(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 107209630 B
(45)授权公告日 2020.09.22

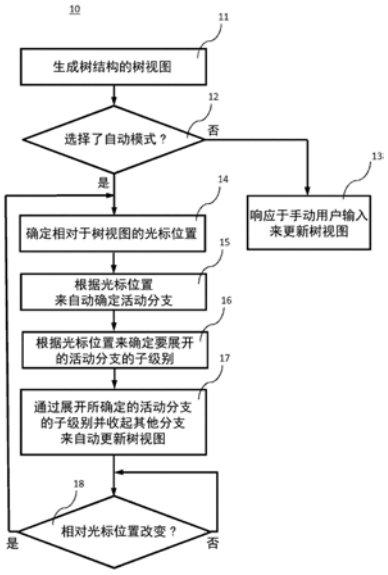
(21)申请号 201580074767.1
(22)申请日 2015.11.27
(65)同一申请的已公布的文献号
申请公布号 CN 107209630 A
(43)申请公布日 2017.09.26
(30)优先权数据
14307100.9 2014.12.19 EP
(85)PCT国际申请进入国家阶段日
2017.07.27
(86)PCT国际申请的申请数据
PCT/EP2015/077876 2015.11.27
(87)PCT国际申请的公布数据
WO2016/096368 EN 2016.06.23
(73)专利权人 交互数字CE专利控股公司
地址 法国巴黎
(72)发明人 延斯·布罗克 弗朗克·克拉斯
斯特凡·库布施 李辉
迈克尔·皮珀 迈克尔·韦伯
(74)专利代理机构 中科专利商标代理有限责任公司 11021
代理人 潘剑颖
(51)Int.Cl.
G06F 3/0482(2013.01)
(56)对比文件
US 8001487 B2,2011.08.16
CN 102117333 A,2011.07.06
JP 特开2004-005527 A,2004.01.08
US 2005/0076312 A1,2005.04.07
US 2009/0125845 A1,2009.05.14
US 6380957 B1,2002.04.30
CN 101978346 A,2011.02.16
CN 102880395 A,2013.01.16
审查员 周辉
权利要求书3页 说明书16页 附图18页

(54)发明名称

用于渲染树结构的数据处理设备和方法

(57)摘要

用于根据指向光标相对于所渲染的树的位置在GUI中渲染树结构的设备(20)和方法(10)使用与树有关的光标位置来控制自动展开/收起哪些树分支。它们在不需要显式展开/收起操作的情况下将树视图重配置用于紧凑呈现。生成(11)树视图。确定(14)相对于视图的包含根据第一和第二方向的第一和第二位置值在内的光标位置。将根据第一方向与光标位置对齐的分支自动选择为活动分支(15)。根据相对于活动分支的子级别的缩进的第二位置值,确定(16)要展开的活动分支的子级别,以及通过展开所确定的活动分支的子级别并收起其他分支(17)来自动更新树视图。



CN 107209630 B

1.一种用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的计算机实现的方法,包括:

-至少生成以多个分支来组织的树结构的树视图;

-确定所述指向光标相对于所述树视图的位置,所述位置包含根据第一方向的第一位置值和根据与所述第一方向正交的第二方向的第二位置值,所述第一位置值确定活动分支被选择为与所述指向光标的根据所述第一方向的位置对齐的分支;

-所述第二位置值根据相对于所述活动分支的子级别的缩进的第二位置值来确定要展开的子级别,使得所述第二位置值的增加值确定树结构中的更深的子级别,并且所述第二位置值的减少值确定树结构中的更浅的子级别;以及

-在所述第一位置值和所述第二位置值中的至少一个改变时,通过展开所确定的所述活动分支的子级别并收起其他分支来更新所述树视图。

2.根据权利要求1所述的计算机实现的方法,其中,在显示所述图形用户界面中的所述树视图的显示装置的屏幕上,所述第一方向对应于垂直方向以及所述第二方向对应于水平方向。

3.根据权利要求1所述的计算机实现的方法,其中,连续重复对所述指向光标的位置的确定、对所述活动分支的确定、对要开展的子级别的确定以及对所述树视图的更新。

4.根据权利要求1所述的计算机实现的方法,其中,如果要展开的子级别的数量小于当前展开的子级别的数量,则通过收起所述活动分支的超出的子级别来更新所述树视图,以及沿所述第一方向移动经更新的树视图,直到隐藏所收起的超出子级别的对应元素变得与所述指向光标的根据所述第一方向的位置对齐为止。

5.根据权利要求1所述的计算机实现的方法,其中,元素的所述树结构对应于概念分类系统。

6.根据权利要求1所述的计算机实现的方法,其中,元素的所述树结构对应于文件的目录结构。

7.根据权利要求1所述的计算机实现的方法,包括:

-响应于由用户输入的模式选择在自动模式和手动模式之间选择;其中

-仅在所述自动模式下执行对所述指向光标的位置的确定、对所述活动分支的自动选择、对要展开的子级别的确定以及对所述树视图的自动更新,以及其中

-在所述手动模式下响应于针对所述图形用户界面的其它手动用户输入来更新所述树视图。

8.根据权利要求1所述的计算机实现的方法,其中,生成所述指向光标以特别用于与所述树视图的用户交互。

9.一种用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备,包括:

-图形表示单元,被配置为生成以多个分支来组织的树结构的树视图;

-位置确定单元,被配置为确定所述指向光标相对于所述树视图的位置,所述位置包含根据第一方向的第一位置值和根据与所述第一方向正交的第二方向的第二位置值,所述第一位置值确定要选择的活动分支;

-活动分支选择单元,被配置为确定活动分支被选择为与所述指向光标的根据所述第

一方向的位置对齐的分支；

-展开确定单元，被配置为根据相对于所述活动分支的子级别的缩进的第二位置值来确定要展开的子级别，使得所述第二位置值的增加值确定所述树结构中的更深的子级别，所述第二位置值的减少值确定所述树结构中的更浅的子级别；以及

-所述图形表示单元，还被配置为在所述第一位置值和所述第二位置值中的至少一个改变时，通过展开所确定的所述活动分支的子级别并收起其他分支来自动更新所述树视图。

10.根据权利要求9所述的数据处理设备，被配置为连续重复以下各项：使用所述位置确定单元对所述指向光标的位置的确定、使用所述活动分支选择单元对所述活动分支的确定、使用所述展开确定单元对要展开的子级别的确定以及使用所述图形表示单元对所述树视图的自动更新。

11.根据权利要求9所述的数据处理设备，其中，所述图形表示单元被配置为：如果所述展开确定单元确定要展开的子级别的数量小于当前展开的子级别的数量，则通过收起所述活动分支的超出的子级别来更新所述树视图，以及沿所述第一方向移动经更新的树视图，直到隐藏所收起的超出子级别的对应元素变得与所述指向光标的根据所述第一方向的位置对齐为止。

12.根据权利要求9所述的数据处理设备，包括：

-选择单元，被配置为响应于由用户输入的模式选择在自动模式和手动模式之间选择；其中

-所述数据处理设备被配置为仅在所述自动模式下执行以下各项：使用所述位置确定单元对所述指向光标的位置的确定、使用所述活动分支选择单元对所述活动分支的自动选择、使用所述展开确定单元对要展开的子级别的确定以及使用所述图形表示单元对所述树视图的自动更新，以及其中

-所述图形表示单元被配置为在所述手动模式下响应于针对所述图形用户界面的其它手动用户输入来更新所述树视图。

13.根据权利要求9所述的数据处理设备，被配置为生成所述指向光标以特别用于与所述树视图的用户交互。

14.一种用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备，包括：

-处理装置；

-显示装置；以及

-存储器装置，存储指令，所述指令在被执行时使所述处理装置：

-生成以多个分支来组织的树结构的树视图；

-确定所述指向光标相对于所述树视图的位置，所述位置包含根据第一方向的第一位置值和根据与所述第一方向正交的第二方向的第二位置值，所述第一位置值确定活动分支被选择为与所述指向光标的根据所述第一方向的位置对齐的分支；

-所述第二位置值根据相对于所述活动分支的子级别的缩进的第二位置值来确定要展开的子级别，使得所述第二位置值的增加值确定所述树结构中的更深的子级别，所述第二位置值的减少值确定所述树结构中的更浅的子级别；以及

-在所述第一位置值和所述第二位置值中的至少一个改变时，通过展开所确定的所述活动分支的子级别并收起其他分支来更新所述树视图。

用于渲染树结构的数据处理设备和方法

技术领域

[0001] 提出了用于渲染元素的树结构的数据处理设备和计算机实现的方法。具体地，本公开涉及用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备和计算机实现的方法，以及涉及相应的计算机可读存储介质。

背景技术

[0002] 用于例如多媒体数据的语义元数据注释的概念经常是根据分级式超概念/子概念方案来排列的，导致了概念分类系统(concept taxonomies)。例如，交通工具分类系统可以包括子概念地面交通工具、水上船只、飞行器等，而地面交通工具还包括如小汽车、卡车、公共汽车等的子概念。在分类系统中，任何子概念是其所有超概念的一类。因此，如果要注释的特定概念在分类系统中丢失，则注释者可能回退到最佳匹配的超概念。例如，主战坦克可以被注释为地面交通工具，而UFO(不明飞行物)可被注释为交通工具。

[0003] 大型且复杂的分类系统可以具有若干独立的顶级概念、每个顶级概念下的大量分级级别、以及单独超概念的很多子概念。这大量的深度嵌套的概念可能难以在使用该分类系统的应用的图形用户界面(GUI)中方便地渲染，例如，在示出了该分类系统以用于选择要注释的概念的视图中。

[0004] 渲染分类系统的一种可能性是采用表视图。此处，每个顶级概念可以得到其自己的表或可以是表列或行的标题(header)。例如，列标题示出了超概念且该列的内容示出了对应的子概念。如果需要进一步的粒度，则列可以包含子表，例如以粗体示出的直接子概念以及以正常字体示出的其下级概念。这种表视图可以在小空间中示出很多概念及其分级并可以提供良好的概念概览。然而，在不引起混淆的情况下，其一次仅可以示出几个分级级别。如果用户需要回顾其他子级别，则用户必须选择要示出的其他子级别的概念，例如通过更新当前示出的表或者示出附加表。在深度嵌套的分类系统中，为了到达特定概念，多个这种选择可能是必要的。如果未找到感兴趣的概念，则用户可能想要再次回退到示出了更高级别概念的表。在分类系统的较粗和较细粒度视图之间的该切换类似于数据库和透视表(pivot table)的在线分析处理(OLAP)中与渲染和处理多维数据有关的下钻(drill down)/上钻(drill up)操作。

[0005] 渲染分类系统的另一可能性是采用树视图。这种视图用于示出各种应用中的深度嵌套的分级，例如文件系统浏览器或文档提纲(outlining)工具。这种应用可以提供控制，以将所选树级别下的当前示出的树级别的数目增加/减少一个或多个级别，展开/收起所选树级别下的所有树级别，将特定配置的子树收起至所选级别且之后再次示出如之前配置的该子树，示出顶级下的特定数目的树级别等。具有这种功能的树视图还可以用于渲染分类系统，使得用户可以浏览并展开感兴趣的子树，以找到并选择感兴趣的概念。这种树视图可以渲染具有很多分级级别的超大型分类系统。然而，其不太适合提供概念的概览。如果展开很多树级别，则在展开式视图中需要很多滚动来找到概念，且难以跟踪概念的关系。如果对于更紧凑的视图来说仅展开较少的树级别，则需要很多展开/收起操作来下钻到低级别概

念或从低级别概念上钻。

[0006] 对于小型分类系统，使用分类系统的应用的GUI可以包含表视图，但是表视图可能难以正确处理大型分类系统的大量分级级别，导致需要频繁地更新表视图或快速地示出子表。另一方面，在一些应用中，手动可配置的树视图用于大型分类系统，但是其由于紧凑视图对所需视图重配置操作数量的问题而缺少了便利性。同样地问题不仅适用于在分类系统中搜索概念，还适用于在其他复杂树视图中搜索元素，例如在深度嵌套的文件目录结构中搜索文件或者在深度嵌套的数据库结构中搜索数据库条目。

[0007] 依然需要解决手动配置的树视图的便利性问题(以例如用于渲染大型分类系统)，并提供更高效的人机界面，该人机界面允许对在树视图中显示的元素(例如，树视图中显示的分类系统中的概念)的树结构中的元素(即，节点)进行增强式寻找和选择。

发明内容

[0008] 建议了根据所附权利要求的用于根据指向光标相对于所渲染的树结构的位置在图形用户界面(GUI)中渲染元素的树结构的计算机实现的方法和数据处理设备、以及计算机可读存储介质。

[0009] 根据实施例，一种用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的计算机实现的方法包括：

[0010] -生成以多个分支来组织的树结构的树视图；

[0011] -确定该指向光标相对于该树视图的位置，该位置包含根据第一方向的第一位置值和根据与该第一方向正交的第二方向的第二位置值；

[0012] -将根据第一方向与指向光标的位置对齐的分支自动选择为活动分支；

[0013] -根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的所述活动分支的子级别；以及

[0014] -通过展开所确定的活动分支的子级别并收起其他分支来自动更新树视图。

[0015] 相应地，一种用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备包括：

[0016] -图形表示单元，被配置为生成以多个分支来组织的树结构的树视图；

[0017] -位置确定单元，被配置为确定指向光标相对于树视图的位置，该位置包含根据第一方向的第一位置值和根据与第一方向正交的第二方向的第二位置值；

[0018] -活动分支选择单元，被配置为将根据第一方向与指向光标的位置对齐的分支自动选择为活动分支；以及

[0019] -展开确定单元，被配置为根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的所述活动分支的子级别；

[0020] -图形表示单元，还被配置为通过展开所确定的活动分支的子级别并收起其他分支来自动更新树视图。

[0021] 该设备中包括的单元(例如，图形表示单元、位置确定单元、活动分支选择单元和展开确定单元)可以作为单独装置来提供，作为至少一个装置或逻辑电路来联合提供，或者功能上由微处理器、微控制器或其他处理装置、计算机或其他可编程设备来实现。

