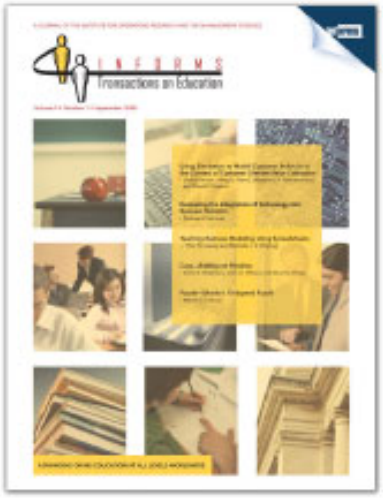

INFORMS Transactions on Education

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Recommendations for an Undergraduate Curriculum at the Interface of Operations Research and Computer Science

Jill R. Hardin, Allen Holder, J. Christopher Beck, Kevin Furman, Arthur Hanna, David Rader, Cesar Rego,

To cite this article:
Jill R. Hardin, Allen Holder, J. Christopher Beck, Kevin Furman, Arthur Hanna, David Rader, Cesar Rego, (2012) Recommendations for an Undergraduate Curriculum at the Interface of Operations Research and Computer Science. INFORMS Transactions on Education 12(3):117-123. http://dx.doi.org/10.1287/ited.1110.0080

Vol. 12, No. 3, May 2012, pp. 117–123
ISSN 1532-0545 (online)
INFORMS Transactions on Education
http://dx.doi.org/10.1287/ited.1110.0080

Recommendations for an Undergraduate Curriculum at the Interface of Operations Research and Computer Science

Jill R. Hardin
Virginia Commonwealth University, Richmond, Virginia 23284, jhwilson3@vcu.edu
Allen Holder
Rose-Hulman Institute of Technology, Terre Haute, Indiana 47803, holder@rose-hulman.edu
J. Christopher Beck
University of Toronto, Toronto, Ontario M5S 3G8, Canada, jcb@mie.utoronto.ca
Kevin Furman
ExxonMobil Upstream Research Company, Houston, Texas 77098, kevin.c.furman@exxonmobil.com
Arthur Hanna
St. Mary's University, San Antonio, Texas 78228, ahanna@stmarytx.edu
David Rader
Rose-Hulman Institute of Technology, Terre Haute, Indiana 47803, david.rader@rose-hulman.edu
Cesar Rego
University of Mississippi, Oxford, Mississippi 38655, crego@bus.olemiss.edu

In March 2007, the INFORMS Computing Society formed the ICS Education Committee, charged with outlining an undergraduate curriculum that would prepare students for graduate study and/or industrial work at the interface of operations research and computer science. The result is a set of essential and recommended skills that undergraduate students should seek to acquire in order to be successful in such endeavors. This article presents the findings of the committee.

Key words: curriculum development; OR/CS interface; undergraduate education; interdisciplinary teaching
History: Received April 2011; accepted August 2011.

1. Introduction

Operations research (OR) has been closely tied to computer science (CS) since the emergence of computer science in the 1960s as a discipline in its own right. Recommendations made in 1965 by the Association for Computing Machinery (ACM) Curriculum Committee on Computer Science included an elective course in mathematical optimization (ACM 1965), and early computer centers commonly offered courses in operations research (Gupta 2007). In 1968, the ACM committee revised these recommendations, eliminating mathematical optimization from the list of recommended electives (ACM 1968). Somewhere along the way, the academy has separated the two disciplines, yet operations research and computer science continue to share substantial common ground. For example, in the 2010–2011 edition of its Occupational Outlook Handbook (U.S. Department of Labor 2010), the Bureau of Labor Statistics indicates that individuals with graduate degrees in both operations research and computer science are attractive to employers seeking operations research analysts; it also lists computer systems design firms as a major employer of operations research analysts.

The INFORMS Computing Society (ICS) is the subdivision of INFORMS that champions the interface of operations research and computing, emphasizing issues such as the impact of computing and other technological developments on the study and practice of OR. In his recollection of the beginnings of the ICS, Harvey Greenberg writes of his belief that the mission of the ICS is "to articulate and lead the development

of interfaces between operations research and computer science" (Greenberg 2006). As testament to its relevance, the ICS boasts approximately 450 members, and the Society's sponsored cluster at the 2010 INFORMS Annual Meeting provided over 50 sessions addressing the OR/CS interface. In addition to supporting research at the interface, the ICS endeavors to assist the research community in education and training through its Leading Edge Tutorials (ICS 2010) and the Mathematical Programming Glossary (Holder 2010). With the formation of the Education Committee, the ICS aspired to provide guidance both to students seeking to pursue study or practice at the OR/CS interface and to the educators who seek to prepare them.

