A Proposal of Credit Transfer System Using OWL in Education

Jiaji GUO[1] Hiromi KOBAYASHI[2]

This paper proposes a credit transfer system using Web Ontology Language (OWL) between universities. Credit transfer has been conducted mainly comparing and transforming curricula among universities. Mutual *1 : 1* comparisons are needed *n(n-1)/2* among n universities. We propose a credit transfer system using IEEE/ACM Computing Curricula as intermediate curricula in the area of computer science in order to reduce the number of mutual comparisons to *n*. Two transformation methods are presented for practical use of the curricula, which consist of the matrix of courses and the body of knowledge with teaching hours.

1. Introduction

The number of transfer students has increased into the second year class or the third year class in Japanese universities. These students have come from various educational institutions such as junior colleges, vocational schools, and foreign universities, recently. Credit transfer is a puzzling problem that is often not administered by faculty staff, but by professors in charge of admission because this work requires knowledge of subjects. The variety of transfer situations results engenders an increased workload of professors. The simplest solution of this problem is to admit the total amount of credits from a sending institution to a receiving one. This method is valid for liberal arts or general culture courses. However, systematic study is recommended for professional education or engineering courses. Consequently, some remedy for the laborious work of matching courses between the respective curricula of two institutions is needed. In addition, the number of comparisons and transformations is ${}_nC_2 = n(n-1)/2$ among curricula of n institutions. To overcome this problem, we propose a credit transfer system using IEEE/ACM Computing Curricula 2001 [1] (abbreviated as IEEE/ACM Curricula) as intermediate curricula in the area of computer science. The number of mutual transformations is reduced to n in this system. IEEE/ACM Curricula may be regarded as a de facto standard in the area of computer science because the curricula of some universities are based on IEEE/ACM Curricula; other institutions refer to these curricula.

We use an ontology description language Web Ontology Language (OWL) [2–8] to aim at transferral between curricula [semi]automatically. First, we propose a method of transforming names using *sameAs*, which means the same contents with different names in OWL. However, name transformations by this method are not generally sufficient. The reason is that every course offered by a university does not correspond to that in the IEEE/ACM Curricula as a *1 : 1* relationship. To resolve this problem, we propose another method using matrix of courses and body of knowledge (BOK) with teaching hours in the IEEE/ACM Curricula. This method is the way in which courses are matched $m : n$ via the total hours of BOK in all units. We propose a credit transfer system using these methods.

2. IEEE/ACM Curricula

IEEE/ACM Curricula exert an important effect on curricula of universities in the area of computer science. In these curricula, not only course names and topics, but also classified BOK with teaching hours are divided into core and elective hours. A pedagogical approach is considered, too. BOK consists of the following 14 areas having the core hours shown in parentheses. DS: Discrete Structures (43), PF: Programming Fundamentals (38), AL: Algorithms and Complexity (31), AR: Architecture and Organization (36), OS: Operating Systems (18), NC: Net-Centric Computing (15), PL: Programming Languages (21), HC: Human-Computer Interaction (8), GV: Graphics and Visual Computing (3), IS: Intelligent Systems (10), IM: Information Management (10), SP: Social and Professional Issues (16), SE: Software Engineering (31), and CN: Computational Science and Numerical Methods (0). Core hours represent the minimum teaching hours in each area. These areas are divided into units denoted by adding a numeric suffix to the area name. Each unit is subdivided into topics. Taking one example, course *CS103I* (Data Structures and Algorithms) comprises the following units. DS5: Graphs and trees (2), PF3: Fundamental data structures (12), PF4: Recursion (5), AL1: Basic algorithmc analysis (2), AL2: Algorithmic strategies (3), AL3: Fundamental computing algorithms (5), AL5: Basic computability (1), PL1: Overview of programming languages (1), PL6: Object-oriented programming (8), and SE6: Software validation (1).

Curricula are based on IEEE/ACM Curricula in a department of computer science at some universities; other departments such as computer science seem to produce curricula referring to IEEE/ACM Curricula. Therefore, IEEE/ACM Curricula may be regarded as a de facto standard and as the most influential ones in the field of computer science.

3. OWL

OWL is an ontology description language to process Web content information using machines automatically. OWL was recommended in 2004 by W3C.

