Self-Tuning Database Systems: The AutoAdmin Experience

Surajit Chaudhuri
Data Management and Exploration Group
Microsoft Research
http://research.microsoft.com/users/surajitc
surajitc@microsoft.com

Research Group Overview

Data Management, Exploration and Mining Group

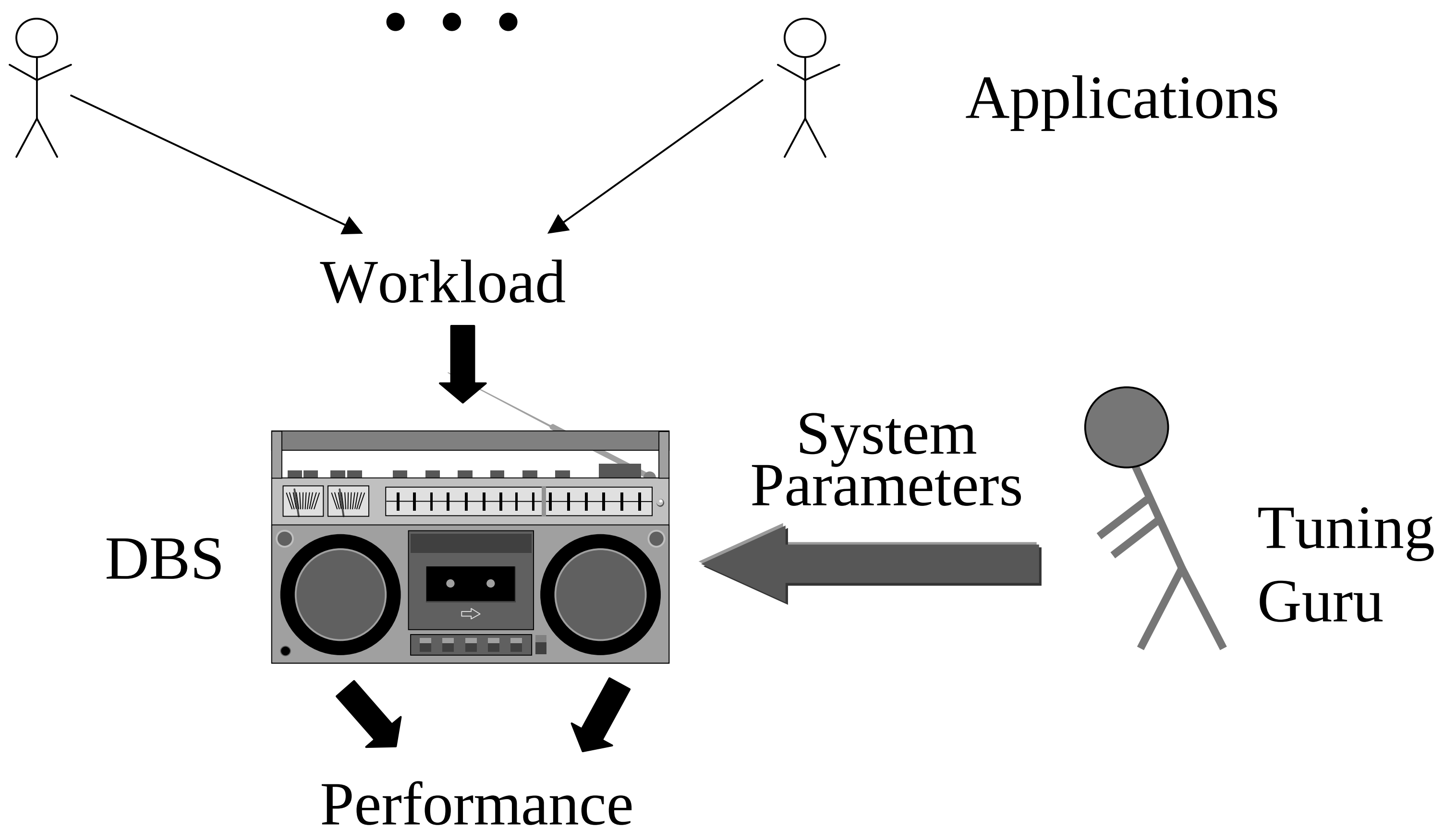
- Formed in 1999 by fusing two projects - AutoAdmin and DB support for DM
- Research with technology transfer
  - Project-oriented
  - Close partnership with SQL Server
- 6 researchers, 5 developers
  - A junior-heavy team
  - Strong internship program

