FLEXIBLE DATA SEARCHS USING CONDITION FORMULAS

Toshio Kodama[1], Tosiyasu L. Kunii[2] and Yoichi Seki[3]

[1]*Venture Business Department, Maeda Corporation, 2-10-26 Fujimi, Chiyoda-ku, Tokyo, Japan*

[2]*Morpho, Inc., The University of Tokyo,7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan*

[3]*Software Consultant, 3-8-2 Sakaemachi, Hino City, Tokyo, Japan*

Keywords: Cellular model, formula expression, topological space, cellular space, condition formula.

Abstract: Cyberworlds are distributed systems on the Web, and are constantly evolving like living things, creating value. Currently, numerous Web business applications, such as cyberworld systems are being built, but in the development of the systems, combinatorial explosion happens because schemas and application programs must be modified whenever schemas change. We designed and implemented the logic of a flexible data search function by employing a condition formula on the cellular data system. This is the starting point to the implementation of the process graph theory, which makes a linear approach to overcoming combinatorial explosion possible.

1 INTRODUCTION

The system of cyberworlds is a distributed system. One of the features of cyberworlds is that data dependencies are constantly changing in them. Cyberworlds are more complicated and fluid than any other previous worlds in human history and are constantly evolving. For example, millions of users manage their own blog information every day through Web services on mobile phones, like SNS in Japan, which is considered one of main elements of cyberworlds. At the same time, user requirements for cyberworlds also change and get more complicated as cyberworlds change. If you analyze data using the existing technology in business application development, you have to modify the schema design and application programs whenever schemas or user requirements for output change. That leads to combinatorial explosion, because user requirements, and their combinations and schemas must be specified clearly at the design stage in general business application development. That is a fundamental problem, so we have to reconsider development from the data model level.

Is there a data model that can reflect the changes in schemas and user requirements for output to analyze data in cyberworlds? We believe that the cellular model proposed by one of authors (T. L. Kunii) is the most suitable model. The cellular model based on the Incrementally Modular Abstraction Hierarchy (IMAH) can model the architecture and the changes of cyberworlds and real worlds from a general level to a specific one, preserving invariants while preventing combinatorial explosion (T. L. Kunii and H. S. Kunii, 1999: 19-21). From the viewpoint of IMAH, existing data models are positioned as special cases. For example, UML can model objects at levels below the presentation level, and in the relational data model, a relation is an object at the presentation level which extends a cellular space because it has necessary attributes in which a type is defined, while the processing between relations is based on the set theoretical level. In the object-oriented model, an object is also the object in the presentation level, which extends a cellular space, while the relation between Class is the tree structure, which is a special case of a topological space. An Object in XML is considered a special case of a cellular space which extends a topological space, because an attribute and a value of it are expressed in the same tag format.

In our research, one of the authors (Y. Seki) proposed an algebraic system called Formula Expression as a development tool to realize the cellular model. T. Kodama has actually implemented CDS using Formula Expression (Toshio Kodama, Tosiyasu L. Kunii and Yoichi Seki, 2006: 65-74). In this paper, we have introduced a new concept of a

Kodama T., L. Kunii T. and Seki Y. (2008).
FLEXIBLE DATA SEARCHS USING CONDITION FORMULAS.
In *Proceedings of the International Conference on e-Business*, pages 21-28
DOI: 10.5220/0001906100210028

condition formula and its processing maps into CDS. A condition formula search is a very effective measure when you want to analyze data in cyberworlds without losing consistency in the entire system, since you can search for the data you want without changing application programs, if you employ a condition formula search. In addition, we put emphasis on practical use by taking up an example. First, we explain the cellular model briefly and add a new definition to Formula Expression. (Section 2) Second, we design logical operation as a condition formula generalizing search conditions of users by Formula Expression, and design its processing maps to process a condition formula to each topological space (Section 3) and implement them. (Section 4) We demonstrate the effectiveness of CDS by developing a business application system, thereby abbreviating the process of designing and implementing most application programs. (Section 5) It is a bidding results data search system where the data of files, which schemas differ, are inputted without designing schemas. A more flexible data search is possible by employing a condition formula search in the system.

