(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 115292160 B
(45)授权公告日 2024.06.28

(21)申请号 202210769407.5
(22)申请日 2015.11.05
(65)同一申请的已公布的文献号
申请公布号 CN 115292160 A
(43)申请公布日 2022.11.04
(30)优先权数据
62/075,451 2014.11.05 US
14/715,807 2015.05.19 US
(62)分案原申请数据
201580072406.3 2015.11.05
(73)专利权人 起元技术有限责任公司
地址 美国马萨诸塞州
(72)发明人 马歇尔·A·伊斯曼
约翰·乔伊斯
(74)专利代理机构 北京林达刘知识产权代理事务所(普通合伙) 11277
专利代理师 刘新宇
(51)Int.Cl.
G06F 11/36(2006.01)
G06F 8/34(2018.01)
G06F 8/35(2018.01)
(56)对比文件
CN 102902529 A,2013.01.30
审查员 王伟

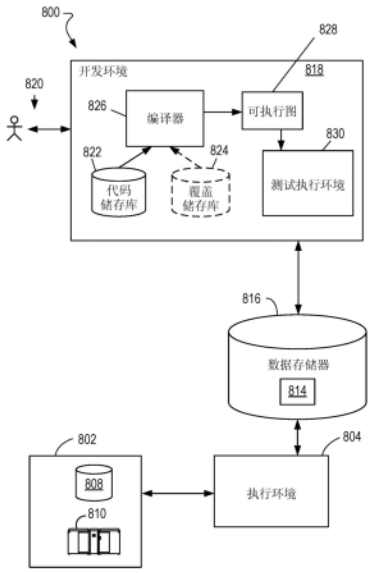
权利要求书4页 说明书16页 附图12页

(54)发明名称

应用程序测试

(57)摘要

本申请涉及应用程序测试。一种方法包括：分析第一版本的计算机程序，包括识别第一版本中的第一数据处理元素，第一数据处理元素引用表示数据记录的源或目的地的第一数据记录元素；识别数据记录的存储，其不同于数据记录的源或目的地；生成表示数据记录的存储的第二数据记录元素，包括定义针对第二数据记录元素的规范，规范包括表示数据记录的存储的位置的信息和与第一数据处理元素相关联的标识符；以及基于第一版本和规范来生成第二版本的计算机程序，第二版本包括所生成的第二数据记录元素和基于第一数据处理元素的第二数据处理元素，其中第一版本未被第二版本的生成修改，在第二版本中第二数据处理元素引用所生成的第二数据记录元素。



CN 115292160 B

1.一种方法,包括:

分析第一版本的计算机程序,所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素,所述第一数据处理元素引用表示所述第一版本的计算机程序外部的数据记录的源或目的地的第一数据记录元素;

识别数据记录的存储,所识别的存储不同于所述数据记录的源或目的地;

生成表示所识别的数据记录的存储的第二数据记录元素,所述生成包括定义针对所述第二数据记录元素的规范,所述规范包括表示所识别的数据记录的存储的位置的信息和表示由所述第一数据处理元素引用的所述第一数据记录元素的位置的信息;以及

基于所述第一版本的计算机程序和针对所述第二数据记录元素的规范来生成第二版本的计算机程序,所述第二版本的计算机程序包括所生成的第二数据记录元素和基于所述第一数据处理元素的第二数据处理元素,其中所述第一版本的计算机程序未被所述第二版本的计算机程序的生成修改,

其中,在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的第二数据记录元素。

2.根据权利要求1所述的方法,其中,识别所述数据记录的存储包括:确定所述数据记录的存储的位置。

3.根据权利要求2所述的方法,其中,确定所述数据记录的存储的位置包括:接收表示所述位置的输入。

4.根据权利要求1所述的方法,还包括:在用户接口中呈现所述第一数据处理元素、所述第一数据记录元素或这两者的标识符。

5.根据权利要求4所述的方法,还包括:接收对所述第一数据处理元素、所述第一数据记录元素或这两者的选择。

6.根据权利要求1所述的方法,其中,分析所述第一版本的计算机程序包括:分析向所述第一版本的计算机程序中所包括的一个或多个数据处理元素的输入流。

7.根据权利要求1所述的方法,其中,分析所述第一版本的计算机程序包括:分析来自所述第一版本的计算机程序中所包括的一个或多个数据处理元素的输出流。

8.根据权利要求1所述的方法,其中,所述计算机程序包括图。

9.根据权利要求8所述的方法,其中,生成所述第二版本的计算机程序包括:当所述第一数据记录元素表示数据记录的源时,使所生成的第二数据记录元素位于向所述第二数据处理元素的输入流处。