[0022] 根据实施例，一种用于根据指向光标相对于所渲染的树结构的位置在图形用户界

面中渲染元素的树结构的数据处理设备包括：

[0023] -处理装置；

[0024] -显示装置；以及

[0025] -存储指令的存储器装置，该指令在被执行时使得处理装置：

[0026] -生成树视图，该树视图使用显示装置示出了以多个分支来组织的树结构；

[0027] -确定指向光标相对于树视图的位置，该位置包含根据第一方向的第一位置值和根据与第一方向正交的第二方向的第二位置值；

[0028] -将根据第一方向与指向光标的位置对齐的分支自动选择为活动分支；

[0029] -根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的活动分支的子级别；以及

[0030] -通过展开所确定的活动分支的子级别并收起其他分支来自动更新由显示装置示出的树视图。各装置通过总线或其他通信线路相连。

[0031] 此外，一种计算机可读存储介质在其中存储有实现了根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的指令，该指令在由计算机执行时，使计算机：

[0032] -生成以多个分支来组织的树结构的树视图；

[0033] -确定指向光标相对于树视图的位置，该位置包含根据第一方向的第一位置值和根据与第一方向正交的第二方向的第二位置值；

[0034] -将根据第一方向与指向光标的位置对齐的分支自动选择为活动分支；

[0035] -根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的活动分支的子级别；以及

[0036] -通过展开所确定的活动分支的子级别并收起其他分支来自动更新树视图。

[0037] 该计算机可读存储介质有形地体现了指令程序，该指令程序在由计算机执行时使计算机执行所述方法步骤。

[0038] 图形用户界面响应于控制指向光标相对于所呈现的元素的树结构的位置的用户输入。可以使用指向光标操控装置(例如，计算机鼠标、轨迹球、轨迹点或触摸板)来移动或操控指向光标。如果显示装置是触摸屏装置，则还可以使用例如用户的手指或触控笔来移动或操控指向光标。在实施例中，指向光标可以响应于在对由相机提供的示出了用户的实况视频的分析期间的手或手指轨迹和手势识别来移动。

[0039] 树结构以多个分支或连接节点的路径来组织，即包括多个分支或连接节点的路径，其可以包含子级别，即父节点下的树级别，其中，子级别中包含的元素或节点作为同一父节点的兄弟节点连接到同一缩进(indentation)。

[0040] 树视图是树结构的图形表示。在GUI元素(例如，窗口或帧)中示出树视图。其允许仅通过评估指向光标相对于树视图的位置来自动选择活动分支，即在不点击的情况下确定指向光标的位置。活动分支是被选择以由用户浏览的分支，例如通过展开子级别来浏览。指向光标的该相对位置可以通过在树视图上移动指向光标或者通过滚动示出了树视图的查看窗口而变化，同时指向光标的绝对位置(即，相对于查看窗口的边界的位置或者相对于显示包含树视图在内的查看窗口的显示装置的屏幕的边界的位置)可以保持不变或可以变化。

[0041] 然后根据相对于活动分支的子级别的缩进的第二位置值来计算要展开活动分支的多少个子级别。之后，在没有用户的手动交互的情况下，自动树视图重配置机制通过展开所确定的活动分支的子级别并收起其他分支来进行对树视图的自动更新。

[0042] 根据所提出方案的自动树视图重配置机制允许快速地检验类似于复杂树的数据结构。其模仿了用户使用手动控制来浏览例如分类系统、文件系统或数据库(该分类系统、文件系统或数据库以树结构组织并在树视图中显示为树)的嵌套数据的行为，以找到并选择感兴趣的元素，即分别是概念、文件或数据库条目，在顶级元素开始，然后一次仅下钻到树的一个分支，并且如果未找到感兴趣的元素则可能再次上钻，包括收起剩下的子级别。该机制使用指向光标相对于树视图的位置来控制自动展开/收起树的哪些分支。其在不需要通过鼠标点击、键盘击键、触摸屏敲击或类似操作的显式展开/收起操作的情况下重配置树视图以用于树的紧凑呈现，并且需要比相应手动控制方案所必需的更少的指向光标移动。

[0043] 根据所提出的方案的机制至少具有以下效果：用户可以用比通过使用手动控制更快速和更方便的方式来执行寻找/选择感兴趣元素的任务，即提供了更高效的人机界面。

[0044] 在一个实施例中，在显示图形用户界面中的树视图的显示装置的屏幕上，第一方向对应于垂直方向以及第二方向对应于水平方向。此处，指向光标相对于树元素的垂直位置用于控制选择树的哪个分支作为可以展开的活动分支，而指向光标相对于树级别的缩进的水平位置用于控制将展开活动分支的多少个子级别。如果发现活动分支的子级别的缩进在指向光标位置左侧，则将该子级别确定为要展开。在不同实施例中，第一方向对应于水平方向且第二方向对应于垂直方向。

[0045] 在一个实施例中，连续重复对指向光标的位置的确定、对活动分支的自动选择、对要展开的子级别的确定以及对树视图的自动更新。该自动机制不断跟踪指向光标的任何移动，且如果必要，该自动机制立刻自动重配置树视图以用于树的紧凑呈现。在实施例中，可以在自动执行展开/收起操作之前引入短延迟。在另一实施例中，将展开和/或收起操作加以延迟，直到执行了(例如，对箭头键的)附加击键为止。

[0046] 在一个实施例中，如果要展开的子级别的数量小于当前展开的子级别的数量，则通过收起活动分支的超出子级别来更新树视图，且在没有进一步用户交互的情况下沿第一方向自动移动经更新的树视图，直到隐藏(即包括)被收起的超出子级别的对应元素变得根据第一方向与指向光标的位置对齐为止。换言之，如果由于展开/收起操作导致指向光标的显示位置变得与经自动重配置的树视图不一致，则沿第一方向(例如，垂直)自动移动经更新的树视图，使得实际指向光标位置反映相对于经更新的树视图的新的逻辑指向光标位置。如果第一方向是垂直方向且由于将指向光标向左移动而导致指向光标与被收起的树级别的直接或间接超元素未垂直对齐，简单地收起该树级别可能引起任意的树元素(如果有的话)与指向光标垂直对齐，这取决于由该树级别的元素之前占据的空间和指向光标相对于这些元素的垂直位置。这可能导致没有自动选择活动分支或者自动选择包括该任意树元素在内的新的活动分支。另一方面，根据实施例来移动树视图使包括超元素在内的该分支保持为活动分支，并实现了在树中的进一步无缝导航。在一些情况下，对包含树视图在内的查看窗口的这种自动重配置或手动垂直滚动可能引起树的最上方的元素以与查看窗口的上边缘间隔某个距离的方式来显示。因此在一个实施例中，将树视图连同所显示的指向光标一起向所述边缘偏移所述距离。

[0047] 在一个实施例中，元素的树视图对应于分类系统，即概念的分类方案，例如来自用于多媒体数据的分类系统的概念。此处，具有交互式树视图的图形用户界面对应于分级元数据的分类系统浏览器，即在GUI中渲染分类系统，例如用于针对多媒体数据的语义元数据注释。根据所提出方案的适用于在显示为树视图中的树的分类系统中寻找和选择概念的任务的自动树视图重配置机制允许对甚至大型分类系统的高效渲染。

[0048] 在另一实施例中，元素的树结构对应于文件的目录结构。一般而言，如果树视图的预期用途是快速浏览并选择树的元素，根据所提出的方案的自动树视图重配置机制也可以用于在各种应用中示出除了分类系统之外的其他内容的视图。例如，与需要显式操作来展开/收起子树(例如，子文件夹)的使用标准树视图的应用相比，使用所提出的树视图重配置机制的包含树视图在内的任何数据库、文档提纲工具、文件浏览器或“打开文件”对话框可能需要更少的控制操作来寻找并选择元素。

[0049] 在一个实施例中，计算机实现的方法包括：

[0050] -响应于由用户输入的模式选择在自动模式和手动模式之间选择；其中

[0051] -仅在自动模式下执行指向光标的位置的确定、活动分支的自动选择、要展开的子级别的确定以及树视图的自动更新，以及其中

[0052] -在手动模式下响应于针对图形用户界面的其它手动用户输入来更新树视图。

[0053] 执行根据所提出的方案的方法的设备可以提供手动树视图配置功能和所提出的自动机制以及在这两种模式之间切换的手段。在实施例中，当打开自动模式时其存储手动建立的视图配置，并在再次关闭自动模式时恢复之前存储的配置。

[0054] 在一个实施例中，专门为了与树视图进行用户交互而生成指向光标。这适用于例如不使用标准指向光标的系统。对于使用例如鼠标指针作为其标准输入的装置来说，可以提供不依赖于滚动条的滚动方案，反之不使用这种标准指向光标的触摸屏控制的装置可以将标准滑动手势用于滚动。在使用由指向输入装置(例如，鼠标、轨迹球、触摸板或轨迹点)来控制的指向光标的系统上，指向光标位置由该指向光标的位置来定义。在不使用这种标准指向光标的系统上，例如触摸屏控制的装置上，代之以使用特别专用于树视图的查看窗口的指向光标。该专用指向光标的位置是通过在查看窗口上由例如用户的手指或触控笔拖动指向光标来控制的。具体地，在使用标准指向光标的系统上以及在根据所提出方案的自动机制活动时，使用该自动机制的应用在查看窗口的边缘处不使用标准滚动条来滚动视图，因为在树视图上移动指向光标以访问这种滚动条可能引起对视图的重配置。取而代之地，使用其他滚动控制手段，例如键盘控制、鼠标滚轮(例如，照常使用滚轮的垂直滚动，以及通过按压滚轮结合鼠标移动的水平滚动，或者在支持的情况下滚轮的左倾/右倾)、或者在指向光标移动到查看窗口的边缘时的自动滚动。另一方面，假如滑动手势并未在指向光标位置处开始，这可被代之以解释为拖动指向光标，则触摸屏控制的应用可以在查看窗口上使用标准滑动手势来滚动视图。

[0055] 在一个实施例中，树视图包括对树元素是否对应于缩进且具有子元素(即，隐藏子树)的视觉指示。该视觉指示可以是图标，例如加号、指向右侧的三角形或箭头。作为另一示例，视觉指示可以通过以不同风格渲染树元素(例如，不同颜色或字体、是否是粗体/斜体等)来给出。在实施例中，视觉指示附加地包括尚未展开级别或子树的复杂度的图形建议，例如指示隐藏元素数量的数字或条(bar)。

[0056] 此外，在一个实施例中，对树结构的可视部分相对于树视图的位置进行可视化的一个或多个位置指示符被示出为查看窗口的一部分。这提供了导航信息，例如，当滚动视图或自动重配置视图导致在查看窗口中没有树视图中的任何部分实际可见时。位置指示符可以是例如滚动条、具有指示位置的较宽部分的小线、和/或指示相对于查看窗口的高度/宽度的位置的平线(plain line)。如果位置指示符提供了滚动控制功能，它们不用于滚动查看窗口，以避免在视图上移动指向光标时对树视图的潜在不想要的重配置。在另一实施例中，在图形用户界面的单独窗口或帧中示出位置指示符。

[0057] 虽然没有明确描述，但是实施例可以被实施为任意组合或子组合。

附图说明

[0058] 图1示意性地示出了用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的方法的实施例；

[0059] 图2示意性地示出了用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备的实施例；

[0060] 图3示意性地示出了用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备的另一实施例；以及

[0061] 图4～32示意性地示出了在经由指向光标的用户交互期间树视图经历树视图重配置机制的实施例。具体地，

[0062] 图4示出了示例树视图；

[0063] 图5示出了指向光标在树视图的查看窗口之外的情形；

[0064] 图6示出了指向光标与树元素未垂直对齐的情形；

[0065] 图7示出了显示仅具有顶级元素的活动分支的情形；

[0066] 图8示出了自动展开活动分支的下一个子级别的情形；

[0067] 图9示出了自动展开活动分支的其他子级别的情形；

[0068] 图10示出了经由子级别树元素来选择活动分支的情形；

[0069] 图11示出了自动展开子级别树元素之下的下一个子级别的情形；

[0070] 图12示出了自动收起树级别的第一部分；

[0071] 图13示出了自动收起树级别的第二部分；

[0072] 图14示出了自动收起树级别的第三部分；

[0073] 图15示出了通过滚动来自动展开子级别的情形的第一部分；

[0074] 图16示出了通过滚动来自动展开子级别的情形的第二部分；

[0075] 图17示出了通过滚动来自动展开子级别的情形的第三部分；

[0076] 图18示出了通过滚动来自动收起子级别的情形的第一部分；

[0077] 图19示出了通过滚动来自动收起子级别的情形的第二部分；

[0078] 图20示出了通过滚动来自动收起子级别的情形的第三部分；

[0079] 图21示出了自动展开最上面分支的情形的第一部分；

[0080] 图22示出了自动展开最上面分支的情形的第二部分；

[0081] 图23示出了自动展开最下面分支的情形的第一部分；

[0082] 图24示出了自动展开最下面分支的情形的第二部分；

[0083] 图25示出了在活动分支改变时自动收起/展开的情形的第一部分；

[0084] 图26示出了在活动分支改变时自动收起/展开的情形的第二部分；

[0085] 图27示出了通过滚动来自动展开最上面分支的情形的第一部分；

[0086] 图28示出了通过滚动来自动展开最上面分支的情形的第二部分；

[0087] 图29示出了通过滚动来自动展开最下面分支的情形的第一部分；

[0088] 图30示出了通过滚动来自动展开最下面分支的情形的第二部分；

[0089] 图31示出了在通过滚动而改变活动分支时自动收起/展开的情形的第一部分；以及

[0090] 图32示出了在通过滚动而改变活动分支时自动收起/展开的情形的第二部分。

具体实施方式

[0091] 为了更好地理解，现将参考附图在以下描述中更详细地解释所提出的方案。应理解：该方案不限于这些示例实施例，且还可以方便地合并和/或修改具体特征，而不脱离如所附权利要求限定的本发明的范围。

[0092] 参见图1，示意性地示出了用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的计算机实现的方法10的实施例。根据所示实施例，该方法提供了不断重配置树视图以用于树的紧凑表示的自动机制。

[0093] 在第一步骤11中，生成以多个分支来组织的树结构的树视图。根据所示实施例中，在下一步骤12中，响应于用户输入的模式选择来执行在自动模式和手动模式之间的选择。在手动模式下，在下一步骤13中响应于针对图形用户界面的进一步手动用户输入来更新树视图。