The ICS Education Committee (hereafter referred to simply as "the Committee") was charged in March of 2007 with outlining an undergraduate curriculum that would prepare students for graduate study and/or industrial work at the OR/CS interface. The authors of this article constituted the committee charged with drafting the recommendations presented here. The Committee members represent diversity in departmental affiliation and in institutional size and mission. The Committee was also privileged to have the counsel of two advisers: Harvey Greenberg (University of Colorado Denver) and Ariela Sofer (George Mason University). The Committee's intent was to initiate a curricular discussion within the ICS that would over time distill into a generally agreed-upon curriculum. This article describes their work and the recommendations that followed, which were approved by a vote of the members of ICS in November 2009. The approved recommendations are presented below, with minor editorial changes for clarity.

2. Purpose and Approach

We base our recommendations on the premise that graduate studies and industrial work at the OR/CS interface require an introductory graduate curriculum in OR/CS. The graduate programs listed in §4.2 commonly state that entering students should be prepared for coursework in mathematical programming (often linear programming), probability and statistics, and simulation. The committee considers these topics as being demonstratively foundational to OR/CS, and hence they are necessary educational components of anyone seeking to continue graduate studies and/or enter the industrial workforce in OR/CS. Furthermore, because these topics are typically part of graduate coursework, we do not consider them to be necessary for an undergraduate who desires to continue with an education and/or a career in OR/CS. Indeed, we note that there are only a few undergraduate programs that would offer such coursework at the undergraduate level, and we expect that most students will have initially studied a related field like business, computer science, industrial engineering, systems engineering, or mathematics. Although students studying these areas will naturally overlap some of the suggested curriculum in §3, the curriculum of this report can be used by any student pursuing a graduate education and/or an industrial career in OR/CS.

The Committee does not aim to make recommendations for departments and programs. Although we hope that these recommendations will be considered when advising students and designing curricula, we respect the autonomy of individual departments, programs, and institutions. As noted above, our primary aim is to help a student navigate his or her way to being well prepared for graduate and/or professional work in OR/CS.

Given the diversity within operations research and its varying institutional characteristics, the Committee decided to organize the curriculum with a set of skills instead of a list of courses. The Committee further determined that the set would be separated into three categories:

- foundational skills: those essential for future work;
- core skills: those important for success and that are often associated with introductory graduate work;
- recommended skills: more-advanced skills useful to future work.

The interpretation of this trichotomy is that students should have all the foundational skills, the majority of the core skills, and as many of the recommended skills as possible. Throughout the skill listings we use the following terms to emphasize the depth to which a subject should be known:

- competency/fluency: lucid control of the topic.
- experience: the central theme of a course.
- familiarity: a cursory introduction to the topic.

In addition to the skill listings, the Committee made general suggestions for coursework. These recommendations are not intended to steer curricula design, but rather are intended to guide students in a variety of programs. The general number of courses equates to an undergraduate minor in both mathematics and computer science, but again, such statements vary from school to school.

To develop the foundational skills, the Committee members individually developed lists, which were discussed via conference calls and e-mail discussions. This led to a dichotomy of skills into those that were foundational and those that were not. The Committee's initial recommendations were presented at the annual INFORMS meeting in 2007, upon which the Committee decided it had neglected essential competencies in computer science. Chris Beck and Arthur Hanna were asked to join the Committee at that time to bolster our expertise in this regard. The discussion

following the 2007 INFORMS meeting also led to the important discussion of defining the OR/CS interface. A questionnaire was designed and sent to several leading figures within the ICS. The information gained showed that the definition of the OR/CS interface is somewhat nebulous; it has changed over time as computational availability has increased, and it is influenced to some degree by the viewpoints of individual OR and CS professionals. After a series of discussions, we decided to move ahead based on the premise stated above—namely, that students need to be prepared for a typical introductory graduate curriculum in OR/CS. The resulting curricular recommendations are presented in §3.

