UNIVERSITY OF MICHIGAN
College of Engineering
Curriculum Committee Meeting
Tuesday, September 3, 2024

Attending: Varun Agrawal, Achilleas Anastasopoulos, Yavuz Bozer, Chris Fidkowski, Saadet Albayrak Guralp, Vineet Kamat, Amir Kamil, Leena Lalwani, Ryan Latimer, Xiaogan Liang, Yili Liu, Frank Marsik, Radoslaw Michalowski, Deepak Nagarth, Kristel Oelke, Nolgi Oquendo-Colon, Yulin Pan, Elyse Vigiletti, Won Sik Yang

Support Staff: Mercedes Carmona, Betsy Dodge, Matthew Faunce

Call to Order: 1:35 PM

Adjourned: 2:57 PM

Agenda:

1. Voting for CoE Curriculum Committee Chair for 2024-2025
    a. Amir Kamil nominated and unanimously voted upon by Curriculum Committee to be the chair for the next academic year.
2. Approval of 4.9.2024 Meeting Minutes – Page 3 - **APPROVED**
3. Proposal for New CARF Email – Information Item – Page 6
    a. New CARF email created that will direct questions, information, and requests directly to the CoE Curriculum Committee administrative support needed to review CARFs, answer questions, and provide information.
    b. An email will be sent to all CoE CC members and support staff to start using this new email as soon as possible. All website and document locations with CARF information will be updated to reflect the new email.
4. Proposal for ROB 102 for EECS-CSE – Action Item – Page 7 - **APPROVED**
    a. EECS-CSE to request that ROB 102 be approved as a substitution for the ENGR 101 requirement for CS-Eng majors. This would be beneficial to add this course as another option for students as EECS 180 and ENGR 151 are currently offered along with ENGR 101.
    b. ROB questions that due to the influx of CS-Eng students being directed to take ROB 102, will the ROB department and instructors be prepared and what influence, if any, will this have on the course taught, such as will the learning outcomes be altered for CS-Eng students. Also, what is the communication between EECS-CSE and Robotics for this type of change.
        i. EECS-CSE says that there have been ongoing conversations with the department and there will be no altering of how ROB 102 is currently taught. Conversations will happen between departments if any issue(s) arise.
            1. Graduate Education asks if a letter of support from the involved departments would need to be created and questions what department is to make changes to a course? What does the workflow for a change like this look like?
                a. Chair says that communication needs to happen between departments with these types of changes and any changes for proposals and CARFs. All departments need to be involved in ongoing conversations before creating a proposal or CARF to be presented at a CoE CC meeting. Currently, there is no workflow, but it is expected departments are having the conversations needed for CARFs and proposals.
    c. CLaSP requests what are the specific numbers for the declared degrees provided in the proposal and why the EAC advised students to take ROB 102. Also expresses concern for the ROB department to not be overwhelmed with an influx of students taking this course.
        i. Currently, there are 35 students enrolled in ROB 102, which is the most this has ever been. This could be due to students who initially wanted to pursue ROB but have changed their mind or want to pursue a dual degree. Also, other majors do not accept ROB 102.
    d. CoE CC members voted unanimously to approve this proposal. The proposal will appear at the next CoE Faculty meeting.

5. Proposal for IOE Undergraduate Minor in Human Factors Engineering (HFE) – Action Item - Page 9 - **APPROVED**
    a. The IOE HFE Minor requires 15 credits of IOE courses and is open to all undergraduate students except students pursuing IOE majors, who can take these HFE courses to gain knowledge and skills to be used as part of the major requirements. Upon approval in the CoE, the IOE department will pursue other schools/colleges within the University. This proposal is to have an effective term of Winter 2025.
        i. Other minor requirements:
            1. Students must declare a Major, other than IOE, before declaring the HFE Minor. No additional prerequisite is required for the HFE minor.
            2. No transfer credit allowed.
            3. Students must have at least a 2.0 GPC for the 15 credits of core and elective HRE courses for the minor.
            4. Courses must be a regular grade and receive a C- or better.
    b. EECS-CSE questions how many students would pursue this minor.
        i. Not a specific number to provide, but rather the departments that would be interested would be ROB, EECS-CS, BIOMEDE, and AEROSP due to the courses offered.
    c. Graduate Education asks what type of marketing or communication is being used to spread the word about this minor and would this minor be a concentration for graduate students.
        i. Upon approval, our department will work on this to market to all students interested in this minor. The minor will only be offered to Undergraduate students, but this is the first step for the minor and to adapt/progress in the future once this minor begins.
    d. CoE CC members voted unanimously to approve this proposal. The proposal will appear at the next CoE Faculty meeting.

CARF SUMMARIES

PAGE	SUBJECT	COURSE #	ACTION	SUMMARY	EFFECTIVE TERM	MIN. GRADE REQ. FOR ENF. PREPREQ	Is Course on LSA Course Guide?	APPROVED	NOTES & REVISIONS	TABLED
13	BIOMEDE	514	NEW		WT 2025	C-	NO	CONDITIONAL APPROVAL	Add parenthesis around the Enforced Prerequisite courses (BIOMEDE 221 and BIOMEDE 418). Modify Course Description, specifically the last sentence and use present tense throughout.	
21	CSE	577	NEW		WT 2025	C	NO	APPROVED		
39	EECS	110	MOD	Change in Course Description and Enforced Prerequisite.	WT 2025	NO	YES	APPROVED		
42	EECS	183	MOD	Change in Enforced Prerequisite.	WT 2025	NO	YES	APPROVED		
45	EECS	477	MOD	Change in Credit Exclusions.	WT 2025	C	YES	APPROVED		
48	EECS	479	NEW		WT 2025	C	NO	APPROVED		
67	EECS	487	MOD	Change in Credit Exclusions.	WT 2025	C	YES	APPROVED		
70	EECS	492	MOD	Change in Credit Exclusions.	WT 2025	C	YES	APPROVED		
73	IOE	333	MOD	Change in Course Description and Enforced Prerequisite.	WT 2025	NO	YES	APPROVED		

UNIVERSITY OF MICHIGAN
College of Engineering
Curriculum Committee Meeting
Tuesday, April 9, 2024

Attending: Achilleas Anastasopoulos, Robert Bordley, Yavuz Bozer, Chris Fidkowski, Fei Gao, Saadet Albayrak Guralp, Amir Kamil, Leena Lalwani, Xiaogan Liang, Emmanuelle Marquis, Frank Marsik, Radoslaw Michalowski, Mika Panagou, Eric Rutherford, Rachael Schmedlen, Ben Spector, Elyse Vigiletti, Roxanne Walker

Support Staff: Mercedes Carmona, Betsy Dodge, Matthew Faunce

Call to Order: 1:35 PM

Adjourned: 2:10 PM

Agenda:

1. Approval of 3.26.2024 Meeting Minutes – Page 3 - **APPROVED**
2. CSE CS-Eng Program Modification – Action Item – Page 7 – **APPROVED**
    a. The Computer Science and Engineering division of the EECS department was contacted to adjust the Computer Science (CS-Eng) Major Sample Schedule as this listed ROB 101 as a program substitution for MATH 214, which had not yet gone through the CoE Curriculum Committee for approval.
    b. Officially, the request is that ROB 101 be approved as a substitution for the linear algebra requirement for CS-Eng majors. The Robotics department has been in contact and is in approval with this course substitution for CS-Eng students. ROB 101 allows students to meet degree requirements as needed as MATH 214 may be at a max capacity and not allow students to take the course for the students' intended term. There is not an intention to change the sample schedule yet as most students take MATH 214 or 217, so the schedule will stay for now with an intention to apply changes for current and future CS-Eng students for Fall 2024, likely Winter 2025, effective term.
        i. A question is raised about what background does a student need to take ROB 101 and/or if the student needs to have an interest in Robotics or Computer Science?
            1. ROB 101 does not have any prerequisites. Taking this course earlier is beneficial for Computer Science students as it provides more mathematical grounding prior to the math/theory requirements that are specific to the Computer Science major.
                a. MATH 214 contains prerequisites and ROB 101 does not. Would any prerequisites need to be implemented for ROB 101?
                    i. CSE Department says if any prerequisites need to be included for the course, this will be discussed with the Robotics department and a modification CARF would be created for any changes needed.
        ii. Another inquiry about what coding is being presented in ROB 101?
            1. Robotics department states this is a 100-level course and the coding that is covered is included in the course description, Julia programming language.
        iii. If a department wanted to discuss similar changes, how are the departments involved communicating such as the course counting for another department's degree requirements?
            1. CSE contacted the ROB 101 instructors as well as the CoE Curriculum Committee Robotics department representative and began the discussion and understanding on how the course would need to be implemented for the department's degree requirements. Communication needs to take place so that all departments involved are on the same page.
                a. With the influx of EECS students taking ROB 101, would course topics or learning outcomes be altered in the future for CSE students rather than Robotics students? Are LSA CS-Eng students also taking ROB 101?

i. CSE says there would need to be a discussion had with the Robotics department if the course material needed to be altered, but the current way ROB 101 is taught will stay the same.
ii. LSA will not allow ROB 101 to be taken as a substitution for MATH 214 as Calculus II, MATH 116, is still needed as a course perquisite. Including LSA Students for ROB 101 would be a significant enrollment with further discussions needing to be had with all departments involved, Robotics, Mathematics, and CSE.

c. CoE Curriculum Committee members voted unanimously to approve the program proposal. The proposal will appear at the next CoE Faculty meeting in Fall 2024.

3. Engineering Physics BSE Program Modification – Action Item – Page 9 – **APPROVED**
   a. The LSA Physics department has updated the course number and title for PHYSICS 351 – Methods of Theoretical Physics I to PHYSICS 316 – Mathematical Methods for Physics and Engineering. The department states this is a cosmetic change to try to encourage students to take the course earlier in their college career. This change will not have an impact on current students with the planned change to take effect in Winter 2025.
   b. CoE Curriculum Committee members voted unanimously to approve the program proposal. The proposal will appear at the next CoE Faculty meeting in Fall 2024.
4. MSE UG Minor Modification – Action Item – Page 11 - **APPROVED**
   a. Per undergraduate students, the current MSE Minor is “too easy” and lacks rigor for MSE students to be prepared for fundamentals concepts. To address these concerns, the MSE department proposes to modify the courses that fulfill the minor requirements, which also increases the minor credits from 17 to 19 credits.
      i. An issue is raised by the MECHENG and BIOMEDE departments due to MATSICE 330 and the coverage of Thermodynamics as some content may be repeated.
         1. There will be very little overlap as content covered for each course is as follows:
            a. BIOMEDE 221 = Coverage overlap of 3 weeks and the rest of the MATSCIE 330 content is different.
            b. MECHENG 335 = Coverage overlap of 2 weeks and the rest of the MATSCIE 330 content is different.
   b. CoE Curriculum Committee members voted unanimously to approve the program proposal. The proposal will appear at the next CoE Faculty meeting in Fall 2024.

CARF SUMMARIES

PAGE	SUBJECT	COURSE #	ACTION	SUMMARY	EFFECTIVE TERM	MIN. GRADE REQ. FOR ENF. PREPREQ	Is Course on LSA Course Guide?	APPROVED	NOTES & REVISIONS	TABLED
47	CEE	481	MOD	Change in Enforced Prerequisite.	WT 2025	C-	YES	APPROVED		
50	ECE	551	MOD	Change in Course Description and Advisory Prerequisite.	WT 2025	NO	YES	APPROVED		

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

July 30, 2024

Hello CoE Curriculum Committee:

The CoE Registrar's Office received a request to create an email for all CARF related requests, questions, and information so CoE CC Members and Maintainers can track CARF emails separately from their work emails. The CoE RO proposes that a new email be created that the CoE CC can use for CARFs, which would be titled, engin-carf@umich.edu.

With the creation of the new email for all CoE CC Members and Maintainers, this will reduce confusion and direct CARF communication directly to the CoE RO members involved in the CoE CC. Once approved,the new email address will be updated on the CoE CC website and course related documents as well as any other documents or websites that pertain to this information. The CoE RO will also send an email to all CoE CC Members and Maintainers regarding the new email address.

We hope to implement this new email address effective for Fall 2024. If you have any questions, please reach out to Mercedes Carmona, carmonam@umich.edu, or Betsy Dodge, elibunce@umich.edu.

Thank you,
CoE Registrar's Office

COLLEGE OF ENGINEERING
COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF MICHIGAN

AMIR KAMIL

UNIVERSITY OF MICHIGAN
COLLEGE OF ENGINEERING
COMPUTER SCIENCE AND ENGINEERING

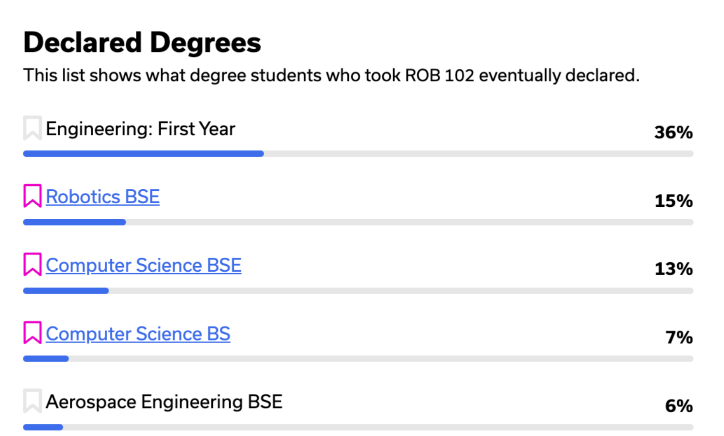
2260 HAYWARD STREET
ANN ARBOR, MI 48109-2121

August 8, 2024

Dear CoE Curriculum Committee:

I am writing on behalf of the Computer Science and Engineering (CSE) division to request that ROB 102 Introduction to AI and Programming be approved as a substitution for the ENGR 101 requirement for CoE Computer Science (CS-Eng) majors. Other introductory programming courses, including ENGR 151 and EECS 180, are already approved alternatives for ENGR 101, and we think it would be beneficial to add ROB 102 to the set of options for our students.

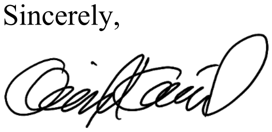
ROB 102 introduces students to computer programming using the C++ and Python languages, with an emphasis on applications to robotics and artificial intelligence. The instructors for EECS 280, the second programming course required of CS-Eng majors, have determined that ROB 102 sufficiently prepares students to take EECS 280 and further courses in CS. Students often take ROB 102 as their first programming course, discover an interest in Computer Science, and go on to major in CS – Atlas data show that at least 20% of the students who take ROB 102 proceed to declare either CS-Eng or CS-LSA (the data also show 36% of students as Engineering: First Year, some of whom will eventually pursue CS, so the 20% number likely undercounts the fraction of ROB 102 students who go on to do CS. We would like to support students who take ROB 102, without requiring them to take an additional 100-level programming course that covers much of the same material as ROB 102).



The addition of ROB 102 as an option for introductory programming does not require a change to the sample schedule for CS-Eng. We would like the addition to apply to both current and future CS-Eng students, with an effective term of Fall 2024.

