http://algs4.cs.princeton.edu

1.4 ANALYSIS OF ALGORITHMS

- *introduction*
- *observations*
- *mathematical models*
- *order-of-growth classifications*
- *theory of algorithms*
- *memory*

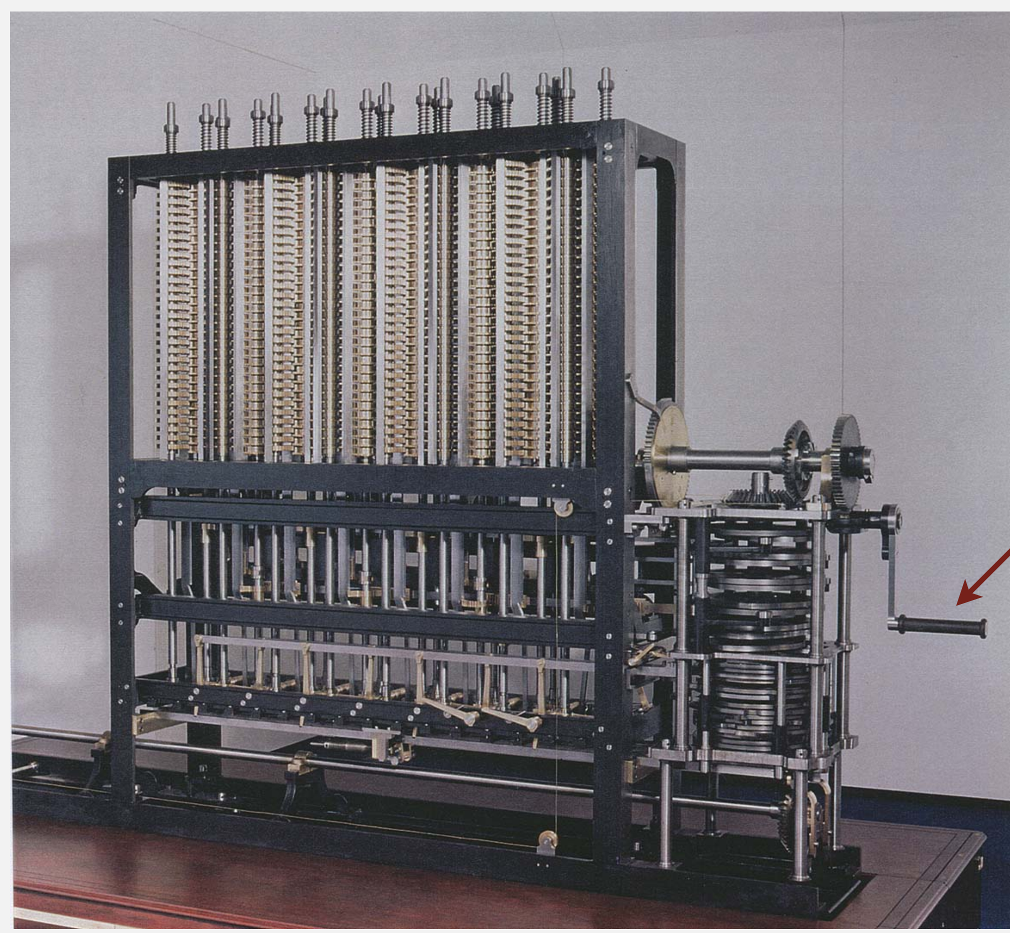
http://algs4.cs.princeton.edu

1.4 Analysis of Algorithms

- *introduction*
- *observations*
- *mathematical models*
- *order-of-growth classifications*
- *theory of algorithms*
- *memory*

Running time

" As soon as an Analytic Engine exists, it will necessarily guide the future course of the science. Whenever any result is sought by its aid, the question will arise—By what course of calculation can these results be arrived at by the machine in the shortest time? " — *Charles Babbage (1864)*



how many times do you have to turn the crank?

Analytic Engine

Cast of characters

Programmer needs to develop a working solution.

Student might play any or all of these roles someday.

Client wants to solve problem efficiently.

Theoretician wants to understand.

Reasons to analyze algorithms

Predict performance.

Compare algorithms.

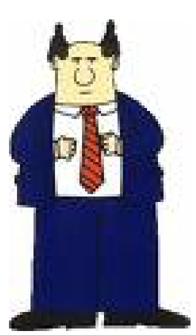
Provide guarantees.

Understand theoretical basis.

this course (COS 226)

theory of algorithms (COS 423)

Primary practical reason: avoid performance bugs.



