A Web-based Curriculum Engineering Tool for Investigating Syllabi in Topic Space of Standard Computer Science Curricula

Takayuki Sekiya
Information Technology Center
The University of Tokyo
3-8-1 Komaba, Meguro,
Tokyo, Japan
sekiya@ecc.u-tokyo.ac.jp

Yoshitatsu Matsuda
Graduate School of Arts and Sciences
The University of Tokyo
3-8-1 Komaba, Meguro,
Tokyo, Japan
matsuda@graco.c.u-tokyo.ac.jp

Kazunori Yamaguchi
Graduate School of Arts and Sciences
The University of Tokyo
3-8-1 Komaba, Meguro,
Tokyo, Japan
yamaguch@graco.c.u-tokyo.ac.jp

Abstract—**For university students, a syllabus gives fundamental information about a course, and is important for choosing a course. However, it is not an easy task for students to grasp the topics actually covered by a course syllabus because they have only little knowledge about topic words in the syllabus before they learn the course. We have been studying on a machine learning method of systematically analyzing syllabi by standard curricular guidelines such as "Computing Science Curricula CS2013," which is released by the ACM and IEEE Computer Society. We acquired a probabilistic topic model of computer science syllabi, and developed a tool for investigating the actual syllabi in the model. In this paper, we introduce a web-based tool and demonstrate its effectiveness by some examples. By applying our tool to a syllabus, students and teachers can know how strongly the syllabus and topics are related quantitatively, where each topic corresponds to the Knowledge Area of CS2013 such as "Algorithms and Complexity (AL)." In addition, the tool utilizes four meta-topics (HUMAN, THEORY, PROGRAMMING, and SYSTEM), which are extracted by investigating the actual syllabi. The tool also provides a list of syllabi similar to the given syllabus, which are selected from the actual syllabi of the top-ranked universities. These information are beneficial for students to understand the courses and for teachers to improve their syllabi.**

I. Introduction

For university students, a syllabus gives fundamental information about a course, and is an important document for choosing a course. When teachers prepare syllabi, they work hard to represent topics dealt in their courses. However, there is no objective index to evaluate the appropriateness of a syllabus. For students it is not an easy task to grasp the topics actually covered by courses through syllabi because they do not always have enough knowledge yet when they read syllabi.

We have been studying on a method of systematically analyzing syllabi that uses machine learning on curricular guidelines [1]. On the basis of "Computing Science Curricula CS2013," a report released by the ACM and IEEE Computer Society [2], we applied our method to analyzing 10 computer science (CS) related curricula offered by CS departments of universities in the United States, and found that CS2013 uniformly covered a wide area of computer science [3]. Through this analysis, we acquired a probabilistic topic model of computer science based on CS2013 Body of Knowledge (BOK). We also applied our method to analyzing CS related course syllabi offered by global universities listed in the Times Higher Education (THE) World University Rankings. Through the analysis of actual course syllabi, we found a hierarchical structure of the Knowledge Areas (KA) of CS2013 BOK. Using the structure, we also found that 18 KAs could be categorized into four clusters [4].

In this paper, we introduce a web-based curriculum engineering tool using a probabilistic topic model in computer science based on CS2013 BOK by use of our curriculum analysis method. The tool does not require any special knowledge or software on our curriculum analysis method. When students and teachers just input a course syllabus into our tool, the tool estimates what extent the syllabus is related to each topic (= KA) and each meta-topic (= cluster of KAs). The tool also extract similar courses among more than 3,000 courses offered by the top-ranked universities in the THE rankings. These information are beneficial for students to understand the courses and for teachers to improve their syllabi.

The rest of this paper is organized as follows. In Section II, we review related works. In Section III, we explain the basic theories of our curriculum analysis method by using a simplified version of supervised LDA (ssLDA). In Section IV, we explain the dataset used for our tool, CS2013 BOK and the syllabi of the courses of the top-ranked universities. In Section V, we introduce the tool and demonstrate the effectiveness of the web-

978-1-5090-5920-1/17/$31.00 ©2017 IEEE

based tool by examples, and Section VI concludes the paper with a summary of key points.

II. Related Works

Since a curriculum is one of the most important assets of higher education, some faculties developed curriculum design tools and made them public [5], [6]. In many tools, teachers are required to define courses with units of knowledge [7], which takes a lot of time and efforts. Tungare et al. created a repository system for computer science syllabi [8]. They developed tools such as Syllabus-Maker for creating and comparing syllabi. They have not developed a technique to grasp characteristics of a whole curriculum. With our method, faculties only have to prepare course syllabi.

A number of studies have been made on methodologies and tools for analyzing curricula by using statistically processed syllabus data [9]–[11]. Our approach enables us to use this data to compare courses and curricula offered by different universities.