Thank you for considering this modification. Please direct any questions to me and to Julie Tashjian (jbtash@umich.edu), who oversees the CS Undergraduate Advising Office.

Sincerely,



COLLEGE OF ENGINEERING
COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF MICHIGAN

Amir Kamil
Chair of the Computer Science Undergraduate Program Committee
Lecturer IV, Computer Science and Engineering
University of Michigan
akamil@umich.edu

DEPARTMENT OF INDUSTRIAL AND OPERATIONS ENGINEERING

UNIVERSITY OF MICHIGAN

G622 IOE BUILDING
1205 BEAL AVENUE
ANN ARBOR, MICHIGAN 48109-2117
734 763-0464 FAX: 734 764-3451

Proposal to Establish an Undergrad Minor in Human Factors Engineering (HFE)
April 3, 2024

U-M Department of Industrial and Operations Engineering
Approved by U-M IOE Department Faculty on April 3, 2024
Contact Person: Prof. Yili Liu or Prof. Yavuz Bozer
Proposed Effective Term for the HFE Minor: Starting from Winter 2025

I. Introduction and Description of the HFE Minor

Human Factors Engineering (HFE) is a scientific and engineering discipline that aims to improve society and people's lives by making technology work better for people. HFE strives to improve products, workplaces, work procedures, and organizations to make them comfortable, efficient, enjoyable, productive, safe, and usable to diverse global human populations. HFE professionals discover, apply, and disseminate knowledge of human behavior in human-machine-environment systems for human-centered engineering and design.

Clearly, Human Factors Engineering is highly relevant to all fields of engineering. HFE knowledge and skills are beneficial to all the students in the UM College of Engineering whose motto is "People-First Engineering." We therefore propose to establish an undergrad Minor in Human Factors Engineering that is sponsored and administered by the Department of Industrial and Operations Engineering (IOE).

This HFE minor requires 15 credits of IOE courses (please see Part II of this document for a detailed description of the *Curricular Plan* for the HFE Minor) and is open to all undergraduate students in the UM College of Engineering except students whose undergrad major is IOE. IOE students can take the HFE courses and acquire HFE knowledge and skills as part of the curriculum requirements for their IOE major. A description of the *Administrative Plan* for the HFE minor can be found in Part III of this document.

After this HFE minor is approved and established in the College of Engineering, further steps will be taken to explore the possibilities and processes of opening this minor to undergraduate students in some other UM colleges and units such as the College of LSA, College of Architecture and Urban Planning, School of Art and Design, School of Business, School of Education, School of Information, School of Kinesiology, etc.

II. Curricular Plan for the HFE Minor

Academic Minor Program in HFE (complete at least 15 credits of IOE courses as listed below):

1. **Required Core Course for the HFE Minor (One)**:

   IOE 333: Introduction to Human Factors and Ergonomics (3 credits)

2. **Elective HFE Courses for the HFE Minor:** Select at least 12 credits from the following courses:

   IOE 430: Global Cultural Systems Engineering
   IOE 431: Human-centered and User Experience Design
   IOE 434: Human Error and Complex System Failures
   IOE 435/535: Quantifying Human Motion Through Wearable Sensors
   IOE 436: Human Factors in Computer Systems
   IOE 437: Automotive Human Factors
   IOE 438: Occupational Safety Management
   IOE 463: Measurement and Design of Work
   IOE 465/570: Design of Experiments
   IOE 533: Human Motor Behavior and Engineering Systems
   IOE 534: Occupational Biomechanics
   IOE 536: Cognitive Ergonomics and Human System Integration
   IOE 539: Safety Engineering Methods
   IOE 563: Advanced Work Design: Volunteer Work

Rules/Policies:

1. Courses taken to meet the requirements of the HFE minor must be taken for a regular grade and receive a grade of C- or better.
2. Students must declare a major (other than IOE) before declaring the HFE minor. No additional prerequisite is required for the HFE minor.
3. No transfer credit is allowed for the HFE minor.
4. Students must have at least a 2.0 GPA for the 15-credits of core and elective HFE courses to complete the minor.
5. The HFE minor will follow all U-M rules on undergrad minors.

III. Administrative Plan for the HFE Minor

Program Advisor (faculty) for the HFE Minor (Yili Liu)
Student Advisor (staff) for the HFE Minor (Leonora Lucaj)

Their duties include but are not limited to:

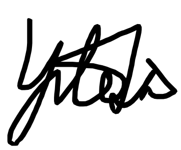
1. Advising students who are interested in and/or already enrolled in the HFE minor program on the academic requirements, career opportunities, and other related questions that students may have about the minor.
2. Maintaining the related academic records of students in the HFE minor, and timely auditing of the minor requirements upon request and during the student's final term. The program advisor will be responsible for approving variances to the minor requirements for individual students.
3. The program advisor and their designated staff are responsible for responding to the CoE Registrar's request for the audit of the minor requirements. Students who declare and complete a minor such as the HFE minor will have a notation on their transcript but not on their diploma.

Questions about the HFE Minor:

Contact: Professor Yili Liu and/or IOE Undergraduate Advising Office
Location: 1729 IOE
Email Address: yililiu@umich.edu or lucajl@umich.edu

Course Information and Planning: Students can reference the IOE course descriptions and Atlas for a basic introduction to IOE courses. Course syllabi for some courses are also available upon request or via course websites. For more planning assistance, students are welcome to email (Yili Liu and/or Leonora Lucaj) and/or schedule an appointment with them.

Thank you for your consideration.



Yili Liu

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

KEVIN P. PIPE
Associate Dean for Undergraduate Education, College of Engineering
Professor of Mechanical Engineering, Electrical Engineering & Computer Science, and Applied Physics
1261 LEC / 1221 Beal Ave. / Ann Arbor, MI 48109-2102
pipe@umich.edu / (734) 647-7150

July 8, 2024

CoE Curriculum Committee
College of Engineering
University of Michigan
145A Chrysler Center
Ann Arbor, MI 48109-2092

Dear CoE Curriculum Committee,

I am writing to express my strong support for the creation of a minor in Human Factors Engineering (HFE) by the Industrial and Operations Engineering Department. This topic has broad relevance across engineering disciplines and will provide important training to students interested in understanding human behavior in human-machine-environment systems.

The proposed minor will have an operational home within the IOE Department and will be offered according to our relevant policies, which state that a minor is a coherent program of study allowing depth in the exploration of a topic outside the student's major (15 credits or 4+ courses). This opportunity will be open to all CoE undergraduate students whose major is not IOE; IOE students have the same courses available for further study within their degree program. I also agree with the proposal's anticipation that students from outside CoE will be interested in this opportunity, and support IOE's subsequent exploration of broadening the minor to other units.

Sincerely,

Kevin Pipe
Associate Dean for Undergraduate Education

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ CHECK APPROPRIATE BOXES FOR ALL CHANGES

Action Requested
- ☑ New Course
- ☐ Modification of Existing Course
- ☐ Deletion of Existing Course

Date of Submission: 2024-08-14
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			REQUESTED LISTING		
☑	Dept (Home): Subject: Catalog:			Dept (Home): Biomedical Engineering Subject: BIOMEDE Catalog: 514		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☑	Course Title (full title)			Course Title (full title) Systems Biology of Human Disease		
☑	Abbreviated Title (20 char)			Abbreviated Title (20 char) Syst. Bio Hum Diseas		
☑	Course Description (Please limit to 80 words and attach separate sheet if necessary) An introduction of skills and concepts necessary for the application of systems-biology approaches to human diseases. Emphasis will be to develop a recipe for maintenance of homeostasis for normal function of various organs. Will analyze complex disease states using engineering principles and use engineering concepts to arrive at disease solutions. At the end, each student should be able to develop a research outline that could form the core of a systems-based engineering solution for a particular human disease.					
☑	Full Term Credit Hours Undergraduate Min: 3 Undergraduate Max: 3	Graduate Min: 3 Graduate Max: 3		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☑	Course Credit Type Undergraduate Student, Rackham Graduate Student, Non-Rackham Graduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Biomedical Engineering Catalog: 514

☑	Grading Basis ☑ Graded (A – E) ☐ Credit/No Credit ☐ Satisfactory/Unsatisfactory ☐ Pass/Fail ☐ Business Administration Grading ☐ Not for Credit ☐ Not for Degree Credit ☐ Degree Credit Only Add Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent Drop Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☑	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char) Biology 172
☑	Enforced Prerequisite (254 char)	Enforced Prerequisite (254 char) Biomede 221, AND Biomede 418, OR Graduate Standing Minimum grade requirement: C-
☐	Credit Exclusions	Credit Exclusions

	Course Components	Graded Component	Terms Typically Offered
☑	☑ Lecture	☑	☑ Fall
	☐ Seminar	☐	☑ Winter
	☐ Recitation	☐	☐ Spring
	☐ Lab	☐	☐ Summer
	☐ Discussion	☐	☐ Spring/Summer
	☐ Independent Study	☐	

Cognizant Faculty Member Name: Deepak Nagrath Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Chris Mueller Email: muchris@umich.edu Phone: 734 647 8040

CoE Curriculum Committee Representative:	Rachael Schmedlen	Print: Rachael Schmedlen	Date: 8/19/24
CoE Curriculum Committee Chair:		Print:	Date:
Home Department Chair:	Ariella Shikanov	Print: Ariella Shikanov	Date: 08/07/2024
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:	**Requested:**
Course Description	Course Description An introduction of skills and concepts necessary for the application of systems-biology approaches to human diseases. Emphasis will be to develop a recipe for maintenance of homeostasis for normal function of various organs. Will analyze complex disease states using engineering principles and use engineering concepts to arrive at disease solutions. At the end, each student should be able to develop a research outline that could form the core of a systems-based engineering solution for a particular human disease.
Class Length	Class Length Full term
Contact hours (lecture):	Contact hours (lecture): 3
Contact hours (recitation)	Contact hours (recitation)
Contact hours (lab)	Contact hours (lab)

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:
Engineering Expertise (UGrad); Technical Elective or Core Course elective (Grad)

Special resources of facilities required for this course:

Supporting statement:
This course has run with a 599 course number for several years and we are now making it a permanent course. This fulfills a need in our curriculum for courses surrounding human diseases.

BIOMEDE 599
Systems Biology of Human Diseases
Syllabus – Fall 2022

General Course Information

Course Description: An important goal in medicinal biology is to uncover the underlying principles that curtail human life. Hence, this course will introduce and develop skills and concepts necessary for application of systems-biology approaches to human diseases. Emphasis will be to develop a recipe for maintenance of homeostasis for normal function of various organs. The course will illustrate the transcriptional and metabolic design principles as applied to cellular systems in the healthy and diseased biological states. This graduate level course will introduce the various regulatory network motifs occurring in various diseases such as metabolic syndrome, cancer, diabetes and potential treatments using embryonic stem cells. The course will analyze these complex disease states using engineering principles and concerted efforts will be made towards using engineering concepts such as optimality, nonequilibrium thermodynamics, dynamics, and spatiotemporal transport, to arrive at disease solutions. At the end of this course, each student should be able to develop, a research outline that could form the core of a systems-based engineering solution for a particular human disease.

Goal:

- Model the cellular behavior
- Understand how cells interact with their environments
- Response of cells to perturbations in environments (both intracellular and extracellular)
- Understanding diseased state in the cell
- Interaction of multiple cell types
- Role of multiple cell types in either flaring or repressing the diseased conditions
- Transition from health to disease

Diseases: Diabetes, Cancer, Aging, Covid19, Autoimmunity, Obesity, and Hormonal diseases

I. Teaching Staff

Instructor: Deepak Nagrath, Associate Professor in Department of Biomedical Engineering. Office located in NCRC Bldg 28, 3048W. Email: dnagrath@umich.edu.

Office Hours: We will use Zoom Meetings for Office Hours. The dates and time will be discussed based on class poll.

II. Textbook and Resources

Reference Textbooks:

An Introduction to Systems Biology, Uri Alon, Second Edition.

Fundamentals of Systems Biology, Markus Covert.

Chemical Biophysics. *Quantitative Analysis of Cellular Systems, Daniel Beard and Hong Qian.*

Fundamental of Enzyme Kinetics. Athel Cornish-Bowden

Free Energy Transduction and Biochemical Cycle Kinetics, Terrence L. Hill

Metabolic Engineering, Gregory N. Stephanopoulos

Computational Modeling of Genetic and Biochemical Networks, Bower and Bolouri

Handouts and updates on course information will be posted on the Canvas website. It is important that you be able to access this page if you are taking this course for a grade. If you do not have access to this page, please contact me.

We will use Copasi for analyzing networks. Matlab will also be used for homework problems.

III. **Table of Contents**

Periodic Tables of Diseases, Aging, Immune System Networks, Covid, Feedback between Organs, Transition from health to disease, Oscillations, Bifurcations, Feedback, Structure-Function Relationship in Biological Circuits, Stability Analysis, Circadian Rhythms, Transcriptional Networks, Epithelial to Mesenchymal transition in Cancer, and Optimality in Biological Networks.

Recent papers on above topics will be discussed for 10-12 min in every lecture.

Grading

Grading will be determined by a combination of homework (30 %), class presentations (20 %), a final project (50 %). Class presentations will be evaluated by the whole class including GSI and the instructor.

There will be around three homework sets. The final project will be done in a group of **three** students with interdisciplinary background. For the classroom participation part of the grade, each group is required to give a short presentation on final project. The presentation will be a summary of what students accomplished in their application of engineering principles on human disease along with a background of the problem, a discussion of methods used, possible limitations or unanswered questions, and future work that could be done to extend the conclusions of the paper.

Each group will be assigned a date on which to present. The presentations will be a maximum of 45 minutes and 30 slides, and time will be given after the presentation for follow up questions and discussion. The purpose of this exercise is to encourage you to apply what you learnt in class in your thesis research and to give you the opportunity to share with the rest of the class what methodology you find interesting in applying for a disease or any other system. Please see Canvas (Files) for a list of journals that regularly publish work in the field.

Please do not post any notes, class materials on any social media, web interfaces, etc.

IV. Honor Code Policy

You are encouraged to have discussions with your peers in this course.

HomeWorks are designed to assess individual retention of the course material and to motivate you to think more deeply about the course subject. We allow you to discuss the homework problems and assignments, but each student should formulate his or her own independent solution to each assignment. Copying is strictly forbidden.

V. Accommodations

Any student with a documented disability seeking academic adjustments or accommodations is requested to speak with me during the first two weeks of class. All such discussions will remain as confidential as possible. Students with disabilities will need to also contact Disability Support Services.

V. Late Homework Policy

Homework turned in late will be penalized 2 % for each hour it is turned in late and 10 % for each day the homework is turned in late. After 4 days (including Saturday and Sunday) late homework will no longer be accepted.

