Course "Empirical Evaluation in Informatics"

Benchmarking

Lutz Prechelt
Freie Universität Berlin, Institut für Informatik

- Example 1: SPEC CPU2000
- Benchmark = measure + task sample + comparison
- Problems: cost, task composition, overfitting
- Quality attributes: accessibility, affordability, clarity, portability, scalability, relevance.
- Example 2: TREC

"Empirische Bewertung in der Informatik"

Vergleichstests (Benchmarks)

Prof. Dr. Lutz Prechelt
Freie Universität Berlin, Institut für Informatik

- Beispiel 1: SPEC CPU2000
- Benchmark = Maß + Aufgabe + Vergleich
- Probleme: Kosten, Aufgabenauswahl, Überanpassung
- Qualitätsmerkmale: Zugänglichkeit, Aufwand, Klarheit, Portierbarkeit, Skalierbarkeit, Relevanz
- Beispiel 2: TREC

"Benchmark"

Merriam-Webster online dictionary, m-w.com:

- a mark on a permanent object indicating elevation and serving as a reference in topographic surveys and tidal observations
- a point of reference from which measurements may be made
- a standardized problem or test that serves as a basis for evaluation or comparison (as of computer system performance)

Example 1: SPEC CPU2000

- SPEC = *Standard Performance Evaluation Corporation*
  - A not-for-profit consortium of HW and SW vendors etc.
  - Develops standardized measurement procedures (benchmarks) for various aspects of computer system performance
    - CPU (including cache and memory)
    - Cloud platforms , virtualization
    - Graphics
    - High-performance computing (msg-passing, shared-memory)
    - Java (client, server)
    - Mail server
    - Storage (network file system etc.)
    - Power consumption
  - We consider the CPU benchmark

Sources

- http://www.spec.org
- John Henning: "*SPEC CPU2000: Measuring CPU Performance in the New Millennium*", IEEE Computer, May 2000
    - The benchmark suite had five versions: CPU92, CPU95, CPU2000, CPU2006, CPU2017.
    - CPU2017 still has the same basic architecture.

CPU2000 approach

- Select a number of real-world programs
  - must be portable to all Unix and Windows systems of interest
  - balance different aspects such as pipelining, cache, memory performance etc.
  - some emphasize floating point computations (SPECfp2000)
  - others have only integer operations (SPECint2000)
    - now SPECspeed2017 Integer, SPECspeed2017 Floating Point, SPECrate2017 Integer SPECrate2017 Floating Point
      - rate vs. speed for multi-core vs. single-core performance
- Specify concrete program runs for each program
- Package programs and runs so as to make them easily applicable on any new system
  - application requires recompilation: SPEC also tests compiler performance!

CPU2000 performance measures

There are 2 x 2 different measurement modes:

- 2 different compiler settings:
    - using basic compiler optimization settings
        - → SPECint_base2000, SPECfp_base2000
    - using aggressive settings
        - → SPECint2000, SPECfp2000
        - requires experimentation and experience with the compiler

Benchmarks need to decide on many details!

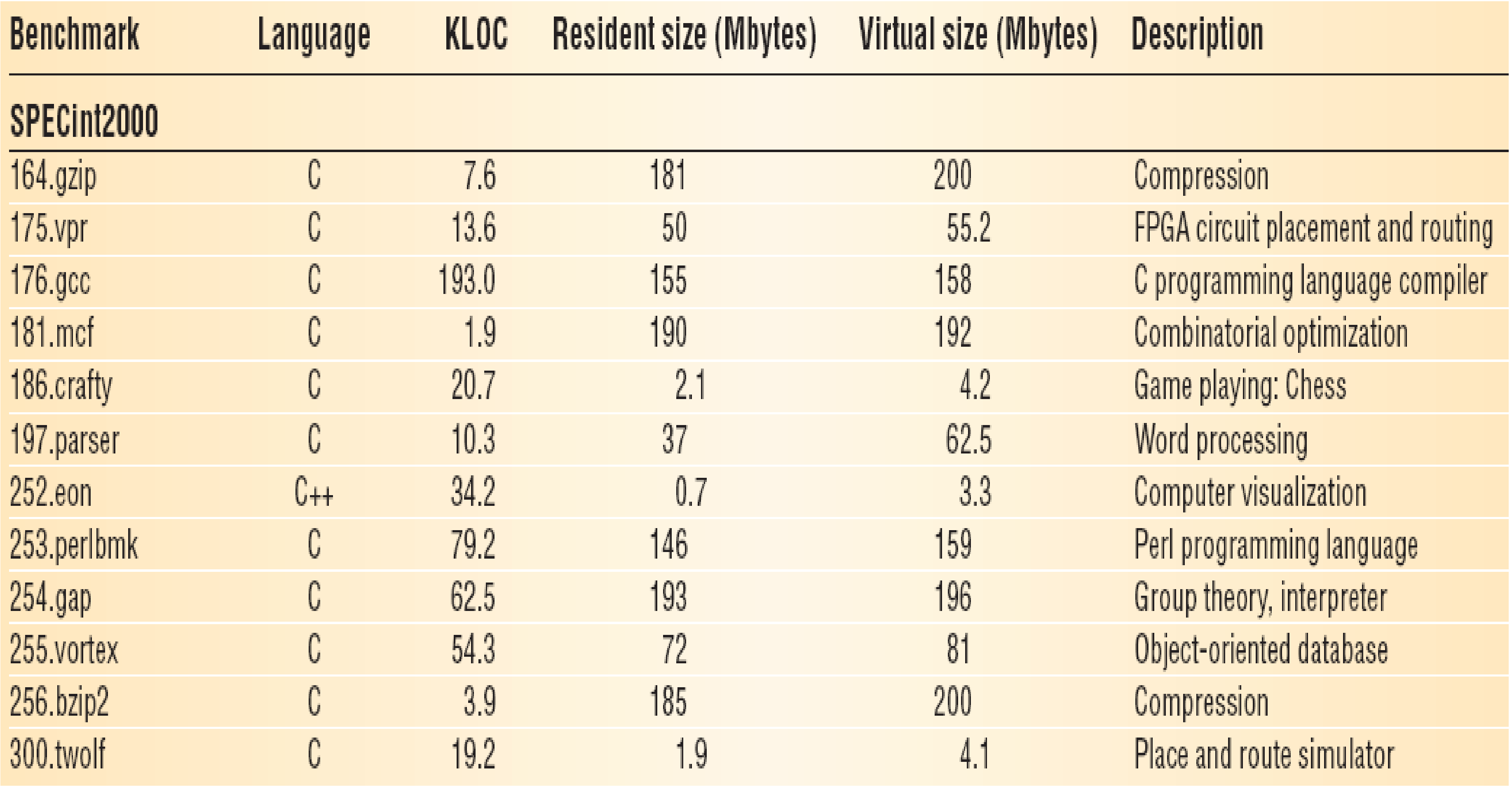
- 2 different measurements:
    - measuring speed (1 task)
    - measuring throughput (multiple tasks)
        - → SPECint_rate2000, SPECint_rate_base2000 etc.
        - throughput is relevant for multi-user systems or long-running processes

