DIAGRAM GENERATION FROM TAGGED TEXTS TOWARD DOCUMENT NAVIGATION

Masashi Murayama *Yuichi Nakamura* *Yuichi Ohta*

IEMS, University of Tsukuba, Tennodai, Tsukuba 305-8573 JAPAN
E-mail: murayama@image.esys.tsukuba.ac.jp

ABSTRACT

We often need too much time for reading documents, since it is often difficult to efficiently grasp its outline. For this purpose, we propose our diagram generation scheme for presenting the structures of a text. The semantic structure of a tagged text is effectively translated to diagrams, and they are linked to the text. In this paper, first, we describe how semantic structures can be expressed by a diagram. Then, we propose our framework for automatic diagram generation from tagged texts to diagrams.

1 INTRODUCTION

Recently, we have easy access to a large amount of electronic documents through intra-network, and the Internet. Although this gives us great benefits, it is often difficult to access the relevant portions, since reading is often a time-consuming work.

Because of the linear structure of a natural language, we are often forced to write or read complicated texts: three or more words with tight semantic relations are often apart in different sentences; chains of important relations such as *cause and effects* spread all over a text and mixed up with other relations.

A diagram is a powerful tool in such situations. Diagrams effectively navigate the readers with the following features:

- Relationships among three or more elements can be easily represented by two or higher dimensional structures.
- Different kinds of relationships can be clearly represented in a same diagram by giving different characteristics to the elements.
- It is usually easy to adapt diagrams to the user's need, since it is easy to add, remove, or emphasize elements.

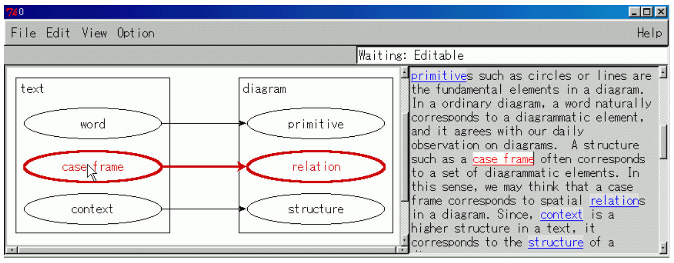
A diagram, however, is not enough to completely substitute for a text, since it is not suitable for expressing detailed meaning. We need media complex of texts and diagrams, and they should be tightly related or linked together.

For this purpose, we propose a novel framework:

- Embed tags for marking up the essential semantic structure of a text
- Generate diagrams that are tightly linked to the text.
- Give a user interface that presents both the text and the diagrams, and enables quick access to the essence of the text.

A simple example is shown in Fig.1. A comprehensible diagram on the left side is generated from a tagged text[1] on the right side. In this example, as the user specified some portion that drew the user's attention, the corresponding portion of the text is highlighted. Thus, the user can quickly access to the relevant information.

[1]The tags are invisible in this viewer, because we assume that the users are not good at reading mark-up languages.



primitives such as circles or lines are the fundamental elements in a diagram. In a ordinary diagram, a word naturally corresponds to a diagrammatic element, and it agrees with our daily observation on diagrams. A structure such as a case frame often corresponds to a set of diagrammatic elements. In this sense, we may think that a case frame corresponds to spatial relations in a diagram. Since, context is a higher structure in a text, it corresponds to the structure of a

Figure 1: Document navigation by diagram generation

One of the possible applications is the authoring of electronic manuals, teaching materials, etc. For example, an author writes texts with a semi-automatic diagram generation process, and readers will take a look at the diagrams for the navigation of a document.

In the following sections, we will first present the basic idea of our diagram generation scheme in section 2 and 3. In section 4, we will describe the correspondence between diagrammatic structures and our XML tag set. Finally, we will briefly show our diagram generating process and our experiments.

2 DIAGRAM GENERATION FOR A TEXT

For the above framework, we have to realize two important processes:

(a) Semantic structure extraction from a text, and tagging to the text.
(b) Diagram generation from a tagged text.

