Seminar
Self-Tuning-Databases

„Data-Placement: Physical Database Design for Data Warehouse“

Thomas Heutling

2004-01-14

Overview

- Introduction
- View-Index-Selection-Problem (VIS)
- A*-Algorithm
- Rules of Thumb
- Conclusion
- References

Introduction

improve query-response-time → materialized primary views

Problem

- consistent with the sources

Goals

- minimize the down-time
- minimize maintenance cost
- minimize maintenance time

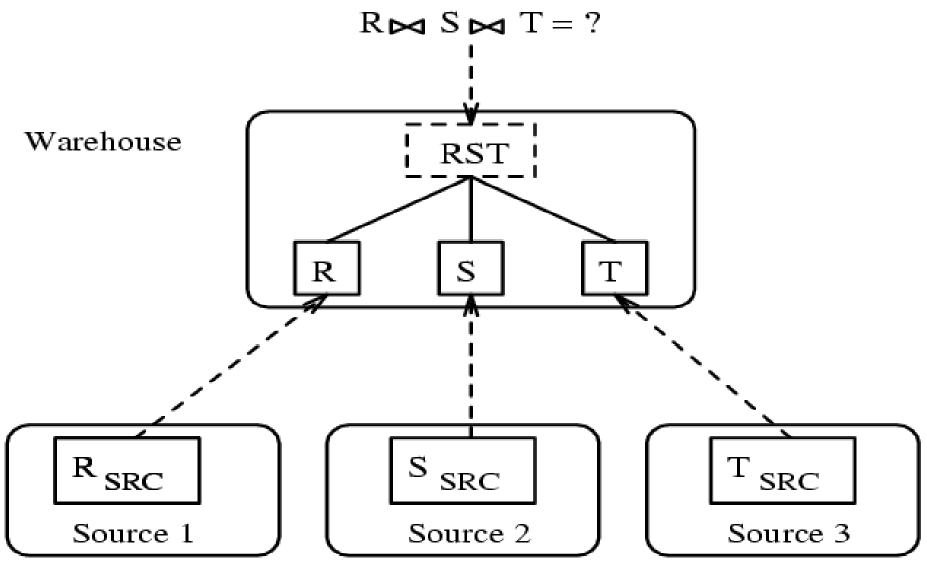
Example – Primary View

Problem

- new calculation too complex

Idea

- incremental maintenance



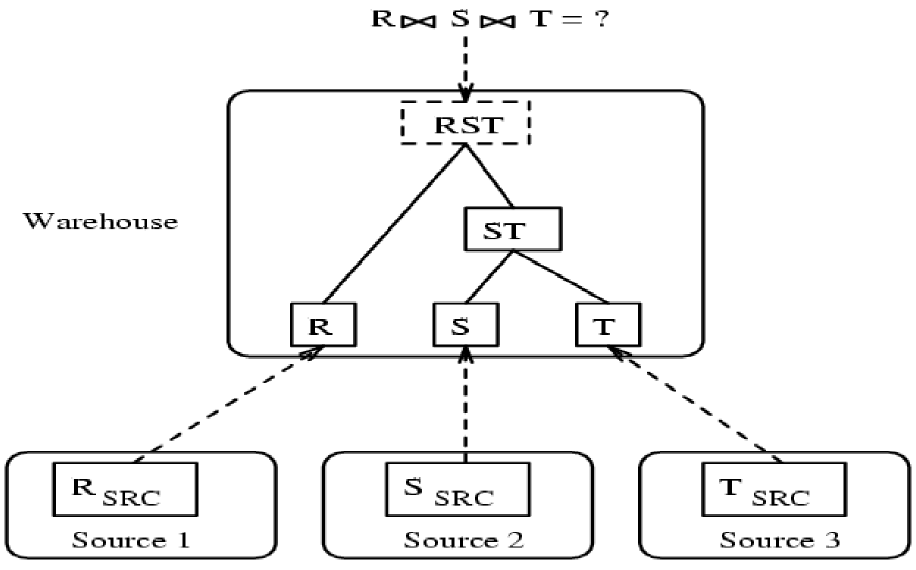
Example – Supporting View

Idea

- materialize supporting view

General Problem

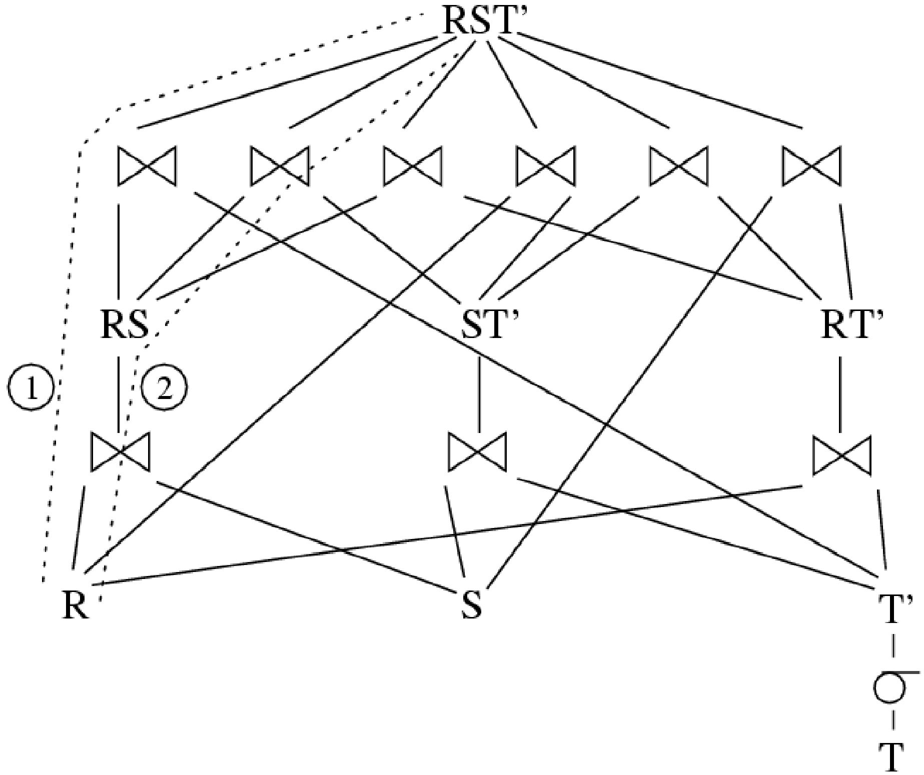
- choose a set of supporting views and a set of indexes to materialize
- minimal total maintenance cost



➔ **V**iew-**I**ndex-**S**election-Problem

View-Index-Selection-Problem

1. $(\triangle R \bowtie S) \bowtie T'$
2. $\triangle(R \bowtie S) \bowtie (S \bowtie T')$



➔ „multiple-query optimization“ problem

View-Index-Selection-Problem II

Choosing the view

- $\mathcal{O}(2^n)$ different nodes in query plan
- $\mathcal{O}(2^{2^n})$ different view state

Choosing the index

- $\mathcal{O}(2^n)$ different candidate indexes
- $\mathcal{O}(2^{2^n})$ different index state

➔ $\mathcal{O}(2^{2^n})$ query optimization problems

