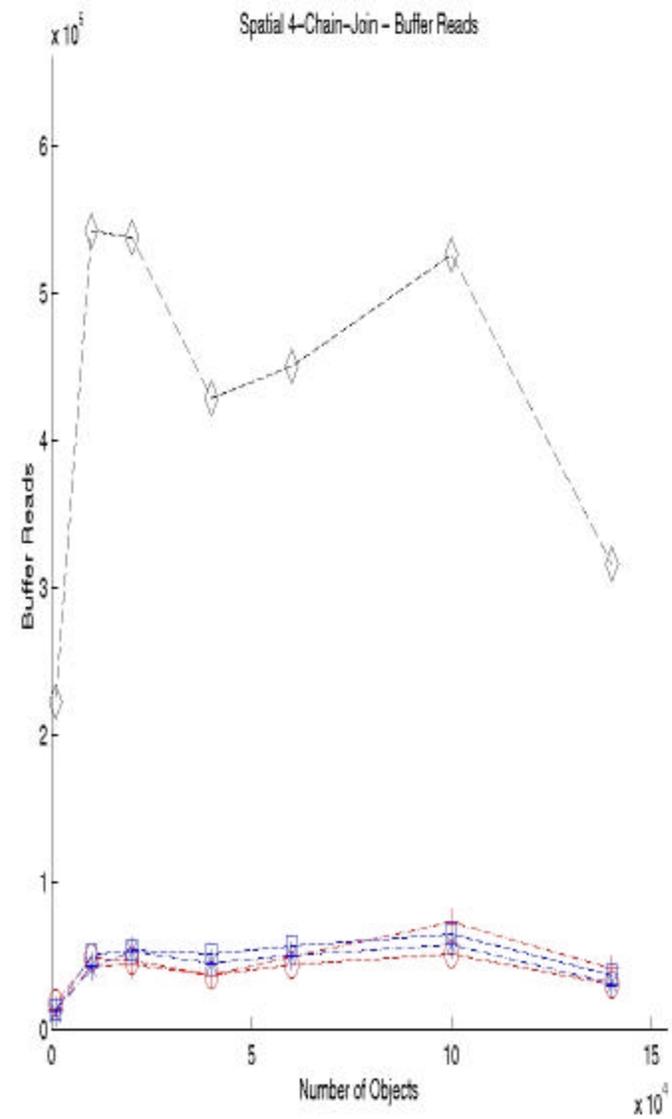
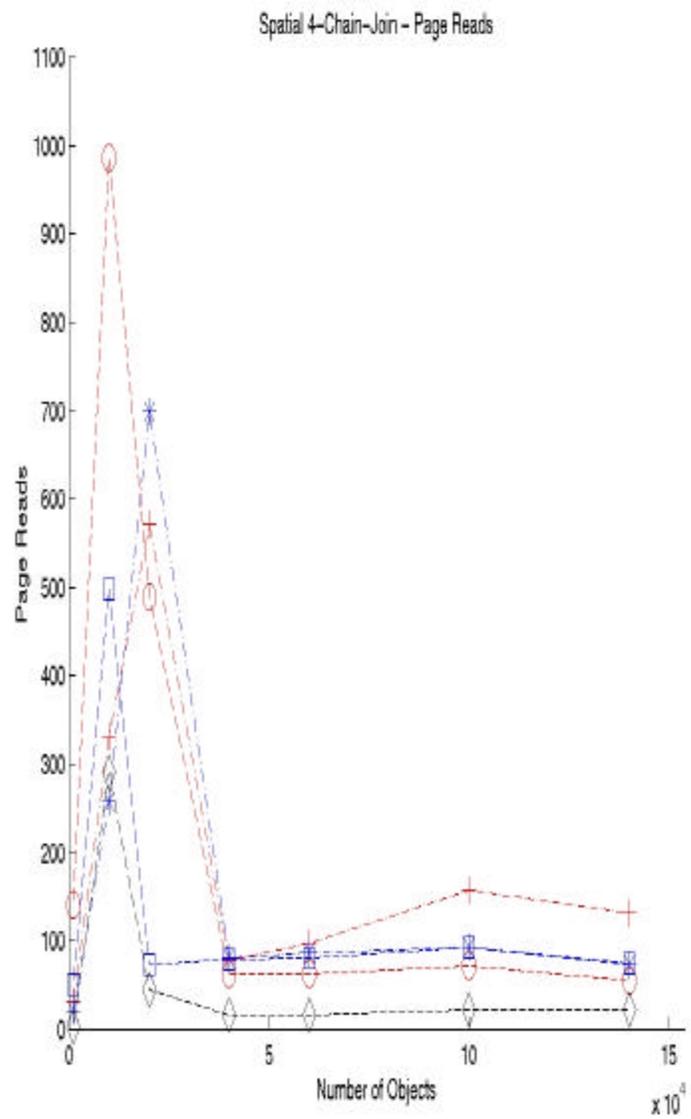