[0094] 在自动模式下，执行以下步骤：

[0095] 在下一步骤14中，确定指向光标相对于树视图的位置。该位置包含根据第一方向的第一位置值和根据与第一方向正交的第二方向的第二位置值。

[0096] 在下一步骤15中，将根据第一方向与指向光标的位置对齐的分支自动选择为活动分支。在下一步骤16中，根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的活动分支的子级别。在下一步骤17中，通过展开所确定的活动分支的子级别并收起其他分支来自动更新树视图。

[0097] 在下一步骤18中，如果指向光标相对于树视图的位置继续改变，则继续重复指向光标位置确定步骤14、活动分支选择步骤15、子级别展开确定步骤16和自动树更新步骤17。否则，在所示实施例中，该方法等待指向光标相对于树视图的任何更多位置改变。

[0098] 现在参见图2和图3，示意性地示出了用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备的实施例。图2中所示的设备和图3中所示的设备允许实现所述的用于渲染树结构的方法的优点和特性，作为用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的设备的一部分。

[0099] 图2中所示的数据处理设备20包括选择单元21，其被配置为响应于由用户输入22的模式选择在自动模式和手动模式之间选择。

[0100] 图形表示单元23被配置为生成以多个分支来组织的树结构的树视图。图形表示单

元23连接到或可连接到或包括用于显示树视图的显示装置。在手动模式下，图形表示单元23响应于针对图形用户界面的进一步手动用户输入来更新树视图。在自动模式下，图形表示单元23响应于至少来自展开确定单元27的输入来更新树视图。

[0101] 位置确定单元24连接到图形表示单元23，并被配置为确定指向光标相对于树视图的位置。该位置包含根据第一方向的第一位置值和根据与第一方向正交的第二方向的第二位置值。该位置是基于输入25来确定的，输入25来自用户通过滚动或移动指向光标来控制指向光标的位置。

[0102] 活动分支选择单元26被配置为将根据第一方向与指向光标位置对齐的分支自动选择为活动分支，且展开确定单元27被配置为根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的活动分支的子级别。

[0103] 图形表示单元23被配置为通过展开所确定的活动分支的子级别并收起其他分支来自动更新树视图。

[0104] 在图2所示实施例中，选择单元21、图形表示单元23、位置确定单元24、活动分支选择单元26和展开确定单元27彼此直接通信。在另一实施例中，该设备包括连接到一个或多个单元并控制其通信的控制器单元。

[0105] 选择单元21、图形表示单元23、位置确定单元24、活动分支选择单元26和展开确定单元27可以作为单独装置来提供，作为至少一个装置或逻辑电路来联合提供，或者在功能上由连接到至少一个存储器装置或包括至少一个存储器装置在内的微处理器、微控制器或其他处理装置、计算机或被布置为执行该处理的其他可编程设备来实现。

[0106] 如图3所示，用于根据指向光标相对于所渲染的树结构的位置在图形用户界面中渲染元素的树结构的数据处理设备30的实施例包括处理装置31、显示装置32和存储指令的存储器装置33，该指令在被执行时使得处理装置：

[0107] -生成树视图，该树视图使用显示装置32示出了以多个分支来组织的树结构；

[0108] -确定指向光标相对于树视图的位置，该位置包含根据第一方向的第一位置值和根据与第一方向正交的第二方向的第二位置值；

[0109] -将根据第一方向与指向光标的位置对齐的分支自动选择为活动分支；

[0110] -根据相对于活动分支的子级别的缩进的第二位置值来确定要展开的活动分支的子级别；以及

[0111] -通过展开所确定的活动分支的子级别并收起其他分支来自动更新由显示装置示出的树视图。

[0112] 例如，处理装置可以是适用于执行根据所述方法之一的步骤的处理器。在实施例中，所述适用于包括：处理器被配置为(例如，被编程为)执行根据所述方法之一的步骤。

[0113] 现在参见图4至32，作为示例，示意性地示出了在自动模式下经由指向光标的用户交互期间分类系统树视图经历自动树视图重配置机制的实施例。

[0114] 图4示出了在查看窗口中的分类系统树的树视图的示例，当前展开了四个子级别。指向光标的位置由箭头来指示。显示为树视图中的树的分类系统具有9个顶级概念(概念1、...、概念9)，且每个超概念或父节点/元素具有9个子概念或子节点/元素，且具有总共五个分级级别。在图4至32中所示的示例中，操控该分类系统。在图4中，仅展开了与指向光标垂直对齐的分支，而收起其他分支，提供了树的紧凑呈现，然而高至级别5的树元素都可见。

根据所提出的方案，指向光标相对于树视图的位置用于控制树视图的自动展开/收起操作。指向光标相对于树元素的垂直位置控制了可以将树的哪个分支作为活动分支来展开。自动收起树的其他分支。这些分支被称为不活动分支。指向光标的相对于树级别的缩进的水平位置控制了可以展开活动分支的多少个子级别。在自动展开/收起树元素时，还可以自动垂直移动经更新的树视图，使得实际的指向光标位置反映相对于经更新的树视图的新的逻辑指向光标位置。

[0115] 图5示出了当指向光标未指向查看窗口时的树视图。该情形仅在指向光标是系统的指向光标的情况下是可能的，即创建的不是专门用于与树视图交互的标准指向光标。在该情况下，不管指向光标的垂直位置如何，按照定义不存在活动分支。因此，仅显示树的顶级元素。此外，在所示实施例中，树始终以在查看窗口的左上角显示其包括相关联的子元素指示符(如果适用于顶级树元素)在内的最上面元素的方式来显示。当将指向光标移动到查看窗口中时，向用户提供树的紧凑和一致的呈现，作为在树中导航的起点。根据指向光标进入查看窗口之处相对于树视图的位置以及移动的方向，这种移动可能已经引起了对树视图的自动重配置。

[0116] 图6示出了当指向光标指向查看窗口但与任何树元素未垂直对齐时的情形。不选择任何分支作为活动分支，也导致仅显示树的顶级元素。因此，当系统的指向光标在进入位置不与任何树元素垂直对齐的情况下被移动到查看窗口中时，树视图不改变。然而，由于在指向光标不指向查看窗口时在查看窗口的上边缘处显示最上面的树元素，除非完全没有显示任何树，否则将指向光标从上面移动到查看窗口中引起进入位置与最上面的树元素垂直对齐。当指向光标指向查看窗口且与树元素垂直对齐时，该元素所属的分支是活动分支。可以展开活动分支的多少个子级别取决于指向光标的水平位置。如果该位置对应于最高树级别，称为树级别1，即如果指向光标位于树级别2的缩进的左侧，则仅显示活动分支的顶级元素连同不活动分支的顶级元素一起，参见图7。从图7所示的位置出发，指向光标可以在查看窗口内向上和向下移动而不改变树视图。是否存在活动分支以及哪个顶级树元素与活动分支相关联随着指向光标相对于树视图的垂直位移而改变。

[0117] 当存在活动分支且指向光标向右移动超过下一个树级别的缩进时，则自动展开活动分支的下一个子级别(如果存在的话)。如果指向光标与被展开的树级别的直接超元素垂直对齐，该下一个子级别自然是该直接超元素的子级别。否则，该下一个子级别被定义为该活动分支的当前展开的最低树级别的最上面的树元素的子级别。例如，如果将指向光标向右移动等价于图7所示的树视图中的树缩进的一个级别的距离，则作为结果的树视图对应于图8，因为作为结果的水平指向光标位置对应于树级别2，即指向光标位于树级别2和3的缩进之间。因此，一旦指向光标在其路上跨过树级别2的缩进，则自动展开活动分支的树级别2。通过将指向光标进一步向右移动4个树缩进级别的距离，一旦跨过对应的缩进则展开活动分支的越来越多的子级别，导致图9中所示的经重配置的树视图，且作为结果的水平指向光标位置有意地对应于树级别6，以示出将指向光标移动到活动分支的最低树级别的缩进右侧的进一步缩进不会引起对树视图的任何更多的改变。根据水平指向光标位置对活动分支的子级别的自动展开还发生在系统的指向光标从左侧、右侧、或上方移动到查看窗口中且进入位置与树元素垂直对齐(该树元素始终是顶级树元素之一)时，因为仅这些树元素在指向光标不指向查看窗口时才被显示。包括与指向光标垂直对齐的顶级树元素在内的分

支变为活动分支，且根据指向光标相对于树视图的水平位置被自动展开，这受到活动分支的子级别的数目的约束。

[0118] 通常，用户将不一次展开活动分支的多于一个子级别，而是将首先浏览最近展开的子级别的树元素以查看是否存在感兴趣的元素以及是否想要展开该元素下的子级别。例如，如果用户决定展开图8所示的树视图中的Concept_3_3下的子级别，他将有可能将指向光标向下移动至Concept_3_3，以使包括Concept_3_3在内的分支成为活动分支，参见图10，然后将指向光标向右移动以展开下一个子级别，参见图11。如果用户继续展开图11所示树视图中的每个树级别的每个第三元素，则树视图将最终对应于图4(除了被较小的查看窗口的下边缘所截断之外)。

[0119] 当存在活动分支且指向光标向左移动跨过除了树级别1之外的当前展开的最低树级别的缩进时，该树级别被自动收起。如果指向光标与被收起的树级别的直接或间接超元素垂直对齐，则活动分支不改变。例如，如果将指向光标向左移动等价于图9所示的树视图中的树缩进的4个级别的距离，则作为结果的树视图对应于图8，因为作为结果的水平指向光标位置对应于树级别2。因此，一旦指向光标在其路上跨过树级别5的缩进时，自动收起活动分支的树级别5，且之后一旦跨过相应缩进，则自动收起活动分支的树级别4和3。通过将指向光标进一步向左移动一个树缩进级别，还收起了活动分支的树级别2，得到图7所示的树视图。类似地，一旦指向光标跨过树级别3的缩进，则在图11所示的树视图中将指向光标向左移动得到图10所示的树视图。

[0120] 当由于将指向光标向左移动而导致指向光标与被收起的树级别的直接或间接超元素未垂直对齐时，简单地收起该树级别将引起任意的树元素(如果有的话)与指向光标垂直对齐，这取决于由该树级别的元素之前占据的空间和指向光标相对于这些元素的垂直位置。因此，在没有用户的控制的情况下，将存在包括该任意树元素在内的新的活动分支或者将没有活动分支。因此，在所示实施例中，自动垂直移动经更新的树视图，使得在该情况下已被收起的树级别的直接超元素与指向光标垂直对齐，且该超元素相对于指向光标的垂直位移与在树视图更新之前曾与指向光标垂直对齐的树元素相同，因为该位置是经更新的树视图内的新的逻辑指向光标位置。这也使得包括该超元素在内的分支成为活动分支，实现了在树中的进一步无缝导航。另一个方案将是代之以将指向光标移动到该超元素的位置，这被假定为让用户困惑。此外，该超元素当前可能是不可见的，无论如何都要求移动经更新的树视图。

[0121] 对经更新的树视图的自动垂直移动可能引起树的最上方的元素以与查看窗口的上边缘间隔某个距离的方式来显示。一旦用户展开感兴趣的接下来的子级别并在其中滚动，快速地化解这种位移。为了方便，在另一实施例中，该机制包括将树视图的最上方元素再次移动到查看窗口的上边缘的校正功能，包括移动指向光标，使得其停留在相对于树视图的相同位置处。该校正功能可以在用户请求时被触发，例如，通过键盘或鼠标控制、或者多点触控手势、或者在配置为用户首选的情况下一旦这种位移将发生时自动触发。

[0122] 在图12至图14中示出了由将指向光标向左移动所引起的对树级别的自动收起以及作为结果的对经更新的树视图的垂直移动的示例。图12示出了在指向光标移动之前的视图。现在，将指向光标向左移动等价于一个树缩进级别的距离，得到图13所示的树视图，因为一旦指向光标在其路上跨过树级别3的缩进，则自动收起树级别3且自动向下移动经更新

的树视图，使得在树视图更新之前曾显示Concept_3_3的垂直位置处显示Concept_3_3的直接超元素。指向光标进一步向左移动一个树缩进级别得到图14所示的树视图，且在与图13中的Concept_3_3相同垂直位置处显示图14中的Concept_3。

[0123] 如果树视图的相对于未改变的指向光标位置的水平位移导致与之前不同的树级别相对应的水平指向光标位置，则对树级别的自动展开或收起还可以由查看窗口的水平滚动来触发，且自动展开受到活动分支的子级别的数目的约束。原则上，将查看窗口向右滚动受到树视图中向右到达最远的与查看窗口的右边缘右对齐的元素的约束，而不管该元素在查看窗口内是否实际可见。然而，由于活动分支的当前展开的最低树级别的缩进右侧的下一个树级别的缩进应当可由指向光标来访问(通过水平滚动和水平指向光标移动的任何组合)，以实现对活动分支的下一个子级别(如果存在的话)的展开，查看窗口允许被向右滚动，使得该缩进可由指向光标清楚地访问。这导致了树视图中被示出的向右到达最远的元素与查看窗口的右边缘间隔某个距离(如果该元素实际上可见且并未长到足以清楚地延伸至该缩进)。另一方面，在树的顶级元素左侧不存在要访问的树级别的缩进，因此向左滚动查看窗口可以受到在查看窗口的左边缘处示出的这些顶级元素(包括相关联的子元素指示符(如果对它们适用的话))的约束。

