SCS Concentrations

The School of Computer Science (SCS) offers concentrations for SCS students in various aspects of computing to provide greater depth to their education. Computer Science majors can substitute an SCS concentration for the minor requirement. Artificial Intelligence and Computational Biology majors can complete an SCS concentration if they wish, but it is not required for these degrees.

Note: At the present time, concentrations are not shown on official transcripts.

SCS Concentrations are currently available to SCS students only and assume these students have taken most/all of the CS core: 15-122/15-150/15-210/15-213/15-251. For SCS students entering in Fall 2018 or later, these students may not pursue a minor within SCS; instead they can pursue the related concentration. For example, instead of pursuing the Software Engineering minor, these SCS students could pursue the new Software Engineering concentration.

Consult the SCS undergraduate concentrations website (https://www.cs.cmu.edu/undergraduate-concentrations/) for information about these concentrations as they are approved. For SCS students, consult with your academic advisor for more information about available concentrations and requirements.

- Algorithms and Complexity
- Computational Biology
- Computer Systems
- Human-Computer Interaction
- Language Technologies (to be approved in late 2019)
- Machine Learning
- Principles of Programming Languages
- Robotics
- Security and Privacy
- Software Engineering

Algorithms and Complexity Concentration

This concentration is available to SCS students only.

Ryan O’Donnell, Concentration Director
Location: GHC 7213

The goal of the Algorithms and Complexity concentration is to give SCS students a deep background in the theory of computation as it relates to algorithms and computational complexity. The expectation is that students who complete this concentration will have the background to pursue topics at the PhD level at any top program in the country. Furthermore we expect the reasoning skills gained as part of this concentration could be a significant help in a wide variety of positions in industry.

The concentration is designed to be reasonably flexible covering a wide area of topics within the area of algorithms and complexity. This includes central topics within the area such as complexity theory, and algorithms, but also includes theory as used in areas such as Computational Geometry, Graph Theory, Cryptography, Machine Learning, Algorithms for Large Data, Error Correcting Codes, and Parallel Algorithms.

Common themes of all courses covered by the concentration are the following:
- Clearly defined formalisms of the subject matter.
- A substantial component involving rigorous mathematical analysis, including proofs.
- Abstracting away from specific applications to a more general context.
- Relating algorithms and/or complexity of computation to a variety of complexity measures such as time, space, communication, or information content.

Any given course does not have to exclusively cover these themes and can, for example, also cover experimental aspects of algorithms, or examples applied to quite specific applications.

Learning Objectives

We do not expect students to have high proficiency in all the examples listed, but to gain at least some proficiency from each category.
- The ability to take a loosely defined problem and clearly pose it as a well defined problem specification.
- The understanding of several advanced algorithms beyond what is covered in the core.
- The appreciation a variety of models for bounding resources, such as information theory, space complexity, parallel complexity, communication complexity, proof complexity, query complexity, and hardness of approximation.
- The ability to understand and apply a variety of advanced algorithmic techniques and proof techniques, such as Lovasz Local Lemma, Johnson Lindenstrauss, Chernoff Bounds, sparsification, expanders, probabilistic method, regret bounds, spectral graph theory, fixed parameter tractability and semi-indefinite programming.
- The ability to recognize flaws in ill-formed proofs.
- The ability to formulate new questions about the field.

Prerequisites

The following courses must be completed before the concentration can be completed:

15-210 Parallel and Sequential Data Structures and Algorithms 12
15-251 Great Ideas in Theoretical Computer Science 12
15-259 Probability and Computing 12
or 21-325 Probability 12
or 36-218 Probability Theory for Computer Scientists
15-451 Algorithm Design and Analysis 12

It is expected that all students will start the concentration after having finished all but 15-451.

Course Requirements

The curriculum consists of one required course and at least three elective courses. The three elective courses must sum to at least 30 units. The elective courses will vary from year to year.

Required:

15-455 Undergraduate Complexity Theory 9

Electives (at least three courses with a total of 30 units or more):

15-354 Computational Discrete Mathematics 12
15-356 Introduction to Cryptography 12
15-456 Computational Geometry 9
15-458 Discrete Differential Geometry 12
15-483 Truth, Justice, and Algorithms 9
21-301 Combinatorics 9
21-484 Graph Theory 9

Special permission required:

47-834 Linear Programming 6
47-835 Graph Theory 6
47-836 Advanced Graph Theory 6

Other graduate-level courses as approved by the concentration director

A student can also use a senior thesis or research-based independent study as one of the elective courses (for 12 units). The topic must be related to algorithms and complexity, and approved by the director. An independent study must be for at least 12 units, and there must be a substation paper writeup on the research topic, and a poster or other presentation.

The choice of available elective courses will be posted prior to registration each semester.

Double Counting

The concentration will require that 3 courses (at least 27 units) are not double counted with any other requirements of any Major, Minor, or other concentration the student is pursuing.

Advising and Management

Courses in the list of electives will be approved by the director on a yearly basis under consultation of the algorithms and complexity group (to help evaluate the relevance of the courses) and the Assistant Dean (to help
flag any logistical issues). Any special requests by a student for counting a course out of the list, will go to the director. The director will also approve any research units.

Students interested in this concentration should contact the concentration director for an initial advising consultation.

Computational Biology Concentration

This concentration is available to SCS students only.

Phillip Compeau, Concentration Director
Location: GHC 7403

Samantha Mudrinich, Concentration Coordinator
Location: GHC 7414

The general goal of the Computational Biology Concentration is to provide foundational coursework in computational biology that will allow undergraduate students in the Carnegie Mellon University School of Computer Science to start building a skillset useful for understanding many of the modern technologies developed by researchers as well as companies in the biotech and biomedical arenas.

