Most Common Courses of Specializations in Artificial Intelligence, Computer Systems, and Theory

Sami Surakka

Helsinki University of Technology
P.O. Box 5400
FI-02015 HUT, Finland

sami.surakka@hut.fi

ABSTRACT

The degree requirements of American institutions offering top-level graduate programs in computer science were analyzed. The sample size was 59 institutions for undergraduate and 32 for graduate programs. The purpose was to solve which courses were the most commonly offered in the specializations of Artificial Intelligence, Computer Systems, and Theory. The results can be useful to professors who are responsible for any of these three specializations.

Keywords

Advanced courses, content analysis, degree requirements, document analysis, graduate program

1. INTRODUCTION

Curriculum design is a complicated issue including many different points of view that often contradict each other. Joint international efforts such as Computing Curricula 2001 (CC2001) [5] have been carried out to build general frameworks for designing and comparing different computer science curricula. CC2001 is useful for designing introductory and intermediate studies but it is less useful for designing specializations because it is limited to undergraduate programs (p. 1) whereas specializations are more typical in graduate programs. Specializations were covered in the report only on a general level but no recommendations for specific specializations were provided.

A benchmarking study was conducted in order to approach this curriculum design problem. The degree requirements of American institutions offering top-level graduate programs in computer science were analyzed. The purpose was to solve which courses were the most commonly offered in certain specializations.

Institutions use different names for the organization of advanced courses: at least the names concentration, option, specialization, and track have been used. In the present paper, specialization is used as a common name for these concepts. According to the ERIC Thesaurus [4], specialization means "concentration of interest and effort, or restriction of function, to a particular aspect of some larger area of endeavor (such as a field of study, occupation, etc.)—also, the process of progressive differentiation of functions."

We found in our previous research [16] that the four most common specializations were Computer Systems, Theoretical Computer Science, Software Systems, and Artificial Intelligence. The main target of our previous paper was Software Systems. The present paper is limited to the *other* common specializations than Software Systems; that is, to Artificial Intelligence, Computer Systems, and Theoretical Computer Science. However, the name Theory is used instead of Theoretical Computer Science because Theory is a more common specialization name than Theoretical Computer Science.

First, it was analyzed which specializations were most commonly offered when the purpose was to assure the previous results [16]. Second, the main purpose of the present research was to solve which courses were the most commonly offered in these three specializations.

The results can be useful to professors who are responsible for any of these three specializations when they consider which courses are required in a specialization and which courses are more suitable for electives. One should note that the course requirements were analyzed only in extent necessary to classify courses but no detailed results about course contents are presented. In other words, the results of the present paper are not useful for the more detailed question: Which topics should be covered on some specific course.

The related work is presented in Section 2 and the research method in Section 3. Section 4 presents the results of the present research. Finally, the research is discussed in Section 5.

2. RELATED WORK

First, the definitions of the main concepts are presented. Second, other publications are presented.

According to Information Technology Vocabulary [10, p. 22], artificial intelligence is

> the branch of computer science devoted to developing data processing systems that perform functions normally associated with human intelligence, such as reasoning, learning, and self-improvement.

According to the same source (p. 8), computer system is "one or more computers, peripheral equipment, and software that perform data processing."

Surprisingly, no standard definition was found for theory, theory of computation, or theoretical computer science. Instead, three sections of the journal Theoretical Computer Science [17] are suitable for characterizing the area: (a) algorithms, automata, complexity and games, (b) logic, semantics and theory of programming, and (c) natural computing. The third section deals "with the theoretical issues in evolutionary computing, neural networks, molecular computing, and quantum computing."

CC2001 [5, p. 235] presents a list of advanced courses divided into thirteen areas. These areas are not called specializations but they are interesting for the present paper anyhow. Out of these thirteen areas, the following four are most relevant to the present paper: Algorithms and Complexity, Architecture and Organization, Operating Systems, and Intelligent Systems.

Computer Engineering 2004 (CE2004) [14] is a curriculum recommendation targeted at computer engineering programs. It

presents a list of eighteen areas of knowledge of which the following three are the most relevant to the present paper (p. 12): Computer Architecture and Organization, Computer Systems Engineering, and Operating Systems. An example list of elective courses is presented as well but these courses are not divided according to the areas of knowledge (p. 35).

MSIS 2000: Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems [7] is relevant because it is targeted at graduate programs unlike the other recent ACM curricula recommendations. It is recommended that an IS graduate program offers a specialization called career track. In a career track, at least twelve units (four courses) should be required which may include practicum (pp. 12 and 14). What is particularly interesting, sixteen examples for career tracks such as Electronic Commerce are listed and four example courses are suggested for each career track (p. 13). Course descriptions were not provided but nevertheless, the abstraction level and target area from the viewpoint of degree structure are almost the same as in the present paper. However, out of these sixteen specializations, none is relevant to the specializations selected for the present research.

