

XBRL 用户自定义元素的可视化比较实现

摘要

随着金融信息技术的发展,所需要处理与分析的数据比以往要多得多。海量数据集合不仅是对聚类技术提出了新的要求,同时也对信息可视化技术的挑战。为迎合处理海量数据集合需求,可视化专家提出了“焦点+背景”技术。这种技术提供了在放大某一子集合的数据时,同时在视图中保留数据整体结构。基于“焦点+背景”技术的双曲线树是实现大型层次结构信息可视化的有效解决方案。本文使用双曲线树浏览器展示具有大量层次的 XBRL 元素,对比用户自定义与行业标准元素。文章综述了 XBRL 的应用现状与信息可视化技术的发展情况,重点介绍了双曲线树的概念,并对三个常用双曲面模型——Klein 模型、上半平面模型和 Poincare 模型进行比较。本文所实现的双曲线树浏览器是一个可以对排版在双曲空间中的树进行可视化和导航的程序。它包括了“焦点+背景”可视化视图、自定义元素的对比显示、导航等功能,实现了在有限的屏幕空间内显示具有大量层次的信息,便于财务专业人员对于 XBRL 用户自定义元素的检索。

关键词：信息可视化, 焦点+背景, 双曲线树, XBRL

VISUALIZING XBRL ELEMENTS IN HYPERBOLIC SPACE

ABSTRACT

The speeding evolvement of financial information technology is demanding for higher level of cluster and visualization as well. Thus, Focus + Context technique is put forward by experts. Visualizing large hierarchical data with hyperbolic tree which is based on the Focus + Context technique is an appropriate solution for visualization of enormous trees. We apply a hyperbolic tree viewer to display large hierarchical data of XBRL business report and distinguish the extended elements. A brief history of the development of the implementation of XBRL is presented in the first part of this article. Several current techniques of visualizing large hierarchical data are introduced next. Then, we compared three common hyperbolic mapping models – Klein model, the upper-half plain model and the Poincare disc model. Our hyperbolic tree viewer is an application to visualize and navigate large trees in hyperbolic space mapping with Poincare disc model. It includes Focus + Context viewing, elements display and navigation, allowing large trees to fit within a limited screen space.

Key words: large hierarchical data visualization, Focus + Context, hyperbolic tree, XBRL

目 录

第一章 引言	1
第二章 XBRL 介绍	2
2.1 什么是 XBRL	2
2.2 XBRL 的层次结构	3
2.2.1 XBRL 中的技术规格	3
2.2.2 XBRL 中的分类标准	3
2.2.3 XBRL 中的实例文档	4
2.3 XBRL 中的自定义元素	5
2.4 本章小结	5
第三章 信息树可视化技术研究现状	7
3.1 信息可视化与信息树	7
3.2 层次信息可视化技术的发展现状	7
3.2.1 节点连接法	7
3.2.2 区域填充法	8
3.3 焦点-背景信息树技术	9
3.3.1 半径树	10
3.3.2 双曲线树	10
3.4 传统树形结构与信息树的比较	11
3.5 本章小结	12
第四章 双曲线层次信息可视化技术介绍	13
4.1 双曲几何	13
4.1.1 非欧氏空间	13
4.1.2 双曲几何变换	14
4.2 双曲面模型	15
4.2.1 Klein 模型	15
4.2.2 上半平面模型	16
4.2.3 Poincare 模型	16
4.3 本章小结	17
第五章 XBRL 双曲线树浏览器的实现	19
5.1 双曲线树可视化方法介绍	19
5.1.1 排版方法	19
5.1.2 映射算法——Poincare 圆盘模型	19
5.1.3 坐标变换算法	20
5.1.4 节点信息展示	21
5.2 双曲线树浏览器结构	21
5.3 输入数据的 XML 结构	22
5.4 标准与自定义 XBRL 元素的显示	23
5.5 本章小结	23
第六章 结论	24

6.1 总结	24
6.2 展望	24
参考文献	25
谢辞	28

第一章 引言

XBRL (eXtensible Business Reporting Language, 可扩展商业报告语言) 是用于企业财务数据电子交流的语言, 是一种基于 XML 语言的实现 (准确的说是规范化)。上世纪 90 年代末美国首先提出 XBRL 理论, 很快为实务界所采纳, 短短几年之内, XBRL 得到超乎寻常的发展。在 XBRL 国际组织 (xbrl.org) 的推动下, 目前 XBRL 在全球范围内已经被大多数会计师事务所、贸易机构、软件开发商、金融机构、投资者以及政府机构采用。

XBRL 标准被广泛的使用, 各国根据各自的财务准则都制定了各自的数据项分类表, 用户还可以根据自身的情况, 自定义增加和删减分类体系, 做到个性化。在我国证券交易所早在 2005 就开始使用 XBRL 标准提交财务报表。但在使用中也发现由于个性化的存在使得自定义的数据有着爆炸增长的趋势, 尤其是美国近两年提交的报表数据中有近一半的数据是自定义的。这种现象很值得研究, 现有问题的研究可以首先集中在标准的修订以及自定义数据的归类上。

XBRL 有众多节点, 且呈现为树形结构, 因此寻找合适的可视化方案是非常重要的。信息树最适合表示具有层次关系的文本信息, 使文本信息更直观明了, 便于用户的理解和利用。具体来说, 它是一种层次式或类似树状的资料结构, 将相关领域的电子资源依据它们的内容分成若干类别, 每一类别下又按照文件内容分成更细的类别, 一直到类别下的文件数量达到限定的数目为止。用户在使用信息树进行浏览时, 先确定自己所要查找的内容应归属的类别, 然后逐层选择适当的分支, 进而使范围逐级缩小, 最后确认信息需求, 在具体实施的过程中, 信息树不仅可以过滤掉不相关的信息, 同时也为用户呈现出某一领域的知识结构体系。树型结构是用来描述层次信息的较为常见的方式。但对于大型的层次结构, 这种传统的树型结构的分支很快就会拥挤交织在一起, 变得混乱不堪。

本课题将实现 XBRL 用户自定义元素的可视化比较, 便于专业人员对于自定义报表数据的趋势研究以及标准修订。

第二章 XBRL 介绍

自从 1998 年 XBRL 诞生时起, XBRL 就获得了迅速的发展, 尤其作为财务信息处理的最新标准和技术, XBRL 增加了公司财务报告披露的透明度, 同时极大的提高了财务报告信息处理的效率和能力。以金融行业为例, 目前国际上许多交易所、会计师事务所和金融服务与信息供应商等机构已采用或准备采用该项标准和技术, 如东京交易所的 TD net 系统采用了 XBRL 技术报送财务数据, 澳洲交易所正在研究并准备使用 XBRL, 德国德意志银行将 XBRL 用于处理贷款信息并使其信用分析过程更加流畅^[1]。在中国 XBRL 也开始得到应用。上海证券交易所和深圳证券交易所都开始了 XBRL 技术的研究和试验。目前已有几十家上市公司参加了使用 XBRL 技术提交公司年报摘要的试点工作。中国已于 2008 年 11 月 12 日成立了 XBRL 中国地区组织。当然以国内目前的现状来说, XBRL 的应用还处于起步阶段, 还有很多的问题。

2.1 什么是 XBRL

XBRL (eXtensible Business Reporting Language, 可扩展商业报告语言) 是用于企业财务数据电子交流的语言, 是一种基于 XML 语言的实现。上世纪 90 年代末美国首先提出 XBRL 理论, 很快为实务界所采纳, 短短几年之内, XBRL 得到超乎寻常的发展。在 XBRL 国际组织 (xbrl.org) 的推动下, 目前 XBRL 在全球范围内已经被大多数会计师事务所、贸易机构、软件开发商、金融机构、投资者以及政府机构采用。

XBRL 标准在美国产生后, 很快在发达国家市场传播应用。它的先进性在于为公司编制各种格式的商业报告提供了新的标准化方法, 使分析和交换公司商业数据及表外说明变得更为简单、可靠。就公司财务信息处理而言, XBRL 将最新信息处理技术应用于财务信息领域, 使公众能够更方便地检索、分析和使用公司披露的财务报告; 提高财务报告数据交换、搜索、共享等方面的效率。

现在对 XBRL 的研究普遍存在仅从技术上理解的倾向, 但其实际情况并非如此。XBRL 的广泛传播并不是仅仅是由于技术规范的成熟和标准的整齐划一, 更重要的是因为它对于现实生活中商业行为的深刻理解和语义层面的把握, 才使得不同语法形式下 (外在表现为异质平台) 信息共享成为可能, 极大地促进了财务信息数据挖掘的进行, 满足了决策者对有用信息和知识的需求。^[2]

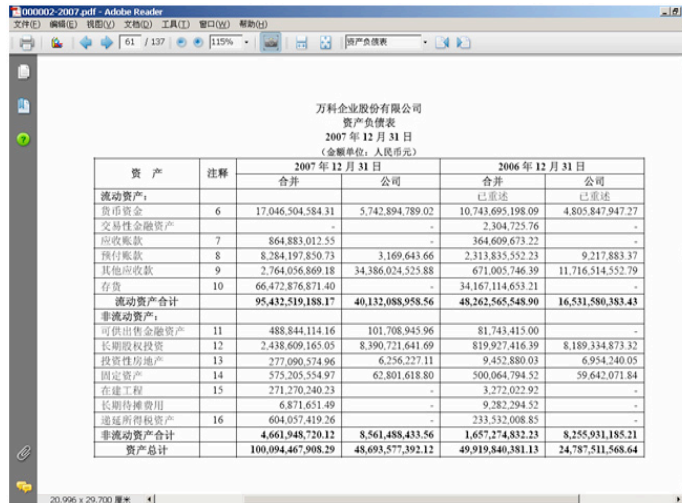
XBRL 的特点在于它根据财务信息披露规则, 将财务报告内容分解成不同的数据元 (data elements), 再根据信息技术规则对数据元 (data elements) 赋予唯一的数据标记, 从而形成标准化规范。以这种语言为基础, 通过对网络财务报告信息的标准化处理, 可以编制出比现行网络财务报告更加先进的报告, 可以将网络财务报告的不能自动读取的信息转换为一种可以自动读取的信息, 大大的方便信息使用者对信息批量需要和批量利用。同传统的网络财务报告相比, 以 XBRL 为基础的网络财务报告具有以下的一些特点:

- (1) 降低信息交换成本、提高财务信息的可获得性、间接增加了财务信息可比性;
- (2) 通过互联网提供具时效性的信息, 提高信息的相关性, 增强了财务信息的利用效率;
- (3) 可自动交换并摘录财务信息而不受个别公司软件和信息系统的限制, 为投资者或分析者使用财务信息提供方便;
- (4) 可以减少为了不同格式需求的资料而重复输入的问题;

