

# Hyperbolic Treebrowser – der hyperbolische Browser für hierarchische Daten

Hans-Christian Jetter, 01/461316

Lehrveranstaltung “Visuelle Suchsysteme”, theoretische Ausarbeitung

## 1 Einleitung

Hierarchische Daten sind im Alltag und in der Computertechnik allgegenwärtig. Durch klare Strukturierung erlauben sie das schnelle Einordnen oder Auffinden von Objekten innerhalb von Verwaltungsstrukturen, Verzeichnissen oder wissenschaftlichen Hierarchien. Sie dienen zur Organisation großer Dokumentenmengen und mit dem Dateisystem moderner Betriebssysteme befindet sich eine hierarchisch organisierte Datenstruktur in jedem PC und damit einem großen Teil aller Haushalte.

Da eine klare Strukturierung und schneller Zugriff die Daseinsberechtigung für hierarchische Datenstrukturen darstellen, ist es sinnvoll, die vorhandenen Werkzeuge zur Arbeit mit und zur Visualisierung von Hierarchien kontinuierlich zu überprüfen und weiterzuentwickeln.

Ein vielversprechender und teilweise bereits im Einsatz befindlicher Ansatz für eine benutzergerechte Darstellung und Interaktion mit Hierarchien stellt der hyperbolische Browser dar. Dieser soll im folgenden umfassend dargestellt und seine Einsatzgebiete, sowie seine Vor- und Nachteile analysiert werden.

Zur Strukturierung dieser Darstellung wird das von Card et al. dargestellte Referenzmodell der Visualisation Pipeline [Card et al.] und die Task by Data Type Taxonomy von Shneiderman [Shneiderman 96] herangezogen, um den hyperbolischen Browser auch anderen Visualisierungstechniken gegenüber stellen zu können.

## 2 Hierarchische Daten

Hierarchische Daten sind in der Form von Bäumen (siehe Abbildung 1) schon seit den ersten Tagen der Computertechnik bekannt und daher ein erschöpfend behandeltes Gebiet der Informatik.

Prinzipiell lassen sich alle hierarchischen Daten in Baumstrukturen abbilden, wobei dabei die einzelnen Knoten des Baumes ein Objekt verschiedenster Klassen beherbergen (in der Abbildung beispielsweise ein Objekt der Klasse „Buchstabe“). Jedem Knoten können dann wiederum weitere Kinderknoten zugewiesen werden („F“ und „G“ sind Kinder von „C“).

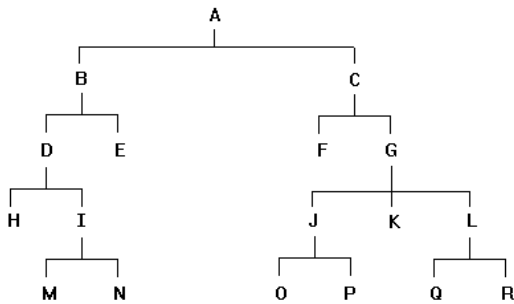


Abbildung 1 - ein Baum als Datenstruktur

Ausgangspunkt der Baumstruktur ist die Wurzel (in der Abbildung Knoten „A“). Auf der untersten Ebene befinden sich die Blätter des Baumes, also die kinderlosen Enden der Zweige (in der Abbildung die Knoten „M“ bis „R“).

Jeder Knoten (z.B. Knoten „Q“) enthält damit nicht nur sein Objekt als Information (Buchstabe „Q“), sondern erhält durch seine Position im Baum Information zu seiner Abstammung („Q ist Kind von L, Enkel von G, Urenkel von C, Ururenkel von A“).

Somit ist also nicht nur der Knoten mit seinem zugewiesenen Objekt, sondern auch seine Position in der Hierarchie – sprich der Kontext in dem der Knoten steht - Informationsträger. Die Topologie des Baumes ist bei hierarchischen Daten daher also genauso von Bedeutung, wie die in den Knoten enthaltenen Objekte an sich.

Neben unzähligen anderen Anwendungen haben sich Bäume beispielsweise bei der Speicherung und dem Auffinden von Daten in Datenbanken (z.B. in der Form von Index-Trees oder Suchbäumen) wegen ihres günstigen Verhältnisses zwischen Speicherbedarf und Zugriffszeiten bewährt. Diese hohe Effizienz kann anhand eines einfachen Beispiels demonstriert werden.

Gegeben seien die Fahrzeugdaten von 40 PKWs, wobei neben anderen Attributen insbesondere der Preis und die Zahl der Sitzplätze bekannt sind. 30 PKWs seien dabei viersitzig, die anderen zweisitzig. Diese Liste von Fahrzeugen wird zunächst in einer flachen Tabelle gespeichert, deren Zeilen die einzelnen 40 Modelle und deren Spalten die jeweiligen Fahrzeugattribute enthalten.

Erhält nun ein Rechner die Aufgabe, die Wagen anzuzeigen, die für 4 Personen Platz bieten und unter 20.000 Euro zu haben sind, so muss er zwangsläufig alle 40 Fahrzeuge in der Tabelle gegen die angegebenen Kriterien überprüfen (40 Vergleichsoperationen).

Bei der Suche in hierarchisch strukturierten Daten kann die Zahl der nötigen Vergleiche auf einen Bruchteil reduziert werden: sind die Daten beispielsweise im Vorfeld so organisiert worden, dass von der Wurzel ausgehend eine Unterteilung in zwei Kategorien erfolgt („zweisitzige PKWs“, „viersitzige PKWs“), so könnte die Zahl der nötigen Preisvergleiche auf die Anzahl der Viersitzer (bzw. der Kinder des Knotens „viersitzige PKWs“) reduziert werden (30 Vergleichsoperationen).

Werden die Viersitzer noch durch eine weitere hierarchische Ebene in Fahrzeuge über und unter 20.000 Euro aufgeteilt, so würden ausgehend von der Wurzel schon zwei Vergleiche ausreichen, um den Knoten zu erreichen, dessen Kinder allen Erfordernissen entsprechen (2 Vergleichsoperationen).

Somit wären also durch geschickte Organisation der Daten im Vorfeld anstatt 40 nur noch 2 Vergleichsoperationen nötig, um die passenden PKWs zu identifizieren.

Die mit der maschinellen Verarbeitung gemachten Erfahrungen lassen sich teilweise auch auf menschliche Arbeit mit hierarchischen Daten übertragen. Im PKW-Beispiel wäre es nicht nur für einen Rechner, sondern auch für einen Menschen wesentlich einfacher, mit der Baumstruktur zu arbeiten, anstatt eine 40-zeilige Tabelle mit mehreren Spalten nach der richtigen Kombination von Attributen zu durchsuchen.

Aus diesem Grund tauchen hierarchische Daten zur Verarbeitung durch den Menschen immer wieder im Alltag auf, angefangen von der Organisation von Gesetzestexten bis hinzu Produktverzeichnissen in Versandhauskatalogen.

Im Rahmen der Information Visualization Pipeline [Card et al.] nimmt die Repräsentation solcher hierarchischen Datenstrukturen im Rechner die Rolle der „Raw Data“ ein.

## 2.1 Hierarchische Daten im Alltag

Ein prominentes weil oft angeführtes Beispiel für hierarchische Daten aus dem Alltag sind die Dateisysteme heutiger Betriebssysteme. Auch wenn diese nicht der „Realwelt“ entstammen, sind sie gut geeignet, um Vorteile hierarchischer Strukturierung zu illustrieren.

