Department of Computer Science

Alexandru Nicolau, Department Chair
Rina Dechter, Department Vice Chair
3019 Donald Bren Hall
949-824-1546
http://www.cs.uci.edu/

Overview
With 40 full-time faculty members, 300+ graduate students, and more than 2,000 undergraduates, we provide a world-class research environment spanning not only the core areas of computer science — including computer architecture, system software, networking and distributed computing, data and information systems, the theory of computation, artificial intelligence, and computer graphics — but also highly interdisciplinary programs, such as biomedical informatics, data mining, security and privacy, and ubiquitous computing.

The diverse research interests of our faculty are reflected directly in our educational programs. Computer Science faculty teach most of the undergraduate and graduate courses for the degree programs in both Computer Science and Information & Computer Science. We jointly offer with our colleagues in The Henry Samueli School of Engineering an undergraduate degree in Computer Science and Engineering, as well as the graduate program in Networked Systems. We also have a major in Computer Game Science, jointly offered with the Department of Informatics.

Our department collaborates with many other institutions in the United States and abroad, and its doors are always open to a multitude of visitors and collaborators from all corners of the globe.

Undergraduate Major in Computer Science
The Computer Science major emphasizes the principles of computing that underlie our modern world, and provides a strong foundational education to prepare students for the broad spectrum of careers in computing. This major can serve as preparation for either graduate study or a career in industry. Students receive a solid background in low-level architecture and systems; middle-level infrastructure, algorithms, and mathematical foundations. This is a highly flexible degree that allows students to explore a broad range of topics in modern computing. In order to achieve some focus in their upper-division studies, students are required to satisfy the requirements for one of the nine specializations described below.

Algorithms. This specialization focuses on fundamental computational techniques, including their analysis and applications to topics in computer vision, computer games, graphics, artificial intelligence, and information retrieval. Topics include data structures, graph and network algorithms, computational geometry, probabilistic algorithms, complexity theory, and cryptography.

Architecture and Embedded Systems. This specialization integrates principles of embedded systems, software, hardware, computer architecture, distributed systems and networks, and prepares students to design and create efficient hardware/software architectures for emerging application areas. Students in this specialization will build upon a strong foundation in software and hardware and learn how to design networked embedded systems, and efficient computer architectures for a diverse set of application domains such as gaming, visualization, search, databases, transaction processing, data mining, and high-performance and scientific computing.

Bioinformatics. This specialization introduces students to the interdisciplinary intersection of biology and medicine with computer science and information technology. Students who complete the specialization will understand biomedical computing problems from the computer science perspectives, and be able to design and develop software that solves computational problems in biology and medicine.

General Computer Science. This specialization allows students to acquire a well-rounded knowledge of computer science that may be tailored to their individual interests. Students choose 11 upper-division computer science courses, including two project courses. This specialization will appeal to those who are interested in a broad education in computer science, or who wish to create their own unique specialization not found in the current list of (other) specializations under this major.

Information. This specialization is intended to prepare students for working with and developing a wide variety of modern data and information systems. Topics covered by this concentration include database management, information retrieval, Web search, data mining, and data-intensive computing.

Intelligent Systems. This specialization will introduce students to the principles underlying intelligent systems, including topics such as representing human knowledge, building automated reasoning systems, developing intelligent search techniques, and designing algorithms that adapt and learn from data. Students in this specialization will use these principles to solve problems across a variety of applications such as computer vision, information retrieval, data mining, automated recommender systems, bioinformatics, as well as individually designed projects.

Networked Systems. This specialization focuses on Internet architecture, Internet applications, and network security. It also encourages students to learn about operating systems, databases, search, programming, embedded systems, and performance.

Systems and Software. This specialization deals with principles and design of systems and software. It emphasizes the interaction between software and the computing infrastructure on which it runs and the performance impact of design decisions. Core topics include the hardware/software interface, languages and compilers, operating systems, parallel and distributed computing. Elective topics include networking, security, graphics, and databases.
**Visual Computing.** This specialization encompasses the digital capture, processing, synthesis and display of visual data such as images and video. This specialization includes computer vision, image processing, and graphics, and covers such topics as the representation of 3D objects, visual recognition of objects and people, interactive and photo-realistic image rendering, and physics and perception of light and color.

The Department also offers a joint undergraduate degree in Computer Science and Engineering, in conjunction with The Henry Samueli School of Engineering; information is available in the Interdisciplinary Studies section of the Catalogue.

**Admissions**

**Freshman Applicants:** See the Undergraduate Admissions section.

**Transfer Applicants:**

Junior-level applicants who satisfactorily complete course requirements will be given preference for admission. Applicants must satisfy the following requirements:

1. Complete one year of approved college-level math, preferably courses in calculus equivalent to UCI’s MATH 2A - MATH 2B; if not available, one year of coursework equivalent to other major-related math courses is acceptable.
2. Completion of one year of transferable computer science courses involving concepts such as those found in Java, Python, C++, data structures, or other object-oriented or high-level programming language.

**NOTE:** The introductory sequence in ICS has moved to Python. The Bren School of ICS strongly encourages all participants to become familiar with this programming language prior to matriculation. Additional computer science courses beyond the two required are strongly recommended, particularly those that align with the major(s) of interest. Java is used extensively in the curriculum; therefore, transfer students should plan to learn it by studying on their own or by completing a Java-related programming course prior to their first quarter at UCI.