The Committee tasked itself with the chore of placing our recommendations within a few supportive contexts. Investigations were undertaken of industrial job postings, graduate admission requirements, and curricular recommendations developed by related professional organizations (e.g., ACM, SIAM, etc.). Individual summaries of these investigations are included in §4.

3. The Curriculum

The recommendations presented below are summarized in Table 1.

3.1. Foundational Skills

These skills are expected of any undergraduate wanting to pursue graduate study and/or industrial work in OR/CS.

Mathematics

- Competency/fluency with:
  - — Functions and relations
  - — Differentiation
  - — Integration
  - — Multivariate calculus
  - — Eigenvectors and eigenvalues
  - — Sets and set operations
  - — Logical expressions
  - — Systems of linear equations
  - — Introductory probability
  - — Random variables, expected value, variance
  - — Sampling distributions, central limit theorem
  - — Hypothesis testing of means and sample proportions
  - — Confidence intervals for means and sample proportions
  - — Introduction to simple linear regression
- Experience with:
  - — Graph theory
  - — Difference between computed and analytical solutions
  - — Error due to numerical approximation
  - — Mathematical modeling
  - — Solution methods and software to solve models
  - — Analyzing, interpreting, and communicating solutions
  - — Determining the appropriateness of exact versus heuristic solution methods

Computer Science

- Competency/fluency with:
  - — Compiled computer programming language
  - — Data modeling
  - — Common data structures
- Experience with:
  - — Contemporary operating systems
  - — Scripting languages
- Familiarity with:
  - — General scientific software
  - — Spreadsheet and other office-type packages
  - — Complexity analysis
  - — Software engineering

Technical Writing

- Competency/fluency with:
  - — Organizing, supporting, and authoring convincing arguments

3.1.1. Explanation of the Foundational Skills. Operations research and computer science use the tools of mathematics to solve and analyze problems, and students who lack a lucid appreciation for mathematics will be limited in their ability to understand and explore the OR/CS interface. With this in mind, the foundational skills begin with a list of essential mathematical skills, each of which is a supportive pillar of OR/CS. The list is not intended to be exhaustive, indeed, if all serviceable areas of mathematics were listed, then the entirety of mathematics would appear. Instead, these mathematical skills are considered to be a core from which a student can progress to a more advanced study.

The Calculus is one of the most impressive accomplishments of humankind, and the primary topics of differentiation, integration, and convergence are paramount to OR/CS. The totality of examples is too immense to state succinctly, but without differentiation one could not model a differential equation, could not capture the essence of rate, and could not state the standard optimality conditions; without integration, one could not define a continuous probability, could not analytically solve a differential equation, and could not calculate simple statistics; and without convergence one could not analyze algorithms. Beyond these and other applications within OR/CS, the study of calculus is typically a student's first step toward mathematical rigor and maturity, both of which are required to understand and apply the methods of OR and CS.

Table 1 Summary of Recommendations

Competency/fluency Experience Familiarity	Mathematics	Computer science	Operations research *Technical writing*
Foundational skills	Functions and relations Differentiation Integration Multivariate calculus Eigenvectors and eigenvalues Sets and set operations Logical expressions Systems of linear equations Introductory probability Random variables, expected value, variance Sampling distributions, central limit theorem Hypothesis testing of means, sample proportions Confidence intervals for means, sample proportions Introduction to simple linear regression	Compiled computer programming language Data modeling Common data structures	*Organizing, supporting, and authoring convincing arguments*
	Graph theory Difference between computed and analytical solutions Error due to numerical approximation Mathematical modeling Solution methods and software to solve models Analyzing, interpreting and communicating solutions Determining appropriateness of exact vs. heuristic solution methods	Contemporary operating systems Scripting languages	
		General scientific software Spreadsheet and other office-type packages Complexity analysis Software engineering	
Core skills	Probability-based statistics Numerical analysis Real analysis Graph theory Mathematical proofs	Object-oriented programming Systems analysis	Linear programming (Mixed) Integer programming Operations research modeling Stochastic modeling and queuing theory Simulation
			The Simplex algorithm Branch-and-bound algorithms Gradient-based algorithms Heuristics
		Parallel computing	OR Applications
Recommended skills	Advanced linear algebra (factorizations, matrix norms, stability analysis)	Computability Complexity theory (P vs. NP, etc.,) Advanced data structures Artificial intelligence Metaheuristics Constraint programming	Advanced queuing theory Polyhedral analysis and cutting planes Industrial and systems engineering applications (supply chains, scheduling, logistics, production, etc.)

