(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 115066673 A
(43)申请公布日 2022.09.16

(21)申请号 202080095553.3
(22)申请日 2020.12.03
(30)优先权数据
20168225.9 2020.04.06 EP
(66)本国优先权数据
PCT/CN2019/122588 2019.12.03 CN
(85)PCT国际申请进入国家阶段日
2022.08.03
(86)PCT国际申请的申请数据
PCT/EP2020/084346 2020.12.03
(87)PCT国际申请的公布数据
WO2021/110785 EN 2021.06.10
(71)申请人 皇家飞利浦有限公司
地址 荷兰艾恩德霍芬市
(72)发明人 胡意仪 欧阳恩
(74)专利代理机构 北京市金杜律师事务所 11256
专利代理师 李春辉
(51)Int.Cl.
G06F 8/41(2006.01)
G06F 9/451(2006.01)

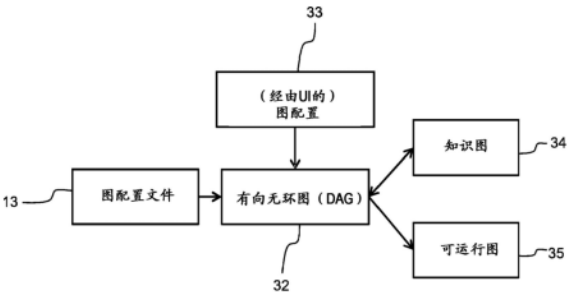
权利要求书2页 说明书12页 附图4页

(54)发明名称

用于ETL流水线处理的系统和方法

(57)摘要

本发明提供了一种ETL流水线系统，其包括被配置为获得多个图配置组件的接口。每个图配置组件包括表示一个或多个计算逻辑规则的信息。该系统还包括被配置为基于所获得的图配置组件来生成计算图的计算图生成器。所生成的计算图包括针对每个图配置组件的节点以及表示节点之间的关系的一个或多个链接。该系统还包括计算图适配器，该计算图适配器被配置为从外部源接收与图配置组件有关的外部信息，并基于外部信息来适配所生成的计算图。该系统还包括被配置为运行所适配的计算图的计算图运行器。



CN 115066673 A

1.一种ETL流水线系统,包括:

接口,其被配置为获得多个图配置组件,每个图配置组件包括表示一个或多个计算逻辑规则的信息;

计算图生成器,其被配置为基于所获得的图配置组件来生成计算图,所生成的计算图包括:针对每个图配置组件的节点;以及表示所述节点之间的关系的一个或多个链接;

计算图适配器,其被配置为从外部源接收与所述图配置组件有关的外部信息,并且基于所述外部信息来适配所生成的计算图;以及

计算图运行器,其被配置为运行所适配的计算图。

2.根据权利要求1所述的系统,其中每个图配置组件包括以下至少一者:

数据库连接;

文档;

数据元素;

算法;以及

计算逻辑规则。

3.根据权利要求2所述的系统,其中所述算法包括计算数据提取规则,和/或所述计算逻辑规则包括计算决策规则。

4.根据权利要求1至3中任一项所述的系统,其中所述外部信息包括以下至少一者:

用户定义的图配置组件;以及

来自外部知识数据库的外部知识。

5.根据权利要求4所述的系统,其中所述外部知识包括存储在数据库中的先前计算图,并且其中所述先前计算图被添加到所生成的计算图以生成所适配的计算图。

6.根据权利要求4至5中任一项所述的系统,其中所述用户定义的图配置组件包括重新定义和/或修改一个或多个图配置组件的信息。

7.根据权利要求4至6中任一项所述的系统,其中配置组件包括图配置文件,并且其中所述系统被配置为基于所述用户定义的图配置组件,重新定义所述图配置文件或指示不同图配置组件之间的依赖性关系的依赖性信息。

8.根据权利要求1至7中任一项所述的系统,其中所述系统还包括计算图解析器,所述计算图解析器被配置为:

从所述接口接收所述图配置组件;以及

基于所述组件信息和指示不同图配置组件之间的依赖性关系的依赖性信息,在不同图配置组件之间建立链接。

9.根据权利要求8所述的系统,其中所述计算图解析器还被配置为基于与所述数据元素和/或所述文档相关联的数据项,在所述数据元素与所述文档之间建立链接。

10.根据权利要求9所述的系统,其中所述数据项包括以下至少一者:

标识符;

时间戳;

内容项;

上下文项;以及

编码对象。

11.根据任一前述权利要求所述的系统,其中所述计算图运行器还被配置为在自适应高吞吐量计算中运行所适配的计算图。

12.一种医学数据处理系统,包括根据任一前述权利要求所述的ETL流水线系统。

13.一种用于ETL流水线处理的方法,所述方法包括:

获得多个图配置组件,每个图配置组件包括表示一个或多个计算逻辑规则的信息;

基于所获得的图配置组件来生成计算图,所生成的计算图包括:针对每个图配置组件的节点;以及表示所述节点之间的关系的一个或多个链接;

从外部源接收与所述图配置组件有关的外部信息;

基于所述外部信息来适配所生成的计算图;以及

运行所适配的计算图。

14.一种用于ETL流水线处理的计算机程序产品,其中所述计算机程序产品包括计算机可读存储介质,所述计算机可读存储介质具有利用其体现的程序指令,所述程序指令能够由处理单元执行以使所述处理单元执行一种方法,所述方法包括:

获得多个图配置组件,每个图配置组件包括表示一个或多个计算逻辑规则的信息;

基于所获得的图配置组件来生成计算图,所生成的计算图包括:针对每个图配置组件的节点;以及表示所述节点之间的关系的一个或多个链接;

从外部源接收与所述图配置组件有关的外部信息;

基于所述外部信息来适配所生成的计算图;以及

运行所适配的计算图。

15.一种处理系统,包括至少一个处理器和根据权利要求14所述的计算机程序产品,其中所述至少一个处理器适于执行所述计算机程序产品的所述计算机程序代码。

用于ETL流水线处理的系统和方法

技术领域

[0001] 本发明涉及数据处理领域，并且尤其涉及提取、转换和加载(ETL)流水线处理领域。

背景技术

[0002] 提取、转换和加载(ETL)是由数据处理和分析系统用来从各种源收集数据、根据业务要求对数据进行转换并且将其加载到目的地数据存储装置中的流水线(即一组进程)。在ETL系统中，转换发生在专门的引擎中，其中分级表通常用于在数据正被转换并最终加载到目的地时临时保持数据。存在在数据转换的进程期间涉及的许多传统操作，诸如过滤、分类、聚集、联接、清理、去重复和验证。

[0003] 对于医学数据应用，可能需要执行复杂的数据转换以生成有意义的洞察。然而，这些转换必须首先依赖于将数据构造成表型(即数据元素)的表，然后才能将它们用于更高级的处理。因此，在传统医学数据处理系统中，提取步骤涉及分离有用的表型。转换是指基于提取的表型执行的计算和推导。转换输出可以是从现有表型或决策规则的计算结果(诸如诊断或治疗选择)导出的新表型。

[0004] 然而，在现有ETL系统中，虽然自然语言处理(NLP)技术被广泛地用于表型提取，但是在转换阶段期间支持知识驱动的计算的架构非常有限。在ETL已经完成之后，通常实现单独的系统或应用来应付高级计算，这导致一致性的损失并引入更多的开销。

[0005] 数据提取传统上由独立的NLP模块来应付。因此，表型提取器通常与提取的数据的二次使用(诸如数据聚集)分开。这导致必须比较多个表型提取算法的性能，从而降低了效率。

[0006] 图结构传统上用于编码临床知识，因为它能够捕获医学概念之间的复杂关系，这有益于许多应用。然而，图表示本身是不可计算的。在常规的ETL流水线中，没有使得能够有益地使用知识来驱动计算的范式。这通常通过在数据转换期间的硬编码编程规则来实现。结果，该方法在应对变化(例如，输入数据类型、算法和决策逻辑的变化)方面缺乏灵活性。

