Principles for Experimental Algorithmics

Michael T. Goodrich

UNIVERSITY of CALIFORNIA IRVINE

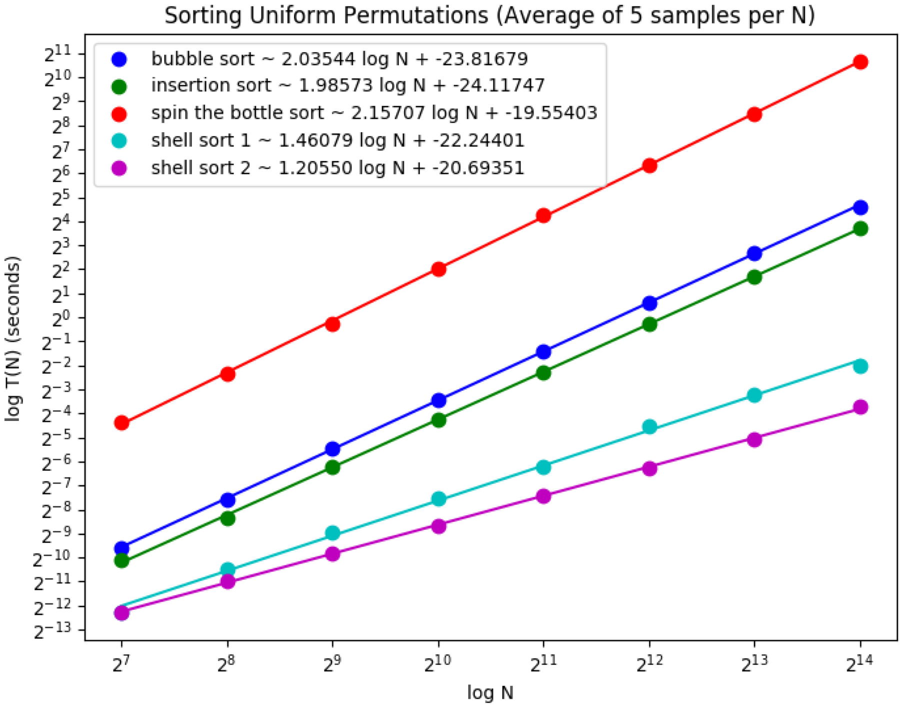
based, in part, on the following papers:

Towards a Discipline of Experimental Algorithmics, by Bernard M.E. Moret

A Theoretician's Guide to the Experimental Analysis of Algorithms, by David S. Johnson

Experimental Algorithmics

- Experimental Algorithmics studies algorithms and data structures by joining experimental studies with the traditional theoretical analyses.
  - Scientists do experiments because they have no choice
  - In experimental algorithmics we combine theoretical analysis with experimentation.

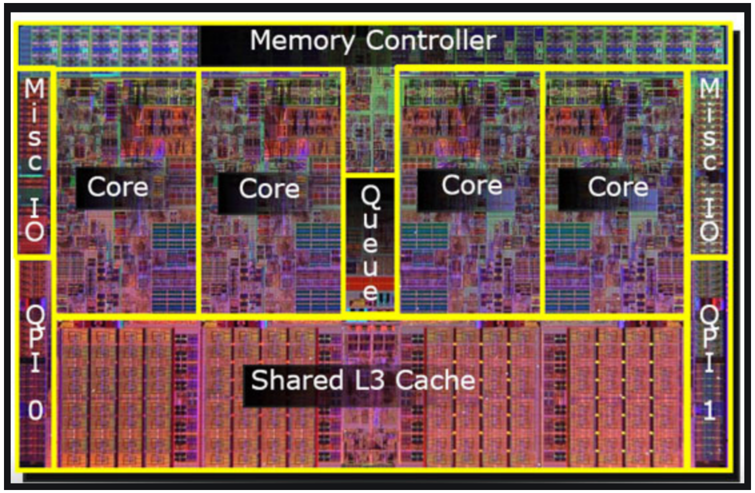


Experimental Algorithmics

- Experimentation with algorithms and data structures can prove to be indispensable for the following tasks:
  - The assessment of heuristics for hard problems
  - The characterization of asymptotic behavior of complex algorithms
  - The comparison of competing designs for tractable problems
  - The formulation of new combinatorial conjectures
  - The evaluation of optimization criteria
  - The transfer of research results from paper to production code

Perform Worthwhile Experiments

- Ask questions worth asking
  - – New problems
  - – New algorithms
  - – New types of input distributions
  - – New types of computer hardware



Measuring Actual Performance

- Random instances should be motivated from real-world data
- Also use real-world data when possible