2 THE CELLULAR MODEL AND FORMULA EXPRESSION

The following list is the Incrementally Modular Abstraction Hierarchy (IMAH) in the cellular model to be used for defining the architecture of cyberworlds and their modeling:

1. the homotopy (including fiber bundles) level
2. the set theoretical level
3. the topological space level
4. the adjunction space level (Fig 1)
5. the cellular space level
6. the presentation (including geometry) level
7. the view/projection level

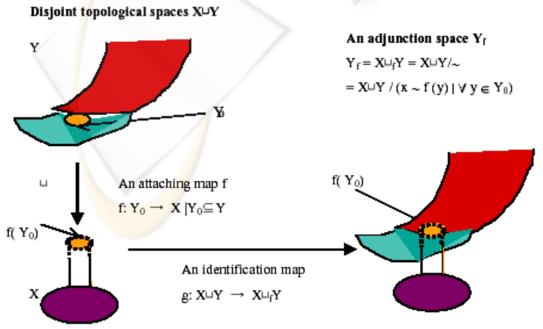

Disjoint topological spaces X⊔Y
Y
Y_0
⊔
An attaching map f
f: $Y_0 \to X \mid Y_0 \subseteq Y$
f(Y_0)
X
An adjunction space Y_f
$Y_f = X \sqcup_f Y = X \sqcup Y/\sim$
$= X \sqcup Y / (x \sim f(y) \mid \forall y \in Y_0)$
f(Y_0)
An identification map
g: $X \sqcup Y \to X \sqcup_f Y$

Figure 1: An example of e-manufacturing on an adjunction space level.

For a detailed explanation of each level, please refer to our earlier paper.

Formula Expression in the alphabet is the result of finite times application of the following (1)-(7).

(1) w ($\in \Sigma$) is Formula Expression
(2) unit element ε is Formula Expression
(3) zero element φ is Formula Expression
(4) when r and s are Formula Expression, addition of $r+s$ is also Formula Expression
(5) when r and s are Formula Expression, multiplication of $r \times s$ is also Formula Expression
(6) when r is Formula Expression, (r) is also Formula Expression
(7) when r is Formula Expression, {r} is also Formula Expression
(8) when r is Formula Expression, [r] is also Formula Expression

In this paper, we have added the 3rd bracket [] of (8) in the definition of Formula Expression. The algebraic structure is the following.

$$[r]\times(s+t) = [r]\times s+[r]\times t, (r+s)\times[t] = r\times[t]+s\times[t]$$

In this way, if [] is added to a formula and becomes the factor, it behaves like an identifier, since [] is never removed by any map.

3 THE DEFINITION OF LOGICAL OPERATION

3.1 A Condition Formula

If users can specify search conditions, data search will become more functional when searching data from data storage. Here, we introduce the function for specifying conditions defining a condition formula by Formula Expression into CDS. Let propositions P, Q be sets which include characters p, q respectively. The conjunction, disjunction and negation of them in logical operation are defined by Formula Expression as follows:

1) Conjunction

$$P \wedge Q = p \times q$$

2) Disjunction

$$P \vee Q = p+q$$

3) Negation

$$\neg P = !p$$

A formula created from these is called a condition formula. Here "!" is a special factor which means negation. Recursivity by () in Formula Expression is supported so that the recursive search condition of a user is expressed by a condition formula. An example is the following.

¬(P∨Q)∧((R∧S)∨(T∧U)) = !(p+q)(r×s+t×u)

3.2 A Quotient Acquisition Map and a Remainder Acquisition Map

A quotient acquisition map *f* is a map that has a term that includes a specified factor, and a remainder acquisition map *g* is a map that has a term that doesn’t include a specified factor. These two maps are fundamental in processing a condition formula. (3.3.) If you assume the entire set of terms to be A, B and the entire set of factors to be C, *f*:A×C ‖ B and *g*:A×C ‖ B. Arbitrary terms r, s, t, u, v, w, x, y (A) and an arbitrary factor p (C) follow these rules:

f: r, p ‖ φ (when r doesn’t include p)
f: r×p×s, p ‖ r×p×s
f: r×(s+t×p×u+v)×w, p ‖ r×t×p×u×w
f: r×{s+t×p×u+v}×w, p ‖ r×t×p×u×w
f: r×[s+t×p×u+v]×w, p ‖ r×[s+t×p×u+v]×w

g: r, p ‖ r (when r doesn’t include p)
g: r×p×s, p ‖ φ
g: r×(s+t×p×u+v)×w, p ‖ r×(s+v)×w
g: r×{s+t×p×u+v}×w, p ‖ r×{s+φ+v}×w
g: r×[s+t×p×u+v]×w, p ‖ φ

If p is an identifier, *f* (or *g*) is usually repeated until p is not enclosed in a bracket. Simple examples of both maps are shown below.