This concentration consists of four core courses providing breadth in computational biology across laboratory methods, machine learning, genomics, and modeling of biological systems, as well as one elective that allows students to complete depth coursework in an area of interest, including undergraduate research.

Learning Objectives

Students will, by way of completing this concentration:

• model biological systems at the molecular and cellular levels using a variety of approaches;
• generate their own high throughput molecular biology data in a laboratory setting, and apply computational techniques to analyze the data they generate;
• transform hazy biological problems involving genomic data into well-defined computational problems, design algorithms to solve these problems, and adapt them to biological data;
• explore additional coursework of interest in genomics, biological research automation, biological image analysis, or computational biology research.

This concentration also provides students completing a computational degree other than the major in computational biology (http://www.cbd.cmu.edu/education-bs-in-computational-biology/) with the opportunity to make a transition toward a career in computational biology. We have compiled information on over 250 companies working on computational biology into a unique web resource for students both inside and outside of Carnegie Mellon (http://careers.cbd.cmu.edu/). These companies work on diverse topics from the automation of biological research to drug discovery to wearable medical devices to genetic diagnostics. Increasingly, when we interact with these companies, they want computationally minded candidates with as much knowledge of standard approaches in computational biology as possible.

Prerequisites

Note that not all of the prerequisites below are required to take every course in this concentration (for example, 02-251 does not have any of the pre-requisites below), but these courses are required to complete all of the required coursework and should be completed early within this concentration.

15-122 Principles of Imperative Computation 10
15-151 Mathematical Foundations for Computer Science 10
15-210 Parallel and Sequential Data Structures and Algorithms 12
21-241 Matrices and Linear Transformations 10
36-218 Probability Theory for Computer Scientists 9

Further, the following two courses are not technically required as prerequisites to the courses in this concentration, but they are strongly suggested prerequisites because they provide students with helpful surveys of fundamental topics in biology and computational biology.

02-251 Great Ideas in Computational Biology 12
03-121 Modern Biology 9

Requirements

Five courses in total are required for this concentration. The following four courses are required as part of a central core of coursework; they consist of three computational biology courses as well as an introductory machine learning course, which today is fundamental for even an introductory understanding of the field.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-261</td>
<td>Quantitative Cell and Molecular Biology Laboratory</td>
<td>Var.</td>
</tr>
<tr>
<td>(03-343, Experimental Techniques in Molecular Biology, may be taken if 02-261 is not offered)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>02-510</td>
<td>Computational Genomics</td>
<td>12</td>
</tr>
<tr>
<td>02-512</td>
<td>Computational Methods for Biological Modeling and Simulation</td>
<td>9</td>
</tr>
<tr>
<td>10-315</td>
<td>Introduction to Machine Learning (SCS Majors)</td>
<td>12</td>
</tr>
</tbody>
</table>

In addition to these four courses above, one elective course is required. Any 02-listed (Computational Biology Department) undergraduate course of at least 9 units at the 300-level or above may satisfy this requirement; graduate courses may be applied to this category with permission. The Computational Biology Department is growing quickly, but at the time of writing, the courses that are regularly offered by the department that would satisfy this requirement are the following:

02-317 Algorithms in Nature 9
02-319 Genomics and Epigenetics of the Brain 9
02-425 Computational Methods for Proteogenomics and Metabolomics 9
02-450 Automation of Scientific Research 9
02-499 Independent Study in Computational Biology Var.
02-500 Undergraduate Research in Computational Biology Var.
(03-441/03-541 may be taken if 02-500 is not offered)
02-514 String Algorithms 12
02-515 Advanced Topics in Computational Genomics 12
02-518 Computational Medicine 12

Double Counting

At most two courses can double count with all program requirements for majors, minors and other concentrations being pursued by the student. Courses used as free electives for a major are not considered double counted.

Accordingly, this concentration is expressly closed to majors and additional majors in computational biology.

CS and AI majors completing this concentration are encouraged to double-count 10-315 as well as 02-261 as their lab science course. Suggested prerequisites 03-121 and 02-251 also count as requirements for these degrees (as a Science & Engineering course and C5 Domains course, respectively).

Advising and Management

The day-to-day management of this concentration (including declaration of the concentration, exception requests, overseeing student audits, advising, etc.) is handled by Phillip Compeau, Assistant Department Head for Education in the Computational Biology Department. Administrative support for the concentration is provided by Samantha Mudrinich. Curricular organization and annual review will be managed by the Computational Biology Undergraduate Review Committee.

SCS students interested in this concentration should set up an appointment with Phillip Compeau for a brief interview.

Computer Systems Concentration

This concentration is available to SCS students only.

Brian Railing, Concentration Director and Advisor
Location: GHC 6005

The goal of the Computer Systems concentration is to give students a broad background in the practical understanding of designing and building systems. Students who complete this concentration are expected to be able to both pursue topics at the Ph.D. level at top programs, as well as industry work, either applying these concepts or directly working within the areas of kernel development, compiler improvements, designing distributed systems, etc. et cetera.

The concentration is designed to be flexible in covering the wide area of systems topics. Two courses from the Computer Science major’s “systems” constrained elective are required (List A below). The other courses come
from a larger list of related courses (List B below). A limited amount of research credit can count toward the requirements.

Typically, systems courses include three aspects:

- A systems course educates students about how a class of computer systems works, both at a conceptual level and in practice. This includes the study of concrete problems faced in building a particular class of systems and successful solutions to these problems.
- Systems courses address how properties of modern hardware (e.g., processors, net-works, storage hierarchies) influence the design and implementation of a class of soft-ware systems. This typically includes reasoning about concurrency, and understanding and measuring performance.
- To solidify the key systems organization principles, there is a significant project/system implementation aspect to the course, both to reinforce understanding of how these systems work, and to learn system building skills (i.e., not just programming, but also design, debugging, testing, etc.). The size of the programming tasks is course dependent, but a significant fraction of the course grade (e.g., at least 40%) is derived from project work.