DIMACS

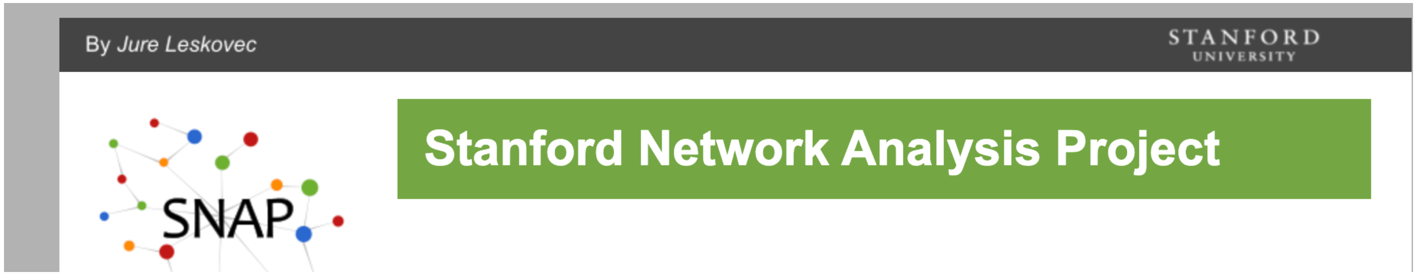
Center for Discrete Mathematics and Theoretical Computer Science

Founded as a National Science Foundation Science and Technology Center

Click here for event search

Enter search string

- http://dimacs.rutgers.edu/programs/challenge/



By *Jure Leskovec*

STANFORD UNIVERSITY

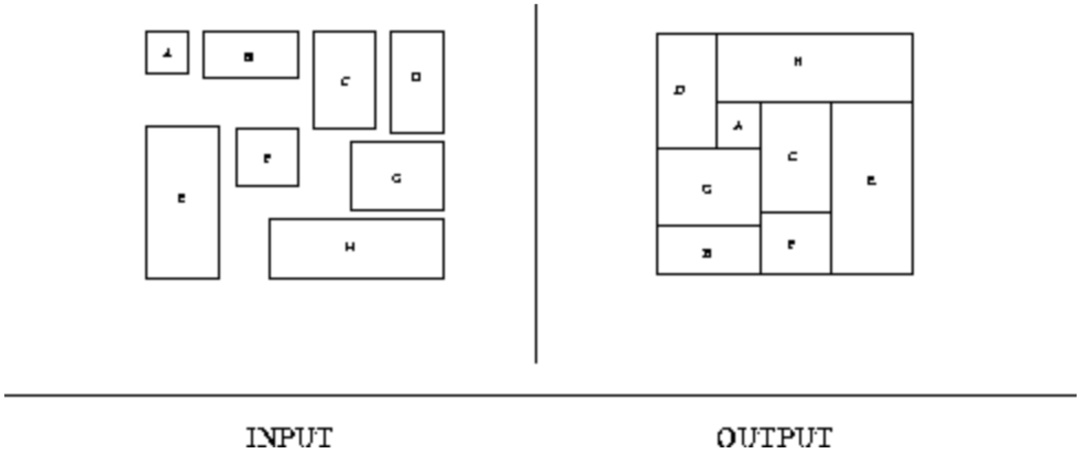
SNAP

Stanford Network Analysis Project

- http://snap.stanford.edu/

Testing the Quality of Solutions

- Find a parameter that can be effectively tested experimentally
  - – Waste in bin packing
  - – Closeness to a known lower bound