Process (a) is basically a process on natural language processing, which detects typical semantic structures of a text. The most possible method is semi-automatic tagging in which the author modifies the outputs of natural language processing. For example, Global Document Annotation (GDA) [4] aims at marking up semantic and pragmatic structures of documents, and helper applications for semi-automatic tagging are already developed. Moreover, since tagged texts such as XML documents are becoming popular these days, it is reasonable to expect a considerable amount of semantically-tagged texts will be available in the near future.

Process (b) is a process of translating semantic descriptions of a text to diagrammatic expressions such as elements or arrangements. In this step, a problem arises from the wide variety of diagram formats. If we arbitrarily put them together, diagrams could be complicated and misleading. To deal with this problem, we studied the linguistic functions of diagrams, and defined the rules for appropriately translating the semantic structures in a text to a diagram.

Thus, we focus on, in this paper, the following points:

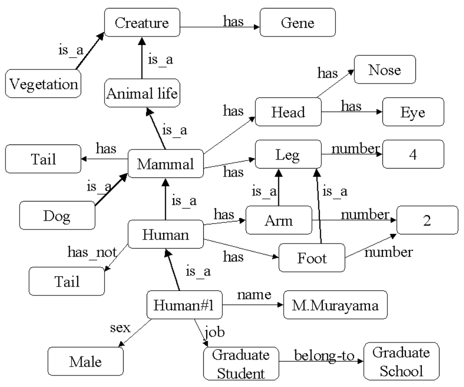


Figure 2: Example of complicated semantic network

- A XML tag set for representing semantic structures of a text.
- A diagram generation process from a tagged text.

3 RELATED WORKS

There has been a lot of research on text-to-text summarization [1] [5]. The most important sentences are selected or generated as the summary of a text. Although this text-to-text summarization is natural, summaries are often difficult to understand, and they sometimes miss the information for which we are looking. As an alternative or complementary framework, we propose our scheme of diagram generation.

Other important works are concerning graphical representation of semantic networks. Basically, a semantic network can be illustrated by using nodes and arcs. More advanced representations are recently proposed as conceptual graphs[6]. Thus, we can illustrate variety of ideas by using graph structures[3]. However, as you may see in a relatively large semantic networks as shown in Fig.2, it becomes confusing and not comprehensible if we use arbitrary relations without any restrictions. In this sense, a diagram is not sufficient to substitute a text.

On the contrary, our research aims at giving supports for reading documents. Diagrams must basically be based on texts, and diagrams must be tightly linked to texts. We chose, therefore, the XML tagging to a text. Moreover, we investigated how naturally semantic structures are expressed by diagrams, and defined the tag set for marking up. Based on these ideas, we are investigating how typical manuals or articles can be tagged and translated to diagrams.

4 TAGS FOR SEMANTIC STRUCTURES

4.1 MARKING UP BY TAGS

We propose a tag set for representing semantic structures in a text. The tag set is the extension of XML[2], and we can use ordinary tools for XML documents. A tag specifies which portion the system illustrates in a diagram, how the system draws the portion, and what kind of semantics the portion has.

Basically, a tag may have one of the following two functions.

- Attach a name or an identifier to an element, which may be a word, a phrase, or a sentence, etc.
- Mark up a semantic relation among elements. In this case, a tag is denoted by a set of "structural category", "semantic category", and elements that hold the relation.

Before showing actual examples, we will describe how we defined "structural category" and "semantic category".

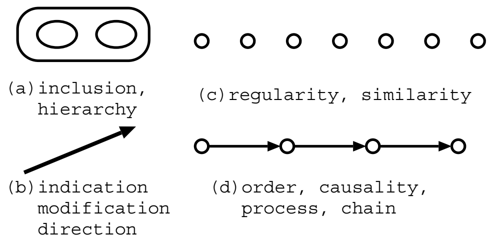

(a) inclusion, hierarchy
(c) regularity, similarity
(b) indication modification direction
(d) order, causality, process, chain

Figure 3: Example of linguistic functions of a diagram

Table 1: Structural classification

Order	order, series, causality order, hierarchy, inclusion, subordination, etc.
Equivalence	equivalence, equality, coordination, etc.
Modification	explanation, attribution, etc.
Binomial	general binary relations (excepting the above).
Complex	general n-ary relations (excepting the above), such as comparison, condition, etc.