Many teachers in the CS field have read the CS2013 report and some of them contributed their knowledge to it. For example, Marshall tried to quantify the changes in the CS structure of the ACM/IEEE curricula series [12]. He focused attention on the BOK of computing curricula and visualized the structure of the curricula. However, he modeled Knowledge Areas (KAs) of BOK as a network graph to represent Knowledge Units (KUs) and topics as edges of the graph in the BOK. In applying his method to comparing the computing curricula, it was necessary to identify corresponding topics among the curricula. On the other hand, our method deals with curricula merely as a set of syllabus documents, and no identification of the topics is needed.

Gluga et al. developed a web-based system called PROGOSS that maps curricula learning goals and mastery levels to individual assessment tasks across entire degree programs [13]. The PROGOSS system has a function for educators to map prerequisites and goals of their courses into CS2013. Szabo et al. also developed curriculum analysis framework which supports the identification of prerequisite concepts [14]. Our tool is considerably easy to use because it can offer the relationships among a course and CS2013 BOK by just inputting a syllabus of the course.

The Open Syllabus Project[1] is a large-scale project to collect numerous syllabi and extract citations and other metadata from them. The project offers various useful information such as a list of popular textbooks in each field. But the project has not yet provided a mechanism for teachers or students to analyze a course syllabus.

TABLE I: Knowledge Areas of CS2013.

ID	KA
AL	Algorithms and Complexity
AR	Architecture and Organization
CN	Computational Science
DS	Discrete Structures
GV	Graphics and Visualization
HCI	Human-Computer Interaction
IAS	Information Assurance and Security
IM	Information Management
IS	Intelligent Systems
NC	Networking and Communication
OS	Operating Systems
PBD	Platform-Based Development
PD	Parallel and Distributed Computing
PL	Programming Languages
SDF	Software Development Fundamentals
SE	Software Engineering
SF	Systems Fundamentals
SP	Social Issues and Professional Practice

III. Curriculum Analysis

A. Structure of CS2013

For over 40 years, the ACM and IEEE Computer Society jointly have been working on curricular guidelines for undergraduate programs in CS and have been releasing reports on roughly a ten-year cycle, such as CC1991 [15], CC2001 [16], and CS2008 [17]. In December 2013, they released the latest report, "Computing Science Curricula CS2013" as a reference curriculum in CS. The report includes a set of principles for CS2013, a redefined BOK, exemplars of actual courses and curricula. According to the CS2013 report, the BOK "does not propose a particular set of courses of curriculum structure," but "In Computer Science terms, one can view the Body of Knowledge as a specification of the content to be covered and a curriculum as an implementation." The BOK consists of a set of 18 Knowledge Areas (KAs), corresponding to topical areas of study in computing. Table I shows the names and abbreviations of KAs. Each KA contains about 10 Knowledge Units (KUs). In our method, KUs correspond to syllabi for curriculum analysis and documents of ssLDA. The total number of KUs in CS2013 is 163. In summary, CS2013 represents each KA (namely, a topic in computer science) as a set of short documents, each of which is regarded as a bag of words. We adopted a simple stemming algorithm to obtain nouns in a singular form, and extracted 3,304 words from the BOK of CS2013.

B. Projection of Syllabi to KA Space

In order to investigate the properties of the KAs by actual syllabi, each syllabus is projected to "the KA space" by the simplified, supervised LDA (ssLDA). This section explains the KA space and ssLDA. See [3] for the details.

[1]http://opensyllabusproject.org

The KA space is defined as follows. It is assumed that each syllabus is represented by a weighted mixture of the KAs (denoted as $\boldsymbol{w} = (w_k)$). If a syllabus is strongly related to some KAs, the weights of the related KAs are relatively higher than the other weights. It is also assumed that every weight is not negative ($w_k \geq 0$) and the sum of the weights is normalized to 1 ($\sum w_k = 1$). The continuous space of the weight vector $\boldsymbol{w}$ under the above constraints is called the KA space. The number of dimensions and the degree of freedom in the KA space are 18 (the number of the KAs) and 17 (one less, because of the normalization constraint), respectively. Some properties of the KAs are expected to be discovered by analyzing the distribution of the actual syllabi in the KA space.

ssLDA is a method projecting any syllabus to the KA space, which was proposed in [3]. ssLDA is a modification of LDA (Latent Dirichlet Allocation) [18]. The original LDA is a probabilistic model-based method in natural language processing and it can extract the topics (corresponding to the KAs in this paper) from a given set of bags of words (the KUs). In addition, the original LDA can project any new bag of words (an actual syllabus) to a point in the extracted topic space (the KA space). However, the original LDA is an unsupervised method, which is not suitable for the KUs labeled with the corresponding KA. Though a supervised version of LDA was proposed in [19], [20], it has too many parameters to represent such a simple model with only the KA labels. ssLDA is a simplification of the supervised LDA, which is designed for appropriately extracting the KA space from the KUs. The generative model of ssLDA is shown in Fig. 1. The meaning of the symbols is given in Table II. The model of ssLDA is essentially equivalent to the original LDA except for the given KA label c and the hyper-parameter of the softmax prediction η. The hyper-parameters $\alpha (= 1)$ and $\eta (= 50)$ was set empirically by a cross validation method. Then, the procedure projecting the actual syllabi to the KA space consists of the following two phases:

