Automatic Hypertext Link Generation Model Using Virtual Path Information

Kyonghun Jeon and Keysun Ghoi
Korea Advanced Institute of Science and Technology
Center for Artificial Intelligence Research
Department of Computer Science

There have been various attempts to automate the process of generating a hypertext from texts. Node segmentation and link generation are the two most difficult processes in automatic hypertext generation systems. Node segmentation separates a big node into several small ones, each of which has the same title. The same title of those nodes, which are resulted from node segmentation or intrinsic homograph, raise a problem of link sense ambiguities in the initial automatic link generation process. This link sense ambiguities can be resolved by the users' real history path. However, the initial phase of automatic link generation process cannot use such information.

In this paper, and automatic link generation model is proposed to
resolve the link sense ambiguities. This model uses an approximation to the users' history path information, which is called the "virtual history path information", thereby making it possible to reflect the information in time of creating hypertext. In the model is defined a sequence of node sets which roughly corresponds to that of nodes in the real users' history path. The sequence of node sets forms a virtual history path. Once a virtual history path has been established, it is used to compute the score of each candidate node, and the node which gets the best score is selected as link destination. The results from our experiments tell that the virtual history path information contributes to resolve the link sense ambiguities.

1. Introduction

1.1 Automatic Hypertext Link Generation

The hypertext is a non-sequential document whose data structure is of directed graph [13, 14]. In this sense, the basic information unit of hypertext is called the "node", as for the directed graph. Each node represents a concept which is a part of one document.

Almost all hypertext authoring tools should manually indicate the start and destination points of links. Such manual tools may be appropriate to handle a small size of documents but not to a large size one due to their low productivity [1]. Manual link generation of typical hypertext authoring tools adopts one of two methods: one is to insert into the node a
series of internal link defining commands [2], and the other is to create links by direct manipulation on node sets visually in the monitor [3]. These two methods, however, cause intellectual load on human user's memory to remember a number of nodes when transforming a large amount of documents into their corresponding hypertext. This problem leads to the necessity for automatic link generation by computer [1].

The automatic hypertext generation is to automatically transform a large amount of documents into its corresponding hypertext. The major processes of automatic text-to-hypertext generation are "node segmentation" and "link generation". The main function of node segmentation is the node size fitting with the monitor size [10]. The issue in the link generation is on how the links represent the semantics of relationship among nodes.

The study on the user behaviour analysis of hypertexts [9] tells that the hypertext usability increases as the node size fits with less than one monitor size. This is the reason of node segmentation: a long explanation on one title is segmented into several small units of information to fit with one monitor size.

The link generation is the process to capture the semantic correlation between segmented nodes and to generate appropriate links. The methods of capturing correlation include simple keyword matching, morphological stem
1.2 Automatic Link Generation and Node Selection Problem

Almost all automatic hypertext link generation systems use the simple keyword matching method [1, 11]. This method generates a link between two nodes when a keyword inside a node matches with a title of the other node. Its implementation is relatively easy, but several problems exist. These problems come from the existence of synonym and homograph. Several synonymous words map into one concept, and one homograph word has multiple concepts. These problems cause the falldrop or the erroneous generation of links. It drops the usability of resulting hypertext [1].

The synonym problem leads to the study on tools of stem analysis or thesaurus [12]. However, a homograph causes a node selection problem among nodes whose titles are the same keyword. This homograph problem needs semantic solution and has hardly been explored.

To avoid the node selection problem, each keyword should correspond to one node [4, 5], or the user should select the right node when browsing [6, 7]. The former is called the "avoidance mode", and the latter the latter the "user-oriented
model!"

The avoidance model forces to avoid the multiple occurrences of the same title node. The unique correspondences between keywords and nodes are practically difficult. Even if possible, the usability of hypertext falls down and the growing size of one node incurs the user's load increase. Here, it needs the automatic node selection.

The user-oriented model needs the users' interaction history with the hypertext system, that is, users' path navigation log, node context, and so on. However, such interaction history cannot be acquired until users navigate the hypertext system. The initial automatic hypertext generation phase cannot use them.

This paper considers how to acquire the user's path navigation information in the initial link generation phase. This information is acquired not actually but virtually. Such information is called the "virtual path information" which will be used for the link sense disambiguation in the automatic link generation phase.

2. Virtual Path Information Model

The virtual path information model neither avoids the link sense disambiguation problem nor lets users select a link. However, this model does not deal with the nongoal-oriented
user behaviour and intrinsic ambiguities which cannot be resolved by using the context in a node. We will have three determinism principles on user behaviour, link and node.

