Mathematics in Computer Science Curricula

Jeannette M. Wing

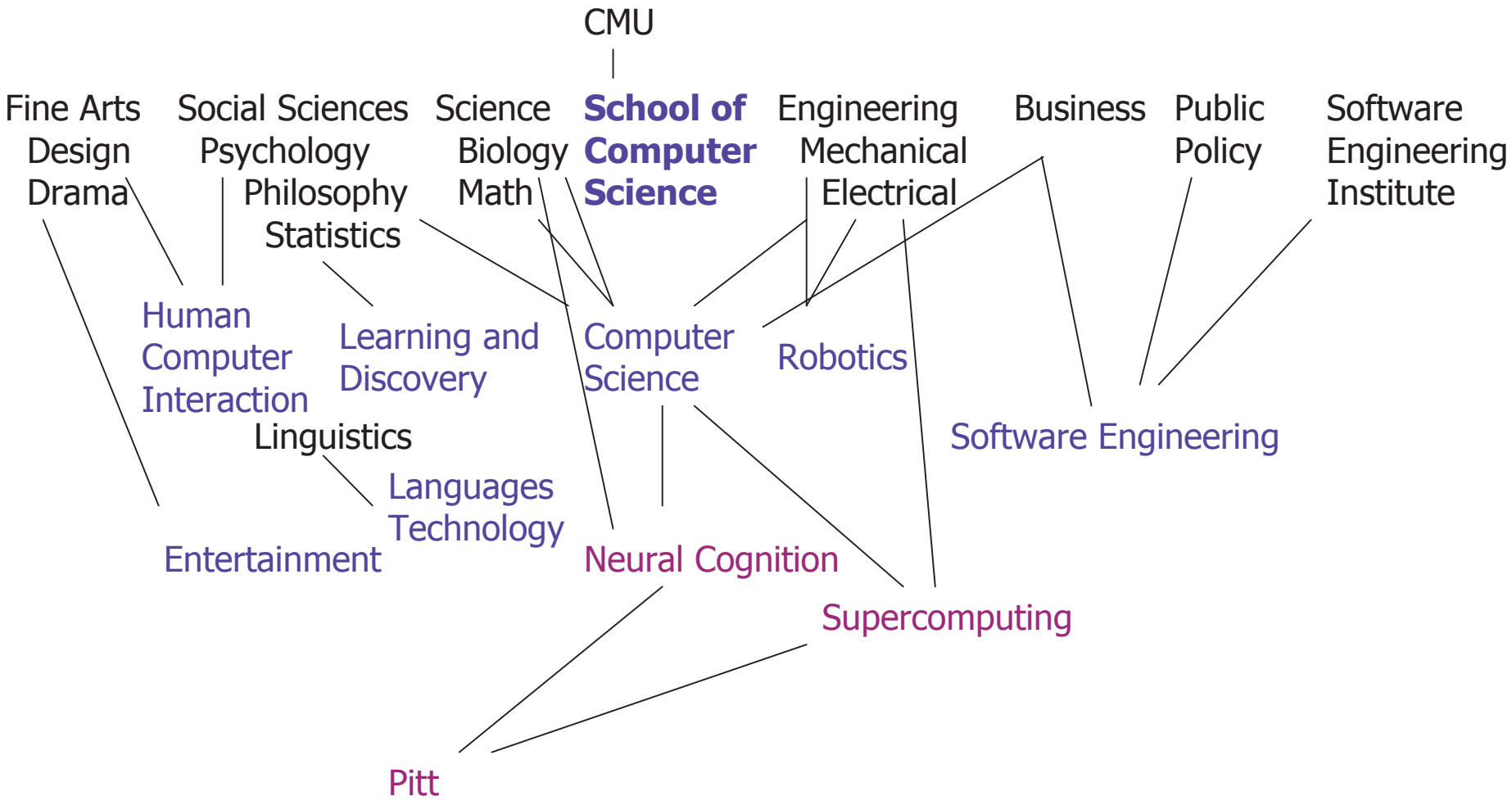
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Sixth International Conference on Mathematics of Program Construction
July 2002, Dagstuhl, Germany

Prelude: Three Observations

- Linear Algebra and Probability & Statistics are increasingly important to Computer Scientists.
- As Computer Science matures, more mathematics enters CS curricula in different guises.
- As Computer Science matures, more course material covering mathematically-based concepts moves from the graduate to the undergraduate level.

Computing at Carnegie Mellon



Some Numbers

- 160 faculty
- 200 courses offered
- 270 doctoral students in 6 Ph.D. programs
- 200 masters students in 8 MS programs
- 540 bachelors students in 1 BS program

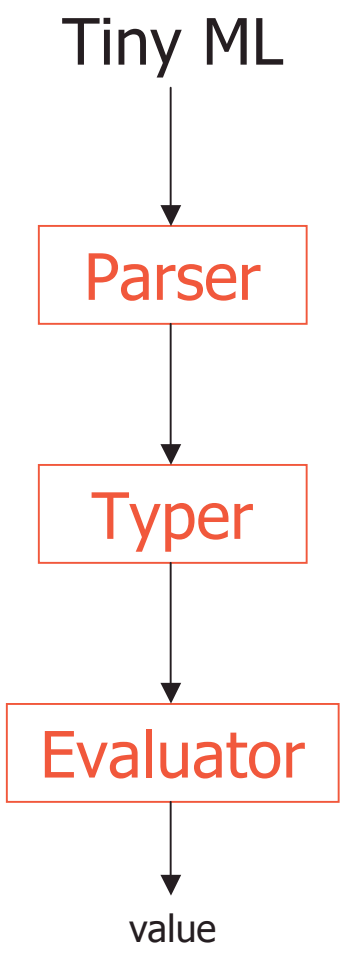
- "Computer" Mellon University (4000 undergrad, 2500 grad)
  - 100 CS minors
  - 400 additional computer or IT-related undergrad majors
  - 350-450 computer or IT-related masters students

