DBench
DependabilityBenchmarking

# Portable Faultloads Based on Operator Faults for DBMS Dependability Benchmarking

Marco Vieira, Henrique Madeira

**University of Coimbra**

Portugal

**COMPSAC 2004 – The 26th Annual International Computer Software and Applications Conference**
Hong Kong, September 28-30, 2004

The 28th Annual International Computer Software and Applications Conference
COMPSAC 2004

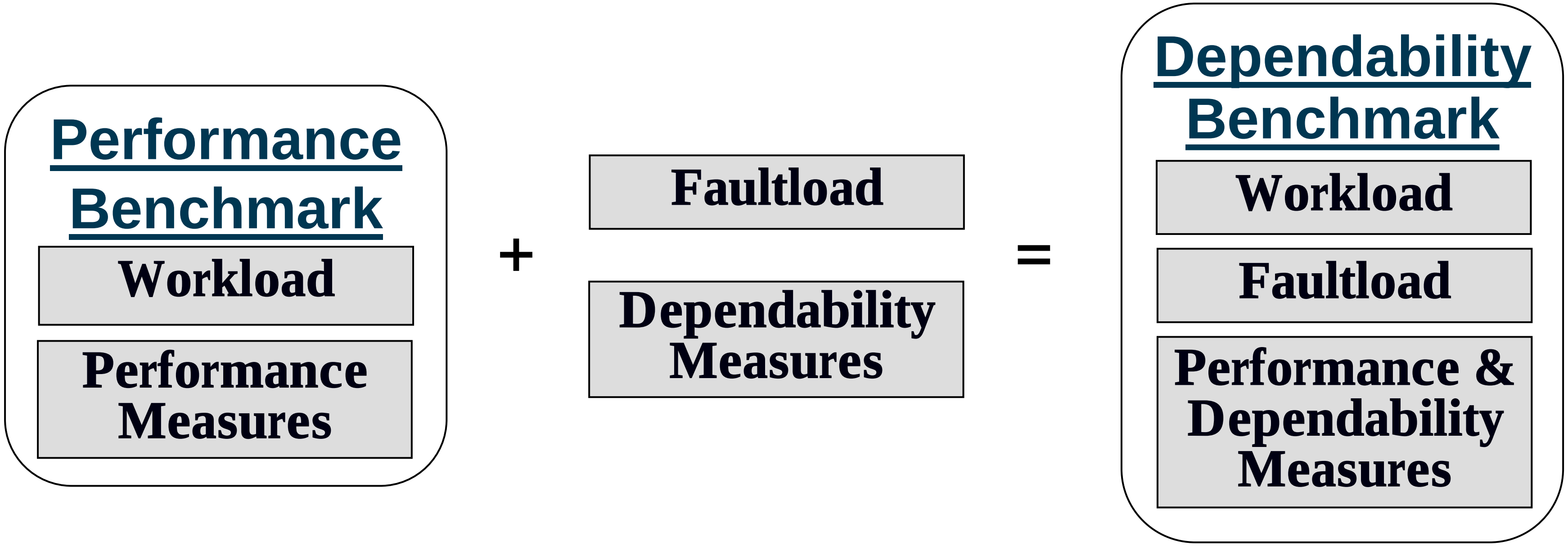
# Outline

- Defining Dependability Benchmarks for DBMS
- Operator Faults in DBMS
- Faultload Definition
- Example: DBench-OLTP Dependability Benchmark
- Conclusions

# Defining Dependability Benchmarks for DBMS

- DBMS Dependability benchmarks are quite useful to:
  - Compare different DBMS products concerning dependability features and performance
  - Help DBA in finding the best configurations for a good balance between performance and dependability

# Components of a Dependability Benchmark



- Procedure and rules
- Experimental setup

# Steps to Define a Dependability Benchmark

1) Identify the benchmark domain

2) Characterize the SUB in terms of typical features and functionalities

3) Define the dependability benchmark measures

4) Define the remaining dependability benchmark elements

   – Experimental setup, workload, faultload, procedures and rules

# Operator Faults in DBMS

- What are operator faults?
  - Administrator mistakes in DBMS administration tasks
  - End-user errors are not considered
- Why operator faults?
  - Responsible for most of the failures in DBMS
  - Can be easily injected in the system
- How can we identify operator faults in DBMS?