发明内容

[0007] 本发明由权利要求定义。

[0008] 根据依照本发明的一个方面的示例，提供了一种ETL流水线系统，其包括被配置为获得多个图配置组件的接口。每个图配置组件包括表示一个或多个计算逻辑规则的信息。该系统还包括被配置为基于所获得的图配置组件来生成计算图的计算图生成器。所生成的计算图包括针对每个图配置组件的节点以及表示节点之间的关系的一个或多个链接。该系统还包括计算图适配器，该计算图适配器被配置为从外部源接收与图配置组件有关的外部信息，并且基于外部信息来适配所生成的计算图。该系统还包括被配置为运行所适配的计算图的计算图运行器。

[0009] 提出了针对ETL架构的概念，其可以对于医学数据处理特别有益，其中表型和临床

知识计算被一致地嵌入在单个架构中。这样的概念可以允许独立于实际临床数据来配置临床知识。然而，在运行时间期间，临床数据可以流过结构，其中将数据库操作、表型算法和决策逻辑应用于数据流，从而得到目标计算结果。

[0010] 提出了提供图配置文件(诸如DSL)以直观地构建计算图。还提出了提供图驱动器以解析、构建和执行计算图。还提出了提供信息建模技术以编码表型与临床文档之间的关系。所提出的后端图数据结构可以与基于事实的知识图和基于数据的知识图集成，以便有益于下游应用。另外，提出了将多层计算、局部重新计算和自适应计算的应用用于在ETL流水线系统中的应用。

[0011] 提出了针对ETL架构的概念，其可以对于医学数据处理特别有益。特别地，要求保护的发明提供了一种ETL流水线系统，其包括被配置为获得多个图配置组件的接口。每个图配置组件包括表示一个或多个计算逻辑规则的信息。所提出的发明基于所获得的图配置组件来生成计算图，其中所生成的计算图包括针对每个图配置组件的节点以及表示节点之间的关系的一个或多个链接。然后采用与图配置组件有关的外部信息(诸如用户定义的图配置组件和/或在先知识)来适配所生成的计算图。

[0012] 在实施例中，每个图配置组件可以包括数据库连接、文档、数据元素、算法和计算逻辑规则中的至少一者。

[0013] 在实施例中，算法可以包括计算决策规则，和/或计算逻辑规则可以包括计算决策规则。

[0014] 在实施例中，外部信息可以包括用户定义的图配置组件和来自外部知识数据库的外部知识中的至少一者。通过示例的方式，外部知识可以包括存储在数据库中的先前/在先计算图。这种先前/在先计算图可以被添加到所生成的计算图以形成所适配的计算图。即，所生成的计算图可以通过并入先前/在先计算图而被适配。通过另一示例的方式，用户定义的图配置组件可以包括被配置为重新定义或修改计算图的图配置组件。

[0015] 在实施例中，接口可以包括用户接口和应用编程接口中的至少一者。

[0016] 在实施例中，该系统还包括图配置文件，并且用户定义的图配置组件包括重新定义图配置文件或依赖性信息，依赖性信息指示不同图配置组件之间的依赖性关系。

[0017] 在实施例中，该系统还包括计算图解析器，该计算图解析器被配置为：从接口接收图配置组件；以及基于组件信息和指示不同图配置组件之间的依赖性关系的依赖性信息来在不同图配置组件之间建立一个或多个链接。

[0018] 在实施例中，计算图解析器还可以被配置为基于与数据元素和/或文档相关联的数据项来在数据元素与文档之间建立链接。

[0019] 在实施例中，数据项可以包括标识符、时间戳、内容项、上下文项和编码对象中的至少一者。

[0020] 在实施例中，计算图运行器还可以被配置为在自适应高吞吐量计算中运行所适配的计算图。

[0021] 根据依照本发明的一个方面的示例，提供了一种包括上述ETL流水线系统的医学数据处理系统。

[0022] 根据依照本发明的一个方面的示例，提供了一种用于ETL流水线处理的方法。该方法包括获得多个图配置组件，每个图配置组件包括表示一个或多个计算逻辑规则的信息。

该方法还包括基于所获得的图配置组件来生成计算图，所生成的计算图包括：针对每个图配置组件的节点；以及表示节点之间的关系的一个或多个链接。该方法还包括从外部源接收与图配置组件有关的外部信息。该方法然后包括基于外部信息来适配所生成的计算图，并且运行所适配的计算图。

[0023] 根据依照本发明的一个方面的示例，提供了一种用于ETL流水线处理的计算机程序产品，其中该计算机程序产品包括计算机可读存储介质，该计算机可读存储介质具有利用其体现的程序指令，该程序指令能够由处理单元执行以使处理单元执行一种方法。该方法包括获得多个图配置组件，每个图配置组件包括表示一个或多个计算逻辑规则的信息。该方法还包括基于所获得的图配置组件来生成计算图，所生成的计算图包括：针对每个图配置组件的节点；以及表示节点之间的关系的一个或多个链接。该方法还包括从外部源接收与图配置组件有关的外部信息。该方法然后包括基于外部信息来适配所生成的计算图，并且运行所适配的计算图。

[0024] 根据依照本发明的一个方面的示例，提供了一种处理系统，其包括至少一个处理器和上述计算机程序产品，其中该至少一个处理器适于执行所述计算机程序产品的计算机程序代码。

[0025] 本发明的这些和其它方面将从下文描述的实施例显而易见并参考下文描述的实施例得到阐述。

附图说明

[0026] 为了更好地理解本发明，并且为了更清楚地示出如何可以将本发明付诸实践，现在将仅通过示例的方式对附图进行参考，在附图中：

[0027] 图1是ETL流水线系统的简化框图；

[0028] 图2示出了ETL流水线系统中的状态转变流的简化框图；

[0029] 图3示出了不同图配置组件之间的依赖性关系；

[0030] 图4示出了所生成的计算图的一个示例；

[0031] 图5是根据一个实施例的用于ETL流水线处理的方法的简化流程图；

[0032] 图6示出了在ETL流水线系统中实现的知识计算结构(特别是自适应计算结构)的简化框图；

[0033] 图7示出了在ETL流水线系统中实现的知识计算结构(特别是多层计算机结构)的简化框图；以及

[0034] 图8图示了根据一个实施例的用于实现控制器或处理器的计算机的一个示例。

具体实施方式

[0035] 将参考附图描述本发明。

[0036] 应当理解，详细描述和特定示例虽然指示了装置、系统和方法的示例性实施例，但是仅旨在用于说明的目的而不旨在限制本发明的范围。本发明的装置、系统和方法的这些和其它特征、方面和优点将从以下描述、所附权利要求和附图中变得更好理解。应当理解，附图仅仅是示意性的并且不是按比例绘制的。还应当理解，在所有附图中使用相同的附图标记来指示相同或相似的部分。

[0037] 本发明提供了一种ETL流水线系统，其包括被配置为获得多个图配置组件的接口。每个图配置组件包括表示一个或多个计算逻辑规则的信息。该系统还包括被配置为基于所获得的图配置组件来生成计算图的计算图生成器。所生成的计算图包括针对每个图配置组件的节点以及表示节点之间的关系的一个或多个链接。该系统还包括计算图适配器，该计算图适配器被配置为从外部源接收与图配置组件有关的外部信息，并且基于该外部信息来适配所生成的计算图。该系统还包括被配置为运行所适配的计算图的计算图运行器。

[0038] 图1示出了根据一个实施例的ETL流水线系统。该系统包括用于用户与系统交互的接口11、12。这帮助用户根据其特定要求来创建(高级别)可计算知识表示。图1的系统包括两个接口11、12。接口12是用户接口(UI)，并且使得能够进行关键组件在系统前端处的“拖放”，从而得到更快的计算图创建。接口11是应用编程接口(API)，其服务于与用户接口12相同的目的，但是在编程级别处交互。API 11主要支持三个动作。第一，提交特别设计的图配置文件(GCF)13，图配置文件(GCF)13稍后将被转换为可计算图。第二，将关于数据库事务、数据计算和NLP表型的用户定义(UDF)算法14提交和保存/加载到中央功能储存库。第三，与共用图数据库15通信，这可以将所创建的图结构保存到图数据库15中或者将来自图数据库15的现有图加载到系统中。