CPU2000 performance measures (2)

- Performance is expressed relative to a reference machine
  - Sun Ultra 5, 300 MHz
  - defined to have performance 100
    - used to normalize the measurements from the different programs

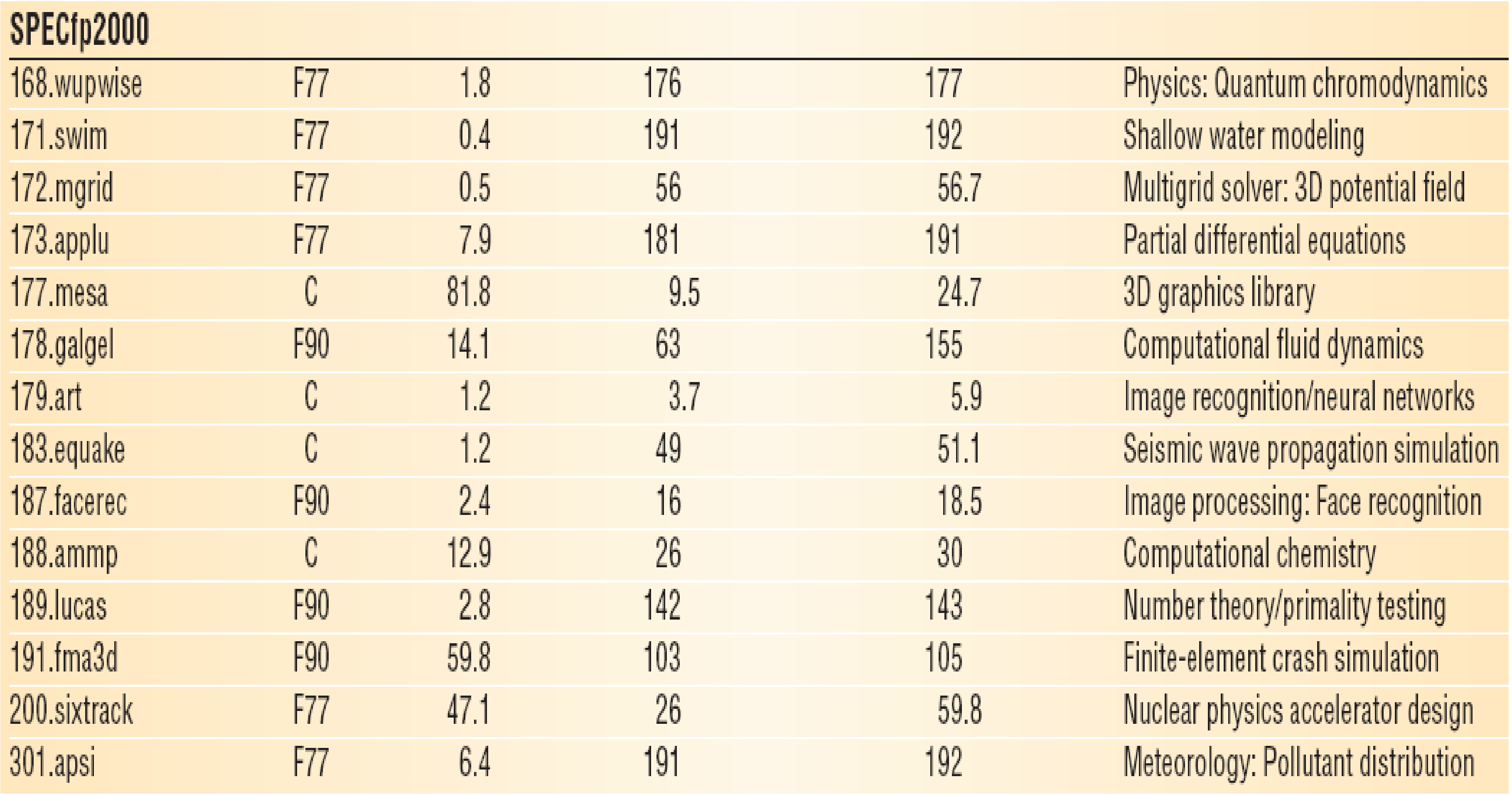
- Overall performance is determined as the geometric mean over the n benchmark programs
  - geometric mean: n-th root of the product
  - e.g. mean of 100 and 200 is 141
  - best results require steady performance across all programs

CPU2000
integer benchmark composition



Benchmark	Language	KLOC	Resident size (Mbytes)	Virtual size (Mbytes)	Description
SPECint2000					
164.gzip	C	7.6	181	200	Compression
175.vpr	C	13.6	50	55.2	FPGA circuit placement and routing
176.gcc	C	193.0	155	158	C programming language compiler
181.mcf	C	1.9	190	192	Combinatorial optimization
186.crafty	C	20.7	2.1	4.2	Game playing: Chess
197.parser	C	10.3	37	62.5	Word processing
252.eon	C++	34.2	0.7	3.3	Computer visualization
253.perlbmk	C	79.2	146	159	Perl programming language
254.gap	C	62.5	193	196	Group theory, interpreter
255.vortex	C	54.3	72	81	Object-oriented database
256.bzip2	C	3.9	185	200	Compression
300.twolf	C	19.2	1.9	4.1	Place and route simulator

floating point benchmark composition



SPECfp2000

168.wupwise	F77	1.8	176	177	Physics: Quantum chromodynamics
171.swim	F77	0.4	191	192	Shallow water modeling
172.mgrid	F77	0.5	56	56.7	Multigrid solver: 3D potential field
173.applu	F77	7.9	181	191	Partial differential equations
177.mesa	C	81.8	9.5	24.7	3D graphics library
178.galgel	F90	14.1	63	155	Computational fluid dynamics
179.art	C	1.2	3.7	5.9	Image recognition/neural networks
183.equake	C	1.2	49	51.1	Seismic wave propagation simulation
187.facerec	F90	2.4	16	18.5	Image processing: Face recognition
188.ammp	C	12.9	26	30	Computational chemistry
189.lucas	F90	2.8	142	143	Number theory/primality testing
191.fma3d	F90	59.8	103	105	Finite-element crash simulation
200.sixtrack	F77	47.1	26	59.8	Nuclear physics accelerator design
301.apsi	F77	6.4	191	192	Meteorology: Pollutant distribution

