Top Ten Algorithms Class 1

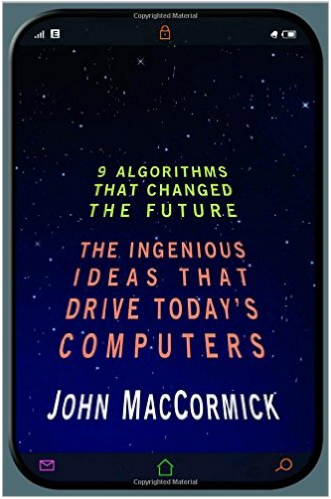
John Burkardt
Department of Scientific Computing
Florida State University

....

http://people.sc.fsu.edu/~jburkardt/classes/tta_2015/class01.pdf

24 August 2015

References



GUEST EDITORS' INTRODUCTION

the Top 10 Algorithms

In putting together this issue of *Computing in Science & Engineering*, we knew three things: it would be difficult to list just 10 algorithms; it would be fun to assemble the authors and read their papers; and, whatever we came up with in the end, it would be controversial. We tried to assemble the 10 algorithms with the greatest influence on the development and practice of science and engineering in the 20th century. Following is our list (here, the list is in chronological order; however, the articles appear in no particular order):

- Metropolis Algorithm for Monte Carlo
- Simplex Method for Linear Programming
- Krylov Subspace Iteration Methods
- The Decompositional Approach to Matrix Computations
- The Fortran Optimizing Compiler
- QR Algorithm for Computing Eigenvalues
- Quicksort Algorithm for Sorting
- Fast Fourier Transform
- Integer Relation Detection
- Fast Multipole Method

With each of these algorithms or approaches, there is a person or group receiving credit for inventing or discovering the method. Of course, the reality is that there is generally a culmination of ideas that leads to a method. In some cases, we chose authors who had a hand in developing the algorithm, and in other cases, the author is a leading authority.

1521-9615/00/$10.00 © 2000 IEEE

Jack Dongarra
University of Tennessee and Oak Ridge National Laboratory
Francis Sullivan
IDA Center for Computing Sciences

In this issue

Monte Carlo methods are powerful tools for evaluating the properties of complex, many-body systems, as well as nondeterministic processes. Isabel Beichl and Francis Sullivan describe the Metropolis Algorithm. We are often confronted with problems that have an enormous number of dimensions or a process that involves a path with many possible branch points, each of which is governed by some fundamental probability of occurrence. The solutions are not exact in a rigorous way, because we randomly sample the problem. However, it is possible to achieve nearly exact results using a relatively small number of samples compared to the problem's dimensions. Indeed, Monte Carlo methods are the only practical choice for evaluating problems of high dimensions.

John Nash describes the Simplex method for solving linear programming problems. (The use of the word *programming* here really refers to scheduling or planning—and not in the way that we tell a computer what must be done.) The Simplex method relies on noticing that the objective function's maximum must occur on a corner of the space bounded by the constraints of the "feasible region."

Large-scale problems in engineering and science often require solution of sparse linear algebra problems, such as systems of equations. The importance of iterative algorithms in linear algebra stems from the simple fact that a direct approach will require $O(N^3)$ work. The Krylov subspace iteration methods have led to a major change in how users deal with large, sparse, nonsymmetric matrix problems. In this article, Henk van der Vorst describes the state of the art in terms of

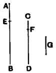
Euclid's Algorithm

PROPOSITION 2.

Given two numbers not prime to one another, to find their greatest common measure.

Let AB, CD be the two given numbers not prime to one another.

Thus it is required to find the greatest common measure of AB, CD.



If now CD measures AB—and it also measures itself—CD is a common measure of CD, AB.

And it is manifest that it is also the greatest; for no greater number than CD will measure CD.

But, if CD does not measure AB, then, the less of the numbers AB, CD being continually subtracted from the greater, some number will be left which will measure the one before it.

For an unit will not be left; otherwise AB, CD will be prime to one another [VII. 1], which is contrary to the hypothesis.

Therefore some number will be left which will measure the one before it.

Now let CD, measuring BE, leave EA less than itself, let EA, measuring DF, leave FC less than itself,

and let CF measure AE.

Since then, CF measures AE, and AE measures DF,

therefore CF will also measure DF.

But it also measures itself;

therefore it will also measure the whole CD.