[0039] 在上述组件之间存在状态转变流。(经由UI和API两者的)图配置被转化成有向无环图(DAG)的底层数据结构。DAG可以被来回转化为共用图数据库(例如Neo4j、GraphDB和Cayley)中的数据结构。在这一点上，DAG是编码所需信息以用于运行的静态图结构。它被进一步转化成可以在运行时执行的可运行图。

[0040] 图配置文件(GCF)13是用于执行数据提取和决策规则计算任务两者的图结构的容器。通过遵循我们特别设计的范式，允许用户声明或定义一组关键组件(即数据库连接、文档、表型、算法和计算逻辑)，并将它们组织成GCF 13的群组，以履行某些计算任务。GCF 13可以被创建为JSON和YAML格式的人类可读文本文件。在每个文件中，用户定义包括表示图中的单个节点的关键组件的结构。允许在相同文件内定义多个和嵌套节点，或将多个和嵌套节点分布到相同文件夹中的若干文件中。在解析期间，相同文件中或相同文件夹中的外部文件中的先前定义的GC节点可以通过名称来引用。

[0041] GCF 13被配置为在编程级别处将现有知识吸收到由我们的系统支持的共用图结构中。这有时对于人类经由UI操办来说是困难的和/或耗时的。比如，BCLC和Child-Pugh评分系统可以经由GCF被持续在我们的系统中，并且在运行时被计算，以导出针对肝癌患者的Child-Pugh评分的分级。

[0042] 图配置UI 12用作GCF 13的替代，以允许直观地构建临床知识和决策规则的图表示。UI适配器16负责将前端输入转换为底层数据结构(DAG)。在UI中，允许用户通过创建节点和链接来构建图。对于每个节点，用户可以指定算法以处理通过该节点行进的数据流或实现从当前节点的父节点取得输入的决策逻辑。

[0043] 图驱动器10位于系统的核心。它负责构建能够取得数据流的计算图，并使该图运行以向用户递送期望的结果。图驱动器10包括三个主要组件：GCF解析器(GCFP)17、图构建器(GST)18、图运行器(GRN)19和两个适配器16、20。图驱动器的实现不限于特定的编程语言；然而，开发者遵循相同的理念以及每个系统组件中的共用接口，以实现语言特定的驱动器。

[0044] 以配置文件13作为输入(格式为JSON、YAML或XML),GCFP 17将文件13中的图定义转换成在对应名称空间下的个体组件(例如,数据库连接、文档、表型、算法和计算逻辑),然后将名称空间链接在一起作为链接对象以准备图构建。

[0045] 更具体地,每个GCF 13可以包括针对不同图配置组件的定义。通过示例的方式,图3定义了与各种名称空间组件相关的不同图配置组件之间的依赖性关系的示例性se。更具体地,标记为“A”的实线定义了数据库与文档之间的依赖性,标记为“B”的实线定义了算法与逻辑之间的依赖性。标记为“C”的实线定义了文档与逻辑之间的依赖性。标记为“D”的实线定义了表型与逻辑之间的依赖性。这种依赖性关系可以由用户定义。

[0046] GCFP 17从接口11接收图配置组件,然后基于组件信息和指示不同图配置组件之间的依赖性关系的依赖性信息,在不同图配置组件之间建立链接。

[0047] GST 18然后可以将链接的组件转译成图节点和关系。也就是说,GST 18取得链接的名称空间对象并且构建底层数据表示,其中链接的组件被转译成DAG形式的图节点和节点之间的关系。另外,在构建进程期间,GST 18频繁地与图配置API交互,以从/向功能储存库(保持所有功能对象的数据库)加载/保存组件。如果GCF 13中的指定组件已经存在于该储存库中,则直接从储存库取得组件引用。否则,所指定的组件是新的,并且GST 18然后将其保存到功能储存库以供再使用。

[0048] GRN 19首先用作注入器,该注入器注入运行图的所有所需信息,例如代码对象和递归地定义的GC组件。同时,图运行器19是使图去向基于图的并行执行系统的现有实现的连接器。作为一个示例,图4示出了所生成的计算图的一个示例。所生成的计算图可以经由接口11或12输出。所生成的计算图包括针对每个图配置组件的节点36以及表示节点37之间的关系的一个或多个链接37。

[0049] 两个适配器(即图数据库适配器20和UI适配器16)被包括在驱动器10中。UI适配器负责将前端输入转化成DAG结构,反之亦然,从而向前端用户呈现所构建的DAG。而且,图DB适配器协调由我们的系统和现有图DB系统使用的数据结构。

[0050] 参考图2,描绘了根据一个实施例的系统组件之间的状态转变流的图示。根据GCF 13和经由用户接口提供的图配置33,构建DAG 32。DAG 32可以被来回转化为知识图34中的数据结构。DAG 32还被转化成可运行图35,可运行图35可以在运行时被执行。

[0051] 现在参考图5,描绘了根据一个实施例的用于ETL流水线处理的方法的简化流程图。该方法从获得多个图配置组件的步骤21开始。这里,每个图配置组件包括表示一个或多个计算逻辑规则的信息。接下来,在步骤22中,基于所获得的图配置组件来生成计算图。所生成的计算图包括:针对每个图配置组件的节点;以及表示节点之间的关系的一个或多个链接。随后,在步骤23中,从外部源接收与图配置组件有关的外部信息。然后,步骤24包括基于外部信息来适配所生成的计算图。最后,在步骤25中,运行所适配的计算图。

[0052] 更具体地,外部信息包括用户定义的图配置组件和来自外部知识数据库的外部知识中的至少一者。在一个实施例中,外部知识包括存储在数据库中的先前计算图,并且该先前计算图被添加到所生成的计算图并且形成所适配的计算图。在另一实施例中,用户定义的图配置组件包括经由接口重新定义或修改图配置组件。

[0053] 受试者可以使用图形用户界面来重新定义所生成的计算图的图配置组件或图配置组件之间的链接。比如,这种重新定义可以通过在(例如,如图4所示的)计算图的交互式

视觉表示中拖动节点、移除链接或重新连接不同节点之间的链接来实现。

[0054] 在另一实施例中，用户定义的图配置组件包括用于重新定义图配置文件和/或不同图配置组件之间的依赖性关系的一个或多个定义。更具体地，受试者可以定义不同的逻辑(通过改变@logic)，定义不同的算法(通过改变@algorithm)，或添加新的组件，即受试者依赖性。

[0055] 通过进一步解释的方式，我们将首先描述将配置文件13转化成链接的名称空间组件的GCF语法和GCF解析。为了解析，我们引入每个构建块。对于图构建，我们讨论形成DAG的细节。然后将详细解释数据模型设计，其在图构建进程期间处置表型与文档之间的关系。然后将解释如何可以将现有知识图与所提出的实施例组合以获取有意义的结果。最后，将提供关于如何可以将实施例用于高效计算的细节。

[0056] • GCF语法

[0057] GCF语法支持关键组件的声明和定义，关键组件以“@”符号开始，其后是组件类别。关键组件分为五个不同类别，即：数据库(database)；表型(phenotype)；文档(document)；算法(algorithm)；和逻辑(logic)。每个服务于不同的目的并且可以被组织成独立的结构。下面在说明书的下一部分中提供关于这五个组件的详细讨论。声明与定义之间的差异取决于组件是否已经被创建。为了声明组件，在“@”之后仅需要跟着组件名称。

[0058] 为了定义组件，“@def”关键字必须放在组件的主体内。为了保存已定义的组件，必须使用关键字“@name”来指定名称，并且应当向“@save”关键字添加设置为“真(True)”的值。用户将功能实现作为值附加到“@def”，并且功能定义将被持续到中央储存库中。然后，可以如上所述声明保存的组件。GCF语法支持组件的嵌套定义，但受名称空间依赖性的约束。