1) **Learning** of the KA space: All the KUs with the corresponding KA label is given as the training dataset consisting of the labeled bags of words. The optimal $\boldsymbol{\beta}$ is estimated by maximizing the likelihood of the ssLDA generative model.
2) **Projection** of actual syllabi to the KA space: The optimal $\boldsymbol{\theta}$ is estimated for each actual syllabus under the estimated $\boldsymbol{\beta}$. Then, the estimated $\boldsymbol{\theta}$ is regarded as the position $\boldsymbol{w}$ of the syllabus in the KA space.

The algorithm of the first learning phase is given as a simple modification of the inference in the original LDA by incorporating the softmax model with the usual inference. The second projection phase is completely equivalent to the prediction in the original LDA. Therefore only $\boldsymbol{\beta}$ and the original LDA are sufficient to verify if the probabilistic model extracted by ssLDA corresponds to BOK of CS2013. The details of the algorithm of ssLDA are described in [3].

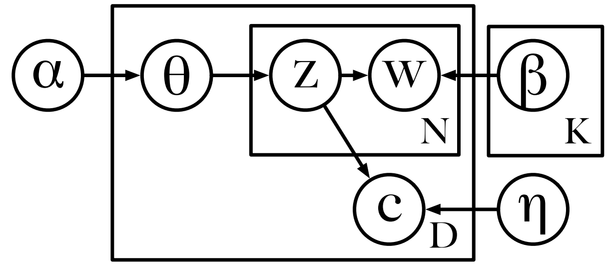

Fig. 1: Generative model of simplified, supervised LDA (cited from [3]).

IV. Dataset

A. Course Syllabi Sets

In this research, we analyzed curricula offered by CS departments of higher educational institutions on the basis of CS2013. In order to obtain such curricula of major universities, we referred to one of the popular university rankings, titled "Times Higher Education (THE) WORLD UNIVERSITY RANKINGS [21], Top 100 universities for engineering and technology 2014-2015 ."

We analyzed the 47 universities of the top 50 universities in the rankings. Table III lists the universities and the departments related to CS.

In our method, each syllabus is characterized by a set of used words which are also used in CS2013. Some syllabi uses few CS2013 words, and our method cannot determine the characteristics of those syllabi. So, we picked up syllabi which include at least 10 CS2013 words. The average ratio of excluded syllabi to total syllabi of each curriculum is about 8.1%, and there is no big difference among the ratio of English curricula (8.2%) and those of non-English curricula (7.4%). As a result, the average number of syllabi of the 47 universities is 65.5, and the average number of words of the syllabi is 39.2.

B. Clusters of KAs

There exists the high interconnection among the KAs in each actual syllabi. The interconnection is utilized for discovering a hierarchical structure of the KAs [4]. By letting $\boldsymbol{w}^i = (w_k^i)$ be the position of the i-th actual syllabus in the KA space, each KA k is characterized as a vector $\boldsymbol{v}_k = (w_k^1, w_k^2, \cdots, w_k^L)$ where L is the number of the actual syllabi ($L = 3077$). Then, the distances among the KAs can be estimated as the distances among $\boldsymbol{v}_k$'s. In order to remove the effects of the mean and the variance, each $\boldsymbol{v}_k$ was normalized to $\bar{\boldsymbol{v}}_k$ so that the mean and the variance of $\bar{\boldsymbol{v}}_k$ over all the actual syllabi are 0 and 1, respectively. Then, the distance between the KAs i and j was estimated as the Euclidean distance between $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$, and Ward's method [22], [23] was applied

TABLE II: Symbols and their meaning in the ssLDA generative model.

Symbols	Meaning
D	Number of documents
N	Number of words in a document
K	Number of KAs
α	Hyper-parameter of the Dirichlet prior
$\boldsymbol{\theta} = (\theta_k)$	Probability of each KA under the Dirichlet prior with α
$\mathbf{Z} = (z_{nk})$	$z_{nk} = 1$ if a KA k is allocated to the n-th word by the multinomial prior with $\boldsymbol{\theta}$ or $z_{nk} = 0$ otherwise
$\boldsymbol{\beta} = (\beta_{ki})$	Probability of a word i under a KA k
$\boldsymbol{W}$	Bag of words for a document
η	Hyper-parameter of the softmax prediction
c	Predicted label of KA by softmax of $\mathbf{Z}$

TABLE III: CS related departments of universities.