University of Michigan
Fall 2021 Instructor Report With Comments
BIOMEDE 599-009: Spec Topics
Deepak Nagrath

16 out of 35 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
This course advanced my understanding of the subject matter.(Q1631)	11	4	1	0	0	0	4.8	4.5	4.6
My interest in the subject has increased because of this course.(Q1632)	7	5	2	1	1	0	4.3	4.2	4.5
I knew what was expected of me in this course.(Q1633)	8	5	1	0	2	0	4.5	4.5	4.5
I had a strong desire to take this course.(Q4)	5	9	1	0	1	0	4.2	4.0	4.5
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	2	4	10	0	0	0	3.3	3.0	3.0

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
Deepak Nagrath seemed well prepared for class meetings.(Q230)	13	2	1	0	0	0	4.9	4.8	4.8
Deepak Nagrath explained material clearly.(Q199)	10	4	1	1	0	0	4.7	4.7	4.7
Deepak Nagrath treated students with respect.(Q217)	15	1	0	0	0	0	5.0	4.9	4.8

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median
Overall, this was an excellent course. (Q1)	8	5	3	0	0	0	4.5

Responses to questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Overall, Deepak Nagrath was an excellent teacher. (Q2)	9	3	3	1	0	0	4.6
Deepak Nagrath acknowledged all questions insofar as possible. (Q216)	10	6	0	0	0	0	4.7

The medians are calculated from Fall 2021 data. University-wide medians are based on all UM classes in which an item was used. The school/college medians in this report are based on classes that are graduate level with enrollment of 16 to 74 in College of Engineering.

University of Michigan
Fall 2022 Instructor Report
BIOMEDE 599-009: Spec Topics
Deepak Nagrath

9 out of 21 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
This course advanced my understanding of the subject matter. (Q1631)	2	7	0	0	0	0	4.1	4.5	4.7
My interest in the subject has increased because of this course. (Q1632)	1	7	1	0	0	0	4.0	4.2	4.5
I knew what was expected of me in this course.(Q1633)	2	7	0	0	0	0	4.1	4.6	4.6
I had a strong desire to take this course.(Q4)	1	8	0	0	0	0	4.1	4.0	4.5
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	2	3	4	0	0	0	3.7	3.0	3.0

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
Deepak Nagrath seemed well prepared for class meetings.(Q230)	7	2	0	0	0	0	4.9	4.8	4.8
Deepak Nagrath explained material clearly.(Q199)	6	3	0	0	0	0	4.8	4.7	4.7
Deepak Nagrath treated students with respect.(Q217)	7	2	0	0	0	0	4.9	4.8	4.9

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median
Overall, this was an excellent course. (Q1)	5	4	0	0	0	0	4.6

Responses to questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Overall, Deepak Nagrath was an excellent teacher. (Q2)	6	3	0	0	0	0	4.8
Deepak Nagrath acknowledged all questions insofar as possible. (Q216)	7	2	0	0	0	0	4.9

The medians are calculated from Fall 2022 data. University-wide medians are based on all UM classes in which an item was used. The school/college medians in this report are based on classes that are graduate level with enrollment of 16 to 74 in College of Engineering.

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
- ☑ New Course
- ☐ Modification of Existing Course
- ☐ Deletion of Existing Course

Date of Submission: 2024-04-26
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			REQUESTED LISTING		
☑	Dept (Home): Subject: Catalog:			Dept (Home): Computer Science and Engineering Subject: CSE Catalog: 577		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☑	Course Title (full title)			Course Title (full title) Formal Verification of Hardware and Software Systems		
☑	Abbreviated Title (20 char)			Abbreviated Title (20 char) Formal Verif HW & SW		

☑	Course Description (Please limit to 80 words and attach separate sheet if necessary) Scalable formal automated reasoning for checking the compliance of a state transition system with its safety requirements. Focus is on application to discrete finite-state systems that model hardware and software as well as infinite-state systems that model distributed protocols. Topics include propositional satisfiability (SAT) and SAT modulo theories (SMT) solving, predicate and data abstraction, and minimal unsatisfiable subset (MUS) extraction. Includes hands-on use of state-of-the-art formal verification tools.
☑	Full Term Credit Hours Undergraduate Min: 4 — Graduate Min: 4 Undergraduate Max: 4 — Graduate Max: 4 Half Term Credit Hours Undergraduate Min: — Graduate Min: Undergraduate Max: — Graduate Max:
☑	Course Credit Type Undergraduate Student, Rackham Graduate Student, Non-Rackham Graduate Student
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits: ☐ Course is Y graded ☐ Can be taken more than once in the same term

Subject: Catalog:

☑	Grading Basis ☑ Graded (A – E) ☐ Credit/No Credit ☐ Satisfactory/Unsatisfactory ☐ Pass/Fail ☐ Business Administration Grading ☐ Not for Credit ☐ Not for Degree Credit ☐ Degree Credit Only Add Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent Drop Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☑	Enforced Prerequisite (254 char) Minimum grade requirement:	Enforced Prerequisite (254 char) EECS 270 or 281 or 376; (C or better, No OP/F) or Graduate Standing Minimum grade requirement: C
☐	Credit Exclusions	Credit Exclusions

	Course Components	Graded Component	Terms Typically Offered
☑	☑ Lecture	☑	☑ Fall
	☐ Seminar	☐	☑ Winter
	☐ Recitation	☐	☐ Spring
	☐ Lab	☐	☐ Summer
	☑ Discussion	☐	☐ Spring/Summer
	☐ Independent Study	☐	

Cognizant Faculty Member Name: Karem Sakallah — Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas Email: vyas@umich.edu Phone: 734-647-1754

CoE Curriculum Committee Representative:	[signature]	Print: Amir Kamil	Date: 5/06/24
CoE Curriculum Committee Chair:		Print:	Date:
Home Department Chair:	[signature]	Print: Emily Provost	Date: 5/6/24
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:	Requested:
Course Description	Course Description Scalable formal automated reasoning for checking the compliance of a state transition system with its safety requirements. Focus is on application to discrete finite-state systems that model hardware and software as well as infinite-state systems that model distributed protocols. Topics include propositional satisfiability (SAT) and SAT modulo theories (SMT) solving, predicate and data abstraction, and minimal unsatisfiable subset (MUS) extraction. Includes hands-on use of state-of-the-art formal verification tools.
Class Length	Class Length Full term
Contact hours (lecture):	Contact hours (lecture): 3
Contact hours (recitation)	Contact hours (recitation) 1
Contact hours (lab)	Contact hours (lab)

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:
Depth course for CSE PhD, technical elective for CSE MS(E), and eULCS elective for UG CS-LSA and CS-Eng

Special resources of facilities required for this course:

Supporting statement:
This course covers the latest advances in automated reasoning algorithms and their application to verify safety properties of complex hardware, software, and distributed protocols. The use of fast scalable reasoning engines, based on Boolean Satisfiability (SAT) and Satisfiability Modulo Theories (SMT), has become routine in industrial settings and familiarity with their theory and application is increasingly valued as a critical skill in the high-tech industry.

The emphasis of the proposed course is on developing an appreciation of the power, as well as the limitations, of automated reasoning and on its universal applicability with suitable encodings to a wide range of domains. Specifically, a major theme of the course is to find common threads that tie together the seemingly disparate methods used in hardware and software verification. The course teaches students how to encode software programs and hardware circuits as transition systems, and how to develop suitable abstractions for checking control-centric properties on these systems.

Past offerings:
Fall 2020: 14 students

Winter 2023: 27 students
Winter 2024: 26 students

EECS 598-002 Syllabus

This course explores the latest advances in automated proof methods for checking whether certain properties hold under all possible executions of a complex hardware or software system. Specifically, we focus on the class of “control-centric” properties, namely those properties that are weakly dependent on the data state of the system. Examples of such properties include, among others, the equivalence between different implementations of an instruction set architecture (hardware), correct usage of an Application Programming Interface (software), and the safety of parameterized systems (distributed protocols).

The key to the scalable verification of such properties is a closed-loop CounterExample-Guided Abstraction Refinement (CEGAR) framework that involves:

1. A suitable state transition system encoding of the software or hardware being checked and the property they are expected to satisfy;
2. Structural abstraction of irrelevant data state that has nothing or little to do with the property;
3. Full unbounded reachability analysis of the abstract state space using efficient incremental induction algorithms;
4. Concretization of any resulting abstract counterexamples to determine their feasibility;
5. Automatic refinement of any spurious counterexamples that bring back only those relevant data constraints needed to provably establish that the property holds or to demonstrate a true violation.

The automated reasoning engines that make this possible are the modern Conflict-Driven Clause-Learning (CDCL) Boolean Satisfiability (SAT) solvers, the SAT modulo Theory (SMT) solvers, and the Minimal Unsatisfiable Subset (MUS) extractors.

1 Course Grading

- 25%: Frequent online quizzes that test basic understanding of lecture material.
- 35%: Seven to eight homework assignments that provide hands-on experience with software tools that perform a variety of verification tasks.
- 40%: Online comprehensive final exam.

Ph.D. students may opt for a semester-long research project in lieu of the final exam.

2 Course Schedule

Session	Topic	Lecture Files
1	Once Over Lightly	Course Overview
2	Binary Decision Diagrams (BDDs)	Basics
3		ite Algorithm
4	Symbolic Model Checking	Symbolic Reachability
5		Symbolic Model Checking
6	Boolean Satisfiability (SAT)	Basics
7		Conflict-Driven Clause Learning
8	Bounded Model Checking	Combinational Model Checking
9		Sequential BMC
10	SAT-Based Incremental Induction	Basics
11		IC3 Algorithm
12	Unounded Model Checking	AVR Verifier
13		Predicate Abstraction
14	Abstraction	Data Abstraction
15		EUF Congruence Closure
16		Basics
17	Satisfiability Modulo Theories (SMT)	SMT-LIB
18		Syntactic Equality Abstraction
19	Explanations of Unsatisfiability	MUS Extraction: MARCO
20		Group Theory Basics
21	Symmetry and Satisfiability	Efficient Symmetry Detection: Saucy
22		Symmetry Breaking
23		Protocols: IVy to I4
24	Model Checking of Infinite-State Systems	Protocols: I4 to IC3PO
25		Protocols: Reachable States
26	Course Review	

3 Reading List

This selection of articles constitutes the reading list for EECS 598-002. They cover seminal papers in the field of formal methods as well as papers on recent advances in automated verification of hardware and software systems. A subset of these papers is **required reading** and are highlighted in boldface.

3.1 Binary Decision Diagrams (BDDs): Symbolic Model Checking

- [Bry86] **Randal E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986**

- [Som99] **Fabio Somenzi. Binary Decision Diagrams. In Manfred Broy and Ralf Steinbrüggen, editors, *Calculational System Design*, volume 173 of *NATO Science Series F: Computer and Systems Sciences*, pages 303–366. IOS Press, 1999**
- [BCM+90] **Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill, and L. J. Hwang. Symbolic Model Checking: 10^{20} States and Beyond. In *Proceedings. Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 428–439, 1990**
- [QS82] Jean-Pierre Queille and Joseph Sifakis. Specification and Verification of Concurrent Systems in CESAR. In *International Symposium on Programming*, pages 337–351, 1982
- [CES83] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic Verification of Finite State Concurrent Systems Using Temporal Logic Specifications: A Practical Approach. In *POPL*, pages 117–126, 1983
- [McM93] Kenneth L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, Norwell, MA, USA, 1993
- [CCGR00] Alessandro Cimatti, Edmund Clarke, Fausto Giunchiglia, and Marco Roveri. NUSMV: A New Symbolic Model Checker. *International Journal on Software Tools for Technology Transfer (STTT)*, 2:410–425, 2000

3.2 Boolean Satisfiability (SAT): Bounded Model Checking

- [MSS99] **Jõao Marques-Silva and Karem A. Sakallah. GRASP: A Search Algorithm for Propositional Satisfiability. *IEEE Transactions on Computers*, 48(5):506–521, May 1999**
- [MMZ+01] **Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an Efficient SAT Solver. In *DAC*, pages 530–535, 2001**
- [ES03] **Niklas Eén and Niklas Sörensson. An Extensible SAT-solver. In *International conference on theory and applications of satisfiability testing*, pages 502–518. Springer, 2003**
- [BCCZ99] **Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu. Symbolic Model Checking without BDDs. In *Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems*, TACAS '99, pages 193–207, London, UK, 1999. Springer-Verlag**
- [MS03] **Maher Mneimneh and Karem Sakallah. SAT-based Sequential Depth Computation. In *Asia and South Pacific Design Automation Conference (ASP-DAC 2003)*, pages 87–92, Kitakyushu, Japan, January 2003**
- [SMS11] **Karem A. Sakallah and Jõao Marques-Silva. Anatomy and Empirical Evaluation of Modern SAT Solvers. In *Bull. of Euro. Assoc. for Theor. Computer Science*, volume 103, pages 96–121, February 2011**

- [DP58] Martin Davis and Hilary Putnam. Feasible computational methods in the propositional calculus. Technical report, Rensselaer Polytechnic Institute, Research Division, 1958

- [DP60] M. Davis and H. Putnam. A Computing Procedure for Quantification Theory. *Journal of the Association for Computing Machinery*, vol. 7:201–215, 1960

- [MLL62] M.Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5:394–397, July 1962

- [MSLM09] Joao Marques-Silva, Ines Lynce, and Sharad Malik. Conflict-Driven Clause Learning SAT Solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 131–153. IOS Press, 2009

- [Tse83] Grigori S Tseitin. On the comdsiplexity of derivation in propositional calculus. In *Automation of reasoning*, pages 466–483. Springer, 1983

3.3 SAT-Based Incremental Induction: Unbounded Model Checking

- [BM07] **Aaron R. Bradley and Zohar Manna. Checking Safety by Inductive Generalization of Counterexamples to Induction. In *Formal Methods in Computer Aided Design (FMCAD'07)*, pages 173 –180, Nov. 2007**

- [Bra11] **Aaron R. Bradley. SAT-Based Model Checking without Unrolling. In *Proceedings of the 12th international conference on Verification, model checking, and abstract interpretation*, VMCAI'11, pages 70–87, Berlin, Heidelberg, 2011. Springer-Verlag**

- [EMB11] **Niklas Een, Alan Mishchenko, and Robert Brayton. Efficient Implementation of Property Directed Reachability. In *Formal Methods in Computer Aided Design (FMCAD'11)*, pages 125 – 134, Oct. 2011**

3.4 Abstraction

- [NO80] **Greg Nelson and Derek C Oppen. Fast decision procedures based on congruence closure. *Journal of the ACM (JACM)*, 27(2):356–364, 1980**

- [BMMR01] **Thomas Ball, Rupak Majumdar, Todd Millstein, and Sriram K Rajamani. Automatic Predicate Abstraction of C Programs. In *Proceedings of the ACM SIGPLAN 2001 Conference on Programming Language Design and Implementation*, volume 36 of *PLDI '01*, pages 203–213. ACM, 2001**

- [BPR01] **Thomas Ball, Andreas Podelski, and SriramK. Rajamani. Boolean and Cartesian Abstraction for Model Checking C Programs. In Tiziana Margaria and Wang Yi, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 2031 of *Lecture Notes in Computer Science*, pages 268–283. Springer Berlin Heidelberg, 2001**