[0059] 以下示例示出了在给定放射学报告数据集群和肿瘤生物标志物的情况下检测恶性肿瘤的任务。

[0060] 履行检测恶性肿瘤的任务的示例性GCF：

```
"@Document" : input_data
"@Document" : tumor_markers
"@Phenotype#1" : tumor_mention
"@Phenotype#2" : report_time
"@Phenotype#3" : AFP
"@Algorithm" : is_MT
"@Algorithm" : select_marker
"@Logic" :
    "@name" : detect_radiology_reports,
[0061]  "@def" : |
      for cluster, _in input_data:
        evidence_list = [
          x for x in cluster
          if is_MT(x.tumor_mention)
        ]
        if len(evidence_list) >=2:
          yield (True, cluster[0].report_time)
        elif len(evidence_list) ==1:
          markers=select_marker(tumor_markers)
      if markers.AFP > 20:
        yield (True, cluster[0].report_time)
[0062]  else:
          yield (False, None)
"@save" : True
```

[0063] 目标是辨别在每个放射学报告集群内部是否发现恶性肿瘤(MT)。决策规则是,如果一个集群中的多于两个放射学报告包含恶性肿瘤提及,则返回真。在另一种情况下(小于2个放射学支持),如果发现异常AFP(肿瘤生物标志物),则也返回真。否则,返回假。在上述示例中,"@Document:input_data"表示放射学报告集群列表(两个月内的报告集群在一起)。"@Document:tumor_markers"将肿瘤生物标志物信息注入文档。利用导入的文档,根据我们的文档-表型关系模型,"@Phenotype:tumor_mention"和"@Phenotype:report_time"将从对应的文档对象中被取回。"@Algorithm:is_MT"和"@Algorithm:select_marker"将触

发在图构建期间从储存库导入两个决策功能。“select_marker”是选择记录到集群内部的那些放射学报告的最新近生物标志物数据的算法。“is_MT”是根据“@phenotype:tumor_mention”确定肿瘤是否是恶性的算法。决策规则在“@Logic”块的“@def”内实现。“@save”指示该逻辑将以名称“detect_radiology_reports”被保存到功能储存库。下一次,可以直接经由声明“@Algorithm:detect_radiology_reports”导入该保存的功能。然而,到目前为止定义的所有导入和代码将不会被执行,直到图被构建并且转化为可运行图并且正被执行。

[0064] • 名称空间组件

[0065] 对应于由GCF语法定义的五个关键项,有五个功能组件对象。数据库组件负责应付数据库操作、数据库连接和诸如数据查询的事务。表型是对某些决策规则计算任务有意义的键-值实体。文档表示包含一组表型的临床报告。算法是处理输入以获取输出的特殊功能。比如,数据提取算法从临床叙述数据中提取相关表型值。逻辑也是类似算法的功能。然而,不同之处在于逻辑负责在不同节点之间进行转变。

[0066] 本发明人已经创建了双向关系模型来表示表型和文档。每个表型和文档包含名称/ID、时间戳、内容的强制性数据项以及上下文和编码的可选数据项。然而,内容和上下文对于表型和文档分别具有不同的意义。对于文档,内容以叙述形式或结构化/半结构化形式表示报告的内容。对于表型,内容是表型值。对于表型,上下文是包含它的文档的内容。对于时间戳,假设表型时间戳等于文档时间戳,但是在某些情况下并非如此。编码表示相关联的标准化代码。通常,该代码指示表型或文档实体的类别。

[0067] • DAG

[0068] 以上面提供的示例性GCF为例,通过NLP表型算法取得表型,NLP表型算法使用上面针对表型和文档讨论的关系信息模型对表型结果建模。在上述示例性GCF中,未明确声明NLP算法。假设这通过一些预配置来完成。在该示例中,明确地示出了用NLP算法从文档中提取表型作为DAG中的功能节点。再次,该DAG表示是静态的,并且直到将该图变成可运行图并执行时才进行计算。

[0069] • 与知识/本体的连接

[0070] 这里,详细描述在外部知识图数据库中的图数据与图结构之间的连接。可以连接两种类型的知识图数据库,即具有事实的知识库和具有数据的知识库。通过将图中的实体与外部实体链接,与知识的连接具有将图处理结果扩大到更大范围的潜力。这在构筑智能搜索或问答系统(QA)时是有用的。

[0071] -具有事实的知识库

[0072] 事实图是现有的知识图和本体,诸如SNOMED-CT、LOINC、MeSH、Drugbank等。通过将我们的DAG中的数据节点连接到这样的外部图和本体,关键的益处是可以执行概念标准化,并且可以桥接内部图与外部图以自动实现知识扩充。这便于开发表现为理解生物医学和健康语言的含义的处理系统。

[0073] -具有数据的知识库

[0074] 快速健康互操作性资源(FHIR)是定义一组数据资源的医学标准,因此其目的是便于医学系统之间的快速数据交换。FHIR数据库本质上是具有可以被链接在一起作为图的不同数据资源的知识库。FHIR支持Turtle格式,其可以用于将数据存储为RDF三元组。

[0075] • 知识计算

[0076] 现在将提供知识计算策略的细节。所提出的实施例可以采用以下各项：(i)根据图中的两个节点之间的数据依赖性来自适应地选择最佳计算策略以提高效率；(ii)将累积的数据和知识组织成多层结构，该多层结构计算起来高效并且更易于用户跟踪；以及(iii)允许可替换图组件到现有计算图中以高效地优化系统性能的局部重新计算。

[0077] -自适应计算结构

[0078] 可运行图被馈送到并行执行组件中。该组件的关键概念是将树状顺序计算转换为高吞吐量同步计算和序列检测以及后续决策制定。在生成了DAG之后，如何高效地处理它仍然是一个挑战，特别是对于复杂的DAG和大规模数据。提出了将顺序决策图计算问题转换为节点状态检测和节点状态序列映射的计算策略。在图6中提供了该概念的图示。

[0079] 具体地，图6图示了根据一个实施例的所提出的自适应计算策略。这里，顺序图计算被转换为高吞吐量同步状态检测和状态序列映射。在“A”中，DAG中的所有节点42都没有数据传输，那么每个节点42将被转换为状态序列43中的状态；在“B”中，在一些节点42之间存在数据传输，那些节点42被转换为状态序列43中的一个状态。

[0080] 首先，根据定义的GCF，生成状态序列列表44，其列举所有可能的状态序列43及其对应的结果。状态序列43与结果之间的关系可以是多对一。

[0081] 如上面(在标题为“DAG”的部分中)详述的，在转换进程中采用自适应策略来提高效率。在第一种情况“A”中，在两个节点之间没有中间数据被传输(即，节点之间没有依赖性)。因此，它可以被认为是独立的决策节点，并且被转换为状态序列中的单个状态。在第二种情况“B”中，其中在DAG中的两个节点之间存在数据传输(即，在节点之间存在依赖性)，那些节点被合并为节点群组，然后被转换为序列中的状态(如由围绕被合并的节点的虚线框所示)。

[0082] 在DAG计算的进程期间，存在有助于效率提高的两个关键方面：(a)并行节点状态检测，而不是顺序决策来导出每个节点状态；以及(b)状态序列映射。当大规模数据被馈送到图中时，将针对每个决策节点或决策节点群组生成决策状态。此后，所有状态将以预定义次序被连结到状态序列。然后，系统针对序列列表进行序列映射以得到结果。比如，在进行决策制定的临床实践中，GCF定义三个节点(A,B,C)，并且每个节点是二元选择(0表示假，1表示真)，状态序列“100”表示A是真、B是假并且C是假，结果是“有疾病”，同时，“001”的结果可以是“没有疾病”。所有那些序列形成状态序列的列表。