[1] Department of Information Media Technology, Tokai University. 5kkc1016@keyaki.cc.u-tokai.ac.jp
[2] Regular Member. Department of Information Media Technology, Tokai University. koba@tokai.ac.jp

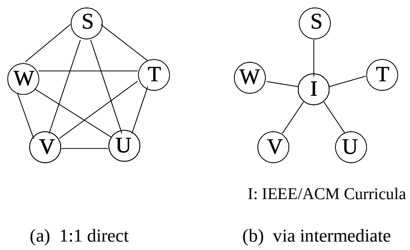

Fig. 1 Comparisons of curricula

The inclusion relationship of sets of resources can be represented explicitly using OWL. The syntax of OWL is usually represented as an RDF/XML style using tags. However, we use a functional style syntax called abstract syntax [5,8] for compact writing convenience. As an example, the equivalence of two classes: *Programming course* in University *S* and *CS111I* (Introduction to Programming) course in IEEE/ACM Curricula, are represented using OWL Full[3] in RDF/XML style as follows.

<owl:Class rdf:ID=``Programming''>
<owl: sameAs rdf:resource=``CS111I'' />
</owl: Class>

This is represented using abstract syntax as follows.[4]

SameIndividual(Programming CS111I)

4. Credit Transfer Methods

To the present, credit transfer has been conducted using *1:1* course matching among curricula of sending and receiving universities. Consequently, the laborious work of *1:1* matching is needed between curricula of two institutions. The number of matchings is ${}_nC_2 = n(n-1)/2$ among curricula of *n* institutions, as shown in Fig. 1(a). To overcome this problem, we propose a credit transfer system using IEEE/ACM Curricula as intermediate ones in the area of computer science. The number of mutual transformations is reduced to *n* in this system, as shown in Fig. 1(b). In addition, we use OWL to aim at automatic transformation among curricula.

4.1 Course Name Transformation

First, we propose a method using the notation of *sameAs*, which means the same contents with different names in OWL Full. A course name is not generally equivalent to that in IEEE/ACM Curricula in a university. Credit transfer means course name transformation between two institutions in this case. As an example, if a *Programming* course is equivalent to *CS111I (Introduction to Programming)* at university *S*, we can describe this equivalence using abstract syntax of OWL Full as follows.

[3] A class is also regarded as an individual in OWL Full.
[4] A function name of abstract syntax is not always equivalent to a tag name in RDF/XML style at present.

i ⟶ / j ↓

BOK area	unit	course MC1	MC2	MC3
BOKA(42)	BOKA1(18)	16		2
	BOKA2(14)	10		4
	BOKA3(10)	6		4
	BOKArest		2	8
BOKB(30)	BOKB1(14)	4	14	
	BOKB2(12)		12	
	BOKB3(4)		4	
	BOKBrest		8	
BOKC(14)	BOKC1(12)	4		10
	BOKC2(2)			4
	BOKCrest			8
total hours		40	40	40

Fig. 2 Model of Curricula

SameIndividual(Programming CS111I)

Whereas, if *Programming Techniques* course is also equivalent to *CS111I* at university *T*, we can describe this as follows.

SameIndividual(Programming Techniques CS111I)

These two expressions, written using OWL, readily engender the following result.

SameIndividual(Programming ProgrammingTechniques)

Only a simple transformation mechanism is required to obtain the above result because only *SameIndividual* is used in this method. When a student transfers from university *S* to *T*, acquired courses are transformed to course in IEEE/ACM Curricula at university *S*. Next, these transformed courses are sent to university *T*. Finally, these courses are transformed to courses in university *T*. A prototype of this system was developed using Jena[9]. In an actual system, the above descriptions using OWL must be conducted preliminarily using computer aided input tools in each university.

4.2 Course Name Transformation via BOK

Recently, outstanding characteristic is required in curricula at universities. On the other side, systematic study based on a standard curricula is demanded to foster professionals. For these discrepant requirements, every course is not always matched as 1:1 between that in curricula at a university and IEEE/ACM Curricula. In addition, each course comprises 40 classroom hours in IEEE/ACM Curricula while a typical course consists of about 20 classroom hours (1.5 h ×15 wk) in Japan. In this case, a course in IEEE/ACM Curricula must be divided into two courses as a vertical or horizontal division; i.e., a course is divided into the first half units and the second half ones, or entire units are divided into two. Therefore, we propose another method in which courses are corresponded to $m : n$ via the total BOK hours in each unit to cope with this problem.