CMU named "Most Wired Campus" by Yahoo Internet Life

Mathematics for Program Construction

		Why in CS?	Courses (UG)
Discrete Math	*Algebraic structures:* groups, rings, fields, graphs, … *Algebraic properties:* commutativity, associativity, idempotency, … *Combinatorics:* counting, summation, permutation, …	data structures algorithms state machines	Data Structures Algorithms Prog. Languages Object-Oriented Prog. Compilers Machine Architecture Operating Systems
Logic	*Logics:* propositional, predicate logics; first-order, higher-order, … *Proof techniques:* induction, deduction, contradiction, case analysis, reduction, diagonalization, pigeonhole principle, pointwise principle, …	invariants pre/post conditions recursion symbolic computation specification/verification	Prog. Principles Functional Prog. Formal Languages Automata Theory Complexity Artificial Intelligence Databases Software Eng.

15-212 Programming Principles: Homework 6



Yes: polymorphic recursive datatypes, recursive functions, pattern matching.
No: mutual recursion/datatype declarations,

$$\frac{}{\Gamma \vdash \texttt{E\_INT}(i) : \texttt{T\_INT}}\ [\text{INT}]$$

$$\frac{}{\Gamma \vdash \texttt{E\_BOOL}(b) : \texttt{T\_BOOL}}\ [\text{BOOL}]$$

$$\frac{}{\Gamma \vdash \texttt{E\_UNIT} : \texttt{T\_UNIT}}\ [\text{UNIT}]$$

$$\frac{}{\Gamma \vdash \texttt{E\_PLUS} : \texttt{T\_FUN}(\texttt{T\_PAIR}(\texttt{T\_INT}, \texttt{T\_INT}), \texttt{T\_INT})}\ [\text{PLUS}]$$

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash \texttt{E\_VID}(x, \_) : \tau}\ [\text{VID}]$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \texttt{E\_PAIR}(e_1, e_2) : \texttt{T\_PAIR}(\tau_1, \tau_2)}\ [\text{PAIR}]$$

$$\frac{\Gamma \vdash d \rightsquigarrow \Gamma' \quad \Gamma' \vdash e : \tau \quad TN(\tau) \subseteq TN(\Gamma)}{\Gamma \vdash \texttt{E\_LET}(d, e) : \tau}\ [\text{LET}]$$

$$\frac{\Gamma \vdash e_1 : \texttt{T\_BOOL} \quad \Gamma \vdash e_2 : \tau \quad \Gamma \vdash e_3 : \tau}{\Gamma \vdash \texttt{E\_IF}(e_1, e_2, e_3) : \tau}\ [\text{IF}]$$

$$\frac{\Gamma \vdash e_1 : \texttt{T\_FUN}(\tau_1, \tau_2) \quad \Gamma \vdash e_2 : \tau_1}{\Gamma \vdash \texttt{E\_APP}(e_1, e_2) : \tau_2}\ [\text{APP}]$$

$$\frac{\Gamma \vdash m : \tau}{\Gamma \vdash \texttt{E\_FUN}\ m : \tau}\ [\text{FUN}]$$

$$\frac{\Gamma \vdash e : \tau' \quad \Gamma \vdash \tau \rightsquigarrow \tau'}{\Gamma \vdash \texttt{E\_TEXP}(e, \tau) : \tau'}\ [\text{TEXP}]$$

$$\frac{}{\Gamma \vdash \texttt{M\_MATCH}(\texttt{nil}) : \texttt{T\_FUN}(\tau_1, \tau_2)}\ [\text{MATCH1}]$$

$$\frac{\Gamma \vdash p \rightsquigarrow (\Gamma', \tau_1) \quad \Gamma' \vdash e : \tau_2 \quad \Gamma \vdash \texttt{M\_MATCH}(L) : \texttt{T\_FUN}(\tau_1, \tau_2)}{\Gamma' \vdash \texttt{M\_MATCH}((p, e) :: L) : \texttt{T\_FUN}(\tau_1, \tau_2)}\ [\text{MATCH2}]$$

Mathematics for ~~Program~~ Construction

Computing **Systems**

(Hardware and Software)

Mathematics for Computing Systems Construction

Discrete Math, Logic,

and

Linear Algebra

Gaussian elimination
<
vector spaces, matrices, eigenvalues, eigenvectors, linear transformations, orthogonalization, determinants, …
<
Abstract algebra

Why in CS?	Courses
linear programming matrices clustering numerical methods	Algorithms Artificial Intelligence Robotics Graphics Vision