Additional courses beyond those required for admission must be taken to fulfill the lower-division degree requirements, as many are prerequisites for upper-division courses. For some transfer students, this may mean that it will take longer than two years to complete their degree.

**Major and Minor Restrictions**

Bren School of ICS majors (including shared majors, BIM and CSE) pursuing minors within the Bren School of ICS may not count more than five courses toward both the major and minor. Some ICS majors and minors outside of the School are not permitted due to significant overlap. Visit the ICS Student Affairs Office website for Majors and Minors restrictions. ([http://www.ics.uci.edu/grad/degrees/MajorMinor_Restrictions_Chart.pdf](http://www.ics.uci.edu/grad/degrees/MajorMinor_Restrictions_Chart.pdf)) All students should check the Double Major Restrictions Chart ([http://www.ics.uci.edu/grad/degrees/Dbl_Major_Restr_Chart.pdf](http://www.ics.uci.edu/grad/degrees/Dbl_Major_Restr_Chart.pdf)) and view our information page ([http://www.ics.uci.edu/grad/degrees/Double_Majors.php](http://www.ics.uci.edu/grad/degrees/Double_Majors.php)) on double majoring to see what degree programs are eligible for double majoring.

**Requirements for the B.S. Degree in Computer Science**

All students must meet the University Requirements.

**Major Requirements**

**Lower-division**

<table>
<thead>
<tr>
<th>A. Core</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I&amp;C SCI 31</td>
<td>Introduction to Programming</td>
</tr>
<tr>
<td>I&amp;C SCI 32</td>
<td>Programming with Software Libraries</td>
</tr>
<tr>
<td>I&amp;C SCI 33</td>
<td>Intermediate Programming</td>
</tr>
<tr>
<td>I&amp;C SCI 45C</td>
<td>Programming in C/C++ as a Second Language</td>
</tr>
<tr>
<td>I&amp;C SCI 46</td>
<td>Data Structure Implementation and Analysis</td>
</tr>
<tr>
<td>I&amp;C SCI 51</td>
<td>Introductory Computer Organization</td>
</tr>
<tr>
<td>I&amp;C SCI 53</td>
<td>Principles in System Design</td>
</tr>
<tr>
<td>I&amp;C SCI 53L</td>
<td>Principles in System Design Library</td>
</tr>
<tr>
<td>I&amp;C SCI 90</td>
<td>New Students Seminar</td>
</tr>
<tr>
<td>IN4MATX 43</td>
<td>Introduction to Software Engineering</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B. Complete:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MATH 2A- 2B</td>
<td>Single-Variable Calculus and Single-Variable Calculus</td>
</tr>
<tr>
<td>I&amp;C SCI 6B</td>
<td>Boolean Algebra and Logic</td>
</tr>
<tr>
<td>I&amp;C SCI 6D</td>
<td>Discrete Mathematics for Computer Science</td>
</tr>
<tr>
<td>I&amp;C SCI 6N</td>
<td>Computational Linear Algebra</td>
</tr>
<tr>
<td>or MATH 3A</td>
<td>Introduction to Linear Algebra</td>
</tr>
<tr>
<td>STATS 67</td>
<td>Introduction to Probability and Statistics for Computer Science</td>
</tr>
</tbody>
</table>

---

**Department of Computer Science**

UCI General Catalogue 2016-2017
C. Two courses approved for General Education category II except those offered by CSE, Economics, ICS, or Mathematics. University Studies courses can be used with the approval of the CS Vice Chair for Undergraduate Studies.

**Upper-division**

A. Core

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 161</td>
<td>Design and Analysis of Algorithms</td>
</tr>
<tr>
<td>I&amp;C SCI 139W</td>
<td>Critical Writing on Information Technology</td>
</tr>
</tbody>
</table>

B. Upper-division electives: Select 11 upper-division courses from the list below. Sections B-1 and B-2 must be completed as part of the 11 upper-division electives.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 111-160, 162-189</td>
<td>Concepts of Programming Language II</td>
</tr>
<tr>
<td>IN4MATX 102</td>
<td>Requirements Analysis and Engineering</td>
</tr>
<tr>
<td>IN4MATX 113</td>
<td>Software Testing, Analysis, and Quality Assurance</td>
</tr>
<tr>
<td>IN4MATX 115</td>
<td>Software Design: Applications</td>
</tr>
<tr>
<td>IN4MATX 121</td>
<td>Software Design: Structure and Implementation</td>
</tr>
<tr>
<td>IN4MATX 122</td>
<td>Internet Applications Engineering</td>
</tr>
<tr>
<td>IN4MATX 131</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>IN4MATX 133</td>
<td>User Interaction Software</td>
</tr>
<tr>
<td>IN4MATX 134</td>
<td>Project in User Interaction Software</td>
</tr>
<tr>
<td>I&amp;C SCI 160</td>
<td>Graphics Processors and Game Platforms</td>
</tr>
<tr>
<td>I&amp;C SCI 161</td>
<td>Game Engine Lab</td>
</tr>
<tr>
<td>I&amp;C SCI 162</td>
<td>Modeling and World Building</td>
</tr>
</tbody>
</table>

