Information Visualization of Attributed Relational Data

Mao Lin Huang

Department of Computer Systems
Faculty of Information Technology
University of Technology, Sydney
PO Box 123 Broadway, NSW 2007 Australia

maolin@it.uts.edu.au

Abstract

Traditional graph drawing is only concerned with viewing of abstract data and relations amount data items. It uses a graph model to present the data items and the relations and tries to geometrically convert the abstract graph into a 2D plane for visualization. There are many applications in this area, such as *family trees, software design diagrams* and *web site-maps*.

The real world data that we want to visualize, however, is more complex than those that can be simply presented with traditional techniques, because it contains many domain-specific attributes.

For example, in a web site-map (visualization) a simple graphical node can be used to represent a web page. However, this node is unable to represent some domain-specific attributes associated with the web page, such as the access frequency and the connectivity of the page in the web locality.

This paper introduces a new model for Attributed Information Visualization that can be used to visualize the relational data with many associated attributes. We split the visualization model into two mappings between the abstract data and physical pictures, Geometric Mapping and Graphical Mapping.

Keywords: information visualization, attributed relational data, web browsing, web graph and graph drawing.

1 Introduction

Traditional techniques of relational information visualization (Battista, Eades, Tamassia, and Tollis 1999, Eades 1984) concern only the viewing of the abstract data and relations amount data items. They use *graphs* to model relational structures: the entities are *nodes*, and the relationships are *edges* (sometimes called *links*). For example, the structure of the World Wide Web can be modeled as a graph: the nodes are HTML documents, and a *hyperlink* (with different communication protocols, such as *http, ftp, mailto* and *telnet*) from one document to another is represented as a directed edge.

These graphs are typically drawn as diagrams with text at the nodes and line segments joining the nodes as edges. These drawings provide a readable and comprehensive

graphic format of the relational structure for easy understanding.

The goal of the traditional *graph drawing* is to convert the abstract relational structure into a 2D/3D geometric space for visualization. This geometrical mapping involves the appending of two geometric attributes (x_a and y_a coordinates) to each *node a*, and four geometric attributes ((x_a, y_a), (x_b, y_b)) to each *edge (a, b)* in a graph. There are many applications in which the traditional graph drawing methods can be used to convey information, such as *family trees, software design diagrams* and *web site-maps*.

However, the data we want to visualize in the real world is much more complex than those can be simply presented in a 2D/3D geometric plane. For example, in the drawing of a web graph, an *edge* can only be used to present a hyperlink with four geometrical attributes ((x_a, y_a), (x_b, y_b)), and it cannot present the communication protocol (such as *http, ftp, mailto* or *telnet*) associated with the hyperlink. The pure geometric presentation of a web graph does not show any further detail about a particular HTML document, such as the *access frequency* and *the connectivity* of the document.

Thus the traditional information visualization techniques and graph drawing methods tend to be inadequate to represent the relational data which has more attributes associated with it. We call this type of the data *Attributed Relational Data*.

Some alternative graph drawing methods have been proposed (Eades, Lai, and Mendonca 1994, Lai 1993, Kamada 1989, Kamada and Kawai 1988). They used a weighted graph model to address the problem of visualizing attributed relational data, such as the data describing the *E-mail Traffic* on the Internet (Eades, Lai, and Mendonca 1994). However, these methods still restrict the solutions of visualization within the domain of graph drawing, which concerns only *the geometric mapping* between the data and target pictorial representation. They do not take the advantages of rich graphics available on most of the PC/workstations to achieve the second mapping: *the graphical mapping* in which a set of rich graphical attributes is used to represent the domain-specific attributes of the data. Other systems, such as described by Becker, Eick and Wilks (1995), concentrate on specific applications.

This paper presents techniques to visualize attributed relational data. We introduce our new visualization model that converts the attributed relational data into a target

pictorial representation through two mappings: *the geometric mapping* and *the graphical mapping*. Our aim is to provide techniques that help human to distinguish the variety of attributes associated with the relational data through the use of graphical attributes.

This technique is implemented as a system for visualizing web graphs. It uses a force-directed graph drawing method (Eades 1984) to achieve the *geometric mapping*, and uses variety of graphical attributes to represent the metrics (attributes) associated with each *HTML document* and each *hyperlink*.