Learning Objectives

Students completing this concentration will be able to demonstrate the following skills and learning:

- Students will be able to design, develop and deploy large computer systems and justify their design decisions.
- Students will synthesize the interaction and tradeoffs between different layers and components in computer systems.
- Students will demonstrate debugging expertise on complex and diverse bugs and issues during software development.
- Students will recognize diverse granularities of parallelism, apply them toward solving problems, and implement solutions that achieve correct execution while accounting for reliability, fault tolerance, performance, security, and scalability.

Prerequisites

All students will start the concentration after having finished 15-213 Introduction to Computer Systems (or its cross-listed equivalents) with a C or better, as 15-213 is a prerequisite either directly or indirectly for all courses in the concentration.

Course Requirements

The curriculum will consist of at least four courses: two courses from List A and at least two elective courses from List B. The courses taken from List A and B must sum to at least 51 units, where each course must be passed with a C or better. The courses in List A will follow the Systems constrained elective list as part of the degree requirements for a B.S. in Computer Science. The elective courses on List B may vary from year to year, with a plan to review these requirements every three years.

List A (select two):

- 15-410 Operating System Design and Implementation 15
- 15-411 Compiler Design 15
- 15-418 Parallel Computer Architecture and Programming 12
- 15-440 Distributed Systems 12
- 15-441 Networking and the Internet 12
- 15-445 Database Systems 12

List B (select at least two):

Anything from List A, plus...

- 15-319 Cloud Computing 12
- 15-330 Introduction to Computer Security 12
- 15-348 Embedded Systems 9
- 15-412 Operating System Practicum Var.
- 15-415 Database Applications 12
- 17-422 Building User-Focused Sensing Systems 12
- 18-341 Logic Design and Verification 12
- 18-349 Introduction to Embedded Systems 12
- 18-447 Introduction to Computer Architecture 12

Special permission required for:

- 15-719 Advanced Cloud Computing 12

Other graduate-level courses in CSD as approved by the concentration director

Students can also apply a senior thesis (or other significant research for credit) in a topic related to Systems, as approved by the concentration advisor, as one of the elective courses for List B. Any significant research credit will include an identifiable output, such as paper or technical report. Any research course counts for at most 12 units, and can be done once.

Double Counting

The concentration will require that 3 courses (at least 27 units) are not double counted with any other requirements of any major, minor, or other concentration.

Advising and Management

The courses on the list of electives will be reviewed yearly by the concentration advisor through consultation with the Systems group (to help evaluate the relevance of the courses) and the Assistant Dean for undergraduate studies (to help flag any logistical issues). Any special requests by a student for counting a course outside of the list will go to the concentration advisor. The concentration advisor will also approve any research units.

Students interested in this concentration should set up an initial advising consultation with Brian Railing.

Human-Computer Interaction Concentration

This concentration is available to SCS students only.

Vincent Aleven, Concentration Coordinator
Location: Newell-Simon Hall 3531
Andrea Gnessin, Concentration Manager
Location: Newell-Simon Hall 3607

In this concentration, students learn techniques, processes, principles, and theory of Human-Computer Interaction (HCI). This interdisciplinary field aims at understanding how interactions with digital technologies and services can augment what humans do. It also aims at understanding what design, prototyping, and evaluation processes lead to innovative digital technologies and services that fulfill human needs. The concentration enhances what is learned in the SCS majors by addressing how digital products and services can be designed and evaluated so they benefit individuals, small groups, organizations, larger networks, and societies.

It is synergistic with SCS majors in that envisioning, designing, and implementing innovative digital interactions benefit from superior technical skill. The concentration consists of 5 courses (2 required courses and 3 electives).

The concentration helps prepare students for jobs as technically-skilled specialists in design and development of interactive systems. The concentration will give students a broader perspective on how technologies impact humans, which may help them move faster into product management positions. It also lays a foundation for graduate study in the field of Human-Computer Interaction.

Learning Objectives

Students will learn skills and methods for:

- Eliciting and understanding human objectives, preferences, and needs through qualitative and quantitative methods for data collection and analysis.
- Generating and imagining possible solutions and design concepts that involve human/technology partnerships.
- Basic visual design, including typography, grids, color and the use of images.
- Design of interactive systems, experiences, and technologies.
- Developing and evaluating interactive prototypes as a way of iteratively refining designs.
- Evaluating interactive technologies to assess and improve their functioning through data-driven redesign, including discount and empirical evaluation methods.

Students will also learn knowledge about:

- Digital technologies, including, possibly, web and mobile platforms, conversational technologies, wearable computing, gadgets, digital fabrication, virtual reality and mixed reality.
- Human psychology, regarding individuals, groups, organizations, societies, and cultures, as it relates to interactions with digital products and services.

SCS Concentrations
Course Requirements

Students in an SCS major wishing to complete a concentration in HCI must complete 5 courses, namely, 2 required courses and 3 electives. The student will be required to get a grade of "C" or better in each course in order for it to count as part of the concentration.