# DBMS Administration Areas

- Administration areas (common to all DBMS):
  - Memory & Processes
  - Security
  - Physical Storage
  - Database objects (tables, clusters, etc)
  - Recovery Mechanisms

**Classes of operator faults**

- Each administration area corresponds to several administration tasks

**Types of operator faults**

# Types of Operator Faults in Different DBMS

| Classes of Faults | Oracle | Sybase | Informix | Postgre | Common |
|---|---|---|---|---|---|
| Memory & Processes | 8 | 7 | 7 | 6 | 6 |
| Security | 7 | 5 | 5 | 2 | 2 |
| Physical Storage | 5 | 3 | 4 | 4 | 3 |
| Database Objects | 5 | 4 | 4 | 3 | 3 |
| Recovery Mechanisms | 6 | 6 | 5 | 3 | 3 |
| Total | 31 | 25 | 25 | 18 | 17 |

# Faultload Definition

- It is possible to establish an equivalence among most of the types of operator faults in different DBMS

↳ **Portability**

- Operator faults can be realistically emulated

↳ **Representativeness**

- Approaches for the faultload definition:
  - Faultload based on fault rates
  - Extensive faultload
  - Predetermined faultload

# Faultload Based on Fault Rates

1) Identify all the administration tasks for each core administration functionality considering a representative set of DBMS
2) Identify all the types of operator faults that may occur when executing each one of those administration tasks
3) Define weights for each fault type according to the fault rates obtained from the field
4) Select the subset of types of faults that is representative of the majority of real faults experienced by DBMS in the field

**Problem**: Fault rates from the field are difficult (or even impossible) to obtain

# Extensive Faultload

1) Identify all the administration tasks for each core administration functionality considering a representative set of DBMS

2) Identify all the types of operator faults that may occur when executing each one of those administration tasks

3) Define the faultload as the exhaustive list of all possible operator faults for all the types identified

**Problem**: Definition of extensive faultloads may become too difficult to implement and run

# Predetermined Faultload

1) Identify all the administration tasks for each core administration functionality considering a representative set of DBMS

2) Identify the tasks common to all the DBMS considered

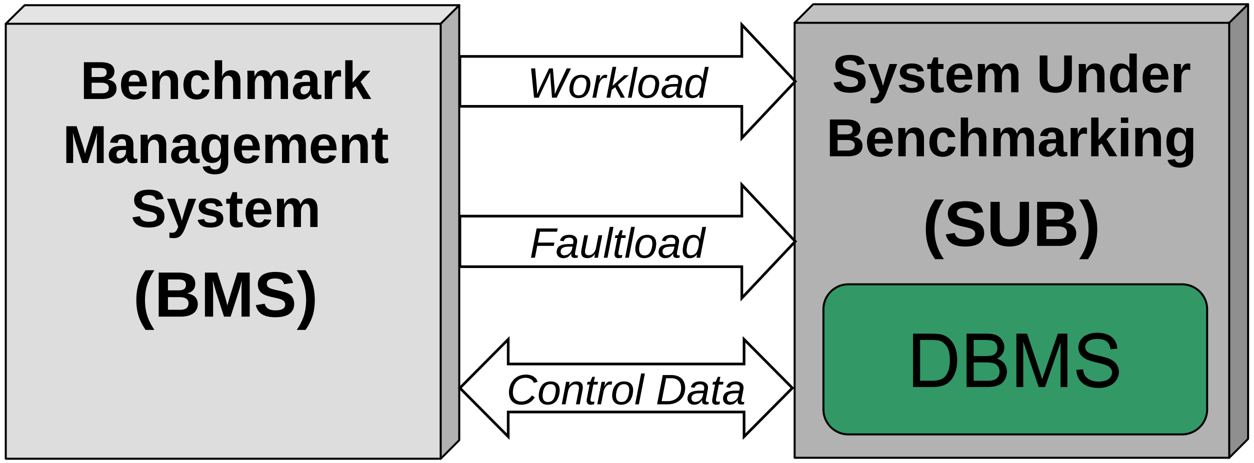
3) Identify the types of operator fault that may occur when executing each one of the administration tasks identified in 2

4) Define weights to each fault type according to the number of times the correspondent administration task is executed

5) The faultload includes the most representative types of operator faults identified in 4

# Example: DBench-OLTP Dependability Benchmark

- Experimental setup:



