Benchmarking in Software Engineering

Alessandro Garcia

October 2013

Benchmark Purposes

• Benchmarking is the process of comparing alternates according to performance metrics
• Benchmarking is an empirical method that has been used in Computer Science to compare the performance of:
  – systems, information retrieval algorithms, databases and many other technologies
  – purposes:
    • motivate rapid technical progress in a research area
    • community building through the comparison of techniques
Examples of Areas of SE with benchmarks

• Feature location tools
  – Goal: improve accuracy of feature location
  – JHotDraw
  – locating a lists of features
  – precision and recall

• Modular programming techniques
  – Goal: improve modularity support
  – HealthWatcher, change tasks, maintainability metrics

• Reverse engineering tools

Overview

• What is a benchmark?
• Benefits of benchmarks?
• Drawbacks of benchmarks?
• When to benchmark?
What is a benchmark?

• A benchmark is a standard test or set of tests used to compare alternatives. It consists of a motivating comparison, a task sample, and a set of performance measures.
  – Becomes a standard through acceptance by a community


Benchmark Components

1. Motivating Comparison
   – Motivation for research area and benchmark
     • E.g. compare the accuracy of feature location techniques
     • E.g. compare the maintainability of modular programming techniques

2. Task Sample
   – Representative sample of tasks from a problem domain
     • E.g. locate elements of crosscutting and non-crosscutting feature, functional and non-functional feature, etc…
     • E.g. effort to change crosscutting and non-crosscutting feature, effort to perform perfective, corrective, evolutionary changes, etc.…
   – Selection of one or more systems and artifacts
   – Most controversial part of a benchmark: representativeness of tasks?
     • Are we exposing strengths and weaknesses of each alternative
3. Performance Measures

- Be careful: not only execution time
- Can be any **quantitative** (or qualitative), measured by human, machine, or both
- Examples:
  - Feature location
    - Hits, false positives and false negatives
  - Maintainability with modular programming techniques
    - Number of modifications: additions, deletions and modifications
    - Time spent to perform a change
    - Faults introduced in each change

Performance Measures

- Do valid, accepted metrics already exist in the field?
- Are there correct and wrong answers (ground truth)?
  - If there aren’t correct answers, then results can be compared…
- Do you have metrics to quantify the relevant attributes of software quality?
  - E.g. modularity = coupling, cohesion, interface complexity, SoC
What is not a benchmark?

- Not an evaluation designed by an individual or single laboratory
  - Potential as starting point, **but not a standard**
- Not a baseline or fixed point
  - Needed for comparative evaluation, but not sufficient
- Not a application that is used repeatedly
  - Possibly a exemplar
  - E.g. **JHotDraw** – demonstrate how pattern-based development improves program maintainability

Benefits of Benchmarking

- Stronger consensus on the community’s research goals
  - mainly captured by the motivating comparison
- Greater collaboration between laboratories
  - benchmark components are defined by the community
- Direct comparison and repeatability
- Rapid dissemination of promising approaches
  - results are published in a public fashion
- Faster technical progress
Dangers of Benchmarking

- Costs to develop and maintain
  - meetings and repositories to publish benchmarks’ updates and application results to alternatives
- Overfitting
  - General performance is sacrificed for improved performance on benchmark
- Closing off other research directions (temporarily)

When to benchmark? Prerequisites for Benchmarking

- Minimum Level of Maturity
  - Proliferation of approaches and implementations
  - Recognized separate research area
- Ethos of Collaboration
  - Research networks
  - Seminars, workshops, meetings
- Tradition of Comparison
  - Use of common examples
The Way Forward…

- Start with an exemplar.
  - Motivating Comparison + Task Sample
- Use the exemplar within the network to learn about each other’s research
  - Comparison, discussions, relative strengths and weaknesses
  - Cross-fertilization, codification of knowledge
  - Hold meetings, workshops, symposia
- Add Performance Measures
- Use the exemplar (or benchmark) in publications
  - Common validation
- Promote use of exemplar (or benchmark) in broader research community