The ACM Computing Classification System [1] is used to classify publications, which is a different purpose than the characterization of specializations. Still, the classification system is relevant enough for the present paper. It lists eleven main categories called first-level nodes. Out of these main categories, at least (a) C. Computer Systems Organization is relevant to Computer Systems specializations, (b) E. Data and F. Theory of Computation are relevant to Theory specializations, and (c) I. Computing Methodologies is relevant to Artificial Intelligence specializations.

3. RESEARCH METHOD

The used research method was content analysis, sometimes called document analysis. The basic goal of content analysis "is to take a verbal, non-quantitative document and transform it into quantitative data (Bailey, 1978)" [3, p. 164].

Degree requirements were used as the data source instead of, for example, job advertisements. We have conducted a job advertisement analysis previously in the area of software systems [15]. Based on our experience, this would probably not work for purely academic specializations such as Artificial Intelligence and Theory for several reasons; for example, it would be hard to find enough suitable advertisements. Job advertisements would be a good data source in the area of computer systems but for the sake of consistency, the same data source was used for all three specializations.

3.1 Sampling

American data sources were used for practical reasons and in order to get results that would be probably interesting for a wider readership. For graduate programs, the 32 best institutions were selected from U.S. News [18] ranking list for computer science graduate programs. For undergraduate programs, the 59 best institutions were selected from the same list. The sample was greater for the undergraduate programs because they offered specializations less often than the graduate programs did.

From each institution, the degree requirements were sought from the web pages of the institution. The data were gathered in July and August 2006, and most of the data were from the academic years 2005–2006 and 2006–2007.

Bachelor's of Science in Computer Science was selected if an institution offered several undergraduate programs in computing. The closest alternative was selected if no such program was offered.

Master's of Science in Computer Science was selected if an institution offered several graduate programs in computing. The closest alternative was selected if no such program was offered. The Doctoral program was selected if no Master's program was offered or the Master's program did not offer specializations but the Doctoral program did.

3.2 Coding

First, the coding of the specializations is explained. Then, the course coding is presented.

3.2.1 Specializations

Typically, in an American computer science program at a research university, a student may or has to choose one specialization out of 3–10 alternatives. Next are explained how the specializations were classified. For example, it had to be decided which specializations were classified to the category Artificial Intelligence.

In some cases, other areas than specializations were used if specializations were not offered. Such areas were typically called breadth or diversity requirements and the number of areas was small, from two to five. For example, a student had to take at least one course from each of three breadth areas and a breadth area listed 3–5 courses. We considered using breadth areas instead of specializations as a small problem because data from breadth areas showed topics that were considered central to that area. This decision was made because specializations were rare in undergraduate programs.

Classification of the category Artificial Intelligence was straightforward because most suitable specializations were named as Artificial Intelligence. Intelligent Systems specializations were included as well.

The categories Computer Systems and Theory were more difficult to classify than Artificial Intelligence. A Computer Systems specialization was always included in the category Computer Systems. A Systems specialization was included if both software and hardware courses were required or elective. A Systems specialization was classified as Software Systems if it was purely software-oriented. Computer Architecture and Hardware specializations were always classified into the category Hardware, not into Computer Systems.

Specializations Algorithms, Formal, Theoretical Computer Science, Theory, and Theory of Computation were always included in the category Theory. A specialization Foundations of Computer Science was included if it was suitable enough according to the characterization presented in Section 2.

Altogether, the selected degree programs offered 319 specializations of which 93 (29%) course names were read before classification. In 71% of the cases, the classification was based on a specialization name only. More details of the classification are presented in Appendix A.

Proportion was counted for each specialization. Greater proportion means that a specialization was offered more often. For example, 24 undergraduate programs offered specializations and 16 of these programs offered a specialization in Computer Systems. Thus, the proportion of Computer Systems was 67% for the undergraduate programs.

3.2.2 Courses

Typically, in an American computer science program at a research university, a specialization consists of 3–5 courses that are required or elective. Next are explained how the courses were classified. For example, it had to be decided which courses were classified to the category Operating Systems.

Data was analyzed twice. The first round was more superficial because only course names were used. During the second round, the course descriptions were read as well and the courses were classified using predefined course definitions (Appendix B). We wrote Appendix B using mainly the results of the first round, CC2001 [5], and the course catalogs of our institution [8; 9]. During the second round, a course was classified into the category "Not found" if we did not find its description from the web. Altogether, the six subsamples included 528 courses of which 23 (4%) descriptions were not found. A course was classified into the category Other if the course description was found but we were not able to classify the course using Appendix B. One hundred and seventy (32%) courses were classified into the category Other.