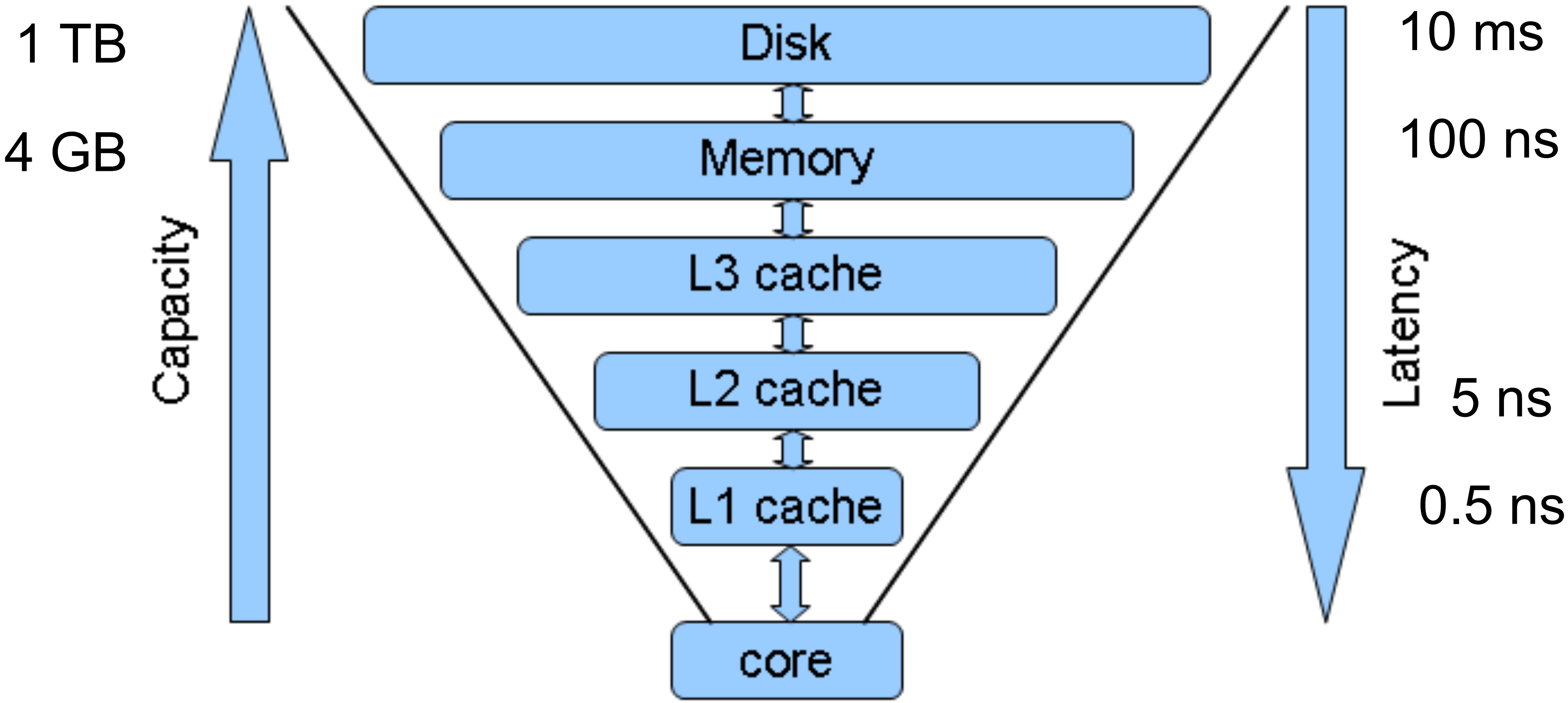


Experimental Setup

- Have clear objectives
- Gather data to answer the questions posed
- Choose hardware appropriately
- Code solutions consistently to allow for good conclusions
- Generate useful problem instances
- Analyze your data

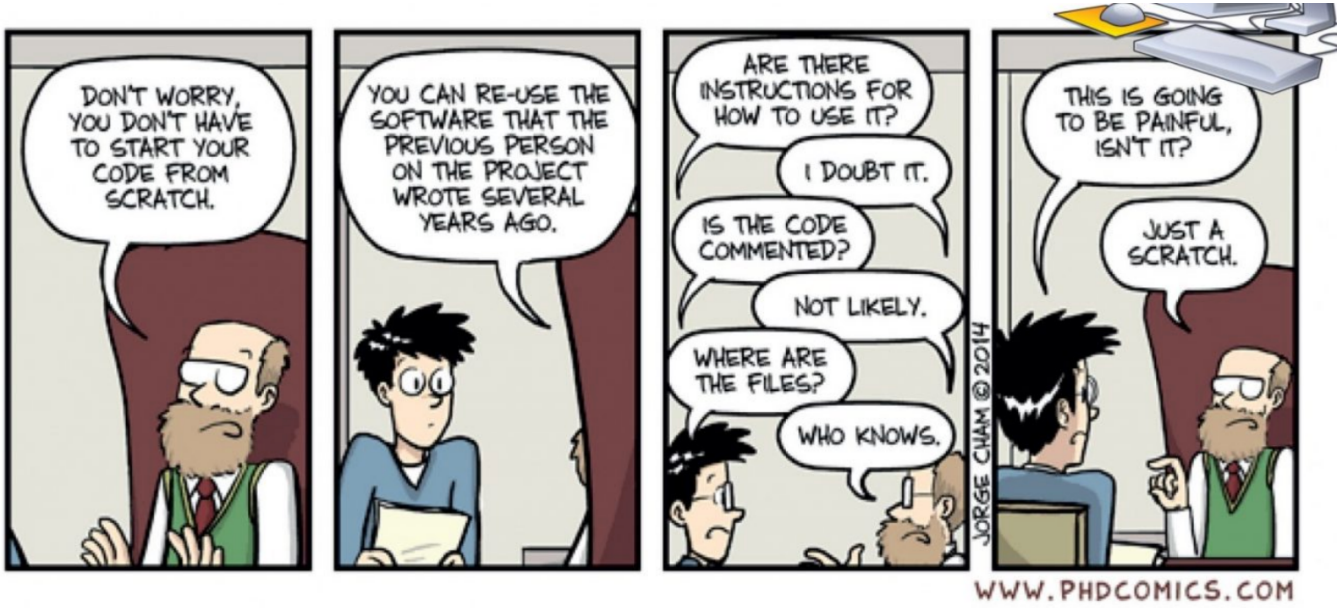
Understand Your Hardware: The Memory Hierarchy