1

Probability & Statistics

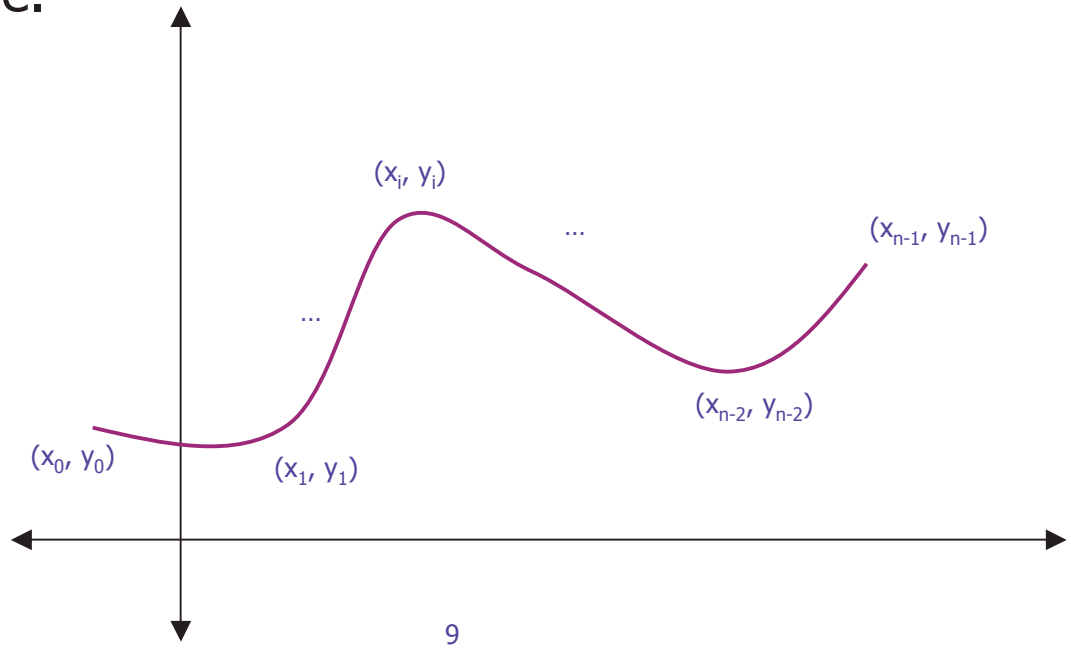
Prob: random processes, conditional probability, distribution functions, limit theorems, …
Stats: confidence intervals, estimation, regression, variance analysis, Bayesian inference, …

Why in CS?	Courses
randomization performance analysis queueing theory scheduling planning learning	Algorithms Operating Systems Networking Artificial Intelligence Speech

2

15-451 Algorithms: Homework 6

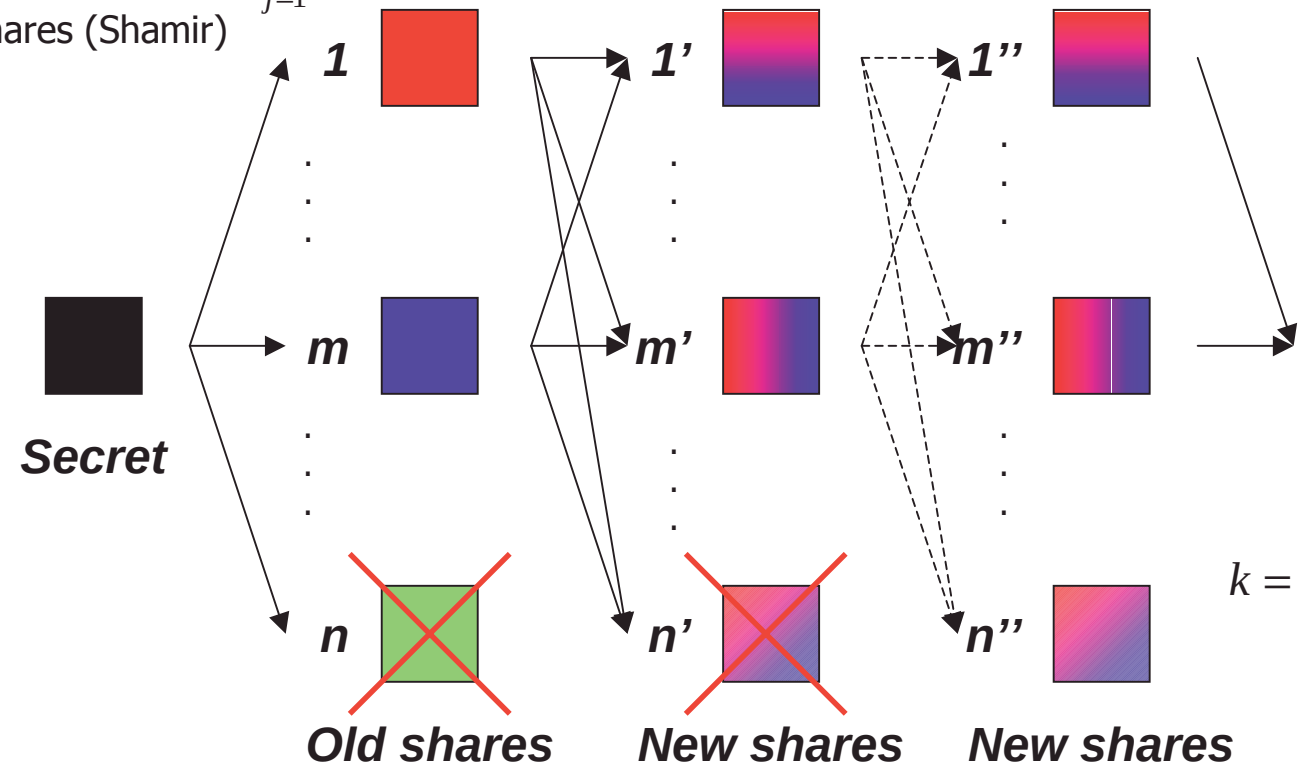
Theorem: Given n pairs (x_0, y_0), …, (x_{n-1}, y_{n-1}) there is a unique polynomial A(x) of degree < n that passes through all these points. I.e., $A(x_i) = y_i$ for i = 0, …, n-1. Complete the proof of this theorem by solving […]. Your job is to show that the determinant of the Vandermonde matrix is nonzero so long as all the x_i's are distinct, which then implies that the matrix is invertible.