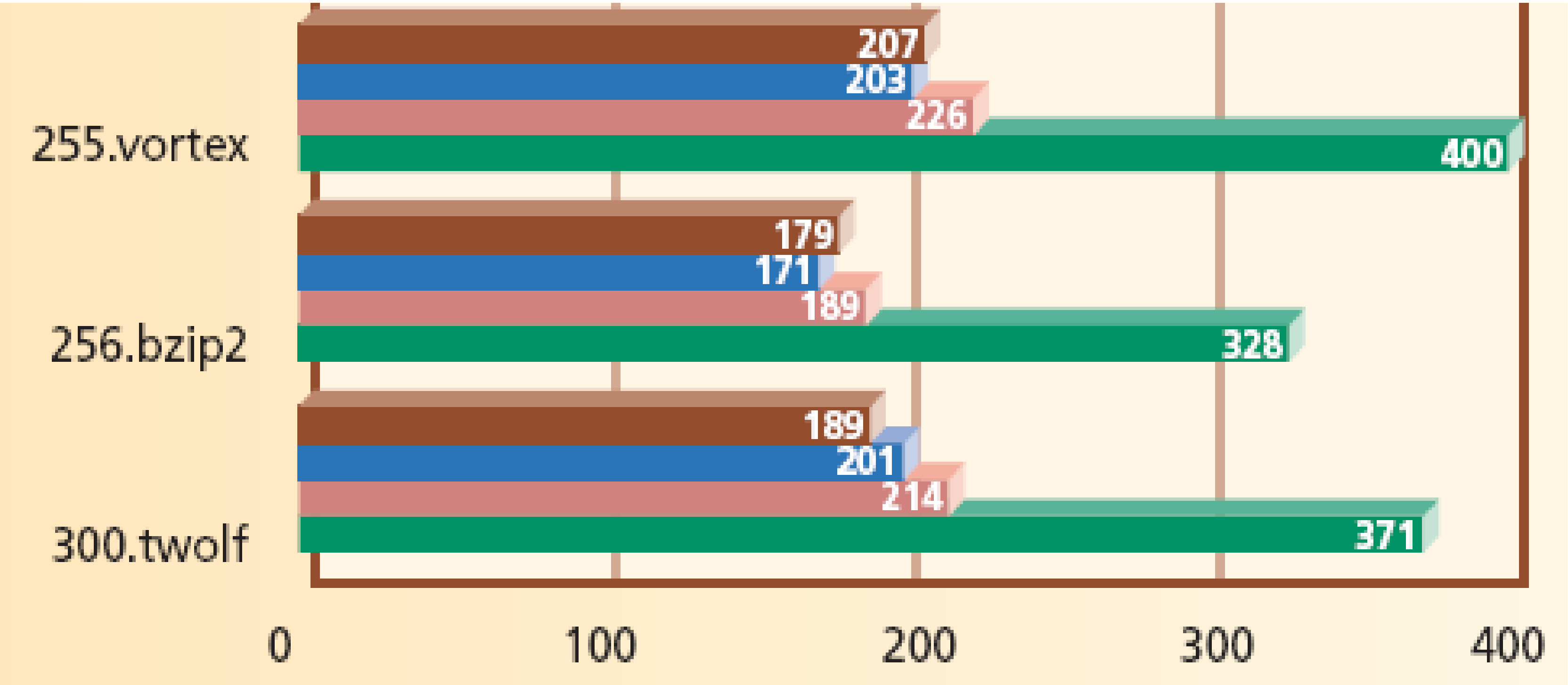
Reasons for selecting a program (or not)

- Should candidate program X be part of the benchmark?
- Yes if:
  - it has many users and solves an interesting problem
  - it exercises hardware resources significantly
  - it is different from other programs in the set
- No if:
  - it is not a complete application
  - it too difficult to port
  - it performs too much I/O
  - it is too similar to other programs in the set
- These factors are weighed against each other

Some results

- From top to bottom (in each group of 4 machines):
  - Processor clock speed: 500, 500, 533, 500 MHz
  - L1 cache size: 16, 16, 16, 128 KB
  - L3 cache size: 8, 2, 4, 4 MB

Which one will be slowest?



Problems of SPEC CPU2000

- Portability
  - It is quite difficult to get all benchmark programs to work on all processors and operating systems
  - SPEC uses 'benchathons': multi-day meetings where engineers cooperate to resolve open problems for the next version of the benchmark

- Which programs go into the benchmark set?
  - Won't one company's SPEC members try to get programs in that favor that company's machines?
  - No, for two reasons:
    1. SPEC is rather cooperative. These are engineers; they value technical merit
    2. The benchmark is too complex to predict what program might benefit my company's next-generation machine more than its competitors

Problems of SPEC CPU2000 (2)