Topics in linear algebra and discrete mathematics are similarly crucial to the development of OR/CS. In particular, the ability to solve and analyze a system of equations or inequalities, together with the geometric insight provided by this analysis, motivates many topics in OR. In addition to the basic logic that underlies much of discrete mathematics, the subdisciplines of graph theory and combinatorics provide methods to define relationships and to model and exploit structure, all of which are important tools in OR and CS.

A taxonomy of OR methods would likely split the discipline into deterministic and stochastic realms, and from this perspective, an introduction to probability and statistics represents one of the two major threads in OR. Moreover, the ability to present and analyze data statistically provides a student with intellectual dexterity as she or he confronts the inherent randomness of many applications. As such, a thorough study of introductory probability and statistics is necessary for any student working within OR/CS.

An integral part of an undergraduate education should be to use mathematics to model, solve, and analyze phenomena. We do not stress any particular

model types at the undergraduate level, but rather, the committee supports the general educational goal of abstracting a situation for mathematical study. Beyond modeling, students should be aware of how to solve a model and how to use it to analyze the phenomena. We encourage that this discussion also include the qualitative awareness of the difference between analytic solutions and computed solutions, which are subject to round-off errors and to heuristic approximations. We are not suggesting a full course in numerical analysis at this level.

A student at the OR/CS interface needs basic computing skills, which includes the ability to work with multiple operating systems, compiled languages, and scripting. Although MS Windows is common, the ability to work in a UNIX/Linux environment is valuable. Moreover, the ability to use a few scientific packages is expected. Students should be aware of run-time analysis and have fundamental software engineering skills. The Committee is not suggesting a full course in complexity analysis or a full course in software engineering, although a full course would certainly suffice. The Committee is instead recommending that students have experience with how the selection of algorithm, algorithmic parameters, data structures, and model alter the speed and quality of the solution procedure. They should also have an awareness of the tasks associated with building a large, complex software system.

3.2. Core Skills

These skills are generally expected of a student in preparation for OR/CS and are commonly available at the undergraduate level. The expectation is that some proficiency with each skill is required to pursue being a practitioner or graduate student at the OR/CS interface. Students should acquire as many of these skills as possible.

Mathematics

- Competency/fluency with:
  - — Probability-based statistics
  - — Numerical analysis
  - — Real analysis
  - — Graph theory
  - — Mathematical proofs

Computer Science

- Competency/fluency with:
  - — Object-oriented programming
  - — Systems analysis
- Familiarity with:
  - — Parallel computing

Operations Research

- Competency/fluency with:
  - — Linear programming
  - — (Mixed) integer programming
  - — Operations research modeling
  - — Stochastic modeling and queuing theory
  - — Simulation
- Experience with:
  - — The simplex algorithm
  - — Branch-and-bound algorithms
  - — Gradient-based algorithms
  - — Heuristics
- Familiarity with:
  - — OR applications

3.2.1. Explanation of the Core Skills. In most cases, the mathematical skills are continuations of the foundational skills. The same is largely true of the computer science skills. The Committee emphasizes the need for a student to be a competent computer programmer who is capable of working in one or more modern object-oriented programming languages (C++, C#, Java) and one or more scripting languages because of the ubiquity of these development platforms in current-day computing environments. Many interesting and important problems have best-known algorithms with exponential time complexities, so the ability to write concurrent or parallel programs to try to ameliorate the run-time problems is necessary. Large OR problems require large-scale computing solutions that necessarily contain many modules with many interdependent complex data relationships. Software engineering and system analysis skills are absolutely a requisite for any such large-scale software development.

The advanced undergraduate who has achieved the foundational skills is prepared to begin taking operations research coursework. Linear programming is often the gateway to mathematical programming, and its use is pervasive throughout operations research practice and study. Mixed-integer programming introduces students to the area of discrete optimization, which is similarly at the bedrock of OR. Stochastic modeling and queuing theory provide a first step into the realm of stochastic optimization, in addition to being valuable in their own right. The ability to use and understand the basic algorithmic methods is necessary to make use of a model. In general, exposure to various problems in areas such as supply chain management, scheduling, logistics, and others provides a broad exposure to the applications of OR.

3.3. Recommended Skills

Each of these skills is important when working at the OR/CS interface and is often part of a graduate education, but they are not always found at the undergraduate level. If a student has an opportunity to learn these topics and is adequately prepared for such study, it is recommended that he or she do so.