Required courses (2 courses)

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>05-391</td>
<td>Designing Human Centered Software</td>
<td>12</td>
</tr>
<tr>
<td>05-392</td>
<td>Interaction Design Overview</td>
<td>9</td>
</tr>
</tbody>
</table>

Electives (3 courses)

1. At least one of the electives must have strong technical content and must be selected from the following list:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>05-434</td>
<td>Machine Learning in Practice</td>
<td>12</td>
</tr>
<tr>
<td>05-499</td>
<td>Special Topics in HCI</td>
<td>12</td>
</tr>
<tr>
<td>05-839</td>
<td>Interactive Data Science</td>
<td>12</td>
</tr>
<tr>
<td>10-315</td>
<td>Introduction to Machine Learning (SCS Majors)</td>
<td>12</td>
</tr>
<tr>
<td>11-411</td>
<td>Natural Language Processing</td>
<td>12</td>
</tr>
<tr>
<td>15-237</td>
<td>Special Topic: Cross-Platform Mobile Web Apps</td>
<td>12</td>
</tr>
<tr>
<td>15-281</td>
<td>Artificial Intelligence: Representation and Problem Solving</td>
<td>12</td>
</tr>
<tr>
<td>15-365</td>
<td>Experimental Animation</td>
<td>12</td>
</tr>
<tr>
<td>15-398</td>
<td>Practical Data Science</td>
<td>9</td>
</tr>
<tr>
<td>15-462</td>
<td>Computer Graphics</td>
<td>12</td>
</tr>
<tr>
<td>15-464</td>
<td>Technical Animation</td>
<td>12</td>
</tr>
<tr>
<td>15-466</td>
<td>Computer Game Programming</td>
<td>12</td>
</tr>
<tr>
<td>15-494</td>
<td>Cognitive Robotics: The Future of Robot Toys</td>
<td>12</td>
</tr>
<tr>
<td>16-467</td>
<td>Human Robot Interaction</td>
<td>12</td>
</tr>
<tr>
<td>17-422</td>
<td>Building User-Focused Sensing Systems</td>
<td>12</td>
</tr>
<tr>
<td>17-428</td>
<td>Machine Learning and Sensing</td>
<td>12</td>
</tr>
<tr>
<td>17-437</td>
<td>Web Application Development</td>
<td>12</td>
</tr>
<tr>
<td>17-537</td>
<td>Artificial Intelligence Methods for Social Good</td>
<td>9</td>
</tr>
</tbody>
</table>

2. At least one of the electives must have strong design content and must be selected from the following list:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>05-317</td>
<td>Design of Artificial Intelligence Products</td>
<td>12</td>
</tr>
<tr>
<td>05-418</td>
<td>Design Educational Games</td>
<td>12</td>
</tr>
<tr>
<td>05-452</td>
<td>Service Design</td>
<td>12</td>
</tr>
<tr>
<td>05-499</td>
<td>Special Topics in HCI (Game Design Studio)</td>
<td>12</td>
</tr>
<tr>
<td>15-465</td>
<td>Animation Art and Technology</td>
<td>12</td>
</tr>
<tr>
<td>51-327</td>
<td>Design Center: Introduction to Web Design</td>
<td>9</td>
</tr>
</tbody>
</table>

3. The remaining elective must be a course in HCI offered by the Human-Computer Interaction Institute, meaning it has a 05 number, and is included in the pre-approved list of electives maintained on the HCII website.

Students interested in doing research or project work in the field of Human-Computer Interaction can do an independent study with an HCII faculty member. The independent study (05-589) will count as an elective for the HCI concentration.

Double Counting

At most 2 courses can be double counted with any major, minor, or other concentration being pursued by the student.

Advising and Management

Management will fall on the HCII Undergraduate Program Director and the HCII Undergraduate Program Manager. The requirements for the courses will be reviewed annually by the HCII Curriculum Committee, in consultation with the URC.

Students in the HCI concentration will be advised by the HCII Undergraduate Program Director and/or the HCII Undergraduate Program Manager, who also oversee and direct the HCI second major and the HCI minor. They will meet once a semester will each student in the concentration.

SCS Students interested in this concentration should contact the program director for an initial advising consultation.

Machine Learning Concentration

This concentration is available to SCS students only.

Matt Gormley, Concentration Director/Advisor
Location: GHC 8103
Dorothy Holland-Minkley, Concentration Coordinator
Location: GHC 8008
ml-concentration@cs.cmu.edu

Machine learning and statistical methods are increasingly used in many application areas including natural language processing, speech, vision, robotics, and computational biology. The Concentration in Machine Learning allows undergraduates to learn about the core principles of this field. The Concentration requires five courses (two core courses and three electives) from the School of Computer Science (SCS) and the Department of Statistics & Data Science. The electives primarily focus on core machine learning skills that could be broadly applicable to either industry or graduate work. A CS Senior Honors Thesis or two semesters of Senior Research may be used to satisfy part of the electives requirement, which could provide excellent research experience for students interested in pursuing a PhD.

Learning Objectives

Upon completion of this concentration, students should be able to:

- Formulate real-world problems involving data such that they can be solved by machine learning
- Implement and analyze existing learning algorithms
- Employ probability, statistics, calculus, linear algebra, and optimization in order to develop new predictive models or learning methods
- Select and apply an appropriate supervised learning algorithm for problems of different kinds, including classification, regression, structured prediction, clustering, and representation learning
- Describe the formal properties of models and algorithms for learning and explain the practical implications of those results
- Compare and contrast different paradigms for learning

Prerequisites

The following courses are expected to be completed before the Core courses in the ML Concentration:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-122</td>
<td>Principles of Imperative Computation</td>
<td>10</td>
</tr>
<tr>
<td>Calculus:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-315</td>
<td>Introduction to Machine Learning (SCS Majors)</td>
<td>12</td>
</tr>
<tr>
<td>21-120</td>
<td>Differential and Integral Calculus</td>
<td>10</td>
</tr>
<tr>
<td>21-122</td>
<td>Integration and Approximation</td>
<td>10</td>
</tr>
<tr>
<td>Probability and Statistics:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36-218</td>
<td>Probability Theory for Computer Scientists</td>
<td>9</td>
</tr>
<tr>
<td>or 36-225</td>
<td>Introduction to Probability Theory</td>
<td></td>
</tr>
<tr>
<td>or 15-259</td>
<td>Probability and Computing</td>
<td></td>
</tr>
<tr>
<td>or 21-325</td>
<td>Probability</td>
<td></td>
</tr>
<tr>
<td>36-226</td>
<td>Introduction to Statistical Inference</td>
<td>9</td>
</tr>
<tr>
<td>or 36-326</td>
<td>Mathematical Statistics (Honors)</td>
<td></td>
</tr>
</tbody>
</table>

(Students with a B or higher in 36-218 do not need to take 36-226 or 36-326)

Course Requirements

The ML Concentration requires that students complete two core courses and their choice of three elective courses of at least 9 units each. The electives can be through a combination of coursework in Machine Learning and optionally senior research.