Or: How to shoot yourself in the foot

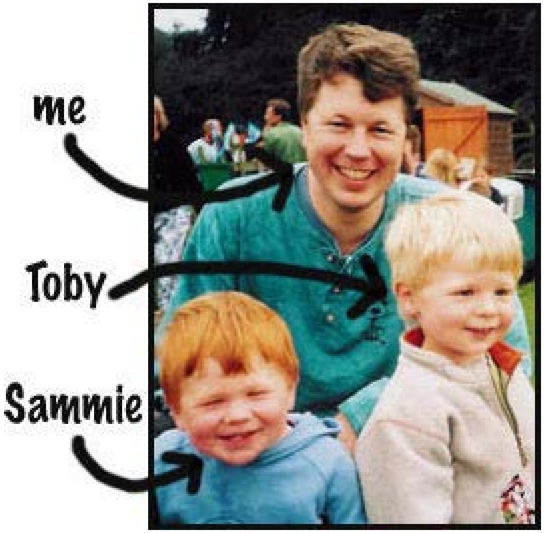
- Compiler optimizations can break a program's semantics
  - SPEC has to check the results produced for correctness
- Is execution time the right basic measurement?
  - The programs do have small source code differences on various operating systems (in particular for C and C++: **`#ifdef`**)
    - library not fully standardized, big-endian vs. little-endian etc.
  - Even identical programs with identical inputs may do different numbers of iterations
    - implementation differences of floating point operations
    - SPEC allows such differences within limits

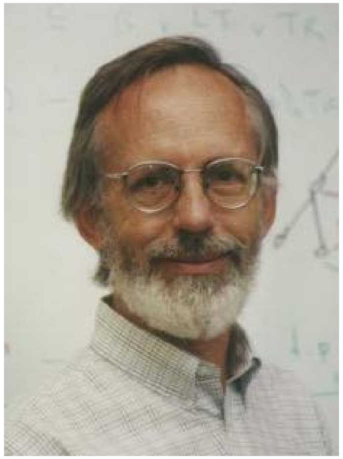
General benchmarking methodology

- Benchmarking is one of several evaluation methods
- We have now seen a concrete example
  - SPEC CPU2000
- Now let us look at the general methodology

Source

- Literature:
  - Susan Sim, Steve Easterbrook, Richard Holt:
    "*Using benchmarking to advance research: A challenge to software engineering*",
    25th Intl. Conf. on SW Engineering, IEEE CS press, May 2003





Benchmark parts

A benchmark consists of three main ingredients:

- Performance measure(s)
  - As a measure of fitness-for-purpose
  - Measurement is often automatic and usually quantitative, but could also be manual and/or qualitative
- Task sample
  - One or several concrete tasks, specified in detail
  - Should be relevant and representative
- Comparison
  - Measurement results are collected and compared
  - Provides motivation for using the benchmark
  - Promotes progress

Benchmarking methodology

1. Agree on a performance measure
2. Agree on a benchmarking approach
3. Define the benchmark content
4. Define a benchmarking procedure
5. Define a result report format
6. Package and distribute benchmark
7. Collect and catalog benchmark results

Benchmarks define paradigms

- A scientific benchmark operationalizes a research paradigm
  - Paradigm: Dominant view of a discipline
  - Reflects consensus on what is important
  - Immature fields cannot agree on benchmarks
- A commercial benchmark (such as SPEC) reflects a mainstream

Why are benchmarks helpful?

- Technical factors
  - Easy-to-understand and easy-to-use technique
  - High amount of control
  - Support replication of findings, hence credibility

- Sociological factors
  - Focus attention to what is (considered) important
  - Define implicit rules for conducting research
    - hence promote collaboration among researchers
    - help create a community with common interest
  - Promote openness
    - force the dirty details into the open
    - make hiding flaws difficult

Problems with benchmarks

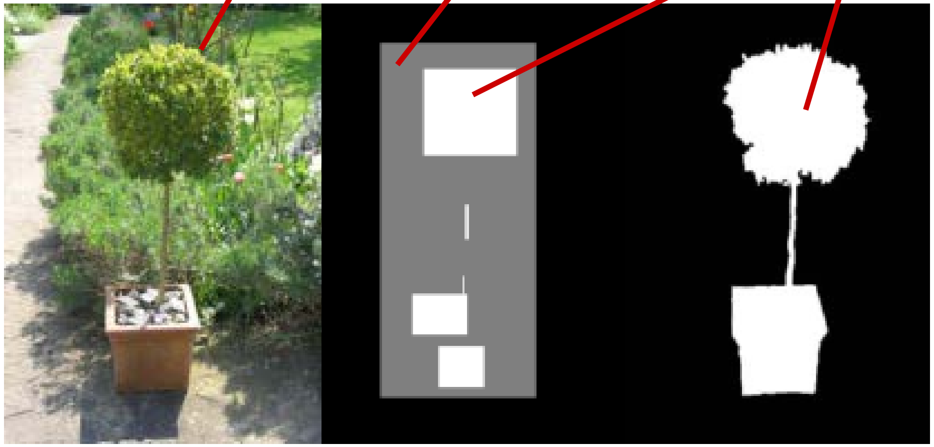
- Cost
  - Designing, composing, implementing, and packaging a benchmark is a very work-intensive task
    - Can only be done by a significant group of experts; takes long
- Task composition
  - Agreeing on what exactly goes into a benchmark task is difficult:
    - different players may have different foci of interest
    - different players may want to emphasize their own strengths
    - real-world usage profiles are usually unkown
- Overfitting
  - If the same benchmark task is used too long, the systems will adapt to it too specifically
    - benchmark performance will increase although real performance does not