4.2 STRUCTURAL CLASSIFICATION

In texts, there are various kinds of structures from syntactic structures to deep semantics. Similarly, there are various implicit rules in drawing diagrams, as shown in Fig.3. A primitive in a diagram expresses a notion, or a combination of them. The diagrammatic structure represents the relations among them. For example, area inclusion shows hierarchy or subordination. An arrow shows direction, order, or modification. A row of primitive diagrams shows flow, order or weak relation. Although those linguistic functions are not strict, the user can easily and quickly grasp the outline when the usage of the diagrammatic structures is consistent with the semantics of a text.

Based on this idea, we determined the structural category of tags as shown in Table1. These are typical relations that we often see in many diagrams. *Order* is the most common category which expresses a chronological order, an order of processes, hierarchy, etc. *Equivalence* means equality, and *modification* means that one element is adding information to another element. *Binomial* and *complex* are the categories for which any of the above types does not fit.

This classification is partially based on the algebraic functions. *Order* satisfies transitive law, and *equivalence* satisfies reflective law and symmetric law. *Modification* satisfies reflective law. The remaining two satisfies no explicit laws. This algebraic classification is useful for determining how the system draws a diagram. Suppose that two *order* type relations, that are A→B and B→C, are properly tagged and the semantics defined below are consistent. The system can safely merge them into one sequence, that is A→B→C.

4.3 SEMANTIC CLASSIFICATION

We need semantic classification of relations. A semantic category is useful in diagram construction as mentioned above. Possible chains of relations are easily found and can be merged if necessary. Another usage is for information filtering. If we need to find something about the chronological order, other kinds of orders such as spatial order, hierarchy, or subordination are useless. Then, we may want to emphasize the elements with the relevant relations, or we may want to filter out the elements for expressing

Table 2: Semantic classification

Time	chronology
Causality	cause and effect
Space	location or locus in physical world
Input/output	material, product, input and output, etc.
Process	process, flow of things, flow of topic, etc.
Set	organization, theory of sets, subordination, etc.
Others	others

```
<n id="xml">XML</n> consists of
<n id="dec">XML declaration</n>,
<n id="dtd">DTD</n> and
<n id="ins">XML instance</n>.
<slink structure="order" semantics="set"
  term1="xml" term2="dec dtd ins" />
```

Figure 4: Sample tagged text

useless relations.

Table2 are the semantic categories that we are currently using. For example, *time* represents general chronological relations. Similarly, *causality*, and *space* are common categories of relations which we often explain by illustrating diagrams. Although *input/output* may sound unfamiliar, the notion of data input/output to/from a module or a device is common in computer science.

Each relation, that is a semantic structure, is specified by a combination of a structural category and a semantic category. Some of the combinations such as { *order* & *causality* } often appears in our experiments, while { *modification* & *input/output* } is less useful.

4.4 EXAMPLE OF MARKING UP

Here, we show an example in Fig.4. The first tag marks up a word `XML` with identifier `xml`. The second, third, and fourth tags are used for the same purpose. The fifth tag is the representation of a semantic relation. It represents that the structural category is *order*, and the semantic category is *set*. The first element that holds the relation is `XML`, and the second element is the other three words. As a result, we can denote the semantics that `XML` consists of `XML declaration`, `DTD` and `XML instance`.

Fig.5 shows an additional tagged text. It describes the components of an element in the above example. Thus, a nest of *set* attribute is given by those two examples. The diagram generation result will be presented in section 6.

Another example is shown in Fig.6. This example is mainly composed of the relation on { *order* & *process* }.

5 DIAGRAM GENERATION

Initially, one diagrammatic element is created for one tagged element in a text. Then, additional diagrammatic elements are created according to semantic relation descriptions.

Since direct translation from tagged texts to diagrams is difficult, we use two kinds of intermediate descriptions.