f: a(b(c+d(e+f))+g)h, d ‖ a×b×d(e+f)h
g: a(b(c+d(e+f))+g)h, d ‖ a(b×c+g)h

3.3 A Condition Formula Processing Map

A condition formula processing map *h* is a map that gets a disjoint union of terms which satisfies a condition formula from a formula. If you assume x to be a formula and x_i to be a term which consists of x (namely $_i x_i$ = x) and *p, !p, p+q, p×q* to be condition formulas, the images of (x, *p*), (x, *!p*), (x, *p+q*), (x, *p×q*), (x, *!(p+q)*), (x, *!(p×q)*) by *h* are the following:

h(x, *p*) = $_i$ *f*(x_i, *p*)
h(x, *p×q*) = $_i$ *f*(*f*(x_i, *p*), *q*)
h(x, *p+q*) = $_i$ *f*(x_i, *p*)+ $_i$ *f*(*g*(x_i, *p*), *q*)
h(x, *!p*) = $_i$ *g*(x_i, *p*)
h(x, *!(p+q)*) = $_i$ *g*(*g*(x_i, *p*), *q*)
h(x, *!(p×q)*) = $_i$ *g*(*f*(x_i, *p*), *q*)+ $_i$ *g*(x_i, *p*)

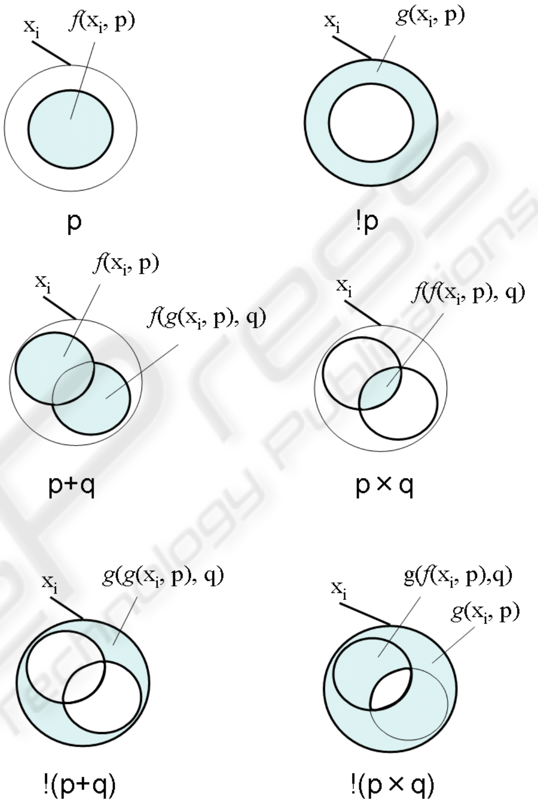


Figure 2: Images from the condition formula processing map.

Fig 2 is images of the map h. Here, *f* is a quotient acquisition map and g is a remainder acquisition map. It is obvious that any complicated condition formula can be processed by the combinations of the above four correspondences. A simple example is shown below.

x= "animal{color+size}(flesheating(bear{brown+big}+monkey{brown+small}+orangutan{darkbrown+big}+tiger{brown×black+big}+fox{brown×white+small}+bear{black+big})+grasseating(horse{(white+brown)+middle}+koala{brown+small}+goat{white+small}+hamster{white+verysmall}+panda{black×white+big}+zebra{black×white+middle}+giraffe{yellow×black+verybig}+elephant{gray+verybig}+mouse{gray+verysmall}))"

Output case 1.

User requirement: "information about a *horse* and a *zebra* in x is required"

A condition formula = "*horse*+*zebra*"

h(x, *horse*+*zebra*)
=*f*(x, *horse*)+*f*(*g*(x, *horse*), *zebra*)
=animal{color+size}grasseating(*horse*{(white+brown)+middle}+*zebra*{black×white+middle})

Output case 2.