B-1. Project Courses: Choose at least two projects courses from the following list:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 113</td>
<td>Computer Game Development</td>
</tr>
<tr>
<td>COMPSCI 114</td>
<td>Projects in Advanced 3D Computer Graphics</td>
</tr>
<tr>
<td>COMPSCI 117</td>
<td>Project in Computer Vision</td>
</tr>
<tr>
<td>COMPSCI 122B</td>
<td>Project in Databases and Web Applications</td>
</tr>
<tr>
<td>COMPSCI 122C</td>
<td>Principles of Data Management</td>
</tr>
<tr>
<td>COMPSCI 133</td>
<td>Advanced Computer Networks</td>
</tr>
<tr>
<td>COMPSCI 142B</td>
<td>Language Processor Construction</td>
</tr>
<tr>
<td>COMPSCI 143B</td>
<td>Project in Operating System Organization</td>
</tr>
<tr>
<td>COMPSCI 145-145L</td>
<td>Embedded Software and Embedded Software Laboratory</td>
</tr>
<tr>
<td>COMPSCI 153</td>
<td>Logic Design Laboratory</td>
</tr>
<tr>
<td>COMPSCI 154</td>
<td>Computer Design Laboratory</td>
</tr>
<tr>
<td>COMPSCI 165</td>
<td>Project In Algorithms And Data Structures</td>
</tr>
<tr>
<td>COMPSCI 175</td>
<td>Project in Artificial Intelligence</td>
</tr>
<tr>
<td>COMPSCI 189</td>
<td>Project in Bioinformatics</td>
</tr>
<tr>
<td>IN4MATX 134</td>
<td>Project in User Interaction Software</td>
</tr>
</tbody>
</table>

B-2. Specialization: Select and satisfy the requirements for one of the specializations below. (Note: Students may not pursue more than one specialization.)

Some of the specializations include recommended electives. These are courses related to the specialization and intended to help students choose courses to take toward their upper-division elective requirement.

**Algorithms**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 162</td>
<td>Formal Languages and Automata</td>
</tr>
<tr>
<td>COMPSCI 163</td>
<td>Graph Algorithms</td>
</tr>
<tr>
<td>COMPSCI 164</td>
<td>Computational Geometry and Geometric Modeling</td>
</tr>
<tr>
<td>COMPSCI 165</td>
<td>Project In Algorithms And Data Structures</td>
</tr>
<tr>
<td>COMPSCI 167</td>
<td>Introduction to Applied Cryptography</td>
</tr>
<tr>
<td>COMPSCI 169</td>
<td>Introduction to Optimization</td>
</tr>
</tbody>
</table>

**Architecture and Embedded Systems: four courses from the following list:**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 144</td>
<td>High-performance Computers and Program Optimization</td>
</tr>
<tr>
<td>COMPSCI 145-145L</td>
<td>Embedded Software and Embedded Software Laboratory</td>
</tr>
</tbody>
</table>
**COMPSCI 151** 
Digital Logic Design

**COMPSCI 152** 
Computer Systems Architecture

**COMPSCI 153** 
Logic Design Laboratory

**COMPSCI 154** 
Computer Design Laboratory

**Recommended electives:**

- **COMPSCI 131** 
Parallel and Distributed Computing

- **COMPSCI 142A** 
Compilers and Interpreters

- **COMPSCI 143A** 
Principles of Operating Systems

**Bioinformatics: three courses from the following list:**

- **COMPSCI 184A** 
Representations and Algorithms for Molecular Biology

and two courses from:

- **COMPSCI 172B** 
Neural Networks and Deep Learning

- **COMPSCI 184B** 
Probabilistic Modeling of Biological Data

- **COMPSCI 184C** 
Computational Systems Biology

- **COMPSCI 189** 
Project in Bioinformatics

**General CS track**

COMPSCI 111-189, except COMPSCI 161

**Information**

- **COMPSCI 121** 
Information Retrieval

- **COMPSCI 122A** 
Introduction to Data Management

- **COMPSCI 178** 
Machine Learning and Data-Mining

and four courses from:

- **I&C SCI 45J** 
Programming in Java as a Second Language

- **COMPSCI 122B** 
Project in Databases and Web Applications

- **COMPSCI 125** 
Next Generation Search Systems

- **COMPSCI 132** 
Computer Networks

- **COMPSCI 134** 
Computer and Network Security

- **COMPSCI 141** 
Concepts in Programming Languages I

- **COMPSCI 142A** 
Compilers and Interpreters

- **COMPSCI 143A** 
Principles of Operating Systems

- **COMPSCI 163** 
Graph Algorithms

- **COMPSCI 165** 
Project In Algorithms And Data Structures

- **COMPSCI 167** 
Introduction to Applied Cryptography

- **COMPSCI 179** 
Algorithms for Probabilistic and Deterministic Graphical Models

at least one of which must be:

- **COMPSCI 122B** 
Project in Databases and Web Applications

- **COMPSCI 125** 
Next Generation Search Systems

- **COMPSCI 179** 
Algorithms for Probabilistic and Deterministic Graphical Models

**Intelligent Systems**

- **COMPSCI 171** 
Introduction to Artificial Intelligence

- **COMPSCI 175** 
Project in Artificial Intelligence

- **COMPSCI 178** 
Machine Learning and Data-Mining

and at least one course from:

- **COMPSCI 177** 
Applications of Probability in Computer Science

- **COMPSCI 179** 
Algorithms for Probabilistic and Deterministic Graphical Models

and at least one course from:

- **COMPSCI 162** 
Formal Languages and Automata

- **COMPSCI 163** 
Graph Algorithms

- **COMPSCI 164** 
Computational Geometry and Geometric Modeling

- **COMPSCI 169** 
Introduction to Optimization

and at least one course from:

- **COMPSCI 116** 
Computational Photography and Vision
COMPSCI 121 | Information Retrieval
COMPSCI 125 | Next Generation Search Systems
COMPSCI 174 | Bioinformatics
COMPSCI 184B | Probabilistic Modeling of Biological Data

**Networked Systems**
COMPSCI 132 | Computer Networks
COMPSCI 133 | Advanced Computer Networks
COMPSCI 134 | Computer and Network Security
COMPSCI 143A | Principles of Operating Systems

Recommended electives:
- One course from:
  - COMPSCI 122B | Project in Databases and Web Applications
  - COMPSCI 143B | Project in Operating System Organization
- Two courses from:
  - COMPSCI 122A | Introduction to Data Management
  - COMPSCI 131 | Parallel and Distributed Computing
  - COMPSCI 137 | Internet Applications Engineering
  - COMPSCI 167 | Introduction to Applied Cryptography
  - COMPSCI 145-145L | Embedded Software and Embedded Software Laboratory
  - COMPSCI 163 | Graph Algorithms
  - COMPSCI 169 | Introduction to Optimization

**Systems and Software: three courses from the following list:**
COMPSCI 131 | Parallel and Distributed Computing
COMPSCI 141 | Concepts in Programming Languages I
COMPSCI 142A | Compilers and Interpreters
COMPSCI 142B | Language Processor Construction
COMPSCI 143A | Principles of Operating Systems
COMPSCI 143B | Project in Operating System Organization

Recommended electives:
- COMPSCI 132 | Computer Networks
- COMPSCI 134 | Computer and Network Security
- COMPSCI 144 | High-performance Computers and Program Optimization
- COMPSCI 152 | Computer Systems Architecture

**Visual Computing: four courses from the following list:**
COMPSCI 111 | Digital Image Processing
COMPSCI 112 | Computer Graphics
COMPSCI 114 | Projects in Advanced 3D Computer Graphics
COMPSCI 116 | Computational Photography and Vision
COMPSCI 117 | Project in Computer Vision
I&C SCI 162 | Modeling and World Building

**Sample Program of Study — Computer Science**

**Freshman**

<table>
<thead>
<tr>
<th>Fall</th>
<th>Winter</th>
<th>Spring</th>
</tr>
</thead>
<tbody>
<tr>
<td>I&amp;C SCI 31</td>
<td></td>
<td>I&amp;C SCI 33</td>
</tr>
<tr>
<td>MATH 2A</td>
<td>MATH 2B</td>
<td>IN4MATX 43</td>
</tr>
<tr>
<td>WRITING 39A</td>
<td>WRITING 39B</td>
<td>I&amp;C SCI 6B</td>
</tr>
<tr>
<td>I&amp;C SCI 90</td>
<td>General Education III</td>
<td>WRITING 39C</td>
</tr>
</tbody>
</table>

**Sophomore**

<table>
<thead>
<tr>
<th>Fall</th>
<th>Winter</th>
<th>Spring</th>
</tr>
</thead>
<tbody>
<tr>
<td>I&amp;C SCI 51</td>
<td></td>
<td>Computer Science Spec./Elective</td>
</tr>
<tr>
<td>I&amp;C SCI 6D</td>
<td>I&amp;C SCI 53</td>
<td>STATS 67</td>
</tr>
<tr>
<td>I&amp;C SCI 45C</td>
<td>I&amp;C SCI 53L</td>
<td>General Education III</td>
</tr>
</tbody>
</table>
### Undergraduate Major in Computer Science and Engineering (CSE)

This program is administered jointly by the Department of Computer Science in the Bren School of ICS, and the Department of Electrical Engineering and Computer Science (EECS) in The Henry Samueli School of Engineering. For information, see the Interdisciplinary Studies section of the Catalogue.

### Requirements for the B.S. Degree in Computer Science and Engineering

All students must meet the University Requirements.

**Major Requirements:** See the Interdisciplinary Studies section.

### Minor in Bioinformatics

The minor provides a focused study of bioinformatics to supplement a student’s major program of study and prepares students for a profession, career, or academic pursuit in which biomedical computing is an integral part but is not the primary focus. The Bioinformatics minor contributes to students’ competence in computing applied to biomedical problems and data, as well as exposing them to the fundamentals of the life sciences from a computer science perspective. The minor allows students sufficient flexibility to pursue courses that complement their major field or address specific interests.

Students who complete the minor requirements will be able to do the following: synthesize computer science, quantitative methods, and biological science; understand the synergistic set of reciprocal influences between life and computational sciences and technologies; discuss biomedical computing problems and corresponding computer science perspectives; and employ principles, methods, and technologies fundamental to biomedical computing.