(5) 降低了信息供给成本, 有利于信息供给者提高财务报表编制效率。^[3]

XBRL 格式与现有信息披露格式区别如下:

现有信息披露主要格式为: WORD、PDF, 便于投资者阅读, 但不利于检索、分析、统计, 其样式如图 1:



资产	注释	2007年12月31日		2006年12月31日	
		合并	公司	合并	公司
流动资产:				已重述	已重述
货币资金	6	17,046,504,584.31	5,742,894,789.02	10,743,695,198.09	4,805,847,947.27
交易性金融资产		-	-	2,304,725.76	-
应收账款	7	864,883,012.55	-	364,609,673.22	-
预付账款	8	8,284,197,850.73	3,169,643.66	2,313,835,552.23	9,217,883.37
其他应收款	9	2,764,056,869.18	34,386,024,525.88	671,005,746.39	11,716,514,552.79
存货	10	66,472,876,871.40	-	34,167,114,653.21	-
流动资产合计		95,432,519,188.17	40,132,088,958.56	48,262,565,548.90	16,531,580,383.43
非流动资产:					
可供出售金融资产	11	488,844,114.16	101,708,945.96	81,743,415.00	-
长期股权投资	12	2,438,609,165.05	8,390,721,641.69	819,927,416.39	8,189,334,873.32
投资性房地产	13	277,090,574.96	6,256,227.11	9,452,880.03	6,954,240.05
固定资产	14	575,205,554.97	62,801,618.80	500,064,794.52	59,642,071.84
在建工程	15	271,270,240.23	-	3,272,022.92	-
长期待摊费用		6,871,651.49	-	9,282,294.52	-
递延所得税资产	16	604,057,419.26	-	233,532,068.83	-
非流动资产合计		4,661,948,720.12	8,561,488,433.56	1,687,174,832.23	8,255,931,185.31
资产总计		100,094,467,908.29	48,693,577,392.12	49,919,840,381.13	24,787,511,568.64

图 1 现有上市公司信息披露样式

而 XBRL 格式信息披露则有便于计算机自动处理, 方便投资者检索、分析、统计、共享等特点, 其样式如下图 2

```
<cn-f:货币资金 contextRef="ins_20071231_c" unitRef="CNY" decimals="2">17046504584.31</cn-f:货币资金>
<cn-f:货币资金 contextRef="ins_20071231_p" unitRef="CNY" decimals="2">5742894789.02</cn-f:货币资金>
<cn-f:货币资金 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">10743695198.09</cn-f:货币资金>
<cn-f:货币资金 contextRef="ins_20061231_p" unitRef="CNY" decimals="2">4805847947.27</cn-f:货币资金>
<cn-f:交易性金融资产 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">2304725.76</cn-f:交易性金融资产>
<cn-f:应收账款 contextRef="ins_20071231_c" unitRef="CNY" decimals="2">864883012.55</cn-f:应收账款>
<cn-f:应收账款 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">364609673.22</cn-f:应收账款>
<cn-f:预付账款 contextRef="ins_20071231_c" unitRef="CNY" decimals="2">8284197850.73</cn-f:预付账款>
<cn-f:预付账款 contextRef="ins_20071231_p" unitRef="CNY" decimals="2">3169643.66</cn-f:预付账款>
<cn-f:预付账款 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">2313835552.23</cn-f:预付账款>
<cn-f:预付账款 contextRef="ins_20061231_p" unitRef="CNY" decimals="2">9217883.37</cn-f:预付账款>
<cn-f:其他应收款 contextRef="ins_20071231_c" unitRef="CNY" decimals="2">2764056869.18</cn-f:其他应收款>
<cn-f:其他应收款 contextRef="ins_20071231_p" unitRef="CNY" decimals="2">34386024525.88</cn-f:其他应收款>
<cn-f:其他应收款 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">671005746.39</cn-f:其他应收款>
<cn-f:其他应收款 contextRef="ins_20061231_p" unitRef="CNY" decimals="2">11716514552.79</cn-f:其他应收款>
<cn-f:存货 contextRef="ins_20071231_c" unitRef="CNY" decimals="2">66472876871.40</cn-f:存货>
<cn-f:存货 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">34167114653.21</cn-f:存货>
<cn-f:流动资产 contextRef="ins_20071231_c" unitRef="CNY" decimals="2">95432519188.17</cn-f:流动资产>
<cn-f:流动资产 contextRef="ins_20071231_p" unitRef="CNY" decimals="2">40132088958.56</cn-f:流动资产>
<cn-f:流动资产 contextRef="ins_20061231_c" unitRef="CNY" decimals="2">48262565548.90</cn-f:流动资产>
<cn-f:流动资产 contextRef="ins_20061231_p" unitRef="CNY" decimals="2">16531580383.43</cn-f:流动资产>
```

图 2 XBRL 格式信息披露样式

2.2 XBRL 的层次结构

作为 XML 的一个应用模式, XBRL 的层次结构也有其特点。可以将现有的 XBRL 分为三层: 技术规格 (specification)、分类标准 (Taxonomy) 和实例文档 (Instance Documents)。下面将对这三层做详细介绍。

2.2.1 XBRL 中的技术规格

技术规格, 或称说明、规范, 主要用于定义 XBRL 的各种专门术语, 描述了 XBRL 文件的结构, 详细规定了 XBRL 分类的标准和 XBRL 实例文档的语法和语义。虽然有 XML 元素和属性的语义上的表述, 但 XBRL 规格是一项侧重技术的文件, 目的在于定义一项符合规范的 XBRL 文档。^[4]

2.2.2 XBRL 中的分类标准

分类标准是财务报告发布的语法格式，也部分定义了各会计报表要素的“语义关系”。如“资产 = 负债 + 所有者权益”、“主营业务收入是利润表的要素”、“Assets 表达资产的概念”等等。分类标准由名为 Taxonomy.xsd 的 XML Schema 文档与相关联的五个 XML 链接库文件 (Definition.xml, Calculation.xml, Presentation.xml, Label.xml, Reference.xml) 组成, Taxonomy 定义的是报表的语法形式, 链接库文件定义的是报表语法要素的语义关系。

Definition 链接库描述 Schema 文件中元素概念之间的关系, 这些关系可取 general-special、similar-tuples、essence-alias、requires-element 等四种值, 分别表示一般与特殊的种属关系、不同 XML 视图中的元组间的定义等价关系、概念间的相似关系、跟随出现关系。Calculation 链接库定义了元素间的线性运算关系, 具体关系式为“TO=FROM1*WEIGHT1+FROM2*WEIGHT2+……+FROM-n*WEIGHT-n”。Label 链接库定义了 Schema 文档中的元素与 XML 中标记的对应关系, 实现一个元素与多个标记相关联。Presentation 链接库规定了元素展现的父子关系与兄弟元素的展现次序。Reference 链接库建立了元素到元素涵义解释的权威参考文献链接。Label 与 Presentation 定义的都是 XBRL 实例文档的展示问题, 而 Reference 显然是为便于人类阅读者索取各元素权威解释而设计的, 它们都不涉及信息项的语义关系定义; 定义 Schema 元素语义关系的只有 Definition 与 Calculation。^[5]

2.2.3 XBRL 中的实例文档

实例文档是一个企业根据 XBRL 规范和 XBRL 分类标准做成的财务报表, 它必须要同时满足分类标准的定义和规范的限制。实例文档封装了具体的商业事实 (fact), 根据信息的汇集程度分为条目 (item)、元组 (tuple)、组 (groups) 三个不同层次。其中, 条目通常与一个数字型的事实对应; 元组是事实的联合体, 等同于关系数据库里的一条记录, 组是实例文档的根结点, 由相关联的数据项的集合构成。在不知道分类标准的情况下, 实例文档没有任何意义, 用户需要借助于分类标准和相应的软件才能从 XBRL 实例中提取所需要的数据并加以分析。

XBRL 的层次结构如图 3 所示:

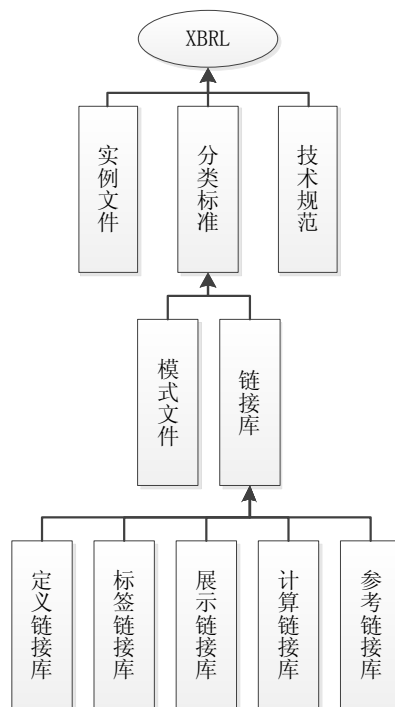


图 3 XBRL 层次结构图

2.3 XBRL 中的自定义元素

根据 XBRL 国际组织关于 XBRL 分类账主要特点的论述及分类账分类标准的论述, XBRL 分类账及分类账分类标准必须满足下列目标:

(1) 多公认会计原则, 将明细信息概括到多 XBRL 财务报告。这也体现了 XBRL 分类账能将交易层面的细节数据与分类账数据和财务报告进行联接。这就意味着可以从报告各项目数据“下钻”至相关明细信息。报告的阅读者将不再仅仅接触抽象的汇总数据, 而是可以追溯其来源, 动态地接触相关明细数据, 更好地进行分析和决策。

(2) 为第三方软件创建标准格式, 以便在客户分类账系统生成会计分录。

(3) 提供从分支机构到合并系统、预算和预测工具、报告工具之间往返移动未过账和已过账信息的标准格式。

(4) 任何类型的强制性审计线索的可扩展性。

(5) 满足任何子分类账需要的可扩展性。

(6) 提供标准格式, 以便于上传分类账信息以及将应付款和应收款主文件和期初余额从一个系统传到另一个系统或者移动信息到网络应用服务供应商或者从网络应用服务供应商收集信息。

(7) 为进行业绩评价提供详细的明细信息。

(8) XBRL 分类标准是为了有效处理组织内的财务和业务信息。这些信息经常分散在不同的会计系统, XBRL 允许使用高效的成本效益方法来汇集、分析和使用数据, 并克服不同会计系统和方法所带来的低效率。

(9) XBRL 分类标准允许描述会计科目表、会计记录或历史交易、财务与非财务数据。它不要求使用标准化的会计科目表来收集信息, 但是它能将会计科目表和会计细节与标准会计科目表联系起来以改善业务中的信息交流。^[6]

