CS 8803-MDS

Human-in-the-loop Data Analytics

Lecture 2

08/23/23

Logistics

Office hour:

- Kexin: Thursday 4PM-5PM, Klaus 3322
- Ashmita/Saumia: Friday TBD

Today’s class

What is research?

How to develop a new idea?

How to do literature review?

How to read a paper?

Help us learn your names!

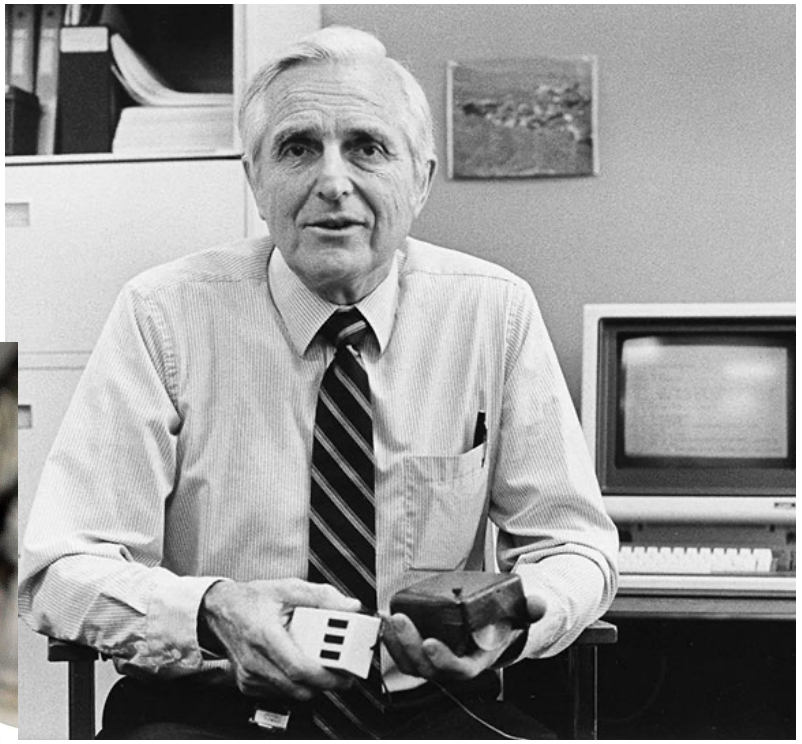
Computer Science Research

What is the goal of research?

Why has it driven major innovations in computing?

What separates research from advanced development?

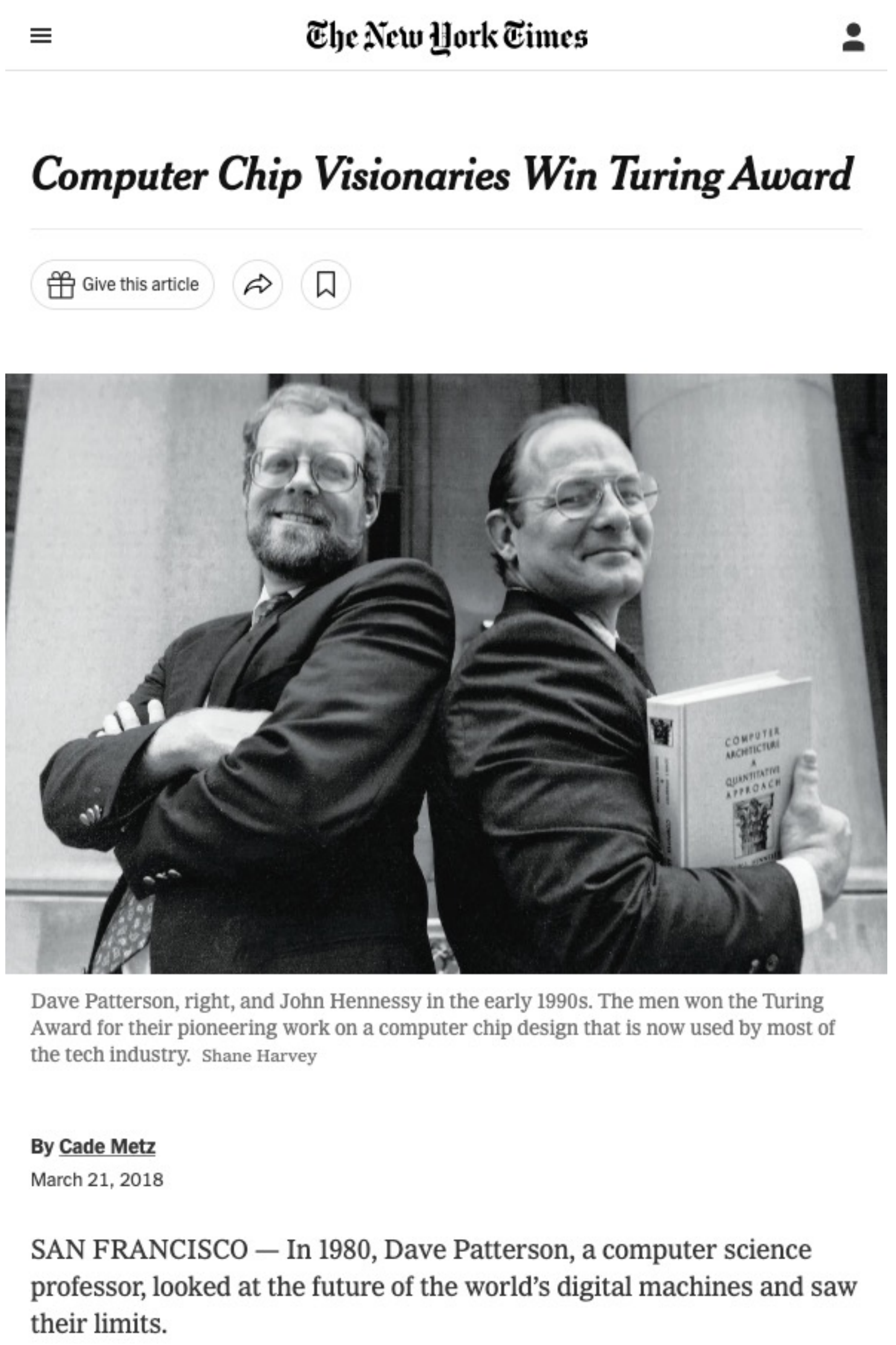
A Tale of Three Turing Awards



Hennessy and Patterson: RISC

Computer architecture was increasing in complexity, in order to enable more and more advanced computation.

Everyone thought that *increasingly powerful processors needed increasingly complicated instruction* sets to take advantage of them.