Wong's Ph.D. Thesis: Verifiable Secret Redistribution

$$\forall i \in P : s_i = k + \sum_{j=1}^{m-1} a_j i^j$$

Create shares (Shamir)

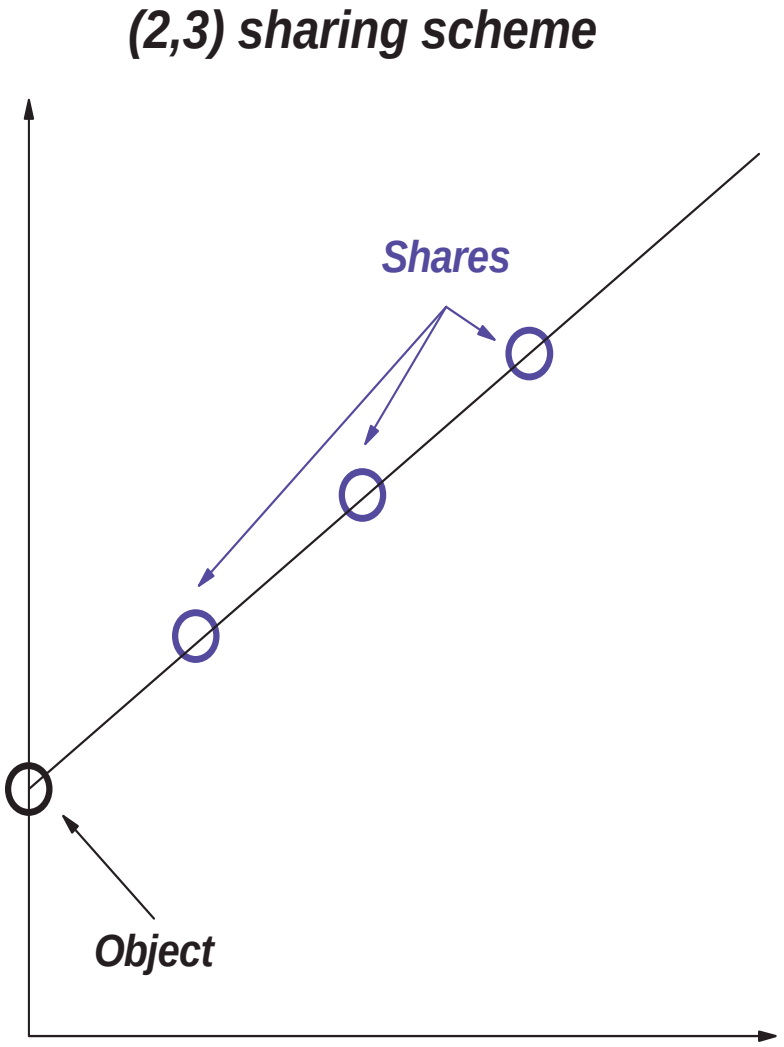


$$k = \sum_{i=1}^{m} b_i s_i ; b_i = \prod_{j=1}^{m} \frac{j}{j-i}$$

Reconstruct secret

Proof of security of protocol relies on previous theorem.

Shamir's Linear Threshold Scheme



- n shares;
  m reconstruct secret:

$$|P| = n; |B| = m$$

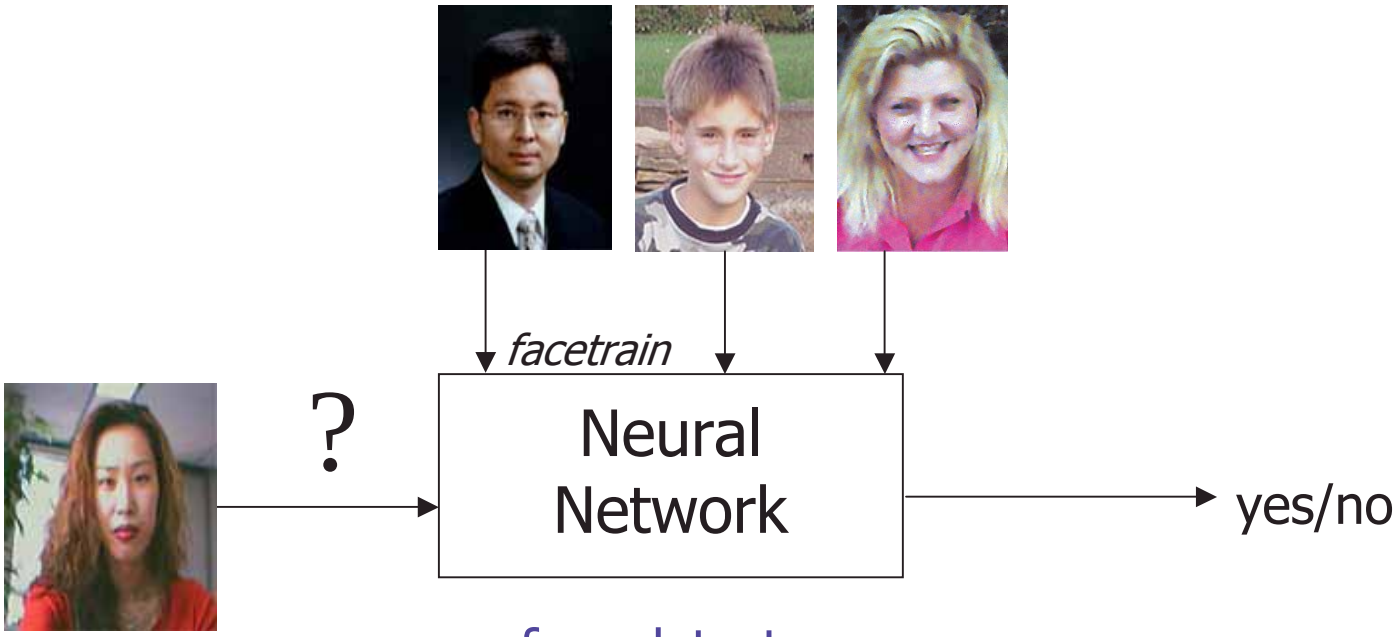
- Create shares:

$$\forall i \in P : s_i = k + \sum_{j=1}^{m-1} a_j i^j$$

- Reconstruct secret:

$$k = \sum_{i=1}^{m} b_i s_i ; b_i = \prod_{j=1}^{m} \frac{j}{j-i}$$

15-681 Machine Learning: Homework 3



face detector

pose recognizer

wearing sunglasses?