2   The Visualization Model

A graph G consists of a finite set N of nodes and a finite set E of edges, where each edge is an unordered pair of nodes. A node μ is said to be *adjacent* to a node v if (μ, v) is an edge of G; in this case, the edge (μ, v) is said to be incident with μ and v.

A *drawing* of a graph $G = (N, E,)$ consists of two functions, D_n: $N \to R^2$ that associates a location $D_n(v)$ to each node $v \in N$, and a function D_e: $E \to C$ that assigns a curve $D_e(u,v)$ to each edge (u,v) in E such that the endpoints of $D_e(u,v)$ are $D_n(u)$ and $D_n(v)$.

In practice, the nodes and edges have domain-specific attributes. For example, a hyperlink in a web graph has a protocol attribute (*http*, *ftp*, *mailto*, *etc*), and a node in a network may have a numerical attribute representing the amount of traffic passing through the node.

An attributed graph $A(G) = (A(N), A(E))$ consists of a finite set $A(N)$ of attributed nodes and a finite set $A(E)$ of attributed edges. Each attributed node $a(u) \in A(N)$ consists of $(u, DA(u))$ where $DA(u)$ is a set of domain-specific attributes associated with node u in the graph. Respectively for each attributed edge $a(u,v) \in A(E)$ we have that $a(u,v) = ((u,v), DA(u,v))$ where $DA(u,v)$ is a set of domain-specific attributes associated with edge (u,v) in the graph.

In practice, the underlying graphics system has a set of available *glyphs* and a set of *linetypes*. Using this terminology, a drawing of a graph G maps each node u to a glyph g at a location $D_n(u)$ on the screen and each edge (u,v) to a linetype l with endpoints attached to the glyphs at $D_n(u)$ and $D_n(v)$.

Each glyph g (respectively each linetype l) has a set A_g (respectively A_l) of *attributes*. The attributes of a glyph include *shape, size*, and *colour*. The attributes of a linetype include *shape* (*Bezier, B-spline, etc.*), *thickness, linestyle* (*solid, dashed, dotted etc.*) and *colour*.

The attributes $\{a_g^1, a_g^2, a_g^3, \dots a_g^x\}$ of a glyph g is a set of specific values of different types (*boolean, Integer, character, etc*) that associated with g. For example, suppose that a_g^i is a *shape* attribute associated with glyph g, we then can have a set of possible values, *"Rectangle", "Oval", "Curve", "Polygon"*, etc of the *character* type assign to this attribute. If a_g^i is a *size* attribute, then we may assign a non-negative integer value (from 0 to 1024) to it as the actual number of pixels.

An *Attributed Visualization* (the underlying graphics) $AV(G) = (GLYPH(N), LINETYPE(E))$ of a graph G consists of a finite set $GLYPH(N)$ of *glyphs* (graphical entities) and a finite set $LINETYPE(E)$ of *linetypes* (graphical entities). Each glyph $g(u) \in GLYPH(N)$ consists of $(A_g, D_n(u))$ where A_g is a set of graphical attributes associated with g in the visualization, and the drawing $D_n(u)$ is a geometric location of u in a 2D plane with x, y coordinates associated with u. Respectively for each linetype $l(u,v) \in LINETYPE(E)$ we have $l(u,v) = (A_l, D_e(u,v))$, where A_l is a set of graphical attributes associated with $l(u,v)$ and the drawing $D_e(u,v)$ is a geometric curve representing the relation (u,v) in a 2D plane, which includes two endpoints for the curve (u,v). Under this schema, the visualization problem becomes the problem of mapping between attributed graphs $A(G)$'s and attributed visualizations $AV(G)$'s.

This model follows a well trodden scheme of relational visualization (see Mackinlay 1986).

3   Translation of Data into Pictures

In this section, we describe a framework for visualizing attributed relational data. In this framework, the translation of data into pictures includes two steps: *the geometric mapping* and *the graphical mapping*. We present the conceptual model of the data translation, and implement this model by using Java.