### Requirements for the Minor in Bioinformatics

Select one of the following:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>I&amp;C SCI 21- 22</td>
<td>Introduction to Computer Science I and Introduction to Computer Science II</td>
</tr>
<tr>
<td>or</td>
<td></td>
</tr>
<tr>
<td>I&amp;C SCI I31/CSE 41</td>
<td>Introduction to Programming</td>
</tr>
<tr>
<td>Complete:</td>
<td></td>
</tr>
<tr>
<td>I&amp;C SCI I32/CSE 42</td>
<td>Programming with Software Libraries</td>
</tr>
<tr>
<td>I&amp;C SCI I33/CSE 43</td>
<td>Intermediate Programming</td>
</tr>
<tr>
<td>BIO SCI 93</td>
<td>From DNA to Organisms</td>
</tr>
<tr>
<td>COMPSCI 183/BIO SCI M123</td>
<td>Introduction to Computational Biology</td>
</tr>
<tr>
<td>COMPSCI 184A</td>
<td>Representations and Algorithms for Molecular Biology</td>
</tr>
<tr>
<td>COMPSCI 184B</td>
<td>Probabilistic Modeling of Biological Data</td>
</tr>
<tr>
<td>or COMPSCI 184C</td>
<td>Computational Systems Biology</td>
</tr>
<tr>
<td>COMPSCI 189</td>
<td>Project in Bioinformatics</td>
</tr>
</tbody>
</table>
NOTE: A maximum of two courses may be taken Pass/Not Pass toward a minor. Bren School majors should refer to the Majors/Minors Restrictions Catalogue section before attempting to minor in Bioinformatics. Students who are considering a major in Computer Science or Computer Science and Engineering must complete the major requirements for a letter grade. Visit the ICS Student Affairs Office website for Majors and Minors restrictions (http://www.ics.uci.edu/ugrad/degrees/MajorMinor_Restrictions_Chart.pdf)

On This Page:

- Master of Science in Computer Science
- Doctor of Philosophy in Computer Science
  - Research Project for the Ph.D. Degree
  - Advancement to Candidacy Examination
  - Doctoral Dissertation Topic Defense
  - Doctoral Dissertation and Final Examination
- Graduate Program in Mathematical, Computational, and Systems Biology

Graduate Program in Computer Science

The field of Computer Science is concerned with the design, analysis, and implementation of computer systems as well as the use of computation as it is applied to virtually every field of study and use in the everyday world. Computer systems can range in scope from small embedded systems to the Internet as a whole. Research in computer science involves mathematical analysis, empirical experimentation, and the implementation of prototype systems. Core research areas include artificial intelligence and machine learning, bioinformatics, computer architecture, embedded systems, graphics and visual computing, databases and information management, multimedia, networked and distributed systems, programming languages and compilers, security and cryptography, design and analysis of algorithms, scientific computing, and ubiquitous computing.

The M.S. and Ph.D. degrees in Computer Science (CS) are broad and flexible programs which offer students opportunities for graduate study in the full spectrum of intellectual activity in computer science.

Master of Science in Computer Science

The course requirements for the M.S. are identical to those of the Ph.D., although completion plans differ. For general information about admissions, the M.S. completion plan options, visit the Bren School of ICS Graduate Programs section of the Catalogue.

Doctor of Philosophy in Computer Science

Required Courses

Each student must complete at least 47 units of course work with an average GPA of at least 3.5 for Ph.D. students and 3.0 for M.S. students. In addition, students must receive at least a B in each course counted toward filling these requirements. The set of core and elective courses chosen by a student must be approved by the student's research advisor before advancement to candidacy. Faculty associated with each research area will provide suggested curricula for that area to guide students in their selection of courses. These curricula will also help Ph.D. students to prepare for their candidacy examination (see below) which must be taken in a specific research area.

Students must complete three quarters of COMPSCI 200S, four core courses, and seven elective courses. The course requirements are as follows:

Students must select four areas from the list of seven areas given below. From each area, they must select at least one of the courses listed for that area.

Data Structures and Algorithms

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 260</td>
<td>Fundamentals of the Design and Analysis of Algorithms</td>
</tr>
<tr>
<td>COMPSCI 261</td>
<td>Data Structures</td>
</tr>
<tr>
<td>COMPSCI 263</td>
<td>Analysis of Algorithms</td>
</tr>
</tbody>
</table>

Architecture/Embedded Systems

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 250A</td>
<td>Computer Systems Architecture</td>
</tr>
<tr>
<td>COMPSCI 244</td>
<td>Introduction to Embedded and Ubiquitous Systems</td>
</tr>
</tbody>
</table>

System Software

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 241</td>
<td>Advanced Compiler Construction</td>
</tr>
<tr>
<td>COMPSCI 243</td>
<td>High-Performance Architectures and Their Compilers</td>
</tr>
<tr>
<td>COMPSCI 230</td>
<td>Distributed Computer Systems</td>
</tr>
</tbody>
</table>

Artificial Intelligence

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPSCI 271</td>
<td>Introduction to Artificial Intelligence</td>
</tr>
<tr>
<td>COMPSCI 273A</td>
<td>Machine Learning</td>
</tr>
</tbody>
</table>
Networks/Multimedia

- COMPSCI 232 Computer and Communication Networks
- COMPSCI 203 Network and Distributed Systems Security
- COMPSCI 212 Multimedia Systems and Applications

Database Systems

- COMPSCI 222 Principles of Data Management
- COMPSCI 223 Transaction Processing and Distributed Data Management

Scientific and Visual Computing

- COMPSCI 206 Principles of Scientific Computing
- COMPSCI 211A Visual Computing

Seven elective courses from any set of CS, Informatics, or Statistics courses, including the above core courses, but excluding COMPSCI 290, COMPSCI 298, COMPSCI 299, or any course with a suffix of “S.”

Two of these courses can be graduate courses offered by a department outside of ICS, with written consent of the advisor (M.S. students must obtain written consent from the Computer Science Vice Chair for Graduate Studies).

