

Implementation of an Agent Architecture for Automated Index Tuning

Rogério Luís de Carvalho Costa

Sérgio Lifschitz

Maíra Ferreira de Noronha

Marcos Antonio Vaz Salles

Departamento de Informática

PUC-Rio – Rio de Janeiro - Brazil

{rogcosta, sergio, maira, mvsalles}@inf.puc-rio.br

Agenda

- ◆ Index Tuning
- ◆ Our Goals and Approach
- ◆ Index Self-tuning Heuristics
- ◆ Agent based Self-tuning
- ◆ Implementation Issues
- ◆ Conclusions and Future Work

Index Tuning

- ◆ Uses of indexes
 - Queries
 - Updates
- ◆ Index tuning: all commands count
- ◆ Previous work
 - Focus on algorithms for index selection
 - Provides tools for DBAs

Autonomic Index Tuning Goals

◆ General Questions:

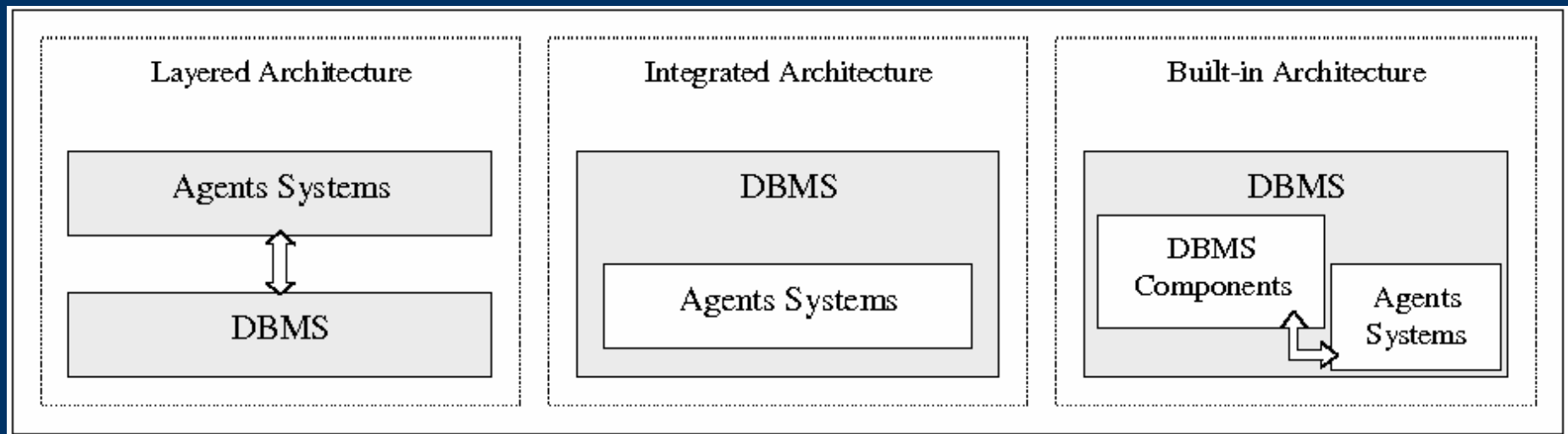
- Investigate architectures for the integration of self-tuning strategies with available DBMS technology
- Achieve clean implementations with good performance

◆ Specific Questions:

- Find a feasible index self-tuning strategy that does not require human intervention

Agents and DBMS

- ◆ Agents: adaptability and proactivity
- ◆ DBMS: performance and scalability



Index Self-tuning based on Differences

- ◆ On-line heuristic
- ◆ Evaluate commands as they are submitted
- ◆ Estimate alternative indexing solutions
 - hypothetical indexes
- ◆ Adapt index design on-the-fly

Index Self-tuning based on Differences

- ◆ Query Evaluation Strategy:

Benefit of Hypothetical Index =

Cost of Query with Actual Indexes –

Cost of Query with Hypothetical Indexes;

Update Accumulated Benefit of Hypothetical Index;

If (Accumulated Benefit of Hypothetical Index >

Cost to Create Hypothetical Index)

Then

Reset Accumulated Benefit of Hypothetical Index;

Materialize Hypothetical Index;

End if;