We show a simple model of IEEE/ACM Curricula to explain this method, as shown in Fig. 2: MC1 to MC3 represent course names and BOKA to BOKC represent

```
<owl: Class   rdf: ID= "CS_Course" >
   <rdfs: subClassOf >
    <owl: Restriction >
        <owl: onProperty    rdf:resource= "#BOKA1_hours" / >
        <owl: allValuesFrom   rdf:datatype= "&xsd; nonNegativeInteger"/ >
        <owl: cardinality  rdf:datatype= "&xsd; nonNegativeInteger" > 1 </owl: cardinality>
    </owl: Restriction >
    </rdfs: subClassOf >

    <rdfs: subClassOf >
    <owl: Restriction >
        <owl: onProperty    rdf:resource= "#BOKA2_hours" / >
        <owl: allValuesFrom   rdf:datatype= "&xsd; nonNegativeInteger"/ >
        <owl: cardinality  rdf:datatype= "&xsd; nonNegativeInteger" > 1 </owl: cardinality>
    </owl: Restriction >
    </rdfs: subClassOf >
                           :
</owl: Class >

<owl:DatatypeProperty   rdf:ID="BOKA1_hours">
   <rdfs: domain  rdf:resource="#CS_Course"/>
   <rdfs: range     rdf:resource= "&xsd; nonNegativeInteger"/ >
</owl:DatatypeProperty>
                           :
<CS_Course  rdf:ID="MC1">
     <BOKA1_hours  rdf:datatype = "&xsd; nonNegativeInteger" > 16 </BOKA1_hours>
     <BOKA2_hours  rdf:datatype = "&xsd; nonNegativeInteger" > 10 </BOKA2_hours>
                           :
</CS Course>
```

(a)　RDF/XML style syntax

```
Class( CS_Course    partial
     restriction( BOKA1_hours  allValuesFrom( xsd:nonNegativeInteger )
                               cardinality(1)    )
     restriction( BOKA2_hours  allValuesFrom( xsd:nonNegativeInteger )
                               cardinality(1)    )
                                :
)
DatatypeProperty( BOKA1_hours
       domain( CS_Course )     range( xsd:nonNegativeInteger )   )
                               :
Individual( MC1  type( CS_Course )
                      value( BOKA1_hours "16"^^ &xsd;nonNegativeInteger )
                      value( BOKA2_hours "10"^^ &xsd;nonNegativeInteger )
                               :
)
```

(b)　Abstract syntax

Fig. 3　Ontology of Curricula Example

area names of BOK. Each unit name is represented adding a numeric suffix to the area name. However, the remaining units in each area with only elective hours (i.e., without core hours) are represented by adding a suffix *rest* to the unit name. Core hours are shown in parentheses in units and areas. As an example, at least 42 core hours are required in the BOK area of BOKA; at least 18, 14, and 10 hours are required for units of BOKA1, BOKA2, and BOKA3, respectively. $\mathrm{BOKA_{rest}}$ is an optional unit.

Course MC1 consists of 16, 10, 6, 4, 4 hours in the unit of BOKA1, BOKA2, BOKA3, BOKB1, and BOKC1, respectively. MC1 and its class called CS_Course can be represented using RDF/XML style syntax as shown in Fig. 3(a) and represented using an abstract syntax as shown in Fig. 3(b). A part of this description can be visualized using an RDF graph visualization tool called RDF Gravity [13] as shown in Fig. 4.

We assume a student transfers from university *S* to *T*. Let d_{ij} be teaching hours of a course in each unit at university *S*. The total teaching hours of each unit *j* is represented as

$$D_j = \sum_{i=1}^{m} d_{ij}$$

where *m* denotes the number of acquired courses at university *S*.

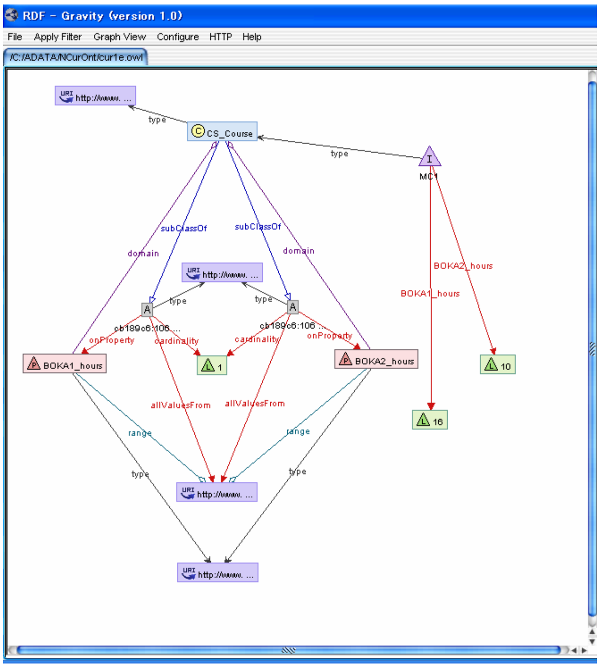

Fig. 4　RDF Graph Visualization

Every total teaching hours in each unit represented as $D_j (j = 1,2,\cdots,n)$ is transferred from university *S* to *T*, where *n* denotes the number of units in BOK.