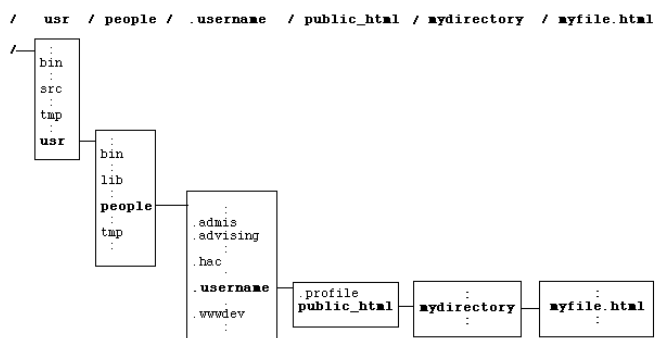


Abbildung 2 - UNIX-Filesystem

Die Wurzel hierarchischer Dateisysteme ist das Root-Verzeichnis „/“ (siehe Abbildung 2). Ein davon ausgehender Zweig wird in der Abbildung exemplarisch dargestellt. Die Knoten enthalten entweder Directories (z.B. „people“), denen weitere Knoten untergeordnet sein können, oder Dateien (z.B. „myfile.html“), die die Blätter des Baumes darstellen.

Um die wichtige Bedeutung der Strukturierung zu erkennen, muß man sich lediglich veranschaulichen, wie der Arbeitsalltag mit PCs aussehen würde, wenn Filesysteme immer noch wie eine flache Tabelle organisiert wären. Bei der heutigen Größe von Massenspeichern würden sich Zehntausende von Dateien auf einer Ebene befinden und dementsprechend würde das Auffinden von Dateien zur anspruchsvollen

Suchaufgabe oder zu stupidem Blättern entarten. Obwohl eine Vielzahl von Attributen wie Dateiname, Größe, Dateistamp etc für die Suche zur Verfügung stehen, können diese den gezielten Zugriff innerhalb einer Hierarchie (Dateiname & Kontext) nicht ersetzen. Eine effiziente Arbeit wäre ab einem gewissen Datenbestand ohne den Kontext schlicht unmöglich.

## 2.2 Hierarchische Daten in der Wissenschaft

Hierarchische Daten finden auch in der Wissenschaft Verwendung, wobei die Ausmaße dieser Hierarchien weit über die aus dem Alltag bekannten hinausgehen können. Ein anschauliches Beispiel für eine Hierarchie zur Kategorisierung von Objekten findet sich in der Biologie mit dem von Carl von Linnés (1707-1778) entwickelten System der Organismen.

Dieses oft modernisierte, aber in den Grundzügen unveränderte System erlaubt eine klare Klassifikation aller auf der Erde beobachteten Arten in der Tier- und Pflanzenwelt. Dazu wird jedem Organismus anhand von 7 Attributen ein Platz innerhalb dieser Hierarchie zugewiesen, aus dem die stammesgeschichtliche Entwicklung und eine eindeutige Benennung hervorgeht. Im folgenden ist die Klassifizierung für ein Tier der Art „*Vombatus ursinus*“ exemplarisch dargestellt:



***Vombatus ursinus*** = Wombat

<b>Reich:</b>	Animalia
<b>Stamm:</b>	Chordata
<b>Klasse:</b>	Mammalia
<b>Ordnung:</b>	Diprotodontia
<b>Familie:</b>	Vombatidae
<b>Gattung:</b>	Vombatus
<b>Art:</b>	Vombatus ursinus

Die 7 angegebenen Attribute können wie die Pfade in einem Dateisystem interpretiert werden: aus dem Root-Verzeichnis „/“ zweigen mehrere Unterverzeichnisse (Tierreich „animalia“, Pflanzenreich ...) ab. Innerhalb dieses Unterverzeichnisses „animalia“ stellen die Wirbeltiere „chordata“ einen von 30 Stämmen dar, dem wiederum die Klasse der „mammalia“ untergeordnet ist. Am Ende dieses Pfades durch die Hierarchie stehen die verschiedenen Arten von Wombats als Blätter des Knotens „Gattung Vombatus“.

In der Notation eines Filesystems wäre der o.g. Wombat also wie folgt beschrieben:

**`/animalia/chordata/mammalia/diprotodontia/vombatidae/vombatus/vombatus ursinus`**

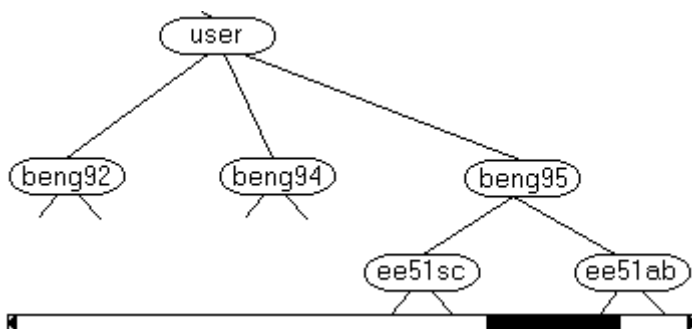
An sich wäre es denkbar, diese wissenschaftliche Hierarchie zu nutzen, um damit eine Multimedia-Datenbank mit allen bekannten Organismen zu erstellen. Man würde jedoch schnell feststellen, dass die klassischen Ansätze zum Umgang mit hierarchischen Daten angesichts der Ausmaße schnell scheitern würden. Allein der Teilbaum „animalia“, der nur einer von sechs von der Wurzel ausgehenden Zweigen ist, würde nach heutigem Stand der Wissenschaft 1,6 Millionen Blätter (Arten) umfassen, die wiederum nur schätzungsweise 10% der heute tatsächlich vorhandenen Arten darstellen. Bei 16 Millionen Blättern wird die Notwendigkeit für neue Ansätze in der Speicherung aber auch in der Darstellung von Hierarchien überdeutlich.

### 3 Probleme bei der Darstellung hierarchischer Daten

Die erste Darstellung hierarchischer Daten erfolgte bereits auf den Nur-Text-Terminals der Vergangenheit. Im Kommandozeilenbetrieb war es nötig innerhalb des Dateisystems (also einer hierarchischen Struktur wie in 2.1) zu navigieren und die Inhalte von Verzeichnissen (die Kinder eines Knotens) darzustellen. Spätestens mit der Einführung grafischer Benutzeroberflächen wurde die Visualisierung des Dateisystems zu einer Kernaufgabe der Oberfläche. Dabei zeigten sich immer wieder ähnlich geartete Probleme.

#### 3.1 Darstellung als Baum

Gerade in Informatik-Lehrbüchern findet man oft die Darstellung von Bäumen entsprechend Abbildung 1. Der Baum wächst von der oben zentral platzierten Wurzel nach unten, wobei die Beziehungen zwischen Eltern und Kindern durch Kanten zwischen den einzelnen Knoten dargestellt werden. Diese Kanten können wie in Abbildung 1 orthogonal ausgerichtet sein, aber auch als Diagonalen verschiedenster Winkel und Abstände gezeichnet werden (Abbildung 3). Diese Form der Darstellung ist intuitiv und schnell erfassbar und entspricht am ehesten der Art und Weise, wie eine Person eine Baumstruktur von Hand darstellen würde. Aus diesem Grund ist sie in der Literatur auch entsprechend populär.



In der Software-Technik findet man diese Form der Darstellung jedoch äußerst selten, was sich anhand einer Vielzahl von praktischen Problemen erklären lässt.

Zunächst ist eine Darstellung dieser Art enorm platzintensiv, da man in der Praxis mit weitaus größeren

Abbildung 3 - Darstellung als Baum

Hierarchien arbeitet, als die zur Illustration in Lehrbüchern dargestellten Kleinstbäume. Das Beispiel aus 2.2 zeigt, daß ein Knoten zwischen einem und Hunderttausenden von Kindern haben kann. Auch wenn in der Praxis sicherlich selten solche Dimensionen auftreten, stellen selbst moderate Anzahlen von Kindern wie 10 oder 20 für eine Darstellung als Baum ein Problem dar. Um eine überschneidungsfreie Darstellung aller Knoten, Kanten und Beschriftungen zu erreichen, müssen entweder sehr große Abstände zwischen den Elementen gewählt (geringe Knotendichte) oder es müssen aufwendige Algorithmen (Labeling-Problem) eingesetzt werden, um eine kompakte aber dennoch korrekte Darstellung zu erzielen.