The New York Times

Computer Chip Visionaries Win Turing Award

Give this article

Dave Patterson, right, and John Hennessy in the early 1990s. The men won the Turing Award for their pioneering work on a computer chip design that is now used by most of the tech industry. Shane Harvey

By Cade Metz

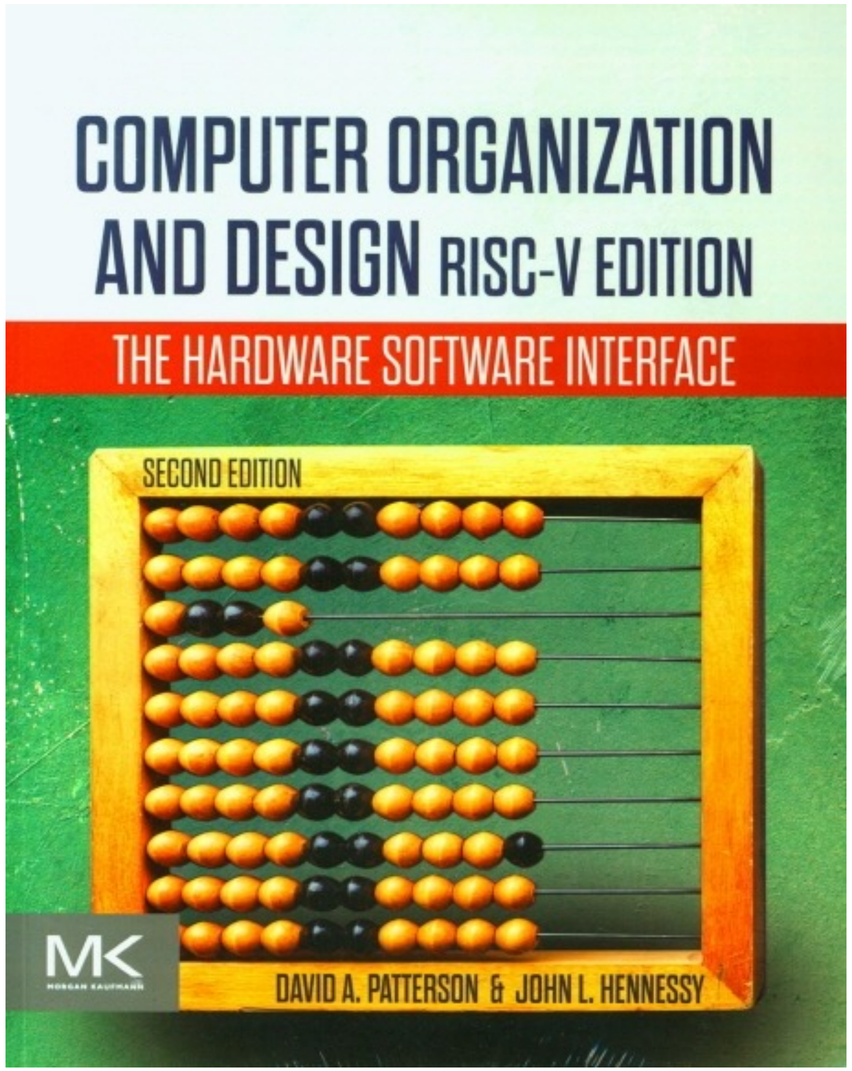
March 21, 2018

SAN FRANCISCO — In 1980, Dave Patterson, a computer science professor, looked at the future of the world's digital machines and saw their limits.

Hennessy and Patterson: RISC

"No, let's do it this way instead:" have a very simple instruction set. That way you can compare performance, optimize, and prevent errors.

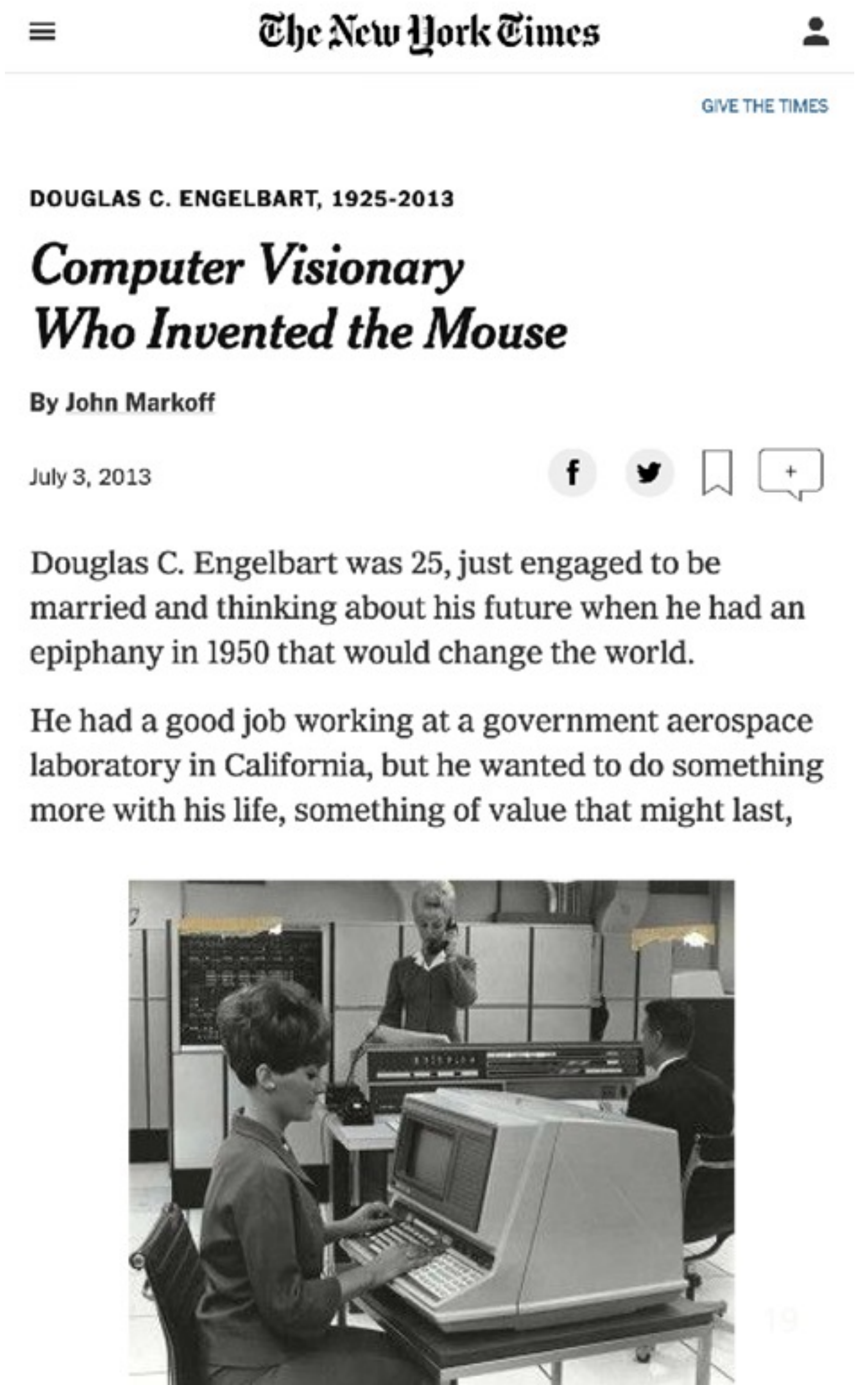
This became known as Reduced Instruction Set Computer (RISC). Today, more than 99 percent of all new chips use the RISC architecture they developed.