- [GS19] **Aman Goel and Karem Sakallah. Model checking of verilog rtl using ic3 with syntax-guided abstraction. In Julia M. Badger and Kristin Yvonne Rozier, editors, *NASA Formal Methods*, pages 166–185, Cham, 2019. Springer International Publishing**

- [CC77] Patrick Cousot and Radhia Cousot. Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '77, pages 238–252, New York, NY, USA, 1977. ACM

- [CC92] Patrick Cousot and Radhia Cousot. Abstract Interpretation and Application to Logic Programs. *The Journal of Logic Programming*, 13(2):103–179, 1992

- [GS97] Susanne Graf and Hassen Saidi. Construction of Abstract State Graphs with PVS. In Orna Grumberg, editor, *Computer Aided Verification*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83. Springer Berlin Heidelberg, 1997

- [JKSC05] Himanshu Jain, D. Kroening, N. Sharygina, and E. Clarke. Word Level Predicate Abstraction and Refinement for Verifying RTL Verilog. In *Proceedings 42nd Design Automation Conference*, pages 445 – 450, june 2005

- [DKW08] Vijay D'Silva, Daniel Kroening, and Georg Weissenbacher. A Survey of Automated Techniques for Formal Software Verification. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 27(7):1165–1178, 2008

3.5 Satisfiability Modulo Theories (SMT)

- [BT18] **Clark Barrett and Cesare Tinelli. Satisfiability modulo theories. In *Handbook of Model Checking*, pages 305–343. Springer, 2018**

- [Sho78] Robert E Shostak. An algorithm for reasoning about equality. *Communications of the ACM*, 21(7):583–585, 1978

- [NO79] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1(2):245–257, 1979

3.6 Minimal Unsatisfiable Subsets (MUSes)

- [LPMMS16] **Mark H Liffiton, Alessandro Previti, Ammar Malik, and Joao Marques-Silva. Fast, flexible mus enumeration. *Constraints*, 21(2):223–250, 2016**

- [LS08] Mark H. Liffiton and Karem A. Sakallah. Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints. *Journal of Automated Reasoning*, 40(1):1–33, January 2008

3.7 Symmetry and Satisfiability

- [AMS03] **Fadi A. Aloul, Igor L. Markov, and Karem A. Sakallah. Shatter: Efficient Symmetry-Breaking for Boolean Satisfiability. In *Proc. 40th IEEE/ACM Design Automation Conference (DAC)*, pages 836–839, Anaheim, California, June 2003**

- [Sak21] **Karem A. Sakallah. Symmetry and Satisfiability. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability, 2nd Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, chapter 13, pages 509–570. IOS Press, 2021**

- [CGLR96] James Crawford, Matthew Ginsberg, Eugene Luks, and Amitabha Roy. Symmetry-breaking predicates for search problems. *KR*, 96(1996):148–159, 1996

- [DLSM04] Paul T. Darga, Mark H. Liffiton, Karem A. Sakallah, and Igor L. Markov. Exploiting Structure in Symmetry Detection for CNF. In *Proc. 41st IEEE/ACM Design Automation Conference (DAC)*, pages 530–534, San Diego, California, June 2004

- [DSM08] Paul T. Darga, Karem A. Sakallah, and Igor L. Markov. Faster Symmetry Discovery using Sparsity of Symmetries. In *Proc. 45th IEEE/ACM Design Automation Conference (DAC)*, pages 149–154, Anaheim, California, June 2008

- [KSM10] Hadi Katebi, Karem A. Sakallah, and Igor L. Markov. Symmetry and Satisfiability: An Update. In O. Strichman and S. Szeider, editors, *Thirteenth International Conference on Theory and Applications of Satisfiability Testing (SAT)*, volume LNCS 6175, pages 113–127, Edinburgh, July 2010

3.8 Verification of Unbounded Distributed Protocols

- [GS21a] **Aman Goel and Karem Sakallah. On symmetry and quantification: A new approach to verify distributed protocols. In Aaron Dutle, Mariano M. Moscato, Laura Titolo, César A. Muñoz, and Ivan Perez, editors, *NASA Formal Methods*, pages 131–150, Cham, 2021. Springer International Publishing**

- [FGS23] **Katalin Fazekas, Aman Goel, and Karem A. Sakallah. SAT-Based Quantified Symmetric Minimization of the Reachable States of Distributed Protocols. In *Formal Methods in Computer-Aided Design (FMCAD 2023)*, pages 152–161, Ames, Iowa, October 2023**

- [PMP+16] Oded Padon, Kenneth L McMillan, Aurojit Panda, Mooly Sagiv, and Sharon Shoham. Ivy: safety verification by interactive generalization. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 614–630, 2016

- [MGJ+19b] Haojun Ma, Aman Goel, Jean-Baptiste Jeannin, Manos Kapritsos, Baris Kasikci, and Karem A. Sakallah. Towards Automatic Inference of Inductive Invariants. In *The 17th Workshop on Hot Topics in Operating Systems (HotOS XVII)*, pages 30–36, Bertinoro, Italy, May 2019

- [MGJ+19a] Haojun Ma, Aman Goel, Jean-Baptiste Jeannin, Manos Kapritsos, Baris Kasikci, and Karem A. Sakallah. I4: Incremental Inference of Inductive Invariants forVerification of Distributed Protocols. In *The 27th ACM Symposium on Operating Systems Principles (SOSP 2019)*, pages 370–384, Huntsville, Ontario, Canada, October 2019

- [GS21b] Aman Goel and Karem A. Sakallah. Towards an Automatic Proof of Lamport's Paxos. In Ruzica Piskac and Michael W Whalen, editors, *Formal Methods in Computer-Aided Design (FMCAD 2021)*, pages 112–122, New Haven, Connecticut, October 2021

4 Software Tools

The course provides hands-on experience with several formal verification tools. These tools are available on CAEN and can be accessed by typing **module load eecs598-002/w24** at the Unix command prompt.

- RePyCUDD: Python wrapper for the CUDD Binary Decision Diagram package.
- PySAT: Python wrapper for several propositional satisfiability solvers.
- AVR: Formal bit- and word-level Verification package for transitions systems specified in the Verilog Hardware Description Language.
- PDR: Formal bit-level verifier for transition systems specified as AIG (And-Inverter-Graph) netlists.
- Z3: Microsoft's Satisfiability Modulo Theories Solver (also available via a web interface)
- MARCO: Minimal Unsatisfiable Subset (MUS) extractor.
- Shatter: Generator of *symmetry-breaking predicates*.

References

[AMS03] Fadi A. Aloul, Igor L. Markov, and Karem A. Sakallah. Shatter: Efficient Symmetry-Breaking for Boolean Satisfiability. In *Proc. 40th IEEE/ACM Design Automation Conference (DAC)*, pages 836–839, Anaheim, California, June 2003.

[BCCZ99] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu. Symbolic Model Checking without BDDs. In *Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems*, TACAS '99, pages 193–207, London, UK, 1999. Springer-Verlag.

[BCM+90] Jerry R. Burch, Edmund M. Clarke, Kenneth L. McMillan, David L. Dill, and L. J. Hwang. Symbolic Model Checking: 10^{20} States and Beyond. In *Proceedings. Fifth Annual IEEE Symposium on Logic in Computer Science*, pages 428–439, 1990.

[BM07] Aaron R. Bradley and Zohar Manna. Checking Safety by Inductive Generalization of Counterexamples to Induction. In *Formal Methods in Computer Aided Design (FMCAD'07)*, pages 173 –180, Nov. 2007.

[BMMR01] Thomas Ball, Rupak Majumdar, Todd Millstein, and Sriram K Rajamani. Automatic Predicate Abstraction of C Programs. In *Proceedings of the ACM SIGPLAN 2001 Conference on Programming Language Design and Implementation*, volume 36 of *PLDI '01*, pages 203–213. ACM, 2001.

[BPR01] Thomas Ball, Andreas Podelski, and SriramK. Rajamani. Boolean and Cartesian Abstraction for Model Checking C Programs. In Tiziana Margaria and Wang Yi, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 2031 of *Lecture Notes in Computer Science*, pages 268–283. Springer Berlin Heidelberg, 2001.

[Bra11] Aaron R. Bradley. SAT-Based Model Checking without Unrolling. In *Proceedings of the 12th international conference on Verification, model checking, and abstract interpretation*, VMCAI'11, pages 70–87, Berlin, Heidelberg, 2011. Springer-Verlag.

[Bry86] Randal E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986.

[BT18] Clark Barrett and Cesare Tinelli. Satisfiability modulo theories. In *Handbook of Model Checking*, pages 305–343. Springer, 2018.

[CC77] Patrick Cousot and Radhia Cousot. Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '77, pages 238–252, New York, NY, USA, 1977. ACM.

[CC92] Patrick Cousot and Radhia Cousot. Abstract Interpretation and Application to Logic Programs. *The Journal of Logic Programming*, 13(2):103–179, 1992.

[CCGR00] Alessandro Cimatti, Edmund Clarke, Fausto Giunchiglia, and Marco Roveri. NUSMV: A New Symbolic Model Checker. *International Journal on Software Tools for Technology Transfer (STTT)*, 2:410–425, 2000.

[CES83] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic Verification of Finite State Concurrent Systems Using Temporal Logic Specifications: A Practical Approach. In *POPL*, pages 117–126, 1983.

[CGLR96] James Crawford, Matthew Ginsberg, Eugene Luks, and Amitabha Roy. Symmetry-breaking predicates for search problems. *KR*, 96(1996):148–159, 1996.

[DKW08] Vijay D'Silva, Daniel Kroening, and Georg Weissenbacher. A Survey of Automated Techniques for Formal Software Verification. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 27(7):1165–1178, 2008.

[DLSM04] Paul T. Darga, Mark H. Liffiton, Karem A. Sakallah, and Igor L. Markov. Exploiting Structure in Symmetry Detection for CNF. In *Proc. 41st IEEE/ACM Design Automation Conference (DAC)*, pages 530–534, San Diego, California, June 2004.

[DP58] Martin Davis and Hilary Putnam. Feasible computational methods in the propositional calculus. Technical report, Rensselaer Polytechnic Institute, Research Division, 1958.

[DP60] M. Davis and H. Putnam. A Computing Procedure for Quantification Theory. *Journal of the Association for Computing Machinery*, vol. 7:201–215, 1960.

[DSM08] Paul T. Darga, Karem A. Sakallah, and Igor L. Markov. Faster Symmetry Discovery using Sparsity of Symmetries. In *Proc. 45th IEEE/ACM Design Automation Conference (DAC)*, pages 149–154, Anaheim, California, June 2008.

[EMB11] Niklas Een, Alan Mishchenko, and Robert Brayton. Efficient Implementation of Property Directed Reachability. In *Formal Methods in Computer Aided Design (FMCAD'11)*, pages 125 – 134, Oct. 2011.

[ES03] Niklas Eén and Niklas Sörensson. An Extensible SAT-solver. In *International conference on theory and applications of satisfiability testing*, pages 502–518. Springer, 2003.

[FGS23] Katalin Fazekas, Aman Goel, and Karem A. Sakallah. SAT-Based Quantified Symmetric Minimization of the Reachable States of Distributed Protocols. In *Formal Methods in Computer-Aided Design (FMCAD 2023)*, pages 152–161, Ames, Iowa, October 2023.

[GS97] Susanne Graf and Hassen Saidi. Construction of Abstract State Graphs with PVS. In Orna Grumberg, editor, *Computer Aided Verification*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83. Springer Berlin Heidelberg, 1997.

[GS19] Aman Goel and Karem Sakallah. Model checking of verilog rtl using ic3 with syntax-guided abstraction. In Julia M. Badger and Kristin Yvonne Rozier, editors, *NASA Formal Methods*, pages 166–185, Cham, 2019. Springer International Publishing.

[GS21a] Aman Goel and Karem Sakallah. On symmetry and quantification: A new approach to verify distributed protocols. In Aaron Dutle, Mariano M. Moscato, Laura Titolo, César A. Muñoz, and Ivan Perez, editors, *NASA Formal Methods*, pages 131–150, Cham, 2021. Springer International Publishing.

[GS21b] Aman Goel and Karem A. Sakallah. Towards an Automatic Proof of Lamport's Paxos. In Ruzica Piskac and Michael W Whalen, editors, *Formal Methods in Computer-Aided Design (FMCAD 2021)*, pages 112–122, New Haven, Connecticut, October 2021.

[JKSC05] Himanshu Jain, D. Kroening, N. Sharygina, and E. Clarke. Word Level Predicate Abstraction and Refinement for Verifying RTL Verilog. In *Proceedings 42nd Design Automation Conference*, pages 445 – 450, june 2005.

[KSM10] Hadi Katebi, Karem A. Sakallah, and Igor L. Markov. Symmetry and Satisfiability: An Update. In O. Strichman and S. Szeider, editors, *Thirteenth International Conference on Theory and Applications of Satisfiability Testing (SAT)*, volume LNCS 6175, pages 113–127, Edinburgh, July 2010.

[LPMMS16] Mark H Liffiton, Alessandro Previti, Ammar Malik, and Joao Marques-Silva. Fast, flexible mus enumeration. *Constraints*, 21(2):223–250, 2016.

[LS08] Mark H. Liffiton and Karem A. Sakallah. Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints. *Journal of Automated Reasoning*, 40(1):1–33, January 2008.

[McM93] Kenneth L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[MGJ+19a] Haojun Ma, Aman Goel, Jean-Baptiste Jeannin, Manos Kapritsos, Baris Kasikci, and Karem A. Sakallah. I4: Incremental Inference of Inductive Invariants forVerification of Distributed Protocols. In *The 27th ACM Symposium on Operating Systems Principles (SOSP 2019)*, pages 370–384, Huntsville, Ontario, Canada, October 2019.

[MGJ+19b] Haojun Ma, Aman Goel, Jean-Baptiste Jeannin, Manos Kapritsos, Baris Kasikci, and Karem A. Sakallah. Towards Automatic Inference of Inductive Invariants. In *The 17th Workshop on Hot Topics in Operating Systems (HotOS XVII)*, pages 30–36, Bertinoro, Italy, May 2019.

[MLL62] M.Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5:394–397, July 1962.

[MMZ+01] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an Efficient SAT Solver. In *DAC*, pages 530–535, 2001.

[MS03] Maher Mneimneh and Karem Sakallah. SAT-based Sequential Depth Computation. In *Asia and South Pacific Design Automation Conference (ASP-DAC 2003)*, pages 87–92, Kitakyushu, Japan, January 2003.

[MSLM09] Joao Marques-Silva, Ines Lynce, and Sharad Malik. Conflict-Driven Clause Learning SAT Solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 131–153. IOS Press, 2009.

[MSS99] Jõao Marques-Silva and Karem A. Sakallah. GRASP: A Search Algorithm for Propositional Satisfiability. *IEEE Transactions on Computers*, 48(5):506–521, May 1999.

[NO79] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1(2):245–257, 1979.

[NO80] Greg Nelson and Derek C Oppen. Fast decision procedures based on congruence closure. *Journal of the ACM (JACM)*, 27(2):356–364, 1980.