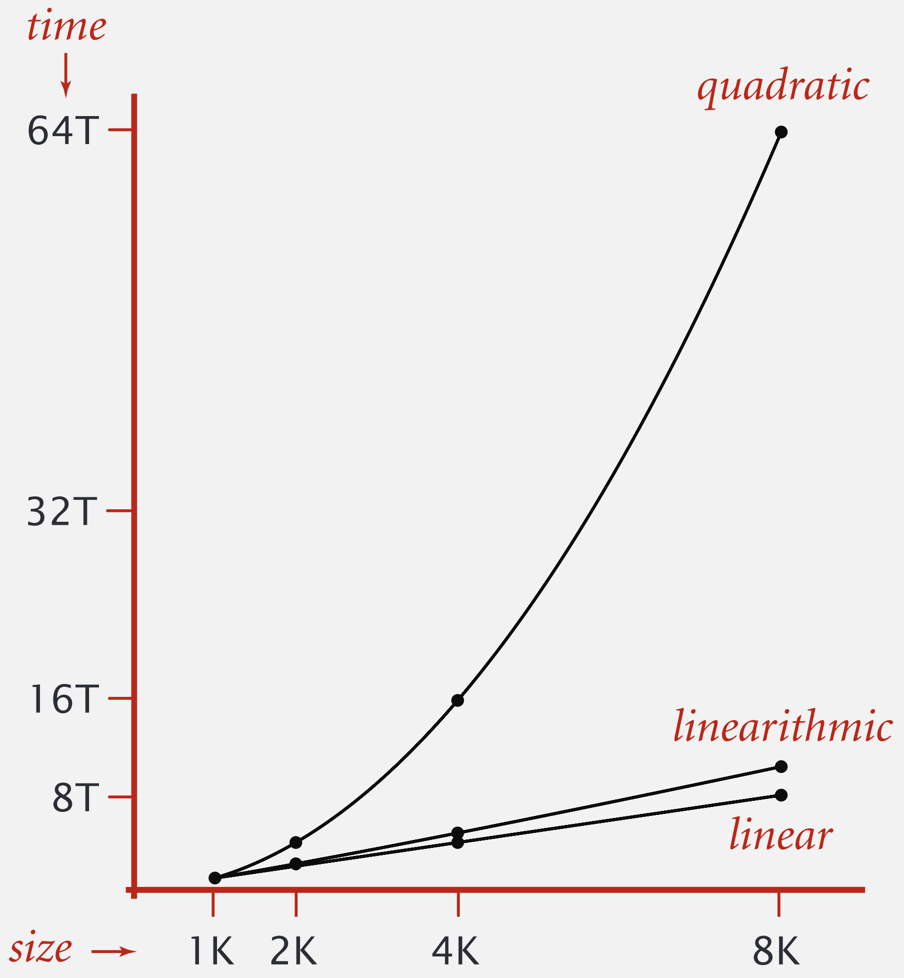
client gets poor performance because programmer did not understand performance characteristics

Some algorithmic successes

Discrete Fourier transform.

- Break down waveform of N samples into periodic components.
- Applications: DVD, JPEG, MRI, astrophysics, ...
- Brute force: N^2 steps.
- FFT algorithm: $N \log N$ steps, enables new technology.