For the courses, weighted averages were counted instead of proportions. Counting weighted averages was more complex but they were used in order to take account whether a course was required or elective. Typically, a specialization included a set of courses that were required or elective. For example, a student had to take two required courses and choose one course from the list of five. One could assume that required courses were more central than elective courses for a given specialization. Each course was given weight 1 if the course was required. The weight was counted according to the number of elective courses if the course was elective. For example, the weigh was 0.2 when a student had to choose one course out of five.

Finally, weighted averages from 0 to 1 were counted for each course category. For example, the weighted average would be 0.46 [(4 * 1 + 3 * 1/5)/10 = 0.46] if the number of specializations was ten, the course was required in four specializations, and offered as elective (choose one out of five) in three specialization. Therefore, greater weighted average means that a course was more common or central to the given specialization. The maximum average 1 would mean that the course was required in every specialization.

3.3 Changes Compared with Previous Research

Compared with our previous research [16], the following changes were made into the research methodology: (a) The sample size of institutions was considerably greater for the undergraduate (N = 59) than for the graduate programs (N = 32) because specializations were offered less often in the undergraduate programs. The goal was that the size of each subsample for a specialization should be at least ten. (b) Only U.S. News [18] ranking list was used because it was the most recent. Previously also the ranking list by Geist and others [6] was used. However, Geist and others' list is already ten years old. (c) The descriptions of all courses were read before a course was classified. Previously, less than 10% of the course descriptions was read and therefore, the classification was based mostly on the course names. (d) For courses, weighted averages were counted instead of proportions. This should show more accurately which courses were required and thus considered to be most central for a certain specialization. Previously required and elective courses were weighted equally. (e) The details of the classification are presented in the appendices in order to make it clearer what specialization and course categories stand for. Previously, no such documentation was provided. (f) Course prerequisites were not analyzed.

4. RESULTS

First, information on the selected institutions is presented. Then, the most common specializations in the degree programs are presented. Finally, the most common courses of the selected three specializations are presented.

4.1 Selected Institutions

According to the Carnegie Classification of Institutions of Higher Education [2], all selected institutions belonged to the category "Research Universities (very high research activity)." Thus, the samples were not representative relative to all institutions that offered computing programs.

4.2 Specializations

The most common specializations in the selected degree programs are presented in Table 1. A specialization is shown in the table if its proportion was at least 10% in the undergraduate or graduate programs. The rows are ordered first according to the column Undergraduate and then according to the column Graduate.

As expected, the graduate programs offered specializations more often than the undergraduate programs did. Ninety-one percent of the graduate programs offered specializations or used equivalent classifications when the proportion was 41% for the undergraduate programs. These are the subsamples of Table 1 (n = 24 and n = 29, respectively).

The proportion of Theory is greater than 100% for the graduate programs because some programs offered more than one specialization that were classified into this category. For example, two specializations were counted if a program offered the specializations "Algorithms" and "Theory of Computation," more information is presented in Appendix A. Similarly, the proportions of the category Other are greater than 100%.

A similar table was presented in our previous paper [16]. There are some differences but for brevity, they are not explained here. For the purposes of the present paper, it is enough to notice that the specializations Artificial Intelligence, Computer Systems, and Theory are still common.

Table 1. Proportions (%) of offered specializations in selected degree programs

Specialization	**Under-graduate** (n = 24)	**Graduate** (n = 29)
Theory	75	110
Computer Systems	58	41
Artificial Intelligence	42	69
Hardware	42	28
Software Systems	33	34
Computer Graphics	29	45
Programming Languages	29	41
Scientific Computing	29	38
Computer Networks	25	21
Applications	25	10
Databases	17	34
Usability	13	7
Software Engineering	8	17
Other	146	131

4.3 Courses

The most central courses of the specializations in Artificial Intelligence, Computer Systems, and Theory are presented in Sections 4.3.1, 4.3.2, and 4.3.3. A course is presented if its weighted average is at least 0.02. However, a course is shown even if its average is smaller than 0.02 when it belongs to the used classification category. The courses are ordered first according to the average and then according to the name. The category Not found/Other is presented last and was explained in Section 3.2.

4.3.1 Artificial Intelligence

The most common courses of the specializations in Artificial Intelligence are presented in Table 2 and Table 3. One could expect that an introductory course Artificial Intelligent was required in every undergraduate specialization and therefore, its weighted average in Table 2 should be 1. However, this was not the case. The weighted average was 0.58 because a course Artificial Intelligence was required in three and elective in six undergraduate specializations. For example, at the undergraduate program of Brown University, a student had to choose one out of four courses: CS141 Introduction to Artificial Intelligence, CS143 Introduction to Computer Vision, CS148 Building Intelligent Robots, or CS149 Introduction To Combinatorial Optimization.