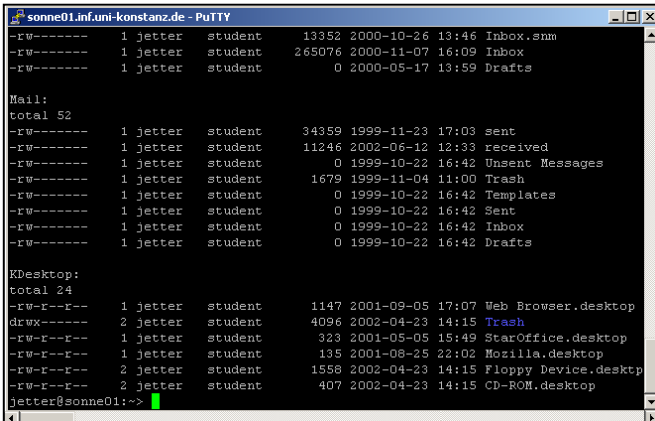
Die geringe Knotendichte führt dazu, dass die großflächige Hierarchie immer nur ausschnittsweise auf dem Bildschirm dargestellt werden kann, was zwangsläufig zu sehr viel Zooming- und Panning-Operationen aufseiten des Benutzers führt. Eine Orientierung in der Hierarchie über mehrere Ebenen hinweg ist dabei schwierig, da der Kontext, in dem der aktuelle Bildausschnitt liegt, nicht mehr nachvollziehbar ist. Es gibt also einen unzureichenden Gesamteindruck von der Topologie des Baumes („Man sieht den Baum vor lauter Zweigen nicht“).

Eine kompakte Darstellung von großer Güte bei höherer Knotendichte kann diesen Nachteil bedingt ausgleichen, ist im Gegenzug aber sehr rechenintensiv und kann die Feedbackzeiten des Systems in die Höhe treiben. Dies verzögert eine direkte Manipulation durch Zooming und Panning und ist damit auf die Dauer uneffizient und für den Benutzer ärgerlich.

### 3.2 Darstellung als Liste

Die Darstellung als Liste ist sicherlich die älteste Form der Visualisierung von hierarchischen Daten und stellt trotz ihrer sehr einfachen Grundidee die Basis für die heute populärsten Methoden der Visualisierung von Hierarchien dar (siehe 3.3).

Bei der Listendarstellung wird jeder Knoten der Hierarchie durch eine Zeile dargestellt, die die Knoteneigenschaften enthält, aber auch z.B. die Position in der Hierarchie über Pfade wiedergeben kann. Ein allgemein bekanntes Beispiel dürfte der UNIX-Befehl „ls“ sein, dessen Ausgabe in Abbildung 4 dargestellt ist.



```

sonne01.inf.uni-konstanz.de - PuTTY
-rw----- 1 jetter student 13352 2000-10-26 13:46 Inbox.smm
-rw----- 1 jetter student 265076 2000-11-07 16:09 Inbox
-rw----- 1 jetter student 0 2000-05-17 13:59 Drafts

Mail:
total 52
-rw----- 1 jetter student 34359 1999-11-23 17:03 sent
-rw----- 1 jetter student 11246 2002-06-12 12:33 received
-rw----- 1 jetter student 0 1999-10-22 16:42 Unsent Messages
-rw----- 1 jetter student 1679 1999-11-04 11:00 Trash
-rw----- 1 jetter student 0 1999-10-22 16:42 Templates
-rw----- 1 jetter student 0 1999-10-22 16:42 Sent
-rw----- 1 jetter student 0 1999-10-22 16:42 Inbox
-rw----- 1 jetter student 0 1999-10-22 16:42 Drafts

RDesktop:
total 24
-rw-r--r-- 1 jetter student 1147 2001-09-05 17:07 Web Browser.desktop
drwx----- 2 jetter student 4096 2002-04-23 14:15 Trash
-rw-r--r-- 1 jetter student 323 2001-05-05 15:49 StarOffice.desktop
-rw-r--r-- 1 jetter student 135 2001-08-25 22:02 Mozilla.desktop
-rw-r--r-- 2 jetter student 1558 2002-04-23 14:15 Floppy Device.desktop
-rw-r--r-- 2 jetter student 407 2002-04-23 14:15 CD-ROM.desktop
jetter@sonne01:~>

```

Abbildung 4 – Listendarstellung mit "ls"

Dabei offeriert „ls“ eine Unzahl von Kommandozeilenparametern, um die Ausgabe zu beeinflussen. Beispielsweise kann eine Auswahl der dargestellten Attribute, des Ausschnitts aus der Hierarchie, des Farbhightlightings und viele anderer Eigenschaften erfolgen. „ls“ hat damit

eher die Natur einer Datenbankanfrage im Dialogbetrieb: wer mit „ls“ arbeitet, muß die Syntax des Dateisystems und der Kommandozeile erlernen.

Dem schlechten Ease-Of-Learning steht bei „ls“ dabei ein akzeptabler Ease-Of-Use gegenüber: Erfahrene Benutzer können damit schnell und gezielt Inhalte aus dem Dateisystem wiedergeben und eine Hierarchie explorieren und darin navigieren. Ein großes Manko ist jedoch, dass die Topologie des Baumes und der Kontext in dem die dargestellten Knoten stehen an keiner Stelle wirklich „visualisiert“ werden, sondern dass die Interpretation von symbolischen Zeichenketten (Pfadangaben, Dateinamen) erst im Kopf des Benutzers zu einem Modell der Hierarchie zusammengesetzt wird.

Eine Darstellung der Topologie des Baumes ist zwar rudimentär durch Einrückung und durch die Länge der Pfade vor den Knoten darstellbar, jedoch ist diese nicht dazu geeignet, sich über mehrere Hierarchieebenen hinweg zu orientieren oder Aussagen über die Gestalt des Baumes zu treffen. Auch bei der Listendarstellung sind oft Panning-Operationen im Terminalfenster notwendig, um sich bei einer großen Anzahl von Knoten auf die Einträge beziehen zu können, die sich bereits außerhalb der aktuellen Listendarstellung befinden.

### 3.3 Darstellung als Treebrowser

Als die heute populärste Form der Darstellung von Hierarchien hat sich eine Synthese aus den Ansätzen in 3.1 und 3.2 durchgesetzt.

Treebrowser kombinieren eine listenartige Darstellung des Knoteninhalts mit einer kompakten grafischen Darstellung der Baumtopologie in einem 2-teiligen Fenster. Bekannt ist dieses Prinzip z.B. aus dem Windows-Explorer, bei dem auf der linken Seite in der Treeview die Baumtopologie eingeblendet wird, mit der das gezielte Navigieren in der Hierarchie durch Anklicken von Verzeichnissen möglich ist. Deren Inhalt wird dann im rechten Teil des Fensters in einer Listendarstellung angezeigt.

Die kompakte Darstellung der Topologie in der Treeview wird durch drei besondere Eigenschaften möglich: Erstens werden in der Treeview nur Verzeichnisse eingeblendet, also Knoten mit Kindern, aber keine Blätter. Zweitens werden die Verzeichnisse zeilenweise eingeblendet, so dass der platzsparende Effekt einer Liste ausgenutzt wird. Drittens wird die Topologie des Baumes in der Liste durch orthogonale Kanten und Einrückung dargestellt, die einen visuellen Eindruck vermitteln und mit denen der Benutzer interagieren kann: ist ein Zweig in der Baumstruktur von Interesse, kann er durch Anklicken „erweitert“ oder „reduziert“ werden, d.h. alle seine Kinder werden ein- oder ausgeblendet. So kann der Detailgrad vom Benutzer gezielt gesteuert werden. Weiterhin können beide Teile des Fensters getrennt voneinander mit Panning-Operationen auf den relevanten Ausschnitt fokussiert werden.