Update Accumulated Benefit of Actual Indexes Used;

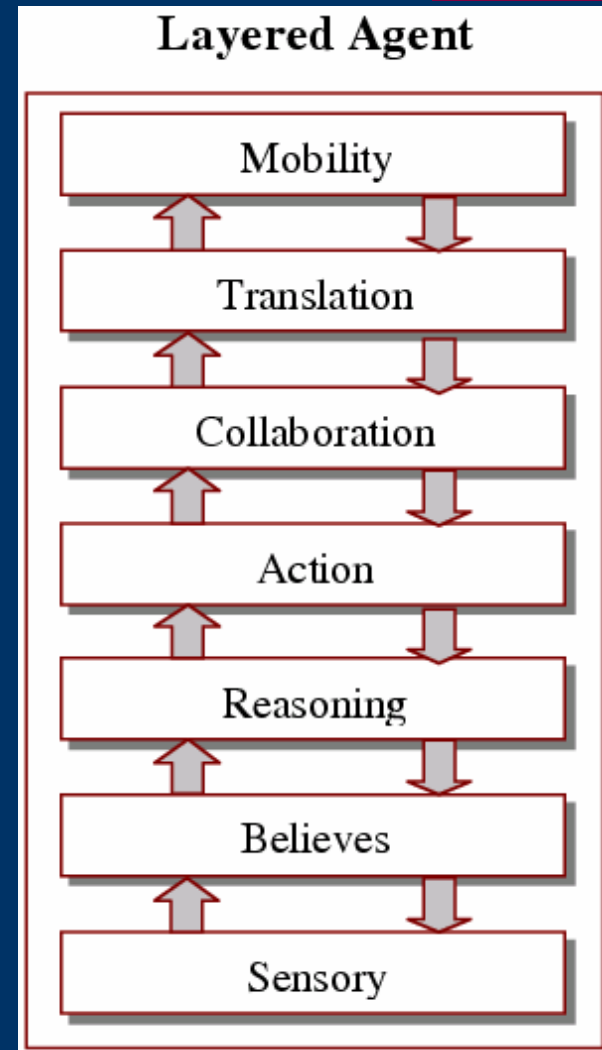
- ◆ Updates follow similar rules, but consider index destruction

Agent based Self-tuning

- ◆ Architectures
 - Agent architecture
 - Integration architecture
- ◆ Pros & Cons
 - *Intrusive, but well delimited*
 - *On-line: no human intervention, but limited time to search solution space*