Table 2. Most common courses (n = 49) **of Artificial Intelligence specializations in undergraduate programs** (n = 9)

Course	Weighted average
Artificial Intelligence	0.58
Robotics	0.25
Computer Vision	0.24
Natural Language Processing	0.14
Machine Learning	0.12
Neural Networks	0.11
Advanced Artificial Intelligence	0.10
Expert Systems	0.01
Multi-Agent Systems	0.00
Planning and Reasoning Systems	0.00
Other/Not found	0.57

Table 3. Most common courses (n = 107) **of Artificial Intelligence specializations in graduate programs** (n = 16)

Course	Weighted average
Artificial Intelligence	0.27
Advanced Artificial Intelligence	0.24
Machine Learning	0.19
Natural Language Processing	0.14
Planning and Reasoning Systems	0.10
Computer Vision	0.06
Robotics	0.06
Expert Systems	0.05
Multi-Agent Systems	0.03
Neural Networks	0.02
Other/Not found	0.53

The results are compared with CC2001 [5] and the ACM Computing Classification System [1] because they are most relevant. According to CC2001 (p. 235), the following courses belong to the area Intelligent Systems: Intelligent Systems, Automated Reasoning, Knowledge-Based Systems, Machine Learning, Planning Systems, Natural Language Processing, Agents, Robotics, Symbolic Computation, and Genetic Algorithms. The CC2001 list matches reasonable well with our results. The biggest differences are that (a) Symbolic Computation and Genetic Algorithms are not central according to our results and (b) Computer Vision is central according to our results but it was not in the CC2001 list.

According to the ACM Computing Classification System [1], the sublevels of the second-level category I.2 Artificial Intelligence are as follows: I.2.0 General, I.2.1 Applications and Expert Systems, I.2.2 Automatic Programming, I.2.3 Deduction and Theorem Proving, I.2.4 Knowledge Representation Formalisms and Methods, I.2.5 Programming Languages and Software, I.2.6 Learning, I.2.7 Natural Language Processing, I.2.8 Problem Solving, Control Methods, and Search, I.2.9 Robotics, I.2.10 Vision and Scene Understanding, I.2.11 Distributed Artificial Intelligence, and I.2.m Miscellaneous. These categories match reasonable well with the top parts of Table 2 and Table 3. The biggest differences are that Automatic Programming and "Programming Languages and Software" are not central according to our results. There are also other categories of the classification system that are not apparent in Table 2 and Table 3 but these are typical subtopics for an Introduction to Artificial Intelligence course (e.g, I.2.8 Problem Solving, Control Methods, and Search).

4.3.2 Computer Systems

The most common courses of the specializations in Computer Systems are presented in Table 4 and Table 5. The results are compared with CC2001 [5], CE2004 [14], and the ACM Computing Classification System [1] because they are most relevant. According to CC2001 (p. 235), the following courses belong to the areas "Architecture and Organization" and Operating Systems: Advanced Computer Architecture, Parallel Architectures, System on a Chip, VLSI Development, Device Development, Advanced Operating Systems, Concurrent and Distributed Systems, Dependable Computing, Fault Tolerance, Real-Time Systems. The CC2001 list matches reasonable well with Table 4. The biggest differences are that (a) Parallel Architectures, Concurrent and Distributed Systems, Dependable Computing, and Fault Tolerance are not central according to our results. The CC2001 course names System on a Chip and Device Development probably refer to the same type of courses as our category Design of Digital Systems.

Thirty-three courses are presented in the CE2004 list of advanced courses [14, p. 35]. Out of these courses, only Advanced Computer Architecture matches with the results of Table 4. This is not surprising because in a computer engineering program related topics are partly covered in introductory and intermediate courses. For example, the courses Computer Architecture and Computer Networks are scheduled as required third year courses in a model program [14, B.5], not as elective courses. It is not reasonable to compare whether the CE2004 list contains courses that are not mentioned in Table 4 because the CE2004 list is not divided into areas.

According to the ACM Computing Classification System [1], there are seven sublevels of the first-level category C. Computer Systems Organization: C.0 General, C.1 Processor architectures, C.2 Computer-communication networks, C.3 Special-purpose and application-based systems, C.4 Performance of systems, C.5 Computer system implementation, and C.m Miscellaneous. The categories C.1 and C.2 match with our results but the other categories do not match or are too general to compare. To sum up, the match is only satisfactory.