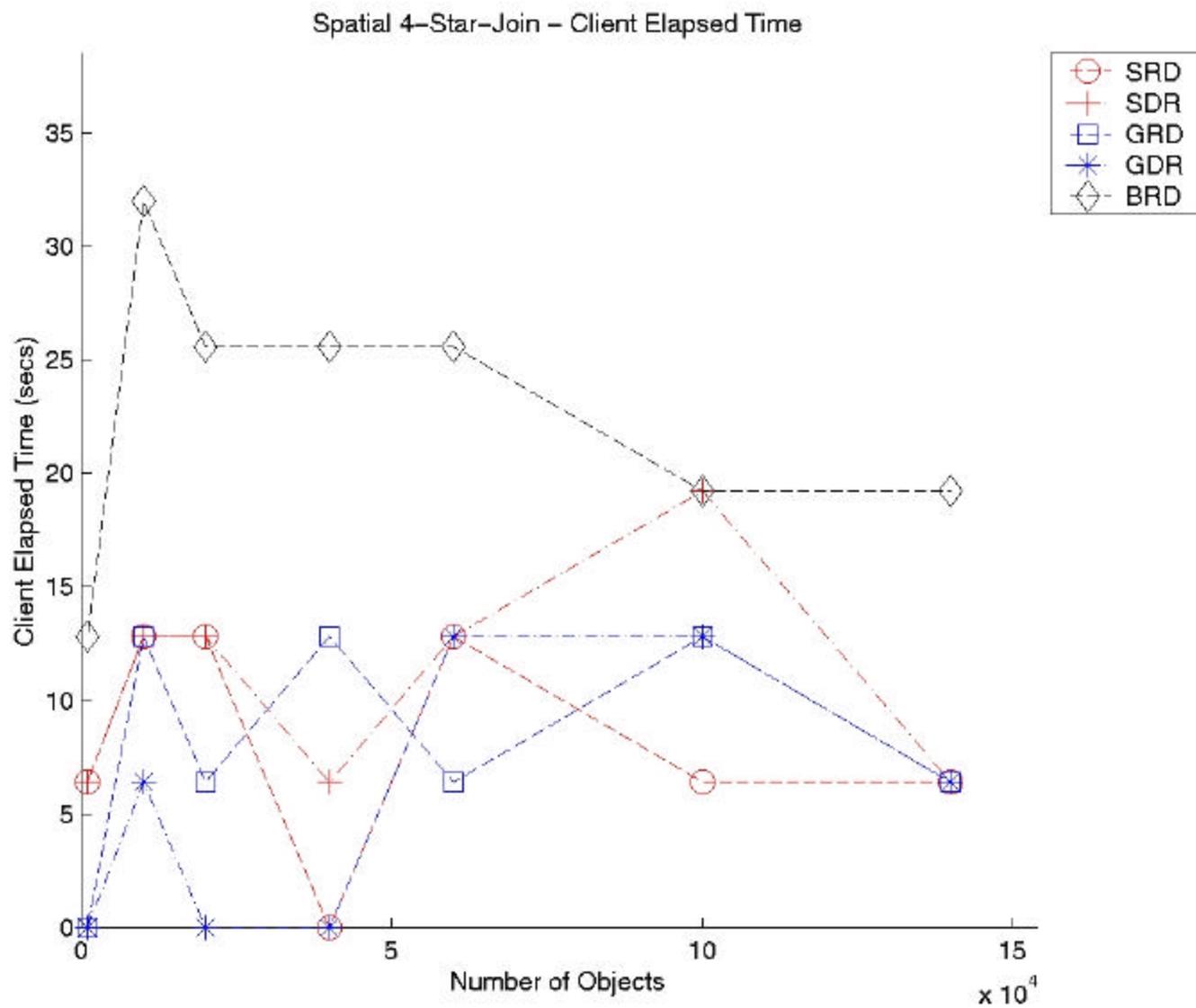


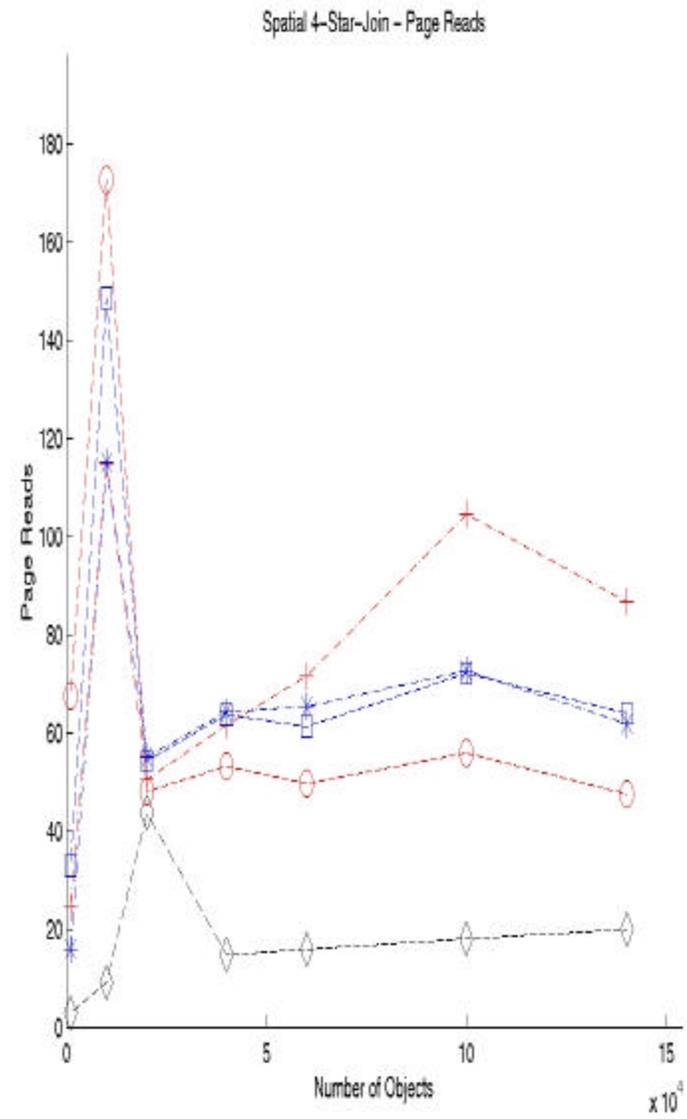