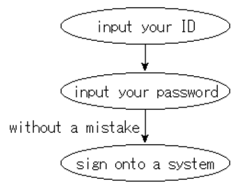
Network description: This description is a 2-dimensional array, which holds attributes and relations of each element together. A diagonal element a_{ii} holds the attributes of element e_i. A non-diagonal element a_{ij} holds the relations between two elements, that is e_i and e_j. By putting all relations and attributes concerning each element together in one place, the system can easily find the constraints on the elements, and find the possibility of manual modification as described below.

```
DTD consists of two declaration:
<n id="ele">ELEMENT</n> and
<n id="att">ATTLIST</n>.
<slink structure="order" semantics="set"
  term1="dtd" term2="ele att" />
```

Figure 5: Additional tagged text

```
To login to UNIX system, first, you must
<n id="input1">input your ID</n>. Next
step is to <n id="input2">input your
password</n>. <n id="nomiss">Without a
mistake</n>, you will <n id="login">sign
onto a system</n>.
<slink structure="order" semantics="process"
term1="input1" term2="input2" />
<slink structure="order" semantics="process"
term1="input2" term2="login" term3="nomiss" />
```

(a) tagged text



(b) generated diagram

Figure 6: Examples of diagram generation

Diagram frame description: This description is a set of frames which hold geometric relations and attributes of diagram elements. Each frame corresponds to each primitive of a diagram, and the slots in a frame hold the attributes, for example, location, color, size or class, of primitive diagram. A frame also stores geometric relations such as contact to or separateness from other primitives. In drawing stage, the system draws diagrams based on the above information.

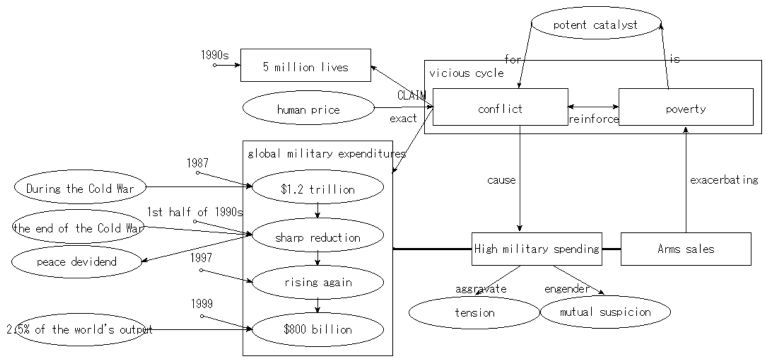
Fig.7 shows the diagram generation flow for their descriptions. We will skip the details of this process because of the lack of space.

A diagram obtained by the above process is often unsatisfactory, since our translation and placement rules are not enough for generating complicated diagrams.

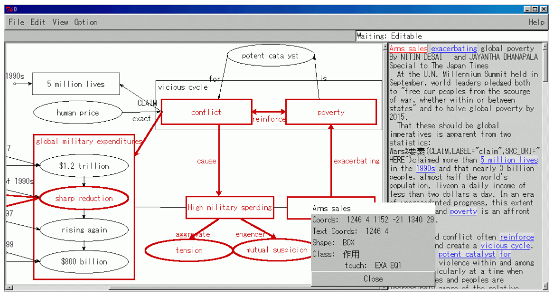
For this purpose, we prepared editing support for modifying a generated diagram. The relations kept in the above descriptions are good hints for supporting manual editing. Since the system can easily distinguish whether diagram primitives are touched each other intentionally or just by chance, the system can help the user's editing by moving or by not moving the related primitives. This keeps the user from breaking correct structures.

Actual rules currently implemented are rather simple, and some of them are listed below.

- A diagrammatic element keeps tracking other elements with which it is contact based on a relation definition.
- An enclosing element changes its shape or moves to keep enclosing other primitives that are enclosed based on a relation definition.
- Shape or color is determined by the structural and semantic



(a) Diagram generated from a newspaper article



```
Arms sales exacerbating global poverty
By NITIN DESAI   and JAYANTHA DHANAPALA
Special to The Japan Times
  At the U.N. Millennium Summit held in
September, world leaders pledged both
to "free our peoples from the scourge
of war, whether within or between
states" and to halve global poverty by
2015.
  That these should be global
imperatives is apparent from two
statistics:
Wars%要素(CLAIM,LABEL="claim",SRC_URI="
HERE")claimed more than 5 million lives
in the 1990s and that nearly 3 billion
people, almost half the world's
population, liveon a daily income of
less than two dollars a day. In an era
               nted progress, this extent
               and poverty is an affront

               d conflict often reinforce
               nd create a vicious cycle.
                potent catalyst for
                violence within and among
               icularly at a time when
               es and peoples are
```

```
Coords:  1246 4 1152 -21 1340 29.
Text Coords:  1246 4
Shape:  BOX
Class:  作用
   touch:  EXA EQ1
Close
```