[0124] 由图15至图17示出了由向右滚动查看窗口所引起的对活动分支的子级别的自动展开的示例。图15示出了在滚动之前的视图。由于指向光标几乎已到达了查看窗口的右边缘，通过向右移动指向光标不能展开活动分支的进一步子级别，但是必须代之以通过向右滚动查看窗口来展开活动分支的进一步子级别。此外，活动分支的树级别3的元素被查看窗口的右边缘所截断，因此用户还可能想要向右滚动视图以能够读取这些元素的更多文本。现在将查看窗口向右滚动等价于一个树缩进级别的距离，得到与树级别4水平相对应的未改变的指向光标位置，因此自动展开活动分支的树级别4，参见图16。假如展开活动分支的下一个子级别或者能够读取活动分支的树级别3的依然被截断的元素，现在将查看窗口再次向右滚动一个树缩进级别，得到与树级别5水平相对应的未改变的指向光标位置，且因此自动展开活动分支的树级别5，参见图17。用户现在可以通过将查看窗口进一步向右滚动来尝试展开活动分支的进一步子级别，但是由于不存在这种子级别，这将仅导致未改变的树视图在查看窗口内向左移动，直到活动分支的树级别5中不再有元素被查看窗口的右边缘所截断(假如所有这些元素长到足够清楚地延伸至树级别6的缩进)，使得这些元素相对于查看窗口的右边缘的进一步位移是不必要的。如果代之以执行了滚动以能够读取活动分支的树级别3的元素的文本，则用户将可能将指向光标向左移动，直到其水平位置对应于树级别3，以再次自动收起活动分支的树级别4和5。当由于向左滚动查看窗口而导致指向光标与被收起的树级别的直接或间接超元素垂直对齐时，则活动分支不改变。例如，如果将图17所示的查看窗口向左滚动等价于一个树缩进级别的距离，作为结果的树视图对应于图16，因为未改变的指向光标位置水平对应于树级别4，因此简单地自动收起活动分支的树级别5。通过向左移动一个树缩进级别的另一个滚动操作，还自动收起了活动分支的树级别4，得到图15所示的树视图。当由于将查看窗口向左滚动而导致指向光标与被收起的树级别的直接或间接超元素未垂直对齐时，还垂直移动经更新的树视图，使得已被收起的树级别的直接超元素与未改变的指向光标位置垂直对齐，类似于在类似情况下和出于相同原因而将指向光标向左移动时的行为。这可能再次导致树视图的最上方元素相对于查看窗口的上边缘的

位移，这可以如上所述来化解。

[0125] 在图18至图20中示出了由将查看窗口向左滚动所引起的对树级别的自动收起以及作为结果的对经更新的树视图的垂直移动的示例。图18示出了在滚动之前的视图。然后将查看窗口向左滚动等价于一个树缩进级别的距离，得到与树级别4水平对应的未改变的指向光标位置，因此自动收起树级别5且自动向下移动经更新的树视图，使得在树视图更新之前曾显示Concept_2_2_2_2_2的垂直位置处显示Concept_2_2_2_2(Concept_2_2_2_2_2的直接超元素)，参见图19。向左滚动一个树缩进级别的另一个滚动操作得到图20所示的树视图，且在与图19中的Concept_2_2_2_2相同垂直位置处显示图20中的Concept_2_2_2。

[0126] 对树级别的自动展开或收起还可以由指向光标在查看窗口内的垂直移动来触发或者由查看窗口的垂直滚动来触发(如果作为结果的指向光标相对于树视图的垂直位移使得不活动分支变为活动分支或反之)。这包括对当前活动分支(如果有的话)的自动收起，以及对新的活动分支(如果有的话)的自动展开，后者取决于指向光标相对于树视图的水平位置，且自动展开受到新的活动分支的子级别的数目的约束。由于在这些操作之前和之后显示的树视图可能多少具有共同的树元素，作为结果的对树视图的更新可能实际上多少是可见的，甚至完全没有改变可见。此外，所提出的机制可以自动垂直移动经更新的树视图，使得实际的指向光标位置反映相对于经更新的树视图的新的逻辑指向光标位置。这可能再次导致树视图的最上方元素相对于查看窗口的上边缘的位移，这可以如上所述来化解。

[0127] 对标准树视图的垂直滚动通常受到树视图的在查看窗口的上边缘处示出的最上方元素的约束以及受到树视图的在查看窗口的下边缘处示出的最下方元素的约束。然而，该方案并不适合使用所提出机制的树视图。首先，在指向光标相对于树视图位移时对树视图的自动重配置无论如何将经常引起树视图的最上方或最下方元素分别相对于查看窗口的上边缘或下边缘的垂直位移，因此仅在查看窗口的垂直滚动时避免该行为没什么用处。其次，对查看窗口的垂直滚动本身将经常引起对视图的重配置，因此根据在滚动之前显示的树视图来约束这种滚动可以导致交替滚动操作且要求指向光标移动以到达感兴趣的树元素，因为可以避免进一步的滚动，直到进一步的指向光标移动再次对其进行解锁。此外，如果使用所提出的机制的应用代之以检查是否可以显示符合这种标准滚动约束的经更新的树视图，且如果预期的滚动距离违反了这些约束，则将不得不减少该滚动距离且将不得不再次执行该检查(有可能是多次)。因此，在所示实施例中，为了分别向上或向下滚动查看窗口，根据所提议的方案的垂直滚动受到树视图中与查看窗口内的指向光标垂直对齐的最上方或最下方元素的约束。这使得用户能够在未改变的指向光标位置之下向上或向下滚动整个树视图。此外，用户可以通过在查看窗口内选择对应的垂直指向光标位置来容易地选择在指向光标之上或之下可以显示多少树视图部分。由指向光标在查看窗口内的垂直移动或者由查看窗口的垂直滚动所引起的指向光标相对于树视图的垂直位移所导致的相对于树视图的新的逻辑指向光标位置(被称为新的逻辑指向光标位置)是借助被称为经裁剪(tailored)的树视图的概念树视图来定义的。经裁剪的树视图被定义为：不管垂直指向光标位置如何，所有元素和分支展开到与指向光标的水平位置相对应的树级别的实际树视图，且水平指向光标位置相对于实际树视图和经裁剪的树视图是相同的。相对于实际树视图和经裁剪的树视图的垂直指向光标位置如下定义：当指向光标位于实际树视图的最上方元素之上时，其在概念上也位于经裁剪的树视图的最上方元素之上，且距离这些元素有相

同的垂直距离。当指向光标与实际树视图的元素垂直对齐时，其在概念上也与经裁剪的树视图的相同元素垂直对齐，且相对于这些元素有相同的垂直位移。当指向光标位于实际树视图的最下方元素之下时，其在概念上也位于经裁剪的树视图的最下方元素之下，且距离这些元素有相同的垂直距离。新的逻辑指向光标位置被定义为：如果指向光标相对于实际树视图的垂直位移代之以应用于经裁剪的树视图，指向光标将具有的相对于经裁剪的树视图的位置，且具有相同方向和距离。

[0128] 当指向光标位于查看窗口之内并在树视图的最上方元素之上时，不存在活动分支，导致仅显示树的顶级元素，参见图21。从图21所示的位置开始，只要指向光标停留在树视图的最上方元素之上，指向光标就可以在查看窗口内向上和向下移动，而不改变树视图。当向下移动指向光标跨过空白区域和树视图的最上方元素之间的垂直边界时，其也将跨过空白区域和经裁剪的树视图的最上方元素之间的垂直边界。因此，新的逻辑指向光标位置与经裁剪的树视图的最上方元素垂直对齐，且包括该最上方元素在内的分支变为新的活动分支且被自动展开。由于树视图的最上方元素与经裁剪的树视图的最上方元素始终是同一树元素，新的逻辑指向光标位置与实际指向光标位置匹配，且不需要对经更新的树视图的垂直移动。例如，如果在图21所示的查看窗口中指向光标向下移动到图22所示的位置处，作为结果的树视图对应于图22，因为一旦指向光标在其路上跨过空白区域和Concept_1之间的垂直边界，则新的逻辑指向光标位置与Concept_1垂直对齐且包括Concept_1在内的分支变为活动分支且由于水平指向光标位置而被展开至树级别2。当在查看窗口内指向光标向上移动跨过树视图的最上方元素和上面空白区域之间的垂直边界时，其也将跨过经裁剪的树视图的最上方元素和上面空白区域之间的垂直边界。因此，新的逻辑指向光标位置就在树视图的最上方元素之上，且包括该最上方元素在内的分支不再是活动分支且被自动收起。由于树视图的最上方元素和经裁剪的树视图的最上方元素一致，则新的逻辑指向光标位置与实际指向光标位置匹配，且不需要对经更新的树视图的垂直移动。例如，如果在图22所示的查看窗口中指向光标向上移动到图21所示的位置处，作为结果的树视图对应于图21，因为一旦指向光标在其路上跨过Concept_1和上面空白区域之间的垂直边界，则新的逻辑指向光标位置不再与任何树元素垂直对齐且包括Concept_1在内的分支不再是活动分支且被自动收起。

[0129] 当指向光标位于查看窗口之内并在树视图的最下方元素之下时，不存在活动分支，导致仅显示树的顶级元素，参见图23。从图23所示的位置开始，只要指向光标停留在树视图的最下方元素之下，指向光标就可以在查看窗口内向上和向下移动，而不改变树视图。当向上移动指向光标跨过空白区域和树视图的最下方元素之间的垂直边界时，其也将跨过空白区域和经裁剪的树视图的最下方元素之间的垂直边界。因此，新的逻辑指向光标位置与经裁剪的树视图的最下方元素垂直对齐，且包括该最下方元素在内的分支变为活动分支且被自动展开。由于树视图的最下方元素可能与经裁剪的树视图的最下方元素不是相同的树元素，这可能导致新的逻辑指向光标位置与实际指向光标位置不匹配。如果是这种情况，经更新的树视图树视图也自动垂直移动，使得新的逻辑指向光标位置和实际的指向光标位置相同，导致在树视图更新之前曾显示树视图的最下方元素的垂直位置处显示经更新的树视图的最下方元素。例如，如果在图23所示的查看窗口中指向光标向上移动到图24所示的位置处，作为结果的树视图对应于图24，因为一旦指向光标在其路上跨过空白区域和

Concept_9之间的垂直边界，则新的逻辑指向光标位置与Concept_9_9垂直对齐，且包括Concept_9_9在内的分支变为活动分支且由于水平指向光标位置而被展开至树级别2，且向上移动经更新的树视图，使得在树视图更新之前曾显示Concept_9的垂直位置处显示Concept_9_9。当在查看窗口中指向光标向下移动跨过树视图的最下方元素和下面空白区域之间的垂直边界时，其也将跨过经裁剪的树视图的最下方元素和下面空白区域之间的垂直边界。因此，新的逻辑指向光标位置就在树视图的最下方元素之下，且包括该最下方元素在内的分支不再是活动分支且被自动收起。由于树视图的最下方元素可能与经更新的树视图的最下方元素不是相同的树元素，这可能导致新的逻辑指向光标位置与实际指向光标位置不匹配。如果是这种情况，经更新的树视图树视图也自动垂直移动，使得新的逻辑指向光标位置和实际的指向光标位置相同，导致在树视图更新之前曾显示树视图的最下方元素的垂直位置处显示经更新的树视图的最下方元素。例如，如果在图24所示的查看窗口中指向光标向下移动到图23所示的位置处，作为结果的树视图对应于图23，因为一旦指向光标在其路上跨过Concept_9_9和下面空白区域之间的垂直边界，则新的逻辑指向光标位置不与任何树元素垂直对齐，包括Concept_9_9在内的分支不再是活动分支且被自动收起，以及向下移动经更新的树视图，使得在树视图更新之前曾显示Concept_9_9的垂直位置处显示Concept_9。

[0130] 跨过树视图的两个垂直相邻元素之间的垂直边界的指向光标移动导致了对活动分支的改变，且自动收起当前活动分支并自动展开新的活动分支。根据移动的方向，新的逻辑指向光标位置与经裁剪的树视图中的与指向光标之前曾垂直对齐的树元素直接上方或下方的元素垂直对齐。如果对树视图的更新导致新的逻辑指向光标位置与实际指向光标位置不匹配，则还自动垂直移动经更新的树视图，使得新的逻辑指向光标位置和实际指向光标位置相同，导致经更新的树视图的元素与在树视图更新之前曾显示的树视图的指向光标所移动到的元素的垂直位置处显示的指向光标垂直对齐。例如，如果在图25所示的查看窗口中指向光标向下移动到图26所示的位置处，作为结果的树视图对应于图26，因为一旦指向光标在其路上跨过Concept_1_9和Concept_2之间的垂直边界，则新的逻辑指向光标位置与Concept_2垂直对齐，包括Concept_1_9在内的分支不再是活动分支且被自动收起，包括Concept_2在内的分支变为新的活动分支且由于水平指向光标位置而被展开至树级别2，且向下移动经更新的树视图，使得在树视图更新之前曾显示Concept_2的相同垂直位置处显示Concept_2。如果在图26所示的查看窗口中指向光标向上移动到图25所示的位置处，作为结果的树视图对应于图25，因为一旦指向光标在其路上跨过Concept_2和Concept_1之间的垂直边界，则新的逻辑指向光标位置与Concept_1_9垂直对齐，包括Concept_2在内的分支不再是活动分支且被自动收起，包括Concept_1_9在内的分支变为新的活动分支且由于水平指向光标位置而被展开至树级别2，且向上移动经更新的树视图，使得在树视图更新之前曾显示Concept_1的垂直位置处显示Concept_1_9。

[0131] 为了一致性，就好像指向光标从下方(从空白区域或从当前活动分支)移动到新的活动分支一样，对活动分支的子级别的自动展开还发生在系统的指向光标从下方移动到查看窗口中且进入位置与树元素垂直对齐(该树元素始终是顶级树元素之一)时，因为仅这些树元素在指向光标不指向查看窗口时才被显示。根据指向光标相对于树视图的水平位置，包括与新的逻辑指向光标位置垂直对齐的树元素在内的分支变为活动分支且被自动展开，

这受到活动分支的子级别的数目的约束，且自动垂直移动经更新的树视图，使得新的逻辑指向光标位置与实际指向光标位置相匹配。

[0132] 以类似于指向光标在查看窗口内的垂直移动的方式来处理对查看窗口的垂直滚动，只是有两个值得注意的不同：首先，与垂直指向光标移动始终连续相反，垂直滚动可能不是连续的，有可能导致新的逻辑指向光标位置与经裁剪的树视图的任意元素(而不是最上方、最下方或垂直相邻的元素)垂直对齐。其次，与在查看窗口内在树视图之上或之下移动指向光标不受约束相反，对查看窗口的垂直滚动可能受到如上所述的约束。在所示实施例中，应用了垂直滚动约束。此外，以下示例仅示出了进行等价于树元素的高度的倍数的距离的垂直滚动。假如在树视图重配置之前和之后指向光标与树元素垂直对齐，其他滚动距离将附加地导致指向光标相对于在重配置的树视图中与指向光标垂直对齐的树元素以及相对于在树视图重配置之前曾与指向光标垂直对齐的树元素的不同垂直位移。