3.1   The Transition Diagram of Attributed Visualization

Figure 3.1 illustrates a transition diagram for attributed visualizing. This diagram is made up of several components. These components are described below:
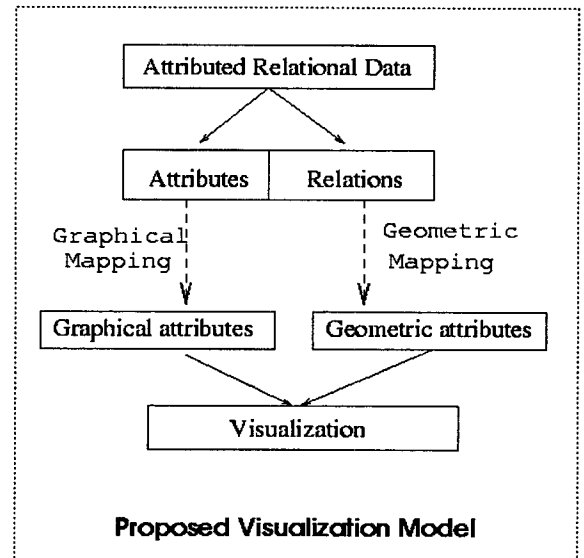

Figure 3.1: The Framework of Attributed Visualization.

- Attributed Relation Data

The real world relational data that we want to visualize. It includes a set of relationships among the data objects, and a set of domain-specific attributes associated with each data object and relationship (relation object).

- Attributes

  A set of domain-specific attributes (properties) that are associated with a particular data object or relation object in a specific domain of the real world.

- Relations

  Relationships among the data objects. As a certain type of the objects, a relation can associate a set of domain-specific attributes with it. These attributes determine the type of the relationship between two objects.

- Graphical Attributes

  A set of graphical properties that are associated with a particular graphical object *glyph* or *linetype*, which is the graphical shadow of a particular data object or relation object.

- Geometric Attributes

  A set of geometric properties that are associated with a data object /or a relation object. For example, in a 2D plane, a data object usually associates two geometric properties, x and y coordinates in a plane.

- Visualization

  The final pictorial representation of attributed relational data. This includes representations for data, relational structure and domain-specific attributes that are associated with the data/relation objects. The basic elements of the visualization are *glyphs* and *linetypes*. Each glyph g (respectively each linetype l) represents a data object $v \in N$ (respectively a relation object $e \in E$). The attributes $A_g = \{a_g^1, a_g^2, a_g^3, \ldots a_g^x\}$ of a glyph g is used to represent the domain-specific attributes of a data object $v \in N$ that is graphically mirrored on g. Respectively the attributes $A_l = \{a_l^1, a_l^2, a_l^3, \ldots a_l^y\}$ of a linetype l is used to represent the domain-specific attributes of a relation object $e \in E$ where l is a graphical shadow of e in the visualization.

During the transition, the original data could have three different representations in each stage of the translation process.

1. The abstract graph representation

  This level of the representation describes the abstract relational structure of the data objects and relations among the objects in the real world. These data objects are represented as a set of *nodes*, and these relations are represented as a set of *edges* in a graph model. Different applications of the relational date in the real world are translated into this universal representation.

2. The geometric representation

The second level of the representation is generated after the geometric mapping. The outcome of the geometric mapping is a geometric structure. A geometric structure means that each data object (node) is assigned with a geometric position and each relation object (edge) in the graph is assigned with two geometric positions, a *start-point* position and a *end-point* position. To make this geometric structure visible and allow users to view this geometric structure in a 2D plane, some simple graphical objects, such as *rectangles* and *lines*, are used to graphically display these geometric objects. This level of the representation is sufficient for the visualization, as the target representation, of *non-attributed relational data*.

3. The geometric + graphical representation

  This level of the representation is the target representation - the actual pictorial representation for the visualization of *attributed relational data*. It is generated after the graphical mapping. The outcome of the graphical mapping is a picture. A picture consists of many graphical objects, and each graphical object can have many graphical properties (including geometric and graphical properties). These properties, such as *position, shape, size, color, brightness* and *z-coordinate*, can be used to represent the main features of the data/relation objects as well as the attributes associated with the data and relations in the real world.

3.2 The Geometric Mapping

The geometric mapping is the process of creating geometric representation of the data. It converts the abstract relational structure of data into a network structure with geometric (x, y) *points* and (*point to point*) *links* on a 2D plane. As the first stage of visualization, it assigns a geometric position $(x, y$ coordinates) to each graphical glyph $g(v)$ that is the shadow of a data object v in a visualization. It also defines the *start-point* and the *end-point* for each graphical linetype $l(e)$ that is the shadow of a relation e in the visualization. Suppose that we have a finite set N of data objects and a finite set E of relation objects; then the actual geometric mapping from data set N to the drawing $D_n(N)$ can be illustrated below:

$$N = \{ n_1, \; n_2, \; n_3, \ldots n_x \}$$

$$D_n = \{D_n(n_1), \; D_n(n_2), \; D_n(n_3), \ldots D_n(n_x)\}$$
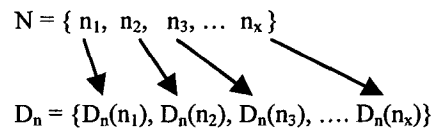
The actual geometric mapping from relation set E to the drawing $D_e(E)$ can be illustrated below:

$$E = \{ e_1, \; e_2, \; e_3 \ldots e_y \}$$

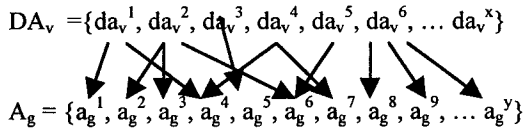$$D_e = \{D_e(e_1), \; D_e(e_2), \; D_e(e_3), \ldots D_e(e_x)\}$$
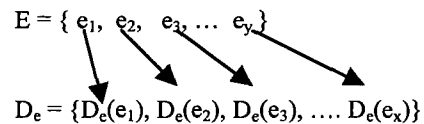
These mappings are one-to-one which means that there is only one geometric shadow created for each data object v and each relation object e after the mapping process. It assigns a location $D_n(v)$ to each data object v and assigns a curve $D_e(e)$ to each relation object e.

3.3 The Graphical Mapping

The graphical mapping is the process of creating pictorial representation of the data. It converts a set of domain-specific attributes DA_v of a data object v (respectively a relation object e) into a set of graphical attributes A_g (respectively A_l) of a graphical object g (respectively l) for visualization. Note that g (or l) is a graphical shadow (representation) of data v (or relation e).

Suppose that v is a data object associated with a set DA_v of domain-specific attributes. There is a graphical object g that can have a set A_g of graphical attributes associated with it. The actual graphical mapping from v to its graphical shadow g can be done as illustrated below:

$$DA_v = \{da_v^1, da_v^2, da_v^3, da_v^4, da_v^5, da_v^6, \dots da_v^x\}$$

$$A_g = \{a_g^1, a_g^2, a_g^3, a_g^4, a_g^5, a_g^6, a_g^7, a_g^8, a_g^9, \dots a_g^y\}$$

Respectively the graphical mapping from a relation object e to its graphical shadow l can be done in the same way.

These mappings are many-to-many mapping which means that there are more than one graphical attributes can be used to represent a domain-specific attribute da_v^i (respectively da_e^i) that is associated with a data object v (respectively a relation object e). This also means that a graphical attribute a_g^i can be used to represent more than one domain-specific attribute in the visualization.

4 Implementation on Web Graphs

The incredible size of the web, accompanied with its inherent lack of structure both at the *inter-* and *intra-*document levels, introduces great challenges for information discovery. The web has no navigation structure or any sort of complete index of the content available. The problem of global navigation across the web space might never be perfectly solved.

One of the ways in which web site designers are trying to address this problem is by providing what is commonly called "site-maps". The idea of a web site-map is based on the geographical metaphor of map. It is used to provide the user with an overall visual picture of the contents of a web site so that the user can easily navigate and obtain the interested information.

Since web site mapping is essentially a process of visualization of the information content of a web site, various approaches are adopted based on human visualization and perception capabilities. Each approach or technique for web site mapping has adopted one or a combination of these capabilities hoping to exploit them to help the user in navigation and comprehending the contents.

In web site-maps, a HTML document can be presented as a *node* in the graph, and a hyperlink can be presented as an *edge* in the graph. However, most of these approaches are only focusing on the pure geometric representations, rather than the graphical representations, of web graphs and they usually just assign some very simple graphical objects *glyphs* (perhaps some simple rectangles of the same size) to the *nodes* with the same graphical

properties (such as *size, color* and *shape*) for visualization. These simple graphics are unable to represent the domain-specific attributes associated with the HTML document, such as *the access frequency* and *the connectivity* of a HTML document.

Therefore the user gains no knowledge about the domain-specific attributes of nodes (web pages) and edges (hyperlinks), which are very important to the user where she/he is surfing through the visual structure of a web graph.