(b) *Causality* concerning "arm sales" is emphasized

Figure 9: Example on a newspaper article

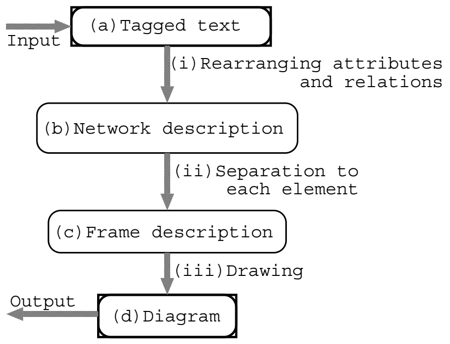
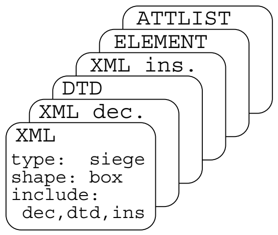
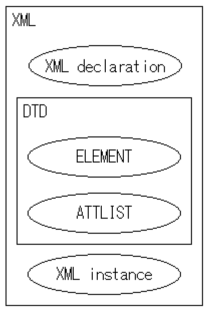


Figure 7: Flow of diagram generation



(a) Frame descriptions



(b) Generated diagram

Figure 8: Frame descriptions and generated diagram

category of an element.

6 EXPERIMENTS

First, we show a simple example that is suitable for explaining the basic idea. Fig.8 shows a diagram generated from a tagged text shown in Fig.4 and 5. Fig.8(a) the frame descriptions generated through diagram generating process, and Fig.8(b) shows the generated diagram.

Next, Fig.9 shows a more complicated example. Fig.9(a) is the figure we got after manual modification[2] of the automatically generated diagram. We can easily guess that this article is on military expenditure, and how the amount of arm sales changed. Fig.9(b) is an example for emphasizing important portions. In this case, the elements related to *causality* on "arm sales" are emphasized according the user's request.

[2]Currently, initial placement is not enough for ordinary use, the system needs our help if a diagram is complicated.

7 CONCLUSION

This paper introduced a novel scheme for generating diagrams from tagged texts. For this purpose, we investigated the correspondence between semantic structures in texts and those in diagrams. Based on the correspondence, we developed a prototype system for diagram generation.

While our system works for any texts tagged in our format, output quality heavily depends on the simplicity of a text. At the current implementation, if a tagged text is complicated, we got an almost collapsed diagram. We need further investigation for a better algorithm on initial element placement. However, it is not difficult to obtain a good diagram, since we can easily modify it with our editing support mechanism.

The most important and interesting area for future work is tag selection and natural language processing. We need a mechanism for sufficiently choosing tagged elements that should be included in a diagram. Moreover, since tags are not always perfect, we need natural language processing for compensating missing tags, eliminating unnecessary tags, or correcting tagging errors.

8 REFERENCES

[1] *The ACL workshop on Intelligent Scalable Text Summarization*, 1997.

[2] T. Bray, J. Paoli, and C. Sperberg-McQueen. Xml 1.0 specification. http://www.w3.org/TR/REC-xml/, 1998.

[3] W. Cyre, S. Balachandar, and A. Thakar. Knowledge visualization from conceptual structures. In *Proc. Second Int'l. Conf. on Conceptual Structures*, pages 275–292, Aug 1994.

[4] K.Hashida. Global document annotation. In *Natural Language Processings Pacific Rim Symposium '97*, http://www.etl.go.jp/etl/nl/GDA/, 1997.

[5] M. Okumura and H. Nanba. Automatic text summarization: A survey (in japanese). *Natural Language Processing*, 6(6), 1999.

[6] J. Sowa. Conceptual structures: Information processing in mind and machine. Addison-Wesley, 1984.