COMPUTER ORGANIZATION AND DESIGN RISC-V EDITION
THE HARDWARE SOFTWARE INTERFACE
SECOND EDITION
MK
DAVID A. PATTERSON & JOHN L. HENNESSY

Engelbart: interactive computing

When computers originated, they were used for, well, computing: calculating mathematical functions.

This meant that computers were seen as most appropriate for slow, batch interaction, shared by entire teams.


The New York Times

GIVE THE TIMES

DOUGLAS C. ENGELBART, 1925-2013

Computer Visionary Who Invented the Mouse

By John Markoff

July 3, 2013

Douglas C. Engelbart was 25, just engaged to be married and thinking about his future when he had an epiphany in 1950 that would change the world.

He had a good job working at a government aerospace laboratory in California, but he wanted to do something more with his life, something of value that might last,

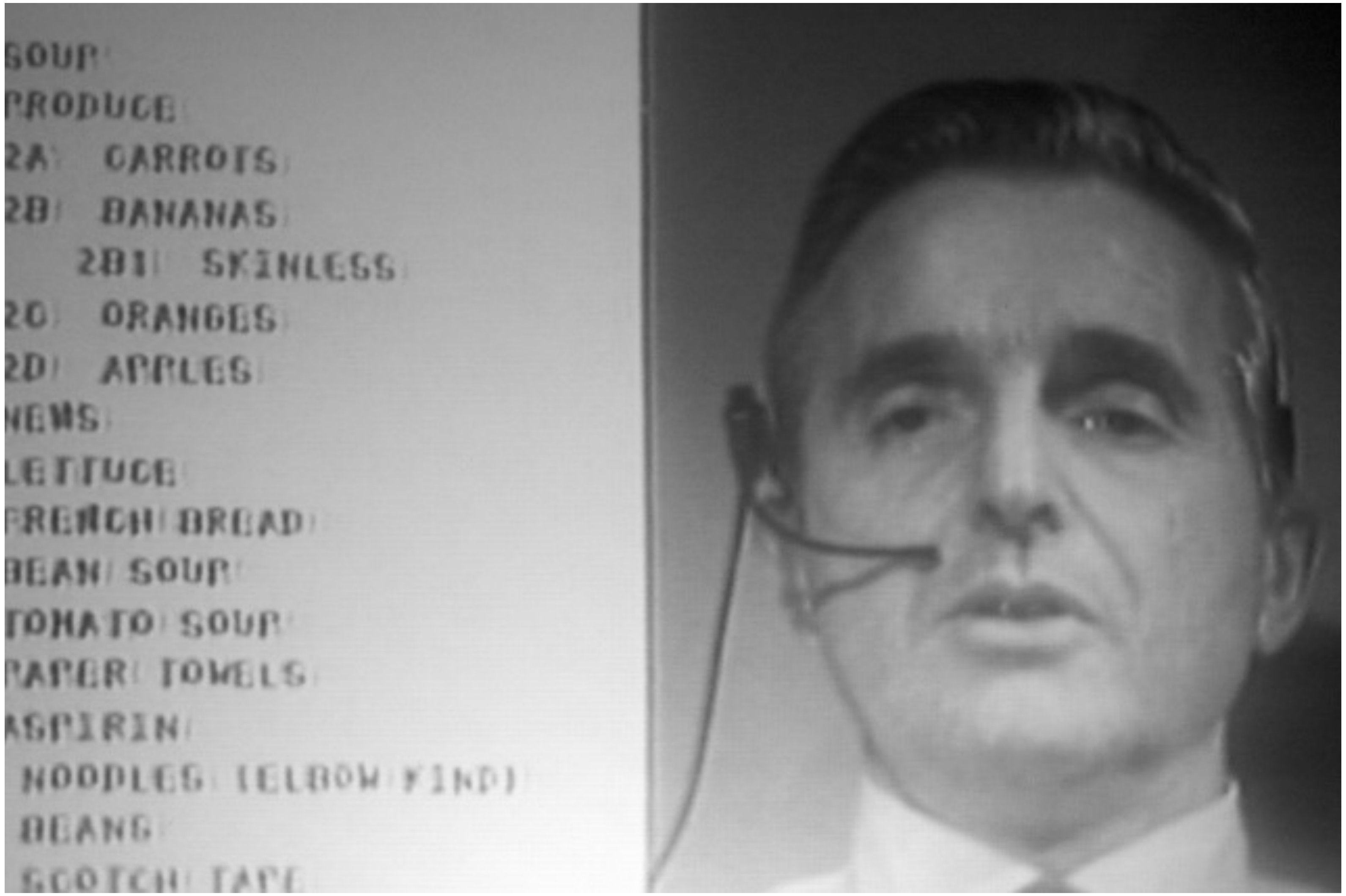
Engelbart: interactive computing

“No, let’s do it this way instead:” computing should be used as a tool for thought. We must move from batch-style computing to interactive computing.

His result was the “Mother of All Demos”: mouse, hypertext, bitmapped screens, collaborative software, and more.

This led to Xerox Star. Steve Jobs saw it, was wow’ed, and infused the ideas into the Mac.

