

Senior Seminar - 399

Adam Poliak

Spring 2025

Lecture 01



BRYN MAWR
COLLEGE



What is Computer Science?



Computer Science: Not about Computers, Not Science

Kurt D. Krebsbach

Department of Mathematics and Computer Science, Lawrence University, Appleton, Wisconsin 54911

Abstract—*This paper makes two claims about the fundamental nature of computer science. In particular, I claim that—despite its name—the field of computer science is neither the study of computers, nor is it science in the ordinary sense of the word. While there are technical exceptions to both claims, the nature, purpose, and ultimately the crucial contributions of the beautiful discipline of computer science is still widely misunderstood. Consequently, a clearer and more consistent understanding of its essential nature would have an important impact on the awareness of students interested in computing, and would communicate a more informed perspective of computer science both within academia and in the larger society.*

first computer scientists, and that his method for computing the greatest common divisor (*GCD*) of any two positive integers is regarded as the first documented algorithm. As Donald Knuth—author of the discipline’s definitive multi-volume series of texts on algorithms—states: “We might call Euclid’s method the granddaddy of all algorithms, because it is the oldest nontrivial algorithm that has survived to the present day” [3]. Knuth’s version of Euclid’s *GCD* algorithm [4] is shown in Figure 1.

E1.	[Find remainder.]	Divide m by n and let r be the remainder. (We will have $0 \leq r < n$.)
E2.	[Is it zero?]	If $r = 0$ the algorithm terminates; n is the answer.
E3.	[Repeat.]	Go to E1.



Computer Science, = Telescope science ?



*"Computer science is no more about computers
than astronomy is about telescopes"*

-- Edsger W. Dijkstra



*"The computer is not our object of study,
It's our observational instrument"*

Object of Study in Computer Science

Algorithms!

Ordered list of instructions that are:

Unambiguous

Expressive – communicate a lot of idea

Study of algorithms:

Developing algorithms that are:

correct

efficient



Subfields of Computer Science



Arxiv -> CS

arXiv > cs

Computer Science (since January 1993)

or a **specific paper**, enter the identifier into the top right search box.

- **Browse:**
 - [new](#) (most recent mailing, with abstracts)
 - [recent](#) (last 5 mailings)
 - [current month's](#) listings
 - specific year/month:
- **Catch-up:**
Categories:
Changes since: , view results abstracts
- **Search** within the [cs archive](#)
- Article statistics by year:
[2025](#) [2024](#) [2023](#) [2022](#) [2021](#) [2020](#) [2019](#) [2018](#) [2017](#) [2016](#) [2015](#) [2014](#) [2013](#) [2012](#) [2011](#) [2010](#) [2009](#) [2008](#) [2007](#) [2006](#) [2005](#) [2004](#) [2003](#) [2002](#) [2001](#) [2000](#) [1999](#) [1998](#) [1997](#) [1996](#) [1995](#) [1994](#) [1993](#)

Categories within Computer Science

- **cs.AI – Artificial Intelligence** ([new](#), [recent](#), [current month](#))
Covers all areas of AI except Vision, Robotics, Machine Learning, Multiagent Systems, and Computation and Language (Natural Language Processing), which have separate subject areas. In particular, includes Knowledge Representation, Planning, and Uncertainty in AI. Roughly includes material in ACM Subject Classes I.2.0, I.2.1, I.2.3, I.2.4, I.2.8, and I.2.11.
- **cs.AR – Hardware Architecture** ([new](#), [recent](#), [current month](#))
Covers systems organization and hardware architecture. Roughly includes material in ACM Subject Classes C.0, C.1, and C.5.
- **cs.CC – Computational Complexity** ([new](#), [recent](#), [current month](#))
Covers models of computation, complexity classes, structural complexity, complexity tradeoffs, upper and lower bounds. Roughly includes material in ACM Subject Classes F.1 (computation by abstract devices) and F.2 (complexity theory). Some material in formal languages may be more appropriate for Logic in Computer Science. Some material in F.2.1 and F.2.2, may also be appropriate here, but is more likely to have Data Structures and Algorithms.



Arxiv -> CS

Artificial Intelligence, Hardware Architecture, Computational Complexity, Computational Engineering, Finance, and Science, Computational Geometry, Computation and Language, Cryptography and Security, Computer Vision and Pattern Recognition, Computers and Society, Databases, Distributed, Parallel, and Cluster Computing, Digital Libraries, Discrete Mathematics, Data Structures and Algorithms, Emerging Technologies, Formal Languages and Automata Theory, General Literature, Graphics, Computer Science and Game Theory, Human-Computer Interaction, Information Retrieval, Information Theory, Machine Learning, Logic in Computer Science, Multiagent Systems, Multimedia, Mathematical Software, Numerical Analysis, Neural and Evolutionary Computing, Networking and Internet Architecture, Other Computer Science, Operating Systems, Performance, Programming Languages, Robotics, Symbolic Computation, Sound, Software Engineering, Social and Information Networks, Systems and Control