[PMP+16] Oded Padon, Kenneth L McMillan, Aurojit Panda, Mooly Sagiv, and Sharon Shoham. Ivy: safety verification by interactive generalization. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 614–630, 2016.

[QS82] Jean-Pierre Queille and Joseph Sifakis. Specification and Verification of Concurrent Systems in CESAR. In *International Symposium on Programming*, pages 337–351, 1982.

[Sak21] Karem A. Sakallah. Symmetry and Satisfiability. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability, 2nd Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, chapter 13, pages 509–570. IOS Press, 2021.

[Sho78] Robert E Shostak. An algorithm for reasoning about equality. *Communications of the ACM*, 21(7):583–585, 1978.

[SMS11] Karem A. Sakallah and Jõao Marques-Silva. Anatomy and Empirical Evaluation of Modern SAT Solvers. In *Bull. of Euro. Assoc. for Theor. Computer Science*, volume 103, pages 96–121, February 2011.

[Som99] Fabio Somenzi. Binary Decision Diagrams. In Manfred Broy and Ralf Steinbrüggen, editors, *Calculational System Design*, volume 173 of *NATO Science Series F: Computer and Systems Sciences*, pages 303–366. IOS Press, 1999.

[Tse83] Grigori S Tseitin. On the comdsiplexity of derivation in propositional calculus. In *Automation of reasoning*, pages 466–483. Springer, 1983.

University of Michigan
Fall 2020 Instructor Report With Comments
EECS 598-008: Special Topics
Karem Sakallah

4 out of 14 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
This course advanced my understanding of the subject matter. (Q1631)	3	1	0	0	0	0	4.8	4.6	4.5
My interest in the subject has increased because of this course. (Q1632)	3	1	0	0	0	0	4.8	4.2	4.2
I knew what was expected of me in this course.(Q1633)	2	1	1	0	0	0	4.5	4.5	4.4
Overall, this was an excellent course.(Q1)	3	1	0	0	0	0	4.8	4.4	4.3
I had a strong desire to take this course.(Q4)	3	1	0	0	0	0	4.8	4.1	4.1
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	0	1	3	0	0	0	3.2	2.9	2.8
How did you participate in this course? (SA=Attended most synchronously, A=Attended most asynchronously, N=Attended most in person, D=Attended some in person and some online) (Q1854)	4	0	0	0	0	0	5.0	4.7	4.5

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
Overall, Karem Sakallah was an excellent teacher.(Q2)	3	1	0	0	0	0	4.8	4.7	4.8
Karem Sakallah seemed well prepared for class meetings.(Q230)	3	1	0	0	0	0	4.8	4.8	4.9
Karem Sakallah explained material clearly.(Q199)	3	0	0	0	0	0	5.0	4.7	4.8
Karem Sakallah treated students with respect.(Q217)	4	0	0	0	0	0	5.0	4.9	4.9

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median	University-Wide Median
Prerequisites provided adequate preparation for this course. (Q61)	3	0	0	0	0	1	5.0	4.5
The textbook made a valuable contribution to the course. (Q64)	3	0	0	0	0	1	5.0	3.9
I felt included and valued when working with other students. (Q253)	3	1	0	0	0	0	4.8	4.7
I felt comfortable asking questions in class. (Q521)	3	1	0	0	0	0	4.8	4.4
I developed confidence in my abilities as an engineer. (Q1769)	3	0	1	0	0	0	4.8	4.2
I developed the ability to solve real world engineering problems. (Q1770)	3	0	1	0	0	0	4.8	4.2

University of Michigan
Winter 2023 Instructor Report
EECS 598-002: Special Topics
Karem Sakallah

8 out of 27 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median	School/College Median	Univ-Wide Median
This course advanced my understanding of the subject matter. (Q1631)	3	5	0	0	0	0	4.3	4.4	4.5
My interest in the subject has increased because of this course. (Q1632)	3	3	2	0	0	0	4.2	4.1	4.2
I knew what was expected of me in this course.(Q1633)	3	4	1	0	0	0	4.3	4.3	4.6
I had a strong desire to take this course.(Q4)	2	5	1	0	0	0	4.1	4.0	4.1
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	1	3	4	0	0	0	3.5	2.8	3.0

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median	School/College Median	Univ-Wide Median
Karem Sakallah seemed well prepared for class meetings.(Q230)	3	5	0	0	0	0	4.3	4.7	4.8
Karem Sakallah explained material clearly.(Q199)	2	5	1	0	0	0	4.1	4.6	4.7
Karem Sakallah treated students with respect.(Q217)	6	2	0	0	0	0	4.8	4.8	4.8

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median
Overall, this was an excellent course. (Q1)	2	4	2	0	0	0	4.0
The textbook made a valuable contribution to the course. (Q64)	0	1	1	1	1	4	2.5
Prerequisites provided adequate preparation for this course. (Q61)	2	4	0	0	0	2	4.3
I felt comfortable asking questions in class. (Q521)	3	3	2	0	0	0	4.2
The discussion section was a valuable part of this course. (Q1771)	2	2	2	1	0	1	3.8
I developed confidence in my abilities as an engineer. (Q1769)	2	4	2	0	0	0	4.0
I developed the ability to solve real world engineering problems. (Q1770)	2	3	3	0	0	0	3.8
I felt included and valued when working with other students. (Q253)	3	2	0	0	0	3	4.7

Responses to questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Overall, Karem Sakallah was an excellent teacher. (Q2)	3	5	0	0	0	0	4.3

University of Michigan
Winter 2023 Instructor Report
EECS 598-021: Special Topics
Karem Sakallah

6 out of 27 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median	School/College Median	Univ-Wide Median
This course advanced my understanding of the subject matter. (Q1631)	1	4	1	0	0	0	4.0	4.4	4.5
My interest in the subject has increased because of this course. (Q1632)	1	3	2	0	0	0	3.8	4.1	4.2
I knew what was expected of me in this course.(Q1633)	2	3	1	0	0	0	4.2	4.3	4.6
I had a strong desire to take this course.(Q4)	1	4	1	0	0	0	4.0	4.0	4.1
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	1	2	3	0	0	0	3.5	2.8	3.0

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median	School/College Median	Univ-Wide Median
Karem Sakallah seemed well prepared for class meetings.(Q230)	1	5	0	0	0	0	4.1	4.7	4.8
Karem Sakallah explained material clearly.(Q199)	2	3	1	0	0	0	4.2	4.6	4.7
Karem Sakallah treated students with respect.(Q217)	5	1	0	0	0	0	4.9	4.8	4.8

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median
Overall, this was an excellent course. (Q1)	3	1	2	0	0	0	4.5

Responses to questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Overall, Karem Sakallah was an excellent teacher. (Q2)	2	4	0	0	0	0	4.3
Karem Sakallah thoroughly understood the subject matter. (Q772)	4	2	0	0	0	0	4.8
Karem Sakallah was sensitive/patient to the level of student comprehension. (Q773)	2	4	0	0	0	0	4.3
Overall, Karem Sakallah was effective. (Q776)	2	4	0	0	0	0	4.3
Karem Sakallah has good English skills. (Q378)	4	2	0	0	0	0	4.8
Karem Sakallah had regular office hours and was available at those hours. (Q770)	2	3	0	0	0	1	4.3

The medians are calculated from Winter 2023 data. University-wide medians are based on all UM classes in which an item was used. The school/college medians in this report are based on classes that are graduate level with enrollment of 16 to 74 in College of Engineering.

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
- ☐ New Course
- ☑ Modification of Existing Course
- ☐ Deletion of Existing Course

Date of Submission: 2024-03-25
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			REQUESTED LISTING		
☐	Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 110			Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 110		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☐	Course Title (full title) Discover Computer Science			Course Title (full title) Discover Computer Science		
☐	Abbreviated Title (20 char) Discover CS			Abbreviated Title (20 char) Discover CS		
☑	Course Description (Please limit to 80 words and attach separate sheet if necessary) Introduction to basic CS concepts (variables, conditionals, loops, functions) using an introductory programming language, such as Python. Students interact with researchers and computing professionals to learn about real-world, interdisciplinary applications of CS. Intended for students without prior programming experience to (optionally) take prior to EECS 183 or ENGR 101.					
☐	Full Term Credit Hours Undergraduate Min: 2 Undergraduate Max: 2	Graduate Min: Graduate Max:		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☐	Course Credit Type Undergraduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Elec Engin & Computer Sci   Catalog: 110

☐

Grading Basis
- ☑ Graded (A – E)
- ☐ Credit/No Credit
- ☐ Satisfactory/Unsatisfactory
- ☐ Pass/Fail
- ☐ Business Administration

Grading
- ☐ Not for Credit
- ☐ Not for Degree Credit
- ☐ Degree Credit Only

Add Consent
- ☐ Department Consent
- ☐ Instructor Consent
- ☑ No Consent

Drop Consent
- ☐ Department Consent
- ☐ Instructor Consent
- ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☑	Enforced Prerequisite (254 char) Minimum grade requirement:	Enforced Prerequisite (254 char) No credit in EECS 280 or 281. Minimum grade requirement:
☐	Credit Exclusions	Credit Exclusions

☐

Course Components	Graded Component
☑ Lecture	☑
☐ Seminar	☐
☐ Recitation	☐
☐ Lab	☐
☐ Discussion	☐
☐ Independent Study	☐

Terms Typically Offered
- ☑ Fall
- ☑ Winter
- ☐ Spring
- ☐ Summer
- ☐ Spring/Summer

Cognizant Faculty Member Name: Laura Burdick   Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas   Email: vyas@umich.edu   Phone: 647-1754

CoE Curriculum Committee Representative:	[signature]	Print: Amir Kamil	Date: 4/15/24
CoE Curriculum Committee Chair:		Print:	Date:
Home Department Chair:	[signature]	Print: Andrew DeOrio	Date: 4/16/2024
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:	**Requested:**
Course Description Introduction to basic CS concepts (variables, conditionals, loops, functions) using an introductory programming language, such a Python. Students interact with researchers and computing professionals to learn about real-world, interdisciplinary applications of CS. Intended for students without prior programming experience to (optionally take prior to EECS 183 or ENGR 101.	Course Description Introduction to basic CS concepts (variables, conditionals, loops, functions) using an introductory programming language, such as Python. Students interact with researchers and computing professionals to learn about real-world, interdisciplinary applications of CS. Intended for students without prior programming experience to (optionally) take prior to EECS 183 or ENGR 101.
Class Length Full term	Class Length Full term
Contact hours (lecture): 2	Contact hours (lecture): 2
Contact hours (recitation)	Contact hours (recitation)
Contact hours (lab)	Contact hours (lab)

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:

Special resources of facilities required for this course:

Supporting statement:
We have had issues with advanced students taking the course for "easy" general credits, as well as to pad their major GPA. The enforced prerequisite will prevent them from doing so.

We are also fixing some typos in the course description.

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ CHECK APPROPRIATE BOXES FOR ALL CHANGES

Action Requested

☐ New Course
☑ Modification of Existing Course
☐ Deletion of Existing Course

Date of Submission: 2024-03-25
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			REQUESTED LISTING		
☐	Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 183			Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 183		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☐	Course Title (full title) Elementary Programming Concepts			Course Title (full title) Elementary Programming Concepts		
☐	Abbreviated Title (20 char) Elem Prog Concepts			Abbreviated Title (20 char) Elem Prog Concepts		
☐	Course Description (Please limit to 80 words and attach separate sheet if necessary) Fundamental concepts and skills of programming in a high level language. Flow of control: selection, iteration, subprograms. Data structures: strings, arrays, records, lists, tables. Algorithms using selection and iteration (decision making, finding maxima/minima, searching, sorting, simulation, etc.). Good program design, structure, and style are emphasized. Testing and debugging. Not intended for Engineering students (who should take ENGR 101), nor for CS majors in LSA who qualify for EECS 280.					
☐	Full Term Credit Hours Undergraduate Min: 4 Undergraduate Max: 4	Graduate Min: Graduate Max:		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☐	Course Credit Type Undergraduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Elec Engin & Computer Sci   Catalog: 183

☐

Grading Basis
- ☑ Graded (A – E)
- ☐ Credit/No Credit
- ☐ Satisfactory/Unsatisfactory
- ☐ Pass/Fail
- ☐ Business Administration

Grading
- ☐ Not for Credit
- ☐ Not for Degree Credit
- ☐ Degree Credit Only

Add Consent
- ☐ Department Consent
- ☐ Instructor Consent
- ☑ No Consent

Drop Consent
- ☐ Department Consent
- ☐ Instructor Consent
- ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☑	Enforced Prerequisite (254 char) Minimum grade requirement:	Enforced Prerequisite (254 char) No credit in EECS 280 or 281. Minimum grade requirement:
☐	Credit Exclusions Credit for only one: EECS 180, EECS 183, ENGR 101, ENGR 151	Credit Exclusions Credit for only one: EECS 180, EECS 183, ENGR 101, ENGR 151

☐

Course Components	Graded Component
☑ Lecture	☑
☐ Seminar	☐
☐ Recitation	☐
☑ Lab	☐
☐ Discussion	☐
☐ Independent Study	☐

Terms Typically Offered
- ☑ Fall
- ☑ Winter
- ☐ Spring
- ☐ Summer
- ☐ Spring/Summer

Cognizant Faculty Member Name: William Arthur   Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas   Email: vyas@umich.edu   Phone: 647-1754

CoE Curriculum Committee Representative: [signature]   Print: Amir Kamil   Date: 4/15/24

CoE Curriculum Committee Chair: Print: Date:

Home Department Chair: [signature]   Print: Andrew DeOrio   Date: 4/16/2024

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:

Course Description
Fundamental concepts and skills of programming in a high level language. Flow of control: selection, iteration, subprograms. Data structures: strings, arrays, records, lists, tables. Algorithms using selection and iteration (decision making, finding maxima/minima, searching, sorting, simulation, etc.). Good program design, structure, and style are emphasized. Testing and debugging. Not intended for Engineering students (who should take ENGR 101), nor for CS majors in LSA who qualify for EECS 280.

Class Length
Full term

Contact hours (lecture):
3

Contact hours (recitation)

Contact hours (lab)
2

Requested:

Course Description
Fundamental concepts and skills of programming in a high level language. Flow of control: selection, iteration, subprograms. Data structures: strings, arrays, records, lists, tables. Algorithms using selection and iteration (decision making, finding maxima/minima, searching, sorting, simulation, etc.). Good program design, structure, and style are emphasized. Testing and debugging. Not intended for Engineering students (who should take ENGR 101), nor for CS majors in LSA who qualify for EECS 280.

Class Length
Full term

Contact hours (lecture):
3

Contact hours (recitation)

Contact hours (lab)
2

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:

Special resources of facilities required for this course:

Supporting statement:
We have had issues with advanced students taking the course for "easy" general credits (after testing into EECS 280 or 281), as well as to pad their major GPA. The enforced prerequisite will prevent them from doing so.