Engelbart: interactive computing



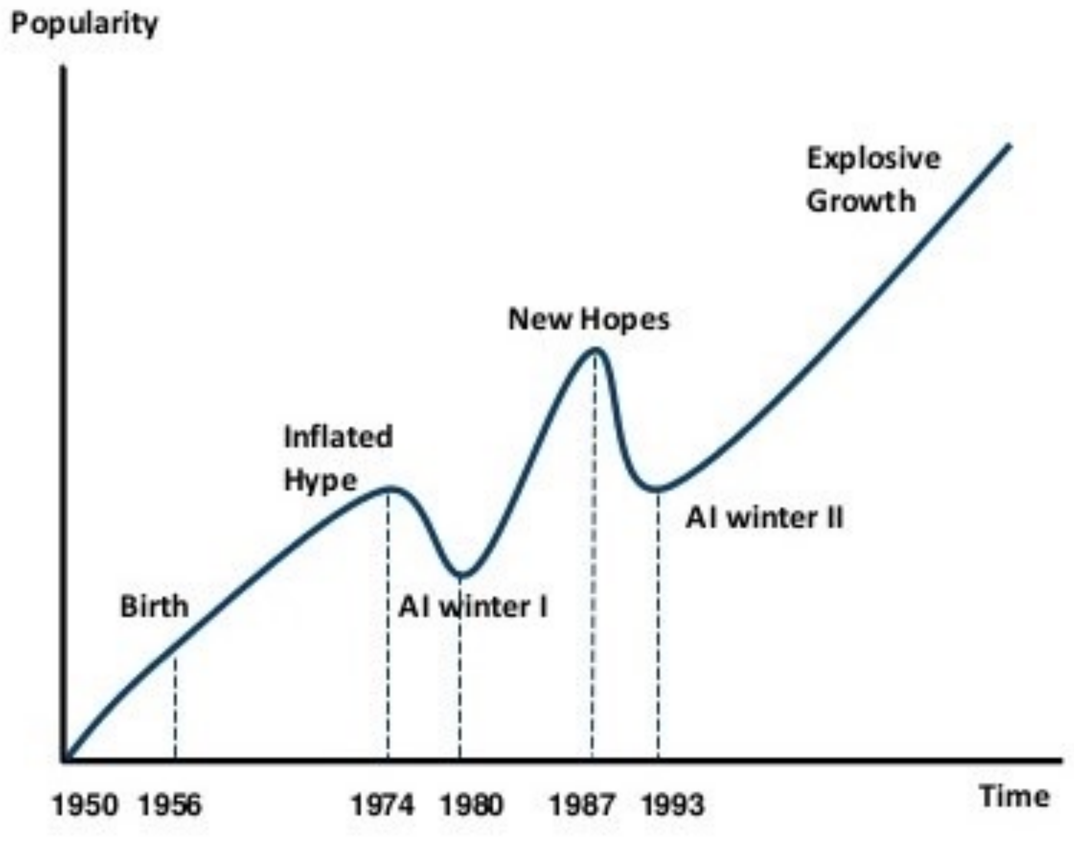
```
SOUP
PRODUCE
2A  CARROTS
2B  BANANAS
    2B1  SKINLESS
2C  ORANGES
2D  APPLES
NEWS
LETTUCE
FRENCH BREAD
BEAN SOUP
TOMATO SOUP
PAPER TOWELS
ASPIRIN
 NOODLES (ELBOW KIND)
 BEANS
 SCOTCH TAPE
```

https://youtu.be/JQ8ZiT1sn88

LeCun, Hinton, Bengio: deep learning

The idea of neural networks had been around for fifty years, but unsuccessful. Major AI figures had trashed it, even proving that early versions had very limited expressiveness.

Instead, machine learning was based on other models, for example the support vector machine and graphical models. Neural networks did not perform well.

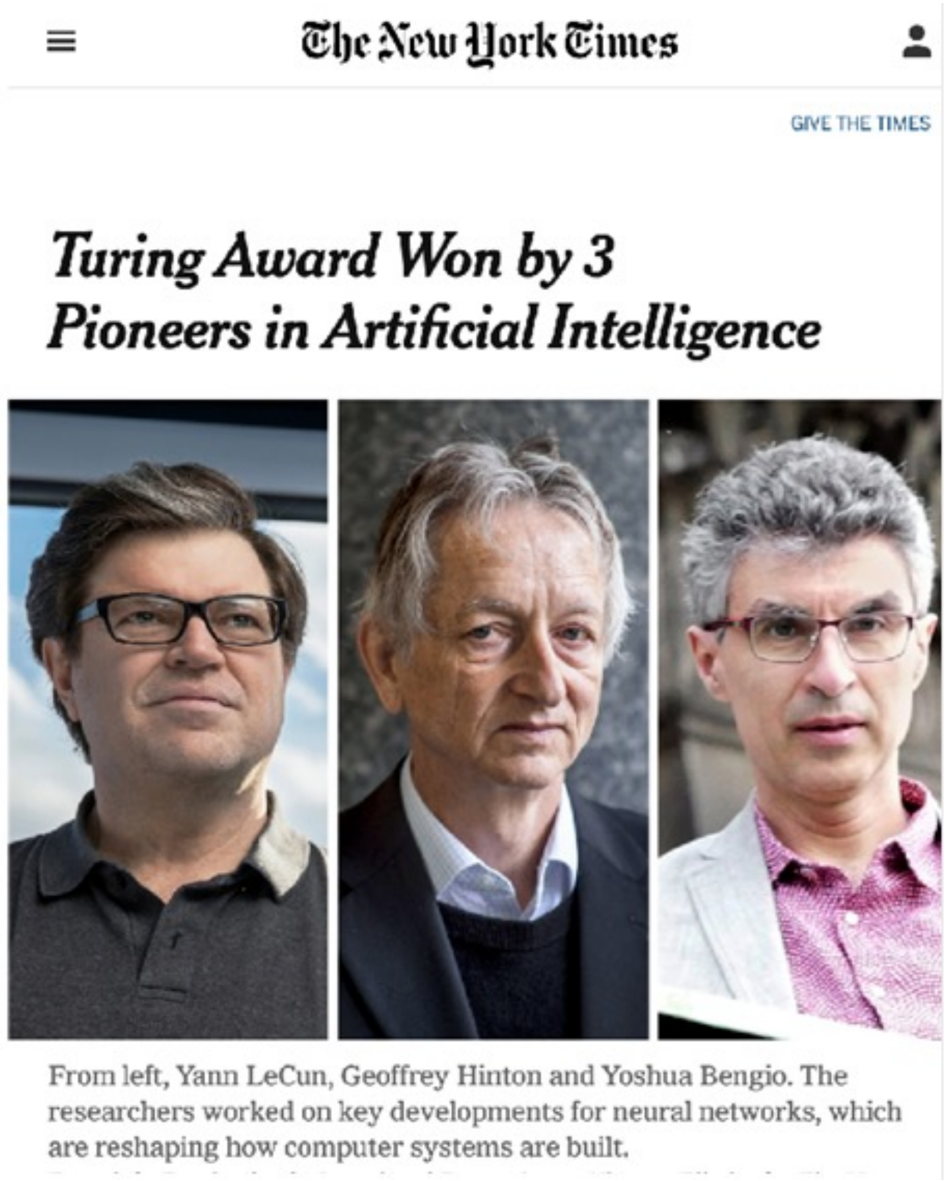


Adapted from Stanford CS197

LeCun, Hinton, Bengio: deep learning

“No, let’s do it this way instead:” these networks learn extremely complex functions, so they need much more data than existing machine learning approaches, GPUs to train, and algorithms to enable them to learn more effectively.

Around 2010, these models began smashing records in speech and image recognition. They are now foundational to ML.



The New York Times

GIVE THE TIMES

Turing Award Won by 3 Pioneers in Artificial Intelligence

From left, Yann LeCun, Geoffrey Hinton and Yoshua Bengio. The researchers worked on key developments for neural networks, which are reshaping how computer systems are built.