User requirement: "information about animals whose *size* is *big* or *very big* and *grass-eating* is required"

A condition formula = "*size (big+verybig) grasseating*"

h(x, *size(big+verybig) grasseating*)
=*f*(*f*(*f*(x, *size*), *big+verybig*), *grasseating*)
=*f*(*f*(*f*(x, *size*), *big*)+*f*(*g*(*f*(x, *size*), *big*), *verybig*), *grasseating*)
=animal×*size*×*grasseating*(panda×*big*+giraffe×*very big*+elephant×*verybig*)

4 IMPLEMENTATION

This system is a web application developed using JSP and Tomcat 5.0 as a Web server. The client and the server are the same machine. (OS: Windows XP; CPU: Intel Core2 Duo, 3.00GHz; RAM: 3.23Gbyte; HD: 240GB)

Fig 3 is the flowchart of the algorithm of a quotient acquisition map which is the main function of a condition formula search. Details are abbreviated due to the restriction on the number of pages. In this algorithm, the absolute position of the specified factor by the function of the language and the term including the factor are acquired first. Next, the nearest brackets of the term are acquired and because the term becomes a factor, a recursive operation is done.

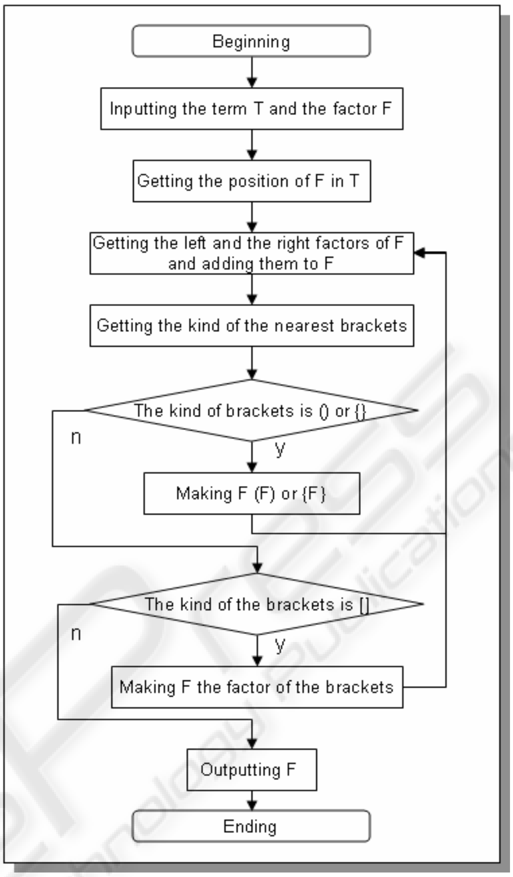

Beginning
Inputting the term T and the factor F
Getting the position of F in T
Getting the left and the right factors of F and adding them to F
Getting the kind of the nearest brackets
The kind of brackets is () or {}
n
y
Making F (F) or {F}
The kind of the brackets is []
n
y
Making F the factor of the brackets
Outputting F
Ending

Figure 3: The flowchart of the algorithm for a quotient acquisition map.

5 CASE STUDY: A BIDDING RESULTS DATA SEARCH SYSTEM

5.1 Outline

We have developed a business application system using CDS for searching bidding results data for public construction projects. Many of the data files were downloaded in CSV format from the official website of each bureau in the Ministry of Land, Infrastructure, Transport and Tourism (MLIT, http://www.mlit.go.jp/chotatsu/kekka/kekka.html) in Japan. The feature of the files is that their schema changes little from month to month or from bureau to bureau. Once you convert the CSV data files to formulas in CDS, you can unify them into a data storage file (.txt) by the function of a disjoint union +. After that, a user can search for the data she/he wants from the data storage by creating a condition formula. This system is actually being used in Maeda Corp. which one of authors (T. Kodama) belongs to.

5.2 The Space Design

We design a formula for the spaces as follows.

$$\Sigma \mathit{file}_{i}\times\mathrm{code}[\{\Sigma \mathit{attribute}_{i,j}\}](\Sigma \mathrm{k}[\{\Sigma \mathit{value}_{i,j,k}\}])$$