Table 4. Most common courses (*n* = 75) of Computer Systems specializations in undergraduate programs (*n* =12)

Course	Weighted average
Operating Systems	0.41
Computer Networks	0.26
Computer Architecture	0.24
Design of Digital Systems	0.20
Advanced Computer Architecture	0.15
Digital Logic	0.13
Distributed Systems	0.09
Compilers	0.08
Databases	0.06
Embedded Systems	0.04
Programming Languages	0.03
Other	0.30

Table 5. Most common courses (*n* = 112) of Computer Systems specializations in graduate programs (*n* = 13)

Course	Weighted average
Advanced Operating Systems	0.17
Computer Architecture	0.13
Advanced Computer Architecture	0.12
Advanced Computer Networks	0.12
Programming Languages	0.11
Design of VLSI Circuits	0.10
Computer Networks	0.09
Distributed Systems	0.07
Operating Systems	0.06
Advanced Compilers	0.05
Computer Security	0.05
Compilers	0.04
Databases	0.04
Design of Digital Systems	0.04
Embedded Systems	0.03
Advanced Databases	0.02
Digital Logic	0.01
Other	0.37

4.3.3 Theory

The most common courses of the specializations in Theory are presented in Table 6 and Table 7. One could expect that the weighted averages of some courses would be greater, in particular the average of Data Structures and Algorithms. However, this course was often not mentioned at all in the requirements of a specialization. The obvious explanation is that in some degree programs Data Structures and Algorithms is required for every student and it is studied already during the first, second, or third year.

The results are compared with CC2001 [5] and the ACM Computing Classification System [1] because they are most relevant. According to CC2001 (p. 235), the following courses belong to the areas Discrete Structures and "Algorithms and Complexity:" Combinatorics, Probability and Statistics, Coding and Information Theory, Advanced Algorithm Analysis, Automata and Language Theory, Cryptography, Geometric Algorithms, and Parallel Algorithms. The courses "Probability and Statistics" and "Coding and Information Theory" of the CC2001 list are not mentioned in Table 7. In addition, Table 7 contains several courses such as Machine Learning that are not mentioned in the CC2001 list. Anyhow, the overall match is reasonably good.

According to the ACM Computing Classification System [1], there are thirteen sublevels of the first-level categories E. Data and F. Theory of Computation: E.0 General, E.1 Data structures, E.2 Data storage representations, E.3 Data encryption, E.4 Coding and information theory, E.5 Files, E.m Miscellaneous, F.0 General, F.1 Computation by abstract devices, F.2 Analysis of algorithms and problem complexity, F.3 Logics and meanings of programs, F.4 Mathematical logic and formal languages, and F.m Miscellaneous. The category F. Theory of Computation matches well with our results but the category E. Data only moderately.

Table 6. Most common courses (*n* = 116) of Theory specializations in undergraduate programs (*n* = 16)

Course	Weighted average
Design and Analysis of Algorithms	0.36
Computational Complexity	0.21
Theory of Computation	0.20
Advanced Data Structures and Algorithms	0.13
Graph Theory	0.11
Cryptography	0.10
Data Structures and Algorithms	0.09
Logic	0.09
Parallel Computation	0.07
Programming Languages	0.07
Formal Languages and Automata	0.04
Combinatorial Optimization	0.03
Computational Geometry	0.02
Machine Learning	0.02
Verification	0.01
Other/Not found	0.93

Table 7. Most common courses (*n* = 69) of Theory specializations in graduate programs (*n* = 11*)

Course	Weighted average
Design and Analysis of Algorithms	0.33
Advanced Data Structures and Algorithms	0.22
Theory of Computation	0.16
Logic	0.15
Computational Complexity	0.13
Cryptography	0.13
Computational Geometry	0.03
Parallel Computation	0.03
Combinatorial Optimization	0.02
Data Structures and Algorithms	0.02
Machine Learning	0.02
Formal Languages and Automata	0.00
Graph Theory	0.00
Programming Languages	0.00
Verification	0.00
Other/Not found	0.78

*) The original subsample was 22 graduate programs but it was cut in half in order to reduce classification work.

5. DISCUSSION

The results of the present research are original in a sense that similar results have not been published previously. However, the results are not surprising because they generally match well enough with the related recommendations or classifications.

The results can be classified as conservative as a consequence of the selected research methodology. A different target group could have been selected if the purpose was to find alternatives that were less conservative or even dramatically different. For example, software engineering or computer science programs in Brazil, Russia, India, and China—known also as BRIC countries—might be a more suitable target group in that case. However, this was not the purpose of the present research. Instead, the purpose was to solve common requirements in the area of advanced computer science studies that the current ACM curricula recommendations cover poorly. The ACM curricula recommendations can be classified as normative studies. Our research rather supports or complements these normative studies than challenges them.