Learning continuous valued functions in context of clustering, classification, locally weighted regression, ...

Importance of Calculus: An Aside

- Calculus is a good means for introducing and reinforcing mathematical rigor.
  - Definitions, proofs, problem solving
- Both differential and integral calculus are important and useful.
- Multivariate calculus is more directly relevant than calculus of approximation to computer scientists.
  - Robotics, graphics, vision

Observation #1

- Discrete Math and Logic are essential for CS.
- But don't forget the importance of Linear Algebra and Probability & Statistics.

Interlude

Four Types (undergrad and grad levels)

1. Plain Old Math
   - Calculus, Discrete Math, Logic, Linear Algebra, Probability, Statistics.

2. CS with lots of Math — 45/200 (23%)
   - Algorithms, Data Structures, Semantics, …

3. *Math adapted for CS* — 3
   - Probability and Statistics for Computer Science, …

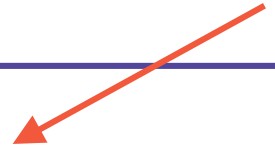
4. *Computational X* — 3, 27

   where X = branch of Math/Science/Engineering
   - Computational Geometry, …

Type 2: CS with Lots of Math

- Core CS
  - Algorithms, Automata Theory, Complexity, Data Structures, Principles of Programming, Programming Languages, Semantics
- Applications
  - Cognitive Modeling, Cryptography, Data Mining, Electronic Commerce, Graphics, Machine Learning, Natural Language Processing, Performance Modeling, Robotics, Speech, Vision, …
- Meta-level: CS with Lots of Math for Applications
  - Algorithms in the Real World
  - Algorithms for Natural Language Processing
  - Algorithms for Knowledge Discovery and Data Mining
  - …

Type 3: Math for CS



- Intro to Feedback Control Systems (UG)
  - dynamic response, feedback control, time and frequency domain analysis, Laplace transforms, state-space design, digital control, robotic control, force feedback robotic devices.

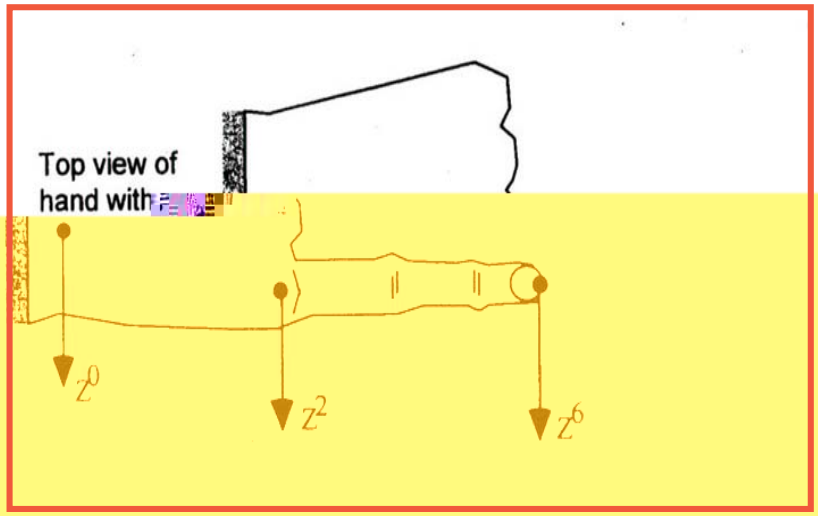
- Math Fundamentals for Robotics (G)
  - polynomial interpolation and approximation, solution of nonlinear equations, roots of polynomials, resultants, solution of linear equations, approximation by orthogonal functions (includes Fourier series), integration of ordinary differential equations, optimization, calculus of variations (with applications to mechanics), probability and stochastic processes (Markov chains), computational geometry.

- Probability and Statistics for Computer Scientists (G)
  - Probability and random variables, estimation, special distributions and sampling, non-parametric methods, Markov chains, queues, experimental design, numerical algorithms.

16-299 Intro to Feedback Control Systems: Homework 3

Find the forward kinematics of the human hand from the wrist through the tip of the index finger.

Assume that the forearm is rigidly fixed to form the "base" of the manipulator. There are then a total of six degrees of freedom: two at the wrist, two at the knuckle, and one at each of the finger joints. Do not include wrist roll (i.e., rotation about the forearm), which is actually located at the elbow joint rather than the wrist.

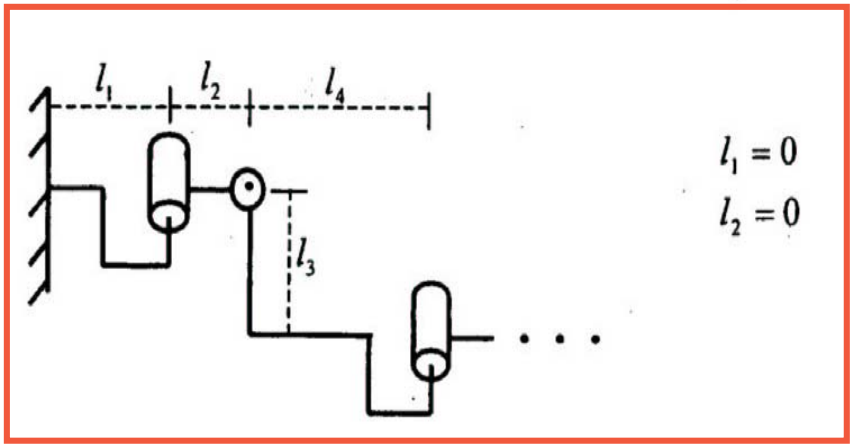


...

b. Assign the z-axes for all coordinate frames (0 to 6) following the Denevit-Hartenberg algorithm we used in class.