file_i: a factor which expresses a file name

$\mathit{attribute}_{i,j}$: a term which expresses an attribute name of file_i

value$_{i,j,k}$: a term which expresses a value of an *attribute*$_{i,j}$

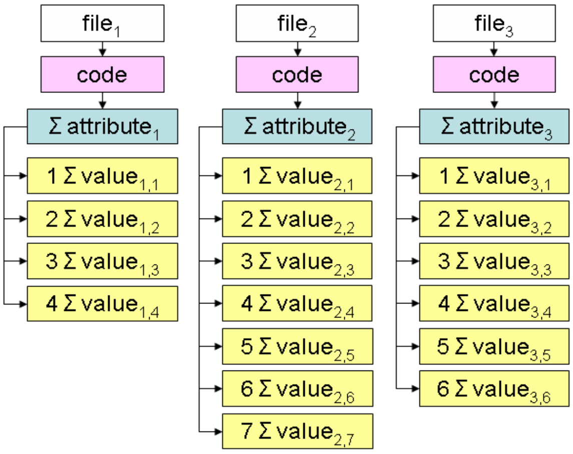


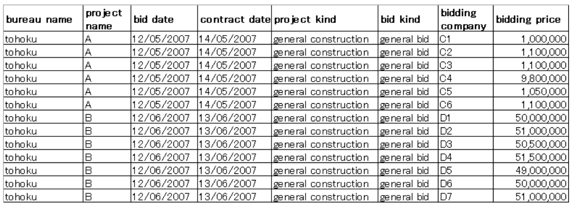
Figure 4: The data structure of the bidding results data search system.

5.3 Data Conversion and Data Input

In this subsection, we simplify the input data without losing generality. Let the CSV data of Figure 5 be bidding results from May, 2007 in the Tohoku bureau. First, convert the downloaded CSV data to a formula (formula 5.3-1) as a cell space and add it to the data storage file.

Formula 5.3-1.

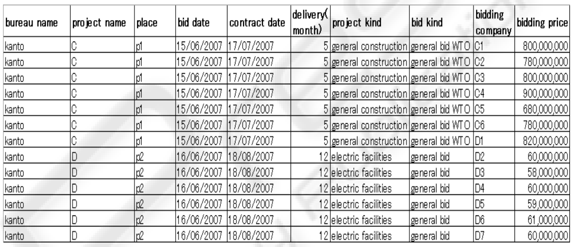
MayOf2007InTohoku×code[{bureau×name+project×name+bid×date+contract×date+project×kind+bid×kind+bidding×company+bidding×price}](1[{tohoku+A+12/05/2007+14/05/2007+general×construction+general×bid+C1+1000000}]+2[{tohoku+A+12/05/2007+14/05/2007+general×construction+general×bid+C2+1100000}]+3[{tohoku+A+12/05/2007+14/05/2007+general×construction+general×bid+C3+1100000}]+4[{tohoku+A+12/05/2007+14/05/2007+general×construction+general×bid+C4+9800000}]+5[{tohoku+A+12/05/2007+14/05/2007+general×construction+general×bid+C5+1050000}]+6[{tohoku+A+12/05/2007+14/05/2007+general×construction+general×bid+C6+1100000}]+7[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×biC×WTO+C1+50000000}]+8[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C2+51000000}]+9[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C3+50500000}]+10[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C4+51500000}]+11[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C5+49000000}]+12[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C6+50000000}]+13[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C7+51000000}])



bureau name	project name	bid date	contract date	project kind	bid kind	bidding company	bidding price
tohoku	A	12/05/2007	14/05/2007	general construction	general bid	C1	1,000,000
tohoku	A	12/05/2007	14/05/2007	general construction	general bid	C2	1,100,000
tohoku	A	12/05/2007	14/05/2007	general construction	general bid	C3	1,100,000
tohoku	A	12/05/2007	14/05/2007	general construction	general bid	C4	9,800,000
tohoku	A	12/05/2007	14/05/2007	general construction	general bid	C5	1,050,000
tohoku	A	12/05/2007	14/05/2007	general construction	general bid	C6	1,100,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D1	50,000,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D2	51,000,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D3	50,500,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D4	51,500,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D5	49,000,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D6	50,000,000
tohoku	B	12/06/2007	13/06/2007	general construction	general bid	D7	51,000,000

Figure 5: CSV data of bidding results from May 2007 in the Tohoku bureau.

Next, add the CSV data from June in the Kanto bureau, which schema is slightly different from that in formula 5.3-1, convert it to a formula (formula 5.3-2) in the same way and add it to the data storage file using + function.



bureau name	project name	place	bid date	contract date	delivery(month)	project kind	bid kind	bidding company	bidding price
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C1	800,000,000
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C2	780,000,000
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C3	800,000,000
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C4	900,000,000
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C5	680,000,000
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C6	780,000,000
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	D1	820,000,000
kanto	D	p2	16/06/2007	18/08/2007	12	electric facilities	general bid	D2	60,000,000
kanto	D	p2	16/06/2007	18/08/2007	12	electric facilities	general bid	D3	58,000,000
kanto	D	p2	16/06/2007	18/08/2007	12	electric facilities	general bid	D4	60,000,000
kanto	D	p2	16/06/2007	18/08/2007	12	electric facilities	general bid	D5	59,000,000
kanto	D	p2	16/06/2007	18/08/2007	12	electric facilities	general bid	D6	61,000,000
kanto	D	p2	16/06/2007	18/08/2007	12	electric facilities	general bid	D7	60,000,000

Figure 6: CSV data of bidding results from June 2007 in the Kanto bureau.