10.根据权利要求8所述的方法,其中,生成所述第二版本的计算机程序包括:当所述第一数据记录元素表示数据记录的目的地时,使所生成的第二数据记录元素位于来自所述第二数据处理元素的输出流处。

11.根据权利要求1所述的方法,还包括:利用基于来自所述第一数据记录元素的数据的数据来填充所述第二数据记录元素。

12.根据权利要求1所述的方法,还包括:执行所述第二版本的计算机程序。

13.根据权利要求1所述的方法,其中,生成所述第二版本的计算机程序包括:生成所述第一版本的计算机程序的至少一部分的副本。

14.根据权利要求13所述的方法,还包括:修改所生成的副本以包括所述第二数据记录

元素。

15.一种系统,包括:

用于分析第一版本的计算机程序的部件,所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素,所述第一数据处理元素引用表示所述第一版本的计算机程序外部的数据记录的源或目的地的第一数据记录元素;

用于识别数据记录的存储的部件,所识别的存储不同于所述数据记录的源或目的地;

用于生成表示所识别的数据记录的存储的第二数据记录元素的部件,所述生成包括定义针对所述第二数据记录元素的规范,所述规范包括表示所识别的数据记录的存储的位置的信息和表示由所述第一数据处理元素引用的所述第一数据记录元素的位置的信息;以及

用于基于所述第一版本的计算机程序和针对所述第二数据记录元素的规范来生成第二版本的计算机程序的部件,所述第二版本的计算机程序包括所生成的第二数据记录元素和基于所述第一数据处理元素的第二数据处理元素,其中所述第一版本的计算机程序未被所述第二版本的计算机程序的生成修改,

其中,在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的第二数据记录元素。

16.一种系统,包括:

一个或多个处理器,其连接至存储器,所述一个或多个处理器和所述存储器被配置为:

分析第一版本的计算机程序,所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素,所述第一数据处理元素引用表示所述第一版本的计算机程序外部的数据记录的源或目的地的第一数据记录元素;

识别数据记录的存储,所识别的存储不同于所述数据记录的源或目的地;

生成表示所识别的数据记录的存储的第二数据记录元素,所述生成包括定义针对所述第二数据记录元素的规范,所述规范包括表示所识别的数据记录的存储的位置的信息和表示由所述第一数据处理元素引用的所述第一数据记录元素的位置的信息;以及

基于所述第一版本的计算机程序和针对所述第二数据记录元素的规范来生成第二版本的计算机程序,所述第二版本的计算机程序包括所生成的第二数据记录元素和基于所述第一数据处理元素的第二数据处理元素,其中所述第一版本的计算机程序未被所述第二版本的计算机程序的生成修改,

其中,在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的第二数据记录元素。

17.根据权利要求16所述的系统,其中,识别所述数据记录的存储包括:确定所述数据记录的存储的位置。

18.根据权利要求16所述的系统,其中,分析所述第一版本的计算机程序包括:分析向所述第一版本的计算机程序中所包括的一个或多个数据处理元素的输入流。

19.根据权利要求16所述的系统,其中,分析所述第一版本的计算机程序包括:分析来自所述第一版本的计算机程序中所包括的一个或多个数据处理元素的输出流。

20.根据权利要求16所述的系统,其中,所述计算机程序包括图。

21.根据权利要求20所述的系统,其中,生成所述第二版本的计算机程序包括:当所述第一数据记录元素表示数据记录的源时,使所生成的第二数据记录元素位于向所述第二数

据处理元素的输入流处。

22.根据权利要求20所述的系统,其中,生成所述第二版本的计算机程序包括:当所述第一数据记录元素表示数据记录的目的地时,使所生成的第二数据记录元素位于来自所述第二数据处理元素的输出流处。