[0133] 当指向光标在查看窗口内且在树视图的最上方元素之上时，仅可以向下滚动查看窗口，导致只要新的逻辑指向光标位置尚未与经裁剪的树视图的任何元素垂直对齐，则未改变的树视图就简单地在查看窗口中向上移动。一旦新的逻辑指向光标位置与经裁剪的树视图的元素第一次垂直对齐，则包括该元素在内的分支变为活动分支且被自动展开。如果对树视图的更新导致新的逻辑指向光标位置与实际指向光标位置不匹配，则还自动垂直移动经更新的树视图，使得新的逻辑指向光标位置和实际指向光标位置相同。例如，如果将图27中所示的查看窗口向下滚动等价于4个树元素的高度的距离，则作为结果的树视图对应于图28，因为新的逻辑指向光标位置与Concept_1_1_1垂直对齐且包括Concept_1_1_1在内的分支变为活动分支且由于水平指向光标位置而展开至树级别3。由于新的逻辑指向光标位置已经匹配实际指向光标位置，不需要对经更新的树视图的垂直移动。

[0134] 当指向光标在查看窗口内且在树视图的最下方元素之下时，仅可以向上滚动查看窗口，导致只要新的逻辑指向光标位置尚未与经裁剪的树视图的任何元素垂直对齐，则未改变的树视图就简单地在查看窗口中向下移动。一旦新的逻辑指向光标位置与经裁剪的树视图的元素第一次垂直对齐，则包括该该元素在内的分支变为活动分支且被自动展开。如果对树视图的更新导致新的逻辑指向光标位置与实际指向光标位置不匹配，则还自动垂直移动经更新的树视图，使得新的逻辑指向光标位置和实际指向光标位置相同。例如，如果将图29中所示的查看窗口向上滚动等价于4个树元素的高度的距离，则作为结果的树视图对应于图30，因为新的逻辑指向光标位置与Concept_9_9_7垂直对齐，包括Concept_9_9_7在内的分支变为活动分支且由于水平指向光标位置而展开至树级别3，且向上移动经更新的树视图，使得新的逻辑指向光标位置和实际指向光标位置相同。

[0135] 当存在活动分支且对查看窗口的垂直滚动导致新的逻辑指向光标位置与经裁剪的树视图的元素垂直对齐，且该树元素不同于之前曾与指向光标垂直对齐的树元素，则活动分支改变。因此，自动收起当前活动分支，且自动展开新的活动分支。如果对树视图的更新导致新的逻辑指向光标位置与实际指向光标位置不匹配，则还自动垂直移动经更新的树视图，使得新的逻辑指向光标位置和实际指向光标位置相同。例如，如果将图31中所示的查看窗口向下滚动等价于4个树元素的高度的距离，则作为结果的树视图对应于图32，因为新的逻辑指向光标位置与Concept_8_9_2垂直对齐，包括Concept_8_8_8在内的分支不再是活动分支且被自动收起，包括Concept_8_9_2在内的分支变为新的活动分支且由于水平指向

光标位置而展开至树级别3，且向下移动经更新的树视图，使得在树视图更新之前曾显示Concept_8_8_8的垂直位置处显示Concept_8_9_2。如果将图32中所示的查看窗口向上滚动等价于4个树元素的高度的距离，则作为结果的树视图对应于图31，因为新的逻辑指向光标位置与Concept_8_8_8垂直对齐，包括Concept_8_9_2在内的分支不再是活动分支且被自动收起，包括Concept_8_8_8在内的分支变为新的活动分支且由于水平指向光标位置而展开至树级别3，且向上移动经更新的树视图，使得在树视图更新之前曾显示Concept_8_9_2的垂直位置处显示Concept_8_8_8。

[0136] 当向上或向下滚动查看窗口将分别导致新的逻辑指向光标位置在树视图的最上方元素之上或最下方元素之下时，滚动约束生效，减少滚动距离，分别使得新的逻辑指向光标位置与经裁剪的树视图的最上方或最下方元素垂直对齐。因此，自动收起当前活动分支(如果有的话)，且包括经裁剪的树视图的最上方或最下方元素在内的分支分别变为新的活动分支并被自动展开。如果对树视图的更新直接导致新的逻辑指向光标位置与实际指向光标位置不匹配，则还自动垂直移动经更新的树视图，使得新的逻辑指向光标位置和实际指向光标位置相同。这是与由对查看窗口的垂直滚动所引起的行为相同的行为，对查看窗口的垂直滚动导致新的逻辑指向光标位置分别与经裁剪的树视图的最上方或最下方元素垂直对齐。

[0137] 如本领域技术人员将认识到的：本原理的各方面可以体现为设备、系统、方法或计算机可读介质。因此，本原理的各方面可以采用硬件实施例、软件实施例或结合软件和硬件方面的实施例的形式。此外，本原理的方案可以采用计算机可读存储介质的形式。可以使用一个或多个计算机可读存储介质的任意组合。

[0138] 本原理的各方面可以例如至少部分地实现为包括代码部分在内的计算机程序，其用于在可编程设备上运行时执行根据本发明的实施例的方法的步骤，或者使得可编程设备能够执行根据本发明的实施例的设备或系统的功能。

[0139] 此外，在图2和图3中示出的任何连接可以是直接或间接的连接。此外，本领域技术人员将认识到在逻辑块之间的边界仅是说明性的，且备选实施例可以合并逻辑块或对各种逻辑块应用对功能的备选分解。

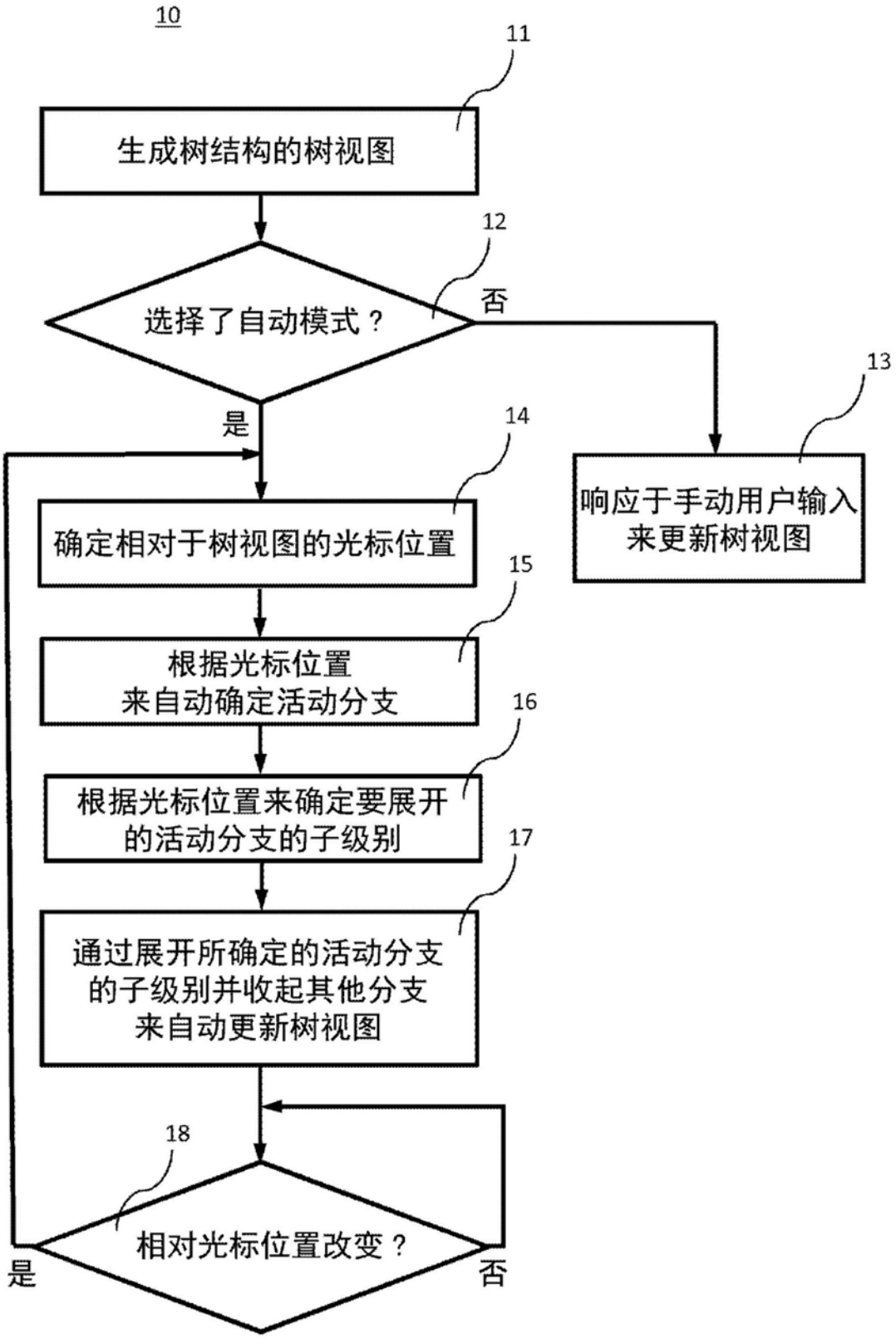

10
11
生成树结构的树视图
12
选择了自动模式？
否
是
13
响应于手动用户输入
来更新树视图
14
确定相对于树视图的光标位置
15
根据光标位置
来自动确定活动分支
16
根据光标位置来确定要展开
的活动分支的子级别
17
通过展开所确定的活动分支
的子级别并收起其他分支
来自动更新树视图
18
相对光标位置改变？
是
否