[0083] -多层计算

[0084] 多层计算处置被注入的临床知识图的数量和数据两者都在增加时的情况，这是通常的情况。纵向数据(表型)随时间变化，其中新的数据保持到来，同时，新的临床知识被连续注入到系统中。利用更多的数据，可以继续所存储的图中的一些计算。利用更完整的知识，可以激活使用系统中的现有数据的新计算以导出新数据。比如，假设存在被注入的知识图，它监测患者是否有复发性肿瘤。当只有一个放射学报告支持癌症复发时，根据临床知识，不能做出判断。然而，当有一个附加放射学支持或肿瘤生物标志物支持进入系统中时，可以进行计算以导出关于肿瘤复发性的新数据变量。如果积累的数据和知识未被很好地组织，则用户将永远不能跟踪每个计算结果的进程。

[0085] 所提出的系统周期性地重新运行ETL以从EMR中获取新数据，并执行所描述的计算。这里，在初始ETL期间创建的数据变量被称为基本数据层。新导出的数据被递归地放置

到基本层顶部的新数据层中。特别地，在注入新的知识图之后，图中的每个节点经由概念编码被映射到标准化独特系统代码。这种编码用于构筑所有图节点的倒排索引，每个图节点与包含该节点以及层号的图结构相关联。基于其数据类型，新的输入数据经由相同的编码进程被分配有概念代码。利用倒排索引搜索概念代码，以得到包含具有相同编码的节点的所有图。这是非常快速的查找进程。通常，节点编码可以被已经注入的图覆盖，数据作为图节点被添加到已匹配序列以用于进一步处理。然而，对于未被覆盖的那些，将其作为节点添加到未匹配序列。对于已匹配序列内部的节点，在导出新数据时将它们添加到系统和活动图计算中。该算法被描述如下。

[0086] 新节点被复制并添加到索引中存在包括该节点的图结构的所有那些层。如果所添加的节点可以与相同层中的其他节点一起工作以导出一些图中的新数据值，则创建新层以包括该新导出的数据节点，或者将该新节点添加到现有的上层。为了决策，算法再次仅在包含新导出节点的上层中与所有图匹配。如果存在包含该节点的上层，则将该节点添加到该层。如果不是，则创建与其它上层并行的新上层。在这样的新层中，包括新导出节点的所有图结构将与新层号相关联。

[0087] 最初，在底部只有称为基本层的一个层。对于未匹配序列中的节点，当前系统中没有覆盖它们的图结构。这些数据节点可以或决不被将来注入到系统中的新知识图覆盖。因此，当注入新知识时，算法总是检查是否可以添加来自未匹配序列的节点。如果不是，则这些数据节点仅服务于我们的系统中的正常数据点。

[0088] 通过进一步描述的方式，在图7中图示了一个示例。更具体地，图7图示了根据所提出的实施例的多层计算和数据组织的一个示例。第一基本层53的A和B可以导出层1.1群组55的E。E和F进一步导出层2.1群组57的H。然而，C和D不被A和B的群组53中的任何图覆盖。因此，创建第二并行基本层54，以包括C和D。C和D导出层1.2群组56的G。而且，因为G未被层1.1中的E和F中的图覆盖，并且由层1.2群组56和层2.1群组57两者所包括，所以G被复制到层56、57两者。

[0089] 多层计算的益处是它基于快速图查找和并行图计算。因此，它整体上非常高效。此外，(通过信息的分层组织)可以清楚地向用户示出何时数据被添加以及这些数据在导出新数据项时如何通过宝贵的ETL与现有数据交互。所提出的这种技术的效果是，在相同层中，对新节点有贡献的旧节点趋向于集群在一起，而且共享相同节点的图结构趋向于集群在一起。这可以用于设计分层信息管理系统，其提供相关数据和知识的良好归档。由此，用户不必遍历离散知识图来定位其感兴趣的每个节点。用户可以容易地检查什么数据元素被遗漏，以便导出更高级的数据节点(在上层中)。

[0090] -局部重新计算

[0091] 用户可以重新配置图中的部分并指定要重新计算的节点。例如，用户可能想要尝试领先分类算法，该领先分类算法标识放射学报告是否已经指示复发性肿瘤。用户替换图中的旧算法节点并重新运行该图。然而，重新运行将不影响其它节点，仅影响相关节点。类似地，用户可以替换图中的任何功能节点。比如，用户重新定义来自数据库的临床文档的查询事务，使其更快。所提出的系统将重新计算流传播到与该事务相关的所有下游节点。

[0092] 本发明的应用

[0093] 所提出的发明的主要应用是信息提取和ETL。所提出的架构允许提取步骤和转换

步骤变得可容易配置并且更加一致，从而减少开销。另一应用类别属于用于决策支持的知识计算和管理。尽管上面已经基于基于规则的决策制定示出了示例，但是每个功能节点是可替换的，使得可以在图结构内插入并验证自动算法。因为所提出的技术将医学数据库与计算知识和外部知识库连接，所以它们可以有助于搜索和QA应用，例如医学搜索引擎或聊天机器人。而且，因为所提出的系统处置逐渐增加的数据，所以它可以对于事件监测是有用的，事件监测例如为复发早期检测、药物不良反应等。

[0094] 图8图示了用于实现上述系统的实施例的计算机60的一个示例。

[0095] 计算机60包括但不限于PC、工作站、膝上型计算机、PDA、掌上设备、服务器、存储设备等。通常，就硬件架构而言，计算机60可以包括经由本地接口(未示出)通信耦合的一个或多个处理器61、存储器62以及一个或多个I/O设备63。如本领域已知的，本地接口可以是例如但不限于一个或多个总线或其他有线或无线连接。本地接口可以具有附加元件以使得能够进行通信，该附加元件诸如为控制器、缓冲器(高速缓存)、驱动器、中继器和接收器。此外，本地接口可以包括地址、控制和/或数据连接，以使得能够在前述组件之间进行适当通信。

[0096] 处理器61是用于执行可以被存储在存储器62中的软件的硬件设备。处理器61实际上可以是任何定制的或商业上可用的处理器、中央处理单元(CPU)、数字信号处理器(DSP)、或与计算机60相关联的若干处理器当中的辅助处理器，并且处理器61可以是基于半导体的微处理器(以微芯片的形式)或微处理器。

[0097] 存储器62可以包括以下项中的任何一项或以下项的组合：易失性存储器元件(例如，随机存取存储器(RAM)，诸如动态随机存取存储器(DRAM)、静态随机存取存储器(SRAM)等)和非易失性存储器元件(例如，ROM、可擦除可编程只读存储器(EPROM)、电可擦除可编程只读存储器(EEPROM)、可编程只读存储器(PROM)、磁带、紧凑盘只读存储器(CD-ROM)、碟、软盘、卡盘、盒式磁带等)。此外，存储器62可以包含电子、磁、光和/或其它类型的存储介质。注意，存储器62可以具有分布式架构，其中各种组件彼此远离，但是可以由处理器61访问。

[0098] 存储器62中的软件可以包括一个或多个单独的程序，其中的每个程序包括用于实现逻辑功能的可执行指令的有序列表。根据示例性实施例，存储器62中的软件包括适当的操作系统(O/S)64、编译器65、源代码66和一个或多个应用67。

[0099] 应用67包括许多功能组件，诸如计算单元、逻辑、功能单元、进程、操作、虚拟实体和/或模块。

[0100] 操作系统64控制计算机程序的执行，并且提供调度、输入-输出控制、文件和数据管理、存储器管理以及通信控制和相关服务。

[0101] 应用67可以是源程序、可执行程序(目标代码)、脚本或包括要执行的指令集的任何其它实体。当为源程序时，那么通常经由可以被包括或可以不被包括在存储器62内的编译器(诸如编译器65)、汇编器、解释器等来转译该程序，以便结合操作系统64恰当地操作。此外，应用67可以被写为：面向对象编程语言，其具有数据和方法的类；或过程编程语言，其具有例程、子例程和/或函数，例如但不限于C、C++、C#、Pascal、BASIC、API调用、HTML、XHTML、XML、ASP脚本、JavaScript、FORTRAN、COBOL、Perl、Java、ADA、.NET等。

[0102] I/O设备63可以包括输入设备，诸如(例如但不限于)鼠标、键盘、扫描仪、麦克风、相机等。此外，I/O设备63还可以包括输出设备，例如但不限于打印机、显示器等。最后，I/O