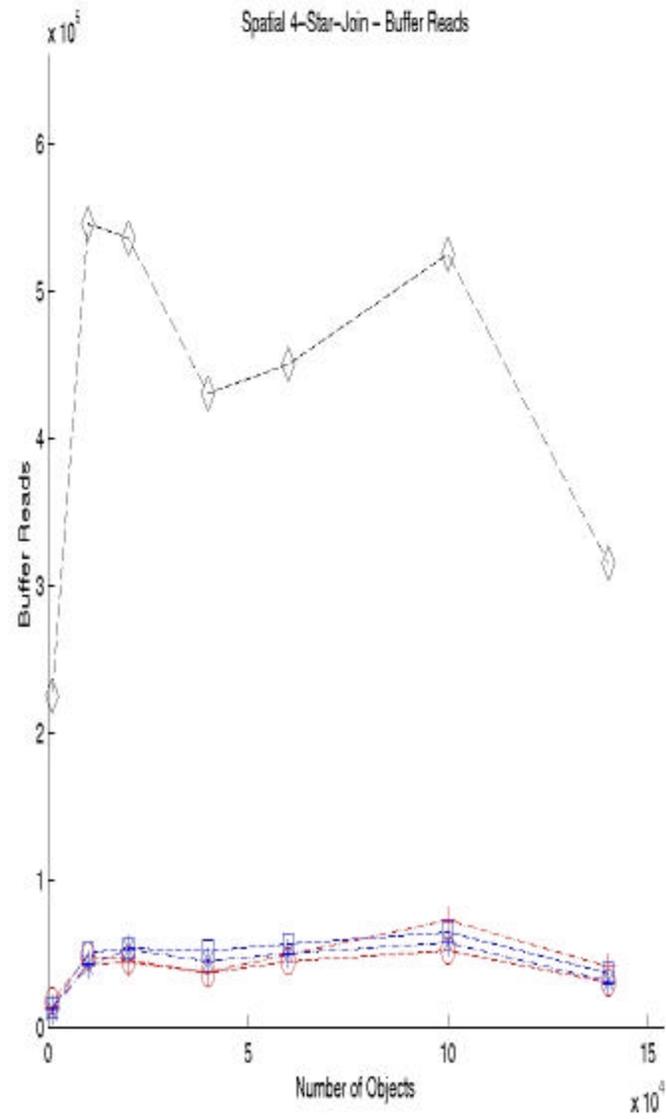