Rank	Country	University (Department)
1	us	Massachusetts Institute of Technology (Electrical Engineering and Computer Science)
2	us	Stanford University (Computer Science Dept.)
3	us	California Institute of Technology (Computing + Mathematical Science Dept.)
4	uk	Princeton University (Dept. of Computer Science)
5	uk	University of Cambridge (Computer Laboratory)
6	uk	Imperial College London (Dept. of Computing)
7	uk	University of Oxford (Dept. of Computer Science)
8	ch	ETH Z"urich - Swiss Federal Institute of Technology Z"urich (Dept. of Computer Science)
9	us	University of California, Los Angeles (Computer Science Dept.)
10	us	University of California, Berkeley (Dept. of Electrical Engineering and Computer Sciences)
11	us	Georgia Institute of Technology (College of Computing)
12	ch	'Ecole Polytechnique F'ed'erale de Lausanne (School of Computer and Communication Sciences)
13	sg	National University of Singapore (School of Computing)
14	us	University of Texas at Austin (Computer Science Dept.)
15	us	University of Michigan (Dept. of Electrical Engineering and Computer Science)
16	us	Carnegie Mellon University (School of Computer Science)
17	us	Cornell University (Dept. of Computer Science)
18	us	University of Illinois at Urbana-Champaign (Dept. of Computer Science)
19	nl	Delft University of Technology
19	us	Northwestern University (Dept. of Electrical Engineering and Computer Science)
21	hk	Hong Kong University of Science and Technology (Dept. of Computer Science and Engineering)
22	us	University of California, Santa Barbara (Dept. of Computer Science)
24	ca	University of Toronto Scarborough (Dept. of Computer and Mathematical Sciences)
25	jp	The University of Tokyo (Dept. of Information Science, School of Science)
27	us	University of Wisconsin-Madison (Dept. of Computer Science)
28	de	Technical University of Munich (Dept. of Informatics)
29	sg	Nanyang Technological University (School of Computer Engineering)
30	se	KTH Royal Institute of Technology
31	dk	Technical University of Denmark (Dept. of Applied Mathematics and Computer Science)
32	us	Columbia University (Computer Science Dept.)
33	us	University of Washington (Dept. of Computer Science and Engineering)
34	be	KU Leuven (Dept. of Computer Science)
35	kr	Seoul National University (Dept. of Computer Science and Engineering)
36	hk	The University of Hong Kong (Dept. of Computer Science)
37	uk	University of Manchester (School of Computer Science)
37	au	University of Melbourne (School of Information)
39	au	University of Queensland (School of Information Technology and Electrical Engineering)
40	us	Rice University (Dept. of Computer Science)
41	jp	Kyoto University (Informatics and Mathematical Science, Faculty of Engineering)
42	fr	'Ecole Polytechnique (Computer Science Department)
43	ca	University of British Columbia (Dept. of Computer Science)
45	us	Purdue University (School of Electrical and Computer Engineering)
46	kr	Pohang University of Science and Technology (Computer Science and Engineering)
46	au	University of Sydney (School of Information Technologies)
48	au	Monash University (Faculty of Information Technology)
49	us	University of Minnesota (Dept. of Computer Science and Engineering)
50	us	University of California, San Diego (Dept. of Computer Science and Engineering)

to the distances for constructing the hierarchical cluster tree of the KAs. Fig. 2 shows the constructed structure. It is an intuitively plausible structure. For example, the three KAs related to "HUMAN" (HCI, SP, and SE) were grouped together. In addition, the highly-related pairs among the KAs could be found in the structure (AR and

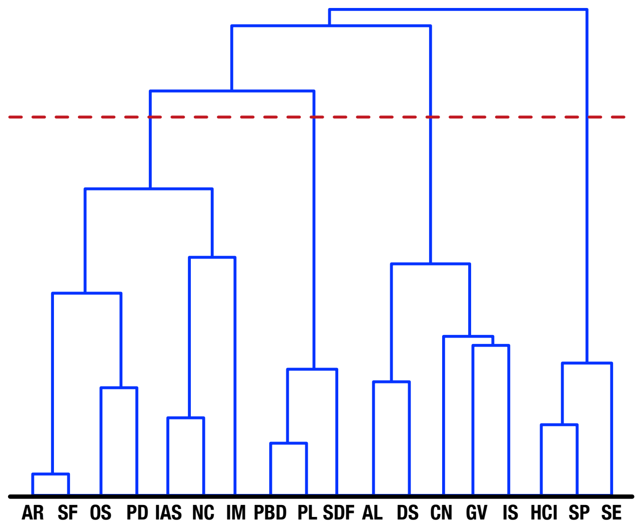

Fig. 2: Hierarchical structure of the KAs by using the interconnection in the actual syllabi.