Der Treebrowser erlaubt somit eine sehr gute Navigation in der Breite des Baums, d.h. er ist sehr anschaulich, wenn es darum gilt Dateien oder Verzeichnisse auf derselben Hierarchie-Ebene auszumachen und beispielsweise seriell abzuarbeiten oder abzusuchen. Die Orientierung in der gesamten Hierarchie (speziell über mehrere Hierarchieebenen hinweg oder bei sehr tiefen Hierarchien) verbleibt trotz der Treeview schwierig. Bei einer Vielzahl von vertikalen Kanten oder Einrückungen (gerade bei tiefen Hierarchien ist dies häufig der Fall) entartet die Treeview-Darstellung. Der Baumcharakter ist nicht mehr auf einen Blick zu erfassen, sondern es ist nötig, sich durch Verfolgung des Kantenverlaufs oder durch Betrachten der horizontalen Einrückungen die Baumtopologie zu erarbeiten.

Auch das Ein-/Ausklappen der interessanten Zweige ist ein durchaus abstrakter Vorgang und damit eine höhere kognitive Belastung, als es eine simplere direkte Manipulation (wie z.B. Zooming oder Panning mithilfe von Slidern) wäre. Die Entscheidung muss klar für oder gegen einen Zweig getroffen werden („ganz oder gar nicht“), da nur eine teilweise Verschiebung zum interessierenden Ausschnitt nicht möglich ist.

## 4 Der hyperbolische Browser

Die Mängel, die unter 3. deutlich wurden, haben viele Arbeiten zu neuen Techniken der Visualisierung von hierarchischen Daten angeregt. Eine Vielzahl von Arbeiten hat sich dabei insbesondere mit der Problematik des begrenzten Raums zur Darstellung beschäftigt. Viele Ansätze verfolgten dabei eine dreidimensionale Darstellung [Robertson et al.], andere versuchten die Informationsdichte in der 2d-Ebene durch alternative Darstellungen zu erhöhen [Shneiderman 92].

Grundsätzlich stellte sich die Frage, wie es möglich ist, sowohl den Gesamtkontext, in dem ein Knoten steht, als auch seine unmittelbaren Nachbarn und Details gleichzeitig darstellen zu können. Im Jahre 1995 wurde in einer Veröffentlichung des Xerox PARC von Lamping et al. ein neuentwickelter Ansatz vorgestellt, der für großes Interesse sorgte [Lamping et al.]. Die Arbeit stellt bis heute, die umfassendste Darstellung des „hyperbolischen Browsers“ dar und ist seit dem die Grundlage für jegliche weitere Arbeiten an zweidimensionalen hyperbolischen Visualisierungen.

### 4.1 Inspiration für den hyperbolischen Browser

Als Inspiration für den hyperbolischen Browser wird von Lamping et al. das in Abbildung 5 gezeigte Bild von M.C. Escher genannt. In der Tat sind in diesem Bild drei wesentliche Aspekte des hyperbolischen Browser erkennbar:

1. Während in der Mitte des Bildes die einzelnen Figuren sehr detailliert dargestellt sind, nimmt der Detailgrad nach außen deutlich ab. Die Anzahl der dargestellten Figuren nimmt hingegen mit der wachsenden Dichte nach außen hin stark zu.

Der detaillierte Bereich im Zentrum des Bildes lässt sich daher als der fokussierte Ausschnitt des Interesses verstehen (viele Details, wenig Objekte), während nach außen hin zugunsten der umfassenden Darstellung des Kontexts, der Detailgrad immer weiter reduziert wird (wenig Details, viele Objekte). Dieser Fischaugeneffekt stellt also eine Form einer Focus+Context Technik dar.



Abbildung 5 - Inspiration durch M.C. Escher

Quelle: Circle Limit IV (Heaven and Hell), 1960, (c) 1994 M.C. Escher  
Cordon Art -- Baarn – Holland

2. Betrachtet man das Zentrum, so lassen sich von dort aus 3 Hauptachsen ausmachen, entlang denen sich die Darstellung nach außen hin ausbreitet. Zieht man diese Eigenschaft als Anregung für eine Baumdarstellung heran, ist diese Form deutlich platzsparender, als die in 3.1 gewählte Variante (Abbildung 6).
3. Aus 2. ergibt sich zwangsläufig eine kreisförmige Ausbreitung der Darstellungsfläche. Die Kreisfläche, die zur Verfügung steht, wächst dabei quadratisch mit dem Radius ( $A = \pi r^2$ ), während eine rechteckige Darstellung mit der konstanten Breite  $b$  wie in 3.1 nur linear mit einer variablen Höhe  $h$  wächst ( $A = b h$ ). Somit ergibt sich aus dem Wachstumsverhalten der Darstellungsflächen ein weiterer Vorteil für Abbildung 6.

Leider ist diese Darstellung mit konventioneller euklidischer Geometrie in einem kartesischen Koordinatensystem nur schwer implementierbar. Die Darstellung würde eine Vielzahl von Transformationen erfordern: Rotationen zur Ausrichtung an den Achsen, Translationen und Skalierungen um den Fischaugeneffekt umzusetzen. Weiterhin ist ein quadratisches Wachstum der Fläche zwar ein Vorteil gegenüber linearem Wachstum, doch typischerweise wachsen hierarchische Daten schneller, so dass es erneut nötig ist, wegen Platzmangel, Überschneidungen mit aufwendigen Algorithmen zu vermeiden.

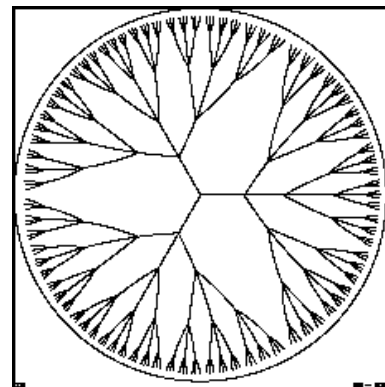


Abbildung 6 – Baum nach 4.1

Quelle: [Lamping et al]

Ein weiterer Nachteil ist, dass bei der Interaktion mit der Darstellung (z.B. ein Verschieben des Fokus auf andere Knoten) das Layout des Baumes komplett neu berechnet werden muss, was die Möglichkeit der direkten Manipulation stark verzögern würde.

Zusammengefasst ist eine derartige Form der Darstellung in der euklidischen Ebene aufwendig und in der Implementierung und auch zur Laufzeit rechenintensiv. Abhilfe schafft hier das Verlassen der euklidischen Ebene und die Umsetzung der Darstellung in der hyperbolischen Ebene.

## 4.2 Darstellung in der hyperbolischen Ebene

Die hyperbolische Ebene ist ein mathematisches Modell, dessen Eigenschaften stark konträr zu unserer alltäglichen Vorstellung von der Geometrie unserer Welt sind und das daher auch nicht einfach visualisierbar ist. Der Versuch der Darstellung der hyperbolischen Ebene scheitert beispielsweise schon an ihrer Eigenschaft, dass Parallelen darin nicht immer denselben Abstand zueinander besitzen. Dies widerspricht unserer Vorstellung der Natur von Parallelen erheblich.

Andere Eigenschaften kommen der Darstellung von Bäumen in der hyperbolischen Ebene jedoch sehr zu Gute, insbesondere die Eigenschaft, dass in der hyperbolischen Ebene der Kreisumfang exponentiell mit dem Radius wächst. Diese rasante Zunahme an Platz führt dazu, dass Hierarchien langsamer wachsen, als der dazu zur Verfügung stehende Raum. Das heisst, dass die Kinder eines Knotens in der Ebene immer nur einen gleichen oder kleineren Winkelabschnitt in Anspruch nehmen, als der Knoten an sich. Das Erstellen eines Layouts eines Baumes in der hyperbolischen Ebene reduziert sich damit zu einem trivialen Problem, das ein einfacher rekursiver Algorithmus lösen kann. Beim Generieren des Layouts sind dank des Überangebots an Raum keine komplexen mathematischen Operationen wie in 4.1 beschrieben notwendig.