Core Courses (24 units):

Students must take two core courses:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-315</td>
<td>Introduction to Machine Learning (SCS Majors)</td>
<td>12</td>
</tr>
<tr>
<td>Plus one of:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10-403</td>
<td>Deep Reinforcement Learning &amp; Control</td>
<td>12</td>
</tr>
</tbody>
</table>
Electives (minimum 33 units):
Students need to take three courses from the following list. Students may substitute one of these courses with one semester of an SCS Senior Honors Thesis or equivalent senior research credit.

10-405 Machine Learning with Large Datasets (Undergraduate) 12
10-417 Intermediate Deep Learning 12
10-418 Machine Learning for Structured Data 12

Advising and Management
The ML Concentration Director will hold advising office hours leading up to registration week each semester. In addition, the ML Concentration Program Coordinator will hold regular office hours to address general questions. All office hours will be detailed on the ML Concentration website.

SCS Students interested in pursuing this concentration should contact Matt Gormley for an initial advising consultation.

Principles of Programming Languages Concentration
This concentration is available to SCS students only.

Robert Harper, Concentration Director and Advisor
Location: GHC 9229

Programming languages play a central role in computer science. All programs are written in a language, and it is obvious that some are better than others, at least for some purposes. The constant demand for new languages reflects the changing demands for constructing reliable and maintainable software systems. Academic research in programming language principles has led to numerous advances in language design, language implementation, and program verification intended to meet these changing expectations through the development of a rigorous theory of programming languages.

Carnegie Mellon is a recognized leader in programming languages, characterized by a strong emphasis on the centrality of type theory, a concretization of ideas in mathematical semantics, programming logics, and programming language design and implementation. The purpose of the PoPL concentration is to teach the comprehensive view of the field that has been developed here over many decades. Type theory teaches how to define a language, and how to show that it is well-defined, free of internal contradictions. It teaches the mathematical foundations for abstraction and modularity, concepts that are fundamental to building maintainable systems. It teaches how to use a rigorous language definition as the basis for building a compiler that correctly implements the definition, and provides the tools necessary to achieve it. It teaches the logical foundations of program development, how to precisely specify the intended behavior of a program, and how to use machine tools to verify that a program meets those expectations. It gives precise meaning to language concepts, relating them to one another, and distinguishing concepts that are often confused or conflated. It teaches how to specify and verify the resource usage of a program (such as its sequential and parallel time and space complexity) without resorting to a model of how it is implemented on a machine; it supports using actual code, rather than pseudo-code, for defining and analyzing algorithms.

The PoPL concentration is of value to a broad range of students. For the practically minded it will provide the foundation for structuring and validating programs, using type systems or more advanced forms of specification. For the theoretically minded it will provide the foundation for understanding the close relationship between specification and programs on one hand and mathematical conjecture and proof on the other. The elegance of the PoPL lies in their unification of these two perspectives: the theory applies directly to the practice, and the practice informs the theory.

Learning Objectives
The PoPL concentration is characterized by a collection of learning outcomes that it seeks to achieve. These may be summarized by the knowledge that students may expect to gain by concentrating in the area. By their choice of electives each student will choose an emphasis within the area; the required courses ensure that this includes at least the first five objectives:

• Specify the concrete and abstract syntax of a programming language, including a precise specification of the binding and scope of declarations.
• Define the static semantics (compile-time constraints) of a programming language using typing judgments, and how to state and prove that it properly defined.
• Define the dynamics semantics (run-time behavior) of a language using operational and denotational methods.
• Verify rigorously that the statics and dynamics of a language are coherent, a property commonly called type safety.
• Understand the propositions-as-types principle, which relates programs to proofs and specifications to theorems, and know how to apply it in language design and program verification.

• Formulate type and assertion languages for specifying the behavior of a program, and to verify that a program satisfies such a specification.
• Specify the cost (sequential and parallel time and space complexity) for a program written in a precisely defined language, and how to verify that a given program meets stated cost bounds.
• Use software tools to verify both the properties of languages and the specifications of programs written in well-defined languages.
• Use the static and dynamic semantics of a language to derive a compiler for it that complies with these definitions, and how to use types and verification tools to ensure compiler correctness.
• Relate a language definition to its implementation, both in terms of the run-time structures required, but also to validate abstract cost measures in an implementation.

Prerequisites
This concentration requires students to complete the following courses before the concentration can be completed:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-122</td>
<td>Principles of Imperative Computation</td>
<td>10</td>
</tr>
<tr>
<td>15-150</td>
<td>Principles of Functional Programming</td>
<td>10</td>
</tr>
<tr>
<td>15-151</td>
<td>Mathematical Foundations for Computer Science</td>
<td>10</td>
</tr>
<tr>
<td>15-210</td>
<td>Parallel and Sequential Data Structures and Algos</td>
<td>12</td>
</tr>
<tr>
<td>15-213</td>
<td>Introduction to Computer Systems</td>
<td>12</td>
</tr>
<tr>
<td>15-251</td>
<td>Great Ideas in Theoretical Computer Science</td>
<td>12</td>
</tr>
</tbody>
</table>

Course Requirements
This concentration requires two courses along with two additional electives.