XBRL 是一种可扩展的商业报告语言, 其优点之一就是具有扩展性。Cohen(2004)指出, 尽管分类标准包含巨额信息, 但其仅是现行共同报告实践的综合, 不可能满足每一个公司的特定信息报告要求, 这必然导致 XBRL 格式的财务报告不能完全反映传统财务报告或公司报告实务的所有内容。XBRL 的可扩展性解决了上述问题, 但同时它也存在潜在威胁, 如果在国际标准组织和国家都没有接受或监管的情况下, 各企业都根据自身的偏好自行向 XBRL 财务报告中添加元素以满足其需要, 将会导致企业间信息可比较性的损失。因此, 解决该问题的重要措施是: 首先根据某一国家或组织的企业财务信息披露规则、会计准则和会计制度等要求, 定义财务报告所要求披露的基本共同元素, 这个定义了共同元素的分类标准称为“基准分类”。然后再基于“基准分类”, 根据不同的目的, 对“基准分类”进行扩展, “扩展分类”运用了“基准分类”里的元素或分享了这些元素中的重要部分。即扩展分类=基准分类+调整元素。其中调整元素是指需要从基准分类中增加或删除的元素。增加的元素是指那些未包括在基准分类中, 但这些元素是某一实体所需要的元素, 因此需要加入基准分类中的元素。拓展分类的创建还包括需要从基准分类中删除的元素, 这些元素是指包括在基准分类中, 但不是某一实体所要求的元素, 因此不需要包括在扩展分类中。^[7]

2.4 本章小结

XBRL 主要是指提供企业决策者的经营管理信息。XBRL 最初称为 XFRL (XML based Financial Report Mark-up Language), 即基于 XML 的会计报表标记语言, 主要是设想为投资人士, 交易方提供财务信息披露用的, 但是, 后来发现, 该语言更可以用于企业内部等更多情况, 所以改称为“商业报告语言”。

XBRL 的作用很广泛, 企业的各种信息, 特别是财务信息, 都可以通过 XBRL 在计算机

互联网上有效地进行处理。信息发布者一旦输入了信息，就无需再次输入，通过 XBRL 就可以很方便地转换成书面文字，PDF 文件，HTML 页面，或者其他相应的文件格式。而且，通过 XBRL 获取到的信息，也无需打印或再次输入，就可以方便快捷地运用于各种财务分析等领域。

第三章 信息树可视化技术研究现状

3.1 信息可视化与信息树

所谓信息可视化 (information visualization, InfoVis 或 IV) 就是利用人们对可视模式的自然识别能力, 将数据、信息以及知识转化为视觉形式的过程。在这个过程中, 人们利用计算机系统从屏幕上观察交互图形、图像, 并通过可视模型处理信息。它融合了科学可视化特征, 是建立在人机交互界面设计、数据挖掘、图形学以及制图理论基础上的领域。^[8]

信息树最适合表示具有层次关系的文本信息, 使文本信息更直观明了, 便于用户的理解和利用。具体来说, 它是一种层次式或类似树状的资料结构, 将相关领域的电子资源依据它们的内容分成若干类别, 每一类别下又按照文件内容分成更细的类别, 一直到类别下的文件数量达到限定的数目为止。用户在使用信息树进行浏览时, 先确定自己所要查找的内容应归属的类别, 然后逐层选择适当的分支, 进而使范围逐级缩小, 最后确认信息需求, 在具体实施的过程中, 信息树不仅可以过滤掉不相关的信息, 同时也为用户呈现出某一领域的知识结构体系。树型结构是用来描述层次信息的较为常见的方式。但对于大型的层次结构, 这种传统的树型结构的分支很快就会拥挤交织在一起, 变得混乱不堪。^[9]

3.2 层次信息可视化技术的发展现状

层次信息可视化的方法主要分为两大类: 节点连接法和区域填充法^[10]。下面将结合例子分别介绍这两大类方法。

3.2.1 节点连接法

节点连接法是在欧拉空间或双曲空间内用直线或曲线段连接父子节点的方法, 如 ConeTrees^[11], Hyperbolic Trees^[12], DOITrees^[13], SpaceTree^[14]等。

ConeTrees 的根节点位于三维空间中的顶部, 子节点连接在其父节点的下方, 应用于 Unix 文件系统, 能够在在一个界面内显示 600 个文件夹、1 万个文件, 其中有许多节点被覆盖。

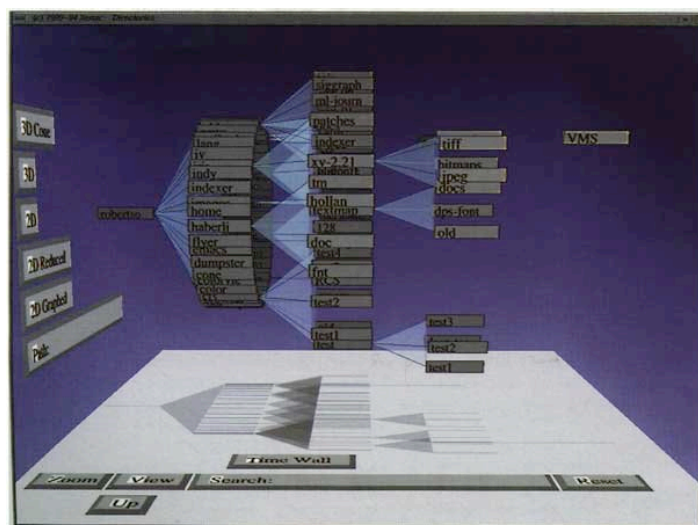


图 4 ConeTrees 例图

Hyperbolic Trees 在一个圆形显示区域内用双曲线连接父子节点, 在 600x600 像素的空间内能够显示 1000 个节点。

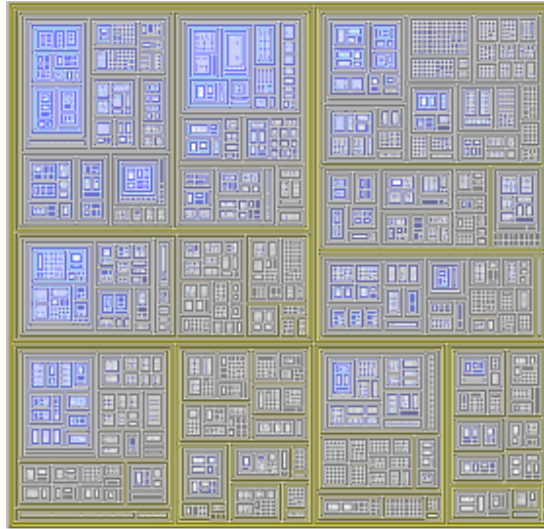


图 7 Treemap 例图

径向区域填充法^[19]采用径向排列的多层圆环表示层次信息，圆环被划分为多个扇形区域，每一扇形表示一个节点，外层圆环的部分扇形是对应内层圆环扇形的子节点。径向区域填充法能够在有限的空间内提供大量信息的整体视图，层次结构关系也比较清晰，但是，随着子节点数目的增多，很多外层扇形被排列得十分狭长，不易分辨。我们基于 Treemap 提出了基于嵌套圆的可视化方法，以圆表示树节点，以圆的嵌套关系表示不同层次的父子节点之间的层次结构关系，以一组相外切的圆表示同层次的一组兄弟节点该方法能够在有限区域内对含有 100 万节点的海量层次信息进行可视化。另一方面，虽然上述方法能够在有限显示空间内呈现大量信息，但随着树节点数目和层数的增多，深层次子节点逐渐变小而难以分辨，需要通过交互技术对其详细信息进行可视化呈现。Furnas^[20]与 Card^[21]等人指出，用户在交互探索大规模可视信息时，倾向于在探索局部焦点信息的详细视图的同时，能够保持信息整体视图的可见性，并且能够感知焦点的上下文信息、保持焦点的上下文信息的一致性。

“焦点+背景”交互式可视化技术^[22]是解决此类问题的主要技术。Furnas 提出了兴趣度 (degree of interest, 简称 DOI) 函数对信息空间进行变形计算，在保持整体视图可视化的前提下，对用户交互操作选中的焦点区域 (focus) 放大，将焦点上下文信息 (context) 逐渐变小。Bederson^[23]与 Paek^[24]将其用于一维信息可视化的交互，应用于菜单与网络搜索结果列表的交互式可视化。Bederson 将其用于二维信息可视化^[25]，应用于表格日历的交互式可视化。Sarkar^[26]提出了针对图结构的 Focus + context 交互式可视化的数学模型，Darling^[27]与 Christian^[28]将其应用于节点和边组成的图形。对于焦点上下文的研究是 Focus + context 交互式可视化技术中的一个重要问题，Gutwin^[29]，Fedak^[30]，Liu^[31]及 Carpendale^[32]等人对交互过程中焦点上下文的变形对用户完成任务的影响进行了研究，通过对焦点上下文进行感知、跟踪、标注等方法，对 Focus + context 交互式可视化技术进行改进。

3.3 焦点-背景信息树技术

随着生物信息技术的发展，所需要处理与分析的数据比以往要多得多。海量数据集不仅是对聚类技术提出了新的要求，同时也对信息可视化技术的挑战。为迎合处理海量数据集需求，可视化专家提出了“焦点+背景”技术。这种技术提供了在放大某一子集合的数据时，同时在视图中保留数据整体结构。用户在使用信息树进行浏览时，先确定自己所要查找的内容应归属的类别，然后逐层选择适当的分支，进而使范围逐级缩小，最后确认信

息需求，在具体实施的过程中，信息树不仅可以过滤掉不相关的信息，同时也为用户呈现出某一领域的知识结构体系。

3.3.1 半径树

Radial tree 是由美国印第安纳大学开发的一项基于“焦点—背景” (focus and context technique) 的可视化技术。浏览界面大致由查询和可视化两大区域组成；在可视化区域中，通过结点、 连线、 颜色以及结点间的层级关系等要素完成对树型结构的可视化；同时支持检索和浏览。

选择一个结点并将其置于半径树的中心，与此同时其周边的树型结构也随之发生相应的变化。这种被称之为“slow-in, slow-out”的设计技术，不同于传统的线性转换表现形式，它的优越性表现在提供了信息的连续性和降低了信息的无序性。^[33]