Das so entstehende Layout der Hierarchie kann als Ergebnis einer Data Transformation nach Card et al. gesehen werden und stellt somit die Data Tables dar, die als Ausgangspunkt für das Visual Mapping dienen. Dieses Layout in der hyperbolischen Ebene ist aber als solches immer noch genauso wenig visualisierbar, wie die hyperbolische Ebene an sich. Es lässt sich aber über ein Projektionsverfahren in die Visual Structures verwandeln, die schließlich am Ende der Pipeline nutzbare Views liefern.

## 4.3 Projektion in die euklidische Ebene

Um die Data Tables (also das Layout) in darstellbare visuelle Strukturen umzuwandeln wird im hyperbolischen Browser nach Lamping et al. das Poincaré Model (oder auch Conformal Mapping) verwendet. Dieses mathematische Verfahren erlaubt eine Visualisierung der hyperbolischen Ebene durch Projektion

in den zweidimensionalen Einheitskreis. Das Verfahren und seine Vor- und Nachteile werden detailliert von Lamping et al. beschrieben, an dieser Stelle seien nur die wichtigsten Aspekte genannt:

1. Das Projektionsverfahren umfasst alle Aspekte, die in 4.1 als Inspiration für die Darstellung von Bäumen, genannt wurden. Das Ergebnis der Projektion des hyperbolischen Layouts ist damit eine zweidimensionale Darstellung, die sich radial vom Zentrum aus ausbreitet und den Fischaugeneffekt als Focus+Context Technik umsetzt (siehe Abbildung 6).
2. Die Eigenschaften in 1. ergeben sich aus der Natur des Projektionsverfahrens und müssen nicht durch besondere weitere Verarbeitung des Projektionsergebnisses berechnet werden.
3. Das Projektionsverfahren ist leichter implementierbar und schneller als eine vergleichbare Darstellung mit euklidischer Geometrie. Pro Datenpunkt sind lediglich 20 Floating-Point-Operationen notwendig.
4. Das Verschieben des Projektions-Fokus innerhalb des hyperbolischen Layouts erfolgt durch einfachste Transformationen in der hyperbolischen Ebene. Der relevante Teil des Layouts wird durch einfaches Verschieben und simples Clipping in das Projektionszentrum gerückt. Somit ist das Layout unabhängig vom gewählten Fokusbereich und muss nie neu berechnet werden.

Als Ergebnis des Conformal Mappings entsteht eine kreisförmige zweidimensionale Visualisierung. In der Visualization Pipeline nach Card et al. stellt dieses Ergebnis die Visual Structures dar, der Prozess der Poincaré-Projektion entspricht dem Visual Mapping.

#### **4.4 Rendering**

Prinzipiell sind die durch die Projektion entstandenen Visual Structures bereits die vollständigen Darstellungen der Hierarchie und damit in der Visualization Pipeline von den Views nicht zu trennen. Eine Manipulation der View Transformations findet lediglich bei der Veränderung der Größe und Form der zur Darstellung verwendeten Kreis- oder Ellipsenfläche statt.

Die Hauptinteraktion zwischen Benutzer und Darstellung findet also nicht in der Form von veränderten Parametern bei der View Transformation statt, sondern beim Visual Mapping in 4.3. Dort nehmen wahlfreie Parameter wie die Position des Fokus oder die Orientierung des Baumes Einfluss (siehe 5.1).

### **5 Interaktion mit dem hyperbolischen Browser**

Eine ausführliche Beschreibung der Interaktionsmöglichkeiten mit dem Browser würde den Rahmen dieser Arbeit sprengen. Es ist jedoch anzumerken, dass die charakteristischen Eigenschaften der Benut-

zung, besonders der Vor- und Nachteile, im Probetrieb schnell deutlich werden und in [Lamping et al.] anschaulich illustriert werden. An dieser Stelle sei deswegen auch auf die Website der INXIGHT Software, Inc. verwiesen, auf der verschiedenste Varianten von hyperbolischen Browsern als Demonstratoren unter dem Produktnamen StarTree zugänglich sind (<http://www.inxight.com>).

Im folgenden sollen dennoch die Interaktionsmöglichkeiten anhand zweier Referenzmodelle dargestellt werden, um eine Einordnung des hyperbolischen Browsers zu ermöglichen. Zum einen mit der bereits erwähnten Visualization Pipeline [Card et al.], zum anderen mit der Task by Data Type Taxonomy von Shneiderman [Shneiderman 96].

## 5.1 Visualization Pipeline

Im folgenden findet sich eine komplette Darstellung der Visualization Pipeline des hyperbolischen Browsers, die die bisherigen Aussagen zusammenfasst und ergänzt:

**Raw Data** (= Repräsentation der hierarchischen Daten im Speicher)



**Data Transformation:** Erstellen des Layouts in der hyperbolischen Ebene  
User-Interaktion: keine

**Data Tables** (= Layout des Baumes in der hyperbolischen Ebene)



**Visual Mapping:** Translation, Projektion des Layouts auf den 2D-Einheitskreis  
User-Interaktion: setze Fokus innerhalb Layout durch Mausbewegung, Orientierung des Baumes

**Visual Structures** (= Projektion des Layouts auf den 2D-Einheitskreis)



**View Transformation:** Rendering des 2D-Einheitskreis in den Darstellungsbereich  
User-Interaktion: wähle Form/Position Darstellungsbereich durch Window Drag/Resize

**Views** (= fertig gerenderter Darstellungsbereich)

## 5.2 Task by Data Type Taxonomy

In der Task by Data Type Taxonomy von Shneiderman werden die an einem visuellen Suchsystem zu verrichtenden Arbeiten auf 6 elementare Tasks (Overview/Zoom, Details-on-Demand, Filter, Relate,

History, Extract) reduziert, mit deren Hilfe nun hier die bisherigen öffentlich zugänglichen und künftig denkbaren Implementierungen des hyperbolischen Browsers charakterisiert werden sollen.

### **5.2.1 Overview/Zoom**

Durch die hyperbolische Projektion ergibt sich im Zentrum ein Fokusbereich, der eine detaillierte Darstellung ähnlich wie eine Zoom Funktion bietet. Die stetige Abnahme des Detailgrades mit wachsendem Abstand zum Zentrum zur Anzeige des umgebenden Kontexts kann wiederum als Overview betrachtet werden. Dieser Fisheye-Effekt führt also zu einer Integration von Overview/Zoom in eine konstante Darstellung, ohne daß durch Zooming oder Panning ein Wechsel in den Modalitäten erfolgen muß.

Das Zooming erfolgt dabei durch die Auswahl des Fokus-Bereichs aus der Gesamthierarchie durch direkte Manipulation mithilfe von Maus Dragging oder durch Anklicken eines Knotens, der daraufhin in das Zentrum der Darstellung gerückt wird. Die animierten Übergänge bei der Manipulation erlauben dem Benutzer dabei alle ausgelösten Veränderungen zu erfassen und unterstützen damit den Overview-Effekt.

### **5.2.2 Details-on-demand**

Den einzelnen Knoten in der Hierarchie können beliebige Attribute zugewiesen werden, die dann durch Selektion (entweder durch Doppelklick oder die Bewegung in den Fokus-Bereich) oder durch einen Rollover-Effekt abgerufen werden können. Zur Darstellung sind Tooltips oder am Browserrand eingeblendete Objekte üblich. Mithilfe der Mehrfach-Selektion (beispielsweise durch die Kombination von SHIFT mit dem Mausklick) könnten auch Detailinformationen über Knotengruppen oder Knotenfamilien angeboten werden.

