Objectives

- Introduction to Algorithms, Analysis
- Course summary
- Introduction to CS Research
  - Presenting research
  - Reviewing research papers

What is an Algorithm?

What are our goals when designing algorithms?

- How do we know when we've met our goals?

> Now, everything comes down to expert knowledge of **algorithms** and **data structures**. If you don't speak fluent **O-notation**, you may have trouble getting your next job at the technology companies in the forefront.
> -- Larry Freeman

Course Content: Subject to Change

- Algorithm analysis
  - Formal – proofs; informal
- Advanced data structures, e.g., heaps, graphs
- Greedy Algorithms
- Dynamic Programming
- Divide and Conquer
- Randomization
- Hashing
- Computer science research

Course Notes

- Textbook: *Algorithm Design*
  - Optional: CLRS
- Participation is encouraged
  - Individual, group, class
- Assignments:
  - Proofs (Latex eventually)
  - Programming
  - Research readings

Course Grading

- 40% Individual written and programming homework assignments
- 20% Midterm
- 20% Final
- 9% Reading, analysis, & discussion of Computer Science research papers
- 8% Participation and attendance
- 3% Attending at least two seminars and writing a summary of each

COMPUTER SCIENCE RESEARCH

Computer Science Research

- Discussion:
  - What are fields of computer science?
  - What research problems do they answer in those fields?

Examples of CS Research

Field	Example Research
Operating Systems	How to best manage resources—allocating to jobs
Compilers	Data flow, control flow → optimizations
Software Engineering	Program comprehension; Automated techniques to help find bugs
AI	New "learning" techniques; Applying AI techniques to problems in various fields
Networks	Better communication protocols
Distributed systems	Managing data across multiple sites; managing multiple computers across multiple sites

Computer Science Research

- Generally: attempts to create something "new"
- Sometimes enabled by new technology
  - E.g., Internet, Multi-core processors, mobile devices
- Often: new algorithms
  - Ideally, faster, less memory, more efficient, or some other benefit/metric (e.g., security, false positives/ false negatives, accurate/representative)
- New representations
  - Allow for easier computation/understanding
- Exploring for understanding
  - Program comprehension, software characteristics, education

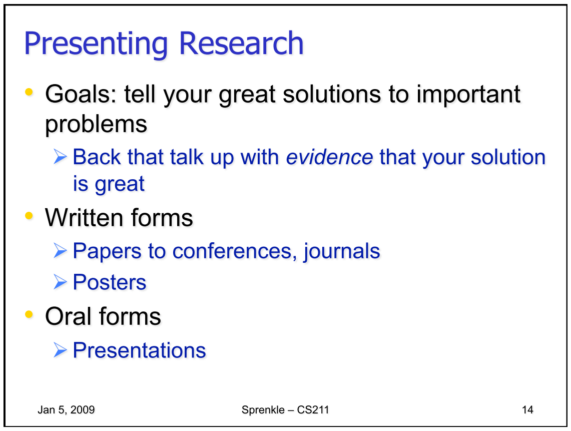
Our Task: Select a New Faculty Member

- What we're looking for:
  - Students will like candidate as professor, work with on research
  - Candidate will attract more CS students
- Process
  - Candidate visits for 2 days; meets with all CS faculty, perhaps related outside faculty members, Dean of the College, and STUDENTS; presents **research** in a talk
  - Make an offer to a candidate after all candidates visit
  - Candidate has up to 2 weeks to accept/decline offer

Your Participation

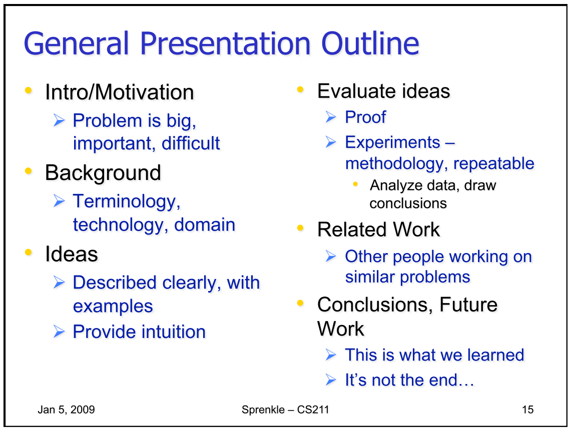
- Attend at least one candidate's talk
  - Beyond Monday's talk
- Attend at least one student meeting with candidate
- If scheduling conflicts, please contact me immediately
- Read faculty candidates' papers
  - Post reviews on Sakai, in forums
  - Read and fill out review form for first one for Friday