Required Courses (complete all of the following):

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-312</td>
<td>Foundations of Programming Languages</td>
<td>12</td>
</tr>
<tr>
<td>15-317</td>
<td>Constructive Logic</td>
<td>9</td>
</tr>
</tbody>
</table>

Electives (complete two of the following):

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-314</td>
<td>Programming Language Semantics</td>
<td>12</td>
</tr>
<tr>
<td>15-316</td>
<td>Software Foundations of Security and Privacy</td>
<td>9</td>
</tr>
<tr>
<td>15-414</td>
<td>Bug Catching: Automated Program Verification</td>
<td>9</td>
</tr>
<tr>
<td>15-417</td>
<td>HOT Compilation</td>
<td>12</td>
</tr>
<tr>
<td>15-424</td>
<td>Logical Foundations of Cyber-Physical Systems</td>
<td>12</td>
</tr>
</tbody>
</table>

Any graduate-level Programming Languages course(s), with prior permission of the concentration advisor and the course instructor(s).

Students may use one semester of a senior thesis supervised by a member of the Principles of Programming faculty in the Computer Science Department as a replacement for one of the two electives.

Transfer of credit for courses taken outside of Carnegie Mellon University toward this concentration will not be allowed.

Double Counting
Either 15-312 or 15-317 (but not both) may be double counted towards any major, minor or other concentration being pursued by the student. No other double counting is permitted.

Advising and Management
Participation in this concentration is supervised by the concentration coordinator in cooperation with the students academic advisor, course instructors, and, as appropriate, thesis supervisor. The current coordinator is Robert Harper. Content for this concentration will be reviewed yearly by the Principles of Programming faculty in the Computer Science Department.

Students interested in pursuing this concentration should contact Robert Harper for an initial advising consultation.

Robotics Concentration
This concentration is available to SCS students only.

Howie Choset, Concentration Director/Advisor
Location: NSH 3205

Barbara (‘B.J.’) Fecich, Concentration Administrator
Location: NSH 4121

The SCS Robotics Concentration provides an opportunity for SCS undergraduate students at Carnegie Mellon to learn the principles and practices of robotics through theoretical studies and hands-on experience with robots. Students initially learn the basics of robotics in an introductory robotics overview course. Additional required courses teach control systems and robotic kinematics. Students also choose from a wide selection of electives in mobile systems, machine learning, computer vision, cognition and cognitive science, or computer graphics. Students have a unique opportunity to undertake independent research projects, working under the guidance of Robotics Institute faculty members; this provides an excellent introduction to robotics practice, for those considering industry and research for those considering graduate studies.

Learning Objectives
Students completing this concentration will be able to demonstrate the following skills and learning:

• construct robots which are driven by a microcontroller through several projects, with each project reinforcing the basic principles of: vision, motion planning, mobile mechanisms, kinematics, inverse kinematics, and sensing
• apply feedback control theory to the development of robotic systems, including the principles of classical linear control theory, linear state-space methods, nonlinear systems theory, and elementary control using computer learning techniques
• program a robot arm using the principles of kinematics and dynamics: transformations, forward kinematics, inverse kinematics, differential kinematics (jacobians), manipulability, and the basic equations of motion
• apply related fields of computing to the construction and testing of robotic solutions: machine learning, AI, graphics and computer vision, cognitive science and learning models, cyber-physical and embedded systems
• work effectively in a team include computer and mechanical engineers to solve challenging robotics problems

Prerequisites
The following courses are expected to be completed before the Core courses in the Robotics Concentration:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-122</td>
<td>Principles of Imperative Computation</td>
<td>10</td>
</tr>
<tr>
<td>15-213</td>
<td>Introduction to Computer Systems</td>
<td>12</td>
</tr>
</tbody>
</table>

One year of calculus:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>21-120</td>
<td>Differential and Integral Calculus</td>
<td>10</td>
</tr>
<tr>
<td>21-122</td>
<td>Integration and Approximation</td>
<td>10</td>
</tr>
</tbody>
</table>

Matrix Algebra (one of the following):

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>21-241</td>
<td>Matrices and Linear Transformations</td>
<td>10</td>
</tr>
<tr>
<td>21-242</td>
<td>Matrix Theory</td>
<td>10</td>
</tr>
</tbody>
</table>

Probability (one of the following):

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-259</td>
<td>Probability and Computing</td>
<td>12</td>
</tr>
<tr>
<td>21-325</td>
<td>Probability</td>
<td>9</td>
</tr>
<tr>
<td>36-218</td>
<td>Probability Theory for Computer Scientists</td>
<td>9</td>
</tr>
<tr>
<td>36-225</td>
<td>Introduction to Probability Theory</td>
<td>9</td>
</tr>
</tbody>
</table>

The probability course can be taken concurrently with the concentration requirements. Depending on specific electives chosen, additional prerequisites may be required (e.g. 21-259).

Course Requirements
The Robotics Concentration requires that students complete three core courses and their choice of two elective courses of at least 9 units each. The electives can be chosen from a specific set of stand-alone courses. Students can opt to do an undergraduate research project as one of their electives.