Not all research wins Turing Awards. But…

It all follows the same formula:

An implicit assumption: Industry and other researchers all thought one way about a problem

“No, let’s do it this way instead:” The researcher offered a new perspective that nobody had ever considered or made feasible before. They proved out their idea as the better approach.

What is research?

Research introduces a fundamental new idea into the world.

These ideas did not exist in any mature or well-articulated way before their creators developed them.

If the idea is already in the world, for example published by someone else, it is not considered novel, and thus not research.

How to develop a novel idea?

Novel ideas rarely come out of a vacuum

They're much more often pivoted off of today's work:

- A realization that an idea has been applied in domains like X and needs to be rethought in domains like ~X
- A recognition that others have tried this technique in users of context A, or data of up to size N, but ~A or >>N breaks the technique.
- Some constraint that exists but shouldn't, or visa versa

The "bit flip" method: invert an assumption

bit flip: an inversion of an assumption that the world has about how the world is supposed to work.

Recipe for a bit flip:

1) Define the bit: articulate an assumption, often left implicit in prior work
2) Introduce the flip: argue for an alternative to that assumption / "No, let's do it this way instead"

Bit	Flip	Project
We need complicated instruction sets to accommodate powerful computer processors.	Simple instruction sets are better since they let you compare performance, optimize, and prevent errors.	RISC architecture
Computing was just for numerical calculations: slow, done in batches, and for teams.	Computing should be interactive, individual, and support thought.	Mother of all demos
Neural networks exist, but don’t perform very well and aren’t accurate.	We need more data and different algorithms for this to work.	Deep learning

Adapted from Stanford CS197

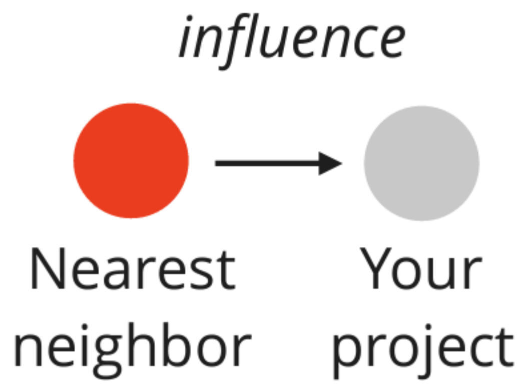
Bit	Flip	Project
A minimum graph cut algorithms should always return correct answers.	A randomized, probabilistic algorithm will be much faster, and we can still prove a limited probability of an error.	Karger’s algorithm
Activity tracking requires custom hardware.	Activity tracking requires just a standard cell phone.	
NLP machine learning models should read sentences word by word	Models should consume the entire sentence at once to make long-range dependency learning easier	Transformer Networks

Adapted from Stanford CS197

Single paper bit slip

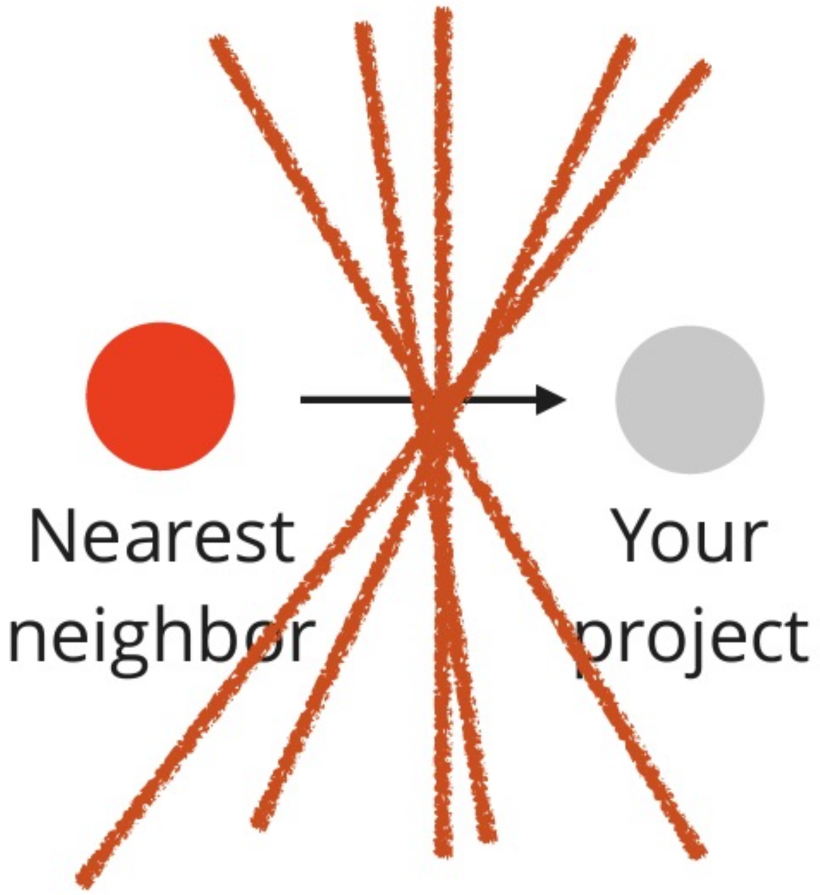
Find a paper that is adjacent to your idea. Think of this as your nearest neighbor paper.

Your project will be some sort of delta off of that paper. What assumption or limitation did it have, that you're erasing?