A*-Algorithm

Input: $\mathcal{M}, \prec$
Output: Optimal $\mathcal{M}'$

Let state set $S = \{s\}$, where s is a partial state having
  $\mathcal{M}_\mathcal{C}(s) = \mathcal{M}'(s) = \phi$, and $\mathcal{M}_\mathcal{U}(s) = \mathcal{M}$ (base relations and V are materialized)
Loop
  Select the partial state $s \in S$ with the minimum value of $\hat{C}$
  If $\mathcal{M}_\mathcal{C}(s) \equiv \mathcal{M}$, return $\mathcal{M}'(s)$
  Let $S = S - \{s\}$
  For each view or index $m \in \mathcal{M}_\mathcal{U}(s)$ such that for all $m' \prec m$: $m' \in \mathcal{M}_\mathcal{C}(s)$
    Construct partial state s' such that
      $\mathcal{M}_\mathcal{C}(s') = \mathcal{M}_\mathcal{C}(s) \cup \{m\}$, $\mathcal{M}_\mathcal{U}(s') = \mathcal{M}_\mathcal{U}(s) - \{m\}$, $\mathcal{M}'(s') = \mathcal{M}_\mathcal{C}(s) \cup \{m\}$
    Construct partial state s'' such that
      $\mathcal{M}_\mathcal{C}(s'') = \mathcal{M}_\mathcal{C}(s) \cup \{m\}$, $\mathcal{M}_\mathcal{U}(s'') = \mathcal{M}_\mathcal{U}(s) - \{m\}$, $\mathcal{M}'(s'') = \mathcal{M}_\mathcal{C}(s)$
    Let $S = S \cup \{s'\} \cup \{s''\}$
  Endfor
Endloop

A*-Algorithm – Comparsion

# of relations	# of selections	# of states visited exhaustive	A*	% pruned
2	0	32	11	67.7
2	1	192	21	89.1
2	2	960	28	97.1
2	4	960	29	97.0
3	1	2115072	17735	99.2
3	2	10575360	22809	99.8