2.1 Three Determinism principles

First, the virtual path information model is to simulate the user's goal-oriented navigation behaviour. It is not to model the user's goal-less exploration. This is expressed as:

**PRINCIPLE 1 (USER BEHAVIOUR DETERMINISM PRINCIPLE)**

Users navigate (or behave) deterministically a hypertext system for seeking their necessary information. The user behaviour determinism implies the goal-oriented navigation such as:

1. if the user finds the best-fit node for his seeking information, that node should be selected;
2. the already visited nodes should not be visited again.

Second, an occurrence of keyword in a node is assumed to have one sense in that node, even if the keyword is of homograph. That is, every keyword has at most one link to its destination node. This assumption is called the "link determinism
principle". The virtual path information model is to select one among multiple node candidates which correspond to multiple senses of a homograph. However, if a homograph keyword in a node implies its multiple senses in one occurrence, its corresponding nodes cannot be uniquely determined. This means that we should have multiple links from one keyword in that context. The virtual path information model does not handle this case. Thus,

**PRINCIPLE 2 (LINK DETERMINISM PRINCIPLE)**

Every anchor keyword is uniquely linked to the destination node. ■

Last, every occurrence of a keyword (anchor) has the unique sense. They should be linked to one destination node even if the keyword is of homograph. Thus,

**PRINCIPLE 3 (NODE DETERMINISM PRINCIPLE)**

Every anchor in one node has one sense. ■

### 2.2 Automatic Link Generation

The first link generation phase is to select possible anchors in a given node and then to map those anchors to nodes by one-to-one according to the LINK and NODE DETERMINISM
PRINCIPLES.

2.2.1 First Link Generation

The first phase of link generation is to find anchors in a node. The anchor function $S$ is defined to determine whether a keyword in a node can be an anchor or not, as:

DEFINITION 1(ANCHOR FUNCTION $S$)

For a given sublist $D_{sub} = [w_1, w_2, ..., w_n]$ in a node, and for a given title $T = [t_1, t_2, ..., t_n]$, the anchor function $S$ is defined as

\[
S(D_{sub}, T) = \begin{cases} 
  \text{TRUE} & \text{if } D_{sub} \text{ is an anchor for link to } T \\
  \text{FALSE} & \text{otherwise}
\end{cases}
\]

When anchor function is a string matching function, $S$ is defined as

\[
S(D_{sub}, T) = \begin{cases} 
  \text{TRUE} & \text{if } w_i = t_i, 1 \leq i \leq n, \\
  \text{FALSE} & \text{otherwise}
\end{cases}
\]

Let $A(N)$ be a set of relation between anchors and linkable
nodes which are determined by anchor function $S$ in a node $N$. This anchor set $A(N)$ is defined as:

**DEFINITION 2(ANCHOR SET $A$)**

The anchor set $A(N)$ of node $N$ is a set of ordered pairs of an anchor $a$ and a set $E$ which are candidate nodes connectable from $a$:

$$A(N) = \{(a_1, E_1), (a_2, E_2), \ldots, (a_r, E_r)\}.$$

The next step is to select only one node among $E$ according to the LINK DETERMINISM PRINCIPLE. This step needs another function called the "node-node similarity function".

**DEFINITION 3(NODE-NODE SIMILAPITY FUNCTION $M$)**

The node-node similarity function is to measure the similarity of two nodes $N_i$ and $N_j$. As the similarity is the higher, the function value is the larger. If the correlation of two nodes is nothing, $M$ has a value zero.

So, $M(N_i, N_j) \geq 0$, i, j.
The first link generation phase is to select one destination node from candidate nodes $E_i$ in the anchor set $A$ by using the node-node similarity function $M$. That is, the destination node of an anchor $a_i$ in node $N$ is

Thus, each anchor in every node is uniquely linked to its destination node in the first link generation phase. However, because the set of unique destination nodes is not complete, the link sense disambiguation is error-prone. The more accurate link sense disambiguation needs the second link generation phase.

2.2.2 Second Link Generation

The second link generation phase is to verify the accuracy of destination nodes which are the result of the first link generation phase, and to enhance the link quality. This link quality verification uses the virtual path information which is an approximation of real path considering all paths from the initial node to the current node. Figure I shows the correspondence of real path and virtual path.
The node sets on the real path can be extracted by the following extended node function.