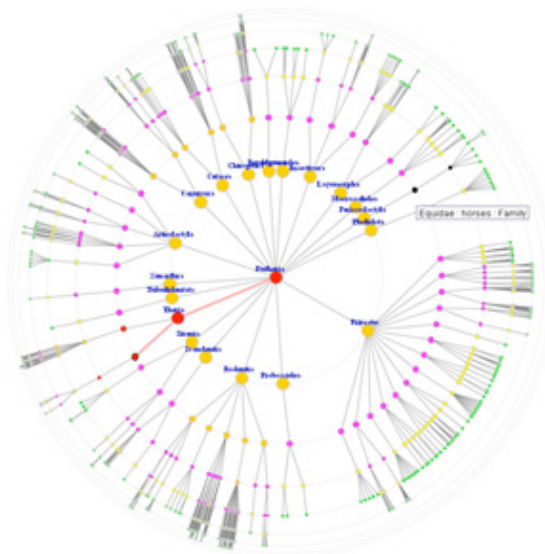


图 8 美国印第安纳大学开发的半径树可视化技术

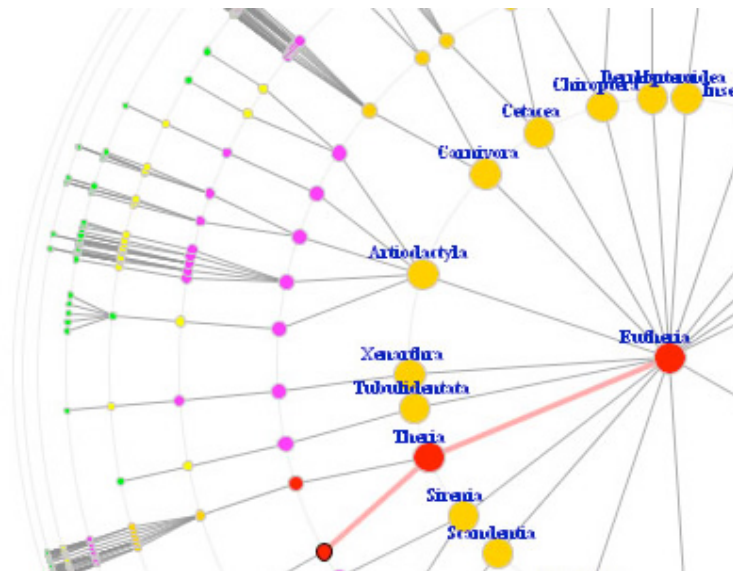


图 9 半径树局部图

3.3.2 双曲线树

同半径树相类似，双曲线树采用的也是“焦点+背景”技术的浏览界面，主要应用于资源的访问、管理、组织以及对对象产品目录、文献集和万维网环境等有层级关系特性的资源的开发利用。

在双曲线树中，代表某一主题的结点连接了许多信息对象，而每个信息对象又连接了若干属性值。鼠标点击某一结点（也可通过拖动结点来变换节点关系），则该节点立即移往图像的中心点，通过整个图像的伸缩变形与位移，将与之联系的对象与属性重新布局：接近中心节点的区域会被放大，反之，双曲线树会在单位圆边缘以极小的空间显示节点。

双曲线树技术最初是受图 10 中 Escher 木雕的启发。图中最显著的两个特征是：一、越靠近边缘的画像越小；二、画像个数呈指数级增长。



图 10 双曲线树浏览器的最初灵感来源

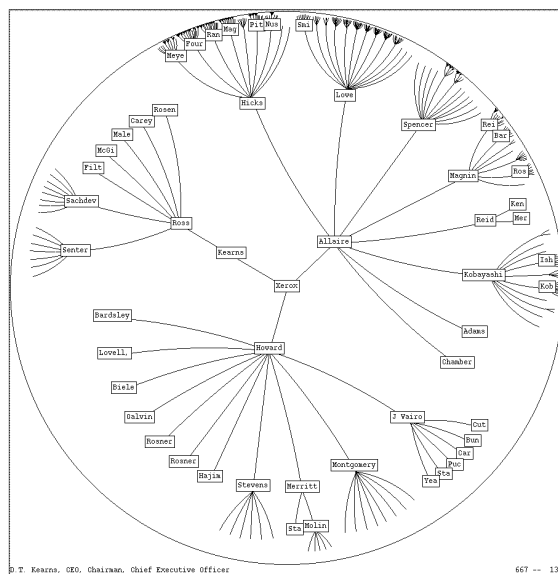


图 11 双曲线树浏览器样图

3.4 传统树形结构与信息树的比较

同传统的树型结构相比较，信息树可视化界面有着显著优越性。其优越性如下：

(1) 在有限的窗口界面中最大限度地显示检索结果。由于传统检索界面通常将检索结果以线性结构的方式排列，势必造成信息显示空间利用的不合理性，用户只能通过“翻页”或链接来浏览结果。而信息树利用“焦点+背景”的可视化技术，有效解决了这个问题。

(2) 不会产生信息迷航。信息树在对信息对象组织、整理以及管理的层次结构模式上，

以及半径树的颜色标识功能、二维双曲线树的放大功能和三维双曲线树的过滤功能等方面,都大大提高了信息查询的导航性。

(3) 查询界面和可视化界面的有效整合。在信息树可视化界面中,用户不仅能通过改变结点位置的方式浏览查询结果,也能够通过改变相应的查询提问从而使可视化结果得到相应变化的方式来满足自己的查询要求。这种整合使得检索的效率和质量都得到了很大的提高。^[9]

3.5 本章小结

为了解决全局图显示上的问题,一个比较好的方法是运用“焦点+背景”技术将抽象信息可视化。在提供给用户背景信息的基础上,通过显示一个特定结点的细节以及紧临结点,将那些对用户理解全局上下文所需要的非常有用的局部信息提供给用户。

第四章 双曲线层次信息可视化技术介绍

Lamping 在 1995 年提出双曲线树浏览器可视化系统, 它为大量层次信息结构进行“焦点+背景”显示提供了一个有效的方法: 利用双曲线的指数增长原理来分配空间。在系统中, 启用树的隐喻方式来表征各个结点之间的空间关系。树的根结点位于一个圆形平面的中间, 占据较多的显示空间, 树的各个分支节点向圆的外周延伸, 离中心越远, 占据的显示空间越少。用户通过拖动平面中的结点, 可以改变该结点所占据的显示空间的大小, 从而达到浏览整个系统的特定区域的目的。比如, 在图上有一个结点位于圆周边上, 把它拖到图的中间, 它的显示空间就会增大 5~10 倍。^[34]

4.1 双曲几何

双曲几何又名罗氏几何(罗巴切夫斯基几何), 是非欧几里德几何的一种特例, 专门研究当平面变成鞍马型之后, 平面几何中有多少定理可以适用, 以及会有什么特别的现象产生。

4.1.1 非欧氏空间

欧几里得的《几何原本》提出了五条公设, 头四条公设分别为: 1.由任意一点到任意一点可作直线; 2.一条有限直线可以继续延长; 3.以任意点为心及任意的距离可以画圆; 4.凡直角都相等。

第五条公设说: 同一平面内一条直线和另外两条直线相交, 若在某一侧的两个内角的和小于两直角, 则这两直线经无限延长后在这一侧相交。

长期以来, 数学家们发现第五公设和前四个公设比较起来, 显得文字叙述冗长, 而且也不那么显而易见。有些数学家还注意到欧几里得在《几何原本》一书中直到第二十九个命题中才用到, 而且以后再也没有使用。也就是说, 在《几何原本》中可以不依靠第五公设而推出前二十八个命题。因此, 一些数学家提出, 第五公设能不能不作为公设, 而作为定理? 能不能依靠前四个公设来证明第五公设? 这就是几何发展史上最著名的, 争论了长达两千多年的关于“平行线理论”的讨论。

到了十九世纪二十年代, 俄国喀山大学教授罗巴切夫斯基在证明第五公设的过程中, 他走了另一条路子。他提出了一个和欧式平行公理相矛盾的命题, 用它来代替第五公设, 然后与欧式几何的前四个公设结合成一个公理系统, 展开一系列的推理。他认为如果这个系统为基础的推理中出现矛盾, 就等于证明了第五公设。

从罗巴切夫斯基创立的非欧几何学中, 可以得出一个极为重要的、具有普遍意义的结论: 逻辑上互不矛盾的一组假设都有可能提供一种几何学。

罗巴切夫斯基几何的公理系统和欧几里得几何不同的地方仅仅是把欧式几何平行公理用“从直线外一点, 至少可以做两条直线和这条直线平行”来代替, 其他公理基本相同。由于平行公理不同, 经过演绎推理却引出了一连串和欧式几何内容不同的新的几何命题。

我们知道, 罗氏几何除了一个平行公理之外采用了欧式几何的一切公理。因此, 凡是不涉及到平行公理的几何命题, 在欧式几何中如果是正确的, 在罗氏几何中也同样是正确的。在欧式几何中, 凡涉及到平行公理的命题, 在罗氏几何中都不成立, 他们都相应地含有新的意义。

所以罗氏几何中的一些几何事实没有像欧式几何那样容易被接受。但是, 数学家们经过研究, 提出可以用我们习惯的欧式几何中的事实作一个直观“模型”来解释罗氏几何是正

确的。

4.1.2 双曲几何变换

双曲几何是在上世纪末发展起来的非欧式几何之一。用它来展示层次信息结构有两个原因：首先，使用已知的映射方法可以很优雅的画出基于焦点+背景技术的可视化视图；其次，我们可以为每个节点分配相同空间并且不会互相冲突，因为双曲线几何中面积是欧式几何中的指数倍。

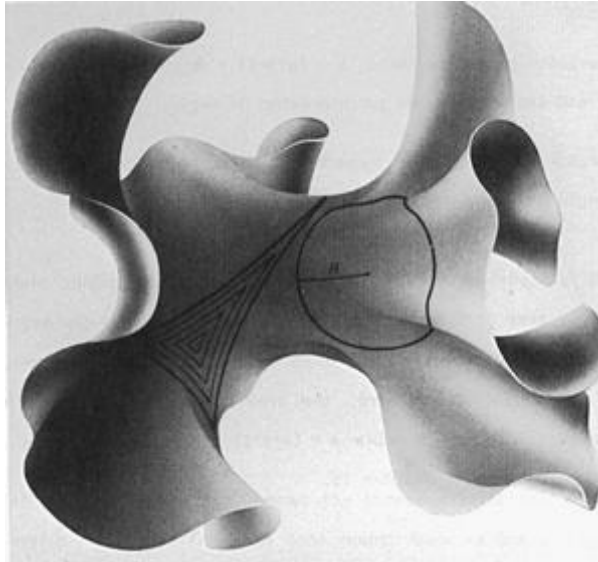


图 12 双曲线几何中的指数级容量

理论上来说，双曲线几何中的空间要比欧式几何中的大，就像图中所展示的将双曲面嵌入欧式的三维空间。在双曲面上的一个圆周随半径呈指数级增长，而不是像欧式空间中的逐渐增长。