Current Projects

- AutoAdmin: Self Tuning Database Systems
- Data Cleaning
- Exploratory Projects
  - Approximate Query Processing
  - Documents + Structured Data
  - XML2SQL
- Past project: SQL-aware Data Mining

Self-Tuning Database Systems: The AutoAdmin Experience

The Black Art of Database Tuning



Applications

Workload

DBS

System Parameters

Tuning Guru

Performance

AutoAdmin: Motivation

- Started in **summer 1996** at Microsoft Research – team of 2
- Our Goal:
    - Make database systems self-tuning and self administering
        - Analogy: Cars
    - Reduce TCO

Vision of a Self Tuning System

- Manager
  - Sets goals, policy, and the budget
  - System does the rest
- Everyone is a CIO
- Build a system
  - Used by millions of people each day
  - Administered and managed by a ½ time person
    - On hardware fault, order replacement part
    - On overload, order additional equipment
    - Upgrade hardware and software automatically

"What Next?
A dozen remaining IT problems"
Turing Award Lecture,
FCRC,
May 1999
Jim Gray
Microsoft

Physical Design Impacts Query Execution

SELECT Name
FROM Employees
WHERE Age < 40 AND Salary > 200K

Execution Plan A:
Filter (Age < 40 AND Salary > 200K)
Table Scan (Employees)

Execution Plan B:
Filter (Age < 40)
Table Lookup (Employees) by Salary

Effect of Workload on Physical Design

```
SELECT Name
FROM Employees
WHERE Age < 40
AND Salary > 200K
```

```
SELECT Name
FROM Employees
WHERE Age < 20
AND Salary > 50K
```

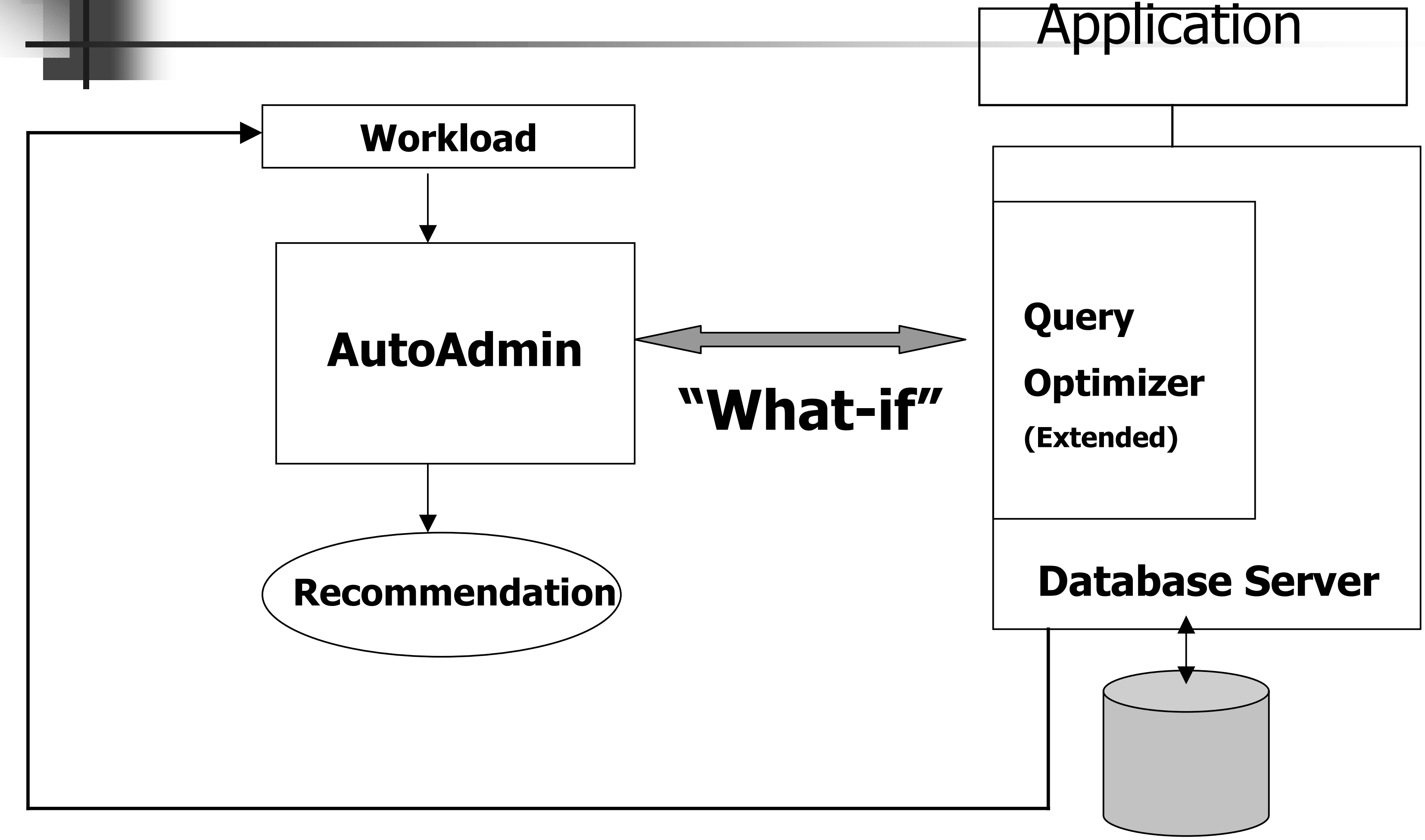
- Which column(s) should we index?
- Right answer may be:
  - Salary
  - Age
  - Both
  - Neither!
- Depends on the workload, and requires knowledge of statistics