图1

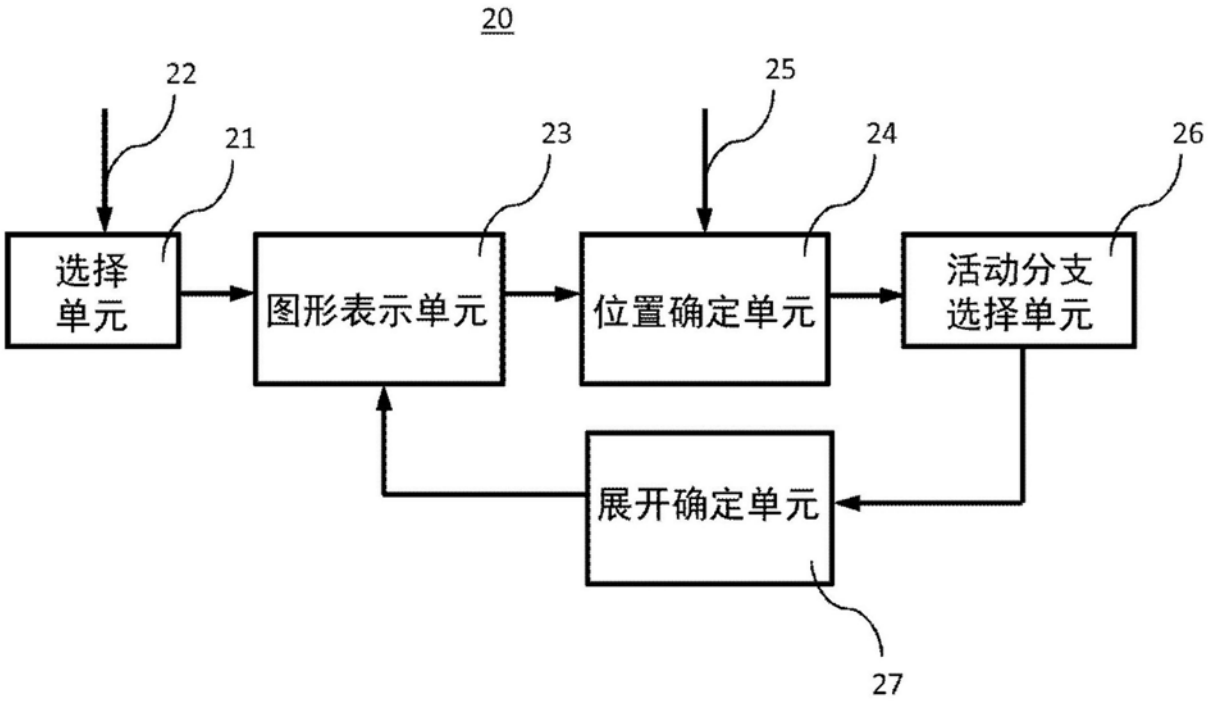

20
22
21
23
25
24
26
选择单元
图形表示单元
位置确定单元
活动分支选择单元
展开确定单元
27

图2

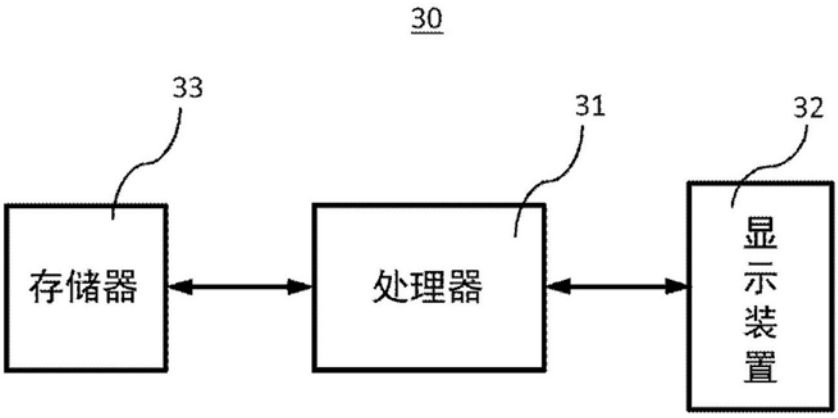

30
33
31
32
存储器
处理器
显示装置

图3

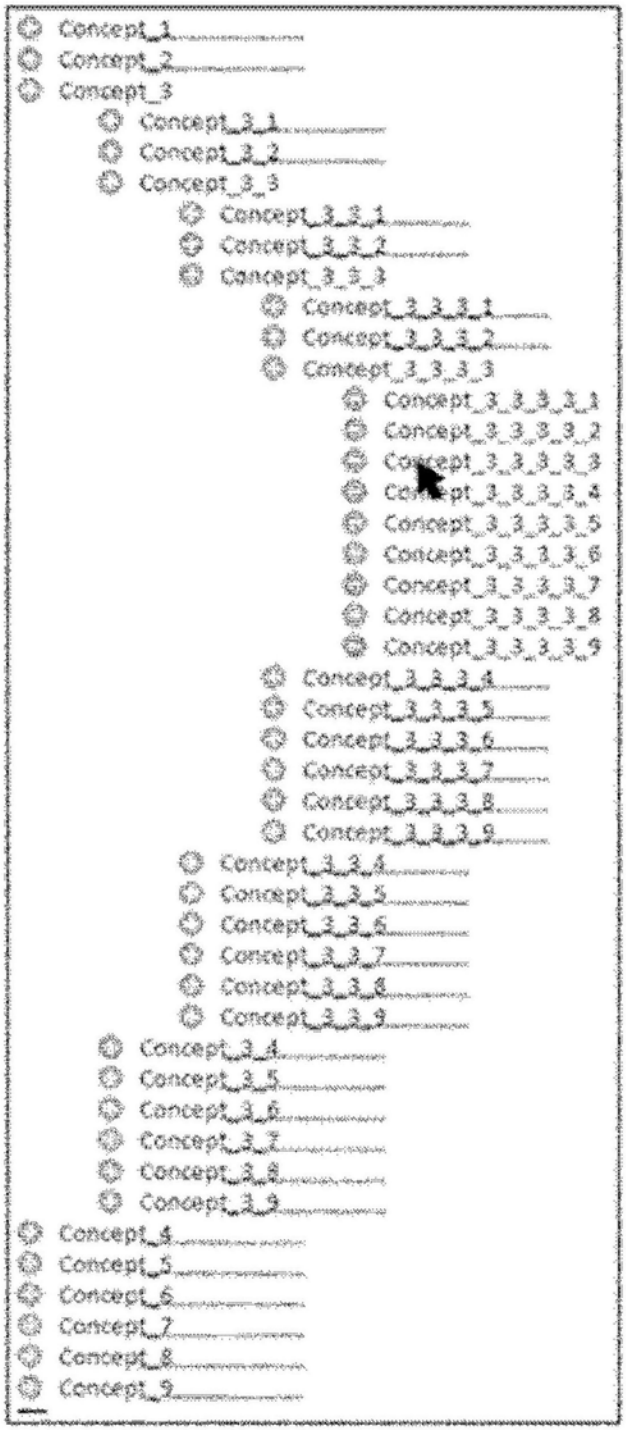

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_2
Concept_3_3
Concept_3_3_1
Concept_3_3_2
Concept_3_3_3
Concept_3_3_3_1
Concept_3_3_3_2
Concept_3_3_3_3
Concept_3_3_3_3_1
Concept_3_3_3_3_2
Concept_3_3_3_3_3
Concept_3_3_3_3_4
Concept_3_3_3_3_5
Concept_3_3_3_3_6
Concept_3_3_3_3_7
Concept_3_3_3_3_8
Concept_3_3_3_3_9
Concept_3_3_3_4
Concept_3_3_3_5
Concept_3_3_3_6
Concept_3_3_3_7
Concept_3_3_3_8
Concept_3_3_3_9
Concept_3_3_4
Concept_3_3_5
Concept_3_3_6
Concept_3_3_7
Concept_3_3_8
Concept_3_3_9
Concept_3_4
Concept_3_5
Concept_3_6
Concept_3_7
Concept_3_8
Concept_3_9
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图4

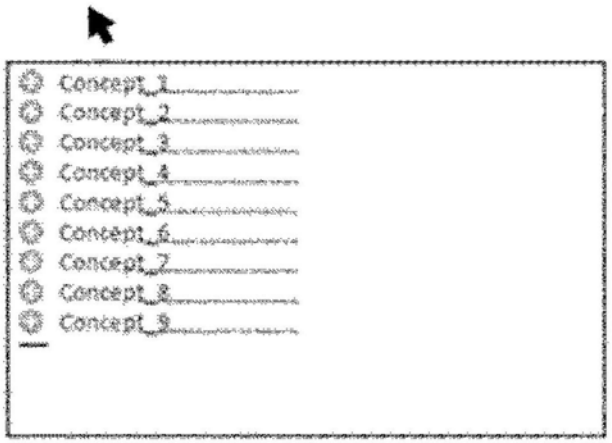

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图5

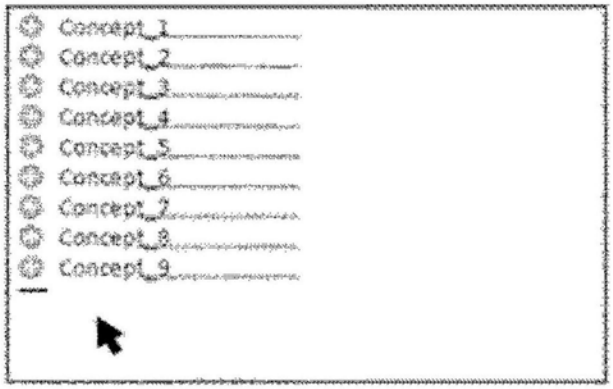

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图6

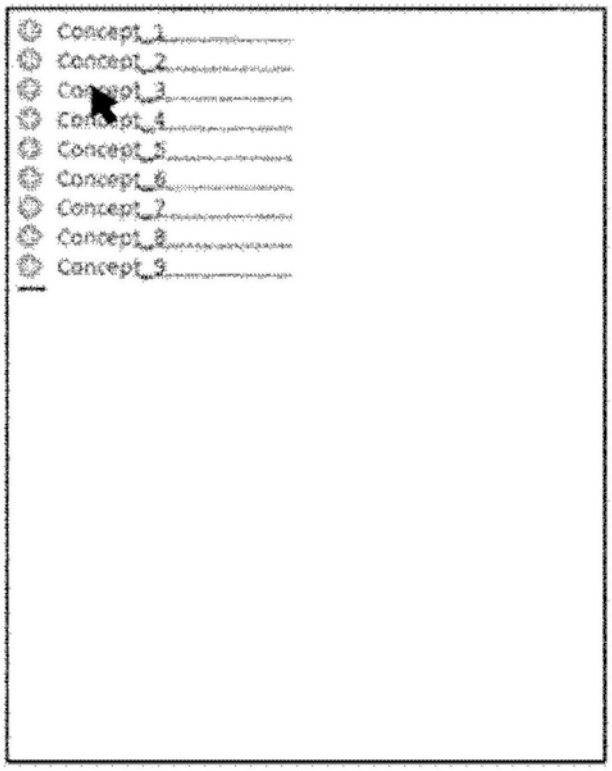

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图7

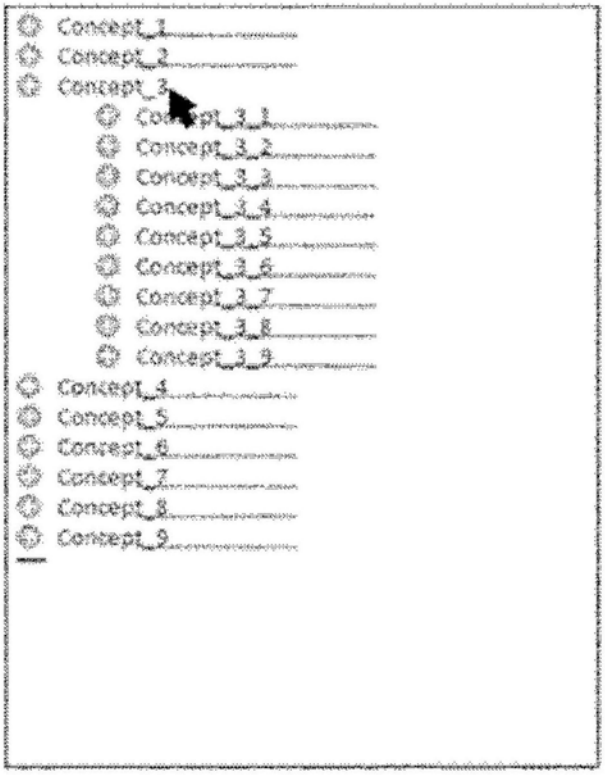

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_2
Concept_3_3
Concept_3_4
Concept_3_5
Concept_3_6
Concept_3_7
Concept_3_8
Concept_3_9
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图8

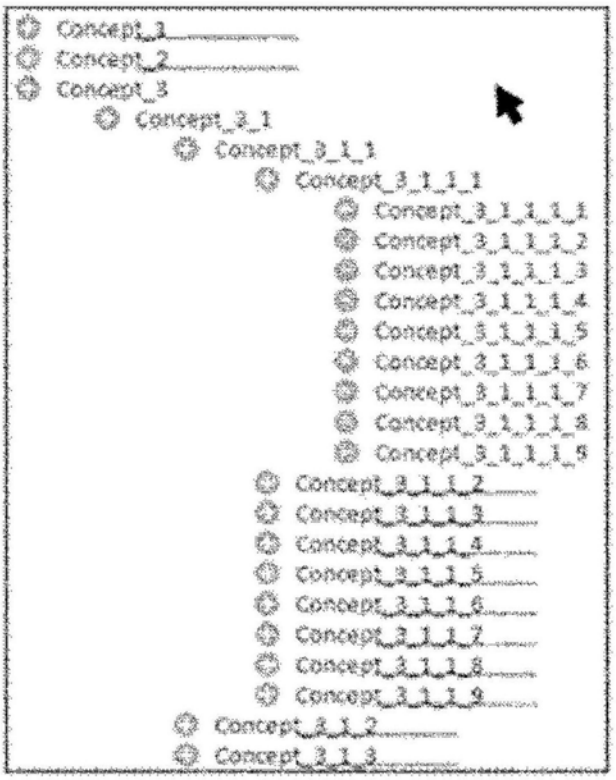

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_1_1
Concept_3_1_1_1
Concept_3_1_1_1_1
Concept_3_1_1_1_2
Concept_3_1_1_1_3
Concept_3_1_1_1_4
Concept_3_1_1_1_5
Concept_3_1_1_1_6
Concept_3_1_1_1_7
Concept_3_1_1_1_8
Concept_3_1_1_1_9
Concept_3_1_1_2
Concept_3_1_1_3
Concept_3_1_1_4
Concept_3_1_1_5
Concept_3_1_1_6
Concept_3_1_1_7
Concept_3_1_1_8
Concept_3_1_1_9
Concept_3_1_2
Concept_3_1_3

图9

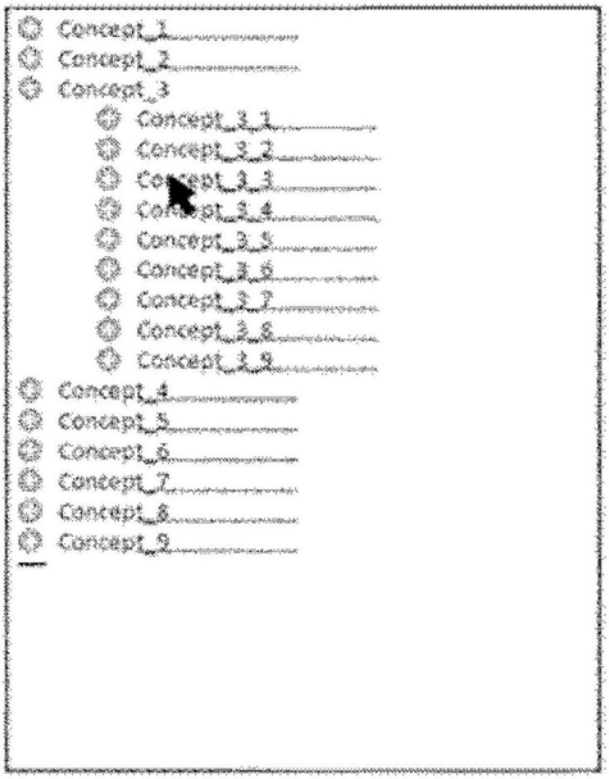

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_2
Concept_3_3
Concept_3_4
Concept_3_5
Concept_3_6
Concept_3_7
Concept_3_8
Concept_3_9
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图10

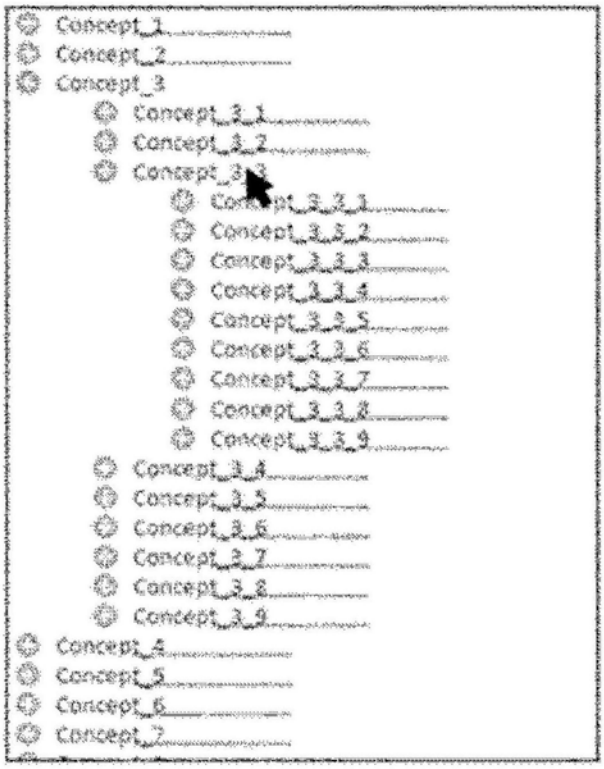

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_2
Concept_3_3
Concept_3_3_1
Concept_3_3_2
Concept_3_3_3
Concept_3_3_4
Concept_3_3_5
Concept_3_3_6
Concept_3_3_7
Concept_3_3_8
Concept_3_3_9
Concept_3_4
Concept_3_5
Concept_3_6
Concept_3_7
Concept_3_8
Concept_3_9
Concept_4
Concept_5
Concept_6
Concept_7