Two of the courses can be undergraduate courses from the following list:

- COMPSCI 111 Digital Image Processing
- COMPSCI 112 Computer Graphics
- COMPSCI 122A Introduction to Data Management
- COMPSCI 132 Computer Networks
- COMPSCI 142A Compilers and Interpreters
- COMPSCI 143A Principles of Operating Systems
- COMPSCI 152 Computer Systems Architecture
- COMPSCI 161 Design and Analysis of Algorithms
- COMPSCI 171 Introduction to Artificial Intelligence
- COMPSCI 178 Machine Learning and Data-Mining
- I&C SCI 161 Game Engine Lab
- I&C SCI 162 Modeling and World Building
- I&C SCI 163 Mobile and Ubiquitous Games
- I&C SCI 166 Game Design

Students may not retake courses they have used toward an undergraduate degree and receive credit toward the graduate requirements.

No more than two undergraduate courses or COMPSCI 295 may be taken to satisfy elective course requirements.

Ph.D. students are required to serve as teaching assistants for at least two quarters.

Research Project for the Ph.D. Degree

Doctoral students must find a faculty advisor and successfully complete a research project with that faculty member by the end of their second year. In coordination with this project the student must also take at least one independent studies course (COMPSCI 299) with their faculty advisor. The objective of the research project is to demonstrate early in the program the student’s ability to carry out basic research in computer science.

Finally, the student must present the outcome of the research in a technical report, which must be approved by the advisor. The project may or may not be a stepping-stone toward a dissertation, and must be completed by the end of the second year, and prior to advancement to candidacy.

Advancement to Candidacy Examination

The objective of the candidacy examination is to demonstrate in-depth knowledge of an area of computer science and readiness to carry out independent research at the doctoral level in that area. The student must complete all pre-candidacy course requirements and the research project prior to advancing to candidacy. All requirements for candidacy including the candidacy examination must be completed by the end of the third year (or, for students entering the program with an M.S. in Computer Science, by the end of the second year). If the student does not pass on the first trial, the student will be allowed until the end of the first quarter of the fourth year to advance to candidacy. Consult the ICS Graduate Office for policies regarding committee membership. The format is an oral examination during which the student is tested on knowledge relevant to the chosen area of specialization. Each area is defined by a set of topics and reading list, which are maintained by the Computer Science Department office. New areas or changes to existing areas must be approved by a majority vote of the CS faculty in accordance with the Department’s bylaws. The current areas include the following: Algorithms and Data Structures; Computer Architecture and Embedded Systems; Database Systems and Multimedia; Computer Networks;
Distributed Systems; Artificial Intelligence and Machine Learning; Informatics in Biology and Medicine; Computer Graphics and Visual Computing; Cryptography and Computer Security; Computational Neuroscience; Scientific Computing; Systems Software.

The examination is graded pass or fail. In order to pass, the Candidacy Committee must unanimously approve the final outcome. In the case of a fail, the examination may be retaken once. Students who fail on the second try will be recommended for disqualification from the doctoral program.

**Doctoral Dissertation Topic Defense**

The student must produce a substantial written document representing the dissertation plan. This must include the proposed dissertation abstract, a dissertation outline, and a detailed plan for completing the work. A dissertation defense committee is formed in accordance with UCI Senate regulations. The dissertation committee must unanimously approve the student’s proposal. At the discretion of the student’s advisor, the student may be required to give an oral presentation of the proposed plan to the committee. This must be completed by the end of the fourth year. It is expected that this will be done at least a year prior to the final examination and before most of the dissertation research and writing are undertaken. The idea is for students to demonstrate that they have a clear plan for carrying out the research for their dissertation. It also gives the student an understanding of what will be expected for final approval of the dissertation.

**Doctoral Dissertation and Final Examination**

Ph.D. students are required to complete a Ph.D. dissertation in accordance with Academic Senate regulations. In addition, they must pass an oral dissertation defense which consists of a public seminar presenting results followed by a private examination by the doctoral committee and other interested members of the Computer Science Department faculty.

Students entering the Ph.D. program with an M.S. in Computer Science must advance to candidacy within two years. All others must advance within three years. The normative time for completion of the Ph.D. is five years, and the maximum time permitted is seven years.

**Graduate Program in Mathematical, Computational, and Systems Biology**

The graduate program in Mathematical, Computational, and Systems Biology (MCSB) is designed to meet the interdisciplinary training challenges of modern biology and function in concert with selected department programs, including the Ph.D. in Computer Science. Detailed information is available at the Mathematical, Computational, and Systems Biology website (http://mcsb.uci.edu) and in the Interdisciplinary Studies section of the Catalogue.