...

f. Find the homogeneous transformation for each link, A01, A12, etc.



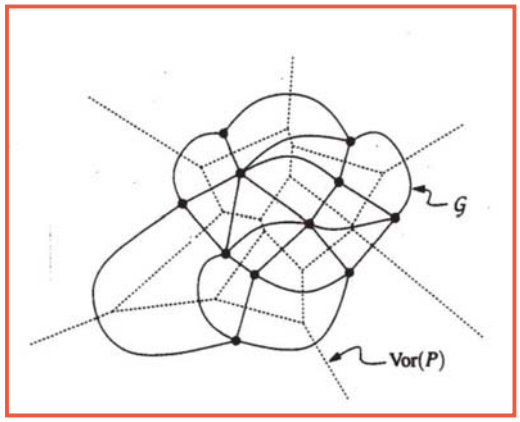
Type 4: Computational X Offered in CS

- Computational Algebra (UG)
  - recursion and the algebra of generating functions, covering problems and polynomial equations, algebra and geometry of complex numbers and complex functions, logical functions as ordinary polynomials relative to their values on {0,1}, iteration and closure as exemplified by cellular automata, polynomial interpolation and n-valued functions, tables and Boolean matrices to implement relational products, permutations and equivalence relations, modular arithmetic, exponential behavior and quadradic reciprocity, computing in finite fields.

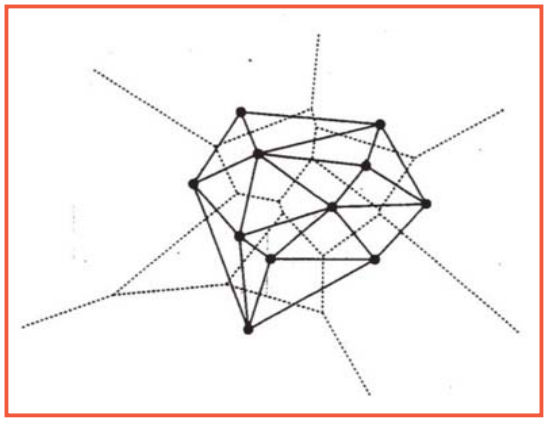
- Computational Geometry (G)
  - geometric primitives, line intersection plus randomized incrementals, triangulation and visibility, linear programming in two and three dimensions, orthogonal range searching, point location and Binary Space Partitions, Voronoi diagrams and Delaunay triangulation, convex hulls, non-orthogonal range searching.
- Scientific Computing (G)
  - linear least squares, eigenvalues, eigenvectors, Newton's method, Gaussian quadrature, initial value problems (Euler's method, Runge-Kutta) and boundary value problems (finite difference methods, finite element methods) for ordinary differential equations, partial differential equations (applications in stress analysis, heat diffusion, fluid flow, radiation, computer graphics), wavelets, mesh generation (Delaunay triangulation), variational methods.

15-859 Computational Geometry: Homework 2

Trading areas of the capitals of the twelve provinces in the Netherlands, as predicted by the Voronoi assignment model.



Dual graph of Vor(P).



Delaunay graph DG(P)

2. A Euclidean Minimum Spanning Tree of a set of points P in the plane is a tree T with vertex set P which minimizes the total edge length over all such trees.

Give an O(n log n) algorithm to find a EMST of a set of n points. Hint: Show that the edge of a Delaunay triangulation of P contains an EMST of P.

Fact: The EMST of a given set P of points is a subgraph of the Delaunay triangulation of P.

Theorem: The Delaunay triangulation of a set P of n points in the plane can be computed in O(n log n) expected time, using O(n) expected storage..

Computational X Offered in Other Disciplines

- Business
  - Mathematical Modeling for Consulting, *Computational Finance* (degree program)
- Chemical Engineering
  - Computational Methods for Large Scale Process Design and Analysis
- Civil Engineering
  - Introduction to Computer Applications in Civil and Environmental Engineering
- Electrical and Computer Engineering
  - Mathematical Software in Engineering + 6 Computer Engineering courses
- Mathematics
  - Introduction to Mathematical Software, Maple Lab, Modeling with Differential Equations
- Philosophy
  - Logic and Computation, Computability and Incompleteness, Modal Logic, Probability and Artificial Intelligence, Constructive Logic, Game Theory, Recursion and Hierarchies, Proof Theory, Intuitionism and Constructive Mathematics, Category Theory, Cognitive Architecture and Bayesian Networks
- Physics
  - Introduction to Computational Physics, Advanced Computational Physics

NOT included is any course that simply requires or teaches computing skills or technology.

Observation #2: More Math in CS Curricula

- As subfields of Computer Science mature
  - there is a growing trend of seeing more math in CS curricula via a range of course types.

- Meta-level: As other disciplines turn to computing as a tool for research
  - there is a growing trend of seeing more computational methods used and taught in those disciplines.