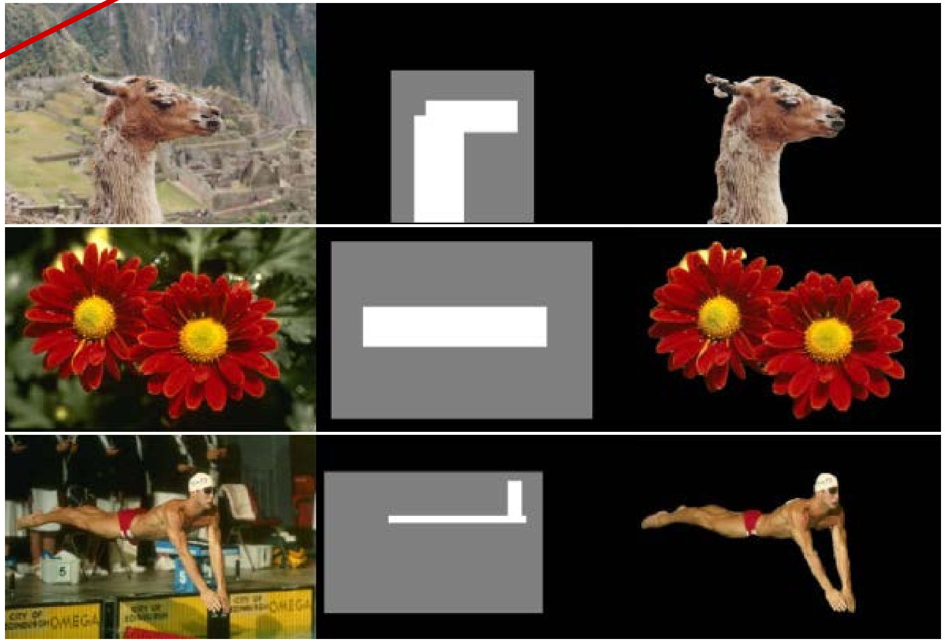
Quality attributes of good benchmarks

- Accessibility
  - should be publicly available and easy to obtain
- Affordability
  - effort required for executing benchmark must be adequate
- Clarity
  - specification must be unambiguous
- Portability, Scalability
  - must be easily applicable to different objects under study
- Relevance
  - task must be representative of real world
- Solvability (relevant for methods benchmarks)
  - objects under study must be able to "succeed"

A short benchmark example

- Image Segmentation benchmark
  - Given a picture, the user marks known foreground (white), and possible foreground (gray)
  - Segmentation algorithm tries to extract exactly all foreground
  - Result is compared against "ground truth"
    - distance measure??





http://csdl.computer.org/dl/proceedings/ism/2005/2489/00/24890253.pdf

Example 2: TREC

- Text Retrieval Conference
  - annually since 1992
  - Topic: *Information Retrieval* of text documents
    - Given large set of documents and query, find all documents relevant to the query and no others (like a web search engine)
    - Documents are ranked by perceived relevance
    - Performance measures:
      Precision: Fraction of retrieved documents that are relevant
      Recall: Fraction of relevant documents that are retrieved
  - Core activity is comparing results (and approaches for getting them) on pre-defined tasks used by the participants

- TREC now has many different *tasks*
  - Each of them is a separate benchmark
    - *number of tasks at TREC overall: 1992:* ***2****, 2005:* ***15****, 2018:* ***7***
      - There is even a formalized procedure for proposing new tracks
  - We will look at only one of them: "Ad-hoc retrieval"

Sources

- Conference homepage http://trec.nist.gov
- Ellen M. Voorhees, Donna Harman:
  "Overview of the Eighth Text REtrieval Conference (TREC-8)", 1999

TREC "Ad hoc retrieval" task

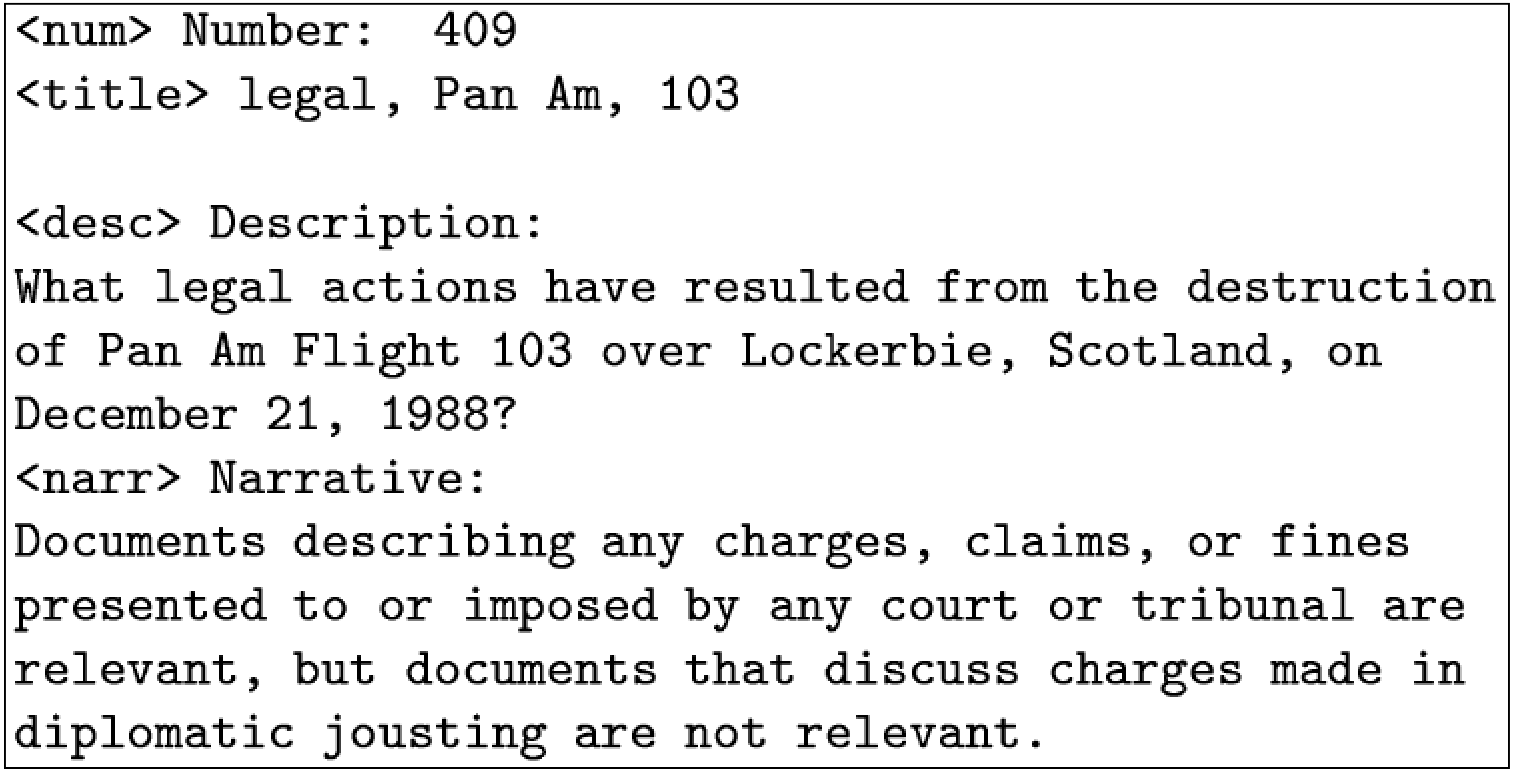
- started at TREC-1 (1992), used through TREC-8 (1999)
  - then discontinued because performance had leveled off
    - no more progress, the benchmark had done its job!
- Corpus contained 740 000 news articles in 1992
  - had grown to 1.5 Mio (2.2 GB) by 1998