DEFINITION 4(EXTENDED NODE FUNCTION E)

Let $X$ be a node set. For the extended node set of $X$, $E(X)$ is defined by a set of nodes which can be directly linked to a node in $X$. The nodes in $E(X)$ is disjoint from $X$, and the path length from $E(X)$ to $X$ is 1. The notation $w \xrightarrow{n} x$ means the existence of path length $n$ from node $w$ to node $x$. Thus,

$$E(X) = \{w | w \xrightarrow{1} x, w \in X, x \in X\}.$$
The extended node function $E$ finds an extended node set of the current node $N$ in which links from anchors in $N$ will be verified.

Definition 5 (General Extended Node Set $G$)

The 0-th extended node set $G_0(N)$ of the current node $N$ is \{N\}. The $n$-th extended node set $G_n$ is extended from the $(n-1)$-th extended node set $G_{n-1}$ by the extended node function $E$ as follows:

\[
G_0(N) = \{N\}
\]
\[
G_1(N) = E(G_0(N)) - G_0(N)
\]
\[
= E(\{N\}) - G_0(N)
\]
\[
G_2(N) = E(G_1(N)) - G_0(N) - G_1(N)
\]
\[
= E(E(G_0(N))) - G_0(N) - G_1(N)
\]
\[
= E(E(\{N\})) - G_0(N) - G_1(N)
\]
\[
\vdots
\]
\[
G_n(N) = E(G_{n-1}(N)) - \bigcup_{i=0}^{n-1} G_i(N)
\]
\[
= E(E(\ldots E(\{N\})\ldots)) - \bigcup_{i=0}^{n-1} G_i(N)
\]

The extended node set corresponds to nodes on the real path. Furthermore, by the User Behaviour Determinism Principle, the user does not redundantly visit the same path. Nodes in all paths are cycle-free. Because we prepare the extended node "set" on possible paths up to the current "node", the set-node similarity function is defined as follows:
Definition 6 (Set-Node Similarity Function $R$)

For a given node $N$ and a given node set $G=\{N_1, N_2, \ldots, N_n\}$, the value of set-node similarity function $R$ is in the range as:

For a given node $N$ and a given node set $G=\{N_1, N_2, \ldots, N_n\}$, the value set-node similarity function $R$ is in the range as:

$$\text{Min}(M(N_i, N)) \leq R(G, N) \leq \text{Max}(M(N_i, N)), 1 \leq i \leq n.$$  

This set-node similarity function is an extended node-node similarity function. The final virtual path similarity function $M^*$ is defined by using the set-node similarity function as follows:

Definition 7 (Virtual Path Similarity Function $M^*$)

The virtual path similarity function $M^*$ replaces a series of extended node sets for the path up to the current node $N$. $M^*$ measures the similarity between the current node $N$ and its destination node candidate $N'$ by using $G(N)$ on the virtual path up to $N$ as follows:

$$M^*(N, N') = \sum_{i=0}^{n} c_i R(G_i(N), N'), \text{ where } \sum_{i=0}^{n} c_i = 1, c_i \geq 0.$$  

Here, $\sigma$ is a constant for path level $i$. $n$ is a path length considered.

The virtual path similarity function $M^*$ selects the best-fit
node among the multiple candidates, which are the most similar node with the virtual path information. The User Behaviour Determinism Principle governs the user's consistent behaviour: "if a user navigates by selecting consistently until the current node, the user consistent behaviour will promise to select the best-fit node among multiple candidates, which is the most probable (or similar) according to the user's consistent navigation path." The goal of $M^*$ is to determine the most probable links during the hypertext generation phase and then not to deliver such disambiguation load to the users.

3. Experimentation: Path Length and Disambiguation Rate

For the experimentation, a student encyclopedia [17] is used, which contains about 22,000 entries, their explanation texts and about 13,000 figures. Each entry is classified into either large or small entry. The explanation of large entry is described by about 2,000 characters, and the small entry is about 400 characters. The number of homograph entries is 341: among them, the large entries are 86. Every large homograph entries cause the link sense disambiguation problem.

This experiment uses two domains: one contains 36 entries related to a homograph "ujuseon (우주선)" in the Korean
alphabet Hangul which corresponds to both "cosmic ray (宇宙線)" and "space ship (宇宙船)". and the other contains 22 entries related to a large entry "semiconductor". The "ujuseon" domain contains 138 links. 18 links among them can be connected to multiple destination nodes. On the other hand, the "semiconductor" domain includes 92 links. Among them, 18 links are ambiguous. The longest navigation path is 7 for both domain.