Required core courses (36 units)

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-311</td>
<td>Introduction to Robotics</td>
<td>12</td>
</tr>
<tr>
<td>16-299</td>
<td>Introduction to Feedback Control Systems</td>
<td>12</td>
</tr>
<tr>
<td>16-384</td>
<td>Robot Kinematics and Dynamics</td>
<td>12</td>
</tr>
</tbody>
</table>

Electives (minimum 18 units)

Students must complete 2 electives from the following list of courses for a minimum of 18 units. At least one of the two electives must be from the Robotics Institute (16-xxx). A maximum of 12 units of research (16-597) can be used toward this requirement.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Name</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-264</td>
<td>Humanoids</td>
<td>12</td>
</tr>
<tr>
<td>16-362</td>
<td>Mobile Robot Algorithms Laboratory</td>
<td>12</td>
</tr>
</tbody>
</table>
Security & Privacy Concentration (SCS)

This concentration is available to SCS students only. (ECE students should consult their department for an equivalent concentration.)

Lujo Bauer, Concentration Director and Advisor
Location: CIC 2203

In a world where data breaches and cyber-attacks are ever-present, the need for technologists who have a solid understanding of the principles that underlie strong security and privacy practices is greater than ever.

The Security & Privacy concentration is designed to expose students to the key facets of and concerns about computer security and privacy that drive practice, research, and legislation. On completing the curriculum, students will be well prepared to continue developing their interests in security or privacy through graduate study; to take jobs in security or privacy that will provide further training in applicable areas; and to be informed participants in public and other processes that shape how organizations and society develop to meet new challenges related to computer security or privacy.

Learning Objectives
After completing this concentration, students should:

- Understand how to reason about the adversary in computer systems.
- Be familiar with common security vulnerabilities, from buffer overflows and return oriented programming to cross-site scripting, and widely deployed defenses against these vulnerabilities.
- Be familiar with and understand how to apply the basic concepts in cryptography and secure system design and analysis.
- Understand the key properties of commonly used cryptographic primitives and properties commonly desired of cryptographic protocols.
- Be familiar with current and upcoming research directions in secure system design, software analysis, and cryptography.
- Be familiar with the breadth of concerns and topics relevant to computer security and privacy, ranging from technical topics to ethics, regulation, usability, and economics.
- Be familiar with the key concepts in privacy, ranging from perceptions of privacy to privacy algorithms to regulation and policy.
- Gain a more in-depth understanding of one “context” area: usable security and privacy, or policy.

Prerequisites
Students interested in pursuing this concentration should have the following courses completed before starting the concentration:

<table>
<thead>
<tr>
<th>Course</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-151 Mathematical Foundations for Computer Science</td>
<td>10</td>
</tr>
<tr>
<td>15-213 Introduction to Computer Systems</td>
<td>12</td>
</tr>
<tr>
<td>15-251 Great Ideas in Theoretical Computer Science</td>
<td>12</td>
</tr>
</tbody>
</table>

Curriculum
A distinguishing feature of this field is the ubiquitous need to consider an adversary, and the resulting interplay between attack and defense that routinely advances both theory and practice. In order to understand widely-deployed defensive techniques and secure-by-design approaches, students must also understand the attacks that motivate them and the “adversarial mindset” that leads to new forms of attack. The curriculum is designed around this principle.

Students in the Security & Privacy concentration will take courses that cover the basic principles (Introduction and Basics Course Area), the underlying theory (Theoretical Foundations Course Area), and the practical application (System Design Course Area) of security and privacy. Additionally, they will be required to select a course which covers either usability or policy (Context Course Area). Finally, students will have the opportunity to dive deep on a particular security & privacy topic by completing an elective of their choosing (Depth Course Area).

Requirements (5 courses, minimum 48 units):

<table>
<thead>
<tr>
<th>Course</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-316 Software Foundations of Security and Privacy</td>
<td>9</td>
</tr>
<tr>
<td>18-335 Secure Software Systems</td>
<td>12</td>
</tr>
<tr>
<td>17-334 Usable Privacy and Security</td>
<td>9</td>
</tr>
<tr>
<td>or one of:</td>
<td></td>
</tr>
<tr>
<td>17-333 Privacy Policy, Law, and Technology</td>
<td>9</td>
</tr>
<tr>
<td>Other graduate-level courses as approved by the concentration director</td>
<td></td>
</tr>
</tbody>
</table>

Depth course (complete one option below):

<table>
<thead>
<tr>
<th>Course</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete an elective course or at least 9 units of independent study in the security or privacy area. Consult with the concentration coordinator for elective options.</td>
<td></td>
</tr>
<tr>
<td>Complete five, rather than four, courses from the list above to satisfy the requirements described above (this might be achieved by taking both a policy and a usability course, or taking the two-course foundations alternative).</td>
<td>9-12</td>
</tr>
</tbody>
</table>

Anti-requisites
When two (or more) courses overlap significantly in the material they cover, only one can count toward the security and privacy concentration. An example pair is 15-316 Software Foundations of Security and Privacy and 18-335 Secure Software Systems. Other such anti-requisites may occur; please consult the concentration director when scheduling courses.

Excluded Courses
Some security and privacy courses may not be counted towards concentration requirements. These courses all serve specific important different purposes, but do not fit into the concentration as currently...
required courses (complete all of the following):

- internship.
- workshop in which the student reflects upon and integrates the lessons of
  the five courses consists of a 6-credit course that serves as a writing
  course.

Overall, the concentration requires five (5) courses and an internship. One
student can show significant development experience.

Students can have the internship counted towards the concentration. For other
excluded courses, please consult the concentration director.

**Double Counting**

Only two of the courses that are counted toward concentration
requirements can also be counted towards core course requirements of
other majors and minors being pursued by the student.

**Advising and Management**

The concentration is open to all undergraduates in the School of Computer
Science. There is no formal admissions process. Students intending to
pursue the concentration should contact the concentration coordinator to
register their intention. Curriculum for the concentration will be reviewed
by the coordinator annually in consultation with faculty in supporting
departments.

**Software Engineering Concentration**

This concentration is available to SCS students only.