Google Scholar: Engineering & CS Subcategories

Categories > Engineering & Computer Science > Subcategories ▾

Subcategories	Databases & Information Systems	Ocean & Marine Engineering
Architecture	Educational Technology	Oil, Petroleum & Natural Gas
Artificial Intelligence	Engineering & Computer Science (general)	Operations Research
Automation & Control Theory	Environmental & Geological Engineering	Plasma & Fusion
Aviation & Aerospace Engineering	Evolutionary Computation	Power Engineering
Bioinformatics & Computational Biology	Food Science & Technology	Quality & Reliability
Biomedical Technology	Fuzzy Systems	Radar, Positioning & Navigation
Biotechnology	Game Theory and Decision Science	Remote Sensing
Ceramic Engineering	Human Computer Interaction	Robotics
Civil Engineering	Library & Information Science	Signal Processing
Combustion & Propulsion	Manufacturing & Machinery	Software Systems
Computational Linguistics	Materials Engineering	Structural Engineering
Computer Graphics	Mechanical Engineering	Sustainable Energy
Computer Hardware Design	Medical Informatics	Technology Law
Computer Networks & Wireless Communication	Metallurgy	Textile Engineering
Computer Security & Cryptography	Microelectronics & Electronic Packaging	Theoretical Computer Science
Computer Vision & Pattern Recognition	Mining & Mineral Resources	Transportation
Computing Systems	Multimedia	Water Supply & Treatment
Data Mining & Analysis	Nanotechnology	Wood Science & Technology



Organizing CS subfields

Theory

- formal frameworks and theoretical underpinnings that guide the development of computing systems and software

Systems

- design, implementation, analysis, and evaluation of computer systems and associated software.

Applications


- applying computing and technology to solve problems across various domains




Arxiv -> CS

Artificial Intelligence, Hardware Architecture, Computational Complexity, Computational Engineering, Finance, and Science, Computational Geometry, Computation and Language, Cryptography and Security, Computer Vision and Pattern Recognition, Computers and Society, Databases, Distributed, Parallel, and Cluster Computing, Digital Libraries, Discrete Mathematics, Data Structures and Algorithms, Emerging Technologies, Formal Languages and Automata Theory, General Literature, Graphics, Computer Science and Game Theory, Human-Computer Interaction, Information Retrieval, Information Theory, Machine Learning, Logic in Computer Science, Multiagent Systems, Multimedia, Mathematical Software, Numerical Analysis, Neural and Evolutionary Computing, Networking and Internet Architecture, Other Computer Science, Operating Systems, Performance, Programming Languages, Robotics, Symbolic Computation, Sound, Software Engineering, Social and Information Networks, Systems and Control

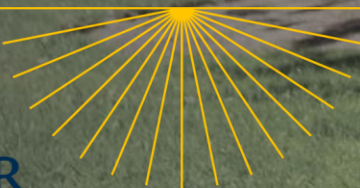





My current projects focus on analyzing motion from video and building multi-modal characters for video games. I primarily use a mix of experimental (e.g. user studies) and modeling approaches to understand the effect of different design decisions on non-player characters, avatars, and game design.




Curves and Surfaces Fitting, Mesh Generation and Optimization, Computer Aided Geometric Design and Computational Geometry in general. I study applied problems in Computer Graphics, Vision and Imaging, with methods strongly rooted in geometric analysis and algorithms. In addition, I am also interested in geometric and topological methods in data analysis and visualization



focusing on the integration of symbolic program analysis and neural techniques to improve software correctness. Methods to productively write secure and correct programs are imperative in our society, which is underpinned by software systems. Software bugs and vulnerabilities are commonplace and can have devastating impacts. Currently, existing techniques to analyze programs for faults have fundamental limitations, preventing widespread deployment. Elizabeth seeks to address their shortcomings with the strengths of neural models. Ultimately, her research objective is to create program analysis techniques that are effective and reliable through Cooperative Program Reasoning and Neural Modeling



I study the reasoning capabilities and biases in natural language processing models. I also apply statistical and computational techniques to other fields to glean insights from large amounts of text



Artificial Intelligence: Intelligent agent architectures, knowledge representation & reasoning, planning and acting, computational linguistics, robotics, machine learning, developmental robotics.

Computer Science Education: Pedagogy of computer science, curriculum design, inclusive computing education.

CS subfields & BMC faculty



Theory

Systems



Applications





CS 399 Overview

Goals of 399

Gain a deep understanding of the seminal works that have shaped the various subfields of computer science.

Cultivate the ability to critically analyze and interpret complex scientific texts and methodologies.

Refine oral and written communication skills through regular discussions, presentations, and written assignments.

Engage in thoughtful reflection and critique of the material, thereby enhancing problem-solving and critical thinking abilities.



Logistics

Course website

- <https://cs.brynmawr.edu/cs399/>

Submitting Homeworks:

- Gradescope
- Course entry code: **DKW8BG**



Course Workload

Weekly Reading & Written Responses

Presentations

Final Report



Weekly Readings

2 – 3 research papers a week

Written response to each, about 2 pages, Latex

Latex/Overleaf template – on course website (waiting for Overleaf Gallery approval)

Upload pdf to Gradescope

Short quizzes



Presentations

2 students will present the weekly papers

25-30 minute presentations discussing the paper

Presenters do not have to write a response that week

Sign up sheet will go out shortly

Weeks 3 – 14 (not 12 & 13)



Final Report

Deep dive in subfield of your choosing

Read about 6 papers on a specific theme in that subfield

Write a 6 page response/report

Required to be done in pairs



Grading

Item	%
Weekly Paper Responses + Quizzes	40%
Final Report/Paper	30%
Presentation	20%
Participation	10%



Weekly Responses Grading

Answer the question/understand the paper

Quality of Writing:

use Grammarly

use Matt Might's bash scripts



On technical writing (Matt Might)

Weasel words

- Salt & Pepper Words
- Beholder Words
- Lazy words
- Adverbs

Passive voice

- Bad: Termination is guaranteed on any input.
- Better: Termination is guaranteed on any input by a finite state-space.
- OK: A finite state-space guarantees termination on any input.



Todos

Course welcome survey

- <https://forms.gle/a847mPb8atKNarh58>

Reading for next week

Sign up for presentation

- <https://forms.gle/kb7xKKWjymNNhJEU5>