PRESENTING RESEARCH IN COMPUTER SCIENCE



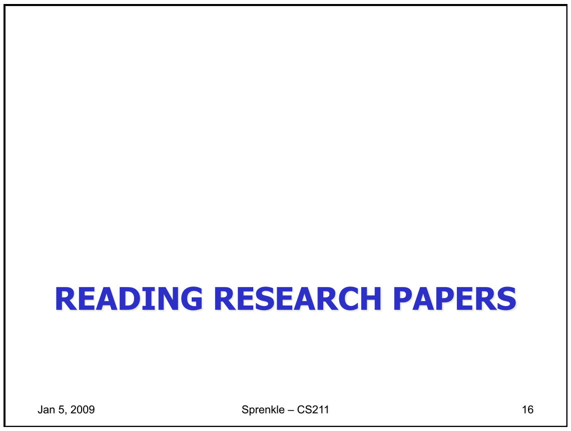
Presenting Research

- Goals: tell your great solutions to important problems
  - Back that talk up with *evidence* that your solution is great
- Written forms
  - Papers to conferences, journals
  - Posters
- Oral forms
  - Presentations

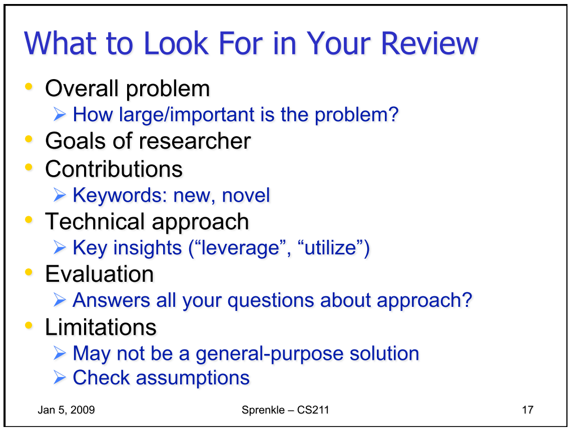


General Presentation Outline

- Intro/Motivation
  - Problem is big, important, difficult
- Background
  - Terminology, technology, domain
- Ideas
  - Described clearly, with examples
  - Provide intuition
- Evaluate ideas
  - Proof
  - Experiments – methodology, repeatable
    - Analyze data, draw conclusions
- Related Work
  - Other people working on similar problems
- Conclusions, Future Work
  - This is what we learned
  - It's not the end…

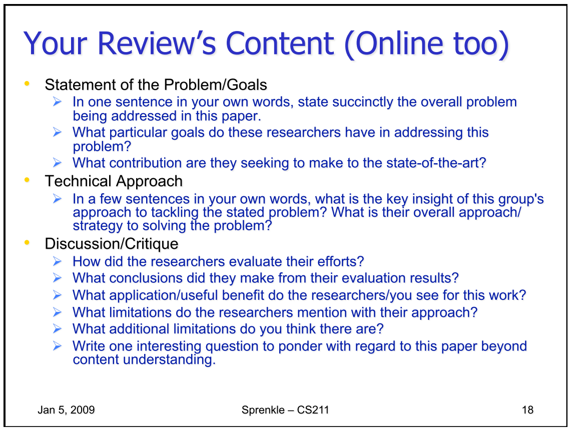


READING RESEARCH PAPERS



What to Look For in Your Review

- Overall problem
  - How large/important is the problem?
- Goals of researcher
- Contributions
  - Keywords: new, novel
- Technical approach
  - Key insights ("leverage", "utilize")
- Evaluation
  - Answers all your questions about approach?
- Limitations
  - May not be a general-purpose solution
  - Check assumptions



Your Review's Content (Online too)

- Statement of the Problem/Goals
  - In one sentence in your own words, state succinctly the overall problem being addressed in this paper.
  - What particular goals do these researchers have in addressing this problem?
  - What contribution are they seeking to make to the state-of-the-art?
- Technical Approach
  - In a few sentences in your own words, what is the key insight of this group's approach to tackling the stated problem? What is their overall approach/ strategy to solving the problem?
- Discussion/Critique
  - How did the researchers evaluate their efforts?
  - What conclusions did they make from their evaluation results?
  - What application/useful benefit do the researchers/you see for this work?
  - What limitations do the researchers mention with their approach?
  - What additional limitations do you think there are?
  - Write one interesting question to ponder with regard to this paper beyond content understanding.