TABLE IV: Four named clusters of KAs extracted from the hierarchical structure in Fig. 2.

cluster	included KAs	name of cluster
Ω_1	HCI, SP, SE	HUMAN
Ω_2	AL, DS, CN, GV, IS	THEORY
Ω_3	PBD, PL, SDF	PROGRAMMING
Ω_4	AR, SF, OS, PD, IAS, NC, IM	SYSTEM

SF, PBD and PL, IAS and NC, AL and DS, and so on).

Here, the KAs were divided into the four clusters (Ω_1, Ω_2, Ω_3, and Ω_4) by cutting the hierarchical structure at the red dotted line in Fig. 2. Then, we named each cluster for appropriately representing the common factor of the included KAs. Table IV shows the clusters with the names. The weight vector $\boldsymbol{w}$ for each actual syllabus can be reduced to the four-dimensional clusters $\boldsymbol{C} = (c_1, c_2, c_3, c_4)$ by

$$c_l = \sum_{k \in \Omega_l} w_k. \tag{1}$$

The space spanned by $\boldsymbol{c}$ is called the cluster space. The three clusters, THEORY, PROGRAMMING, and SYSTEM seem to correspond to the major assessment indicators in computer science employed by ETS major field test [24], (a) Programming and Software Engineering, (b) Discrete Structures and Algorithms, and (c) Systems. See [4] for the details. For the readers' convenience, we call cluster of KAs "Meta-topic."

V. Curriculum Engineering Tool

In this research, we developed a web-based curriculum engineering tool which estimates relationships among a syllabus and CS2013 KAs. In the following explanation, we call KA by "Topic," and "cluster of KA" explained in Section IV-B by "Meta-topic."

It also finds related courses held in the top-ranked universities in the world. The tool is available at our website[2].

Fig. 3 shows the brief mechanism of the tool. The mechanism consists of the two parts, "Setup Steps" and "Main Steps." In the setup step, the model of CS2013 BOK is extracted based on the methodology explained in Section III is generated beforehand. "CS2013 Model" corresponds to $\boldsymbol{\beta}$ in Fig. 1. Course syllabi of the universities listed in THE2014 (Table III) are projected into Topic space by CS2013 Model. The "Position Vectors of Syllabus in Topic space" for all syllabi are extracted.

In "Main Steps," there are the following five steps:

1) Input Target Syllabus: A user inputs a course syllabus into the tool. We call the syllabus "Target Syllabus" in the following explanation.
2) Preprocess: After the tool removes stop words and adopts a stemming algorithm, the tool extracts words used in CS2013 Model from Target Syllabus .
3) Projection: The tool projects Target Syllabus into Topic space by CS2013 Model. The tool also calculates the weight vector of the Meta-topic.
4) Similarity Analysis: In Topic space, the tool calculates cosine similarity between the position vector of Target Syllabus and those of course syllabi of the universities listed in THE2014, and extracts the courses.
5) Display: The tool displays the position vector in Topic space, the weight vector of Meta-topic, and the similar courses.

The tool is mainly implemented by Perl and PostgreSQL. The program code of ssLDA is based on LDA-C [25] developed by Blei et al. GNU Octave is used for Similarity Analysis. The tool runs on a virtual server[3]

Fig. 4 shows the screen-shot of the analysis result at Step 5. The position vector of Target Syllabus in Topic space is displayed in a table and a bar graph, and the weight vector is also displayed in the same way. The total execution time for Main Steps depends on the length of Target Syllabus. It takes about 2 seconds if Target Syllabus contains about 50 words of CS2013 BOK, and about 10 seconds if it contains about 300 words. Setup Steps take about several hours, but are not executed when teachers and students use the tool, because Setup Steps are executed beforehand as we mentioned. The tool also offers a function to analyze much longer syllabus and several syllabi all together in batch process. After a user uploads a zip file compressing multiple syllabus files, the tool executes Main Steps as a background job. Table V shows the execution times of Main Steps as background jobs. The background job takes about one

[2] https://rp3.ecc.u-tokyo.ac.jp/cs2013data/

[3] We use a virtual private server (CPU: 4 core and Memory: 4GB) offered by SAKURA Internet Inc. (http://vps.sakura.ad.jp/).