AutoAdmin: Key Contributions

- A What-if architecture for exploring the space of hypothetical designs (SIGMOD 98)
- Workload driven
  - Integrated physical database design tool (VLDB 97, VLDB 00)
    - Recommends indexes and Materialized Views
    - Part of Microsoft SQL Server product since 1998
  - Statistics selection (ICDE 00, SIGMOD 02)
- Execution feedback driven statistics building (SIGMOD 99, SIGMOD 01)

“What-If” Architectures

“What-If” Architecture Overview



"What-If" Analysis of Physical Design

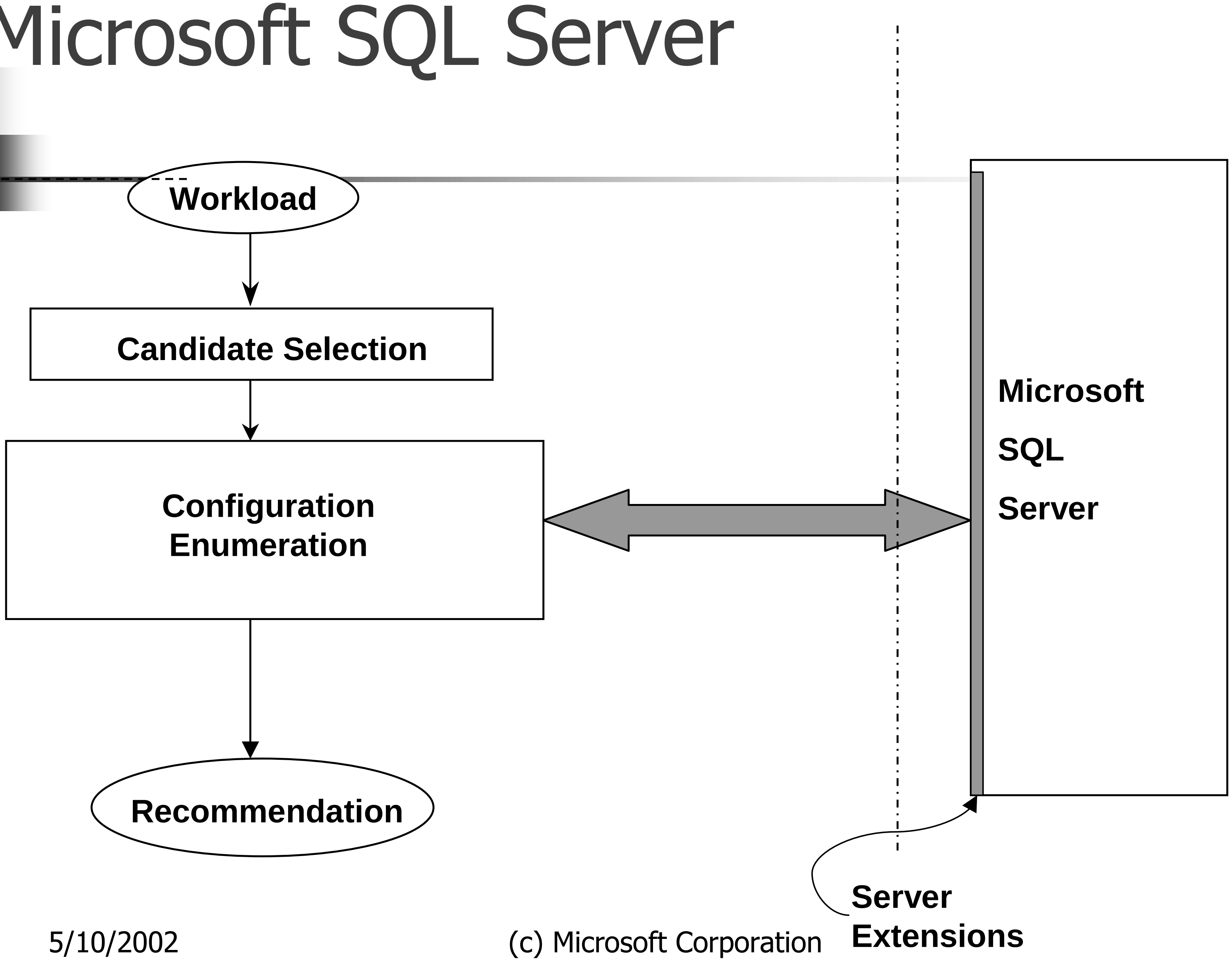
- Estimate quantitatively the impact of physical design on workload
  - e.g., if we add an index on T.c, which queries benefit and by how much?
- Without making actual changes to physical design
    - Time consuming
    - Resource intensive
- Search efficiently the space of hypothetical designs