在双曲线空间中，周长和面积随圆的半径呈指数级增长。因此，理论上来说，双曲线几何中的空间比欧几里得几何中大，因为欧式几何中的圆周长和面积是呈多项式增长的。^[35]

双曲线圆的周长随半径指数级增长，周长公式为 $2\pi \sinh r$ ，而欧式几何中的公式为 $2\pi r$ 。

图 12 展示了 2 维双曲面中的一个圆盘嵌入 3 维欧式空间中的情形，意在提供直观的说明确明双曲面比欧氏平面多多少空间。可以想象，在双曲面投影到圆盘上时，可以比原来的欧式几何中的圆盘多容纳多少节点。

欧式几何中的两条平行线是等距的，而双曲面上则不然。图 13 所展示的是欧式几何和双曲线几何中平行线的对比。左边是欧式空间中的平行线，右边是双曲面模型中的平行线。欧式空间中的平行线处处等距，而双曲面模型中的则不然。

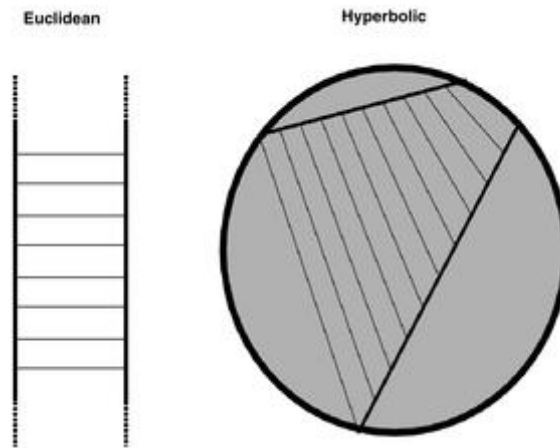


图 13 欧式几何中等距的平行线与双曲线几何中分歧的平行线。

4.2 双曲面模型

4.2.1 Klein 模型

在 Klein 模型中，双曲面就是一个单位圆盘，也就是在欧式几何中单位圆的内部区域。这个模型中，欧式几何中的“点”和“线”的概念都和该模型中的一样。

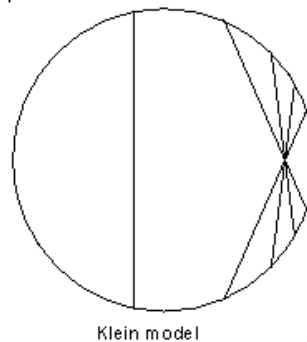


图 14 Klein 模型中的直线概念

从图 14 中可以看出这个模型和欧式空间很相似。然而，这两者之间是有不同的。在欧式空间中，直线是无限长的，因此，对于“距离”这个概念我们还是需要重新定义，定义如下：如果 (x, y) 和 (u, v) 是欧式坐标中的两个点的坐标，那么 Klein 模型中，这两点间的双曲线“距离”是：

$$\cosh^{-1} \frac{1 - xu - yv}{\sqrt{(1 - x^2 - y^2)(1 - u^2 - v^2)}} \quad (1)$$

由于在 Klein 模型中，圆盘内的任意一点到圆周的“距离”都是无限长的，所以在模型中“直线无限长”的假设仍然成立。另外，模型中“角度”的概念与欧式几何中所定义的也不一样。从图 15 中可见，在 Klein 模型中的正五边形的五个角的角度是不同的。

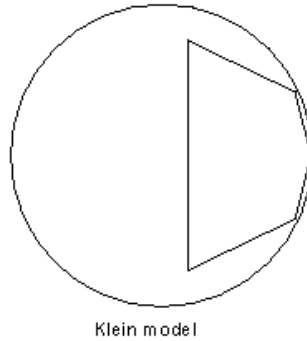


图 15 Klein 模型中的角度概念

4.2.2 上半平面模型

在上半平面模型中，欧式几何中的上半平面就是双曲面。这个模型中的“直线”是圆心在边界线上的圆弧，如图 16 所示。

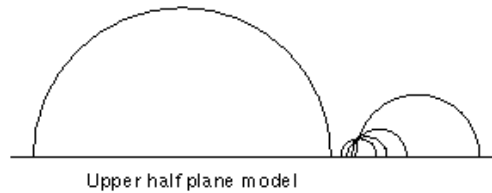


图 16 上半平面模型中的直线概念

欧式坐标 (x, y) 和 (u, v) 之间的“距离”为

$$\cosh^{-1}\left(1 + \frac{(x-u)^2 + (y-v)^2}{2yv}\right) \quad (2)$$

这样从任何点到 x 轴的“距离”是无限长的，因此“直线无限长”的假设仍然成立。和 Klein 模型不同的是，上半平面模型中的“角度”和欧式几何中的一样，如图 17 所示，通过对模型中的“直线”画切线可以得到角度。

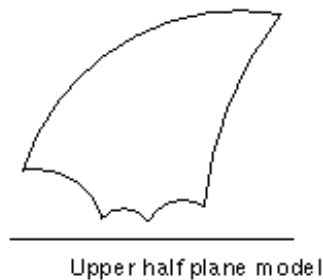


图 17 上半平面模型中的角度概念

4.2.3 Poincare 模型

在 Poincare 模型中，双曲面是单位圆盘，“点”的概念和欧式几何中的一样。“直线”是圆盘上与边界垂直相交的圆弧，如图 18。复平面中的两点 z 和 w 间的双曲线“距离”是

$$2 \tanh^{-1} \left(\left| \frac{z-w}{1-wz^*} \right| \right) \quad (3)$$

这里 z^* 是 z 的共轭复数。

这就像 Klein 模型中的一样，到圆盘边界的“距离”是无限长的。Poincare 模型中的“角度”就和欧式几何中的一样，如图 19，这点与上半平面模型相似。图 20 是在 Poincare 模型中两个完全相同的经过平移的等八边形。^[36]

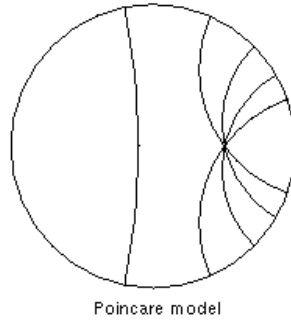


图 18 Poincare 模型中的直线概念

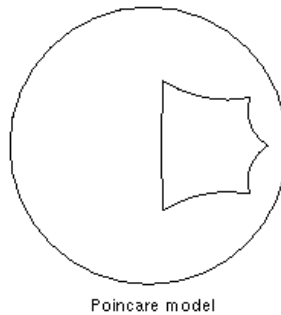


图 19 Poincare 模型中的角度概念

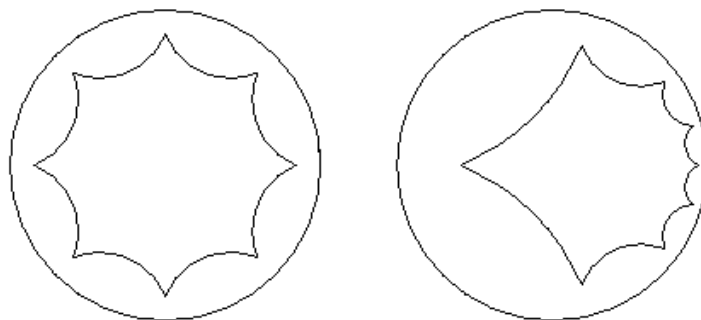


图 20 Poincare 模型中经过平移的两个完全相等的正八边形

4.3 本章小结

在继承了传统树型结构界面的优点外，双曲线树能够支持更纷繁复杂的层级关系。根据试验，在 600*600 像素的窗口中，传统二维树可显示 100 个结点，而双曲线树可显示 1000 个结点，并在中心点的附近有 50 个与主题相关的信息对象，其中每一信息对象都包含从 3 个甚至到几十个数目不等的对象属性。显然，除了在数量上是传统树结构的 10 倍外，这种

逐级分层的可视化表达方式，不会使用户产生信息迷航的困惑。双曲线树的这些优点使得对于 XBRL 的可视化变得非常轻松。

第五章 XBRL 双曲线树浏览器的实现

该浏览器将树形结构的 XBRL 元素以双曲线树的形式展示，由树形 XBRL 数据作为输入，输出基于“焦点+背景”技术的双曲线树，并对用户提供良好的交互体验。

5.1 双曲线树可视化方法介绍

5.1.1 排版方法

在欧式平面中排版的树形结构不可避免的问题在于节点随层次呈指数级增长，而展示节点的空间确是多项式级增长。我们知道，圆的周长和面积是随半径多项式级增长，如果要让节点之间不互相重叠的话就要对更深层次的节点分配更小的空间。这样在查看节点时，由于根节点与叶子节点大小上的差异，每次只能看到一部分相邻的节点。在查看整棵树的结构性，叶子节点就会显得很小而无法查看；而在放大某一区域查看叶子节点时，又会看不到整体结构。

将一棵树排版在双曲面中是非常容易的，因为圆的周长和面积随半径指数级增长，空间是很大的。双曲线树浏览器采用了递归算法将每个节点根据自身信息放置。每个节点都会被分配一个楔形区域，从自身扩展，来放置其子孙节点。该节点所有的子节点都沿着圆弧，与节点自身等距离放置。每个子节点又会被分配一个子楔形区域来放置它的子孙节点。

双曲线树浏览器在双曲面上对树进行排版，然后将其投射到圆形平面上。这种方法取代了传统的在欧式平面中对树进行排版的方法。如果树的焦点发生变化，只需在双曲面中对树的节点进行刚性变化。

在双曲面中放置节点，首先要计算与父节点之间的距离，然后计算角度与宽度，随后对树中的所有节点进行递归与遍历。

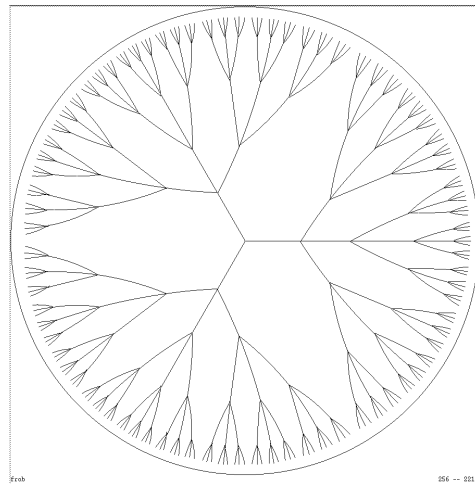


图 21 双曲线树的初始排版

5.1.2 映射算法——Poincare 圆盘模型

要显示一棵在双曲面中排版的树，必须将其映射到欧式二维平面。将双曲面映射到圆盘平面有两种标准方法——Poincare 模型和 Klein 模型。两种方法都是将一个节点置于圆盘中央，其他的节点以透视的形式排布在圆盘边缘。这里的双曲线浏览器使用的是一种保角的映射方法，也叫做 Poincare 模型。这种模型保留了双曲面中的角度，但将双曲面中的直线扭曲变换为圆弧。反之，保留直线而变换角度的是 Klein 模型。