But CD measures BE;

therefore CF also measures BE.

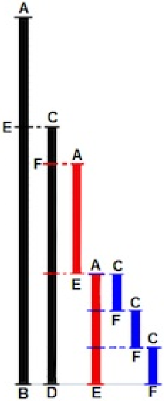
But it also measures EA;

therefore it will also measure the whole BA.

But it also measures CD;

therefore CF measures AB, CD.

Therefore CF is a common measure of AB, CD.



Euclid's example

Algorithm == al Khwarizmi

Kitab al jabr wal-muquabala (Rules of reuniting and reducing) by Abu Jafar Mohammed ibn Musa al-Khwarizmi, (9th century).

"al jabr" (=reunite) gave us **"algebra"**
"al-Kowarizm" gave us **algorithm**


كتاب الخوارزمي

Automation

Charles Babbage designs the Difference Machine and the Analytical Engine, to do arithmetic “by steam”, avoiding errors, and carrying out long, tedious computations.

Who wants to compute the Bernoulli numbers?

$$\left.\begin{array}{c} 0 = -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + \mathrm{B}_1 \left(\frac{2n}{2}\right) + \mathrm{B}_3 \left(\frac{2n\cdot(2n-1)\cdot(2n-2)}{2\cdot3\cdot4}\right) + \\ +\mathrm{B}_5 \left(\frac{2n\cdot(2n-1)\ldots(2n-4)}{2\cdot3\cdot4\cdot5\cdot6}\right) + \cdots + \mathrm{B}_{2n-1} \end{array}\right\}$$

Automation

The program for the Bernoulli computation:

Number of Operation	Nature of Operation	Variables acted upon	Variables receiving results	Indication of change in the value on any Variable	Statement of Results	Data $^1V_1$	Data $^1V_2$	Data $^1V_3$	Working Variables $^0V_4$	$^0V_5$	$^0V_6$	$^0V_7$	$^0V_8$	$^0V_9$	$^0V_{10}$	$^0V_{11}$	$^0V_{12}$	$^0V_{13}\cdots$	Result Variables $^1V_{21}$	$^1V_{22}$	$^1V_{23}$	$^0V_{24}\cdots$
						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
						0	0	0	0	0	0	0	0	0	0	0	0	0	B in a dec. fract.	B in a dec. fract.	B in a dec. fract.	0
						0	0	0	0	0	0	0	0	0	0	0	0	0				0
						1	2	4	0	0	0	0	0	0	0	0	0	0				0
						$\boxed{1}$	$\boxed{2}$	$\boxed{n}$											$\boxed{B_1}$	$\boxed{B_3}$	$\boxed{B_5}$	$\boxed{B_7}$
1	×	$^1V_2 \times {}^1V_3$	$^1V_4, {}^1V_5, {}^1V_6$	$^1V_2 = {}^1V_2$, $^1V_3 = {}^1V_3$	$= 2n$	......	2	n	$2n$	$2n$	$2n$											
2	−	$^1V_4 - {}^1V_1$	$^2V_4$	$^1V_4 = {}^2V_4$, $^1V_1 = {}^1V_1$	$= 2n - 1$	1	......	......	$2n-1$													
3	+	$^1V_5 + {}^1V_1$	$^2V_5$	$^1V_5 = {}^2V_5$, $^1V_1 = {}^1V_1$	$= 2n + 1$	1	......	......	......	$2n+1$												
4	÷	$^2V_5 \div {}^2V_4$	$^1V_{11}$	$^2V_5 = {}^0V_5$, $^2V_4 = {}^0V_4$	$= \frac{2n-1}{2n+1}$	......	......	......	0	0	......	......	......	......	......	$\frac{2n-1}{2n+1}$						
5	÷	$^1V_{11} \div {}^1V_2$	$^2V_{11}$	$^1V_{11} = {}^2V_{11}$, $^1V_2 = {}^1V_2$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$	......	2	......	......	......	......	......	......	......	......	$\frac{1}{2} \cdot \frac{2n-1}{2n+1}$						
6	−	$^0V_{13} - {}^2V_{11}$	$^1V_{13}$	$^2V_{11} = {}^0V_{11}$, $^0V_{13} = {}^1V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$	......	......	......	......	......	......	......	......	......	......	0	......	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$				
7	−	$^1V_3 - {}^1V_1$	$^1V_{10}$	$^1V_3 = {}^1V_3$, $^1V_1 = {}^1V_1$	$= n - 1 (= 3)$	1	......	n	......	......	......	......	......	......	$n-1$							
8	+	$^1V_2 + {}^0V_7$	$^1V_7$	$^1V_2 = {}^1V_2$, $^0V_7 = {}^1V_7$	$= 2 + 0 = 2$	......	2	......	......	......	......	2										
9	÷	$^1V_6 \div {}^1V_7$	$^3V_{11}$	$^1V_6 = {}^1V_6$, $^0V_{11} = {}^3V_{11}$	$= \frac{2n}{2} = A_1$	......	......	......	......	......	$2n$	2	......	......	......	$\frac{2n}{2} = A_1$						
10	×	$^1V_{21} \times {}^3V_{11}$	$^1V_{12}$	$^1V_{21} = {}^1V_{21}$, $^3V_{11} = {}^3V_{11}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$	......	......	......	......	......	......	......	......	......	......	$\frac{2n}{2} = A_1$	$B_1 \cdot \frac{2n}{2} = B_1 A_1$	......	B_1			
11	+	$^1V_{12} + {}^1V_{13}$	$^2V_{13}$	$^1V_{12} = {}^0V_{12}$, $^1V_{13} = {}^2V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$	......	......	......	......	......	......	......	......	......	......	......	0	$\left\{-\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}\right\}$				
12	−	$^1V_{10} - {}^1V_1$	$^2V_{10}$	$^1V_{10} = {}^2V_{10}$, $^1V_1 = {}^1V_1$	$= n - 2 (= 2)$	1	......	......	......	......	......	......	......	......	$n-2$							
13	−	$^1V_6 - {}^1V_1$	$^2V_6$	$^1V_6 = {}^2V_6$, $^1V_1 = {}^1V_1$	$= 2n - 1$	1	......	......	......	......	$2n-1$											
14	+	$^1V_1 + {}^1V_7$	$^2V_7$	$^1V_1 = {}^1V_1$, $^1V_7 = {}^2V_7$	$= 2 + 1 = 3$	1	......	......	......	......	......	3										
15	÷	$^2V_6 \div {}^2V_7$	$^1V_8$	$^2V_6 = {}^2V_6$, $^2V_7 = {}^2V_7$	$= \frac{2n-1}{3}$	......	......	......	......	......	$2n-1$	3	$\frac{2n-1}{3}$									
16	×	$^1V_8 \times {}^3V_{11}$	$^4V_{11}$	$^1V_8 = {}^0V_8$, $^3V_{11} = {}^4V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3}$	......	......	......	......	......	......	......	0	......	......	$\frac{2n}{2} \cdot \frac{2n-1}{3}$						
17	−	$^2V_6 - {}^1V_1$	$^3V_6$	$^2V_6 = {}^3V_6$, $^1V_1 = {}^1V_1$	$= 2n - 2$	1	......	......	......	......	$2n-2$											
18	+	$^1V_1 + {}^2V_7$	$^3V_7$	$^2V_7 = {}^3V_7$, $^1V_1 = {}^1V_1$	$= 3 + 1 = 4$	1	......	......	......	......	......	4										
19	÷	$^3V_6 \div {}^3V_7$	$^1V_9$	$^3V_6 = {}^3V_6$, $^3V_7 = {}^3V_7$	$= \frac{2n-2}{4}$	......	......	......	......	......	$2n-2$	4	......	$\frac{2n-2}{4}$	......							
20	×	$^1V_9 \times {}^4V_{11}$	$^5V_{11}$	$^1V_9 = {}^0V_9$, $^4V_{11} = {}^5V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = A_3$	......	......	......	......	......	......	......	......	0		$\left\{\frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4}\right\} = A_3$						
21	×	$^1V_{22} \times {}^5V_{11}$	$^0V_{12}$	$^1V_{22} = {}^1V_{22}$, $^0V_{12} = {}^2V_{12}$	$= B_3 \cdot \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = B_3 A_3$	......	......	......	......	......	......	......	......	......	......	0	$B_3 A_3$	......	......	B_3		
22	+	$^2V_{12} + {}^2V_{13}$	$^3V_{13}$	$^2V_{12} = {}^0V_{12}$, $^2V_{13} = {}^3V_{13}$	$= A_0 + B_1 A_1 + B_3 A_3$	......	......	......	......	......	......	......	......	......	......	......	0	$\{A_0 + B_1 A_1 + B_3 A_3\}$				
23	−	$^2V_{10} - {}^1V_1$	$^3V_{10}$	$^2V_{10} = {}^3V_{10}$, $^1V_1 = {}^1V_1$	$= n - 3 (= 1)$	1	......	......	......	......	......	......	......	......	$n-3$							
					Here follows a repetition of Operations thirteen to twenty-three																	
24	+	$^4V_{13} + {}^0V_{24}$	$^1V_{24}$	$^4V_{13} = {}^0V_{13}$, $^0V_{24} = {}^1V_{24}$	$= B_7$	......	......	......	......	......	......	......	......	......	......	......	......	......	......	......	......	B_7
25	+	$^1V_1 + {}^1V_3$	$^1V_3$	$^1V_1 = {}^1V_1$, $^1V_3 = {}^1V_3$, $^5V_6 = {}^0V_6$, $^5V_7 = {}^0V_7$	$= n + 1 = 4 + 1 = 5$ by a Variable-card. by a Variable-card.	1	......	$n+1$	......	......	0	0										