Adapted from Stanford CS197

Single paper bit slip

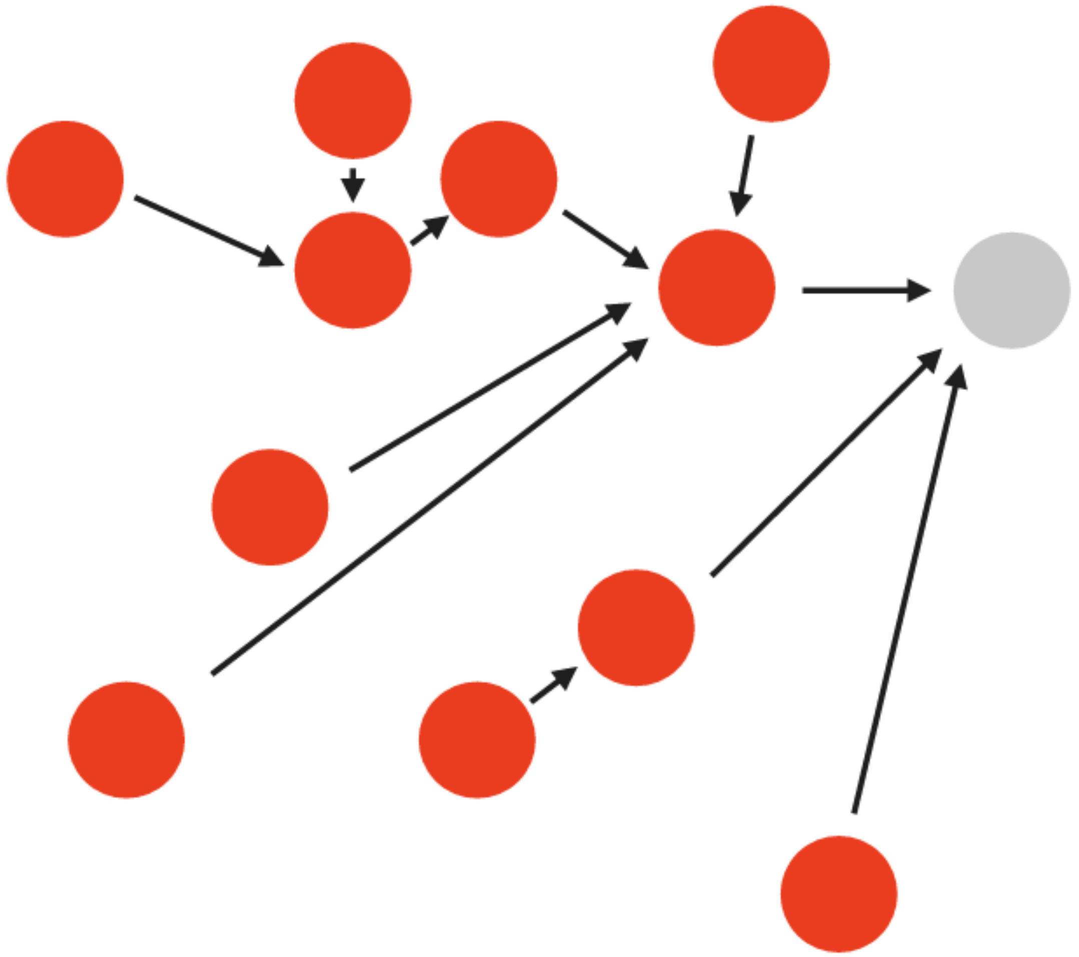


Each separating line is a possible bit flip.

Which one should you go for?

Adapted from Stanford CS197

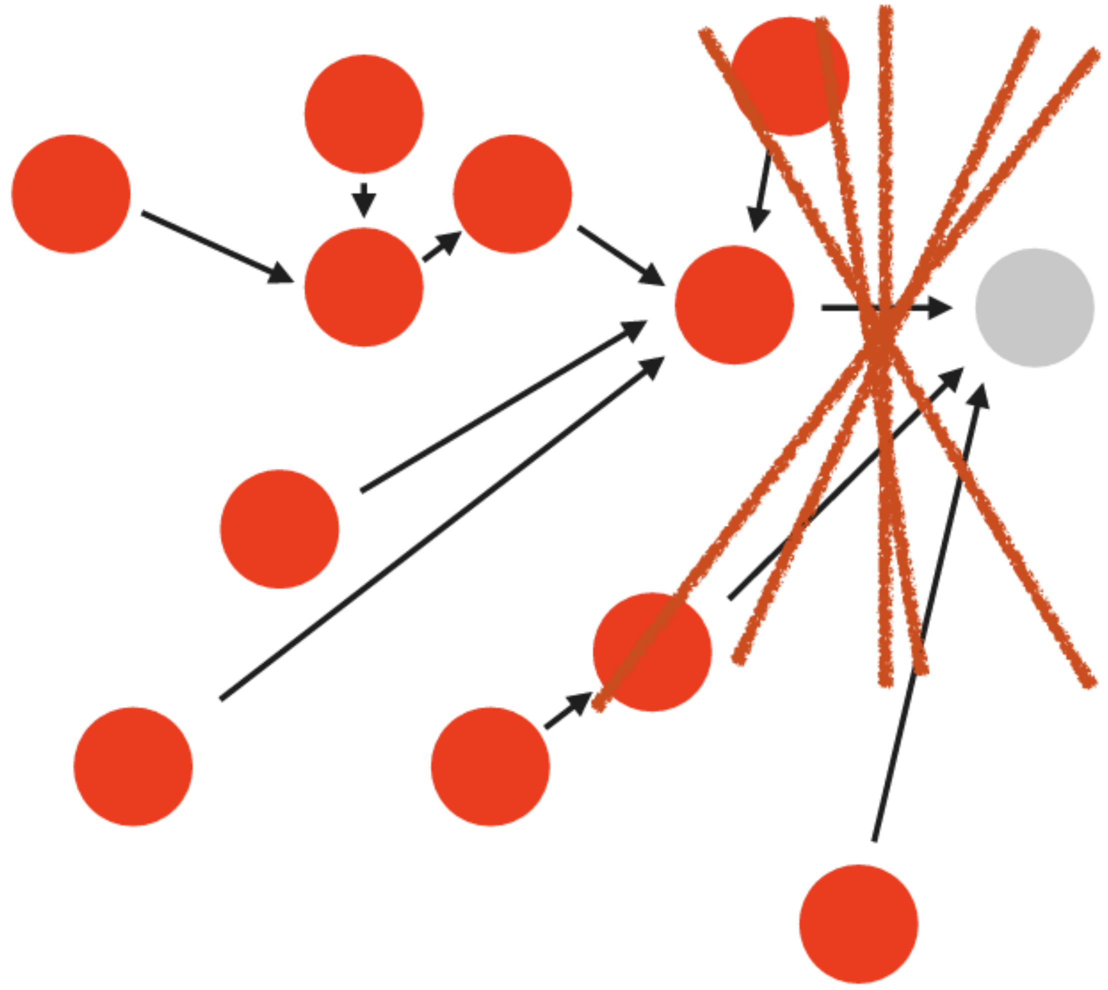
Literature bit flip



The broader an understanding you have of the literature and the design axes underneath it, the more effectively you can pick the right bit flip.

Adapted from Stanford CS197

Literature bit flip



The broader an understanding you have of the literature and the design axes underneath it, the more effectively you can pick the right bit flip.

Adapted from Stanford CS197

How to do a literature review

1. Pick your favorite academic search engine (e.g., Google scholar) and start with keywords
2. Find 3-5 recent and highly cited papers
    - From reputable venues and by reputable institution/author
    - If you find a survey paper, start from the survey paper
3. Do the first pass to identify key papers and researchers that these works cite
4. Track down these papers/researchers
5. Iterate as needed