In this section, we apply our new visualization technique to create web site-maps. We want to use a set of graphical attributes A_g (respectively A_l) associated with a glyph g (respectively l) to represent a set of domain-specific attributes DA_v (respectively DA_e) of a HTML document v (or a hyperlink e), where g (or l) is the graphical shadow (representation) of a HTML document v (or a hyperlink e).

4.1 Domain-Specific Attributes in Web Graphs

Suppose that a HTML document (node) v in a web site has three domain-specific attributes $DA_v = \{da_v^1, da_v^2, da_v^3\}$, where $da_v^1 \rightarrow Connectivity$, $da_v^2 \rightarrow Access frequency$ and $da_v^3 \rightarrow Depth$ and we usually use these attributes to measure the importance of a node v (a web page) in the web locality. We now list these attributes as follows

<u>Node (web page) attributes</u>

- <u>Connectivity</u> (outdegree & indegree)

  This attribute indicates how a node is connected to other nodes in the web space. If a node is connected to more other nodes in the web space, then we say this node is to be more important in the web locality. To find out the connectivity of a node the hyperlink structure (the topology) among the web pages needs to be extracted.

  We use two terms, outdegree and indegree to measure the connectivity of a node. The outdegree of a node is the number of other nodes that can be directly reached from this node with the graph theoretic distance of 1. Similarly, the indegree of a node is the number of other nodes that can directly reach this node with the graph theoretic distance of 1.

- <u>Access frequency</u>

  This attribute indicates how many times the node has been accessed in the recent time period. The importance of a node could also be reflected by the access frequency of the node. This means that if a node (page) had more visitors in a periodical time, then we say this node is to be more important in the web locality. This information can be obtained by analysis of the log files. For each node we may calculate the number of times it was accessed in the preceding month.

- <u>Depth</u>

  This attribute indicates at what depth the node resides in the file system hierarchy of the web locality. The importance of a node could also be

reflected by the position of the node in the hierarchical file system of a web site. Generally, when a web site is developed, the file system hierarchy is formed so that we assume that the important nodes are normally higher up in the hierarchy. For example, in the *Faculty of Information Technology* site at UTS, the top directory of the web site has the home page of the Faculty.

These pages in the top directory of a site normally give more general information and therefore more important for understanding a web locality are higher up in the hierarchy than the pages containing detailed specific information.

Suppose that a hyperlink (edge) e in a web site has a domain-specific attribute $DA_e = \{da_e^1\}$ where $da_e^1 = Protocol$ and we use this attribute to indicates the type of protocol used for data transmission through the link. We define this attribute below

Edge (hyperlink) attribute

• Protocol

This attribute indicates the type of communication protocol shosen for data transmission. The possible values of this attribute are *HTTP, FTP, TELNET* and *MAILT.*

4.2 Graphical Attributes Associated with glyphs & linetypes

In our implementation, each glyph g in a web site-map has a set $A_g = \{a_g^1, a_g^2, a_g^3, a_g^4, a_g^5, a_g^6\}$ of six graphical attributes. They are:

$a_g^1 \rightarrow$ size of the graphic entity

$a_g^2 \rightarrow$ background color

$a_g^3 \rightarrow$ shape of the nodes

$a_g^4 \rightarrow$ brightness

$a_g^5 \rightarrow$ highlight/shadows

$a_g^6 \rightarrow$ Z-coordinate at overlaps

In a web site-map, each linetype l has a set $A_l = \{a_l^1, a_l^2, a_l^3, a_l^4\}$ of four graphical attributes. They are:

$a_l^1 \rightarrow$ length

$a_l^2 \rightarrow$ thickness

$a_l^3 \rightarrow$ brightness

$a_l^4 \rightarrow$ Z-coordinate at crossings

4.3 Creating Pictorial Representation of Web Graphs

The actual pictorial representation of a web site-map is generated after the *graphical mapping*, in which a set of domain-specific attributes associated with each web object v (or e) in the graph is mapped to a set of graphical attributes associated with graphical objects *glyphs* (or *linetypes*) in the visualization. This graphical mapping is a many-to-many mapping as illustrated in Figure 4.1.

4.4 An Example of Visualizing Web Graphs

Figure 4.2 shows the web graph of a retrieved page, *http://www.it.uts.edu.au/*. This is the home page of the *Faculty of Information Technology, University of Technology, Sydney*. This graph is drawn using a *spring* algorithm [2] (a kind of force-directed drawing), without applying Attributed Visualization technique. In the visualization, we can only see the relationships (hyperlinks) among the nodes (pages), and cannot see any domain-specific attributes of web objects in this visual site-map. Therefore, the user gains no knowledge about the properties (attributes) associated with these web objects (pages & links) in a web site-map while she/he is browsing through the map.