**Faculty**

Shannon L. Alfaro, M.S. University of California, Irvine, *Lecturer of Computer Science*

Animashree Anandkumar, Ph.D. Cornell University, *Assistant Professor of Electrical Engineering and Computer Science; Computer Science* (statistical inference and learning of graphical models, scalable network algorithms)

Nader Bagherzadeh, Ph.D. University of Texas at Austin, *Professor of Electrical Engineering and Computer Science; Computer Science* (parallel processing, computer architecture, computer graphics, VLSI design)

Pierre F. Baldi, Ph.D. California Institute of Technology, *UCI Chancellor's Professor of Computer Science; Biological Chemistry; Biomedical Engineering; Developmental and Cell Biology* (bioinformatics, computational biology)

Lubomir Bic, Ph.D. University of California, Irvine, *Professor of Computer Science; Electrical Engineering and Computer Science* (design automation and synthesis for embedded systems, VLSI CAD, reconfigurable computing)

Rina Dechter, Ph.D. University of California, Los Angeles, *Professor of Computer Science* (human-computer interaction, computer-supported cooperative work)

Elaheh Bozorgzadeh, Ph.D. University of California, Los Angeles, *Associate Professor of Computer Science; Electrical Engineering and Computer Science* (design automation and synthesis for embedded systems, VLSI CAD, reconfigurable computing)

Michael Carey, Ph.D. University of California, Berkeley, *Donald Bren Professor of Information & Computer Sciences and Professor of Computer Science*

Pai H. Chou, Ph.D. University of Washington, *Professor of Electrical Engineering and Computer Science; Computer Science* (embedded systems, wireless sensor systems, medical devices, real-time systems, hardware/software co-synthesis)

Rina Dechter, Ph.D. University of California, Los Angeles, *Professor of Computer Science*

Brian C. Demsky, Ph.D. Massachusetts Institute of Technology, *Associate Professor of Electrical Engineering and Computer Science; Computer Science* (compiler programming, language software engineering, fault tolerance)

Michael B. Dillencourt, Ph.D. University of Maryland, College Park, *Professor of Computer Science*

Rainer B. Doemer, Ph.D. Dortmund University, *Associate Professor of Electrical Engineering and Computer Science; Computer Science* (system-level design, embedded computer systems, design methodologies, specification and modeling languages)

James P. Dourish, Ph.D. University College London, *Professor of Informatics; Computer Science* (human-computer interaction, computer-supported cooperative work)
Nikil D. Dutt, Ph.D. University of Illinois at Urbana–Champaign, *UCI Chancellor’s Professor of Computer Science; Cognitive Sciences; Electrical Engineering and Computer Science* (embedded systems, computer architecture, electronic design automation, software systems, brain-inspired architectures and computing)

Magda S. El Zarki, Ph.D. Columbia University, *Professor of Computer Science; Electrical Engineering and Computer Science; Informatics* (telecommunications, networks, wireless communication, video transmission)

David A. Eppstein, Ph.D. Columbia University, *UCI Chancellor’s Professor of Computer Science*

Julian Feldman, Ph.D. Carnegie Institute of Technology, *Professor Emeritus of Computer Science*

Charless C. Fowlkes, Ph.D. University of California, Berkeley, *Associate Professor of Computer Science; Cognitive Sciences; Electrical Engineering and Computer Science* (computer vision, machine learning, computational biology)

Michael S. Franz, Ph.D. Swiss Federal Institute of Technology in Zurich, *Professor of Computer Science; Electrical Engineering and Computer Science* (systems software, particularly compilers and virtual machines, trustworthy computing, software engineering)

Daniel H. Frost, M.S. University of California, Irvine, *Senior Lecturer of Computer Science; Informatics* (artificial intelligence, software engineering, computer graphics, teaching of programming)

Jean-Luc Gaudiot, Ph.D. University of California, Los Angeles, *Professor of Electrical Engineering and Computer Science; Computer Science* (parallel processing, computer architecture, processor architecture)

Tony D. Givargis, Ph.D. University of California, Riverside, *Professor of Computer Science* (embedded systems, platform-based system-on-a-chip design, low-power electronics)

Michael T. Goodrich, Ph.D. Purdue University, *UCI Chancellor’s Professor of Computer Science; Electrical Engineering and Computer Science* (computer security, algorithm design, data structures, Internet algorithmics, geometric computing, graphic drawing)

Richard H. Granger, Ph.D. Yale University, *Professor Emeritus of Computer Science*

Ian G. Harris, Ph.D. University of California, San Diego, *Associate Professor of Computer Science; Electrical Engineering and Computer Science* (hardware/software covalidation, manufacturing test)

Wayne B. Hayes, Ph.D. University of Toronto, *Associate Professor of Computer Science*

Dan S. Hirschberg, Ph.D. Princeton University, *Professor of Computer Science; Electrical Engineering and Computer Science* (analyses of algorithms, concrete complexity, data structures, models of computation)

Alexander T. Ihler, Ph.D. Massachusetts Institute of Technology, *Associate Professor of Computer Science*

Sandra S. Irani, Ph.D. University of California, Berkeley, *Professor of Computer Science*

Ramesh Chandra Jain, Ph.D. Indian Institute of Technology Kharagpur, *Donald Bren Professor of Information & Computer Sciences and Professor of Computer Science*

Stanislaw M. Jarecki, Ph.D. Massachusetts Institute of Technology, *Professor of Computer Science*

Scott A. Jordan, Ph.D. University of California, Berkeley, *Professor of Computer Science; Electrical Engineering and Computer Science* (pricing and differentiated services in the Internet, resource allocation in wireless networks, telecommunications policy)

David G. Kay, J.D. Loyola Marymount University, *Senior Lecturer of Informatics; Computer Science* (computer law, computer science education)

Dennis F. Kibler, Ph.D. University of California, Irvine, *Professor Emeritus of Computer Science*

Raymond O. Klefstad, Ph.D. University of California, Irvine, *Lecturer of Computer Science*

Alfred Kobsa, Ph.D. University of Vienna, *Professor of Informatics; Computer Science* (user modeling, human-computer interaction, artificial intelligence, cognitive science, interdisciplinary computer science)