Friedrich Gauss
1805

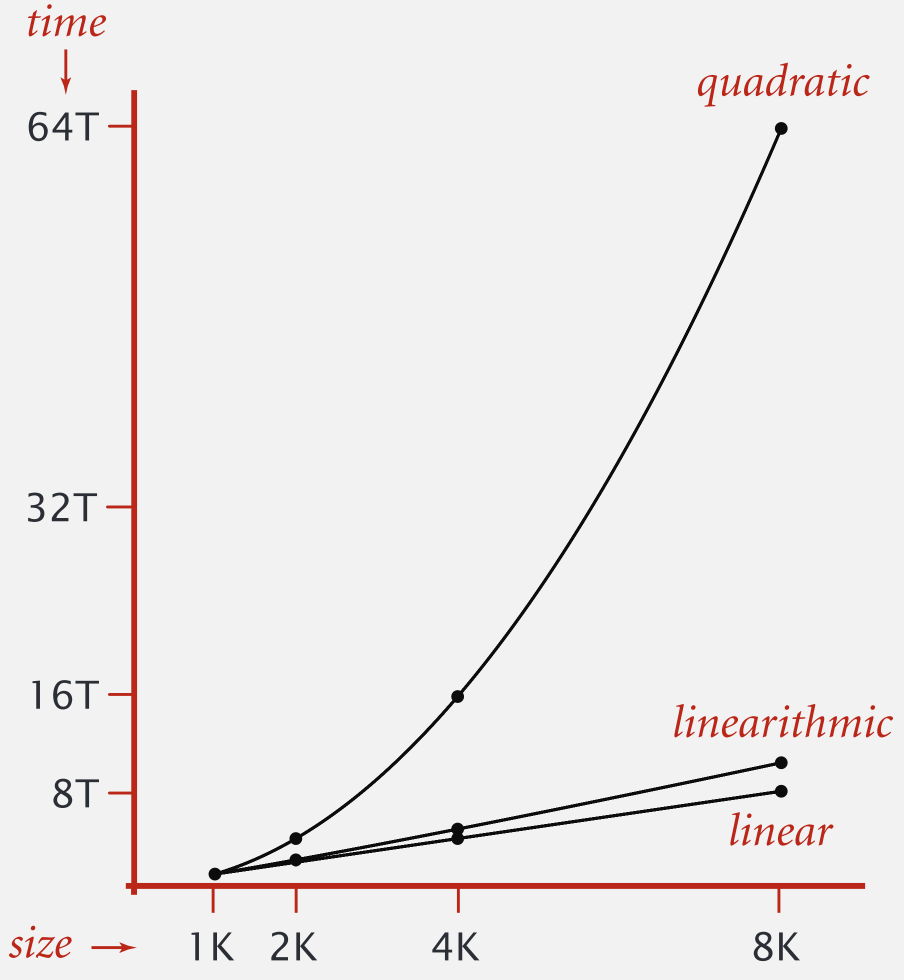


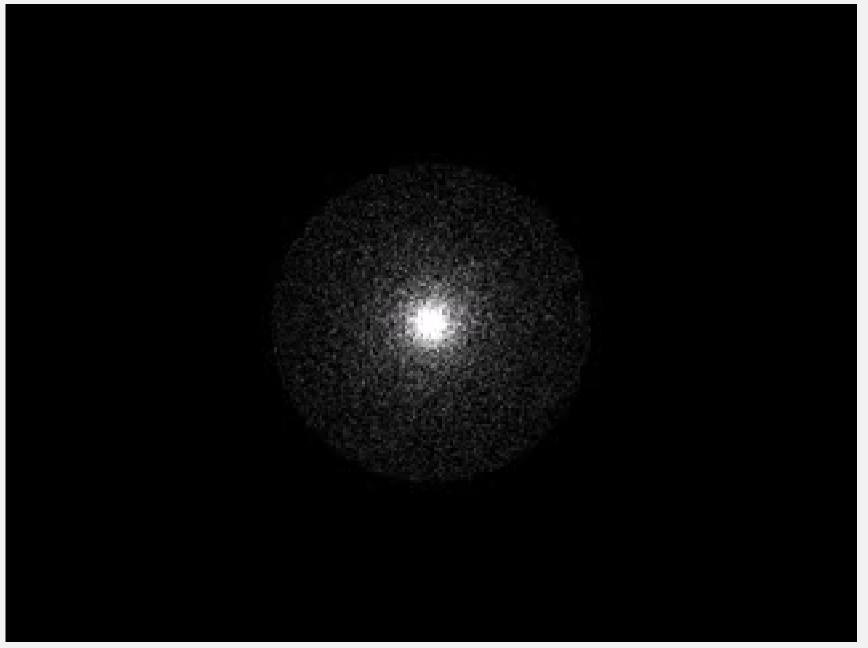
Some algorithmic successes

N-body simulation.

- Simulate gravitational interactions among N bodies.
- Brute force: N^2 steps.
- Barnes-Hut algorithm: $N \log N$ steps, enables new research.

Andrew Appel
PU '81





The challenge

Q. Will my program be able to solve a large practical input?

Insight. [Knuth 1970s] Use scientific method to understand performance.

Scientific method applied to analysis of algorithms

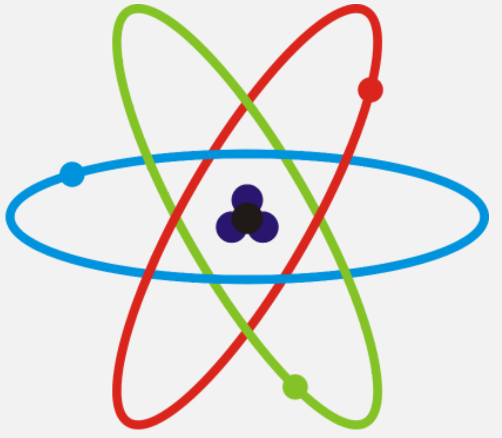
A framework for predicting performance and comparing algorithms.

Scientific method.

- Observe some feature of the natural world.
- Hypothesize a model that is consistent with the observations.
- Predict events using the hypothesis.
- Verify the predictions by making further observations.
- Validate by repeating until the hypothesis and observations agree.

Principles.

- Experiments must be reproducible.
- Hypotheses must be falsifiable.



Feature of the natural world. Computer itself.