Formula 5.3-2.

(formula5.3-1)+MayOf2007InKanto×code[{bureau×name+project×name+place+bid×date+contract×date+delivery-month+project×kind+bid×kind+bidding×company+bidding×price}](1[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C1+800000000}]+2[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C2+780000000}]+3[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C3+800000000}]+4[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C4+900000000}]+5[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C5+680000000}]+6[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C6+780000000}]+7[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C1+820000000}]+8[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C2+60000000}]+9[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C3+58000000}]+10[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C4+60000000}]+11[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C5+59000000}]+12[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C6+61000000}]+13[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C7+60000000}])

In this way, you can add data to the data storage after converting it to a formula for a cell space using

+ function. In doing this, you don’t need to consider differences in schema at all.

In the same way, you can add data from another organization, which schema is completely different from others', as a formula for a cell space to the data storage file.

5.4 Data Conversion and Data Input

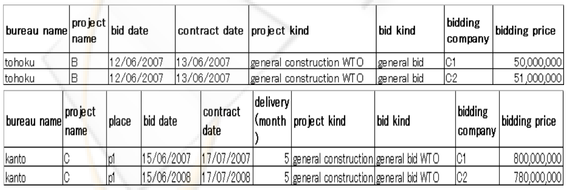
When you search for data you want, you create condition formulas according to requirements and get an image of the formula in data storage by the condition formula processing map *h*, you can get the data you want. Examples and figures (Fig 7,8) are shown below.

If you want to search for data for "construction projects of Company *C1* or *C2* and for *WTO* (World Trade Organization) property", you make the condition formula "*(C1+C2)WTO*", and get the image of formula 5.3-2 by the condition formula processing map *h*.

h (*formula 5.3-2*, *(C1+C2)WTO*)

= MayOf2007InTohoku×code[{bureau×name+project×name+bid×date+contract×date+project×kind+bid×kind+bidding×company+bidding×price}](7[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×biC×WTO+C1+50000000}]+8[{tohoku+B+12/06/2007+13/06/2007+general×construction+general×bid×WTO+C2+51000000}])+MayOf2007InKanto×code[{bureau×name+project×name+place+bid×date+contract×date+delivery-month+project×kind+bid×kind+bidding×company+bidding×price}](1[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C1+800000000}]+2[{kanto+C+p1+15/06/2007+17/07/2007+5+general×construction+general×bid×WTO+C2+780000000}])

Next, if you want to search for data for "construction projects in the Kanto bureau which are not for WTO", you create the condition formula "*kanto×!WTO*", and get the image of formula 5.3-2 by the map *h*.



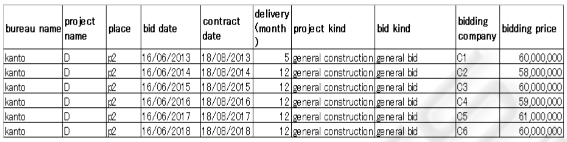
bureau name	project name	bid date	contract date	project kind	bid kind	bidding company	bidding price
tohoku	B	12/06/2007	13/06/2007	general construction WTO	general bid	C1	50,000,000
tohoku	B	12/06/2007	13/06/2007	general construction WTO	general bid	C2	51,000,000

bureau name	project name	place	bid date	contract date	delivery (month)	project kind	bid kind	bidding company	bidding price
kanto	C	p1	15/06/2007	17/07/2007	5	general construction	general bid WTO	C1	800,000,000
kanto	C	p1	15/06/2008	17/07/2008	5	general construction	general bid WTO	C2	780,000,000

Figure 7: The output result by the condition formula "(C1+C2) WTO".

h (*formula 5.3-2*, *kanto×!WTO*)