The results of most common courses (Section 4.3) should be quite similar regardless the country because they show relationships between the various subject matters. What can differ strongly from a country to another, is how commonly a certain specialization is offered; that is, the results of Section 4.2. Indeed, it is likely that Computer Systems specializations are less commonly offered outside the USA. A possible explanation is that American companies such as IBM and Hewlett-Packard are major designers and manufactures of computers. The situation is different in many other countries where computer engineering industry is not as strong. For example in Finland, Computer Systems specializations are rarely offered in the universities. However, Computer Systems specialization is not targeted to computer engineering positions only but might be useful for systems administration positions as well. According to the draft of Computing Curricula 2005 [12, p. 31]: "…, there is a fourth career path that CS programs do not target but nonetheless draws many computer science graduates: *Career Path 4: Planning and managing organizational technology infrastructure*."

As far as we know, the same limitation is not true for Artificial Intelligence and Theory; that is, these specializations are offered quite often outside the USA as well. These two specializations are interesting because they can be characterized as traditional academic specializations and as being more academic than many other specializations. Here, more academic means that a specialization is not at all or only rarely offered at community colleges; that is, it is specific to universities. Some other specializations are less academic in that sense that they are offered at the community colleges as well. For example, specializations Software Engineering, Databases, and Computer Networks are often or sometimes offered at Finnish community colleges.

Specializations Computer Systems and Theory were much more difficult to analyze than Artificial Intelligence for several reasons. Apparently, these two concepts refer to broader areas than specializations on average.

6. ACKNOWLEDGMENTS

We thank Prof. M. Syrjänen, Dr. T. Janhunen, Dr. V. Hirvisalo, and H. Arppe from the Helsinki University of Technology for commenting on the manuscripts of the present paper.

7. REFERENCES

[1] Association for Computing Machinery. *The ACM Computing Classification System [1998 Version]*. Retrieved on August 11, 2006, from ACM web site: http://www.acm.org/class/1998/. 1998.

[2] Carnegie Foundation. *The Carnegie Classification of Institutions of Higher Education*. Retrieved on August 15, 2006, from Carnegie Foundation web site: http://www.carnegiefoundation.org/classifications/.

[3] Cohen, L., Manion, L., and Morrison, K. *Research Methods in Education* (5th ed.). RoutledgeFarmer, London, 2000.

[4] Educational Resources Information Center. *ERIC Thesaurus*. Retrieved October 31, 2005, from Educational Resources Information Center web site: http://www.ericfacility.net/extra/pub/thessearch.cfm.

[5] Engel, G., and Roberts, E. (Eds.). *Computing Curricula 2001: Computer Science*. IEEE Computer Society and Association for Computing Machinery. Retrieved on October 18, 2002, from IEEE Computing Society web site: http://www.computer.org/education/cc2001/final/cc2001.pdf.

[6] Geist, R., Chetuparambil, M., Hedetniemi, S., and Turner, A. J. Computing research programs in the U.S. *Communications of the ACM, 39*, 12 (Dec. 1996), 96-99.

[7] Gorgone, J.T, and Gray, P. (Eds.). MSIS 2000: Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems. *Communications of the Association for Information Systems*, vol. 3, January 2000. Retrieved on August 12, 2006, from ACM web site: http://www.acm.org/education/curricula.html#MSIS2000.

[8] Helsinki University of Technology. *Study programme. ECTS guide 2003–2004*. 2003.

[9] Helsinki University of Technology. *Study programme. ECTS guide 2006–2007*. 2006.

[10] International Organization for Standardization. *Information technology. Vocabulary. Part 1: Fundamental terms*. 3rd ed. ISO/IEC 2382–1: 1993 (E/F). 1993.

[11] Radatz, J. (Ed.). *The IEEE standard dictionary of electrical and electronics terms*. Institute of Electrical and Electronics Engineers. 1996.

[12] Shackelford, R., Cross, J. H., II, Davies, G., Impagliazzo, J., Kamali, R., LeBlanc, R., et al. (2005). *Computing Curricula 2005. The overview report*. Draft, April 11, 2005. Retrieved on June 21, 2005, from ACM web site: http://www.acm.org/education/Draft_5-23-051.pdf.

[13] Shapiro, S.C. (Ed.). *Encyclopedia of Artificial Intelligence*, 2nd ed. Wiley, New York.

[14] Soldan, D., Hughes, J.L.A, Impagliazzo, J., McGettrick, A., Nelson, V.P., Srimani, P.K., Theys, M.D. *Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering*. Retrieved on August 12, 2006, from ACM web site: http://www.acm.org/education/curricula.html#CE2004. 2004.

[15] Surakka, S. Analysis of technical skills in job advertisements targeted at software developers. *Informatics in Education, 4*, 1, 101-122. 2005.

[16] Surakka, S. Specialization in Software Systems: Content Analysis of Degree Requirements. In T. Salokoski, T. Mäntylä, and M. Laakso (Eds.). *Proceedings of Koli*

Calling. Fifth Koli Calling Conference on Computer Science Education, November 17-20, 2005, Koli, Finland. TUCS General Publication, 41, pp. 162-165. URL: http://www.it.utu.fi/koli05/proceedings/final_composition.b5.060207.pdf.