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
- ☐ New Course
- ☑ Modification of Existing Course
- ☐ Deletion of Existing Course

Date of Submission: 2024-02-16
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			**REQUESTED LISTING**		
☐	Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 477			Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 477		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☐	Course Title (full title) Introduction to Algorithms			Course Title (full title) Introduction to Algorithms		
☐	Abbreviated Title (20 char) Intro to Algorithms			Abbreviated Title (20 char) Intro to Algorithms		
☐	Course Description (Please limit to 80 words and attach separate sheet if necessary) Fundamental techniques for designing efficient algorithms and basic mathematical methods for analyzing their performance. Paradigms for algorithm design: divide-and-conquer, greedy methods, graph search techniques, dynamic programming. Design of efficient data structures and analysis of the running time and space requirements of algorithms in the worst and average cases.					
☐	Full Term Credit Hours Undergraduate Min: 4 Graduate Min: 4 Undergraduate Max: 4 Graduate Max: 4			Half Term Credit Hours Undergraduate Min: Graduate Min: Undergraduate Max: Graduate Max:		
☐	Course Credit Type Undergraduate Student, Rackham Graduate Student, Non-Rackham Graduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Elec Engin & Computer Sci   Catalog: 477

☐	Grading Basis ☑ Graded (A – E) ☐ Credit/No Credit ☐ Satisfactory/Unsatisfactory ☐ Pass/Fail ☐ Business Administration Grading ☐ Not for Credit ☐ Not for Degree Credit ☐ Degree Credit Only	Add Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent	Drop Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☐	Enforced Prerequisite (254 char) EECS 281 and EECS 376; (C or better, No OP/F). Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C	Enforced Prerequisite (254 char) EECS 281 and EECS 376; (C or better, No OP/F). Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C
☑	Credit Exclusions No credit to a student who has taken EECS 586.	Credit Exclusions No credit to a student who has taken CSE 586.

	Course Components	Graded Component	Terms Typically Offered
☐	☑ Lecture	☑	☑ Fall
	☐ Seminar	☐	☑ Winter
	☐ Recitation	☐	☐ Spring
	☐ Lab	☐	☐ Summer
	☑ Discussion	☐	☐ Spring/Summer
	☐ Independent Study	☐	

Cognizant Faculty Member Name: Seth Pettie   Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas   Email: vyas@umich.edu   Phone: 734-647-1754

CoE Curriculum Committee Representative: [signature]   Print: Amir Kamil   Date: 4/05/24

CoE Curriculum Committee Chair: Print: Date:

Home Department Chair: [signature]   Print: Andrew DeOrio   Date: 04/05/2024

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:

Course Description
Fundamental techniques for designing efficient algorithms and basic mathematical methods for analyzing their performance. Paradigms for algorithm design: divide-and-conquer, greedy methods, graph search techniques, dynamic programming. Design of efficient data structures and analysis of the running time and space requirements of algorithms in the worst and average cases.

Class Length
Full term

Contact hours (lecture):
3

Contact hours (recitation)
1

Contact hours (lab)

Requested:

Course Description
Fundamental techniques for designing efficient algorithms and basic mathematical methods for analyzing their performance. Paradigms for algorithm design: divide-and-conquer, greedy methods, graph search techniques, dynamic programming. Design of efficient data structures and analysis of the running time and space requirements of algorithms in the worst and average cases.

Class Length
Full term

Contact hours (lecture):
3

Contact hours (recitation)
1

Contact hours (lab)

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:
Tech Elective

Special resources of facilities required for this course:

Supporting statement:
We are modifying the credit exclusion to reflect the new subject code for CSE 586.

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

Course Approval Request Form

Office of the Registrar, University of Michigan

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
- ☑ New Course
- ☐ Modification of Existing Course
- ☐ Deletion of Existing Course

Date of Submission: 2024-04-26
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			**REQUESTED LISTING**		
☑	Dept (Home): Subject: Catalog:			Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 479		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☑	Course Title (full title)			Course Title (full title) Introduction to Quantum Computing		
☑	Abbreviated Title (20 char)			Abbreviated Title (20 char) Quantum Computing		
☑	Course Description (Please limit to 80 words and attach separate sheet if necessary) A practical approach towards exploring how each layer of the computing stack is impacted by quantum computing. Quantum logic design using classical oracles, phase kickback, and entanglement. Quantum algorithms including Deutsch-Jozsa, Grover's, and Shor's. Error correction schemes including Shor and Steane codes. Building fault-tolerant architectures. Several programming assignments.					
☑	Full Term Credit Hours Undergraduate Min: 4 Undergraduate Max: 4	Graduate Min: 4 Graduate Max: 4		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☑	Course Credit Type Undergraduate Student, Rackham Graduate Student, Non-Rackham Graduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Catalog:

☑ Grading Basis
- ☑ Graded (A – E)
- ☐ Credit/No Credit
- ☐ Satisfactory/Unsatisfactory
- ☐ Pass/Fail
- ☐ Business Administration

Grading
- ☐ Not for Credit
- ☐ Not for Degree Credit
- ☐ Degree Credit Only

Add Consent
- ☐ Department Consent
- ☐ Instructor Consent
- ☑ No Consent

Drop Consent
- ☐ Department Consent
- ☐ Instructor Consent
- ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☑	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char) MATH 214 or other linear algebra introduction
☑	Enforced Prerequisite (254 char) Minimum grade requirement:	Enforced Prerequisite (254 char) EECS 280 and 370; (C or better, No OP/F) or Graduate Standing in CSE. Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C
☐	Credit Exclusions	Credit Exclusions

☑

Course Components	Graded Component
☑ Lecture	☑
☐ Seminar	☐
☐ Recitation	☐
☑ Lab	☐
☐ Discussion	☐
☐ Independent Study	☐

Terms Typically Offered
- ☑ Fall
- ☑ Winter
- ☐ Spring
- ☐ Summer
- ☐ Spring/Summer

Cognizant Faculty Member Name: Jonathan Beaumont | Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas | Email: vyas@umich.edu | Phone: 734-647-1754

CoE Curriculum Committee Representative:	[signature]	Print: Amir Kamil	Date: 5/03/24
CoE Curriculum Committee Chair:		Print:	Date:
Home Department Chair:	[signature]	Print: Andrew DeOrio	Date: 5/3/24
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:
Cross-Listed Department Chair:		Print:	Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:	**Requested:**
Course Description	Course Description A practical approach towards exploring how each layer of the computing stack is impacted by quantum computing. Quantum logic design using classical oracles, phase kickback, and entanglement. Quantum algorithms including Deutsch-Jozsa, Grover's, and Shor's. Error correction schemes including Shor and Steane codes. Building fault-tolerant architectures. Several programming assignments.
Class Length	Class Length Full term
Contact hours (lecture):	Contact hours (lecture): 3
Contact hours (recitation)	Contact hours (recitation)
Contact hours (lab)	Contact hours (lab) 2

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:
ULCS elective course for Computer Science, UCLE elective course for Computer Engineering

Special resources of facilities required for this course:

Supporting statement:
No quantum computing courses are offered regularly in the CSE division. This course will add an opportunity for students to study this rich topic from a CS/CE perspective, making heavy use of the program's core requisites to dive deep into logic design, algorithm implementations and programming pragmatics. A follow-on course for graduate students and interested undergrads is currently offered as a 500-level special topics course. These two courses have been developed in tandem with the goal that students who take both will be prepared to engage in meaningful research in quantum architectures / programming.

Past offerings:
Fall 2022: 36 students
Winter 2024: 74 students

EECS 498-001: Quantum Computing for the Computer Scientist

Course Overview

Quantum computing, should current technical barriers be overcome, makes bold promises to revolutionize key applications including cryptography, machine learning, and computational physics. This course will explore the potential impact and limitations of this paradigm shift from a computer science perspective. Lectures will cover the bare physics and mathematics needed to investigate how each layer of the computing stack (logic, system architecture, algorithm, and application design) is impacted. Labs and programming assignments will provide students a hands-on approach towards writing quantum programs, simulating their execution, deploying them to real quantum hardware available on the cloud, and analyzing their performance.

Course Components

Lectures are offered via in-person and recorded formats. Attendance is not required.

Labs are held in-person and involve graded worksheets. Partnerships are optional. Attendance is not required.

Projects must be completed individually and submitted to the autograder.

Assessments will be administered in-person and during lecture time.

Office hours will be offered in-person and virtually.

Prerequisites

EECS 370

Prior experience with Python and linear algebra is helpful but not required. No physics background is required

Quick Links

Administrative Requests	Class Projects	Labs
Lecture Format	Turning in Projects	Exams
Discussion Format	Project Late Days	Lecture Participation
Grading Policy	Project Grading	Textbook
Homework Problem Set	Doing Your Own Project	Right to Revise

Administrative Requests

Please let us know as soon as possible if you experience interruptions to your studies and will need accommodations for homeworks, projects, or exams. **Students do not need to submit requests for lecture or lab absences, as attendance is not required or graded.**

For assignment extension requests (given for medical / personal emergencies, or approved conflicts communicated in advance, **must be made at least 24 hours before deadline** unless an emergency prevents prompt communication), please fill out the admin request forms linked on the course website

Email eecs498-001-staff@umich.edu for ***urgent issues*** and please include "EECS 498" in the subject line

Lecture Format

Lectures will be held live with recordings posted on the course website. Attendance is not required.

Lab Format

Lab sections will be in person. Slides and assignments will be released beforehand. Attendance is not mandatory but strongly encouraged. Labs will include assignments that must be submitted electronically to Gradescope and/or the Augorader by 11:55 pm the Wednesday after the lab.

Grading Policy

Final grades will be based on the total points earned on homework, projects and exams. The grade distribution is as follows:

Category	**#**	**%**
Projects	3	40% (12%, 14%, 14%)
Labs	9	18% (lowest 2 dropped)
Exams	3	42% (14% each)

The average grade in the course is expected to be a B+. Final grades will be based on a straight scale, but will be curved up if assignments are more difficult than expected and the average falls below this point.

Class Projects

Three projects will be assigned during the term, each of which will require a substantial time commitment on your part. Each project will involve writing code written in Python to meet provided specifications. Specifications will be released for each project on the course website once available. Lectures will provide the big picture aspects of the projects and labs will provide more details on implementation. Students may attend professor or staff office hours for extra help on the projects.

Turning in Projects

Projects are due at **11:55pm Eastern time exactly on the due date**.

You will be submitting your projects electronically by going to autograder.io. Your projects will be graded automatically using an autograder program. You are allowed to submit your programs as frequently as you wish. However, to deter you from using the autograder as a debugger, you may only submit to the autograder **THREE TIMES** a day. For each project, your final score will be derived from your best submission to the autograder.

Project Late Days

Sometimes unexpected problems make it difficult to get a project in on time. For this reason, each person will have a total of **3 free late days** to be used for projects throughout the semester.

If a project is due Monday 11:55 PM, and a person submits it any time before Tuesday 11:55 PM it will use up one late day. Similarly, if they submit it after Tuesday 11:55 PM but before Wednesday 11:55 PM it will use up two late days. The same logic applies to using three late days.

Note that late days are shared across all projects.

These late days should only be used to deal with unexpected problems such as illness. They should not be used simply to start later on a project or because you are having difficulty completing the project. Thus, please plan your work accordingly so that you won't need to use any of the late days unless there is a personal emergency. If you are having trouble understanding the material or designing a program, please come to office hours for help right away.

Project Grading

The projects will be graded primarily for correctness (doing all the required tasks and giving correct results in the proper format). The projects will be run on the CAEN Linux system. A small portion of a project's grade is submitting test cases to expose bugs in an instructor made “buggy” solution. Exposing these bugs will help you debug your own code and as well give you points towards the project. The autograder will give minimal feedback, only letting you know if you've passed the public test cases provided with the assignment and how many buggy solutions you have exposed. The spec associated with each project will provide more details.

Academic Integrity

We encourage collaboration in EECS 498, especially on concepts, tools, specifications, and strategies.

For lab assignments, you are allowed (but not required) to work with one or two other students.

For projects, you are welcome to use any code provided in lecture, labs, or on the Qiskit Website. While you are allowed to show another student your project code to discuss concepts (see below), you may not copy code.

See below for examples of approved collaboration:

Encouraged Collaboration	Unacceptable collaboration
Sharing high-level design strategies, e.g., helper function organization or data structure choices	Walking through an important piece of code step-by-step, sharing pseudocode, sharing comments in your code
Helping others understand the spec or project nuances	Providing your code as a reference, or looking at solutions found online as a reference
Helping someone debug	Debugging someone's code for them
Explaining a compiler error to someone	Fixing a compiler error for someone
Discussing test strategies	Sharing test code to verify someone's design, even if test cases are not submitted
Brainstorming edge cases for testing	Discussing specifics about what tests exposed instructor bugs on the autograder
Using starter code provided with a project or based on examples shown in lecture	Copying code in whole or in part, even if the code is modified Writing original code for someone else, or paying someone to write your project

Looking at someone else's code to understand concepts or help someone debug	Sharing your code in a way that could be copied, e.g., sending code over email or taking a picture of code

We consider getting help from AI tools like ChatGPT the same as getting help from humans. You are encouraged to use these tools to ask questions and improve your conceptual understanding (while verifying any answers it gives), but all work you submit (e.g. code or homework solutions) must be written solely by you or approved group members.

You are still responsible for following these rules even after finishing the course.

If you are unsure about what constitutes an honor code violation, please contact the course staff with questions.

For those retaking the course, you are allowed to resubmit work from a previous semester as long as it is entirely your own work.

Labs

There will be several lab assignments during the semester, and there is NO late day for lab assignments. These labs consist of a coded portion to be submitted to the autograder and worksheets to be submitted to Gradescope. Labs may be worked on individually or with a partner. Lab worksheets consist of programming and short answer problems and must be submitted by **11:55pm** the Wednesday after the lab meets, however labs are intended to be finished within the 2 hour lab period.

Students can re-submit as much as they would like until the deadline. Only the last submission will be graded, and labs are graded for accuracy.

Lab partners do not need to be the same people each week, and they can be students in different registered lab sections.

We will automatically drop the lowest two lab scores at the end of the semester.

Assessments

There will be **three** in-person assessments (mini-exams) this semester and will be delivered during normal lecture time. They will follow the format of questions posed in labs, containing a mix of programming, math, and conceptual questions. The exams will be non-cumulative and will focus on the following areas, respectively.

You are expected to take the exams at the scheduled times. If you do not take an exam without verifying a documented medical or personal emergency causing you to miss an exam, you will receive a zero for that exam. If you anticipate conflicts with the exam time, let the staff know by filling out the appropriate form by the posted deadline. The exam dates are announced at the beginning of the semester so you can avoid scheduling job interviews or other commitments on exam days. Outside commitments are not considered a valid reason for missing an exam.

We may not be able to accommodate requests submitted after the deadline. For last-minute emergency accommodations (*e.g.,* documented illness), please contact the staff at eecs498-001-staff@umich.edu.

Textbook

There is no required textbook for the class, but we will frequently make use of the Qiskit Online Textbook

If you are looking for a more in-depth focus on course material, *Quantum Computation and Quantum Information* by Michael Nielsen and Isaac Chung is an excellent source, but may be a bit intense for readers not very familiar with linear algebra.