Benchmark composition:

- 50 different query classes (called 'topics') are used
  - and changed each year
- Performance measures are Precision and Recall
- Comparison is done at the conference

An example 'topic definition'

- From TREC-8 (1999)



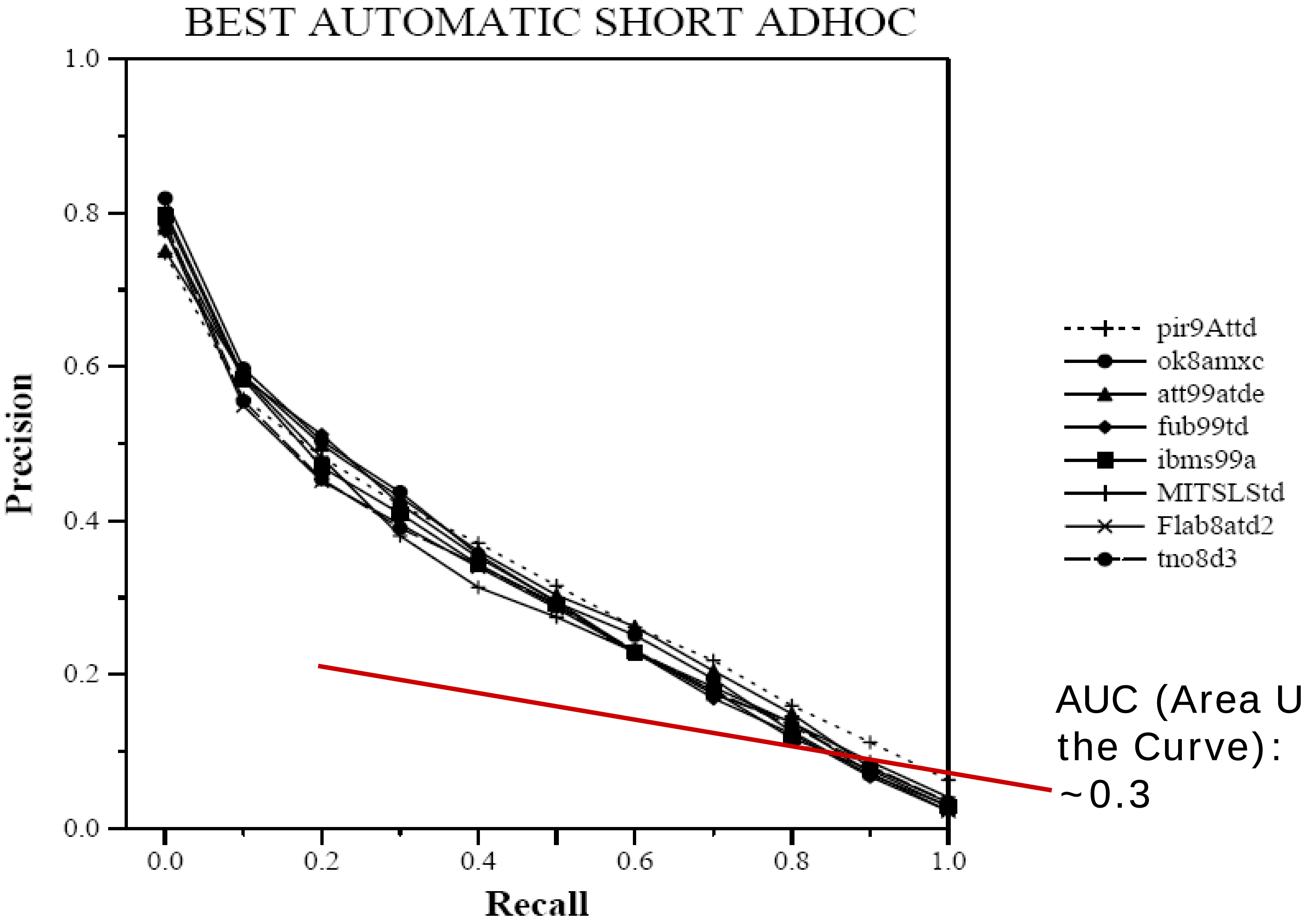
```
<num> Number:  409
<title> legal, Pan Am, 103

<desc> Description:
What legal actions have resulted from the destruction
of Pan Am Flight 103 over Lockerbie, Scotland, on
December 21, 1988?
<narr> Narrative:
Documents describing any charges, claims, or fines
presented to or imposed by any court or tribunal are
relevant, but documents that discuss charges made in
diplomatic jousting are not relevant.
```