[17] *Theoretical Computer Science*. Retrieved August 28, 2006, from Elsevier web site: http://www.elsevier.com/wps/find/journaldescription.cws_home/505625/description#description.

[18] U.S. News & World Report. *America's Best Graduate Schools 2007*. Premium online edition. 2006. (URL is not provided because this is a charged service.)

APPENDICIES

Appendix A: Classification of Specializations

Table A1 presents how the specializations were classified to the categories used in the present paper. Not all original names of the data set are presented in the table but only typical ones. For example, the specialization Intelligent Systems was classified into category Artificial Intelligence.

Table A1. Categories of specializations

Category	Specialization name
Applications	Applications
Artificial Intelligence	Artificial Intelligence Intelligent Systems
Computer Systems	Computer Systems Operating Systems* Systems*
Computer Graphics	Computer Graphics Graphics
Computer Networks	Computer Networks Networking and Communications
Databases	Database Systems Databases
Hardware	Computer Architecture Hardware
Other	A specialization that was not classified to the other categories
Programming Languages	Compilers Programming Languages
Scientific Computing	Numerical Analysis Scientific Computing
Software Engineering	Software Engineering
Software Systems	Operating Systems* Software Software Systems Systems*
Theory	Algorithms Formal Foundations of Computer Science* Theoretical Computer Science Theory Theory of Computation
Usability	Human-Computer Interaction Usability

* An asterisk means that a specialization was classified to one of alternative categories according to the course requirements. For example, a Systems specialization was classified to the category Computer Systems or Software Systems according to the course requirements while a Theory specialization was always classified to the category Theory.

Appendix B: Course Definitions

The definitions used to classify the courses are presented next. Artificial Intelligence courses are presented in Appendix B.1, Computer Systems courses in B.2, and Theoretical Computer System courses in B.3. Inside each list, the courses are ordered by name.

A CC2001 course description was used when available. In most cases, the course descriptions of our institution [8; 9] were used. A description from some other institution was used when no suitable course was offered at our institution. These references to the web pages are presented only here, not in the reference list.

No course by course definitions are presented for the courses of which names include the word "Advanced." For example, an Advanced Artificial Intelligence course covers the same type of topics as Introduction to Artificial Intelligence but at a more advanced level. An alternative course name would be Artificial Intelligence II when Artificial Intelligence I is used for an introductory course. Typically, an introductory course is a prerequisite for an advanced course.

B.1 Artificial Intelligence

Artificial Intelligence
: Introduction to artificial intelligence. See the CC2001 definition of CS260 Artificial Intelligence [5, p. 220].

Computer Vision
: Introduction to the use of computers in analysis of visual data. Image forming, image geometry, low-level vision, motion perception, feature extraction, 2D area, and 3D object representation, forming and recognition of structural patterns. [9, p. 96]

Expert Systems
: Design principles of computer system designed to solve a specific problem or class of problems by processing information specific to the problem domain. A domain area can be, for example, law or medicine. [13, p. 380]

Machine Learning
: Machine learning studies the automated acquisition of expert knowledge. Representation of experience and acquired knowledge. Defining the performance task. Supervised and unsupervised learning. Incremental and nonincremental learning. Inductive and analytic learning. [13, pp. 785–788]

Multi-Agent Systems
: Theory, architectures, and applications for agent-based computing. Decision-making on the basis of uncertain information. [8, p. 83]

Natural Language Processing
: Overview of application of statistical and adaptive methods for analysis of natural language, for example, the analysis, organization and search from text collections, language modeling for natural language recognition, syntactic and semantic analysis, probabilistic grammars and parsing, and statistical machine translation. [9, p. 95]

Neural Networks
: Introduction to neural networks, learning processes, single layer networks, multi-layer perception networks and back-propagation algorithm, radial-basis function networks, self-organizing maps and learning vector quantization. [9, p. 95]

Planning and Reasoning Systems
: Planning can be thought of as determining all the small tasks that must be carried out in order to accomplish a goal. Constraints and scheduling. Different types of reasoning such as case-based, causal, and commonsense

reasoning. [13, pp. 1159 and 1265–1339, and http://ic.arc.nasa.gov/projects/remote-agent/pstext.html].

Robotics
: Basics in robotics. Industrial robots and mobile robots. Subsystems and physical component of robots. Basic kinematics and motion control principles. Examples of various practical applications of robotics. [9, p. 46]

B.2 Computer Systems

Compilers
: Introduction to compilers. See the CC2001 definition of CS240 Programming Language Translation [5, p. 215].