Right to Revise

The course staff reserve the right to make changes to the syllabus at any time, as they see fit. When a revision occurs, it will be announced through Piazza, and it is your responsibility to be informed of such.

Lecture Topics:

1. Class Intro
2. Math Primer I
3. Math Primer II
4. Qubits
5. Multi-Qubit Gates
6. Phase Kickback
7. Intro to Algorithms
8. Deutsch-Jozsa Algorithm I
9. Deutsch-Jozsa Algorithm II
10. Grover's Algorithm I
11. Grover's Algorithm II
12. Quantum Fourier Transform
13. Quantum Phase Estimation
14. Shor's Algorithm
15. Quantum Architectures
16. Error Correction
17. Shor Code
18. Steane Code I
19. Steane Code II
20. Fault Tolerant Computing I
21. Fault Tolerant Computing II
22. Near Term Applications

Labs:

1. Linear Algebra
2. Into to Python / NumPy
3. Qiskit Overview
4. Designing Oracles
5. Phase Kickback Applications
6. Grover's Algorithm
7. Phase Estimation Applications
8. Shor's Algorithm
9. Error Correction
10. Fault Tolerance

Project 2: Study Group Scheduler

Checkpoint 1: 11:55 pm Tue 3/19
Checkpoint 2: 11:55 pm Tue 3/26
Final Deadline: 11:55 pm Tue 4/2

Direct Autograder Link
Starter Code
Useful Qiskit Functionality

Introduction

You have been tasked with designing a program to form study groups that meet a set of constraints. Being enrolled in EECS 498-001, you believe you can speed this task up by writing a quantum algorithm, specifically using Grover's algorithm and quantum counting.

When forming study groups, there may be multiple restrictions on what constitutes a valid group. For example, there may be a minimum size, we may want to avoid putting people with known time conflicts together, or we may want to guarantee that at least one student who is on track to pass the class is in each group.

A common way to express constraints (and the format we will be following in this project) is the Conjunctive Normal Form (CNF), i.e. an AND of ORs. For example, given three students Richard, Leon, and Jon, the constraints that

- At least least two people are placed in the group, and
- Leon and Jon should not both be picked

can be summarized in the following CNF form:

(Richard or Leon) and (Richard or Jon) and (Leon or Jon) and (Richard or Leon or Jon) and (~Leon or ~Jon)

(How to construct CNFs from constraints is beyond the scope of this project and not something we will worry about: we will just assume the CNFs are already provided for us)

This is a small enough example that it's easy to inspect by hand and verify that there are 2 possible solutions which meet these constraints

1. Richard and Leon
2. Richard and Jon

Each instance of a variable (in its regular or negated form) is called a "literal" and each OR statement forms a "clause".

Your work for this project will be divided across several components and implemented using the Qiskit SDK.

Part I: Oracle Design

[Reference]

Inside oracle.py, you will implement a function to generate a "Bitflip Oracle" from a CNF formula, and a function to convert a Bitflip oracle into a "Phase Oracle". The CNF formula will be passed in as a list of integer lists, where each integer corresponds to a unique variable (negative values indicating negation of the corresponding variable). Elements in the inner list form a clause of ORed values, and all clauses in the outer list are ANDed together. For example:

```
[[1,2,3],[2,-3],[4]]
```

represents the formula:

```
(var1 or var2 or var3) and (var2 or not(var3)) and (var4)
```

Note that because each variable must have an negated value, indexing starts at 1, not 0. For simplicity of implementation, you may assume that a variable does not appear for the first time (reading left to right) before a higher valued integer, no integers are skipped, and that a variable will not appear in the same clause more than once in either its normal or negated form (you do not need to check these conditions - you may assume they are always followed). For example, the following are invalid inputs and you may assume will never be passed in as arguments:

```
[[1,4],[2,3]]     # 4 appears before 3
[[1,-2],[2,4]]    # 3 is skipped
[[1,2],[3,-3]]    # 3 appears twice in second clause
```

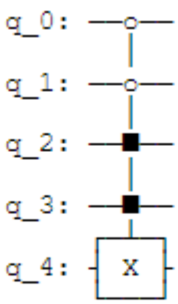
You also don't have to worry about empty CNFs for this project (i.e. every test is guaranteed to have at least one variable).

You have flexibility in how you design your oracles. In general, oracles can often be designed without the need for "ancilla bits" (bits used to hold an intermediate value), but you will likely find it simpler to include them. A straightforward solution is to store the OR result of each individual clause in a separate ancilla bit, and then AND each ancilla bit to store the final result. We recommend using the MCX gate (i.e. a multi-controlled X gate) which can be used to implement both multi-bit AND and OR gates (review De Morgan's Law if this is not clear). Individual inputs can be negated by setting "ctrl_state" to the appropriate bit-mask.

```
qc = QuantumCircuit(5)

num_ctrl_bits = 4

mcx_state = 0b1100 # invert all control bits except 3 and 2
```

```
gate = MCXGate(num_ctrl_bits, ctrl_state=mcx_state)

qc.append(gate, range(5))
```

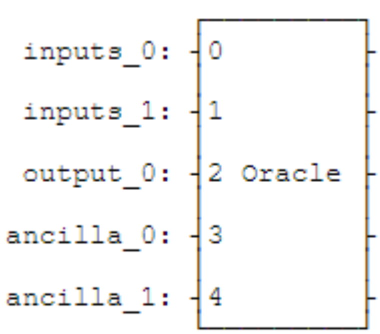
The above code generates a quantum circuit which flips the state of q_4 iff {q_3,q_2,q_1,q_0} = 4'b1100, i.e.:



Your bitflip oracle must read its inputs from the circuit's lowest indexed bits (with the higher indexed variables passed in via larger qubit index), the output should be stored in the next highest bit, and any ancilla bits should be placed on higher index bits. Any computation done on anything besides the target qubits must be "uncomputed" back to their original state so they can be reused for later computation.

For example, if we are provided a CNF for two variables and we use an additional 2 ancilla bits, the bits should be used as follows:

```
inputs = QuantumRegister(2, "inputs")
output = QuantumRegister(1, "output")
ancilla = AncillaRegister(2, "ancilla")
qc = QuantumCircuit(inputs, output, ancilla)
qc.append(oracle, range(5))
```



Where var_2 is fed into inputs_1 and var_1 into inputs_0 (the indices differ by one since the variable indices must start at 1, not 0). You should assume that all non-input qubits are initialized to |0> when passed into the circuit.

Your Phase Oracle must follow the same rules, except that there is no requirement of an "output bit" (the output is instead encoded in the relative phase of the state). However, the simplest solution is to keep the output bit, but prepare it in the |-> state, as described in lecture. You should assume that all non-input qubits are initialized to |0> when passed into the circuit and should be uncomputed back to |0> by the end.

Part II: Grover's Algorithm and Quantum Counting

References [1] [2] [3] [4]

Inside grover.py, you will implement functions to create a single iteration of the Grover operator (i.e. phase oracle implementing the provided CNF followed by a diffuser), as well as a full Grover implementation for a specified number of iterations. You can use your own oracle functions to test these functions, **but they should work with any oracle implementations that meet the above specifications** (i.e. your Grover implementation should work with oracles containing any number of ancilla bits).

counter.py contains the prototypes for functions to implement a quantum counting circuit, so that you can estimate the number of solutions to a constraint problem. Note that this circuit must return an estimate for the number of **solutions**. Implementing the diffuser as described in class results in a phase that would calculate the number of non-solutions. To fix this, you will need to alter the phase of the diffuser by -1. A simple way to do this is by placing the sequence ZXZX in the circuit.

The control method will be helpful for creating controlled versions of the Grover operator.

Part III: Driver

Once the other components of the project are completed, you will have everything you need to implement the constraint solver. driver.py will be run with a command line argument specifying the name of a comma-separated-value (CSV) file describing the constraints.

Each row of the CSV file is a comma-separated list of names (optionally prefixed by a tilde (~) character to indicate its negation) which forms a single clause. Each row is ANDed together to form the overall CNF formula. For example, the contents of file `test_1.csv`:

```
Richard,Leon,Jon
~Leon,~Jon
```

correspond to the CNF:

(Richard or Leon or Jon) and (~Leon or ~Jon)

Your driver should operate as follows, printing the specified messages to standard output when appropriate:

- Read in the CSV, and generate a corresponding CNF formatted array as described in Part I

  - Print "COUNT - Counting solutions for [N] variables..." where [N] is replaced by the number of distinct variables in the CNF
- Generate a quantum counter circuit using oracles implementing the CNF, using a precision of 5 bits in order to calculate the expected number of solutions to the constraint problem
  - Print "COUNT - Estimated number of solutions: [S]" where [S] is replaced by the estimated number of solutions (**not** rounded), with two decimal places.
- If the number of expected solutions (rounded to the nearest integer) is 0, the program should terminate.
  - Print "COUNT - No solutions expected, exiting"
- If the number of estimated iterations needed for optimality is less than 1 (this may or may not correspond to the solution space taking up half or more of the search space), then a dummy variable (set to false) should be added to the CNF and the quantum counting algorithm should be rerun
  - Print "COUNT - Solution space too large, rerunning with additional variable"
- Otherwise, print the estimation for the number of iterations (using the rounded number of solutions above, **but not rounding** the number of iterations)
  - Print "COUNT - Estimated number of Grover Iterations: [I]" where [I] is replaced by the estimated number of optimal iterations
- Grover's algorithm should then be run using the number of iterations estimated by the quantum counter, rounded down to the nearest integer
  - For this calculation, you should round the number of expected solutions to the nearest integer
  - Print "GROVER - Running search with [I] Grover iteration(s)" where [I] is replaced with the number of iterations
- The returned value from Grover's algorithm should be checked for correctness. If verified, a solution has been found and the program ends.
  - Print "GROVER - Solution identified: " followed by a space-separated list of names that satisfy the constraints (for simplicity, leave an extra trailing space at the end). Any valid solution can be accepted. **Each name should appear in the order they first appeared in the original CSV file**
- If a solution is not found, the algorithm should be run again for a maximum of 10 times before exiting.
  - Print ""GROVER: No solution found after 10 attempts"
  - NOTE: this stipulation is just made for completion sake. The probability of not measuring the correct solution after 10 attempts for our test cases is very low and is not something we will be checking.

It's recommended that you keep the number of "shots" low for these experiments low to avoid timing out. In particular, you probably only need one or two shots for running Grover's algorithm.

For the given file `test_1.csv`

```
Richard,Leon,Jon
~Leon,~Jon
```

The output should be (blank lines are optional and ignored):

```
COUNT - Counting solutions for 3 variables...
```

```
COUNT - Estimated number of solutions: 4.78
COUNT - Estimated number of Grover Iterations: 0.99
COUNT - Solution space too large, rerunning with additional variable

COUNT - Counting solutions for 4 variables...
COUNT - Estimated number of solutions: 4.94
COUNT - Estimated number of Grover Iterations: 1.40


GROVER - Running search with 1 Grover iteration(s)
GROVER - Solution identified: Richard Jon
```

Any of the other four possible solutions are also valid.

`test_2.csv` gives an example where no solutions are possible:

```
Richard
Leon,Jon
~Richard,~Leon
~Richard,~Jon
~Leon,~Jon
```

The output should be:

```
COUNT - Counting solutions for 3 variables...
COUNT - Estimated number of solutions: 0.00
COUNT - No solutions expected, exiting
```

Restrictions

While you are encouraged to reference these for your own testing and understanding, your submitted code may not use the following Qiskit libraries:

- qiskit.circuit.classicalfunction
- qiskit.circuit.library.PhaseOracle
- qiskit.circuit.library.GroverOperator
- qiskit.circuit.library.QFT
- qiskit.algorithms

Otherwise, you may use anything in the Qiskit SDK and the numpy, math, random, and unittest packages.

Testing

You must provide a set of test functions written in tests_p2_oracle.py and tests_p2_algorithms.py to the autograder. tests_p2_oracle.py should contain unit tests for each method in oracle.py and tests_p2_algorithms should contain unit tests for each method in counter.py and grover.py. You must use the Unittest model discussed in lab. Your tests will be graded on whether they cause assertion failures when run on buggy solutions, but do not cause assertion failures on correct implementations.

Submission and Grading

Submit oracle.py, grover.py, counter.py driver.py, tests_p2_oracle.py, and tests_p2_algorithms.py to the autograder using the direct link at the top of this page.

Because this project is larger in scope than P1, there are 2 checkpoints worth a moderate amount of your overall grade to let you know if you are on track to finish:

- Checkpoint 1 is worth 5% of the overall project grade. It will be calculated using your score for the oracle public tests, private tests, and mutation tests (i.e. the contents of oracle.py and tests_p2_oracle.py).
- Checkpoint 2 is worth 5% of the overall project grade. It will be calculated using your score for the algorithm public tests, private tests, and mutation tests (i.e. the contents of grover.py, counter.py and tests_p2_algorithms.py).
- The final submission is worth 90% of the overall project grade. It will be calculated by running all tests (including those from the checkpoints and the driver). Therefore, you can still earn some points that you missed from the checkpoints.

We will grade your code on functional correctness. As a reminder, you may not share any part of your solution with others. This includes both code and test cases. You are however encouraged to discuss the projects in a way that does not involve sharing code. You will get feedback on your total score, but you will not have access to what the private test cases are checking for.

Efficiency is not graded, but your code must complete in a reasonable amount of time. Note that for general quantum circuits, simulation takes an exponential amount of time. We will only grade your code on CNFs with up to 4 variables and 4 clauses.

Due to rounding, your unitary matrices and state vector calculations may slightly deviate from the correct answer. We will check that every value calculated is within .00001.

Your driver is the only design file that should add measurements to your circuits. The private tests will assume that you do not already have measurements on the circuits produced by oracle.py, grover.py, and counter.py. The private tests will fail if you already have measurements added or classical bits in your circuit. Accordingly, your test functions should add measurements when necessary.

In addition to checking simulated output, we will also test your code by checking the unitary matrices corresponding to your circuits*. You should therefore ensure that any phases match the documentation and that you properly uncompute when necessary.