图11

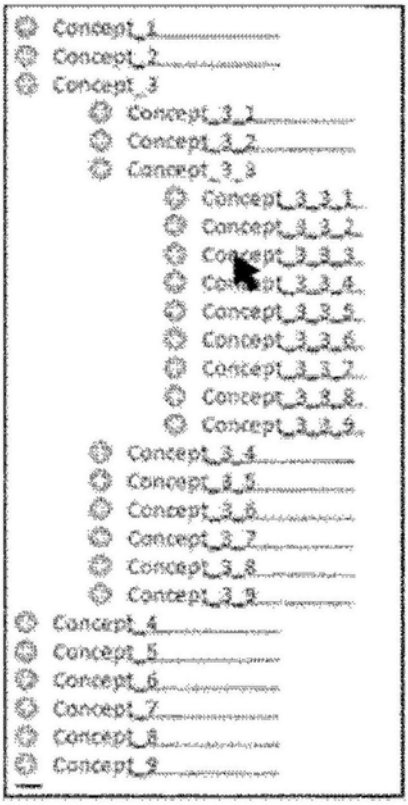

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_2
Concept_3_3
Concept_3_3_1
Concept_3_3_2
Concept_3_3_3
Concept_3_3_4
Concept_3_3_5
Concept_3_3_6
Concept_3_3_7
Concept_3_3_8
Concept_3_3_9
Concept_3_4
Concept_3_5
Concept_3_6
Concept_3_7
Concept_3_8
Concept_3_9
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图12

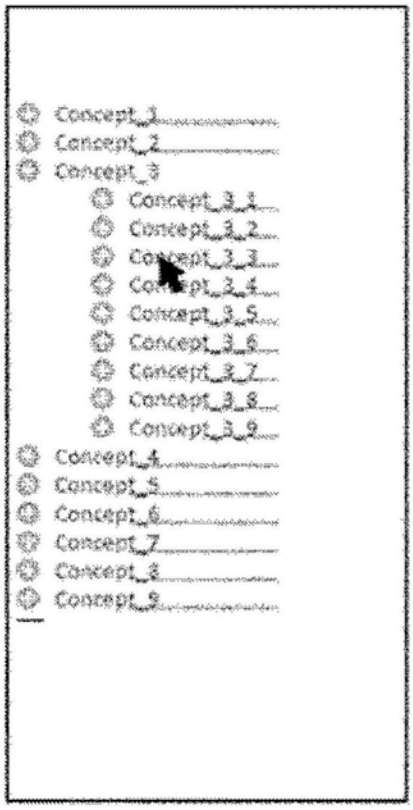

Concept_1
Concept_2
Concept_3
Concept_3_1
Concept_3_2
Concept_3_3
Concept_3_4
Concept_3_5
Concept_3_6
Concept_3_7
Concept_3_8
Concept_3_9
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图13

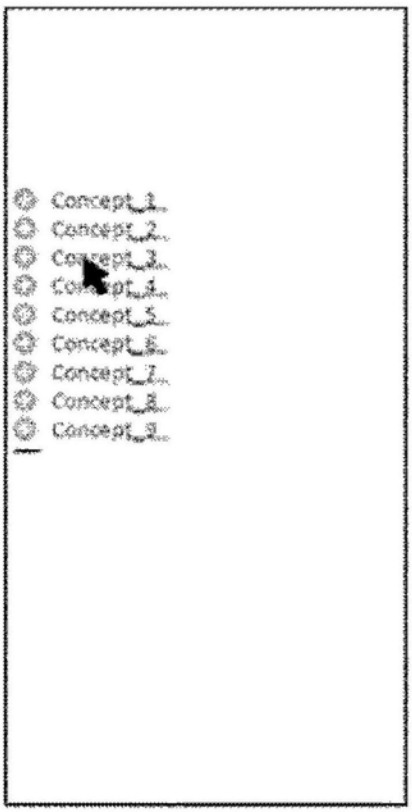

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图14

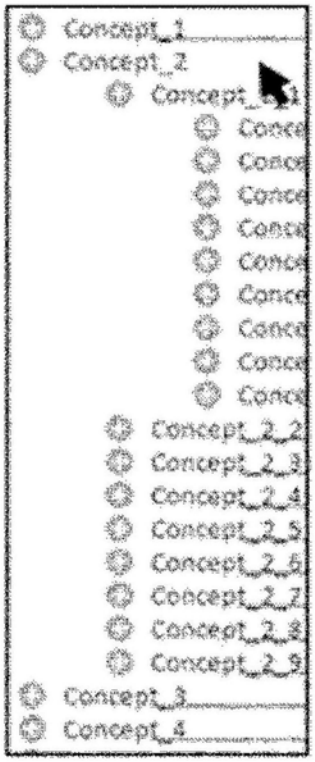

Concept_1
Concept_2
Concept_2_2
Concept_2_3
Concept_2_4
Concept_2_5
Concept_2_6
Concept_2_7
Concept_2_8
Concept_2_9
Concept_3
Concept_4

图15

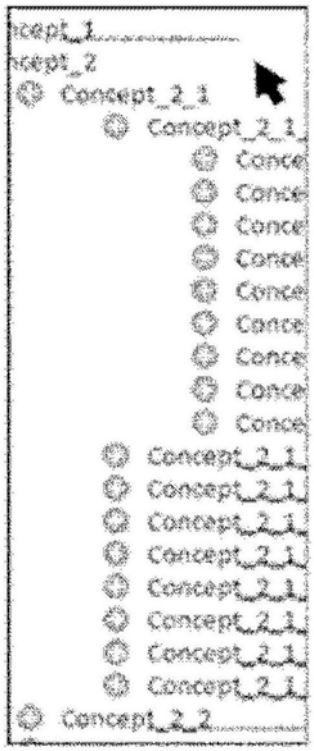


图16

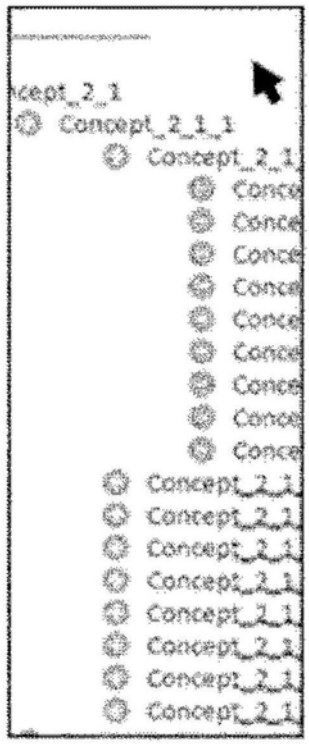


图17

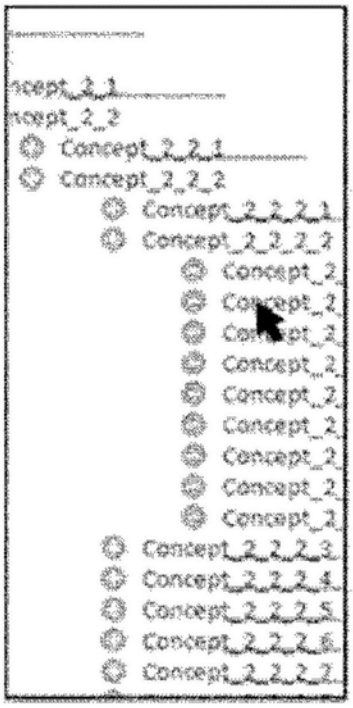

Concept_2_2_1
Concept_2_2_2
Concept_2_2_2_1
Concept_2_2_2_2
Concept_2_2_2_3
Concept_2_2_2_4
Concept_2_2_2_5
Concept_2_2_2_6
Concept_2_2_2_7

图18

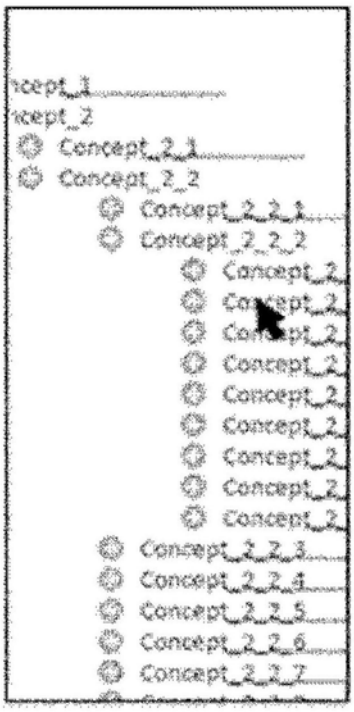

Concept_2_1
Concept_2_2
Concept_2_2_1
Concept_2_2_2
Concept_2_2_3
Concept_2_2_4
Concept_2_2_5
Concept_2_2_6
Concept_2_2_7

图19

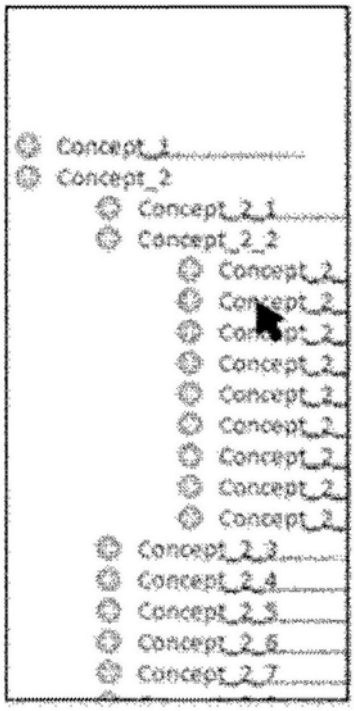

Concept_1
Concept_2
Concept_2_1
Concept_2_2
Concept_2_3
Concept_2_4
Concept_2_5
Concept_2_6
Concept_2_7

图20

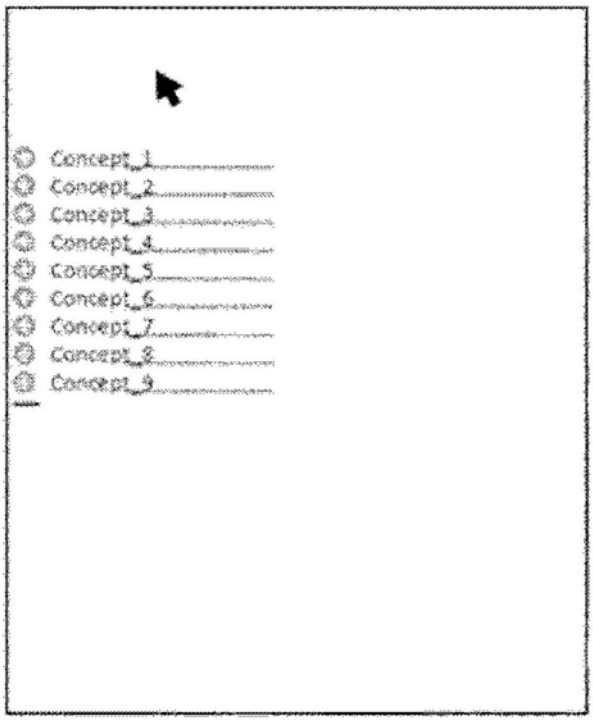

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图21

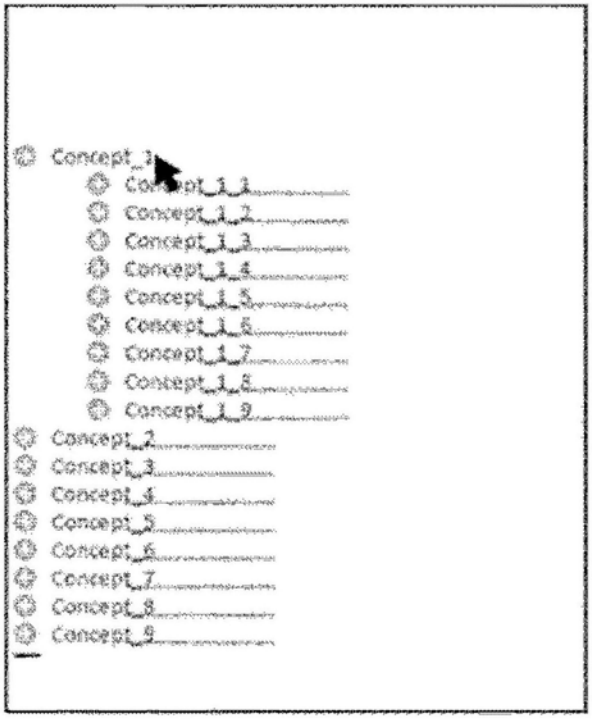

Concept_1
Concept_1_1
Concept_1_2
Concept_1_3
Concept_1_4
Concept_1_5
Concept_1_6
Concept_1_7
Concept_1_8
Concept_1_9
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图22

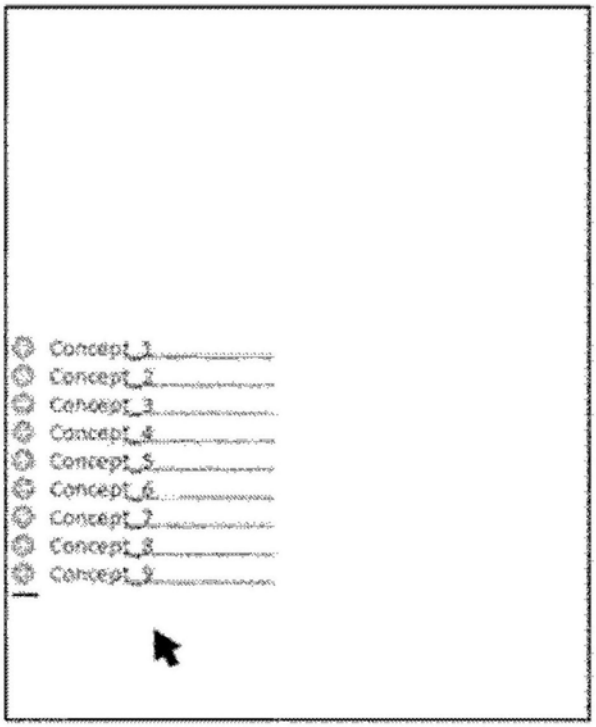

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图23

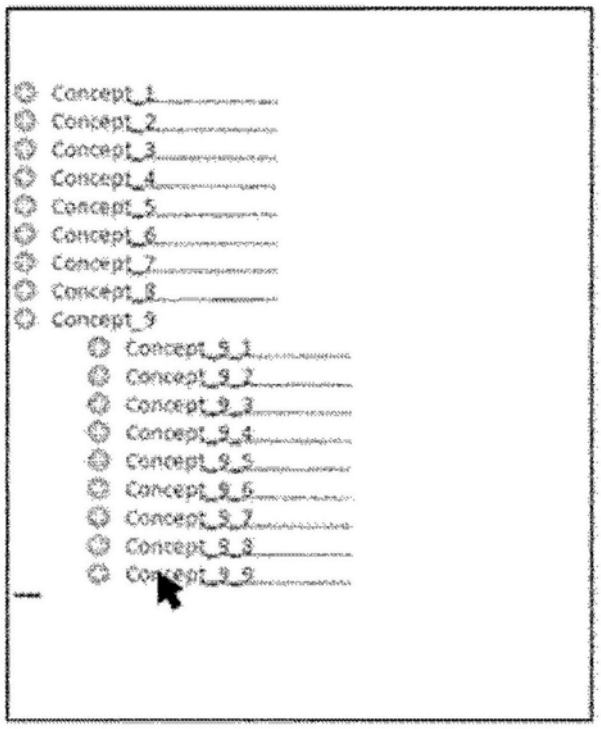

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9
Concept_9_1
Concept_9_2
Concept_9_3
Concept_9_4
Concept_9_5
Concept_9_6
Concept_9_7
Concept_9_8
Concept_9_9