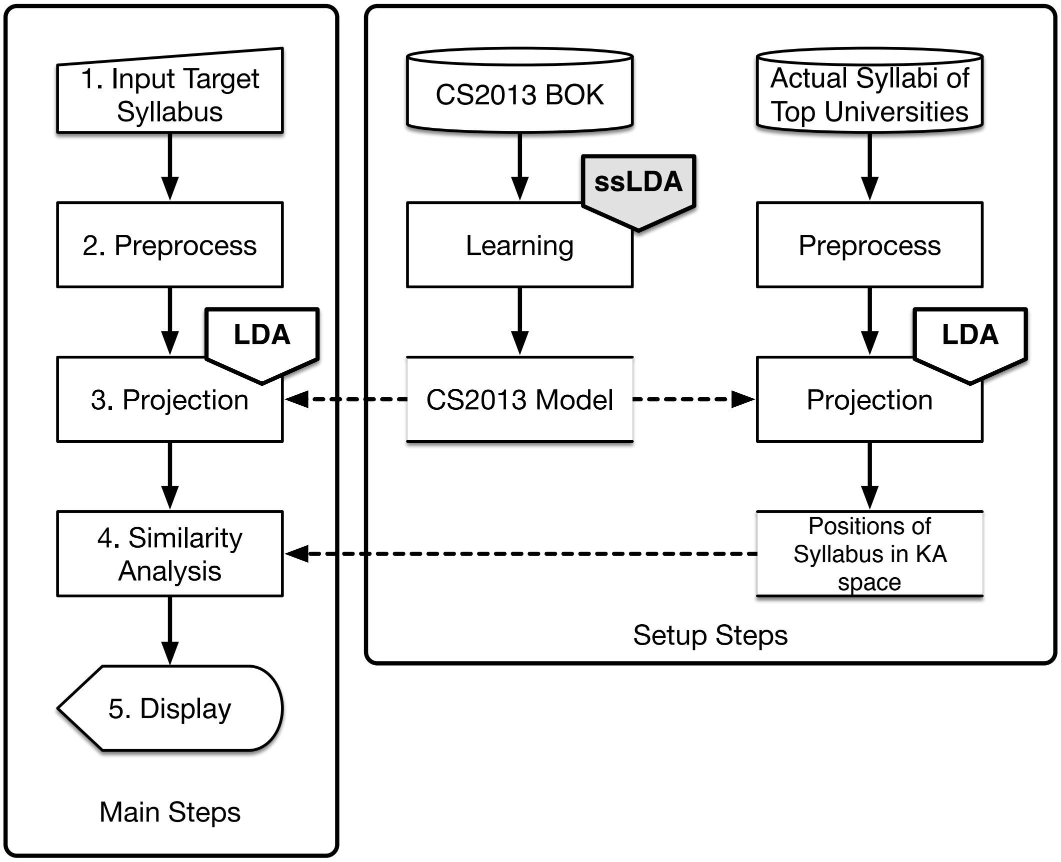

Fig. 3: The process of analyzing a course syllabus by the curriculum engineering tool.

TABLE V: The execution times of Main Steps as background jobs. "Num" means the number of Target Syllabus files which are compressed in a zip file. All Target Syllabus files are completely same, and each of the file contains 50 words of CS2013 BOK. "Pro(s)" and "Sim(s)" are the average of the five trials of "Projection" and "Similarity Analysis" respectively.

Num	Pro(s)	Sim(s)	Total(s)
10	2.4	4.6	7.0
20	4.4	8.8	13.2
50	11.0	20.6	31.6
100	22.6	42.0	64.6

minute if a user uploads a zip file which contains 100 syllabi.

In Fig. 4, the course description of "Human-computer interaction" held in Monash University is analyzed. This course is introduced as one of the course exemplars of HCI with in CS2013 report [2]. According to the result of the tool, this course is the most highly related to HCI and "HUMAN" Meta-topic (Table IV). This result corresponds to the description of this course in the CS2013 report.

As "Similar Courses," the courses with relatively high cosine similarities are listed. The most similar course is the course "Human-computer interaction" itself, and the second most similar one is also held by the same university. The second course is offered with a different course code, but its course description is almost the same as the first one. The third one is "Introduction to Human-Computer Interaction (University of Wisconsin-Madison)" and the fourth one is "Intro To HCI: User Interface Design, Prototype, Evaluation (University of Washington)." They are offered by difference universities, but the title of each course includes HCI-related words, and seem at least to be related to HCI. In this way, the tool can extract courses which are similar to Target Syllabus.

We show another example of a course offered by our university. The course "Introduction to Algorithms" is taught in Japanese, and its syllabus is originally written in Japanese. We translated the syllabus into English with Google Translate, and analyzed the translated text with the tool. Fig. 5 shows the result of the analysis. The value of AL is the highest in those of all topics, and the similar courses cover introductory topics in data structures and algorithms.

Our tool can extract similar courses regardless of length of text in each syllabus, because the tool deals with words used in CS2013 BOK, and calculates cosine

Fig. 4: The screen-shot of the analysis result of the course "Human-computer interaction" by the tool.

similarity among courses by use of position vectors of courses in Topic space. As Main Steps, a user only have to enter a syllabus.

In this way, students and teachers can understand what extent the course is related to each Topic (= KA of CS2013). They can also find similar courses offered by world top-ranked universities. Through these functions the tool helps students to grasp a course even if they do not have enough information on topics targeted by the course. The tool can also provide teachers with an objective estimation of their syllabi by CS2013. Teachers can improve their course syllabi based on KAs.