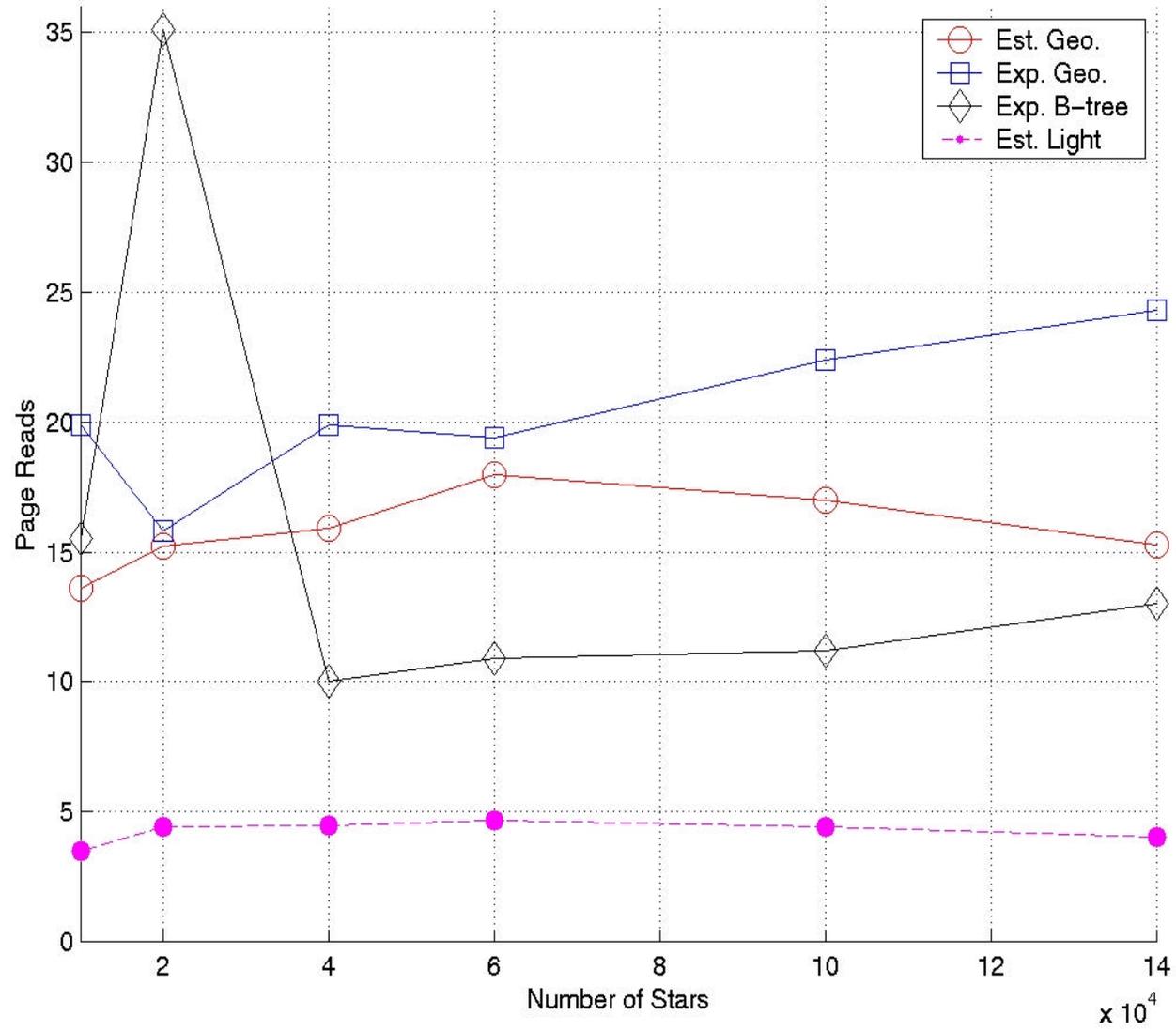


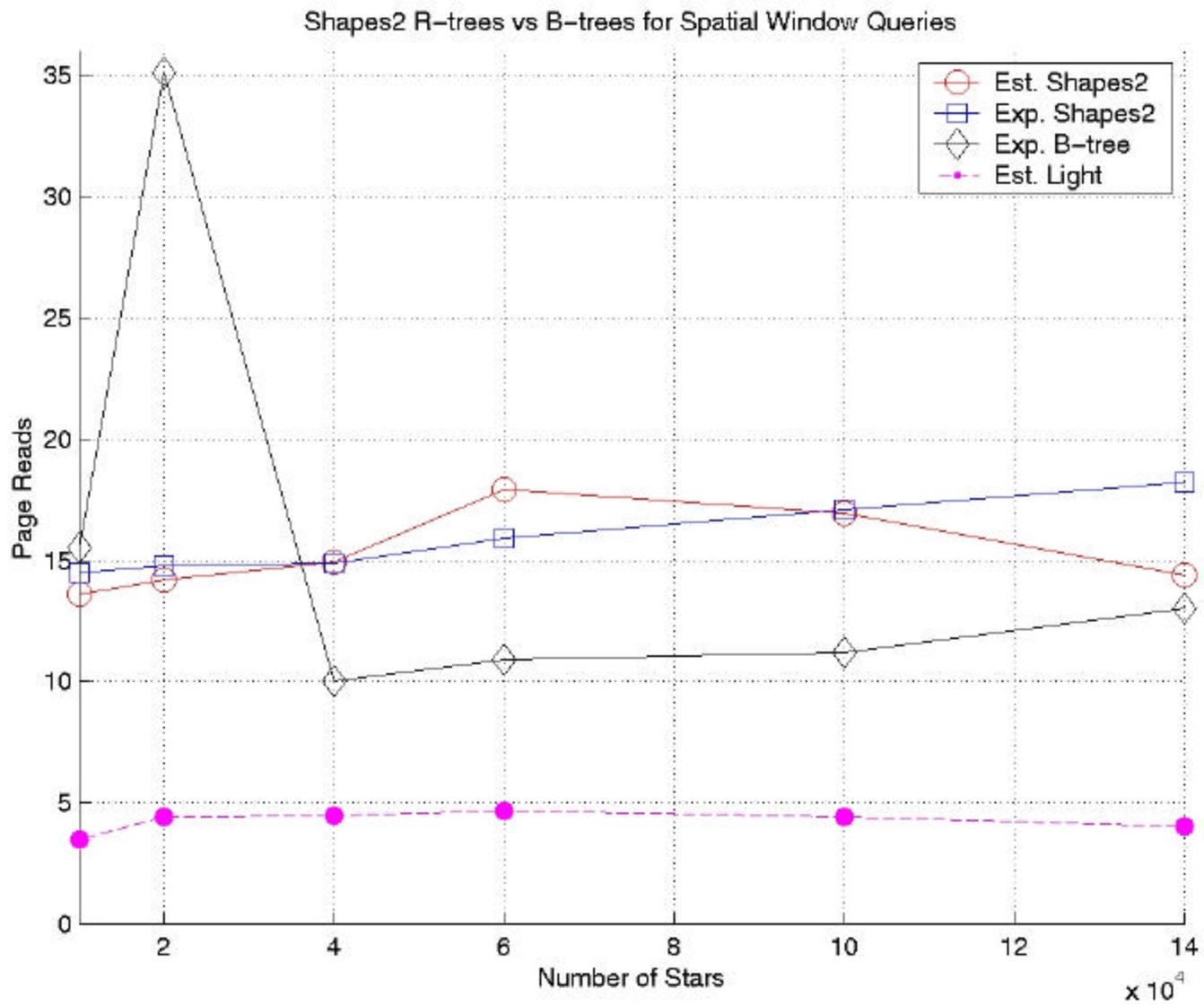


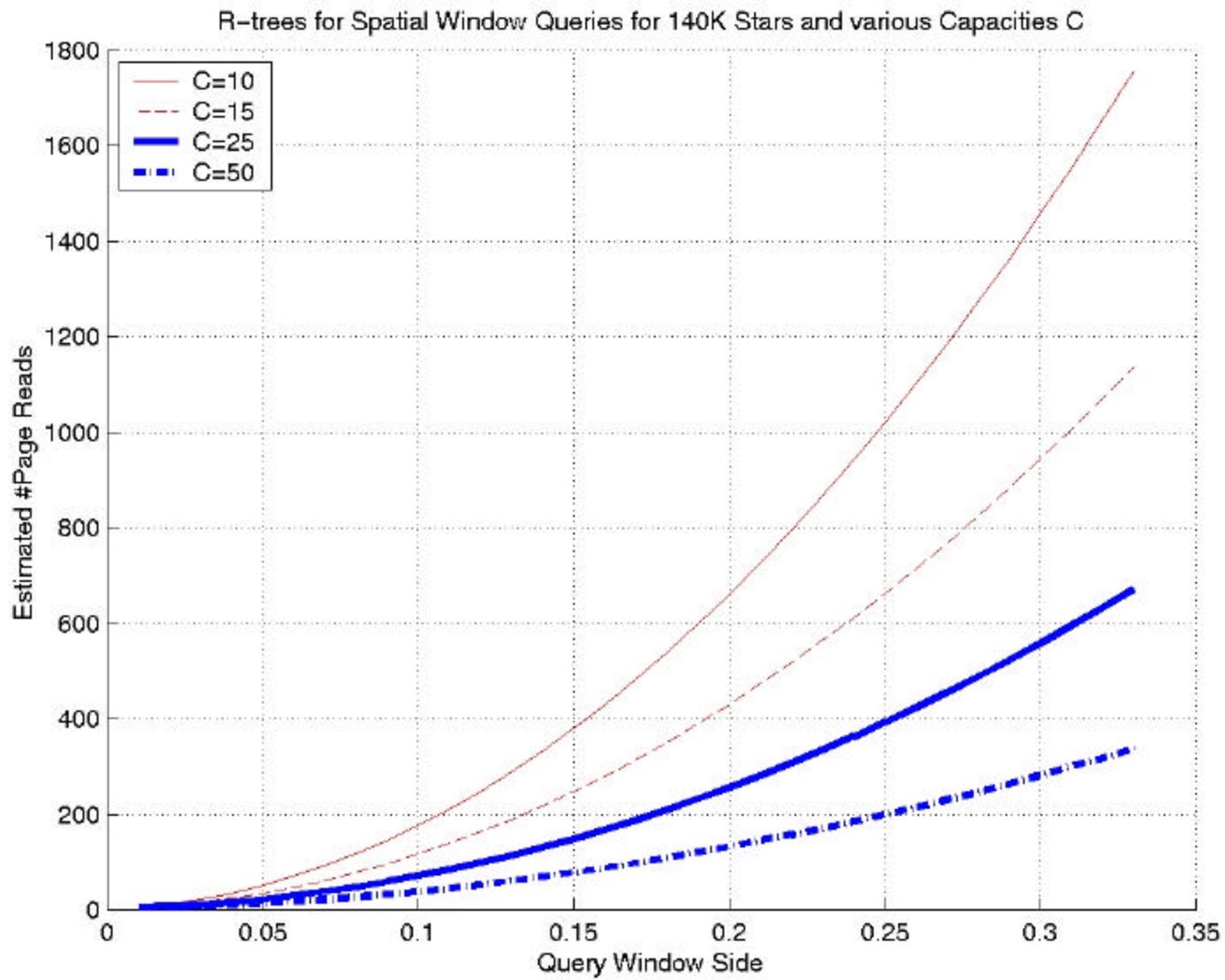


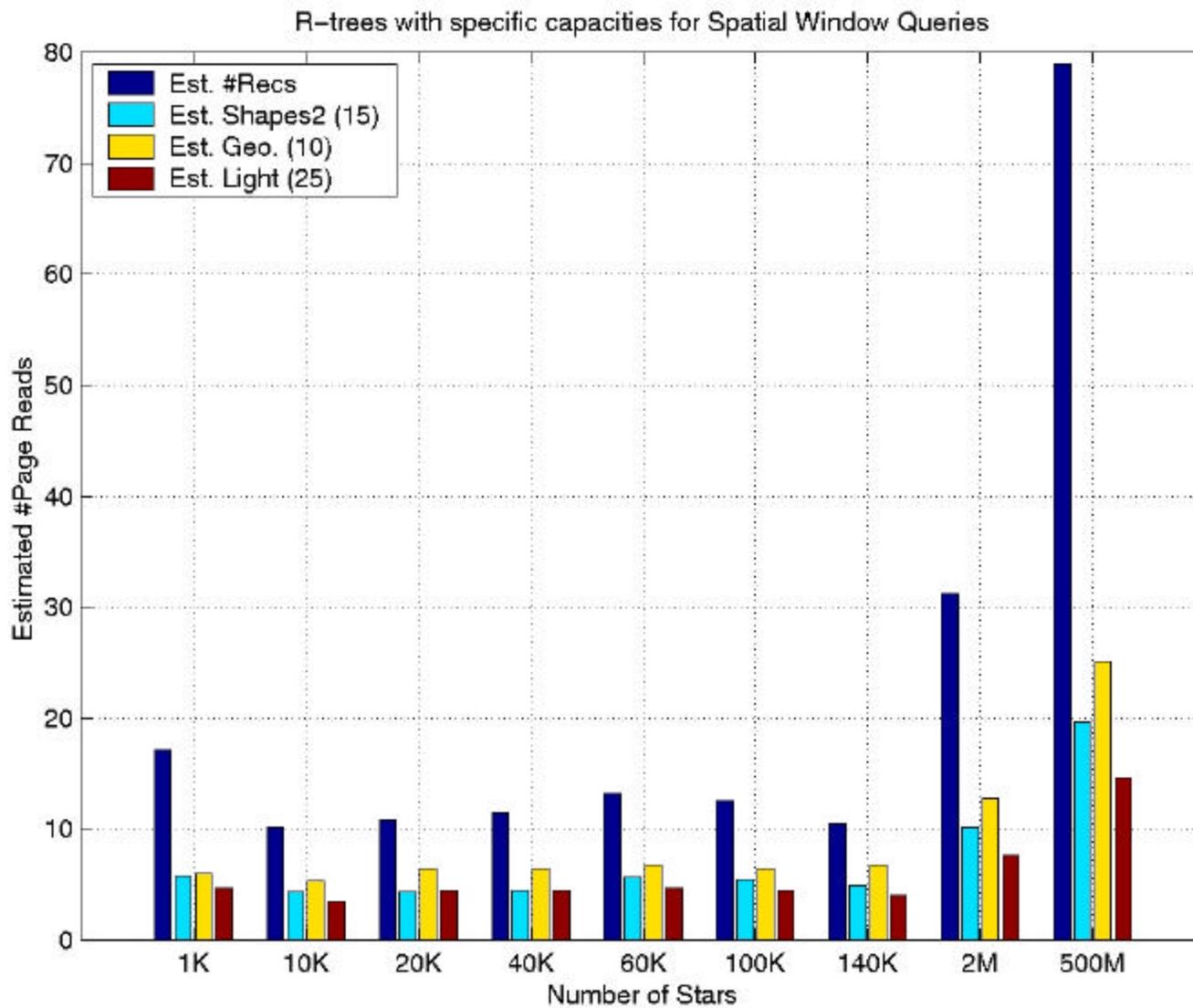


Geodetic R-trees vs B-trees for Spatial Window Queries









# Conclusions

*How difficult is it to move to a different schema & what are the advantages?*

- New schemas outperform the traditional on spatial joins (when you correlate catalogs) and self-joins
- A “lightweight” datablade and R-tree index provided by the new schemas are best
  - Incorporating smaller overhead reduces storage costs
- R-tree indexes are more forgiving than B-trees

# Conclusions

*What are the disadvantages to use of spatial schemas?*