For each domain, four kinds of node-node similarity function $M$ are experimented: inner product, Dice coefficient, cosine coefficient, and Jaccard coefficient [12] after automatic indexing by using KAIS system [16]. The experiment of set-node similarity function $R$ uses the method of "maximal similarity". This "maximal" set-node similarity between a node $N$ and a node set $G$ is the largest among the node-node similarity values between $N$ and each node in $G$. (In like manner, the "minimal" similarity value adopts the smallest one, and the "average" does the just average.) The sense disambiguation results are shown in table 1. The column of $M$ shows the result after the first link generation. The column of $M^i$ is the result after the second link generation when the calculated path length is $i$.

According to the table 1, the disambiguation rate of "ujuseon" domain amounts to 56% (=10/18) without virtual path information, and 78% (=14/18) with virtual path information. On the other hand, in case of "semiconductor"
domain, the disambiguation rate amounts to 44% (=8/18) without the virtual path information, but in the best case, it amounts to 94% (=17/18). This experiment shows the great performance enhancement when using the virtual path information.

<table>
<thead>
<tr>
<th>&quot;ijuseon&quot;</th>
<th>$M$</th>
<th>$M_2^-$</th>
<th>$M_3^+$</th>
<th>$M_4^+$</th>
<th>$M_5^+$</th>
<th>$M_6^+$</th>
<th>$M_7^+$</th>
</tr>
</thead>
<tbody>
<tr>
<td>inner product</td>
<td>9/18</td>
<td>12/18</td>
<td>13/18</td>
<td>13/18</td>
<td>13/18</td>
<td>13/18</td>
<td>13/18</td>
</tr>
<tr>
<td>Dice coefficient</td>
<td>10/18</td>
<td>12/18</td>
<td>14/18</td>
<td>11/18</td>
<td>11/18</td>
<td>11/18</td>
<td>11/18</td>
</tr>
<tr>
<td>cosine coefficient</td>
<td>10/18</td>
<td>12/18</td>
<td>14/18</td>
<td>11/18</td>
<td>11/18</td>
<td>11/18</td>
<td>11/18</td>
</tr>
<tr>
<td>Jaccard coefficient</td>
<td>10/18</td>
<td>12/18</td>
<td>14/18</td>
<td>12/18</td>
<td>11/18</td>
<td>11/18</td>
<td>11/18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>&quot;semiconductor&quot;</th>
<th>$M$</th>
<th>$M_2^-$</th>
<th>$M_3^-$</th>
<th>$M_4^-$</th>
<th>$M_5^-$</th>
<th>$M_6^-$</th>
<th>$M_7^-$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dice coefficient</td>
<td>8/18</td>
<td>13/18</td>
<td>16/18</td>
<td>16/18</td>
<td>12/18</td>
<td>10/18</td>
<td>8/18</td>
</tr>
<tr>
<td>cosine coefficient</td>
<td>7/18</td>
<td>13/18</td>
<td>16/18</td>
<td>16/18</td>
<td>12/18</td>
<td>11/18</td>
<td>8/18</td>
</tr>
<tr>
<td>Jaccard coefficient</td>
<td>8/18</td>
<td>14/18</td>
<td>16/18</td>
<td>17/18</td>
<td>16/18</td>
<td>12/18</td>
<td>11/18</td>
</tr>
</tbody>
</table>

Table 1: disambiguation rate according to the path length

The figure 2 shows the variation of link sense disambiguation rate compared with the path length, in use of four kinds of *node-node similarity function*. In general, the curve increases in the first half, but it decreases gradually. The reason comes from the exponential increase of the number $|G|$ of nodes as the path length increases. That is, while the participant number of nodes increases, the number of nodes with useful information (+information) relatively decreases. As the path
length becomes the longer, the useless information (−information) grows the more and effects the more negatively to the total virtual path similarity value. Therefore, if we can find the path length such that +information and −information get balanced, the result will be the best.

(a) "ujuseon" domain

(b) "semiconductor" domain

Figure 2. Disambiguation Rate Graph compared with the Path Length.
4. Conclusion

This paper suggested an automatic link generation model based on the so-called "virtual path information". The link sense disambiguation problem occurs in the automatic text-to-hypertext generation process. In the result, the virtual path information is claimed to be useful for the link sense disambiguation problem through several experimentation.