Workload-driven Physical Design for Databases

Physical Database Design: Problem Statement

- **Workload**
  - queries and updates
- **Configuration**
  - A set of indexes, materialized views from a **search space**
  - **Cost** obtained by "what-if" realization of the configuration
- **Constraints**
  - Upper bound on storage space for indexes
- **Search**: Pick a configuration that is of "lowest" cost for the given database and workload (VLDB 1997)

Architecture of Tuning Wizard in Microsoft SQL Server



Search Space

- Large Search Space for indexes
  - Many columns to choose from
  - Kinds of indexes
- Explosive search space for materialized views
- Query optimizers use physical design in novel ways
- Physical design choices interact

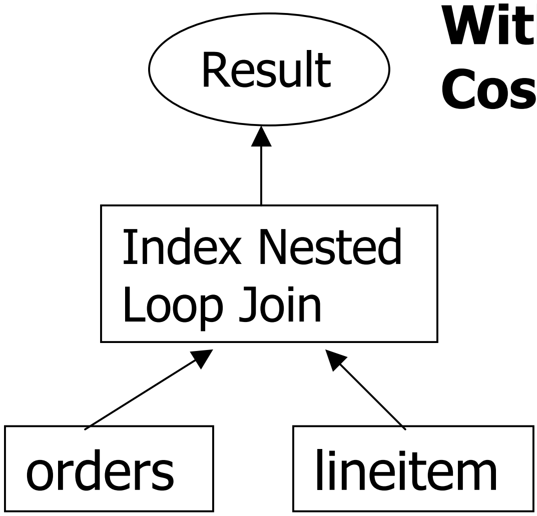
AutoAdmin Milestones

- Started in late summer 1996
- SQL Server 7.0: Ships index tuning wizard (1998)
- SQL Server 2000: Integrated recommendations for indexes and materialized Views
- Shared research results widely

Workload Driven Statistics Management

Example

- **SELECT** * **FROM** lineitem, orders
  WHERE l_orderkey = o_orderkey **AND**
  l_shipdate = '01-02-99' **AND** o_orderdate = '01-01-99'