What is an Algorithm

An algorithm has been characterized as a precisely defined finite sequence of steps that efficiently are guaranteed to produce the correct answer to a numerical problem.

But algorithms are no longer just rules for long division!

- not precisely defined!
- not finite!
- not a sequence!
- not efficient!
- not guaranteed!
- not correct!
- not numerical!

An algorithm is **a plan (which can be put into words) for dealing with some class of problems**.

Top Ten Algorithms of the 20th Century

- Metropolis Algorithm for Monte Carlo
- Simplex Method for Linear Programming
- Krylov Subspace Iteration Methods
- The Decompositional Approach to Matrix Computations
- The Fortran Optimizing Compiler
- QR Algorithm for Computing Eigenvalues
- Quicksort Algorithm for Sorting
- Fast Fourier Transform
- Integer Relation Detection
- Fast Multipole Method

Top 10 Scientific Algorithms of 20th Century

1. Metropolis Algorithm/ Monte Carlo method (von Neumann, Ulam, Metropolis, 1946). Through the use of random processes, this algorithm offers an efficient way to stumble toward answers to problems that are too complicated to solve exactly.

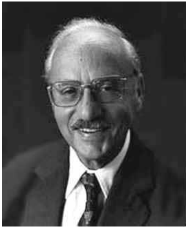
- **Approximate solutions to numerical problems with too many degrees of freedom.**
- **Approximate solutions to combinatorial optimization problems.**
- **Generation of random numbers.**

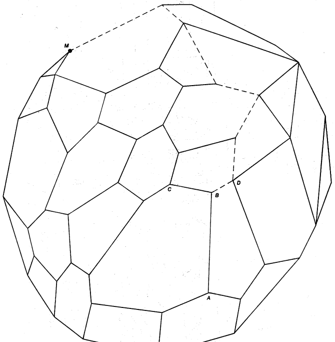
From: Kevin Wayne, Princeton

Top 10 Scientific Algorithms of 20th Century

2. Simplex Method for Linear Programming (Dantzig 1947).
An elegant solution to a common problem in planning and decision-making: max {cx : Ax ≤ b, x ≥ 0}.

- **One of most successful algorithms of all time.**
- **Dominates world of industry.**





Top 10 Scientific Algorithms of 20th Century

3. Krylov Subspace Iteration Method (Hestenes, Stiefel, Lanczos, 1950).
A technique for rapidly solving Ax = b where A is a huge n x n matrix.

- **Conjugate gradient method for symmetric positive definite systems.**
- **GMRES, CGSTAB for non-symmetric systems.**

Compute $r^{(0)} = b - Ax^{(0)}$ for some initial guess $x^{(0)}$
for $i = 1, 2, \ldots$
  solve $Mz^{(i-1)} = r^{(i-1)}$
  $\rho_{i-1} = r^{(i-1)^T} z^{(i-1)}$
  if $i = 1$
    $p^{(1)} = z^{(0)}$
  else
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}$
    $p^{(i)} = z^{(i-1)} + \beta_{i-1} p^{(i-1)}$
  endif
  $q^{(i)} = Ap^{(i)}$
  $\alpha_i = \rho_{i-1}/p^{(i)^T} q^{(i)}$
  $x^{(i)} = x^{(i-1)} + \alpha_i p^{(i)}$
  $r^{(i)} = r^{(i-1)} - \alpha_i q^{(i)}$
  check convergence; continue if necessary
end

Preconditioned Conjugate Gradient

From: Kevin Wayne, Princeton

Top 10 Scientific Algorithms of 20th Century

4. Decompositional Approach to Matrix Computations (Householder, 1951). A suite of technique for numerical linear algebra that led to efficient matrix packages.