- Workload:
  - Adopted from the TPC-C performance benchmark

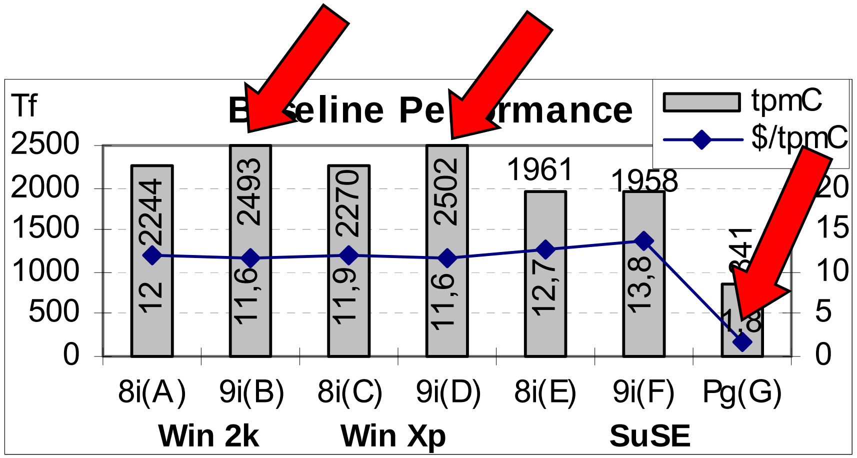
# DBench-OLTP Measures

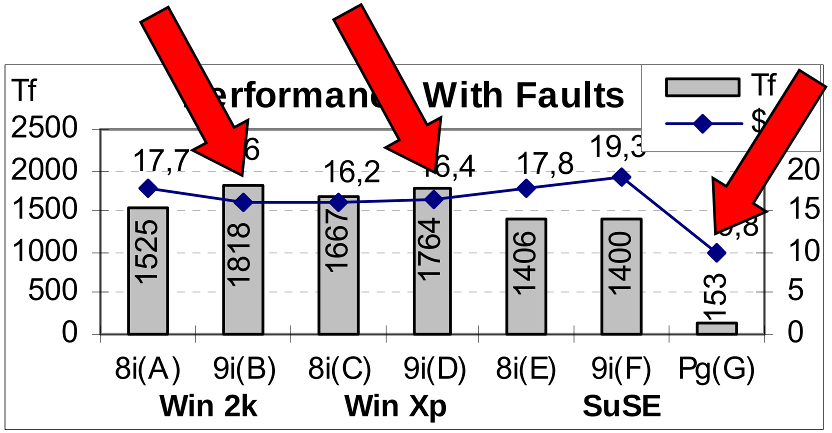
- Baseline performance measures:
  - tpmC – number of transactions executed per minute
  - $/tpmC – price per transaction
- Performance measures in the presence of the faultload:
  - Tf – number of transactions executed per minute (with faults)
  - $/Tf – price per transaction (with faults)
- Dependability measures:
  - Ne – number of data integrity errors
  - AvtS – availability from the server point-of-view
  - AvtC – availability from the clients point-of-view

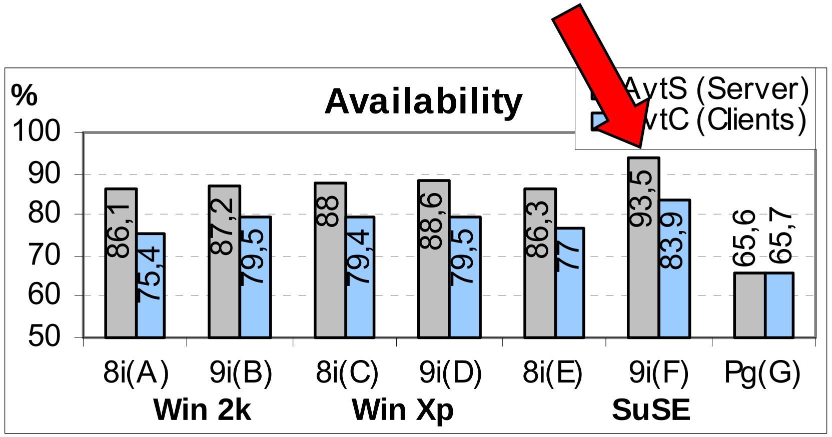
# DBench-OLTP Faultload Definition

- Predetermined faultload approach:
  - Four different DBMS:
    - Oracle 9i, Sybase Adaptive Server 12.5, Informix Dynamic Server 9, and PostgreSQL 7.3
  - Seven main types of operator faults have been selected:
    - Abrupt OS shutdown
    - Kill set of user sessions
    - Delete user schema
    - Delete set of files from disk
    - Abrupt DBMS shutdown
    - Delete table
    - Delete file from disk
    - Delete all files from one disk

# DBench-OLTP Benchmarking Results







- Performance
- Availability
- Price

# Conclusions

- Operator faults can be used to benchmark DBMS dependability

- Operator faults are a good starting point for the definition of faultloads for DBMS dependability benchmarking

- Software and hardware faults can also be considered in DBMS dependability benchmarks

- More info: www.dbench.org gbd.dei.uc.pt