- The trade-off of size and speed



Ensure Reproducibility

- Unless it is truly confidential, post your code for others to use.
- For random data, post how you generated it
- For real-world data, post how to get it

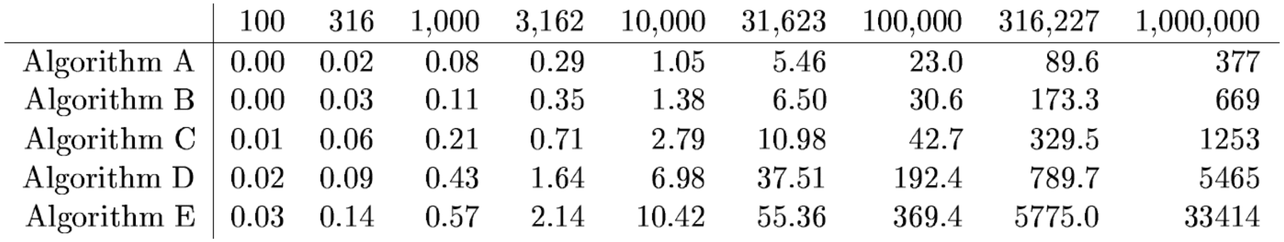

DON'T WORRY, YOU DON'T HAVE TO START YOUR CODE FROM SCRATCH.
YOU CAN RE-USE THE SOFTWARE THAT THE PREVIOUS PERSON ON THE PROJECT WROTE SEVERAL YEARS AGO.
ARE THERE INSTRUCTIONS FOR HOW TO USE IT?
I DOUBT IT.
IS THE CODE COMMENTED?
NOT LIKELY.
WHERE ARE THE FILES?
WHO KNOWS.
JORGE CHAM © 2014
THIS IS GOING TO BE PAINFUL, ISN'T IT?
JUST A SCRATCH.
WWW.PHDCOMICS.COM

Ensure Comparability

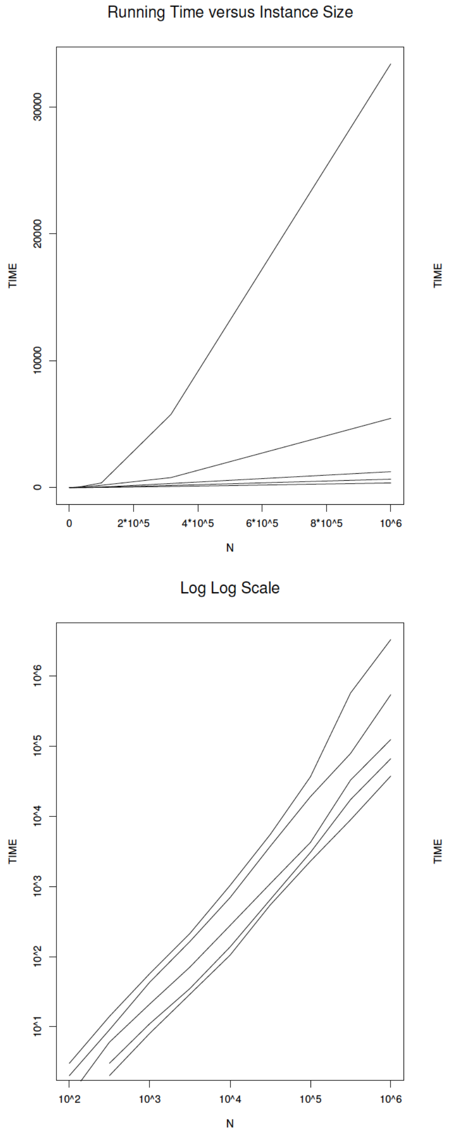
- Perform all experiments on the same hardware
- Report the type of hardware used
- Code all algorithms with the same level of code optimizations and tuning

Present your Data in Meaningful Ways

- Use tables only for small data sets



	100	316	1,000	3,162	10,000	31,623	100,000	316,227	1,000,000
Algorithm A	0.00	0.02	0.08	0.29	1.05	5.46	23.0	89.6	377
Algorithm B	0.00	0.03	0.11	0.35	1.38	6.50	30.6	173.3	669
Algorithm C	0.01	0.06	0.21	0.71	2.79	10.98	42.7	329.5	1253
Algorithm D	0.02	0.09	0.43	1.64	6.98	37.51	192.4	789.7	5465
Algorithm E	0.03	0.14	0.57	2.14	10.42	55.36	369.4	5775.0	33414



Running Time versus Instance Size

Log Log Scale