Die bisherigen Versionen des INXIGHT StarTrees erlauben das Unterlegen von Knoten mit Hyperlinks oder Grafiken. Für die direkte Darstellung eines quantitativen Attributs im Knoten werden Balken oder Kreisdiagramme angeboten. Somit können weitere also Detailinformationen direkt oder über Links in die Darstellung integriert werden.

### **5.2.3 Filter**

In den bisherigen Implementierung wurde eine Filter-Funktion nur durch Ein-/Ausklappen von Zweigen realisiert. Gerade Filter bergen jedoch ein großes Potential und könnten in Zukunft umfangreiche Möglichkeiten bieten. In 8.2 wird näher auf diese Thematik eingegangen.

### **5.2.4 Relate**

Es liegt in der Natur von hierarchischen Daten, dass Beziehungen untereinander durch die hierarchische Struktur sichtbar werden. Eltern-Kind-Beziehungen und damit die semantische Strukturierung gehen also

selbst aus der rudimentärsten Darstellung hervor. Das Darstellen anderer Querbeziehungen (z.B. über gleiche oder ähnliche Knotenattribute) sind jedoch für die Zukunft ebenfalls denkbar. Durch Einfärbung von Knotenfamilien oder Pfaden im Baum könnten nicht nur wie bisher einzelne Zweige optisch voneinander abgetrennt, sondern es könnten umfangreiche Möglichkeiten zur Analyse der Hierarchie im Hinblick auf die individuellen Eigenschaften der Knoten angeboten werden. An dieser Stelle sei nochmals auf den Abschnitt 8.2 verwiesen, der sich mit möglichen Erweiterung dieser Art beschäftigt.

Bereits realisiert ist eine simple Suchfunktion, die die Knotennamen als Suchbasis heranzieht und die Knoten entsprechend dem Auftreten eines Suchbegriffs im Namen einfärbt. Diese Konzeption ist genauso für andere Attribute der Knoten denkbar, bis hin zur Möglichkeit, die Relevanzen eines Knotens oder eines gesamten Zweiges zu einer Eigenschaft durch zunehmende Farbintensität der Kanten zu visualisieren.

Die Möglichkeiten der Analyse von Querbeziehungen sind bisher also noch kaum entwickelt, aber vielversprechend.

### **5.2.5 History**

Eine History-Funktion ist für den hyperbolischen Browser eine triviale Speicherung vergangener Fokus-Bereiche und bereits in vielen Implementierungen integriert.

### **5.2.6 Extract**

In den bekannten Implementierungen ist eine derartige Funktion bisher nicht integriert, aber durchaus denkbar. Durch Multiple-Selection von Knoten oder die Selektion eines Zweiges kann eine Knotenfamilie selektiert und beispielsweise mithilfe eines Kontext-Menüs über „Copy“ in das Clipboard übernommen werden. Mit einer Paste-Operation wäre dann ein Import in andere Applikationen möglich. Dies könnte durch grafisches Rendering eines Baumdiagramms im Falle von grafischen Anwendungen erfolgen. Im Falle von Tabellenkalkulation oder Textverarbeitung wäre ein Export als CSV- oder Textdatei denkbar. Diese Formate werden vom INXIGHT StarTree Studio selber verwendet, um hierarchische Daten zu importieren (siehe 6.1).

## 6 Der hyperbolische Browser in der Praxis

### 6.1 Produkte mit hyperbolischen Browsern

Dem hyperbolischen Browser wurde schon in einem frühen Stadium der Entwicklung ein großes kommerzielles Potential bescheinigt, weshalb diese Visualisierungstechnik von Xerox patentrechtlich geschützt wurde. Diese Patente sind an den Xerox Spin-Off INXIGHT Software, Inc. übergegangen, was dazu führt, dass INXIGHT bislang der einzige Anbieter von kommerziellen hyperbolischen Browsern ist. Unter der Produktbezeichnung StarTree<sup>TM</sup> wurden von INXIGHT verschiedenste Applikationen und Entwicklungswerkzeuge realisiert und zu einer umfangreichen Produktfamilie weiterentwickelt:

- Für eine Verwendung des hyperbolischen Browsers in eigenen Entwicklungen werden mit dem **StarTree SDK** fertige Funktionsbibliotheken und Klassen als API für Java und C++ vertrieben.
- Mit dem **StarTree Studio** wird versucht, ein grafisches Entwicklungswerkzeug und einen Dateistandard (STZ-Dateien) für hyperbolische Bäume zu etablieren. Diese Dateien werden mit dem StarTree Studio editiert (Abbildung 7), wobei Baumtopologien auch aus Datenfiles importiert und darin enthaltene Knoten individuell gestaltet werden können. So kann beispielsweise eine Hierarchie von Artikeln aus einem Content Management System in das Studio exportiert und mit Hyperlinks und grafischen Illustration hinterlegt werden, um als Sitemap oder Navigation in einem Intranet eingesetzt zu werden.
- Eine Visualisierung solcher STZ-Files erfolgt entweder mithilfe des **StarTree Viewers** oder einem Java-Applet, das es erlaubt, die hyperbolischen Bäume direkt in Websites zu integrieren.
- Im umfassenden Enterprise-Paket INXIGHT **VizServer** wurde der StarTree als eine von vielen Visualisierungsoptionen integriert. Das Paket verspricht Lösungen zum schnellen und einfachen Zugriff und zur kooperativen Arbeit an unternehmensweiten Datenbeständen durch Einsatz fortgeschrittener Visualisierungstechniken.

### 6.2 Einsatz in der Praxis

Mit einer Lizenzgebühr von 5000 US-Dollar pro Web-Server (Stand 2001) zielt das StarTree Studio noch auf den mittleren bis unteren Enterprise Bereich ab, während das VizServer Paket sich dagegen klar an Großunternehmen, Verwaltungen und Behörden richtet (Kunden sind u.a. das US Department of Defense, Mercedes Benz, Intel, Aventis). Für kleine und mittelständische Unternehmen ist VizServer zwar nach unten skalierbar, jedoch immer noch mit erheblichen Investitionen verbunden, weshalb der Einsatz von

vielen dieser Unternehmen als unattraktiv betrachtet wird. Entsprechende Untersuchungen zum Wissensmanagement in kleinen und mittelständischen Unternehmen wurden beispielsweise im EU-Projekt KINX (Knowledge Integration and Network eXpertise) an der RWTH Aachen durchgeführt.

Die Nutzung des Visual Studios zur Generierung von Sitemaps oder Produktverzeichnissen für E-Commerce Anbieter ist dagegen eine populärere und häufig genannte Anwendung, die durch viele Beispiele auf der INXIGHT Website reichlich illustriert wird (BestBuy.com, Porsche, siehe Abbildung 8).