- **Factor matrices into triangular, diagonal, orthogonal, tri-diagonal, and other forms.**
- **Analysis of rounding errors.**
- **Applications to least squares, eigenvalues, solving systems of linear equations.**
- **LINPACK, EISPACK.**

From: Kevin Wayne, Princeton

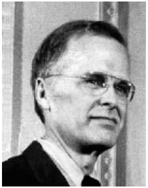
Top 10 Scientific Algorithms of 20th Century

5. Fortran Optimizing Compiler (Backus, 1957). Turns high-level code into efficient computer-readable code.

- **Among single most important events in history of computing: scientists could program computer without learning assembly.**

Fortran Code

```
500    C = 0.0
C   *** START LOOP ***
       DO 540 I=L,K
          F = S*RV1(I)
          RV1(I) = C*RV1(I)
          IF (ABS(F).LE.EPS) GO TO 550
          G = W(I)
          H = SQRT(F*F+G*G)
          W(I) = H
          C = G/H
          S = -F/H
510    CONTINUE
```



Top 10 Scientific Algorithms of 20th Century

6. QR Algorithm for Computing Eigenvalues (Francis 1959). **Another crucial matrix operation made swift and practical.**

- Eigenvalues are arguably most important numbers associated with matrices.

$$\mathbf{A}\,\mathbf{x} = \lambda\,\mathbf{x}$$

- Differential equations, population growth, building bridges, quantum mechanics, Markov chains, web search, graph theory.

QR(A)

```
Initialize A0 = A

FOR k = 0, 1, 2, ...
   Factor  Ak   = Qk Rk
   Compute Ak+1 = Rk Qk
```

$$\begin{aligned} A_{k+1} &= R_k\, Q_k \\ &= Q_k^{-1}\, Q_k\, R_k\, Q_k \\ &= Q_k^{-1}\, A_k\, Q_k \end{aligned}$$

$$\Rightarrow\ A_{k+1} \text{ and } A_k \text{ have same eigenvalues}$$

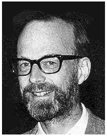
Under fairly general conditions, A_k converges to diagonal or upper triangular matrix with eigenvalues on main diagonal.

From: Kevin Wayne, Princeton

Top 10 Scientific Algorithms of 20th Century

7. Quicksort (Hoare, 1962). Given N items over a totally order universe, rearrange them in increasing order.

- **O(N log N) instead of O(N^2).**
- **Efficient handling of large databases.**



8. Fast Fourier Transform (Cooley, Tukey 1965). Perhaps the most ubiquitous algorithm in use today, it breaks down waveforms (like sound) into periodic components.

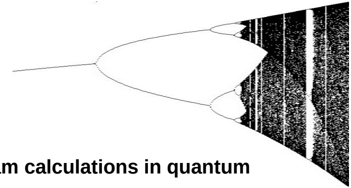
- **O(N log N) instead of O(N^2).**

Top 10 Scientific Algorithms of 20th Century

9. Integer Relation Detection (Ferguson, Forcade, 1977).
Given real numbers $x_1, \ldots, x_n$, find integers $a_1, \ldots, a_n$ (not all 0 if they exist) such that $a_1 x_1 + \ldots + a_n x_n = 0$?

- **PSLQ algorithm generalizes Euclid's algorithm: special case when n = 2.**
- **Find coefficients of polynomial satisfied by 3rd and 4th bifurcation points of logistic map.**

$$x_{n+1} = a\, x_n (1 - x_n)$$



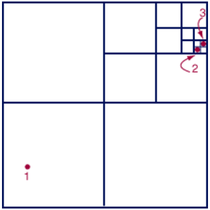
- **Simplify Feynman diagram calculations in quantum field theory.**
- **Compute nth bit of π without computing previous bits.**
- **Experimental mathematics.**

From: Kevin Wayne, Princeton

Top 10 Scientific Algorithms of 20th Century

10. Fast Multipole Method (Greengard, Rokhlin, 1987). Accurate calculations of the motions of N particles interacting via gravitational or electrostatic forces.