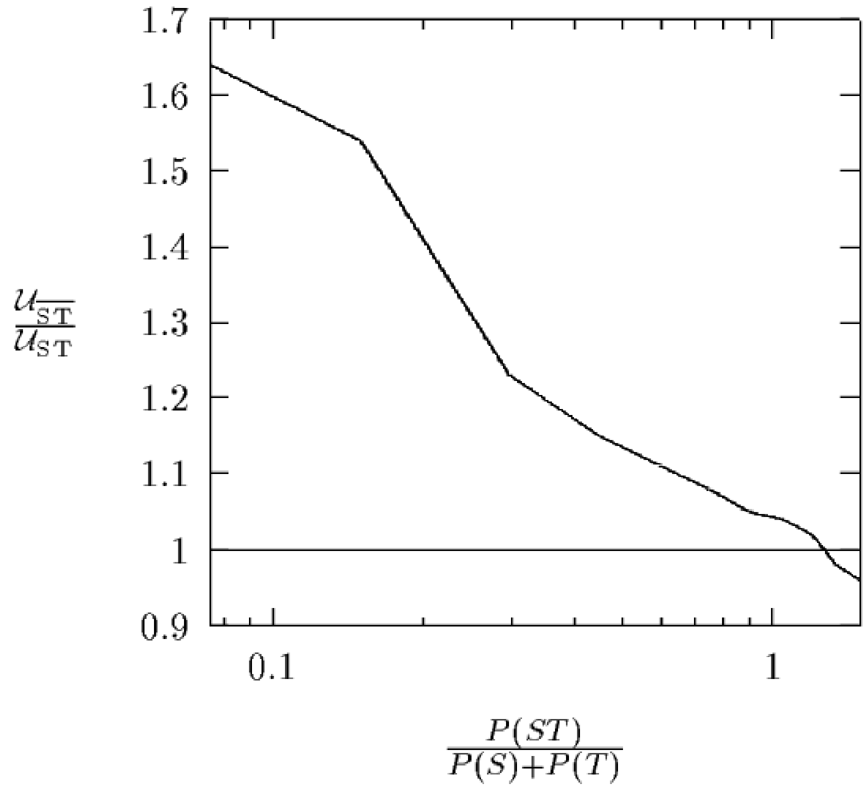
But

- often impractical except small views

Rules of Thumb

Rule 1
"Materialize Selective Supporting Views."

- $P(V) \ll P(\mathcal{E}(V))$

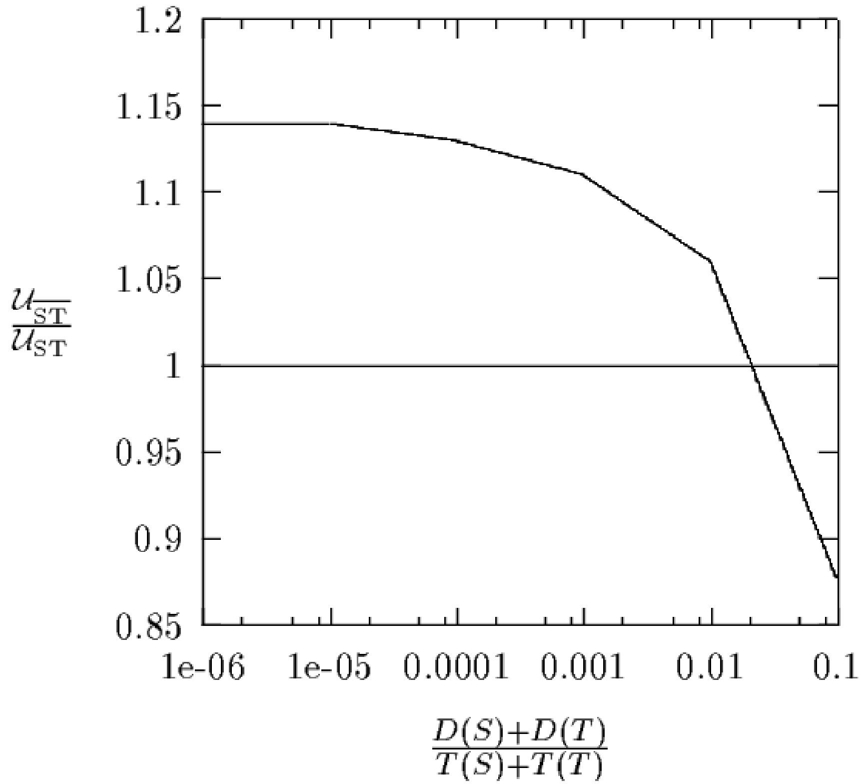


Rules of Thumb

Rule 2

"Materialize Supporting Views Having No Deletions or Updates."