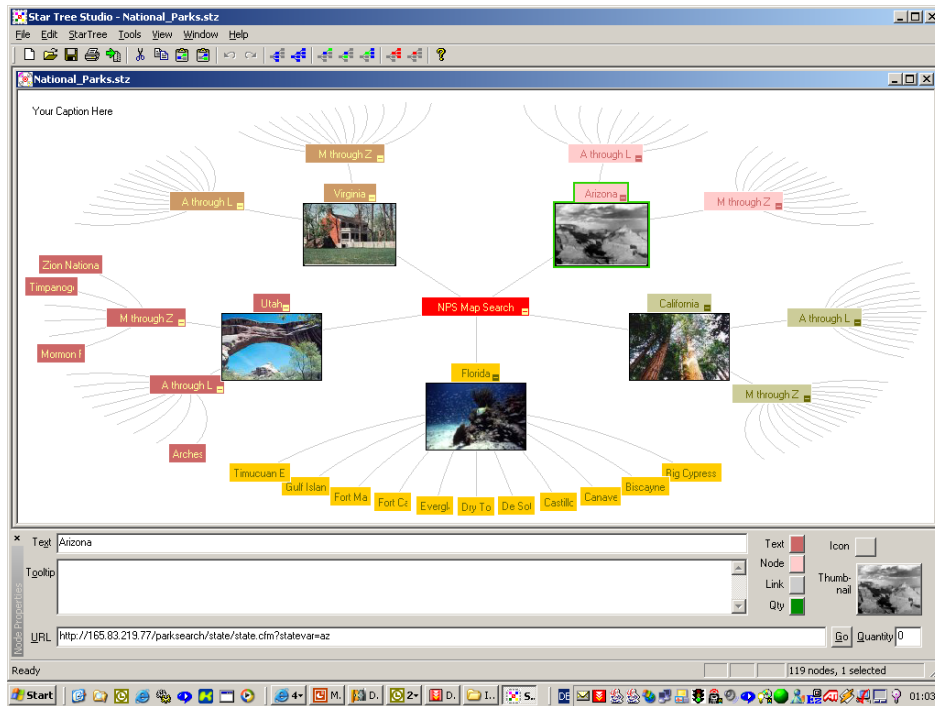


Abbildung 7 - INXIGHT StarTree Studio

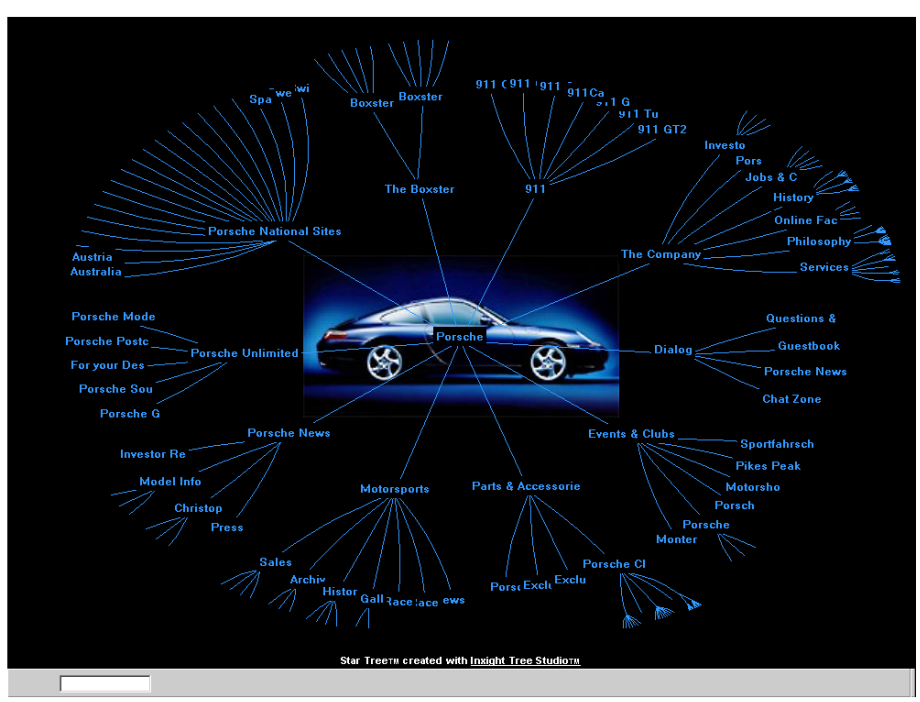


Abbildung 8 - Der StarTree als Sitemap auf der Porsche-Website

## 7 Usability-Aspekte des hyperbolischen Browsers

Bei einer neuen und unkonventionellen Form der Visualisierung wie dem hyperbolischen Browser sind frühzeitige Usability-Tests sinnvoll, um Aussagen darüber treffen zu können, ob eine weitere Entwicklung sinnvoll erscheint und ob die propagierten Vorteile auch in der Praxis beobachtbar sind. Schon 1995 wurde daher von Lamping et al. eine kleine Untersuchung mit 4 Testpersonen über die Gebrauchstauglichkeit des hyperbolischen Browsers durchgeführt. Dabei wurde ein konventioneller Treebrowser mit Scrollbars einer frühen Form des hyperbolischen Browsers gegenüber gestellt. In beiden Fällen war es die Aufgabe, einen Knoten gezielt in einer Hierarchie aufzufinden, die aus einer Website entnommen war.

Das Ergebnis zeigte keine signifikante Verbesserung der Such- und Zugriffszeiten zwischen beiden Browser-Konzepten. Lediglich in der Posttest-Befragung äußerten sich die Testpersonen überwiegend positiv gegenüber dem hyperbolischen Browser, der als ansprechend empfunden wurde [Lamping et al.].

Eine ähnliche Vergleichsstudie wurde 2 Jahre später von Czerwinski und Larson im Auftrag von Microsoft Research durchgeführt und auf der HCI 97 Konferenz in Bristol vorgestellt. Diese Untersuchung wurde dabei mit 17 Testpersonen größer angelegt und quantitativ umfangreicher analysiert [Czerwinski, Larson]. Auch hier zeigte das Ergebnis keine signifikante Verbesserung der Effizienz der Arbeit. In der Posttest-Befragung wurde sogar eine leichte Abneigung gegenüber dem hyperbolischen Browser deutlich. Dennoch treffen Czerwinski und Larson die Aussage „Hyperbolic browser better for keeping global/local focus, seeing category size and related topics“, was als Bestätigung der Keyfeatures des hyperbolischen Browsers betrachtet werden kann.

Auf der Website von INXIGHT Software, Inc. wird dem StarTree im Vergleich zur Windows Tree Control eine 62%ige Verbesserung der Effizienz bei der Navigation bescheinigt. Leider sind auf der Website keine weiteren Angaben über die zugrundeliegende Studie von Xerox zu finden, weshalb sich der Verdacht einer eher Marketing-orientierten Werbeaussage aufdrängt.

Dennoch sind diese drei an sich widersprüchlichen Ergebnisse vor folgendem Hintergrund bedingt erklärbar: Die erste und zweite Studie fanden 1995 bzw. 1997 statt, als der hyperbolische Browser noch in einem frühen Entwicklungsstadium war. Seit dem hat sich die technische Umsetzung und die visuelle Gestaltung vom prototypischen hyperbolischen Browser zum heutigen INXIGHT StarTree entwickelt, bei dem Farbgestaltung, Typografie, Geschwindigkeit und Stabilität gegenüber den Vorgängern erheblich verbessert wurden. Damit kann auch die Effizienz der Arbeit deutlich verbessert worden sein.

Einen wichtigen Aspekt stellt auch die Hierarchie, auf der die Untersuchung durchgeführt wurde, dar:

Die semantische Strukturierung und die Breite und Tiefe des Baumes haben wesentlichen Einfluss auf die Effizienz der Arbeit. Es erscheint logisch, dass kleine Hierarchien mit semantisch nicht klar abgetrennten Knoten mit einem konventionellen Treebrowser schneller zugänglich sind, während große Verzeichnisse

mit hoher Tiefe und Spezialisierung (z.B. Patentverzeichnisse, Warenkataloge) eher die Domäne des hyperbolischen Browsers darstellen.

In den zwei zuerst aufgeführten Studien wurde der Aspekt „hyperbolischer Browser versus Treebrowser“ in den Vordergrund gestellt, aber das Augenmerk nur unzureichend auf die zur Untersuchung verwendeten Hierarchien gelenkt.

Es erscheint sinnvoll, in Zukunft keine Untersuchung nur zur Klärung der Frage nach dem überlegenen Browser-Konzept durchzuführen, da diese entscheidend von dem zur Untersuchung verwendeten Datenbestand abhängt und damit je nach gegebener Testkonzeption zu unterschiedlichsten Resultaten führen wird.