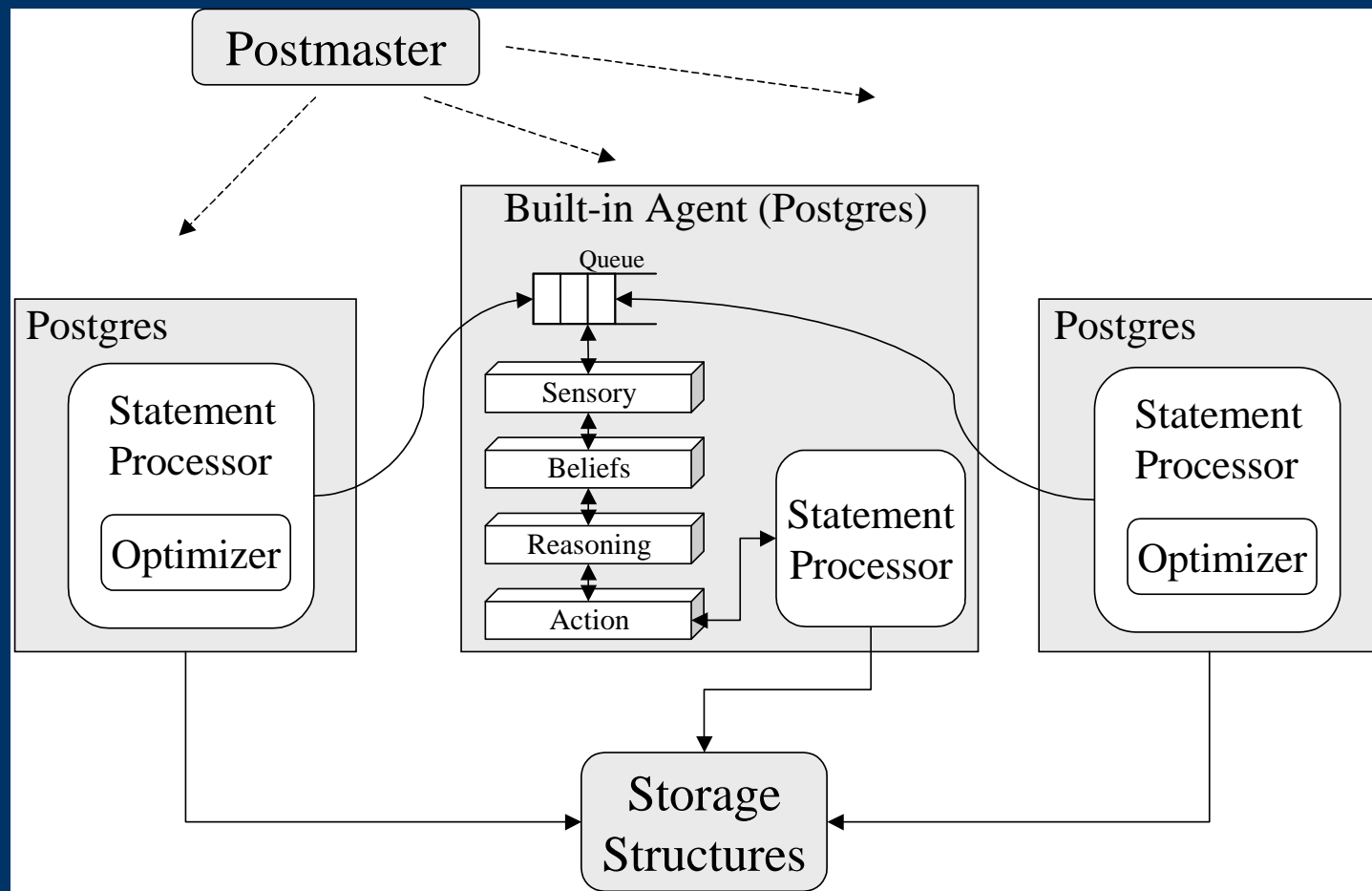
Agent based Self-tuning

- ◆ Agent Architecture (Kendall et al)
- ◆ Layered: ease of construction
- ◆ Well delimited points of interaction with DBMS



Agent based Self-tuning

◆ Integrating the tuning agent with PostgreSQL



Implementation Issues

- ◆ Environment:
 - PostgreSQL 7.4 beta 3
 - Agent coded in C++
- ◆ The DBMS must be altered to:
 - Integrate agent with DBMS: collection of system information and carrying of actions
 - Simulate “what-if” index configurations
- ◆ Focus on hypothetical indexes

Implementation

- ◆ Hypothetical Indexes – create and explain
 - Extended syntax for hypothetical indexes
 - Estimation of statistics
 - Selectivity
 - Table/Index order correlation
 - Number of tuples
 - Number of index pages
 - Optimizer mode for plan simulation

Implementation

- ◆ Hypothetical vs. Actual Indexes – creation time
- ◆ Actual index creation time grows with number of tuples
- ◆ Hypothetical indexes creation is almost instantaneous
- ◆ No significant overhead to create hypothetical indexes

Implementation

◆ Hypothetical vs. Actual Indexes – quality

Query Type	Query Text	Estimation Difference
Point query	<pre>select * from sales where s_num = 100;</pre>	none
Range and aggregation query	<pre>select s_prod_num, s_date, sum(s_value) as total from sales where s_value > 1500000 and s_date between '20040101' and '20040131' group by s_prod_num, s_date;</pre>	0.1%
Ordering query	<pre>select * from sales order by s_num;</pre>	26.94%

Conclusions and Future Work

◆ Contributions

- Agents can be integrated in DBMS to provide self-tuning functionality, with some impact on DBMS code
- Hypothetical indexes in PostgreSQL
- <http://www.inf.puc-rio.br/~postgresql>

◆ Ongoing and future work

- Use better ways to estimate number of pages
- Experimental results of index self-tuning performance to be published
- Investigate other uses of agents for DBMS self-tuning