Figure 4.3 shows the same layout of a web graph as shown in Figure 4.2. However, it uses graphical attributes to represent the domain-specific attributes associated with web objects. This gives the user some ideas of what nodes are more important in the web localities and worthwhile to have a look.

For example, the node "*www.uts.edu.au/*" is represented by a graphical *glyph* of large size in the visualization. This is because it's the default home page of UTS and it has a high access frequency. The node "*/courses*" is also represented by a large *glyph* because this page contains the course information with a high access rate. It is accessed heavily by large number of students.

5 Conclusion

Attributed Information Visualization can be used to visualize not only the relational structure of the data, but also the domain-specific attributes that are associated with the data. The visual representations of these attributes are very important for users, who are browsing through the visual structure, in understanding of the data objects and the relationship objects that are represented as a set of *glyphs* and *linetypes* in a picture. Pictures produced by Attributed Information Visualization are much meaningful than those produced by traditional Information Visualization.

The use of Attributed Visualization for web site mapping can improve the quality of visual web browsing. Under this scheme, the user can gain the knowledge about each node (page) in the web locality from the visual map. This will directly help the user in making up her/his decision of where to go next while he/she is interactively browsing through the web via the visualization.

6 References

BATTISTA, G., EADES, P., TAMASSIA R. & TOLLIS, I. (1999): *Drawing Graphs: Algorithms for the Visualization of Graphs*. USA, Prentice Hall.

EADES, P. (1984): A Heuristic for Graph Drawing. *Congressus Numerantium* 42:149–160.

P. EADES, P., LAI, W. and MENDONCA, X. (1994): A Visualizer for E-mail Traffic. *In 4th Int. Conf. Proc. Pacific Graphics'94 / CADDM'94*, pages 64-67.

LAI, W. (1993): Building Interactive Diagram Application. Ph.D. thesis. University of Newcastle, Australia.

KAMADA, T. (1989): Visualizing Abstract Objects and Relations. *World Scientific Series in Computer Science* - vol 5.

KAMADA, T. and KAWAI, S. (1988): Automatic Display of Network Structures for Human Understanding. Technical Report 88:007. Department of Information Science, Faculty of Science, University of Tokyo.

MACKINLAY, J. (1986): Automating the Design of Graphical Presentations of Relational Data. *ACM Transactions on Graphics* 5(2): 110 - 141.

