The COMFORT Automatic Tuning Project

Stefan Gerdelbracht
28.01.04
Why Automatic Tuning?

• there are typically 20-100 parameters just for system startup profile
• there is virtually no help in choosing settings in an intelligent way
• human resources are very cost intensive ("tuning gurus")

**Question:** Is automatic tuning indeed feasible or simply wishful thinking?
Introduction

**Comfortable Performance Tuning**

- started in 1990
- aims:
  - simplifying the job of human tuning experts
  - automating the entire tuning process
- two lines of research:
  - exploring architectural principles
  - building a prototype system
Self-Tuning Load Control for Locking
Tuning Problem

- transactions are executed concurrently: ACID-Properties must be kept
  - strict two-phase locking protocol
- lock conflicts can result in deadlocks
- probability of thrashing is influenced by:
  - number of concurrently executed transactions
  - access frequency of database objects
  - granularity of locking
  - length of transactions (number of locks, locking-time)
  - variance of transactions length
Sphinx Method

- MPL (multiprogramming level) limits number of concurrently executed transactions
Sisyphus Method

- specific MPL-limit for each transaction type
- scheduling-tree

![Diagram showing MPL limits for transaction types]
Conflict-driven Load Control

- **Idea:** adjusting the MPL limit in real-time
- **Problem:** which metric should be used?
  - interval-based (e.g. ratio of lock waits for lock requests)
  - status-based (e.g. blocked TAs / processed TAs)
- **conflict ratio** = \( \frac{\text{num. locks}}{\text{num. locks (non-blocked transactions)}} \)
Self-Tuning Load Control for Locking
Experiments

- CONF guarantees acceptable response time
- response time never worse than 150% of best methods response time
- results do not depend on load-specific fine-tuning of critical conflict ratio (values 1.3-1.9 acceptable)
  - is a self-tuning and robust method
Self-Tuning Memory Management
Tuning Problem

- much memory is used to buffer popular pages
- goal: reduce the overall disk I/O rate
- "Five-minute Rule": pages accessed more often than every five minutes should reside in memory
- data access frequencies may change over time
- popular pages have to be estimated correctly
- LRU (Least Recently Used) decides-bad in many practically relevant situations
Tuning Methods

- **LFU (Least Frequently Used)**: poor in reactivity to evolving workloads
- **“Spinx Method”**: density-based buffering (LRD-V2)
- **“Sisyphus Method”**: subdividing memory into several buffer pools (which are managed by LRU)
Automatic Tuning: LRU-K

- best metric for tuning: reference frequency
- basic idea: record the last K references for each page
- compute the interarrival time
- problem: storage overhead
- solution: if page was not re-referenced in the right time, the information can be dropped (can not be hot in this case)
Summary

• truly, self-reliant solutions have been lacking for most problems
• need of automatic performance tuning
• self-tuning approaches are feasible
• the authors hoped that their "work encourages more research in this increasingly important area"