- Traditional schemas (B-tree indexes) are probably adequate for spatial window queries with small windows
  - will show better performance if you can cluster your data to support specific queries
- Despite the advertising, Informix doesn't completely support its OR capabilities

# Conclusions

*How long would a conversion take?*

- *Expertise* needed was not elaborate:
  - Able to convert to several new schemas within months at UMBC; R-trees are no more difficult to make than B-trees
  - Lightweight DataBlade was created within a month
  - Using the OR features and spatial query language can reduce the amount and complexity of the application code
- *Costs* associated with ORDBMS/Spatial:
  - To support very large catalogs like Monet; can't be done efficiently on small workstations
  - Costs can be determined based on the function to be performed by the dbms
    - Spatial window queries are more efficiently done by a traditional DBMS
    - Advanced queries to support correlation and mining are more efficiently done by an OR-DBMS

# Data Mining Implications

- DM requires thoughtful data organization
  - Mining spatial data will be faster in ORDBMS
  - Plan to spend a long time with mining runs
  - Knowledge of the dataset is useful
- Allows us more time to focus on actual statistical functions that the DBMS can do very well
  - Hypothesis generation algorithm is critical
  - Parallel query processing is available
  - Plan to implement mining operators in the server
  - Datacube with small size is needed
  - Faster spatial-join algorithms are needed

# Data Mining Implications

- Improvements in loading the database and query speeds are adjustable with hardware
  - Expect improvements in software technology
- ORDBMS (R-trees) vs RDBMS (B-trees):
  - Keep spatial data records small
  - API to access R-tree internals is needed
  - Spatial self-joins on 140K items in ~33 vs ~145 mins
  - Spatial correlations with five 140K tables (chained or starred) ~3 times faster (~32 secs)
  - less sensitive to data distribution
  - more disk space
- Development effort is reasonable

# Future Work

- We are exploring use of IDL with the DBMS
  - Need algorithms that will that can be used in data mining studies
- Experimenting with recursive regression algorithms
- Looking into high-dimensional data (e.g. time-series)

# Further Discussion

- “Performance of Spatial Queries in Object-Relational Database Systems” - paper resulting from this work is available on request
- Email for further discussion
  - [jeanne.behnke@gsfc.nasa.gov](mailto:jeanne.behnke@gsfc.nasa.gov)
  - [kalpakis@cs.umbc.edu](mailto:kalpakis@cs.umbc.edu)
- This work is supported by the Information Systems Center/ISC
  - Code 586 & 587