图24

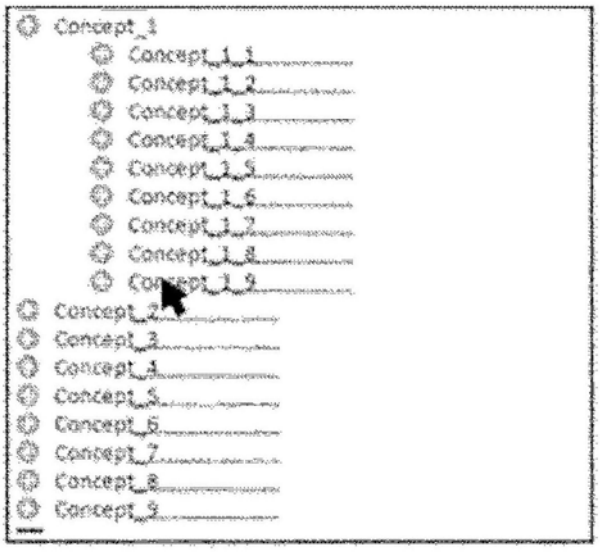

Concept_1
Concept_1_1
Concept_1_2
Concept_1_3
Concept_1_4
Concept_1_5
Concept_1_6
Concept_1_7
Concept_1_8
Concept_1_9
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图25

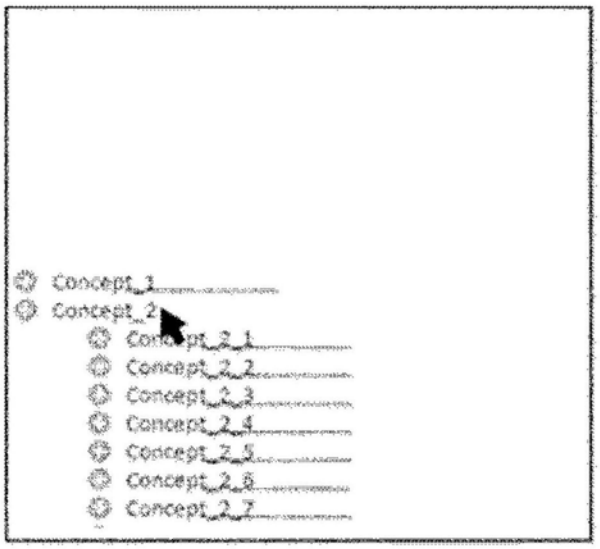

Concept_1
Concept_2
Concept_2_1
Concept_2_2
Concept_2_3
Concept_2_4
Concept_2_5
Concept_2_6
Concept_2_7

图26

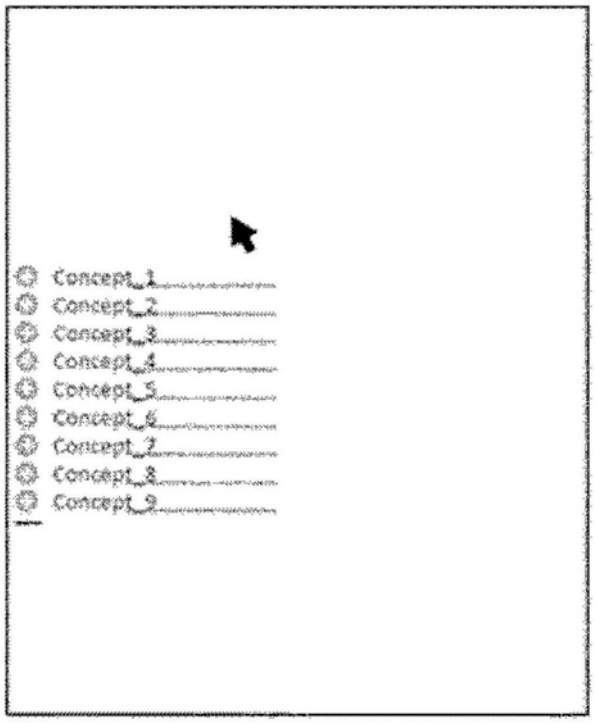

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图27

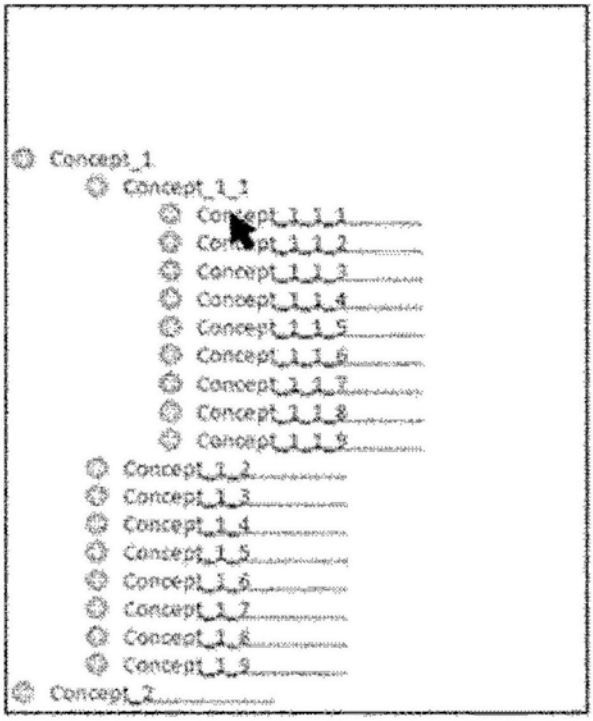

Concept_1
Concept_1_1
Concept_1_1_1
Concept_1_1_2
Concept_1_1_3
Concept_1_1_4
Concept_1_1_5
Concept_1_1_6
Concept_1_1_7
Concept_1_1_8
Concept_1_1_9
Concept_1_2
Concept_1_3
Concept_1_4
Concept_1_5
Concept_1_6
Concept_1_7
Concept_1_8
Concept_1_9
Concept_2

图28

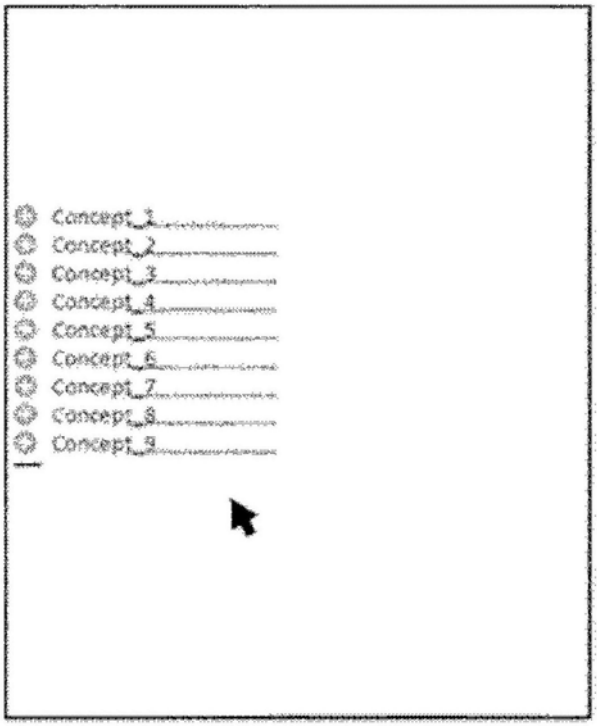

Concept_1
Concept_2
Concept_3
Concept_4
Concept_5
Concept_6
Concept_7
Concept_8
Concept_9

图29

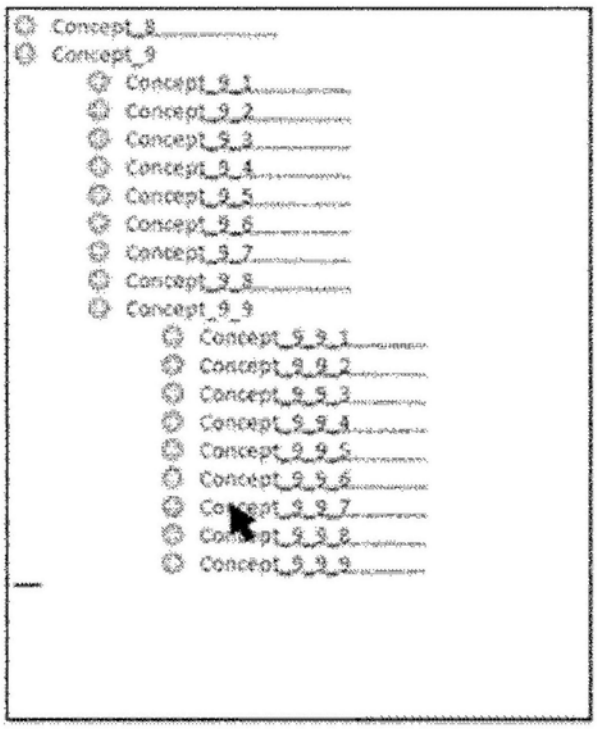

Concept_8
Concept_9
Concept_9_1
Concept_9_2
Concept_9_3
Concept_9_4
Concept_9_5
Concept_9_6
Concept_9_7
Concept_9_8
Concept_9_9
Concept_9_9_1
Concept_9_9_2
Concept_9_9_3
Concept_9_9_4
Concept_9_9_5
Concept_9_9_6
Concept_9_9_7
Concept_9_9_8
Concept_9_9_9

图30

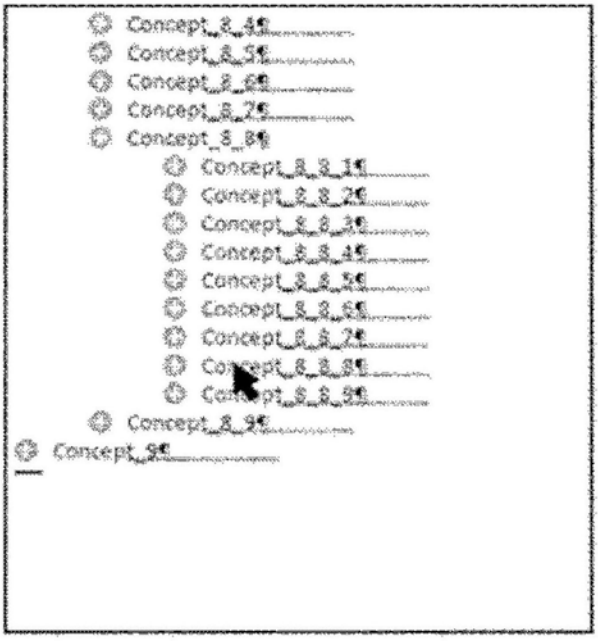

Concept_8_4¶
Concept_8_5¶
Concept_8_6¶
Concept_8_7¶
Concept_8_8¶
Concept_8_8_1¶
Concept_8_8_2¶
Concept_8_8_3¶
Concept_8_8_4¶
Concept_8_8_5¶
Concept_8_8_6¶
Concept_8_8_7¶
Concept_8_8_8¶
Concept_8_8_9¶
Concept_8_9¶
Concept_9¶

图31

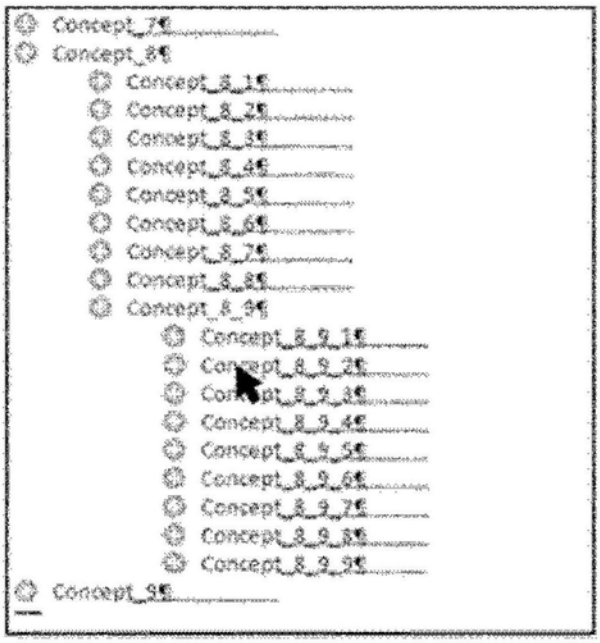

Concept_7¶
Concept_8¶
Concept_8_1¶
Concept_8_2¶
Concept_8_3¶
Concept_8_4¶
Concept_8_5¶
Concept_8_6¶
Concept_8_7¶
Concept_8_8¶
Concept_8_9¶
Concept_8_9_1¶
Concept_8_9_2¶
Concept_8_9_3¶
Concept_8_9_4¶
Concept_8_9_5¶
Concept_8_9_6¶
Concept_8_9_7¶
Concept_8_9_8¶
Concept_8_9_9¶
Concept_9¶

图32