VI. Summary

In this paper, we developed a web-based tool for investigating syllabi, and demonstrated the effectiveness of the tool by examples. By just inputting a syllabus into our tool, the tool offers the strengths of relationships among the syllabi and KAs of CS2013, and also offers similar courses. Students and teachers can understand what extent the syllabus is related to each KA of CS2013 such as "Algorithms and Complexity (AL)." They can also find similar syllabi of the courses offered by the top-ranked universities. These information are beneficial for students to understand the courses and for teachers to

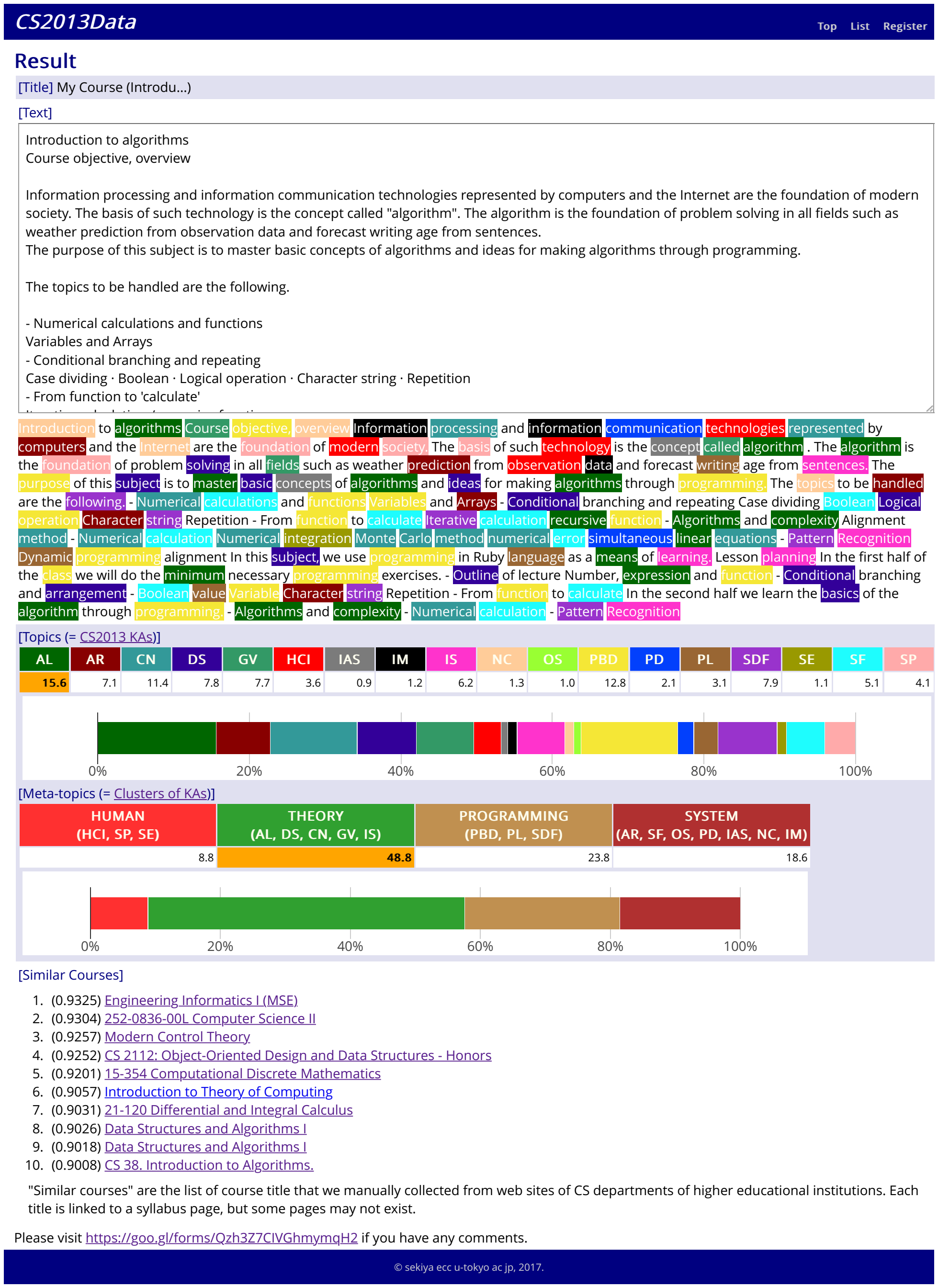

Fig. 5: The screen-shot of the analysis result of the course “Introduction to Algorithms” by the tool.

improve their syllabi.

In future work, we hope many teachers and students to use our tool (https://rp3.ecc.u-tokyo.ac.jp/cs2013data/) and to give some comments to us. To this goal, it is important to realize useful functions for them. For example, the tool will be able to assist students with their study, if the tool offer textbooks and references. We have a plan to implement such functions. For the present the tool has only about 3,000 syllabi, but we have a plan to collect more syllabi semi-automatically with web crawling techniques.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 17H01837.