Mathematics and Operations Research

- Familiarity with:
  - — Advanced linear algebra (factorizations, matrix norms, stability analysis)
  - — Advanced queuing theory
  - — Polyhedral analysis and cutting planes
  - — Industrial and systems engineering applications (supply chains, scheduling, logistics, production, etc.)

Computer Science

- Familiarity with:
  - — Computability
  - — Complexity theory (P vs. NP, etc . . .)
  - — Advanced data structures
  - — Artificial intelligence
  - — Metaheuristics
  - — Constraint programming

3.4. Standard Degree Tracks

Although the Committee intentionally defined its recommendation in terms of skills in order to avoid the cumbersome variety in programs and coursework, there are sufficient commonalities that we make the following general recommendation for an undergraduate student who desires to pursue OR/CS as a career.

- Major or minor in either mathematics or computer science
- Major in mathematics, computer science, industrial engineering, systems engineering, or management science
- Mathematics coursework:
  - — Calculus through vector calculus (3–4 courses)
  - — Probability and statistics, preferably probability-based statistics (2 courses)
  - — Mathematical modeling (1 course)
  - — Linear algebra (1–2 courses)
  - — Operations research, linear programming, optimization (1–2 courses)
- Computer science coursework:
  - — Introduction to computer programming and object-oriented programming (2 courses)
  - — Data structures (1 course)
  - — Algorithm analysis (1 course)
  - — Discrete mathematics (1 course)
  - — Numerical methods (1 course)
  - — Data base design/data modeling (1 course)
  - — Software engineering and systems analysis (1 course)
  - — Introduction to artificial intelligence (1 course)

4. Review of Industrial and Academic Requirements

The Committee undertook the task of reviewing industrial skills expressed through job advertisements and graduate entrance requirements. Although the committee did not see its role as entirely descriptive of the status quo, we did find it important to position our recommendations against the backdrop of what both industry and graduate schools expect. In both cases, the committee is confident that our curricular recommendations are solid preparation to either enter graduate study or the work force.

4.1. Industrial Qualifications

The foundational skills generally support a nonacademic career as observed by reviewing three months of the 2007 job descriptions in *OR/MS Today*. Beyond technical abilities, nearly all postings required strong communication skills. Most positions request a degree in computer science, industrial engineering, mathematics or operations research; or a related area such as econometrics, finance, physics, or statistics. All but one position included a proficiency with software; among those advertisements, the following software recommendations were explicitly stated:

- 62% listed experience with a programming language such as C/C++/C#, Java, Visual Basic, or Perl
- 38% desired proficiency with mathematical programming software such as CPLEX, GAMS, AMPL, MPL, or SAILS
- 46% stated a need for experience with statistical software like SAS or S-Plus
- 58% wanted proficiency with database software such as Access, SQL, Excel, or Oracle.

Detailed knowledge of a targeted subdiscipline is often required. For example, instead of saying that a candidate should have a knowledge of mathematical programming, the desired skills listed were instead focused on topics like dynamic programming or column generation, and instead of graph theory or network optimization, advertisements asked for an understanding of social network theory or network flows. This level of detail is difficult to support within the general guidelines of the ICS Education Committee because the union of all possible job skills would include the entirety of OR and CS. However, the breadth of such expertise is significant, and this review highlights the fact that there are quality careers for individuals who are broadly trained in OR/CS, who have programming and software skills, who have specific knowledge of a subdiscipline, and who are good communicators. Anecdotal evidence from the members of the ICS Education Committee supports this claim.

4.2. Graduate School Requirements

Using the list of 141 U.S. educational programs found on the INFORMS website (ORMS 2010), members of the Committee conducted a review of the educational requirements for graduate programs at the OR/CS

interface and other OR-related fields. These programs came from business schools, IE programs, mathematical sciences programs, and independent OR or OR-statistics groups. In addition, the Committee undertook a review of the graduation requirements for various undergraduate programs in OR-related fields. Roughly 50% of the education programs listed were examined for this study. When examining the requirements for graduate program admittance, programs in mathematical sciences departments were not included because their prerequisites would already include material we are considering.

Many (but not all) of the undergraduate programs surveyed had similar requirements or recommendations, regardless of whether they were in business, engineering, or mathematics departments. They required the following courses for graduation:

- Calculus through multivariate calculus
- Linear algebra
- One or two computer programming courses

Many undergraduate programs also required a course in economics and a course in probability and statistics (calculus-based).