Poincare 模型将双曲面几何中的直线变换为一段圆弧,这段圆弧将与圆盘边界互相垂直。两条不相交的圆弧相当于双曲面中的两条平行线;两条正交的圆弧相当于双曲面中的两条垂直线;而在圆盘边界相交的圆弧则是一组射线。

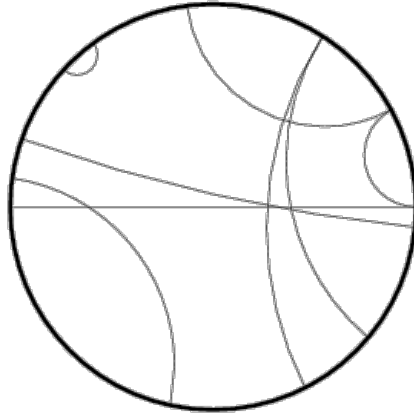


图 22 Poincare 圆盘模型中的直线

可视化双曲线几何最常用的模型就是 Poincare 圆盘模型。Poincare 圆盘将一个无限大的双曲面映射到了一个圆内,在该圆内直线都是以圆弧的形式出现,并与圆周垂直相交。当我们从双曲面的原点渐渐离开时,空间会渐渐增大。因此当我们接近 Poincare 圆盘的周界时,空间的生长是指数级的,可以不断划分为无限多的空间。

一般情况下, Poincare 圆盘的假定半径为 1,但设置为其他值依然可行。

欧式空间和双曲空间最主要的区别在于曲率。Poincare 模型中直线的映射是圆弧说明了这一点。这些圆弧是在 Poincare 圆盘之内,而其所在圆的圆心在圆盘之外,或者在圆盘的圆周上。通过圆盘圆心的线是一个半径为无限长的圆弧,因此看上去像直线一样。

5.1.3 坐标变换算法

双曲面浏览器的用户可以通过点击界面上的节点将其置于界面中央,或是将节点拖曳至其他位置来变化焦点。在两种情况下,其余的节点与连线都会适当变形。朝中央靠近的区域会被放大,而向边缘移动的区域会缩小。图 23 和图 24 展示的是同一棵树,但焦点不同。

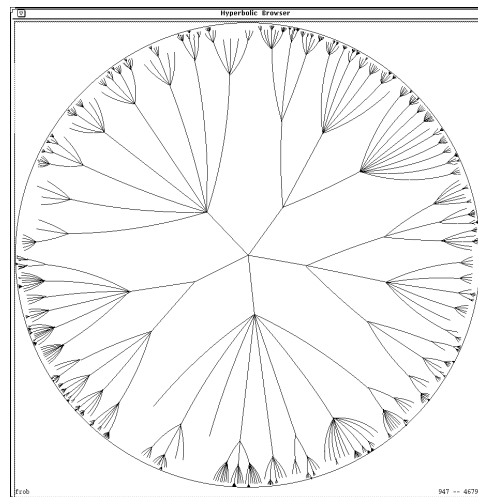


图 23 树的根节点在界面中央

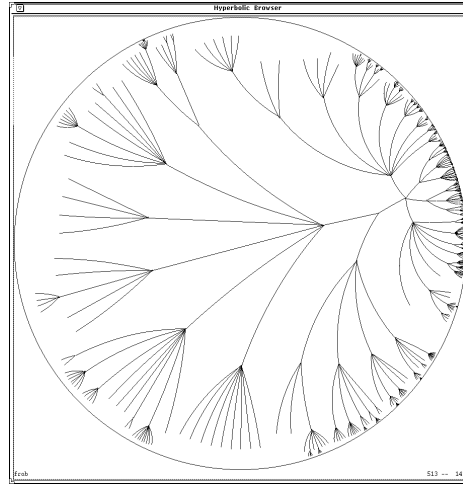


图 24 树的焦点改变

5.1.4 节点信息展示

在一般的树形结构浏览器中，一般使用长方形的标签来显示节点的信息。而本设计所需显示的节点信息比较长，因此如果使用长方形标签来显示的话难免造成重叠和覆盖。因此本程序算出具有足够显示空间的节点，并直接将节点信息字符串显示在节点附近。当用户将鼠标放在节点上时会弹出提示条显示节点信息。

5.2 双曲线树浏览器结构

在实现双曲线树浏览器的过程中，运用了接口、代理等设计模式，程序的主要结构包含以下几个重要组件：

节点模型：存放节点信息；

节点绘制模块：根据节点模型来绘制节点；

坐标变换模块：整棵树在双曲空间中的移动，以及映射到屏幕后坐标变化；

双曲线模型中的直线绘制：应用 Poincare 圆盘模型来绘制双曲空间中的“直线”；

事件处理模块：处理鼠标监听事件，本程序中是拖拽事件；

节点信息标签：绘制节点信息的模块；

用户界面接口：用于显示的界面接口；

通用节点接口：要使用节点模型和节点绘制等组件需要实现通用节点接口。

程序的主要结构图如图 25：

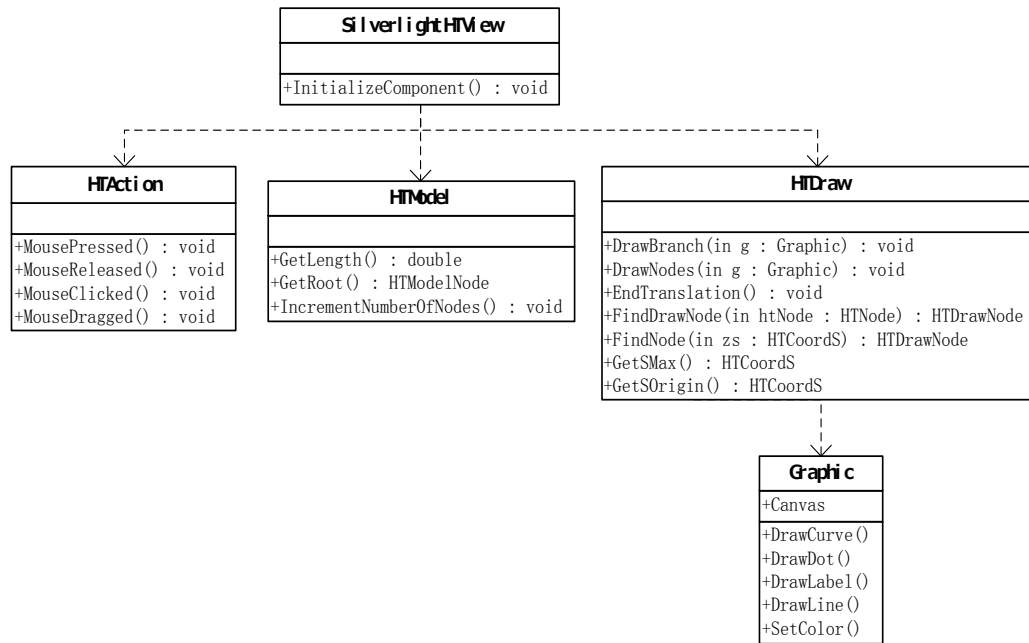


图 25 程序主要 UML 图

5.3 输入数据的 XML 结构

本文的双曲线树浏览器的输入数据采用 XML 可扩展标记语言来存储。XML (Extensible Markup Language) 即可扩展标记语言，它与 HTML 一样，都是 SGML(Standard Generalized Markup Language,标准通用标记语言)。Xml 是因特网环境中跨平台的，依赖于内容的技术，是当前处理结构化文档信息的有力工具。扩展标记语言 XML 是一种简单的数据存储语言，使用一系列简单的标记描述数据，而这些标记可以用方便的方式建立，虽然 XML 占用的空间比二进制数据要占用更多的空间，但 XML 极其简单易于掌握和使用。

本程序中所要展示的每一个 XBRL 元素在 XML 文件中定义为元素名为“entry”的元素节点，并添加属性“name”，设置该元素节点的“name”属性的值为 XBRL 元素的名称；如果当前 XBRL 节点为公司自定义的非行业标准 XBRL 元素，则为该节点添加“color”属性，并设置其值为“red”。

XML 文档结构图如图 26 所示：

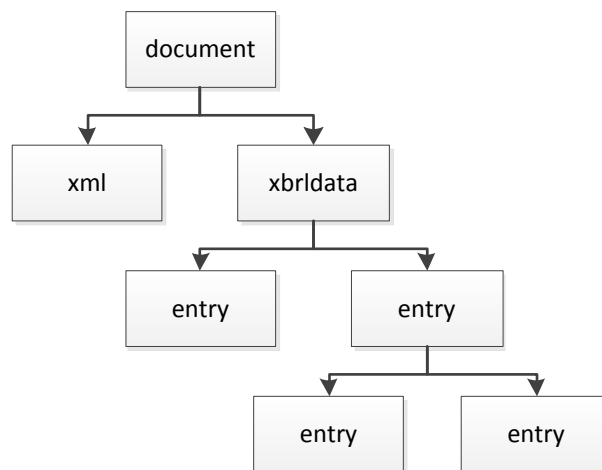


图 26 XML 文档结构图

5.4 标准与自定义 XBRL 元素的显示

所有 XBRL 展示链接库中所定义的元素均以双曲线树的形式展现在双曲线树浏览器中。行业标准所规定的元素用黑色节点和黑色标签信息来表示；公司自定义的元素用红色的节点和标签信息来表示，见图 27 与图 28。另外，连接线的颜色也随分支下自定义节点个数的变化而改变——分支下的自定义节点越多，则连接线的颜色越红，反则越黑。