References

[1] T. Sekiya, Y. Matsuda, and K. Yamaguchi, "Analysis of computer science related curriculum on LDA and isomap," in *ITiCSE'10, Proceedings of the 15th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 2010, pp. 48–52.

[2] ACM/IEEE-CS Joint Task Force on Computing Curricula, "Computer science curricula 2013," ACM Press and IEEE Computer Society Press, Tech. Rep., December 2013.

[3] T. Sekiya, Y. Matsuda, and K. Yamaguchi, "Curriculum analysis of CS departments based on CS2013 by simplified, supervised LDA," in *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*, ser. LAK '15. New York, NY, USA: ACM, 2015, pp. 330–339.

[4] Y. Matsuda, T. Sekiya, and K. Yamaguchi, "Discovery of interconnection among knowledge areas of standard computer science curricula by a data science approach," 2017, preprint.

[5] M. S. Wiske, M. Sick, and S. Wirsig, "New technologies to support teaching for understanding," *International Journal of Educational Research*, vol. 35, no. 5, pp. 483 – 501, 2001. [Online]. Available: http://www.sciencedirect.com/science/article/B6VDF-458P6TG-1/2/6414d7ffbc5d8d016382dc50b02b73f7

[6] Harvard Graduate School of Education, "The Collaborative Curriculum Design Tool (CCDT)," 2010, http://learnweb.harvard.edu/ccdt/ (Accessed February 22, 2010).

[7] M. Pedroni, M. Oriol, and B. Meyer, "A framework for describing and comparing courses and curricula," in *Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education*, ser. ITiCSE '07. New York, NY, USA: ACM, 2007, pp. 131–135.

[8] M. Tungare, X. Yu, W. Cameron, G. Teng, M. A. Perez-Quinones, L. Cassel, W. Fan, and E. A. Fox, "Towards a syllabus repository for computer science courses," in *Proceedings of the 38th SIGCSE technical symposium on Computer science education*, 2007, pp. 55 – 59.

[9] X. Yu, M. Tungare, W. Fan, Y. Yuan, M. Pérez-Quinones, E. Fox, W. Cameron, and L. Cassel, "Automatic syllabus classification using support vector machines," *Handbook of Research on Text and Web Mining Technologies. Information Science Reference*, 2008.

[10] M. Ida, "Textual information and correspondence analysis in curriculum analysis," in *Proceedings of the 18th international conference on Fuzzy Systems*. IEEE Press, 2009, pp. 666–669.

[11] S. Ota and H. Mima, "Machine learning-based syllabus classification toward automatic organization of issue-oriented interdisciplinary curricula," *Procedia - Social and Behavioral Sciences*, vol. 27, pp. 241–247, 2011.

[12] L. Marshall, "A comparison of the core aspects of the ACM/IEEE computer science curriculum 2013 strawman report with the specified core of CC2001 and CS2008 review," in *Proceedings of Second Computer Science Education Research Conference*, ser. CSERC '12. New York, NY, USA: ACM, 2012, pp. 29–34.

[13] R. Gluga, J. Kay, and R. Lister, "PROGOSS: Mastering the curriculum," in *Proceedings of The Australian Conference on Science and Mathematics Education (formerly UniServe Science Conference)*, 2012.

[14] C. Szabo and K. Falkner, "Neo-piagetian theory as a guide to curriculum analysis," in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '14. New York, NY, USA: ACM, 2014, pp. 115–120.

[15] Allen B. Tucker and Bruce H. Barnes and others, "Computing Curricula 1991 Report of the ACM/IEEE-CS Joint Curriculum Task Force," 1990.

[16] The Joint Task Force on Computing Curricula IEEE Computer Society/Asccociation for Computing Machinery, "Computing curricula 2001 computer science," 2001, http://www.acm.org/education/curric_vols/cc2001.pdf/view.

[17] The CS2008 Review Taskforce, "Computing science curriculum 2008: An interim revision of cs2001," 2008, http://www.acm.org/education/curricula/ComputerScience2008.pdf.

[18] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[19] C. Wang, D. Blei, and F.-F. Li, "Simultaneous image classification and annotation," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1903–1910.

[20] D. M. Blei and J. D. McAuliffe, "Supervised topic models," in *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. Curran Associates, Inc., 2008, pp. 121–128.

[21] Times Higher Education (THE), "World University Rankings," https://www.timeshighereducation.com/world-university-rankings (accessed August 8, 2016).

[22] J. H. Ward, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[23] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.

[24] Educational Testing Service, "ETS®Major Field Tests," https://www.ets.org/mft/ (accessed June 12, 2017).

[25] D. M. Blei, "Lda-c," http://www.cs.columbia.edu/~blei/lda-c/ (accessed March 23, 2017).