**Note that because the spec offers some ambiguity in how you implement your functions, we will only check sub-matrices corresponding to bits that are actually measured. We will also give credit for any matrix that is within a global phase of the expected output.*

University of Michigan
Fall 2022 Instructor Report
EECS 498-001: Special Topics
Jonathan Beaumont

14 out of 36 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
This course advanced my understanding of the subject matter. (Q1631)	11	3	0	0	0	0	4.9	4.5	4.5
My interest in the subject has increased because of this course. (Q1632)	10	4	0	0	0	0	4.8	4.2	4.2
I knew what was expected of me in this course.(Q1633)	12	2	0	0	0	0	4.9	4.6	4.4
I had a strong desire to take this course.(Q4)	9	5	0	0	0	0	4.7	4.0	4.1
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	3	6	4	1	0	0	3.8	3.0	2.8

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median	Univ-wide Median	School/College Median
Jonathan Beaumont seemed well prepared for class meetings.(Q230)	11	3	0	0	0	0	4.9	4.8	4.8
Jonathan Beaumont explained material clearly.(Q199)	11	3	0	0	0	0	4.9	4.7	4.7
Jonathan Beaumont treated students with respect.(Q217)	13	1	0	0	0	0	5.0	4.8	4.8

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median
Overall, this was an excellent course. (Q1)	11	2	1	0	0	0	4.9
The textbook made a valuable contribution to the course. (Q64)	9	1	3	0	0	1	4.8
Prerequisites provided adequate preparation for this course. (Q61)	12	2	0	0	0	0	4.9
The laboratory was a valuable part of this course. (Q331)	8	5	1	0	0	0	4.6
Laboratory assignments were relevant to what was presented in class. (Q337)	12	2	0	0	0	0	4.9
I developed confidence in my abilities as an engineer. (Q1769)	8	2	2	0	0	2	4.8
I developed the ability to solve real world engineering problems. (Q1770)	6	4	2	0	0	2	4.5
Laboratory assignments required a reasonable amount of time and effort. (Q336)	8	3	1	1	0	1	4.7

Responses to questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Overall, Jonathan Beaumont was an excellent teacher. (Q2)	12	2	0	0	0	0	4.9

University of Michigan
Winter 2024 Instructor Preliminary Report
EECS 498-001: Special Topics
Jonathan Beaumont

66 out of 72 students responded to this evaluation.

Responses to University-wide questions about the course:

	SA	A	N	D	SD	N/A	Your Median
This course advanced my understanding of the subject matter.(Q1631)	51	15	0	0	0	0	4.9
My interest in the subject has increased because of this course.(Q1632)	38	25	3	0	0	0	4.6
I knew what was expected of me in this course.(Q1633)	47	18	1	0	0	0	4.8
I had a strong desire to take this course.(Q4)	41	23	2	0	0	0	4.7
As compared with other courses of equal credit, the workload for this course was (SA=Much Lighter, A=Lighter, N=Typical, D=Heavier, SD=Much Heavier). (Q891)	3	26	32	3	2	0	3.4

Responses to University-wide questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Jonathan Beaumont seemed well prepared for class meetings.(Q230)	58	8	0	0	0	0	4.9
Jonathan Beaumont explained material clearly.(Q199)	40	25	0	1	0	0	4.7
Jonathan Beaumont treated students with respect.(Q217)	61	5	0	0	0	0	5.0

Responses to questions about the course:

	SA	A	N	D	SD	N/A	Your Median
Overall, this was an excellent course. (Q1)	45	19	2	0	0	0	4.8
I felt included and valued when working with other students. (Q253)	36	16	2	0	0	11	4.8

Responses to questions about the instructor:

	SA	A	N	D	SD	N/A	Your Median
Overall, Jonathan Beaumont was an excellent teacher. (Q2)	56	10	0	0	0	0	4.9

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
☐ New Course
☑ Modification of Existing Course
☐ Deletion of Existing Course

Date of Submission: 2024-05-08
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			**REQUESTED LISTING**		
☐	Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 487			Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 487		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☐	Course Title (full title) Introduction to Natural Language Processing			Course Title (full title) Introduction to Natural Language Processing		
☐	Abbreviated Title (20 char) Intro to NLP			Abbreviated Title (20 char) Intro to NLP		
☐	Course Description (Please limit to 80 words and attach separate sheet if necessary) Fundamental theories and practical methods in natural language processing (NLP). Topics include syntax and parsing, lexical semantics and compositional semantics, discourse analysis, as well as applications in information extraction, sentiment analysis, question answering, summarization, dialogue systems, machine translation, and text generation.					
☐	Full Term Credit Hours Undergraduate Min: 4 Undergraduate Max: 4	Graduate Min: Graduate Max:		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☐	Course Credit Type Undergraduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Elec Engin & Computer Sci   Catalog: 487

☐	Grading Basis ☑ Graded (A – E) ☐ Credit/No Credit ☐ Satisfactory/Unsatisfactory ☐ Pass/Fail ☐ Business Administration Grading ☐ Not for Credit ☐ Not for Degree Credit ☐ Degree Credit Only	Add Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent	Drop Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent

	CURRENT LISTING	REQUESTED LISTING
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☐	Enforced Prerequisite (254 char) EECS 281; (C or better, No OP/F). Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C	Enforced Prerequisite (254 char) EECS 281; (C or better, No OP/F). Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C
☑	Credit Exclusions EECS 595	Credit Exclusions Credit for only one: EECS 487 or CSE 595

	Course Components	Graded Component	Terms Typically Offered
☐	☑ Lecture	☑	☑ Fall
	☐ Seminar	☐	☑ Winter
	☐ Recitation	☐	☐ Spring
	☑ Lab	☐	☐ Summer
	☐ Discussion	☐	☐ Spring/Summer
	☐ Independent Study	☐	

Cognizant Faculty Member Name: Rada Mihalcea   Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas   Email: vyas@umich.edu   Phone: 734-647-1754

CoE Curriculum Committee Representative: [signature]   Print: Amir Kamil   Date: 5/08/24

CoE Curriculum Committee Chair: Print: Date:

Home Department Chair: [signature]   Print: Andrew DeOrio   Date: 5/9/2024

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:

Course Description
Fundamental theories and practical methods in natural language processing (NLP). Topics include syntax and parsing, lexical semantics and compositional semantics, discourse analysis, as well as applications in information extraction, sentiment analysis, question answering, summarization, dialogue systems, machine translation, and text generation.

Class Length
Full term

Contact hours (lecture):
3

Contact hours (recitation)

Contact hours (lab)
2

Requested:

Course Description
Fundamental theories and practical methods in natural language processing (NLP). Topics include syntax and parsing, lexical semantics and compositional semantics, discourse analysis, as well as applications in information extraction, sentiment analysis, question answering, summarization, dialogue systems, machine translation, and text generation.

Class Length
Full term

Contact hours (lecture):
3

Contact hours (recitation)

Contact hours (lab)
2

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:

Special resources of facilities required for this course:

Supporting statement:
We are updating the credit exclusion to reflect the new subject code for CSE 595.

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
- ☐ New Course
- ☑ Modification of Existing Course
- ☐ Deletion of Existing Course

Date of Submission: 2024-02-16
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			REQUESTED LISTING		
☐	Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 492			Dept (Home): Elec Engin & Computer Sci Subject: EECS Catalog: 492		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☐	Course Title (full title) Introduction to Artificial Intelligence			Course Title (full title) Introduction to Artificial Intelligence		
☐	Abbreviated Title (20 char) Intro Art Intell			Abbreviated Title (20 char) Intro Art Intell		
☐	Course Description (Please limit to 80 words and attach separate sheet if necessary) Introduction to the core concepts of AI, organized around building computational agents. Emphasizes the application of AI techniques. Topics include search, logic, knowledge representation, reasoning, planning, decision making under the uncertainty, and machine learning.					
☐	Full Term Credit Hours Undergraduate Min: 4 Undergraduate Max: 4	Graduate Min: Graduate Max:		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☐	Course Credit Type Undergraduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Elec Engin & Computer Sci Catalog: 492

☐	Grading Basis ☑ Graded (A – E) ☐ Credit/No Credit ☐ Satisfactory/Unsatisfactory ☐ Pass/Fail ☐ Business Administration Grading ☐ Not for Credit ☐ Not for Degree Credit ☐ Degree Credit Only	Add Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent	Drop Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent

	CURRENT LISTING	**REQUESTED LISTING**
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☐	Enforced Prerequisite (254 char) EECS 281; (C or better, No OP/F). Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C	Enforced Prerequisite (254 char) EECS 281; (C or better, No OP/F). Enrollment in one minor elective allowed for Computer Science Minors. Minimum grade requirement: C
☑	Credit Exclusions Not for graduate credit	Credit Exclusions Credit for only one: EECS 492 or CSE 592

	Course Components	Graded Component	Terms Typically Offered
☐	☑ Lecture	☑	☑ Fall
	☐ Seminar	☐	☑ Winter
	☐ Recitation	☐	☐ Spring
	☐ Lab	☐	☐ Summer
	☑ Discussion	☐	☐ Spring/Summer
	☐ Independent Study	☐	

Cognizant Faculty Member Name: Emily Mower Provost Cognizant Faculty Member Title:

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Punam Vyas Email: vyas@umich.edu Phone: 734-647-1754

CoE Curriculum Committee Representative: [signature] Print: Amir Kamil Date: 4/05/24

CoE Curriculum Committee Chair: Print: Date:

Home Department Chair: [signature] Print: Andrew DeOrio Date: 04/05/2024

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:	**Requested:**
Course Description Introduction to the core concepts of AI, organized around building computational agents. Emphasizes the application of AI techniques. Topics include search, logic, knowledge representation, reasoning, planning, decision making under the uncertainty, and machine learning.	Course Description Introduction to the core concepts of AI, organized around building computational agents. Emphasizes the application of AI techniques. Topics include search, logic, knowledge representation, reasoning, planning, decision making under the uncertainty, and machine learning.
Class Length Full term	Class Length Full term
Contact hours (lecture): 3	Contact hours (lecture): 3
Contact hours (recitation) 1	Contact hours (recitation) 1
Contact hours (lab)	Contact hours (lab)

Additional Info:

Submitted by:
Home dept

Describe how this course fits with the degree requirements:

Special resources of facilities required for this course:

Supporting statement:
CSE 592 has a credit exclusion in place with EECS 492, but that was not reflected on the CARF for EECS 492. We are adding the reciprocal exclusion here.

Course Approval Request Form

Office of the Registrar, University of Michigan

1210 LSA Building
500 S. State Street
Ann Arbor, MI 48109-1382
Phone: 734.763.2113
Fax: 734.936.3148
ro.curriculum@umich.edu
ro.umich.edu

☑ **CHECK APPROPRIATE BOXES FOR ALL CHANGES**

Action Requested
☐ New Course
☑ Modification of Existing Course
☐ Deletion of Existing Course

Date of Submission: 2024-04-13
Effective Term: Winter 2025

☑	Course Offered ☑ Indefinitely ☐ One term only	**RO USE ONLY** Date Received: Date Completed: Completed By:

	CURRENT LISTING			**REQUESTED LISTING**		
☐	Dept (Home): Industrial & Operations Engin Subject: IOE Catalog: 333			Dept (Home): Industrial & Operations Engin Subject: IOE Catalog: 333		
☐	☐ Course is Cross-Listed with Other Departments			☐ Course is Cross-Listed with Other Departments		
	Department	Subject	Catalog Number	Department	Subject	Catalog Number
☐	Course Title (full title) Human Factors and Ergonomics			Course Title (full title) Human Factors and Ergonomics		
☐	Abbreviated Title (20 char) Human Factors Ergo			Abbreviated Title (20 char) Human Factors Ergo		
☑	Course Description (Please limit to 80 words and attach separate sheet if necessary) Introduction to human sensory, perceptual, cognitive, and physiological systems in the context of human-machine-environment systems. Discussion of methods for human-centered evaluation and design of cognitive and physical tasks, products and devices, vehicles and workplaces, such as displays, controls, human-computer and human-robot interactions, illumination and sound environments, repetitive and high physical effort tasks.					
☐	Full Term Credit Hours Undergraduate Min: 3 Undergraduate Max: 3	Graduate Min: Graduate Max:		Half Term Credit Hours Undergraduate Min: Undergraduate Max:	Graduate Min: Graduate Max:	
☐	Course Credit Type Undergraduate Student					
☐	Repeatability ☐ Course is Repeatable for Credit Maximum number of repeatable credits:			☐ Course is Y graded ☐ Can be taken more than once in the same term		

Subject: Industrial & Operations Engin   Catalog: 333

☐	Grading Basis ☑ Graded (A – E) ☐ Credit/No Credit ☐ Satisfactory/Unsatisfactory ☐ Pass/Fail ☐ Business Administration Grading ☐ Not for Credit ☐ Not for Degree Credit ☐ Degree Credit Only Add Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent Drop Consent ☐ Department Consent ☐ Instructor Consent ☑ No Consent

	CURRENT LISTING	**REQUESTED LISTING**
☐	Advisory Prerequisite (254 char)	Advisory Prerequisite (254 char)
☑	Enforced Prerequisite (254 char) Preceded or accompanied by IOE 265 Minimum grade requirement: C-	Enforced Prerequisite (254 char) Minimum grade requirement:
☐	Credit Exclusions	Credit Exclusions

	Course Components	Graded Component	Terms Typically Offered
☐	☑ Lecture	☑	☑ Fall
	☐ Seminar	☐	☑ Winter
	☐ Recitation	☐	☐ Spring
	☐ Lab	☐	☐ Summer
	☐ Discussion	☐	☐ Spring/Summer
	☐ Independent Study	☐	

Cognizant Faculty Member Name: Yili Liu   Cognizant Faculty Member Title: Professor

SIGNATURES ARE REQUIRED FROM ALL DEPARTMENTS INVOLVED (Please Print AND Sign Name)

Contact Person: Leonora Lucaj   Email: lucajl@umich.edu   Phone: 734-764-3297

CoE Curriculum
Committee Representative: Yavuz Bozer   [signature]   Print:Yavuz Bozer   Date: 05/06/24

CoE Curriculum Committee Chair: Print: Date:

Home Department Chair: Julie Ivy   [signature]   Print:Julie Ivey   Date: 05/06/24

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

Cross-Listed Department Chair: Print: Date:

DEPARTMENTAL/COLLEGE USE ONLY

Current:	**Requested:**
<u>Course Description</u> Introduction to human sensory, decision, control, and motor systems in the context of visual, auditory, cognitive and manual task evaluation and design. Problems with computer displays, illumination, noise, eye-hand coordination, as well as repetitive and high physical effort tasks are presented. Work place and vehicle design strategies used to resolve these are discussed.	<u>Course Description</u> Introduction to human sensory, perceptual, cognitive, and physiological systems in the context of human-machine-environment systems. Discussion of methods for human-centered evaluation and design of cognitive and physical tasks, products and devices, vehicles and workplaces, such as displays, controls, human-computer and human-robot interactions, illumination and sound environments, repetitive and high physical effort tasks.
<u>Class Length</u> Full term	<u>Class Length</u> Full term
<u>Contact hours (lecture):</u> 3	<u>Contact hours (lecture):</u> 3
<u>Contact hours (recitation)</u>	<u>Contact hours (recitation)</u>
<u>Contact hours (lab)</u>	<u>Contact hours (lab)</u>

Additional Info:

<u>Submitted by:</u>
Home dept

<u>Describe how this course fits with the degree requirements:</u>
This course is part of the required 33 credits of the IOE core requirements.

<u>Special resources of facilities required for this course:</u>

<u>Supporting statement:</u>
IOE 333 uses only basic concepts and simple calculations of probability (flip a fair coin—1/2, roll a 6-sided fair dice—1/6) and statistics such as normal distribution (its shape and meaning, not its complex math representation), mean, standard deviation, correlation, and percentile values. These are covered in IOE 333 textbook with easy graphical explanations and simple calculation formulas and tables, and can be taught with a few slides and 15-20 minutes of class time to anyone. Therefore, it is not necessary to be preceded or accompanied by IOE 265.