设备63还可以包括传送输入和输出两者的设备，比如但不限于网络接口控制器(NIC)或调制器/解调器(用于访问远程设备、其它文件、设备、系统或网络)、射频(RF)或其它收发器、电话接口、桥接器、路由器等。I/O设备63还包括用于通过诸如因特网或内联网的各种网络进行通信的组件。

[0103] 当计算机60在操作中时，处理器61被配置为执行存储在存储器62内的软件，将数据传送到存储器62和从存储器62传送数据，并且通常根据该软件来控制计算机60的操作。应用67和操作系统64整体或部分地由处理器61读取，可能缓冲在处理器61内，然后被执行。

[0104] 当应用67以软件实现时，应当注意，应用67可以被存储在几乎任何计算机可读介质上，以供任何计算机相关的系统或方法使用或与任何计算机相关的系统或方法结合使用。在本文档的上下文中，计算机可读介质可以是电子、磁、光或其它物理设备或装置，其可以包含或存储计算机程序，以供计算机相关的系统或方法使用或与计算机相关的系统或方法结合使用。

[0105] 通过研究附图、公开内容和所附权利要求，本领域技术人员在实践要求保护的本发明时可以理解和实现所公开的实施例的其它变型。在权利要求中，词语“包括”不排除其他元件或步骤，并且不定冠词“一”或“一个”不排除多个。在相互不同的从属权利要求中记载某些措施的仅有事实并不指示不能有利地使用这些措施的组合。权利要求中的任何附图标记不应被解释为对范围的限制。

[0106] 通过研究附图、公开内容和所附权利要求，本领域技术人员在实践要求保护的本发明时可以理解和实现所公开实施例的变型。在权利要求中，词语“包括”不排除其他元件或步骤，并且不定冠词“一”或“一个”不排除多个。单个处理器或其它单元可以履行权利要求中记载的若干项的功能。在相互不同的从属权利要求中记载某些措施的仅有事实并不指示不能有利地使用这些措施的组合。如果上面讨论了计算机程序，则它可以被存储/分布在适当的介质上，诸如与其他硬件一起供应或作为其他硬件的一部分供应的光学存储介质或固态介质，但是它也可以以其他形式分布，诸如经由因特网或其他有线或无线电信系统。如果在权利要求书或说明书中使用术语“适于”，则注意，术语“适于”旨在等同于术语“被配置为”。权利要求中的任何附图标记不应被解释为对范围的限制。