Tools

Backward influence: influential citations in the papers that you've read

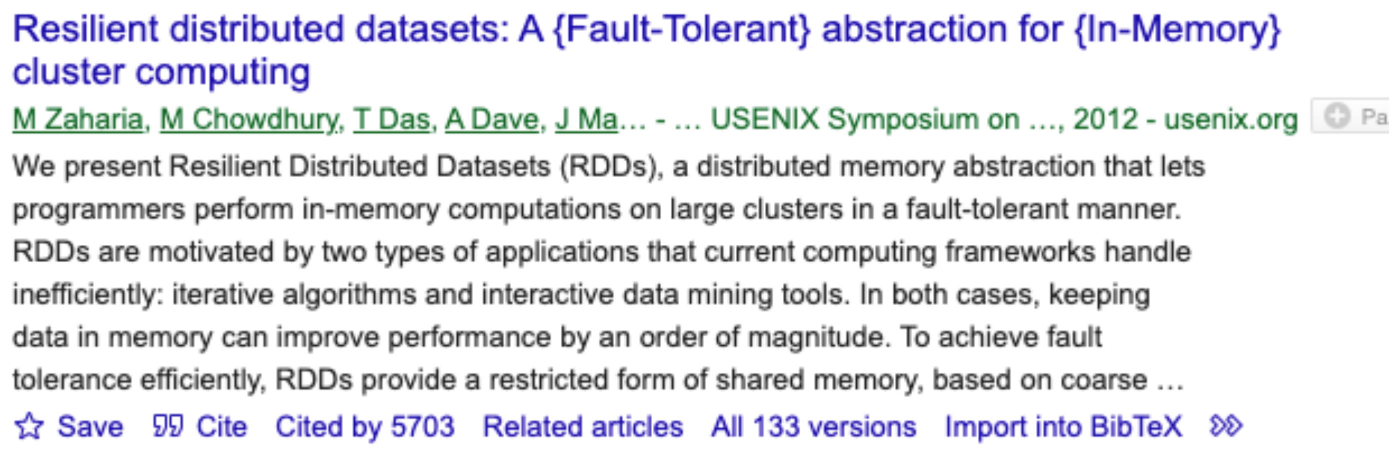
How: reading

Forward influence: papers citing the ones that you've read

How : Google Scholar's "Cited By"

Relatedness: contemporaneous but not citing

How: Google Scholar's "Related articles"



Resilient distributed datasets: A {Fault-Tolerant} abstraction for {In-Memory} cluster computing

M Zaharia, M Chowdhury, T Das, A Dave, J Ma… - … USENIX Symposium on …, 2012 - usenix.org

We present Resilient Distributed Datasets (RDDs), a distributed memory abstraction that lets programmers perform in-memory computations on large clusters in a fault-tolerant manner. RDDs are motivated by two types of applications that current computing frameworks handle inefficiently: iterative algorithms and interactive data mining tools. In both cases, keeping data in memory can improve performance by an order of magnitude. To achieve fault tolerance efficiently, RDDs provide a restricted form of shared memory, based on coarse …

Filtering your horizon

Not all papers achieve the same level of quality. Especially on white paper archives such as arXiv.org, quality can be variable.

How do I know what to read and what to ignore?

- If the paper is from a reputable venue
- If the paper has been cited frequently (a vote of confidence from the community)
- Some filter by institution or lab, but beware of propagating institutional bias. Great research can and does happen at most universities!

Reputable venues

Databases and Data Management:

- VLDB, SIGMOD, ICDE, CIDR

Data Mining:

- KDD, WWW

HCI:

- CHI, UIST, InfoVis

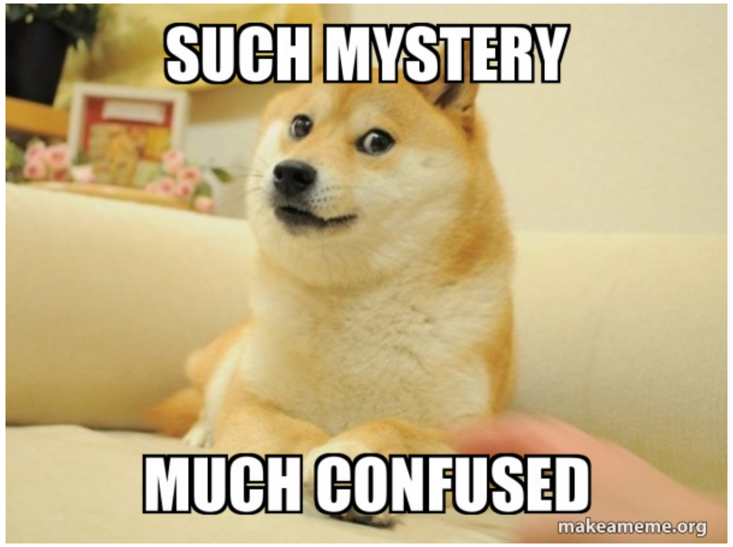
Machine Learning:

- NeurIPS (formerly NIPS), ICML, ICLR, AAAI
- CVPR, ECCV, ICCV (vision), ACL, EMNLP (NLP)

The halting problem

How do you know when to stop reading? When do you have enough confidence that your idea is the right contribution to pursue?

What if you've missed something?

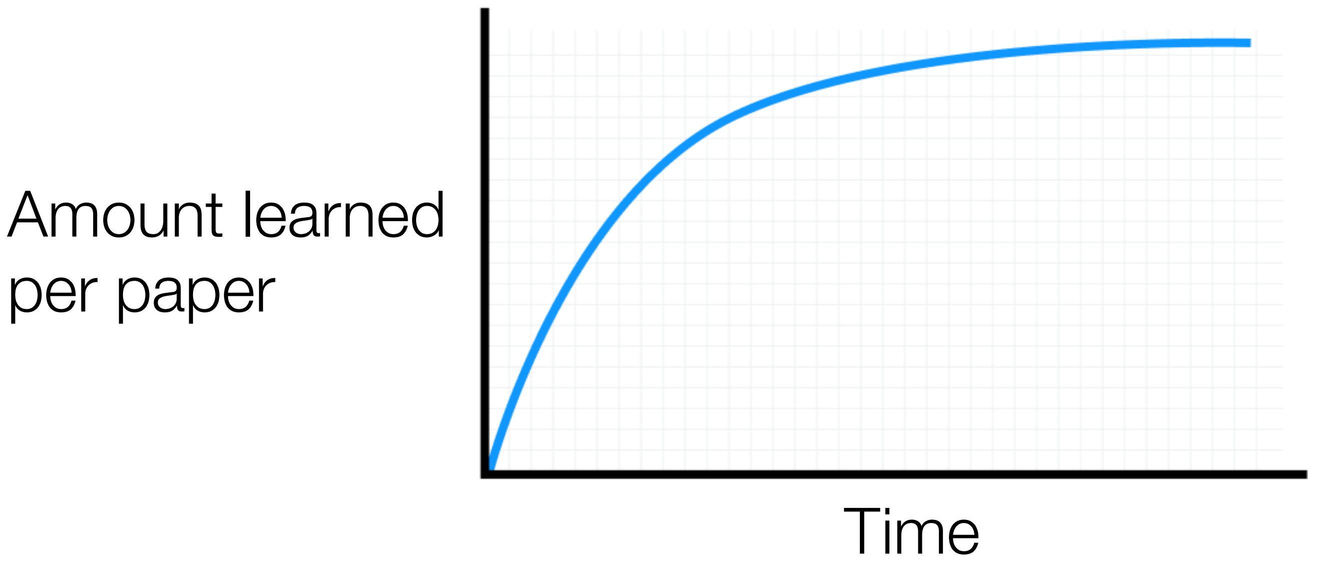


Asymptoting

Keep track of how much you're learning about the bit flip axes as you consume the additional papers. Typically, you are learning the most at the very beginning, and the amount per paper starts going down after five papers or so.