Jeffrey L. Krichmar, Ph.D. George Mason University, *Professor of Cognitive Sciences; Computer Science* (computational neuroscience, robotics)

Fadi J. Kurdahi, Ph.D. University of Southern California, *Director, Center for Embedded Computer Systems and Professor of Electrical Engineering and Computer Science; Computer Science* (VLSI system design, design automation of digital systems)

Richard H. Lathrop, Ph.D. Massachusetts Institute of Technology, *Professor of Computer Science* (modeling structure and function, machine learning, intelligent systems and molecular biology, protein structure/function prediction)
Marco Levorato, Ph.D. University of Padua, Assistant Professor of Computer Science; Electrical Engineering and Computer Science

Chen Li, Ph.D. Stanford University, Professor of Computer Science

Kwei-Jay Lin, Ph.D. University of Maryland, College Park, Professor of Electrical Engineering and Computer Science; Computer Science (real-time systems, distributed systems, service-oriented computing)

Cristina V. Lopes, Ph.D. Northeastern University, Professor of Informatics; Computer Science (programming languages, acoustic communications, operating systems, software engineering)

George S. Lueker, Ph.D. Princeton University, Professor Emeritus of Computer Science

Aditi Majumder, Ph.D. University of North Carolina at Chapel Hill, Professor of Computer Science; Electrical Engineering and Computer Science (novel displays and cameras for computer graphics and visualization, human-computer interaction, applied computer vision)

Athina Markopoulou, Ph.D. Stanford University, Associate Professor of Electrical Engineering and Computer Science; Computer Science (networking—reliability and security, multimedia networking, measurement and control, design and analysis of network protocols and algorithms, internet reliability and security, multimedia streaming, network measurements and control)

Gopi Meenakshisundaram, Ph.D. University of North Carolina at Chapel Hill, Professor of Computer Science; Electrical Engineering and Computer Science (geometry and topology for computer graphics, image-based rendering, object representation, surface reconstruction, collision detection, virtual reality, telepresence)

Sharad Mehrotra, Ph.D. University of Texas at Austin, Professor of Computer Science

Eric D. Mjolsness, Ph.D. California Institute of Technology, Professor of Computer Science; Mathematics (applied mathematics, mathematical biology, modeling languages)

Alexandru Nicolau, Ph.D. Yale University, Department Chair and Professor of Computer Science; Electrical Engineering and Computer Science (architecture, parallel computation, programming languages and compilers)

Donald J. Patterson, Ph.D. University of Washington, Associate Professor of Informatics; Computer Science (ubiquitous computing, pervasive computing, human-computer interaction, artificial intelligence, intelligent context for situated computing)

Richard Pattis, M.S. Stanford University, Senior Lecturer of Computer Science; Informatics (MicroWorlds for teaching programming, debugging, computational tools for non-computer scientists)

Amelia C. Regan, Ph.D. University of Texas at Austin, Professor of Computer Science

Ardalan Amiri Sani, Ph.D. Rice University, Assistant Professor of Computer Science (involves building efficient, high performance, and reliable systems)

Isaac D. Scherson, Ph.D. Weizmann Institute of Science, Professor of Computer Science; Electrical Engineering and Computer Science (parallel computing architectures, massively parallel systems, parallel algorithms, interconnection networks, performance evaluation)

Babak Shahbaba, Ph.D. University of Toronto, Associate Professor of Statistics; Computer Science

Phillip C-Y Sheu, Ph.D. University of California, Berkeley, Professor of Electrical Engineering and Computer Science; Biomedical Engineering; Computer Science (database systems, interactive multimedia systems)

Alice Silverberg, Ph.D. Princeton University, Professor of Mathematics; Computer Science (algebra and number theory)

Patrick J. Smyth, Ph.D. California Institute of Technology, Professor of Computer Science; Statistics

Mark Steyvers, Ph.D. Indiana University, Professor of Cognitive Sciences; Computer Science; Psychology and Social Behavior (higher-order cognition, cognitive neuroscience, computational modeling, collective intelligence)

Alexander W. Thornton, B.S. University of California, Irvine, Lecturer of Computer Science

Gene Y. Tsudik, Ph.D. University of Southern California, UCI Chancellor's Professor of Computer Science

Kojiro Umezaki, M.A. Dartmouth College, Associate Professor of Music; Computer Science

Alexander Veenendaal, Ph.D. University of Illinois at Urbana-Champaign, Professor of Computer Science

Nalini Venkatasubramanian, Ph.D. University of Illinois at Urbana-Champaign, Professor of Computer Science

Richert Wang, Ph.D. University of California, Irvine, Lecturer of Computer Science
Xaohui Xie, Ph.D. Massachusetts Institute of Technology, Associate Professor of Computer Science; Developmental and Cell Biology (computational biology, bioinformatics, genomics, neural computation, machine learning)

Guoqing Xu, Ph.D. Ohio State University, Assistant Professor of Computer Science

Charles S. Zender, Ph.D. University of Colorado Boulder, Professor of Earth System Science; Computer Science

Hong-Kai Zhao, Ph.D. University of California, Los Angeles, Professor of Mathematics; Computer Science (applied and computational mathematics, inverse problems and imaging)

Shuang Zhao, Ph.D. Cornell University, Assistant Professor of Computer Science (computer graphics with a focus on material appearance modeling and physically-based rendering)