- earlier topic definitions were more detailed

TREC procedure

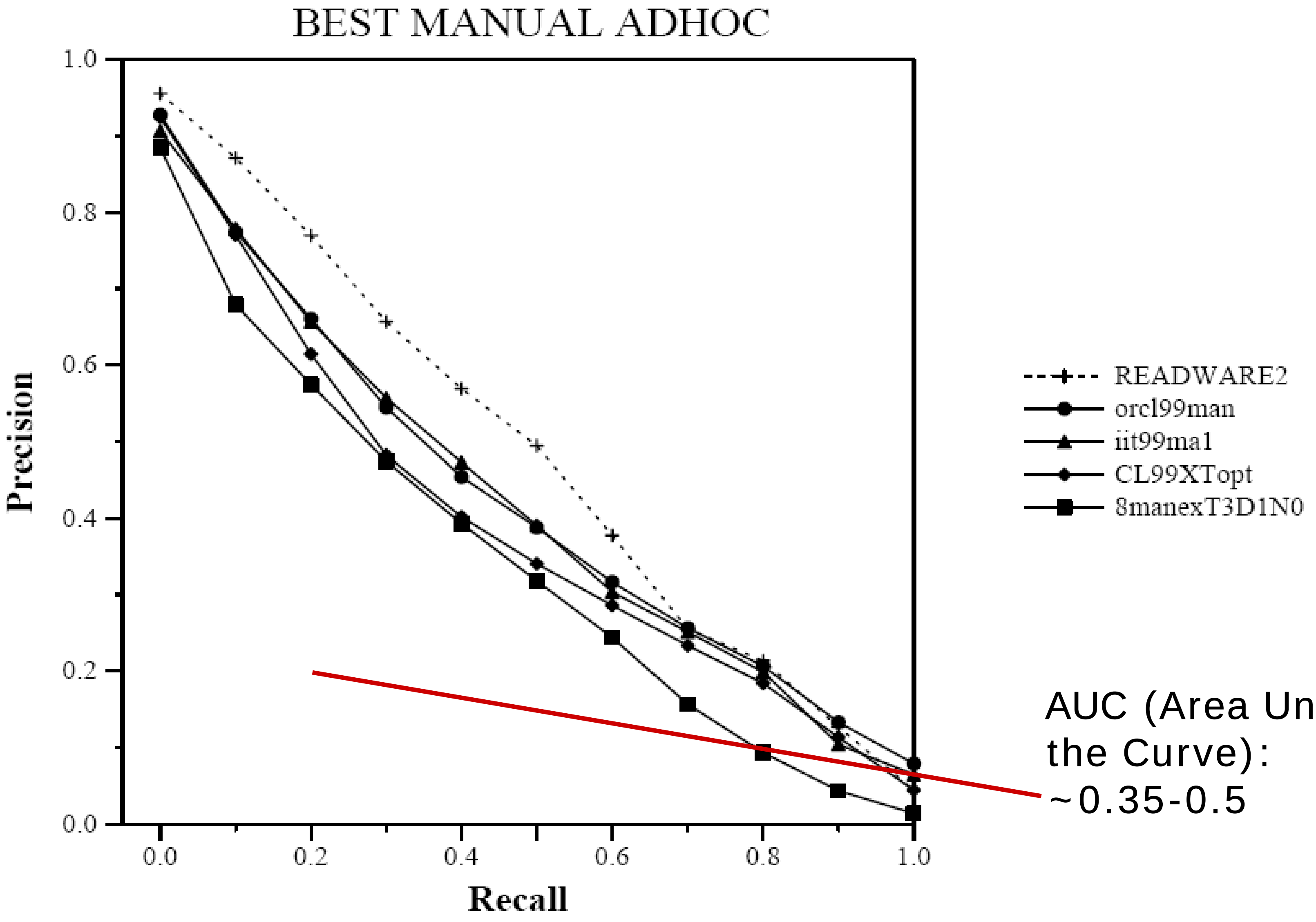
- Dozens of research groups from universities and companies participate:
    - run all 50 queries through their system
        - conversion from topic definition to query can be automatic or manual → two separate performance comparisons
    - submit raw retrieval results
    - conference organizers evaluate results and compile performance statistics
        - Precision: fraction of results that are correct
        - Recall: fraction of eligible documents that are in the results
    - at the conference, performance of each group is known
    - presentations explain the techniques used

Results (TREC-8, automatic query formulation)



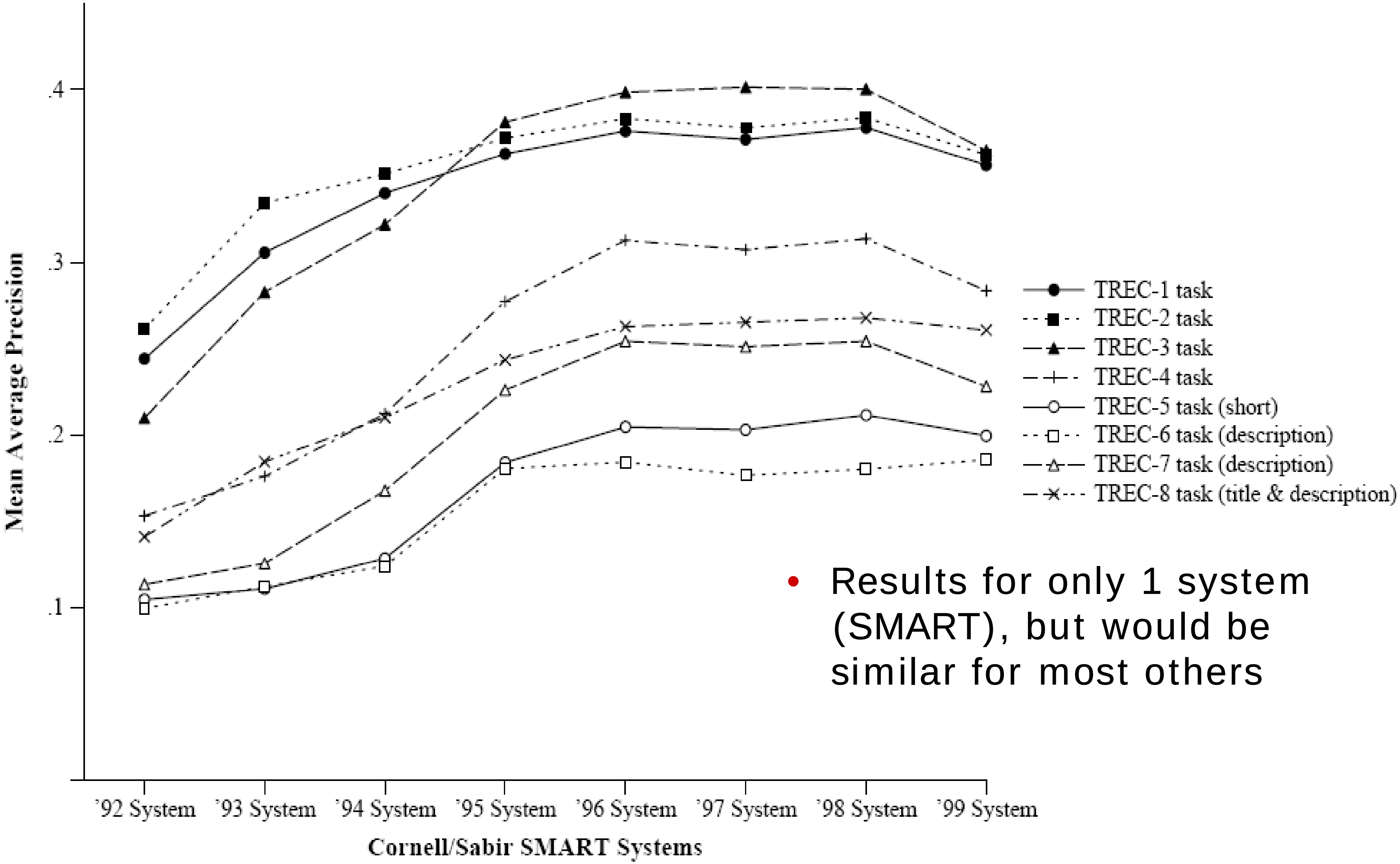
AUC (Area Under the Curve): ~0.3

Results (TREC-8, manual query formulation)