Result

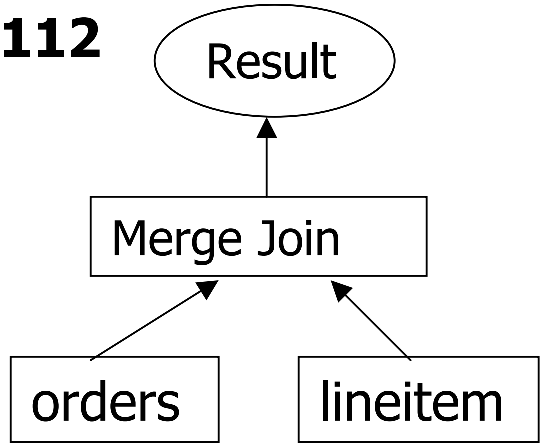
Index Nested Loop Join

orders

lineitem

With stats
Cost = 25

Without stats
Cost = 112



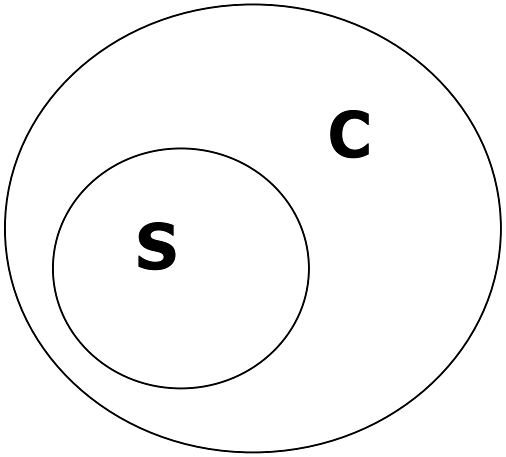
Result

Merge Join

orders

lineitem

Essential Set of Statistics



- "Chicken-and-egg" problem
  - Cannot tell if additional statistics are necessary until we actually build them!
  - Need a test for equivalence *without* having to build any statistics in (**C** – **S**)

Example

- **SELECT** E.EmployeeName, D.DeptName
  FROM Employees E, Department D
  WHERE E.DeptId = D.DeptID
  AND E.Age < 40 **AND** E.Salary > 200K
- Statistics on E.Age are missing
- May not need statistics on E.Age if predicate E.Salary > 200K is very selective

Essential Statistics (IEEE ICDE 2000)

- In the absence of statistics:
  - Query Optimizers use “magic numbers” for selectivity of predicates
    - For Age < 40, assume selectivity = 0.30
  - Data distribution independent
- MNSA (Magic Number Sensitivity Analysis)
  - Set magic numbers to a few different values
  - If *varying* selectivity does not affect plan

  ⇒ additional statistics will not help
- Else

  ⇒ Select a “promising” statistics to build

Statistics on Queries

- Reduce optimizer error by building statistics on query expressions (SIT)
- A very promising idea
- Like materialized views – a manageability challenge
- Recent work from AutoAdmin (SIGMOD 2002)

Execution Feedback Driven Statistics Building

Self-Tuning Statistics

- Think Maps
  - Why care about maps for Greenland?
  - Need detailed maps for areas you visit
  - Make maps more detailed each time you visit
- Idea: Start with “uniformity” assumption
  - Progressively refine with execution feedback
  - Single and multidimensional histograms
  - SIGMOD 99, SIGMOD 2001

More on Self-Tuning Database Systems

- More at Microsoft
  - SQL Server 7.0 introduced several auto-tuning features
- IBM Almaden
  - Work by Mario and Shel
  - LEO at IBM ARC has similar goals as AutoAdmin

Rethinking Database Systems

Featurism hurts Self-Tuning

- Featurism has turned into a curse
  - Yet another indexing smart /join method/optimizer transformation added
- Abusing Extensibility
  - Eliminate all second-order optimizations
- Turning into black magic
  - Hard to abstract principles
  - Cannot educate next generation of engineers
    Performance is unpredictable
- Self-Tuning is difficult

Role Models

- Ex. 1: Aircraft with many subsystems (engine, fuselage, electrical control, etc.)
- Ex. 2: RISC hardware
- No single engineer understands entire system
- Local theories for individual subsystems and reasonable understanding of interactions
  - Few points of interaction with stable and narrow interfaces
  - Built-in system support for debugging subcomponents (incl. Performance tuning)

RISC Philosophy for DBMS

- Details in VLDB 2000 vision paper
- Package as components with simplified functionality
- Enforce
  - Layered approach
  - Strong limits on interaction (narrow APIs)
  - Multiple consumers for a component
- Components must have manageable complexity
- Encapsulation must include predictable performance and self-tuning
- Not a new idea – but an idea worth revisiting

Final Words

- DBMS has to be self-tuning to be a good software component
- AutoAdmin
  - Exploit workload and execution feedback richly for enabling self-tuning
  - Demonstrated through technology incorporated in Microsoft SQL Server
- Despite advances, self-tuning remains a very formidable challenge
  - Need to think “self-tuning” globally by paying attention “locally”
  - RISC DBMS architectures – worth revisiting?

More Information

- Data Management, Exploration and Mining Group Homepage
  - http://research.microsoft.com/dmx
- Microsoft SQL Server White papers on Self-Tuning technology
- My contacts
  - http://research.microsoft.com/users/surajitc
  - surajitc@microsoft.com