BECKER, R., EICK, S. and WILKS, A. (1995): Visualizing Network Data. *IEEE Transactions on Visualization and Graphic* 1(1): 16 - 28.
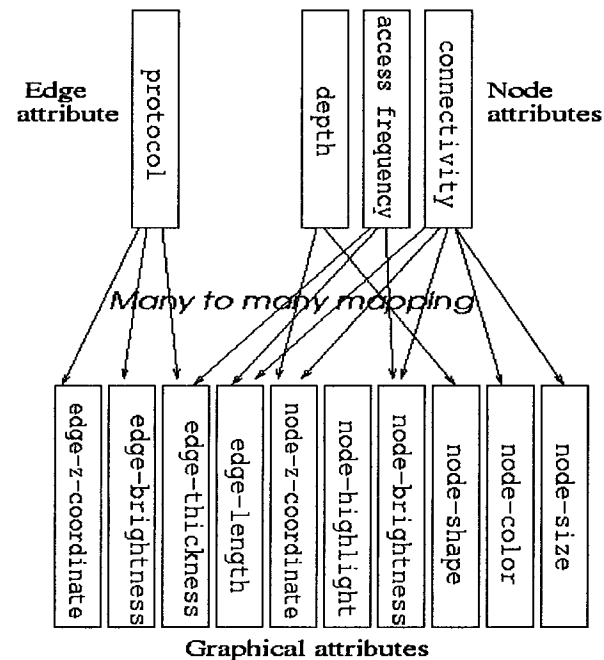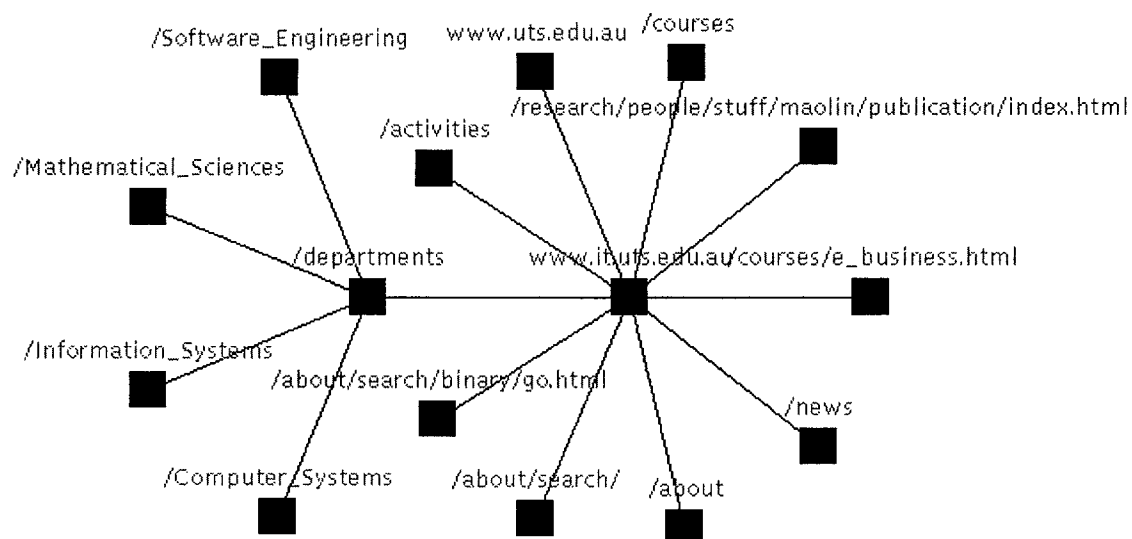
Figure 4.1: The Graphical Mapping.

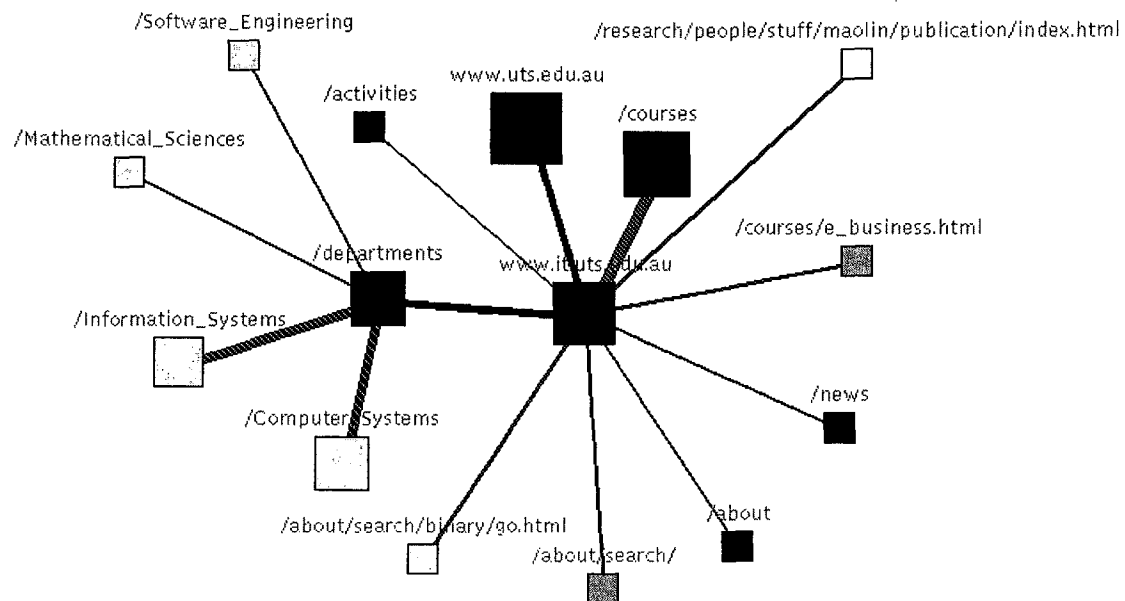**Figure 4.2: A visualization of a web graph without applying graphical mapping.**

Figure 4.3: The same drawing of a web graph as illustrated in Figure 4.2 with the application of Attributed Visualization. The user can easily see the different attributes associated with nodes (pages) and edges (links) in the site-map.