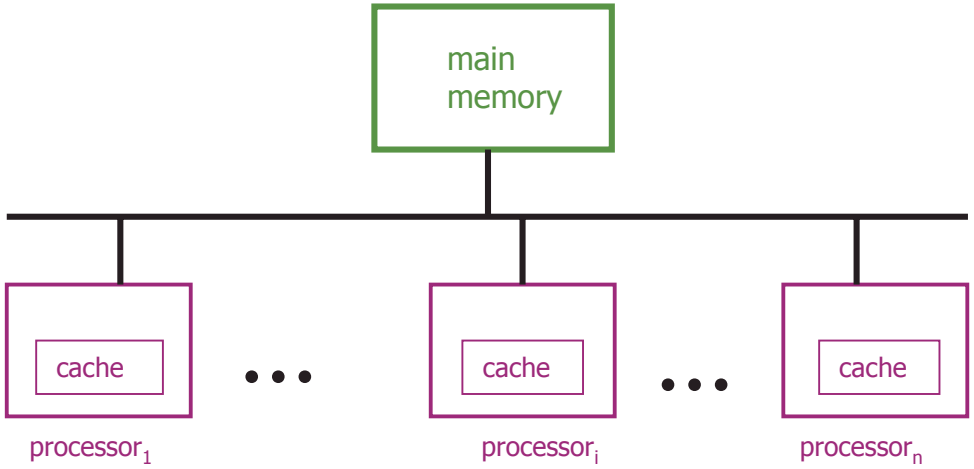
Interlude

Evolution of Course Material

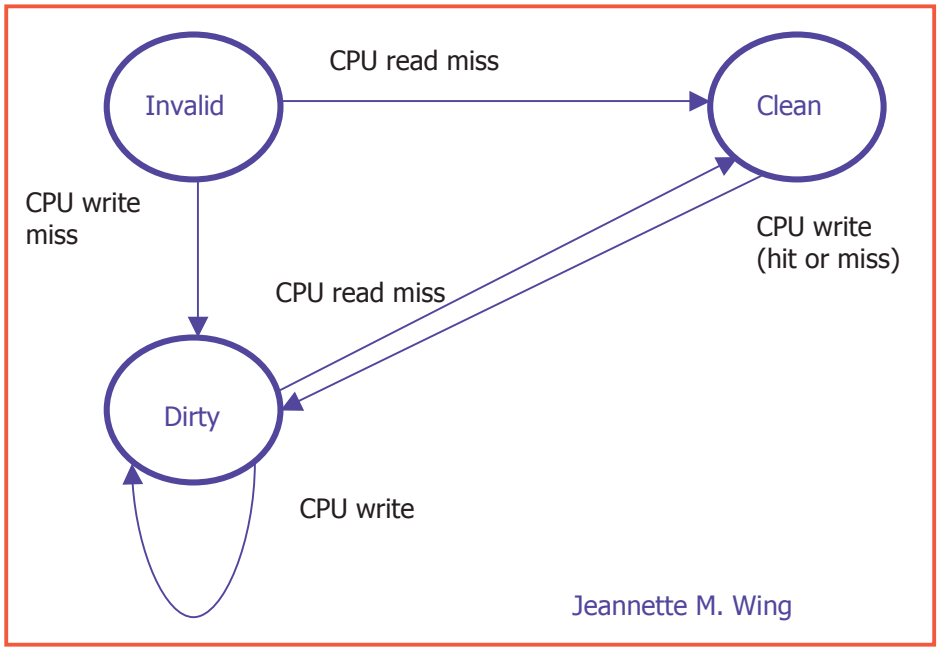
Ph.D. Graduate Seminar: Verification of Concurrent, Reactive and Real-Time Programs

- Modeling concurrent programs with state transition systems
- Temporal logics
- The mu-calculus and fixpoint theory
- The basic model checking algorithm
- Binary decision graphs and symbolic model checking
- Using Omega-automata to specify properties of concurrent systems
- Notions of equivalence for concurrent systems (observational equivalence, etc.)
- Compositional reasoning techniques (e.g., the "assume—guarantee" paradigm)
- Exploiting abstraction and symmetry
- Using induction to reason about systems with many similar processes
- True concurrency and models based on partial orders
- Extending model checking techniques to handle real-time programs
- Model checking techniques for the mu-calculus

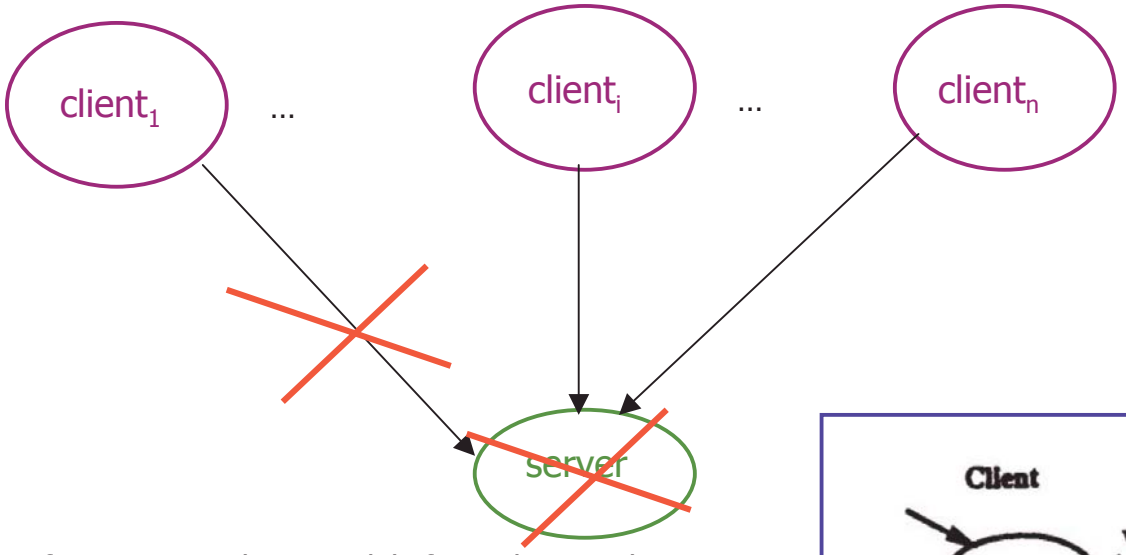
Ph.D. First-Year Course: 15-740 Computer Architecture



1. Trace interesting behaviors (four different scenarios given).
2. Eliminate an atomic operation. Show a safety property is preserved.
3. Eliminate an atomic operation. Show a liveness property is violated.
4. Modify the bus arbitration scheme. Show a fairness property is preserved.
5. Add a prefetch operation. Modify the spec and show the protocol is still correct.