Next, the corresponding courses are admitted to gain in university *T* such that select a subset of courses $\{c_1, c_2, \cdots, c_k\}$ from a set of l courses in university *T* in order to minimize the total difference of teaching hours in every unit between university *S* and *T*. Let c_{ij} be teaching hours of a course in each unit and l denote the number of courses in university *T*. This transformation can be represented as the following Knapsack Problem included in combinatorial optimization problems [10-12].

[Problem]　Assign 0 or 1 to x_i such that

$$\textit{Maximize} \quad f = \sum_{j=1}^{n} \sum_{i=1}^{l} c_{ij} x_i$$

$$\textit{subject to} \quad \sum_{i=1}^{l} c_{ij} x_i \le D_j \ (j = 1,2,\cdots,n)$$

except for optional units denoted $\mathrm{BOK^*_{rest}}$ in Fig. 2.

The difference of this problem from the original Knapsack Problem is that (1) each profit is equivalent to each weight denoted by c_{ij} and (2) the constraint of the maximum capacity is divided into that of every BOK denoted by $D_j \ (j = 1,2,\cdots,n)$. This method is the way in which courses are corresponded indirectly via total hours of BOK. We could solve this problem using a branch-and-bound method for several test cases as *m=10, n=10* and *L=20*.

5. Concluding Remarks

This paper proposed a credit transfer system using OWL between universities in computer science to aim at reducing the work of transfer. The number of mutual comparisons and transformations is reduced from $n(n-1)/2$ to n via IEEE/ACM Curricula among n universities. Advantages of OWL representation are strict definition of classes to reduce errors and utility of common vocabulary if inference is not used. Two transformation methods were presented for practical use of IEEE/ACM Curricula, which consist of matrix of courses and BOK with hours. First, we used OWL to aim at transforming course names between curricula automatically. However, this method is insufficient because all courses do not correspond in $1:1$ relationships with those in the IEEE/ACM Curricula.

Next, we proposed another method by which courses are mapped $m:n$ via the total hours of BOK. A prototype was developed and several test cases could be solved using a branch-and-bound method. However, heuristic methods with relaxation techniques are considered to be suited for a practical size problem. We have been developing program using genetic algorithms. Further work is needed to develop a support tool for data entry. Not only constructing data using OWL, but also the consent of usage of such an application is needed to diffuse ontology languages like OWL.

[References]

[1] IEEE/ACM Computing Curricula 2001 Computer Science: Final Report, IEEE Computer Society,2001.
[2] OWLOverview:http://www.w3c.org/TR/owl-features/
[3] OWL Guide: http://www.w3c.org/TR/owl-guide/
[4] OWL Reference: http://www.w3c.org/TR/owl-ref/
[5] OWL Semantics and Abstract Syntax: http://www.w3c.org/TR/owl-absyn/
[6] A. Gomez-Perez, M. Fernandez-Lopetz, and O. Corcho : "Ontological Engineering," Springer, 2004.
[7] D. Fensel : "Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce: Second Edition," Springer, 2004.
[8] G. Antoniou and F. van Harmelen : "A Semantic Web Primer," MIT Press, 2004.
[9] Jena Homepage:http://jena.sourceforge.net/
[10] D. Du and P.M. Pardalos eds. : "Handbook of Combinatorial Optimization," Kluwer Academic Publishers, 1998.
[11] C. R. Reeves ed. : "Modern Heuristic Techniques for Combinatorial Problems," Alfred Waller Ltd. U.K.,1993.
[12] S. M. Sait and H. Youssef: ``Iterative Computer Algorithms with Applications in Engineering - Solving Combinatorial Optimization Problems," IEEE Computer Society, 1999.
[13] RDF Gravity Homepage: http://semweb.salzburgresearch.at/apps/rdf-gravity/index.html

Jiaji GUO
Researcher, Graduated from Tokai University, received the Bachelor Degree of Engineering. His research interests include semantic Web and Web applications.

Hiromi KOBAYASHI
Professor, Department of Information Media Technology, School of Information Technology and Electronics, Tokai University. Graduated Doctor Engineering program from Graduate School of Engineering, Shinshu University.
Dr. Eng., His research interests include semantic Web, software engineering, and dependable systems.
Member of IEEE Computer Society, IEICE, IPSJ, etc.