PRACTICE

Abstract 1: TerraStream

We consider the problem of extracting a river network and a watershed hierarchy from a terrain given as a set of irregularly spaced points. We describe TerraStream, a pipelined solution that consists of four main stages: construction of a digital elevation model (DEM), hydrological conditioning, extraction of river networks, and construction of a watershed hierarchy. Our approach has several advantages over existing methods.

First, we design and implement the pipeline so that each stage is scalable to massive data sets; a single non-scalable stage would create a bottleneck and limit overall scalability. Second, we develop the algorithms in a general framework so that they work for both TIN and grid DEMs. Furthermore, TerraStream is flexible and allows users to choose from various models and parameters, yet our pipeline is designed to reduce (or eliminate) the need for manual intervention between stages.

We have implemented TerraStream and we present experimental results on real elevation point sets, which show that our approach handles massive multi-gigabyte terrain data sets. For example, we can process a data set containing over 300 million points—over 20GB of raw data—in under 26 hours, where most of the time (76%) is spent in the initial CPU-intensive DEM construction stage.

Abstract 1: TerraStream – Theory, GIS

- Problem/Goals
  - Extracting river network, watershed hierarchy, given irregularly spaced points
  - Handle large data sets
- Technical Approach
  - Scalable, pipelined process
- Discussion
  - Evaluation: execution time on large (e.g., 20 GB), real elevation point sets

Abstract 2: Proverb

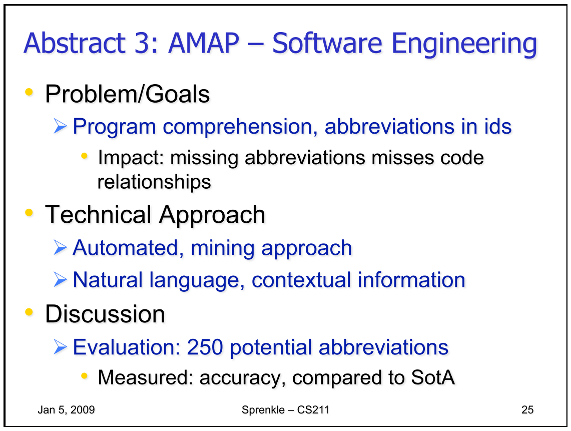
We attacked the problem of solving crossword puzzles by computer: given a set of clues and a crossword grid, try to maximize the number of words correctly filled in. In our system, "expert modules" specialize in solving specific types of clues, drawing on ideas from information retrieval, database search, and machine learning. Each expert module generates a (possibly empty) candidate list for each clue, and the lists are merged together and placed into the grid by a centralized solver. We used a probabilistic representation throughout the system as a common interchange language between subsystems and to drive the search for an optimal solution. Proverb, the complete system, averages 95.3% words correct and 98.1% letters correct in under 15 minutes per puzzle on a sample lf 370 puzzles taken from the New York Times and several other puzzle sources. This corresponds to missing roughly 3 words or 4 letters on a daily 15x15 puzzle, making Proverb a better-than-average cruciverabalist (cross-word solver).

Abstract 2: Proverb - AI

- Problem/Goals
  - Solving crossword puzzles
  - Maximize # of words correctly filled in
- Technical Approach
  - Expert modules, lists of candidates
  - IR, DB Search, ML
  - Probabilistic representation
- Discussion
  - Evaluation: 370 *NY Times* puzzles
    - Measured: time to execute; words, letters correct
- Aside: oneacross.com