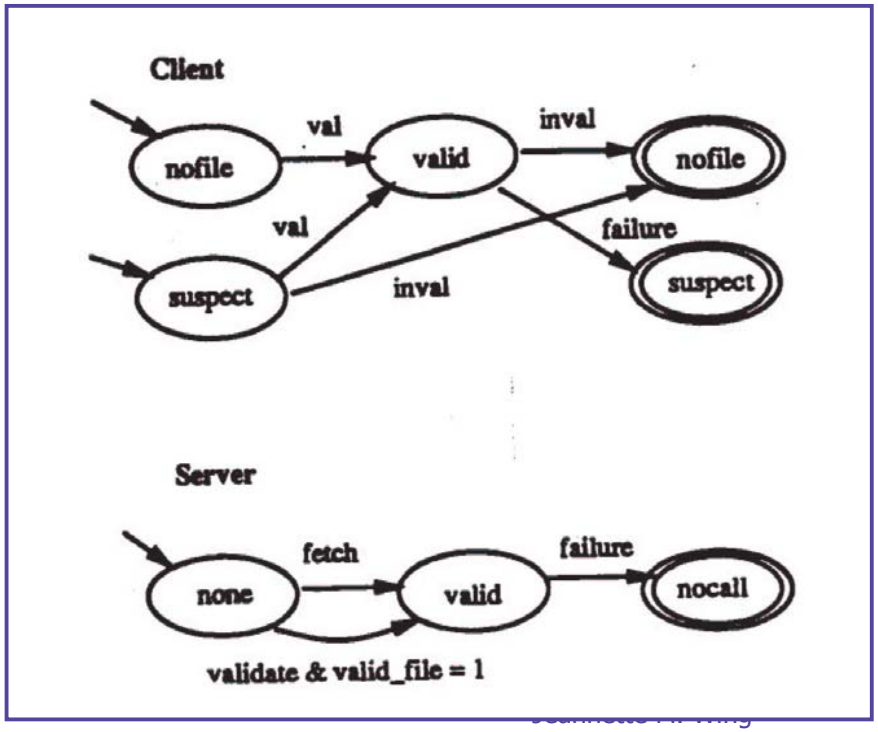


Professional Masters in Software Engineering: 17-654 Analysis of Software Artifacts

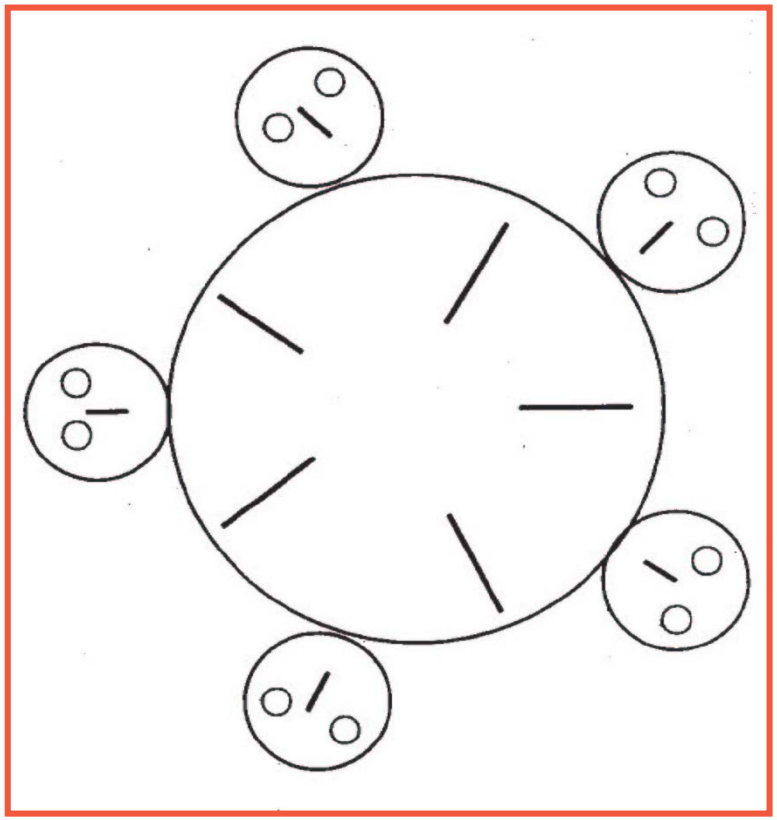


1. Define state machine models for a client and a server suitable for modeling the AFS protocol [...]. You should ignore the timing problem due to failures and transmission delays. However, you should model the possibility of failure. To do this, you may find it useful to introduce and define another state machine.
2. Use SMV notation for describing each state machine you define in (1).
3. Write out a CTL formula that captures the cache coherence correctness condition about belief, i.e.,

   If a client believes its cached file is valid then the server (who is the authority of that file) believes the client's copy is valid.



Undergraduate Elective:
15-398 Bug-Catching: Automated Verification and Testing



Five philosophers…deep thought…chopsticks…A solution to the problem has to guarantee mutually exclusive access to the resources, absence of deadlock, and absence of starvation.

1. Model protocol using SMV.
2. CTL properties for safety, no alternation, mutual exclusion, deadlock freedom, and starvation freedom.
3. Verify properties using SMV.

Observation #3: Towards a More Principled Science

- As subfields of Computer Science mature
  - related course material also matures, to the point of our being able to teach undergraduates more principles, i.e., mathematically-based concepts, underlying our discipline.

Summary

- Linear Algebra and Probability & Statistics are increasingly important to Computer Scientists.
- As Computer Science matures, more mathematics enters CS curricula in different guises.
- As Computer Science matures, more course material covering mathematically-based concepts moves from the graduate to the undergraduate level.

Postlude

Computer Science sprung out of Math and EE departments.

- There is a double irony behind the trend of seeing more math in CS and the trend of seeing more CS in EE as an "application" area.