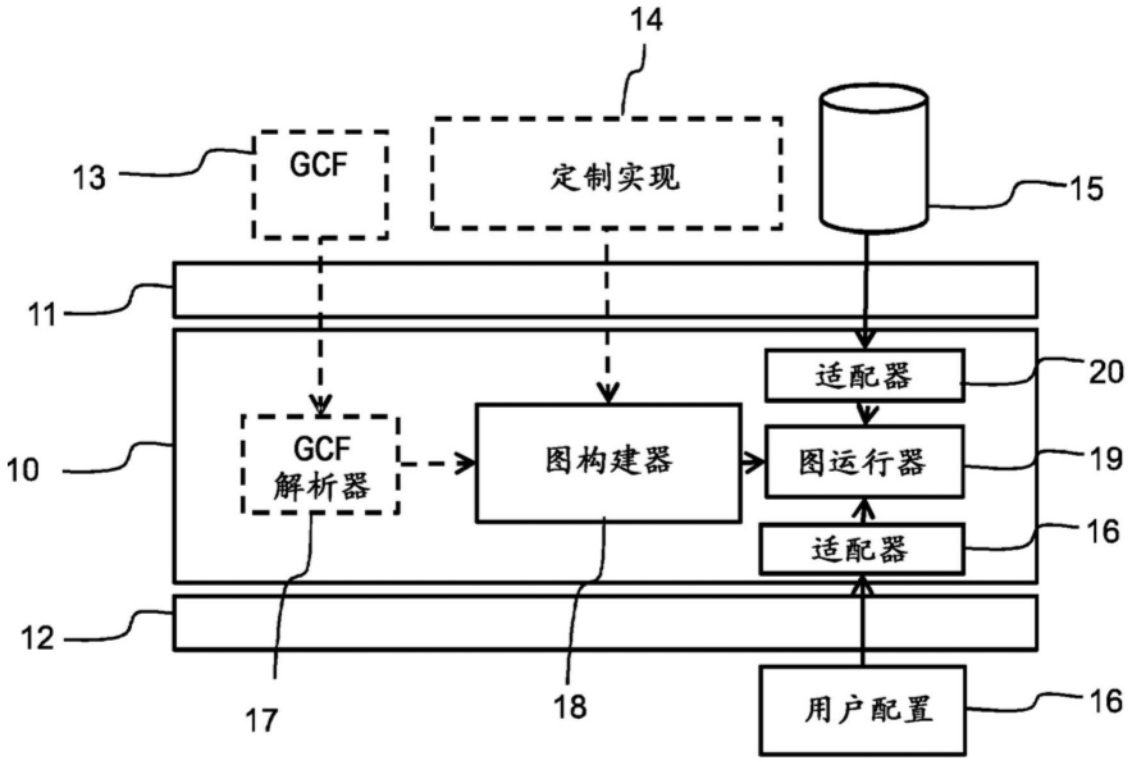

14
13
GCF
定制实现
15
11
适配器
20
10
GCF
解析器
图构建器
图运行器
19
适配器
16
12
17
18
用户配置
16

图1

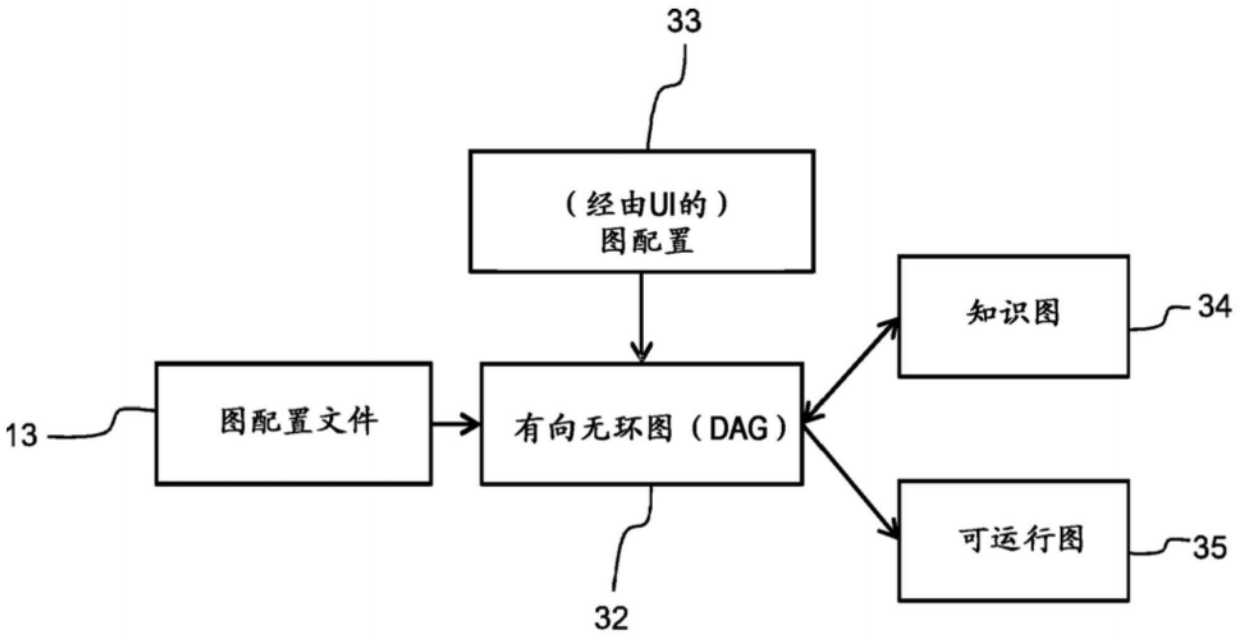

33
（经由UI的）
图配置
知识图
34
13
图配置文件
有向无环图（DAG）
可运行图
35
32

图2

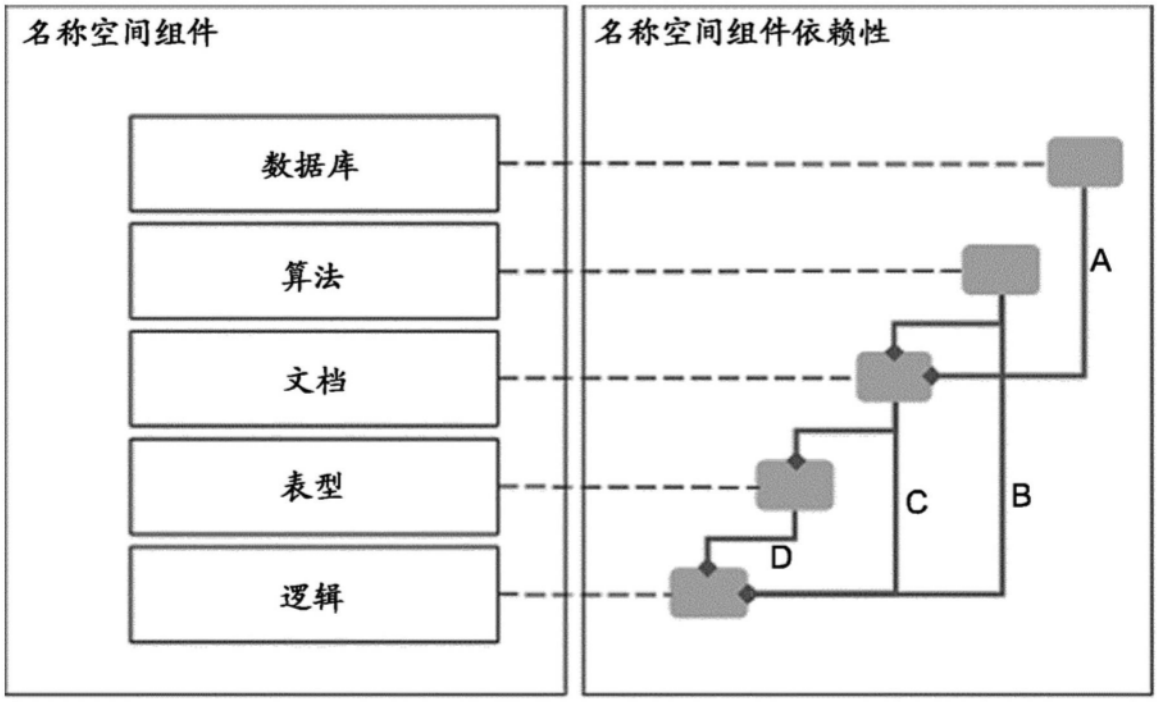

名称空间组件
名称空间组件依赖性
数据库
算法
文档
表型
逻辑
A
B
C
D

图3

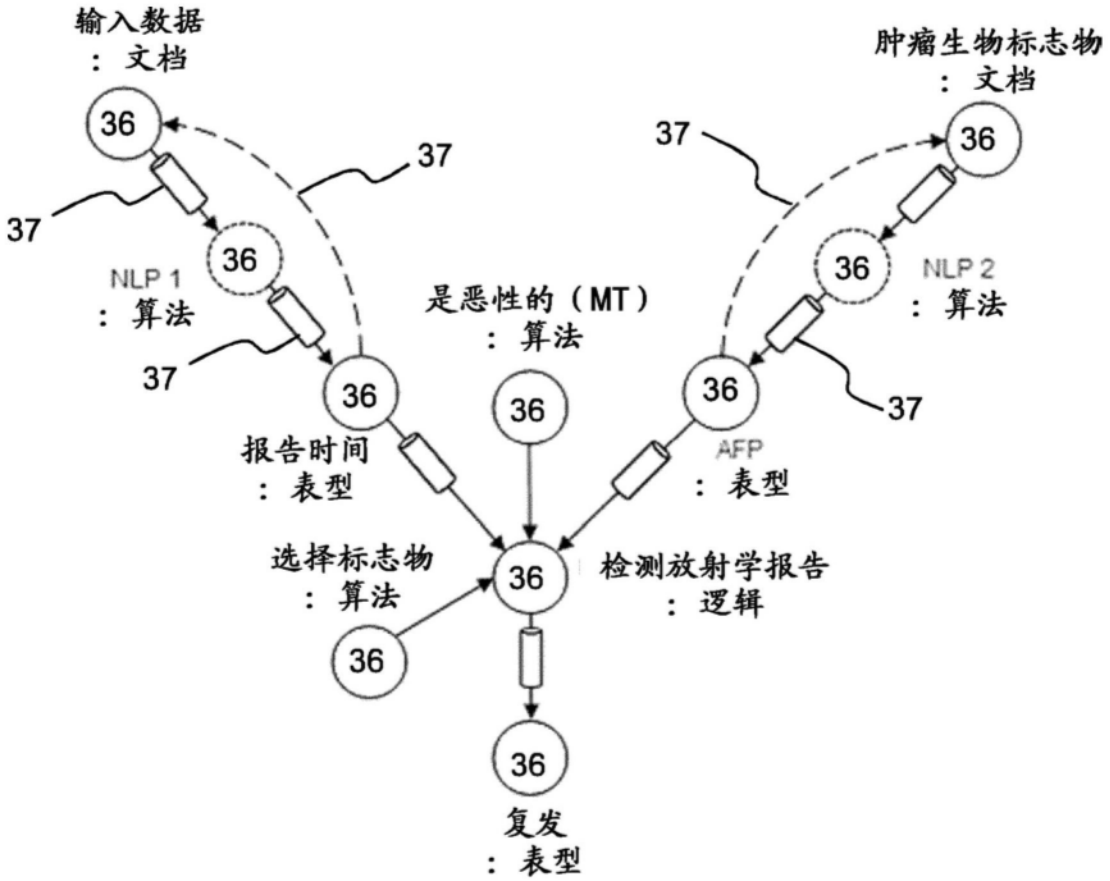

输入数据
：文档
肿瘤生物标志物
：文档
NLP 1
：算法
NLP 2
：算法
是恶性的（MT）
：算法
报告时间
：表型
AFP
：表型
选择标志物
：算法
检测放射学报告
：逻辑
复发
：表型
36
37