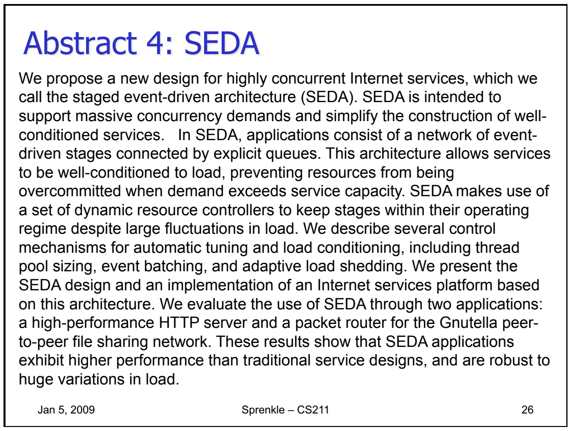
Abstract 3: AMAP

When writing software, developers often employ abbreviations in identifier names. In fact, some abbreviations may never occur with the expanded word, or occur more often in the code. However, most existing program comprehension and search tools do little to address the problem of abbreviations, and therefore may miss meaningful pieces of code or relationships between software artifacts. In this paper, we present an automated approach to mining abbreviation expansions from source code to enhance software maintenance tools that utilize natural language information. Our scoped approach uses contextual information at the method, program, and general software level to automatically select the most appropriate expansion for a given abbreviation. We evaluated our approach on a set of 250 potential abbreviations and found that our scoped approach provides a 57% improvement in accuracy over the current state of the art.



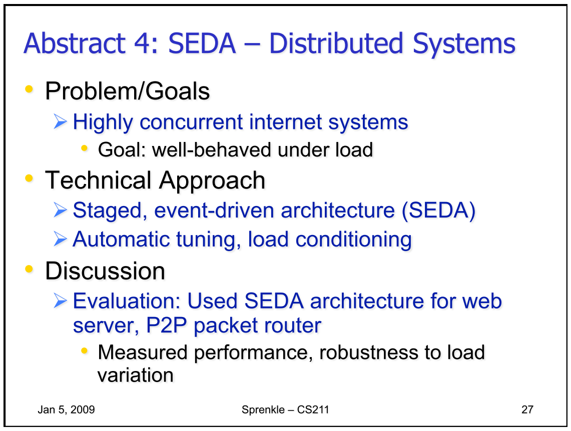
Abstract 3: AMAP – Software Engineering

- Problem/Goals
  - Program comprehension, abbreviations in ids
    - Impact: missing abbreviations misses code relationships
- Technical Approach
  - Automated, mining approach
  - Natural language, contextual information
- Discussion
  - Evaluation: 250 potential abbreviations
    - Measured: accuracy, compared to SotA



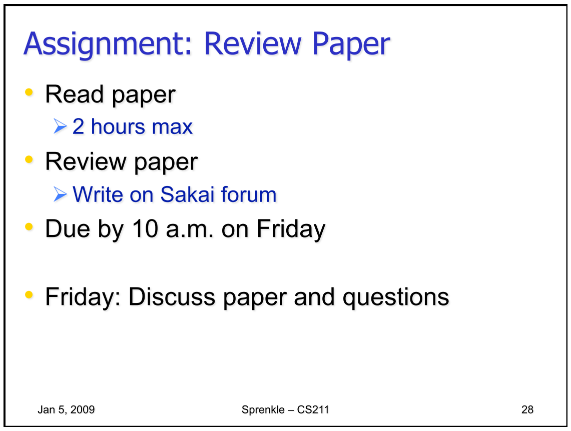
Abstract 4: SEDA

We propose a new design for highly concurrent Internet services, which we call the staged event-driven architecture (SEDA). SEDA is intended to support massive concurrency demands and simplify the construction of well-conditioned services. In SEDA, applications consist of a network of event-driven stages connected by explicit queues. This architecture allows services to be well-conditioned to load, preventing resources from being overcommitted when demand exceeds service capacity. SEDA makes use of a set of dynamic resource controllers to keep stages within their operating regime despite large fluctuations in load. We describe several control mechanisms for automatic tuning and load conditioning, including thread pool sizing, event batching, and adaptive load shedding. We present the SEDA design and an implementation of an Internet services platform based on this architecture. We evaluate the use of SEDA through two applications: a high-performance HTTP server and a packet router for the Gnutella peer-to-peer file sharing network. These results show that SEDA applications exhibit higher performance than traditional service designs, and are robust to huge variations in load.



Abstract 4: SEDA – Distributed Systems

- Problem/Goals
  - Highly concurrent internet systems
    - Goal: well-behaved under load
- Technical Approach
  - Staged, event-driven architecture (SEDA)
  - Automatic tuning, load conditioning
- Discussion
  - Evaluation: Used SEDA architecture for web server, P2P packet router
    - Measured performance, robustness to load variation



Assignment: Review Paper

- Read paper
  - 2 hours max
- Review paper
  - Write on Sakai forum
- Due by 10 a.m. on Friday

- Friday: Discuss paper and questions