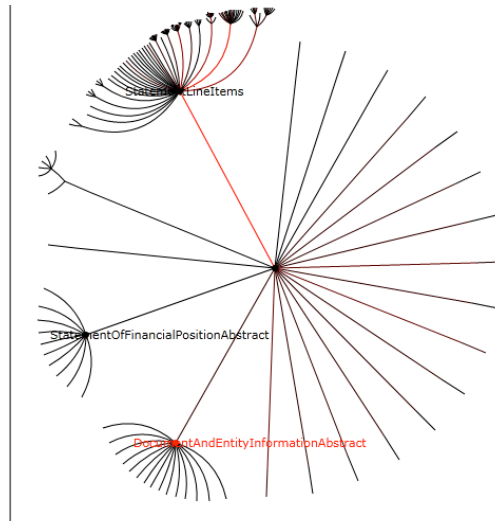


图 27 初始状态下自定义元素的对比显示

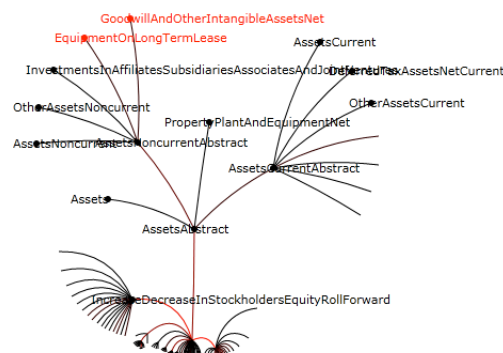


图 28 自定义叶子节点的对比显示

5.5 本章小结

本章介绍了双曲线树浏览器的实现方案，以及实现细节。分别从双曲线树可视化的四个重点——排版方法、映射模型、焦点变换以及节点信息的展示进行描述，并对程序的结构以及输入数据的格式做了介绍。同时，为了使程序更有应用价值，对行业标准 XBRL 节点与企业自定义的 XBRL 节点做了区分显示。

第六章 结论

6.1 总结

本课题的研究内容是大型层次结构信息的可视化技术，这里的大型层结构信息来源于 XBRL 电子财务报表的数据内容。通过对具有大型层次结构的 XBRL 数据的展示，提供财务及分析人员对于 XBRL 元素的可扩展性的分析依据便于专业人员对于自定义报表数据的趋势研究以及标准修订。

本课题所实现的双曲线树浏览器取代了传统的在欧式平面上的树形结构排版，通过将树形结构先排版在双曲面中然后映射到一个圆形平面上。当双曲线树的中心焦点发生改变时，我们通过将树形结构在双曲面上进行刚性变换，而不改变树形结构的原始排版结构，因此，树的排版只需要在初始化时进行一次就够了。节点信息的显示位置也是在排版时就计算好的，在每次焦点变换移动时进行更新。

双曲线树的输入数据由 XML 文件提供，利用 XML 标记语言的树形数据结构以及自我描述特性来存放 XBRL 数据并区分自定义节点，方便节点的展示。

6.2 展望

本课题所实现的双曲线浏览器应用于 XBRL 企业财务数据元素的展示，将行业标准的 XBRL 元素与企业自定义元素区分开来，以便于财务人员对于新标准的制定与研究。由于企业财务数据的元素相当多，传统的树形结构在显示节点众多的信息时效果不佳，因此本文采用基于“焦点+背景”技术的双曲线树来展示 XBRL 的层次信息。

本程序最主要的问题在于节点信息的显示上。由于显示的节点名称都比较长，而界面上需要同时显示的节点又很多，因此在显示节点的时候难免会造成文字的重叠。为了使用户能够清楚辨别节点信息内容，程序有提示条功能，也就是用户在移动鼠标至节点时，会显示节点相应的显示名称。然而，这虽然在功能上解决了看不清重叠文字的问题，但界面上的文字重叠仍然是不太友好的。在今后进一步的研究中希望能找出更合适的解决方案。

参考文献

- [1] www.xbrl.org, 2011-3-5
- [2] 曲吉林. XBRL 及其对财务报告的影响[J]财会月刊, 2004,(12)
- [3] <http://xbrl.cninfo.com.cn/XBRL/about-3.htm>, 2011-3-8
- [4] 潘琰. 可扩展企业报告语言及其对会计的影响[J]会计研究, 2003,(01).
- [5] 姚靠华, 洪昀. XBRL 层次结构与财务信息数据挖掘[J]. 会计之友(下旬刊) 2009(1):46-57
- [6] 李立成. 浅谈我国 XBRL 总账分类标准的制定[J]中国管理信息化, 2008,(03)
- [7] 高锦萍. XBRL 财务报告分类标准创建模式研究[J]财会通讯(综合版), 2008,(06)
- [8] 刘永, 邱均平. 信息树与信息可视化[J]档案管理, 2005,(06)
- [9] 方亮. 信息树在信息可视化中的应用[J]图书与情报, 2007,(02).
- [10] Wang WX, Dai GZ. Information visualization. *Advances in Information Science*, 2006,6:80-118
- [11] Robertson GG, Mackinlay JD, Card SK. Cone trees: Animated 3D visualizations of hierarchical information. In: Robertson SP, Olson GM, Olson JS, eds. *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*. New York: ACM Press.1991. 189-194.
- [12] Lamping J, Rao R, Pirolli P.A Focus + Context technique based on hyperbolic geometry for visualizing large hierarchies. In: Katz LR, Mack R, Marks L, Rosson MB, Nielson J, eds. *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*. New York: ACM Press, 1995.401-08.
- [13] Heer J, Card SK. DOITrees revisited: Scalable, space-constrained visualization of hierarchical data. In: Costabile MF, ed. *Proc. of the Working Conf. on Advanced Visual Interfaces*. New York: ACM Press, 2004.421-424.
- [14] Plaisant C, Grosjean J, Bederson BB. SpaceTree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In: Wong PC, Andrews K, eds. *Proc. of the IEEE Symp. on Information Visualization*. Los Alamitos: IEEE Computer Society Press, 2002.57-64.
- [15] Johnson B, Shneiderman B. Tree-Maps: A space-filling approach to the visualization of hierarchical information. In: Nielson GM, Rosenblum L, eds. *Proc. of the 2nd IEEE Visualization Conf*. Los Alamitos: IEEE Computer Society Press, 1991, 284-291.
- [16] Van HF, Van WJ. Beamtrees: Compact visualization of large hierarchies. In: Wong PC, Andrews K, eds. *Proc. of the IEEE Symp. on Information Visualization*. Los Alamitos: IEEE Computer Society Press, 2002.31-39.
- [17] Itoh T, Yamaguchi Y, Ikehata Y, Kajinaga Y. Hierarchical data visualization using a fast rectangle-packing algorithm. *IEEE Trans. on Visualization and Computer Graphics*, 2004,10(3):302-313.
- [18] Balzer M, Deussen O. Voronoi Treemaps for the visualization of software metrics. In: Naps T, paw WD, eds. *Proc. of the 2005 ACM Symp. on Software visualization*. New York: ACM Press, 2005.165-172.
- [19] Chi EH, Pitkow J, Mackinlay J, Pirolli P, Gossweiler R, Card SK. Visualizing the evolution of Web ecologies. In: Karat CM, Lund A, Coutaz J, Karat J, eds. *Proc. of the SIGCHI Conf.*

- on Human Factors in Computing Systems. New York: ACM Press, 1998.400-407.
- [20] Furnas GW. Generalized fisheye views. In: Mantei M, Orbeton P, eds. Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. New York: ACM Press, 1986.16-23.
- [21] Card SK, Maekinlay JD, Shneiderman B. Readings in Information Visualization: Using Vision to Think. San Francisco: Morgan Kaufmann Publishers, 1999.
- [22] Lamping J, Rao R, Pirolli P. A Focus + Context technique based on hyperbolic geometry for visualizing large hierarchies [A]. Proceedings of ACM SIGCHI' 95 [C]. New York: ACM Press,1995:401-408.
- [23] Bederson BB. Fisheye menus. In: Ackerman M, Edwards K, eds. Proc. of the 13th Annual ACM Symp. on User Interface Software and Technology. New York: ACM Press, 2000:217-225.
- [24] Paek T, Dumais S, Logan R. Wave Lens: A new view onto Internet search results. In: Elizabeth DE, Manfred T, eds. Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. New York: ACM Press, 2004:727-734.
- [25] Bederson BB, ClamageA, CzerwinskiMP, Robertson GG. DateLens: A fisheye calendar interface for PDAs. ACM Trans.on Computer-Human Interaction, 2004, 11(1): 90-119.
- [26] Sarkar M, Brown MH. Graphical fisheye views of graphs. In: Bauersfeld P, Bennett J, Lynch G, eds. proc. of the SIGCHI Conf. on Human Factors in Computing Systems. New York: ACM Press, 1992:83-91.
- [27] Darling E, Recktenwald K, Kalghatgi N, Burgman A. Effects of fisheye on visualizing connections between nodes. In: Kellogg W, Zhai S, eds. Proc. of the SIGCHI Conf. on Human Factors in Computing systems. New York: ACM Press, 2005.1328-1330.
- [28] Christian T, James A, Frank H, Heidrun S. Fisheye tree views and lenses for graph visualization. In: Ikeuchi K, Suda Y, eds. Proc. of the 10th Int'l Conf. on Information Visualization. Washington: IEEE Computer Society Press, 2006:17-24.
- [29] Gutwin C. Improving focus targeting in interactive fisheye views. In: Wixon D, ed. Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. New York: ACM Press, 2002:267-274.
- [30] Fedak C, Gutwin C. Improving revisitation in fisheye views with visit wear. In: Kellogg W, Zhai S, eds. Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. New York: ACM Press, 2005:771-780.
- [31] Liu F, Gleicher M. Automatic image retargeting with fisheye-view warping. In: Baudisch P, Czerwinski M, eds. Proc. of the 18th Annual ACM Symp. on User Interface Software and Technology. New York: ACM Press, 2005:153-162.
- [32] Carpendale S, Light J, Pattison E. Achieving higher magnification in context. In: Feiner SK, Landy JA, eds. Proc. of the 17th Annual ACM Symp. on User Interface Software and Technology. New York: ACM Press, 2004:71-80.
- [33] Nihar Sheth, Katy Borner, Jason Baumgartner, Ketan Mane, Eric Wernert. Treemap, Radial Tree, and 3D Tree Visualizations [EB/OL], 2006-07-23. <http://iv.slis.indiana.edu/ref/iv03contest/nihar.pdf>.
- [34] 王哲,胡凤培,葛列众. WWW 系统 Focus + Context 信息可视化研究综述[J]人类工效学, 2004,(01) .
- [35] Paula J. Hane. LEXIS- NEXIS tests data visualization technology [J] .Information Today, 2000, (1).

-
- [36] For a clearer view of data, go climb a “hyperbolic tree”. April, 01, 1999 [EB/OL].
[2006-09-02]. <http://www.highbeam.com/doc/1G1-55727357.html>.