Computer Architecture
: Introduction to computer architecture. See the CC2001 definition of CS220 Computer Architecture [5, p. 206].

Computer Networks
: Implementation principles of telecommunications software and fundamental principles of computer networks, especially IP networks. Routing, name service, network administration, protocol development and network programming. [9, p. 110]

Computer Security
: Basic methods for implementing security and applying them. Building safe systems. Identification, authentication and access control. Possibilities offered by cryptography. Security models. Security of operating systems and services. [9, p. 110]

Databases
: Introduction to database systems. See the CC2001 definition of CS270 Databases [5, p. 224].

Digital Logic
: Introduction to digital systems. Design and implementation methods of digital electronic devices. Coupling functions, combinatorial and sequential logic, MSI and LSI circuits, PLA circuits, memory, timing and interface issues, digital arithmetic and codes. [9, p. 139]

Design of Digital Systems
: Design and laboratory exercises concerning digital circuits and devices and electrical phenomena. [9, p. 140]

Design of VLSI Circuits
: Implementation of digital logic elements: combinational circuits, latches and flip-flops. Synchronous and asynchronous digital systems. Implementation of finite state machines, testing of digital circuits. [9, p. 139]

Distributed Systems
: Architecture and implementation techniques of distributed systems. [9, p. 108]

Embedded Systems
: Development of embedded systems, their hardware, solutions and application areas. Programming with real-time, fault-tolerance, and correctness requirements. [9, p. 108]

Operating Systems
: Introduction to operating systems. See the CC2001 definition of CS225 Operating Systems [5, p. 210].

Programming Languages
: The interpretation of the typical mechanism of programming languages. The concepts, structure, and implementation of interpreters. Definition and implementation of domain-specific languages. [9, p. 105]

B.3 Theory

Combinatorial Optimization
: Algorithms for matching problems. Introduction to matroids; polyhedral combinatorics. Complexity theory and NP completeness. Perfect graphs and the ellipsoid method. [http://www.college.columbia.edu/…]

Computational Complexity
: NP-completeness. Randomized algorithms. Cryptography. Approximation algorithms. Parallel algorithms. Polynomial hierarchy. PSPACE-completeness. [9, p. 102]

Computational Geometry
: Introductory course to computational geometry. Designing and analyzing efficient algorithms and data structures for computational problems in discrete geometry, such as convex hulls, geometric intersections, geometric structures such as Voronoi diagrams and Delaunay triangulations, arrangements of lines and hyperplanes, and range searching. [http://www.cs.umd.edu/class/fall2005/cmsc754/]

Cryptography
: Data and communications security. Principles of cryptographic security. Symmetric cryptosystems. Stream ciphers. Block ciphers: DES, IDEA, AES. Modes of operation. Asymmetric cryptosystems. Digital signatures. Authentication and key agreement. Applications: SSL, TLS, IPSec, GSM, Bluetooth. [9, p. 101]

Data Structures and Algorithms
: Introduction to data structures and algorithms. See the CC2001 definition of CS103I Data Structures and Algorithms [5, p. 165].

Design and Analysis of Algorithms
: Introduction to design and analysis of algorithms. See the CC2001 definition of CS210 Algorithm Design and Analysis [5, p. 204].

Formal Languages and Automata
: An introduction to the fundamental ideas and models underlying computing: finite automata, regular sets, pushdown automata, context-free grammars, Turing machines, undecidability, and complexity theory. [http://www.cs.cmu.edu/afs/cs/usr/cathyf/www/ugcourses.htm]

Graph Theory
: Introduction to graph theory. Trees, planar graphs and digraphs. Graph coloring. Random graphs. Algorithms for central graph problems. Applications. [9, p. 102]

Logic
: Introduction to logic. Propositional and predicate logic, their syntax, semantics and proof theory. Applications of logic in computer science. [9, p. 101]

Machine Learning
: See Appendix B.1.

Parallel Computation
: Fundamental theoretical issues in designing parallel algorithms and architectures. Shared memory models of parallel computation. Parallel algorithms for linear algebra, sorting, Fourier Transform, recurrence evaluation, and graph problems. Interconnection network based models. Algorithm design techniques for networks like hypercubes, shuffle-exchanges, threes, meshes and butterfly networks. Systolic arrays and techniques for generating them. Message routing. [http://www.eecs.berkeley.edu/Gradnotes/Content/Section6.pdf]

Programming Languages
: See Appendix B.2.

Theory of Computation
: Introduction to theory of computation. Finite automata and regular languages. Context-free grammars and pushdown automata. Context-sensitive and unrestricted grammars. Turing machines and computability. [9, p. 101]

Verification
: Verification and analysis of parallel and distributed systems using computer aided tools. Practical verification methods, e.g. partial reachability analysis. [8, p. 83]