A PhD student often asymptotes after 25–35 papers.

For this class, we'll go with ~10.



Reading a paper for a literature search

Temptation: understand everything.

Typically, when we come to a paper, we want to understand everything about it. We stop and reread any point we don’t get.

This can take an hour or two per paper for a new researcher.

This strategy can be useful at the beginning, but it is actively harmful in constructing a related work section.

Understand the main point

Instead, articulate to yourself: what is the main point (the bit flip!) that this paper is making?

Then, focus your reading and effort most closely on the parts of the paper that are supporting or evaluating that flip.

It's OK not to understand every sentence in the paper.

Your goal isn't to understand the paper — it's to understand the literature, what works and what doesn't (and why!), and the additional bits that are available to flip.

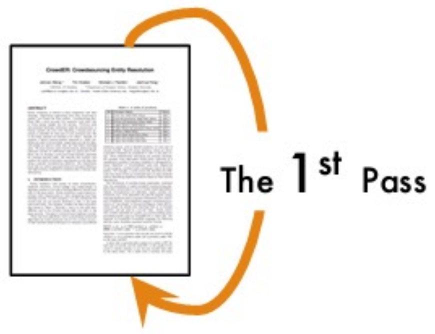
How to read a paper in depth

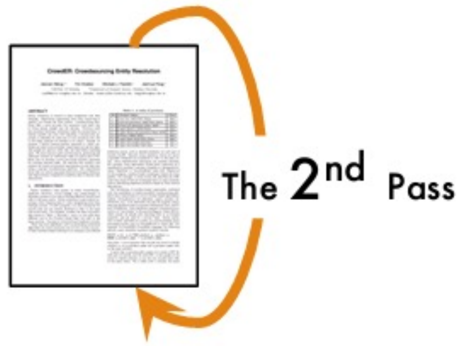
The "three-pass" approach [1]

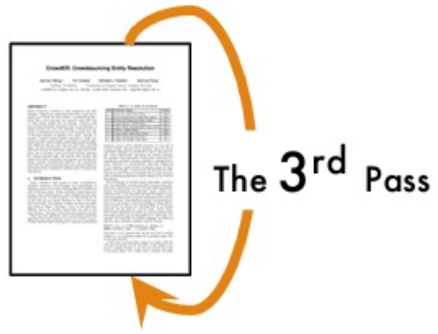
first pass: a quick scan

second pass: with greater care, but ignore the details

third pass: re-implementing the paper







[1] S. Keshav. How to read a paper? http://blizzard.cs.uwaterloo.ca/keshav/home/Papers/data/07/paper-reading.pdf

The first pass: a quick scan

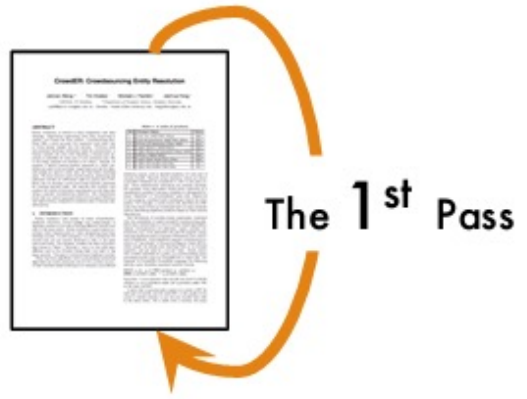
Goal: get bird's-eye view of the paper (5~10 min)

What to read:

- Title, abstract, introduction and conclusion
- Section and sub-section headings
- Main figures
- Scan of bibliography

You should be able to answer:

- What type of paper is this?
- What are the main contributions?



The second pass: grasp the content

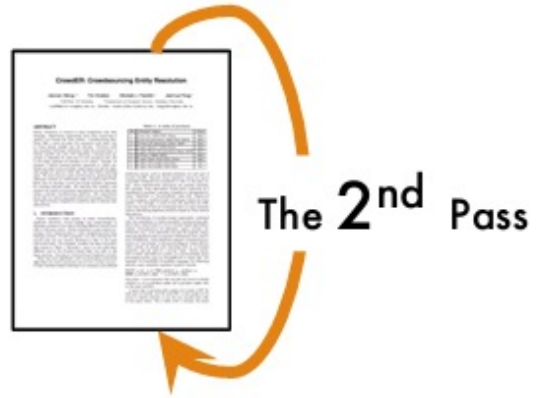
Goal: get a good understanding of the ”meat” of the paper

How to read:

- Look carefully at figures, diagrams and examples
- Take notes of questions, unread references etc.
- Ignore proofs, appendix, extensions etc.

You should be able to:

- Summarize main thrusts of the paper, with supporting evidence, to someone else



The third pass: all about the details

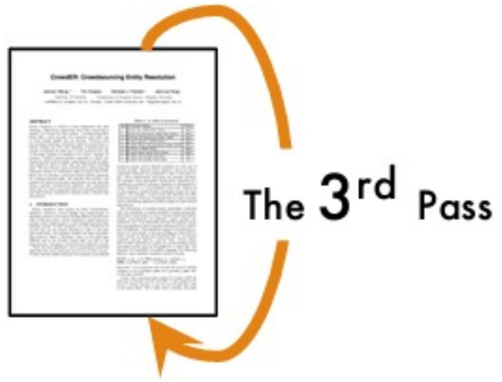
Goal: think about what you would have done if you were to re-implement such an idea

How to read:

- Challenge every assumption
- Compare your version with the actual paper
    - Often leads to questions like: why not do it this way?

You should be able to:

- Identify hidden assumptions/potential design flaws
- Get ideas for future work



Let’s try the first pass!

1. **Category**: What type of paper is this? A measurement paper? An analysis of an existing system? A description of a research prototype?
2. **Context**: Which other papers is it related to?
3. **Correctness**: Do the assumptions appear to be valid?
4. **Contributions**: What are the paper’s main contributions?
5. **Clarity**: Is the paper well written?

How to write an introduction

After reading an introduction, the readers should have an idea of 1) what it the problem and 2) how you are solving it.

Outline Format:

1. Problem
2. Set up the Bit
3. Introduce the Flip
4. Instantiate the flip in a solution
5. Evaluation Plan
6. Implications

Next class

Topic 1 lecture: approximate query processing

Your task:

- Sign up for presentation: https://tinyurl.com/2s3amhrv
- First paper review due next Tuesday (08/29) midnight
- Come to my OH if you need help preparing your presentations