AUC (Area Under the Curve): ~0.35-0.5

Year-to-year improvement levels off



- Results for only 1 system (SMART), but would be similar for most others

Problem: How to judge query results

- How can anyone possibly know which of 1.5 Mio documents are relevant for any one query?
  - necessary for computing recall

- TREC procedure:
  - For each query, take the results of a subset of all participants
  - Take the top 100 highest ranked outputs from each
    - e.g. TREC-8: 7100 outputs from 71 systems
  - Merge them into the candidate set
    - e.g. TREC-8: 1736 unique documents (24 per system on average)
  - Have human assessors judge relevance of each document
  - Overall, consider only those documents relevant that were (a) in this set and (b) were judged relevant by the assessor
    - e.g. TREC-8: 94 relevant documents

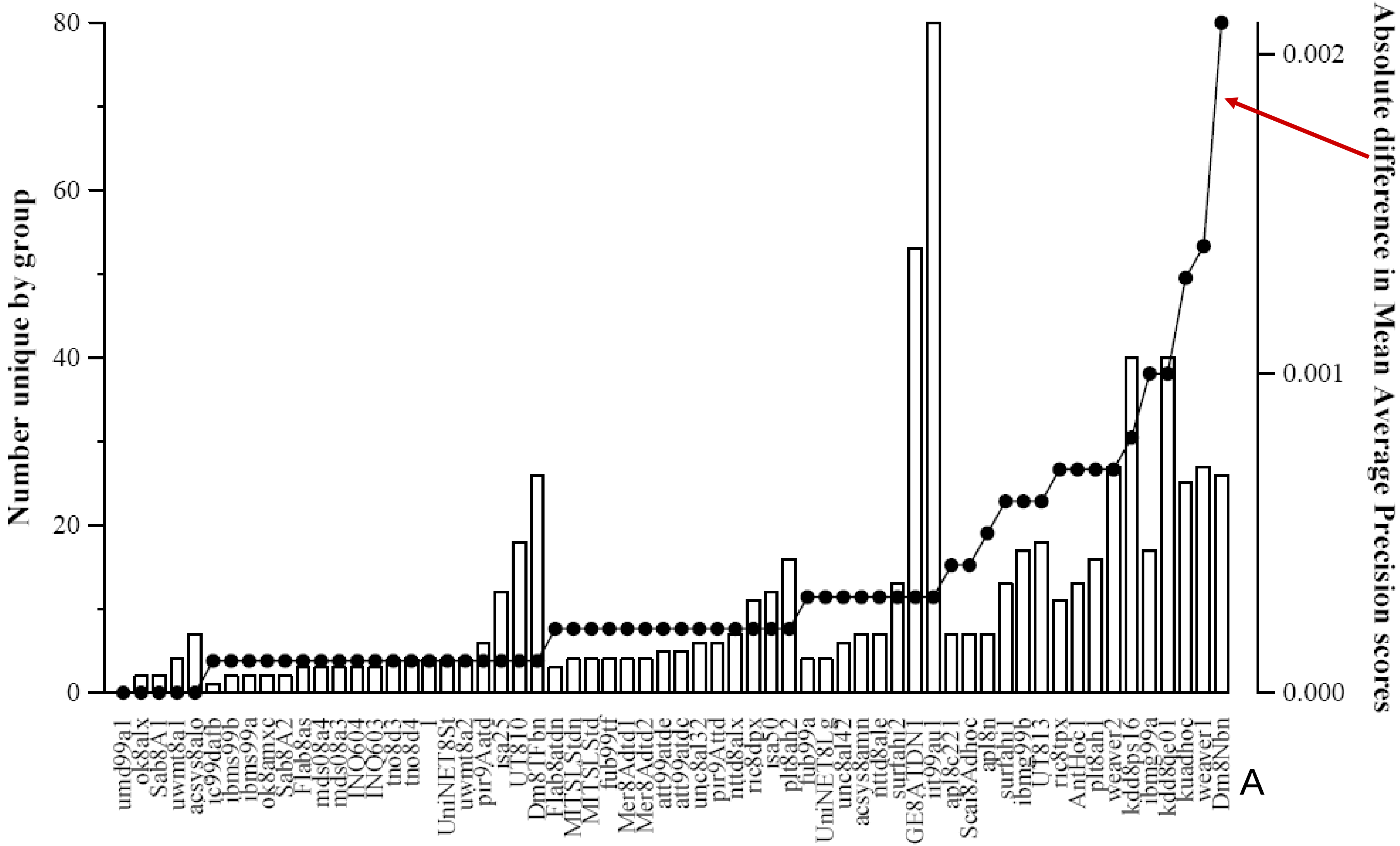
- (What are the problems with this procedure?)

TREC recall measurement problems

1. Human assessors make errors
   - This is bad for all participants who (at those points) do not
2. There are often many more relevant documents in the corpus beyond the candidate set
   - The procedure will consider them all irrelevant
   - This is bad for participants who did not contribute to the candidate set and
     - find documents of a different nature than the contributors or
     - rank relevance different than the contributors

How could TREC evaluate how serious this problem is?

Precision decrease for system A when hits unique to system A are left out



Summary

- Benchmarks consist of a performance measure, a task, and direct comparison of different results
  - Selecting tasks (and sometimes measures) is not straightforward!
- They apply to classical performance fields such as hardware, to capabilities of intelligent software (e.g. TREC), or even to methods to be applied by human beings
  - Measurement in a benchmark may even have subjective components
  - Even benchmarks can have credibility problems
- Putting together a benchmark is difficult, costly, and usually produces disputes over the task composition
- A good benchmark is a powerful and cost-effective evaluation tool.

Further literature

- ICPE: Int'l. Conf. on Performance Evaluation
- Web search for other computer benchmarks
- Related approach: RoboCup
    - Robot performance cannot be quantified,
      so use direct games and tournaments instead

Thank you!