谢辞

本次毕业设计的内容虽然只是四年本科所学的冰山一角，却是为这四年学习所画下的一个完整句号。本科的学习虽然结束，然而求知探索之路正要开始，从此要靠自己的动手实践以及搜索资源来获取新的知识。从这个角度来说，本次毕业设计是一个起点，在本科毕业之际靠自己的努力来完成一个全新的项目。在这里我要感谢在这四年间陪伴过我一起学习生活的所有人。

特别的，我要感谢本次毕设的指导老师马颖华老师，结合我的兴趣给我开设了这个课题，还在毕设过程中为我提出了不少非常有用的建议，使我能够很顺利地进行工作。还要感谢与我同一组的张帅同学，及时为我提供测试用例，并且耐心的帮我讲解他所做的工作。还要感谢与我同一导师的陈邃、李俊彦、臧云波、谭兴龙同学，在初期的几次见面会时从他们的报告中学到不少知识。

另外，还要感谢发明双曲线树的 Lamping 博士，因为他的发明得以是我找到如此优雅而又实用的方式来显示大型结构的树。

VISUALIZING XBRL ELEMENTS IN HYPERBOLIC SPACE

What is XBRL?

XBRL is a language for the electronic communication of business and financial data which is revolutionizing business reporting around the world. It provides major benefits in the preparation, analysis and communication of business information. It offers cost savings, greater efficiency and improved accuracy and reliability to all those involved in supplying or using financial data.

XBRL stands for eXtensible Business Reporting Language. It is one of a family of "XML" languages which is becoming a standard means of communicating information between businesses and on the internet.

XBRL is being developed by an international non-profit consortium of approximately 450 major companies, organizations and government agencies. It is an open standard, free of license fees. It is already being put to practical use in a number of countries and implementations of XBRL are growing rapidly around the world.

The idea behind XBRL, eXtensible Business Reporting Language, is simple. Instead of treating financial information as a block of text - as in a standard internet page or a printed document - it provides an identifying tag for each individual item of data. This is computer readable. For example, company net profit has its own unique tag.

The introduction of XBRL tags enables automated processing of business information by computer software, cutting out laborious and costly processes of manual re-entry and comparison. Computers can treat XBRL data "intelligently": they can recognize the information in a XBRL document, select it, analyze it, store it, exchange it with other computers and present it automatically in a variety of ways for users. XBRL greatly increases the speed of handling of financial data, reduces the chance of error and permits automatic checking of information.

Companies can use XBRL to save costs and streamline their processes for collecting and reporting financial information. Consumers of financial data, including investors, analysts, financial institutions and regulators, can receive, find, compare and analyze data much more rapidly and efficiently if it is in XBRL format.

XBRL can handle data in different languages and accounting standards. It can flexibly be adapted to meet different requirements and uses. Data can be transformed into XBRL by suitable mapping tools or it can be generated in XBRL by appropriate software.

How XBRL Works?

XBRL is a member of the family of languages based on XML, or Extensible Markup Language, which is a standard for the electronic exchange of data between businesses and on the internet. Under XML, identifying tags are applied to items of data so that they can be processed efficiently by computer software.

XBRL is a powerful and flexible version of XML which has been defined specifically to meet the requirements of business and financial information. It enables unique identifying tags to be applied to items of financial data, such as ‘net profit’. However, these are more than simple identifiers. They provide a range of information about the item, such as whether it is a monetary item, percentage or fraction. XBRL allows labels in any language to be applied to items, as well as accounting references or other subsidiary information.

XBRL can show how items are related to one another. It can thus represent how they are calculated. It can also identify whether they fall into particular groupings for organizational or presentational purposes. Most importantly, XBRL is easily extensible, so companies and other organizations can adapt it to meet a variety of special requirements.

The rich and powerful structure of XBRL allows very efficient handling of business data by computer software. It supports all the standard tasks involved in compiling, storing and using business data. Such information can be converted into XBRL by suitable mapping processes or generated in XBRL by software. It can then be searched, selected, exchanged or analyzed by computer, or published for ordinary viewing.

XBRL offers major benefits at all stages of business reporting and analysis. The benefits are seen in automation, cost saving, faster, more reliable and more accurate handling of data, improved analysis and in better quality of information and decision-making.

XBRL enables producers and consumers of financial data to switch resources away from costly manual processes, typically involving time-consuming comparison, assembly and re-entry of data. They are able to concentrate effort on analysis, aided by software which can validate and manipulate XBRL information. As just one example, searches for particular information which might in the past have taken hours can be completed with XBRL in a fraction of a second.

Those who stand to benefit include all who collect business data, including governments, regulators, economic agencies, stock exchanges, financial information companies and the like, and those who produce or use it, including accountants, auditors, company managers, financial analysts, investors and creditors. Among those who can take advantage of XBRL include accountancy software vendors, the financial services industry, investor relations companies and the information technology industry.

Focus-plus-Context Techniques

Information Visualization

Information visualization (InfoVis) produces (interactive) visual representations of abstract data to reinforce human cognition; thus enabling the viewer to gain knowledge about the internal structure of the data and causal relationships in it. The main problem of information visualization is the insufficient space, which restricts the user in showing detail and context contemporaneous, is called “presentation problem”. The Focus + context system allows the user to show detailed information linked with the context, by also having the possibility to focus on other information by interacting with the system.

The basic idea with focus-plus-context-visualizations is to enable viewers to see the object of primary interest presented in full detail while at the same time getting a overview-impression of all the surrounding information — or context — available.

The limitation of the viewing space leads to the problem that either an overview of all data or a zoomed view of an interesting subspace can be shown at the same time with a uniform (linear)

magnification factor.

Hyperbolic Tree

Three Models of the Hyperbolic Plane

The following is a description of some of the models for the hyperbolic plane. In order to understand the descriptions, refer to the figures. They may seem a bit strange. However, a result due to Hilbert says that it is impossible to smoothly embed the hyperbolic plane in Euclidean three-space using the usual Euclidean geometry. (Technical note: In fact it is possible to have a C^1 embedding into R^3 , according to a 1955 construction of Nicolas Kuiper, but according to William Thurston, the result would be "incredibly unwieldy, and pretty much useless in the study of the surface's intrinsic geometry." Since there is no such smooth embedding, any model of the hyperbolic plane has to use a different geometry. In other words, we must redefine words like point, line, distance, and angle in order to have a surface in which the parallel postulate fails, but which still satisfies Euclid's postulates 1-4 (stated in the previous article). Here are brief descriptions of three models:

Klein Model

In the Klein model of the hyperbolic plane, the "plane" is the unit disk; in other words, the interior of the Euclidean unit circle. We call Euclidean points the "points" for our model. We call the portions of Euclidean lines which intersect the disk "lines." See the Klein model in Figure 1. From the above, the model seems similar to Euclidean space. However, there are differences; since we insist that postulate 2 holds, which says that we can make every line infinitely long, we must have a new description of "distance." We define the "distance" between two points as follows: If (x,y) and (u,v) are the Euclidean coordinates of two points, then the hyperbolic "distance" between them in the Klein model is $\text{arccosh}\{(1-xu-yv)/\sqrt{(1-x^2-y^2)(1-u^2-v^2)}\}$. Thus postulate 2 holds in this model, since the "distance" from any point in the disk to the unit circle is infinite. Also, the way we define "angles" in the model is not the same as Euclidean definition of angles. To see this, look at the regular pentagon in the Klein model in Figure 2. As you can see, although all five of the "angles" are the same, the Euclidean angles are not the same.

Upper Half Plane Model

The upper half plane model takes the Euclidean upper half plane as the "plane." Now the "lines" are portions of circles with their center on the boundary, as shown in Figure 1. The "distance" between two points with Euclidean coordinates (x,y) and (u,v) is $\text{arccosh}\{1+[(x-u)^2+(y-v)^2]/2yv\}$. Thus the "distance" from any point to the x-axis is infinite, ensuring that postulate 2 holds in this model. Unlike the Klein model, "angles" are the same as Euclidean angles. Remember, you have to draw tangents to the "lines" to calculate the Euclidean angles. See Figures 1 and 2.

Poincare Model

In the Poincare model, the "plane" is the unit disk, and "points" are Euclidean points. "Lines" are portions of circles intersecting the disk and meeting the boundary at right angles. The "distance" between two points with coordinates z and w in the complex plane is: $2 \text{arctanh}\{|(z-w)/(1-wz^*)|\}$, where z^* is the complex conjugate of z . Thus as in the Klein model, the

"distance" to the boundary of the disk is infinite, and postulate 2 holds. Like the upper half plane model, the "angles" for the model are the same as Euclidean angles. See Figures 1 and 2. Also compare the isometric octahedrons in Figure 3.

Hyperbolic Tree Viewer

Layout

Laying a tree out in the hyperbolic plane is an easy problem, because the circumference and area of a circle grow exponentially with its radius. There is lots of room. We use a recursive algorithm that lays out each node based on local information. A node is allocated a wedge of the hyperbolic plane, angling out from itself, to put its descendants in. It places all its children along an arc in that wedge, at an equal distance from itself, and far enough out so that the children are some minimum distance apart from each other. Each of the children then gets a sub-wedge for its descendants. Because of the way parallel lines diverge in hyperbolic geometry, each child will typically get a wedge that spans about as big an angle as does its parent's wedge, yet none of the children's wedges will overlap.

Mapping

Once the tree has been laid out on the hyperbolic plane, it must be mapped in some way to a 2-d plane for display. (We can barely imagine the hyperbolic plane, not to mention see it.) There are two canonical ways of mapping the hyperbolic plane to the unit disk. In both of them, one vicinity in the hyperbolic plane is in focus at the center of the disk while the rest of the hyperbolic plane fades off in a perspective-like fashion toward the edge of the disk, as we desire. We use the conformal mapping, or Poincare model, which preserves angles but distorts lines in the hyperbolic space into arcs on the unit disk, as can be seen in the takes lines in the hyperbolic plane to lines in the unit disk, but distorts angles. You can't have it both ways.

Change of Focus

The user can change focus either by clicking on any visible point to bring it into focus at the center, or by dragging any visible point interactively to any other position. In either case, the rest of the display transforms appropriately. Regions that approach the center become magnified, while regions that were in the center shrink as they are moved toward the edge. Figure 6 shows the same tree as Figure 5 with a different focus. The root has been shifted to the right, putting more focus on the nodes that were toward the left.

Changes of focus are implemented as rigid transformations of the hyperbolic plane that will have the desired effect when the plane is mapped to the display; there is never a need to repeat the layout process. A change of focus to a new node, for example, is implemented by a translation in the hyperbolic plane that moves the selected node to the location that is mapped to the center of the disk.

Node Information

Another property of the Poincare projection is that circles on the hyperbolic plane are mapped into circles on the Euclidean disk, though they will shrink in size the further they are from the origin. We exploit this property by calculating a circle in the hyperbolic plane around each node that is guaranteed not to intersect the circle of any other node. When those circles are

mapped onto the unit disk they provide a circular display region for each node of the tree in which to display a representation of the node. This can be combined with a facility that uses different representations for nodes depending on the amount of real space they receive.