Michael Hilton, Concentration Director and Advisor
Location: Wean Hall 5122

Effectively building modern software systems at scale requires not just
programming skills, but also engineering skills. These skills include the
ability to interact effectively with customers to gather the requirements
for a system in a precise way; to develop a design that resolves competing
quality attributes; to make tradeoffs among schedule, cost, features, and
quality to maximize value to stakeholders; to work effectively with other
engineers; and to assure the quality of the delivered software system.
We hear regularly from industry that these skills are crucial to them, and
that they are interested in students with a strong software engineering
background.

**Learning Objectives**

The software engineering concentration is designed to teach the
fundamental tools, techniques, and processes of software engineering.
Through internships and a mentored project experience, students gain
an understanding of the issues of scale and complexity that motivate
software engineering tools and techniques. The core curriculum includes
material both on engineering the software product and on the process,
teamwork, and management skills that are essential to successful
engineering. Graduates of the program should have the technical, process,
and teamwork skills to be immediately productive in a mature engineering
organization.

**Prerequisites**

SCS students should have completed the following courses before
completing this concentration:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-122</td>
<td>Principles of Imperative Computation</td>
<td>10</td>
</tr>
<tr>
<td>15-151</td>
<td>Mathematical Foundations for Computer Science</td>
<td>10</td>
</tr>
<tr>
<td>15-213</td>
<td>Introduction to Computer Systems</td>
<td>12</td>
</tr>
</tbody>
</table>

In addition, students must ensure they have larger-scale software
development experience which can be gained by taking the following
course:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>17-214</td>
<td>Principles of Software Construction: Objects, Design, and Concurrency</td>
<td>12</td>
</tr>
</tbody>
</table>

Students can have the 17-214 prerequisite waived by the director if the
student can show significant development experience.

**Course Requirements**

Overall, the concentration requires five (5) courses and an internship. One
of the five courses consists of a 6-credit course that serves as a writing
workshop in which the student reflects upon and integrates the lessons of
the internship.

**Required Courses (complete all of the following):**

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>17-313</td>
<td>Foundations of Software Engineering</td>
<td>12</td>
</tr>
<tr>
<td>17-413</td>
<td>Software Engineering Practicum</td>
<td>12</td>
</tr>
</tbody>
</table>

**Electives (complete one from each category):**

A domain-independent course focused on technical software engineering
material:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-414</td>
<td>Bug Catching: Automated Program Verification</td>
<td>9</td>
</tr>
<tr>
<td>17-355</td>
<td>Program Analysis</td>
<td>12</td>
</tr>
<tr>
<td>17-356</td>
<td>Software Engineering for Startups</td>
<td>12</td>
</tr>
<tr>
<td>17-651</td>
<td>Models of Software Systems</td>
<td>12</td>
</tr>
<tr>
<td>17-652</td>
<td>Methods: Deciding What to Design</td>
<td>12</td>
</tr>
<tr>
<td>17-653</td>
<td>Managing Software Development</td>
<td>12</td>
</tr>
<tr>
<td>17-654</td>
<td>Analysis of Software Artifacts</td>
<td>12</td>
</tr>
<tr>
<td>17-655</td>
<td>Architectures for Software Systems</td>
<td>12</td>
</tr>
</tbody>
</table>

Other courses, with prior approval from the Director of the
Software Engineering Program:

A course that explores computer science problems related to existing and
emerging technologies and their associated social, political, legal, business,
and organizational contexts:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-390</td>
<td>Entrepreneurship for Computer Science</td>
<td>9</td>
</tr>
<tr>
<td>17-200</td>
<td>Ethics and Policy Issues in Computing</td>
<td>9</td>
</tr>
<tr>
<td>17-331</td>
<td>Information Security, Privacy, and Policy</td>
<td>12</td>
</tr>
<tr>
<td>17-333</td>
<td>Privacy Policy, Law, and Technology</td>
<td>9</td>
</tr>
<tr>
<td>17-334</td>
<td>Usable Privacy and Security</td>
<td>9</td>
</tr>
<tr>
<td>17-562</td>
<td>Law of Computer Technology</td>
<td>9</td>
</tr>
<tr>
<td>17-801</td>
<td>Dynamic Network Analysis</td>
<td>12</td>
</tr>
</tbody>
</table>
| 17-821      | Computational Modeling of Complex Socio-
  Technical Systems                          | 12      |
| 19-402      | Telecommunications Technology and Policy for
  the Internet Age                           | 12      |
| 19-403      | Policies of Wireless Systems                | 12      |
| 70-311      | Organizational Behavior                     | 9       |
| 70-415      | Introduction to Entrepreneurship            | 9       |
| 70-421      | Entrepreneurship for Computer Scientists    | 9       |
| 70-471      | Supply Chain Management                     | 9       |

**Internship and Reflection**

The concentration requires a software engineering internship of a minimum
of 8 full-time weeks in an industrial setting (i.e., integrated into a team and
exposed to industry pressures). The student may work in development,
management, quality assurance, or other relevant positions. Students
should confirm with the director that an internship is appropriate, but
internships that fulfill the criteria will be accepted after the fact. Students
must further complete:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>17-415</td>
<td>Software Engineering Reflection</td>
<td>6</td>
</tr>
</tbody>
</table>

**Double Counting**

No more than two of the courses used to fulfill the concentration
requirements may be counted towards any other degree or concentration.
This rule does not apply to 17-214 (a prerequisite for the concentration) or
courses counted for general education requirements.

**Advising and Management**

The concentration coordinator, Michael Hilton, is responsible for academic
advising, handling exceptions and updating the curriculum each year, in
consultation with faculty in the Institute for Software Research.

Students who are interested in pursuing this concentration should contact
Michael Hilton for an initial advising consultation.