= MayOf2007InKanto×code[{bureau×name+project×name+place+bid×date+contract×date+delivery-month+project×kind+bid×kind+bidding×company+bidding×price}](8[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C2+60000000}]+9[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C3+58000000}]+10[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C4+60000000}]+11[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C5+59000000}]+12[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C6+61000000}]+13[{kanto+D+p2+16/06/2007+18/08/2007+12+electric×facilities+general×bid+C7+60000000}])



bureau name	project name	place	bid date	contract date	delivery (month)	project kind	bid kind	bidding company	bidding price
kanto	D	p2	16/06/2013	18/08/2013	5	general construction	general bid	C1	60,000,000
kanto	D	p2	16/06/2014	18/08/2014	12	general construction	general bid	C2	58,000,000
kanto	D	p2	16/06/2015	18/08/2015	12	general construction	general bid	C3	60,000,000
kanto	D	p2	16/06/2016	18/08/2016	12	general construction	general bid	C4	59,000,000
kanto	D	p2	16/06/2017	18/08/2017	12	general construction	general bid	C5	61,000,000
kanto	D	p2	16/06/2018	18/08/2018	12	general construction	general bid	C6	60,000,000

Figure 8: The output result by the condition formula "kanto×!WTO".

If you want to get attribute values by specifying an attribute name, you remove "[]" once from the formula and get the image by the quotient acquisition map *f*. An example is shown.

If you want to search for data for "values of an attribute of *bid date* in the *Tohoku* bureau", you get the image by the composition map of *f* and *h*. Assume that *formula 5.3-2'* is the formula after removing all "[]" from the formula 5.3-2.

f (*h* (*formula 5.3-2'*, *kanto*), *project×name*)

=MayOf2007InKanto×code{bid×date}(1{15/06/2007}+2{15/06/2007}+3{15/06/2007}+4{15/06/2007}+5{15/06/2007}+6{15/06/2007}+7{15/06/2007}+8{16/06/2007}+9{16/06/2007}+10{16/06/2007}+11{16/06/2007}+12{16/06/2007}+13{16/06/2007})

5.5 Considerations

When a business application system like the one above is developed in the existing way, user requirements are analyzed first. Next, the system, schemas and application programs are designed according to requirement analysis. Then, implementation and testing are done. The fundamental development process is changed if CDS is used.

1. Schema Design and Data Input

It is almost impossible for a database designer to design schema of this application system since she/he cannot predict the changes in schema of MLIT bidding results data. And whenever a new file which schema is different from that already designed appears, it is actually impossible to modify the schema design and application programs or to develop data conversion programs. If you employ

CDS in the development of this application system, you don't have to worry about the above problems. This is because the concept of the disjoint union + of the cellular model is supported in CDS, so that you can add the data which schema are different to the data storage one after another, if you only have to convert the data to formulas of CDS.

2. Data Output

Data output design and application programs for data output have to be done during application system development, and they have to be modified when there is a new user requirement for output which was not expected in the user requirement analysis. This can be costly. But if you use CDS in the development, a user only has to create a condition formula according to a user requirement for output. This is because user requirements can be generalized by condition formulas of CDS.

3. Processing Speed

Detailed benchmark tests have not been conducted yet, but when we actually tried this system, the output processing speeds of 500 records and 1,000 records from more than 200,000 records were 3.2s and 6.7s respectively. This system is considered practical for analyzing business data on a client PC.

6 RELATED WORKS

The distinctive features of our research are the application of the concept of topological process, which deals with a subset as an element, and that the cellular space extends the topological space, as seen in Section 2. Relational OWL as a method of data and schema representation is useful when representing the schema and data of a database (Takashi Washio and Hiroshi Motoda, 2003: 59-68), but it is limited to representation of an object that has attributes. Our method can represent both objects: one that has attributes as a cellular space and one that doesn't have them as a set or a topological space. Many works applying other models to XML schema have been done. The motives of most of them are similar to ours. The approach in (Giovanna Guerrini, Marco Mesiti, Daniele Rossi, 2005: 39-44) aims at minimizing document revalidation in an XML schema evolution, based on a part of the graph theory. The X-Entity model (Bernadette Farias Lósio, Ana Carolina Salgado and Luciano do Rêgo GalvĐo, 2005: 39-44) is an extension of the Entity Relationship (ER) model and converts XML schema to a schema of the ER model. In the approach of (N. Routledge, L. Bird and A. Goodchild, 2002: 157-166), the conceptual and logical levels are represented using a standard UML class and the XML represents the physical level. XUML (HongXing Liu, YanSheng Lu, Qing Yang, 2006: 973-976) is a conceptual model for XML schema, based on the UML2 standard. This application research concerning XML schema is needed because there are differences in the expression capability of the data model between XML and other models. On the other hand, objects and their relations in XML schema and the above models can be expressed consistently by CDS, which is based on the cellular model. That is because the tree structure, on which the XML model is based, and the graph structure, on which the UML and ER models are based, is special cases of a topological structure mathematically. Entity in the models can be expressed as the formula for a cellular space in CDS. Moreover, the relation between subsets, as we showed in 3.2, cannot in general be expressed by XML. Although CDS and the existing deductive database look alike apparently, the two are completely different. The deductive database (Q. Kong, G. Chen, 1995: 973-976) raises the expression capability of the relational database (RDB) by defining some rules. On the other hand, CDS is a proposal for a new tool for data management and has nothing to do with the RDB. Plenty of CASE tools are currently available, but they support system development according to existing data models. The differences from CDS are mainly that we apply a novel model, the cellular model, for building CDS, and that the customer side can confirm the output by changing formulas using the defined maps after creating formulas as the input.