Not all graduate programs listed prerequisite coursework for incoming students. However, those that did mimicked the above list in recommended coursework, including courses in probability and statistics. Some programs also required a course in real analysis (at least at an undergraduate level) so that the student is exposed to the notion of a mathematical proof; this requirement, when found, primarily occurred for incoming Ph.D. students. A few programs listed preference for courses in introductory operations research and engineering economics.

More information on the programs surveyed can be found in the full report (ICS 2009).

4.3. Review of Curriculum Recommendations from Other Professional Organizations

Curriculum recommendations from the Mathematical Association of America (MAA) (Barker et al. 2004) and the Society for Industrial and Applied Mathematics (SIAM) (SIAM 2006) were examined, as well as a joint report produced by the Association for Computing Machinery (ACM), the Association for Information Systems (AIS), and the IEEE Computer Society (IEEECS) (Joint Task Force for Computing Curricula 2005) (JTF). Most of the reports examined focus on making recommendations to programs and departments about development of specific degree programs, which as noted above is somewhat different from our aim. Documents available from MAA and SIAM barely mention operations research, whereas documents produced jointly by ACM, AIS, and IEEECS mention operations research as a valuable elective course. No report outlining curriculum recommendations was available on the website for the American Mathematical Society (AMS).

5. Conclusion

These recommendations are intended to guide undergraduate students in preparation for graduate work or professional careers at the OR/CS interface. Although these recommendations are not intended to be used in the design of degree programs, departments may find them helpful as they consider curricular issues, provided that work at the OR/CS interface is a priority. Students at all levels may wish to consider these recommendations if they wish to prepare for work at the OR/CS interface. The recommendations presented here are subject to revision as The Committee continues its work.

Acknowledgments

The Committee would like to thank Harvey Greenberg and Ariela Sofer for their advice in developing these recommendations. They would also like to thank past ICS presidents John Chinneck and Robin Lougee for their vision and their unwavering support of this process.

References

ACM Curriculum Committee on Computer Science. 1965. An undergraduate program in computer science—Preliminary recommendations. *Comm. ACM* **8**(9) 543–552.

ACM Curriculum Committee on Computer Science. 1968. Curriculum 68: Recommendations for the undergraduate program in computer science. *Comm. ACM* **11**(3) 151–197.

Barker, W., D. Bressoud, S. Epp, S. Ganter, B. Haver, H. Pollatsek. 2004. *Undergraduate Programs and Courses in the Mathematical Sciences: CUPM Curriculum Guide* 2004. Mathematical Association of America. Retrieved December 1, 2010, http://www.maa.org/cupm/cupm2004.pdf.

Greenberg, H. J. 2006. A personal history of ICS, 1974–1998. Retrieved August 23, 2010, http://www.informs.org/content/download/159575/1646735/file/GreenbergHistory.pdf.

Gupta, G. K. 2007. Computer science curriculum developments in the 1960s. *IEEE Ann. Hist. Comput.* **29**(2) 40–54.

Holder, A., ed. 2010. *Mathematical Programming Glossary*. INFORMS Computing Society, originally authored by Harvey J. Greenberg, 1999–2006. Retrieved November 5, 2010, http://glossary.computing.society.informs.org/.

INFORMS Computing Society. 2009. ICS educational report. Retrieved November 5, 2010, http://computing.society.informs.org/pdf/ICSed.pdf.

INFORMS Computing Society. 2010. Leading edge tutorials. Retrieved November 5, 2010, http://www.informs.org/Community/ICS/Projects/Tutorials.

ORMS Educational Programs. 2010. Retrieved November 5, 2010, http://www.informs.org/Build-Your-Career/INFORMS-Student-Union/ORMS-Educational-Programs.

SIAM Working Group on CSE Undergraduate Education. 2006. Undergraduate computational science and engineering education. http://www.siam.org/about/pdf/CSE_Report.pdf.

The Joint Task Force for Computing Curricula. 2005. Computing curricula 2005: The overview report. http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf.

U.S. Department of Labor, Bureau of Labor Statistics. 2010. Occupational outlook handbook, 2010–11 edition: Operations research analysts. Retrieved August 30, 2010, http://www.bls.gov/oco/ocos044.htm.