Aussagekräftiger könnte die Entwicklung von Modellen zur Klassifizierung von Hierarchien sein, die sowohl die Breite und Tiefe, als auch die semantische Strukturierung berücksichtigen. Auf der Basis dieser Klassifizierung könnte dann die Effizienz der Arbeit mit verschiedenen Browsern für die verschiedenen Klassen von hierarchischen Daten ermittelt werden. Mit der Ermittlung der Effizienz in Abhängigkeit von Datenbestand und Visualisierung in einer zukünftigen Studie würde eine handhabbare Entscheidungshilfe für die Praxis geschaffen und der Wissenschaft eine erste umfassende Darstellung der Vor- und Nachteile des hyperbolischen Browsers zur Verfügung gestellt.

## 8 Ausblick

### 8.1 Der bifokale hyperbolische Browser

Eines der Keyfeatures des hyperbolischen Browsers ist die gleichzeitige Darstellung eines detaillierten Fokus-Bereichs bei gleichzeitiger Bewahrung des Kontextes, in dem der Fokus steht. Eine weitere Verbesserung dieser Eigenschaft erhoffen sich Cava et al. durch die Einführung eines zweiten Fokus-Bereichs und der Aufteilung des Browser-Bereichs in zwei Sektionen. Ihr Konzept des „bifocal trees“ stellten sie im Jahr 2002 auf der IHC 2002 in Fortaleza, Brasilien vor [Cava et al.].

Das Konzept umfasst eine an den hyperbolischen Browser angelehnte Darstellung des Kontexts in der linken Hälfte des Browser-Bereichs, der mit einer detaillierten Darstellung im rechten Bereich gekoppelt ist (Abbildung 9). Der linke Bereich wird dabei als „context focus“, der rechte Bereich als „detail focus“ bezeichnet. Wird im linken Bereich aus der Gesamthierarchie ein neuer Fokus gewählt, so wird gleichzeitig im rechten Bereich eine detaillierte Darstellung des Bereichs und seiner nahen Verwandtschaft eingeblendet. Über den derzeitigen Stand der Implementierung oder Demonstratoren ist zurzeit noch nichts bekannt.

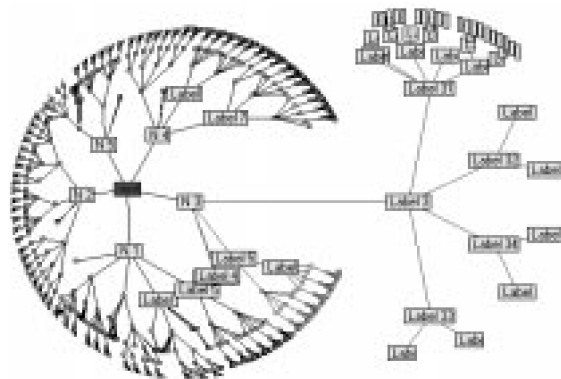


Abbildung 9 - bifocal tree mit context focus (links) und detail focus (rechts)

### 8.2 Integration von Filtern und Suchfunktionen

Betrachtet man die Einordnung in die Taxonomie von Shneiderman (siehe 5.2), so wird deutlich, dass die Integration von Filtern und Suchfunktionen die Aufwertung des hyperbolischen Browser von einer Visualisierung zu einem vollständigen visuellen Suchsystem bedeuten würde.

Wie diese Integration aussehen könnte, kann einer Veröffentlichung von Kumar et al. entnommen werden. Dort werden verschiedene Konzepte zur Integration von Dynamic Queries und Pruning (also von Suchfunktionen und der „Beschneidung“ des Baumes) in ihre Form der Visualisierung von hierarchischen Daten vorgestellt. Kumar et al. arbeiten dabei mit einem Dynamic Query Panel, das die Reduktion und das Highlighting von Knoten in einer darunterliegenden Baumvisualisierung ermöglichen [Kumar et al.].

Die vorgestellten Konzepte lassen sich aufgrund der ähnlichen Aufgabenstellung leicht auf den hyperbolischen Browser übertragen und könnten die Arbeit mit großen Hierarchien erheblich unterstützen.

Vorstellbar wäre beispielsweise die Reduktion eines Produktkataloges durch die visuelle Eingrenzung von Preis oder Lieferbarkeit durch Slider am Browserrand oder eine Filterung auf Produkte einer speziellen Sprache durch Dropdown-Boxen. Die so reduzierte Hierarchie wäre durch ihren geringeren Umfang wesentlich einfacher zu handhaben und zu explorieren. Derartige Funktionen wären dem bisher nur rudimentäre vorhandenen Such- und Highlighting Funktionen (siehe 5.2) weit überlegen.

## 9 Fazit

Es kann sicherlich nicht davon ausgegangen werden, dass der hyperbolische Browser ein in allen Fällen überlegenes Konzept für die Visualisierung von hierarchischen Daten ist. So viele Aspekte an der hyperbolischen Darstellung von Vorteil erscheinen, so viele sprechen auf für die Beibehaltung traditioneller Konzepte wie dem Treebrowser (Erwartungskonformität, Platzbedarf, Patent-Situation etc).

Dennoch sollte der hyperbolische Browser keinesfalls als futuristische Spielerei unterbewertet werden, da er gerade durch die in 8. dargestellten Weiterentwicklungen zu einem überaus mächtigen Werkzeug zur Arbeit auf hierarchischen Daten avancieren könnte. Gerade im E-Commerce oder im Wissens- oder Dokumentenmanagement kann INXIGHTs StarTree eine Schlüsselrolle einnehmen.

Es ist jedoch unverzichtbar, dass in Zukunft der Einfluss der Baumtopologie und der semantischen Strukturierung der hierarchischen Daten auf die Effizienz des hyperbolischen Browser stärker berücksichtigt wird. Eine Studie, wie sie in 7. umrissen wurde, wäre nicht nur für die wissenschaftliche Einordnung, sondern gerade auch für Softwareentwickler und IT-Einkäufer eine wichtige Entscheidungshilfe. Die Durchführung müsste daher auch im Interesse von INXIGHT Software, Inc. liegen, die als Patentbesitzer diese neue Technologie aufgrund der vorhandenen Skepsis und der Investitionshürden möglichst klar und umfassend nach außen darstellen sollten.

## 10 Literaturverzeichnis

- [Card et al.]: Card S.; Mackinlay J.; Shneidermann B. (Hrsg.) (1998): Readings in Information Visualization, San Francisco, 1998: Morgan Kaufmann. 1-34.
- [Cava et al.]: Cava, R. A.; Luzzardi, P.R.G.; Freitas, C.M.D.S. (2002): The Bifocal Tree: a Technique for the Visualization of Hierarchical Information Structures. In: *IHC 2002 – 5th Workshop on Human factors in computer systems, 2002, Fortaleza. IHC 2002*. Fortaleza: SBC/UFCE.
- [Czerwinski, Larson]: Czerwinski, M.P., and Larson, K. (1997): The new Web browsers: They're cool but are they useful? In: H. Thimbleby, B. O'Connell, and P. Thomas (Eds.), *People and Computers XII: Proceedings of HCI'97*, Springer Verlag, Berlin.
- [Kumar et al.]: Kumar, H. P.; Plaisant, C.; Shneiderman, B. (1996): Browsing hierarchical data with multilevel dynamic queries and pruning. *International Journal of Human-Computer Studies*, 46 (1), 103-124.
- [Lamping et al.]: Lamping, J.; Rao, R.; Pirolli, P. (1995): A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. In: Proceedings of the SIGCHI conference on Human factors in computing systems 1995, S. 401- 408, ACM Press/Addison-Wesley Publishing Co.
- [Robertson et al.]: Robertson, G.G.; Mackinlay, J.; Card S.K. (1991): Cone trees: Animated 3d visualizations of hierarchical information. In *Proceedings of SIGCHI'91*, pages 189–194. ACM
- [Shneiderman 92]: Shneiderman, B. (1992): Tree Visualization with Tree-maps: A 2-D space-filling approach. In: ACM Transactions on Graphics
- [Shneiderman 96]: Shneiderman, B. (1996): The eyes have it: A task by data type taxonomy for information visualizations. In: *Proceedings IEEE Visual Languages*, S. 336 - 343