The link sense disambiguation processes consist of two steps: first, a node-node similarity function is designed to measure the semantic similarity between nodes. This function is used to determine the first approximate sense disambiguation. The next step is to use the virtual path similarity function which represents the virtual path information. This function determines the final link sense.

The implementation of node-node similarity function uses the automatic indexing system KAIS [16] and then the indexed results of nodes are experimented by several vector-space methods: inner product, Dice coefficient, cosine coefficient, and Jaccard coefficient [12].

The experimental text is a student encyclopedia whose number of entries (or nodes) is about 22,000. The focus is
drawn on two domains: "ujuseon" homograph of the total 22 nodes, and "semiconductor" large entry of the total 36 nodes. The experimental variation of parameters amounts to 2,016. The result is that the disambiguation rate in "semiconductor" domain increases up to 94% when the virtual path is adopted, but the rate is at best 44% without using the virtual path. In case of "ujuseon" domain, the disambiguation rate increases from 56% up to 78% as the virtual path gets used.

The virtual path model is implemented by using the vector-space model. The Jaccard coefficient shows the best result among other coefficients. The set-node similarity function shows the best result when experimented by the maximal similarity function. When the path length is three or four, the result is the best.

The virtual path similarity function has a contribution to get a virtual path information before the real user uses the hypertext system. Users' real path information will be got after users' browsing. Such real path information may be feedback to enhance the performance.

References


(16) Jin-Sung Jeong. *Automatic Indexing using Linguistic and Statistical...*
가상경로정보를 이용한
자동적 하이퍼텍스트 링크생성 모델

전경훈, 최기선

대부분의 하이퍼텍스트 링크 자동생성 시스템은 단순 검색어 매칭 방법을 사용한다. 이 방법은 한 노드내의 키워드가 다른 노드의 제목과 일치하면 두 노드 사이에 링크를 생성하는 방법이다. 이 방법은 비교적 구현이 용이한 장점이 있으나, 동의어(synonym)와 동형이의어(homograph)에 의해 야기되는 여러 문제들이 따른다. 여러 동의어들은 단일 개념(concept)으로 사상되는 반면 동형이의어는 복합개념을 지니게 된다. 이러한 문제점들은 링크생성의 부족 및 오류를 야기시키며, 따라서 하이퍼텍스트의 가용성을 떨어뜨린다.

동의어 문제는 기본형 분석 도구나 동의어사전(thesaurus)에 대한 연구로 어려웠으나, 동형이의어는 여전히 키워드를 지니고 있는 타이틀의 노드 간의 노드 선택문제를 가지고 있게 된다. 동형이의어 문제는 의미론적 해결
방식이 요구되므로 거의 깊이 다루어진 적이 없다.

노드선택 문제를 해결하기 위해서는 하나의 키워드에 하나의 노드만이 대응되도록 만들든지, 아니면 사용자가 브라우징(browsing)을 할 때 올바른 노드를 선택해야 한다. 전자는 "회피 모델"(avoidance model), 후자는 "사용자중심 모델"(user-oriented model)이라 부른다.

회피 모델은 동일한 목록을 지닌 여러 다른 노드가 존재할 경우 이를 피하도록 만든다. 키워드와 노드를 대응시키는 것은 실질적으로 매우 어려우며, 가능하다고 하더라도 하이퍼텍스트의 가용성이 저하되며, 한 노드의 크기가 증가하면 사용자에게의 부담도 늘게 된다. 이러한 문제점으로 인하여, 노드가 자동선택이 요구된다. 사용자중심 모델은 사용자의 하이퍼텍스트 시스템과의 상호작용에 대한 기록, 즉 경로 항해일지(path navigation log), 노드 문맥(node context) 등을 필요로 한다. 하지만 그러한 상호작용에 대한 기록은 사용자가 하이퍼텍스트 시스템을 항해하기 전에 얻어질 수 없으므로 초기의 하이퍼텍스트 자동생성 단계에서는 사용할 수 없다.

본 논문에서는 초기의 링크생성 단계에서 사용자의 경로항해 정보(path navigation information)을 획득하는 방법을 논의하고자 한다. 이러한 정보는 가상에 근거하여 얻어지므로 "가상경로정보"(virtual path information)라 불리으며, 링크 자동생성단계에서 링크 의미결정(sense disambiguration)에 사용된다.

한국과학기술원
전산학과
대전 305-701, 대한민국