图4

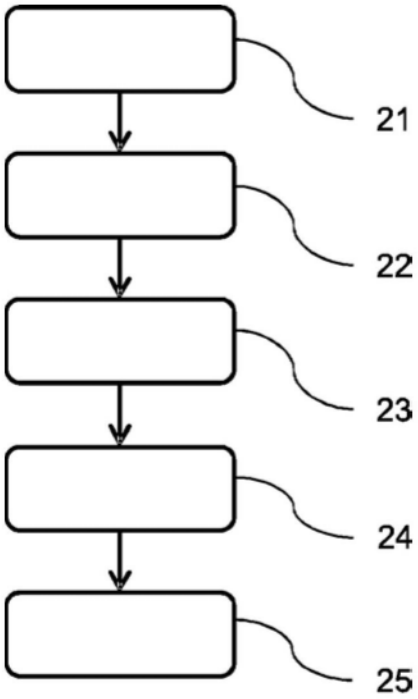


图5

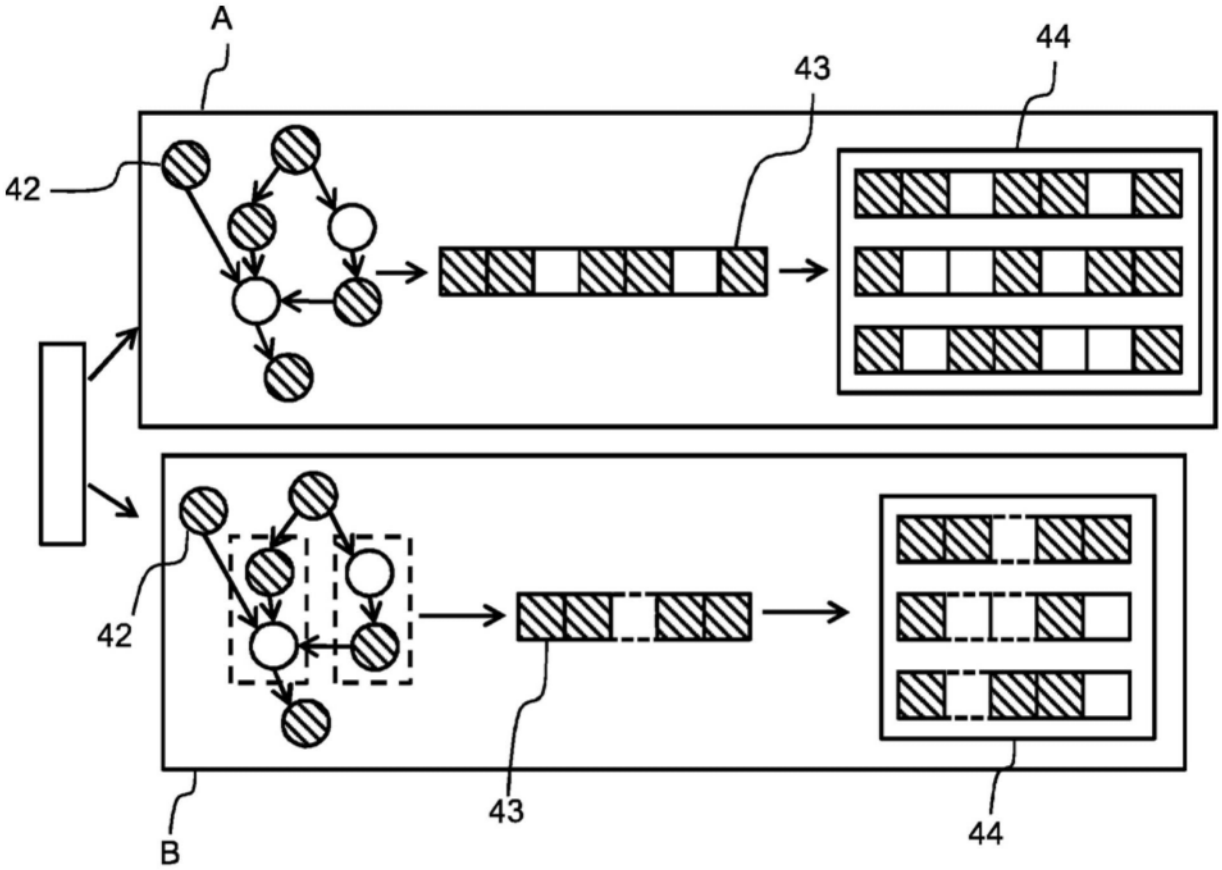


图6

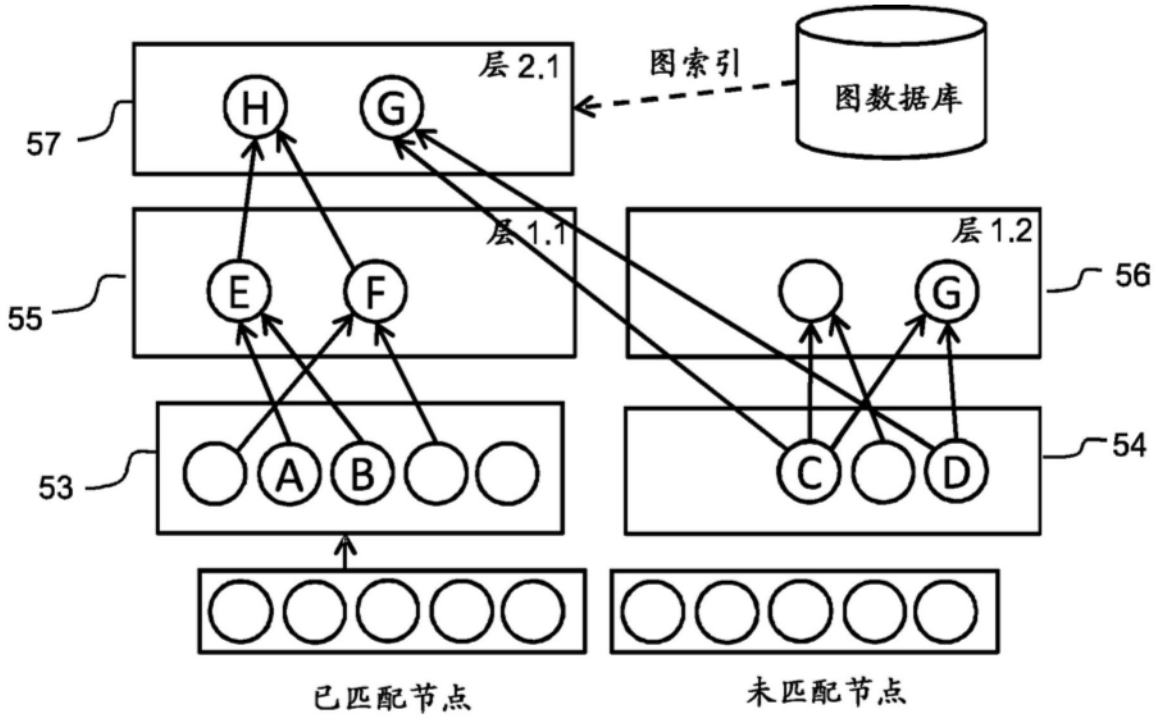

层 2.1
图索引
图数据库
H
G
57
层 1.1
层 1.2
56
E
F
55
G
54
53
A
B
C
D
已匹配节点
未匹配节点

图7

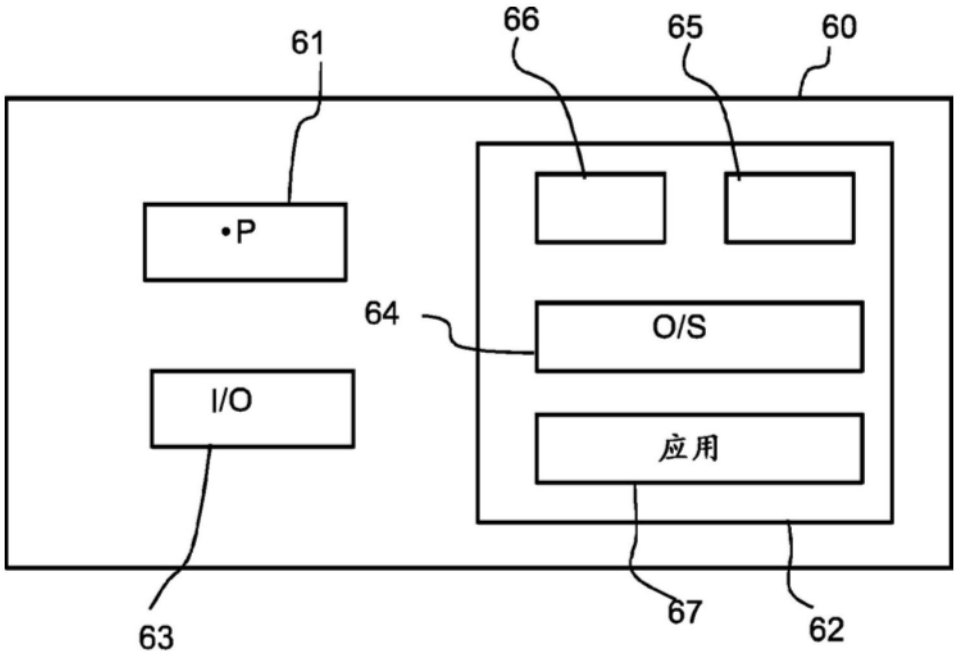

61
66
65
60
•P
64
O/S
I/O
应用
63
67
62

图8