- **Central problem in computational physics.**
- **O(N) instead of O(N^2).**
- **Celestial mechanics, protein folding, etc.**



A Quad-Tree

Reactions to Dongarra/Sullivan List

- You left out my field of interest (data mining)
- These are very "20th Century" algorithms.
- Fortran is not an algorithm!
- This list is boring!
- The algorithms we use every day aren't here.

Dan Givoli

http://msvlab.hre.ntou.edu.tw/article/iacm.pdf

1. the Finite Element Method
2. Iterative Linear Algebraic Solvers
3. Algebraic Eigenvalue Solvers
4. Matrix Decomposition Methods
5. Finite Difference Methods for Wave Problems
6. Nonlinear Algebraic Solvers
7. Fast Fourier Transform
8. Nonlinear Programming
9. Soft Computing Methods (neural networks, genetic algorithms, fuzzy logic)
10. Multiscale Methods

Datamining List

http://www.cs.umd.edu/ samir/498/10Algorithms-08.pdf

1. C4.5
2. k-means clustering
3. Support vector machines
4. The Apriori algorithm
5. The EM algorithm (expectation maximization)
6. PageRank
7. AdaBoost (ensemble learning)
8. kNN: k-nearest neighbors classification
9. Naive Bayes
10. CART: Classification and Regression Trees

Otero

https://medium.com/@_marcos_otero/the-real-10-algorithms-that-dominate-our-world-e95fa9f16c04

1. Merge Sort, Quick Sort, Heap Sort
2. Fourier Transform
3. Dijkstra's Algorithm (shortest path on network)
4. RSA encryption
5. SHA, Secure Hash Algorithm
6. Integer Factorization
7. Link Analysis
8. Proportional Integral Derivative (feedback control)
9. Data compression
10. Random Number Generation

http://io9.com/the-10-algorithms-that-dominate-our-world-1580110464

1. Google Search
2. Facebook News Feed
3. OKCupid Date Matching
4. NSA Data collection, interpretation, encryption
5. "You May Also Enjoy..."
6. Google AdWords
7. High Frequency Stock Trading
8. MP3 compression
9. IBM's CRUSH (Crime Reduction Using Statistical History)
10. Auto-Tune

MacCormick's List

http://press.princeton.edu/titles/9528.html

1. Search Engine Indexing
2. PageRank
3. Public Key Cryptography
4. Error-Correcting Codes
5. Pattern Recognition
6. Data Compression
7. Database Consistency
8. Digital Signatures

Let's Look Again

Consider a new list: "Top Ten Algorithms of the 21st Century"

There are many real life examples we could explore:

- Credit card fraud detectors
- Driverless vehicles
- Fingerprint matching
- GPS and maps
- Protein unfolding prediction
- Speech recognition

What about algorithms from biology, physics, chemistry, graphics, simulation, game design?

There are many algorithms associated with the Oculus Rift.

Weekly Course Plan

Each week, I expect we will consider:

- Non-numerical algorithm (MacCormick?)
- 5 minute presentation by a student
- Numerical algorithm (Sullivan/Givoli?)

Overall Course Plan

By the end of the semester:

- Every student will have made a presentation;
- Every student will submit a proposed top ten list;
- We'll construct a final 10 ten algorithms list;
- We'll make a poster of our list to hang in 499.

Next Week

The FSU libraries have 3,000,000 books.

How can I determine which books:

- contain the word **multipole**?
- contain the words **multipole** and **n-body**?
- contain the words **multipole** and **n-body** in proximity?
- are probably about multipole n-body problems?

Can these questions be answered:

- quickly?
- exactly?
- approximately?

What would be a good algorithm (a plan, to solve this problem?)

Next Week (Student volunteer?)

https://www.youtube.com/watch?v=kk-_DDgoXfk

Bryan Hayes, "Sorting out the genome", American Scientist.

You have a stack of pancakes, of different sizes.

You want to sort the stack so it runs from largest to smallest.

You have a spatula which you can insert into the stack, flipping the order of all the pancakes above the spatula.

- Can you sort them? (of course!)
- Is there a way to organize this operation?
- What is the most difficult stack to sort?
- For an arbitrary stack of N pancakes, what is the most number of flips needed?