7 CONCLUSIONS AND FUTURE WORKS

In this paper, we have developed a condition formula search as an important function of CDS. Using this function of CDS, you can search for data you want from formulas as data storage by creating a condition formula according to user requirements, so that you don't have to analyze user requirement for output in typical business application development. The point we should emphasize for future work is that the search condition of a user as well as data for input/output is expressed as a formula. This certainly brings the system which is developing, including user requirements recursively. This will be connected to the implementation of a process graph

(T. L. Kunii, 2003: 86-96). It is the next step where a situation as a node is transferred to the next situation selecting a path as an edge. Implementation has been difficult up to the present time because there is no tool to realize it, although one of authors (T.L. Kunii) outlined the plan many years ago. The appearance of Formula Expression will enable it in the near future. If we implement the process graph by developing CDS as future work, automation of business application development will be done. We believe that CDS brings great social impact, changing existing development fundamentally. Our research is still in its infancy, but it is progressing every day. We are collaborating with companies and universities worldwide.

REFERENCES

T. L. Kunii and H. S. Kunii, "A Cellular Model for Information Systems on the Web - Integrating Local and Global Information", *In Proc. of DANTE'99*, IEEE Computer Society Press, pp.19-24, 1999.

Bernadette Farias Lósio, Ana Carolina Salgado, Luciano do Rêgo GalvĐo, "Conceptual modeling of XML schemas", *In Proc. of WIDM'03*, ACM Press, pp.102-105, 2003.

Takashi Washio, Hiroshi Motoda, "State of the art of graph-based data mining", *In ACM SIGKDD Explorations Newsletter*, ACM Press, pp.59-68, 2003.

David W. Embley, "Toward semantic understanding: an approach based on information extraction ontologies", *In Proc. of ADC'04*, Australian Computer Society, Inc., pp.3-12, 2004.

Cristian Pérez de Laborda, Stefan Conrad, "Relational.OWL: a data and schema representation format based on OWL", *In Proc. of APCCM '05*, Australian Computer Society, Inc., pp.89-96, 2005.

Nicholas Routledge, Linda Bird, Andrew Goodchild, "UML and XML schema", *In Proc. of ADC'02*, Australian Computer Society, Inc., pp.157-166, 2002.

Sean K. Bechhofer, Jeremy J. Carroll, "Parsing owl dl: trees or triples?", *In Proc. of WWW'04*, ACM Press, Inc., pp.266-275, 2004.

Giovanna Guerrini, Marco Mesiti, Daniele Rossi, "Impact of XML schema evolution on valid documents", *In Proc. of WIDM'05*, ACM Press, pp.39-44, 2005.

Bernadette Farias Lósio, Ana Carolina Salgado, Luciano do Rêgo GalvĐo, "Conceptual Modeling of XML Schemas", *In Proc. of WIDM'03*, ACM Press, pp.102-105, 2003.

HongXing Liu, YanSheng Lu, Qing Yang, "XML conceptual modeling with XUML", *In Proc. of ICSE'06*, ACM Press, pp.973-976, 2006.

Q. Kong, G. Chen, "On deductive databases with incomplete information", *In ACM TOIS*, pp.354-370, 1995.

Toshio Kodama, Tosiyasu L. Kunii, Yoichi Seki, "A New Method for Developing Business Applications: The Cellular Data System", *In Proc of CW'06*, pp. 65-74, IEEE Computer Society Press. http://www.mlit.go.jp/chotatsu/kekka/kekka.html (22 Apr. 2008)

Tosiyasu L. Kunii, "What's Wrong with Wrapper Approaches in Modeling Information System Integration and Interoperability? ", In Proc of DNIS2003, pp. 86-96, Lecture Notes in Computer Science, Springer-Verlag.