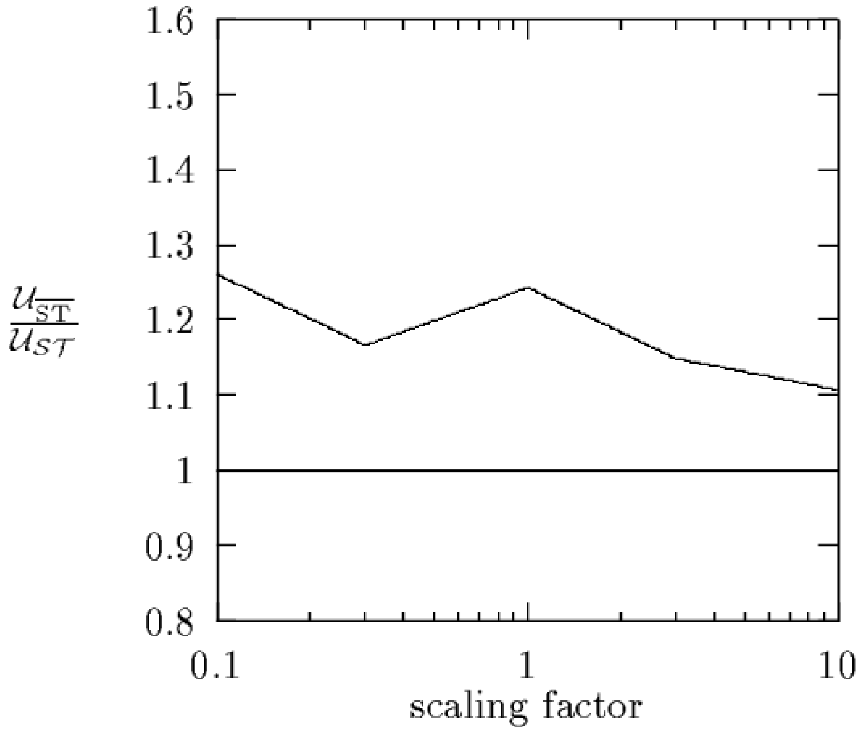
- $D(\mathcal{E}(V)) + U(\mathcal{R}(V)) = 0$



Rules of Thumb

Rule 3

"In considering whether to materialize a supporting view, the ratio of its size to the size of the memory buffer does not matter."



Rules of Thumb

Rule 4 – Build Indexes on Keys

"Build an index on a supporting view V for an attribut R.A that is the key of base relation R involved in V if"

- $D(R) + U(R) > 0$
- $D(R) + U(R) \ll P(V)$
- $I(\mathcal{R}(V)) + D(\mathcal{R}(V)) \ll P(V)$

Rules of Thumb

Rule 5 – Build Indexes on Join Attributes - Sometimes

"Build an index on supporting view V for an attribut R.A that is involved in a join condition R.A = S.B in the primary view when"

- $S \in \overline{\mathcal{R}(V)}$
- $I(\overline{\mathcal{R}(V)}) \ll P(V)$
- $I(\mathcal{R}(V)) + D(\mathcal{R}(V)) \ll P(V)$

Rules of Thumb

Rule 6 – Do Not Build Indexes on Local Selection Attributes

"Don't build an index on base relations R for an attribut R.A involved in a selection condition C unless"

- Indexes on R for attributes involved in join conditions have not been built
- a view $R' = \sigma_C R$ has not been materialized
- $S(R, C) \ll P(R)$
- $I(R) + D(R) \ll P(R)$

Rules of Thumb

Rule 7 – Build Indexes When the Index Fits In Memory

"Build an index on a supporting view V for an attribute R.A if for any of the above Rules 4, 5 or 6, all but the final condition hold,"

- $P(V, R.A) < P_m$

Conclusion

- supporting views and indexe minimize the maintenance time
- A*-Algorithm presented a optimal solution, but impractical for many real world problems
- avoid poor view sets, pick a good index set

References

[LQA97] W. J. Labio, D. Quass, B. Adelberg: Physical database design for data warehouse.

[LQA96] W. J. Labio, D. Quass, B. Adelberg: Physical database design for data darehouse – the vis problem. Technical Report, Stanford University, 1996. Available by anonymous ftp from db.stanford.edu in /pub/labio/1996