23.根据权利要求16所述的系统,所述一个或多个处理器和所述存储器被配置为利用基于来自所述第一数据记录元素的数据的数据来填充所述第二数据记录元素。

24.根据权利要求16所述的系统,所述一个或多个处理器和所述存储器被配置为执行所述第二版本的计算机程序。

25.根据权利要求16所述的系统,其中,生成所述第二版本的计算机程序包括:生成所述第一版本的计算机程序的至少一部分的副本。

26.一种非暂时性计算机可读介质,其存储有指令,所述指令用于使计算系统进行:

分析第一版本的计算机程序,所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素,所述第一数据处理元素引用表示所述第一版本的计算机程序外部的数据记录的源或目的地的第一数据记录元素;

识别数据记录的存储,所识别的存储不同于所述数据记录的源或目的地;

生成表示所识别的数据记录的存储的第二数据记录元素,所述生成包括定义针对所述第二数据记录元素的规范,所述规范包括表示所识别的数据记录的存储的位置的信息和表示由所述第一数据处理元素引用的所述第一数据记录元素的位置的信息;以及

基于所述第一版本的计算机程序和针对所述第二数据记录元素的规范来生成第二版本的计算机程序,所述第二版本的计算机程序包括所生成的第二数据记录元素和基于所述第一数据处理元素的第二数据处理元素,其中所述第一版本的计算机程序未被所述第二版本的计算机程序的生成修改,

其中,在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的第二数据记录元素。

27.根据权利要求26所述的计算机可读介质,其中,识别所述数据记录的存储包括:确定所述数据记录的存储的位置。

28.根据权利要求26所述的计算机可读介质,其中,分析所述第一版本的计算机程序包括:分析向所述第一版本的计算机程序中所包括的一个或多个数据处理元素的输入流。

29.根据权利要求26所述的计算机可读介质,其中,分析所述第一版本的计算机程序包括:分析来自所述第一版本的计算机程序中所包括的一个或多个数据处理元素的输出流。

30.根据权利要求26所述的计算机可读介质,其中,所述计算机程序包括图。

31.根据权利要求30所述的计算机可读介质,其中,生成所述第二版本的计算机程序包括:当所述第一数据记录元素表示数据记录的源时,使所生成的第二数据记录元素位于向所述第二数据处理元素的输入流处。

32.根据权利要求30所述的计算机可读介质,其中,生成所述第二版本的计算机程序包括:当所述第一数据记录元素表示数据记录的目的地时,使所生成的第二数据记录元素位于来自所述第二数据处理元素的输出流处。

33.根据权利要求26所述的计算机可读介质,其存储有指令,所述指令用于使所述计算系统利用基于来自所述第一数据记录元素的数据的数据来填充所述第二数据记录元素。

34.根据权利要求26所述的计算机可读介质,其存储有指令,所述指令用于使所述计算系统执行所述第二版本的计算机程序。

35.根据权利要求26所述的计算机可读介质,其中,生成所述第二版本的计算机程序包括:生成所述第一版本的计算机程序的至少一部分的副本。

应用程序测试

[0001] 本申请是申请日为2015年11月05日、申请号为201580072406.3、发明名称为“应用程序测试”的申请的分案申请。

技术领域

[0002] 本说明书涉及调试图。

背景技术

[0003] 代码开发人员经常在整个开发过程中调试源代码。如此,可以利用可能会影响源代码的功能的调试代码来修改源代码。经常期望在调试结束时从源代码中去除所添加的调试代码以恢复源代码的原始功能。

发明内容

[0004] 在一般方面中,一种方法,包括:利用处理器来分析第一版本的计算机程序。所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素。所述第一数据处理元素引用位于所述第一版本的计算机程序外部的第一数据源。所述方法包括:生成表示与所述第一数据源不同的第二数据源的数据源元素。所述方法包括:生成第二版本的计算机程序。所述第二版本的计算机程序包括所生成的数据源元素和基于所述第一数据处理元素的第二数据处理元素。在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的数据源元素。

[0005] 实施例可以包括以下特征中的一个或多个特征。

[0006] 所述方法包括:确定所述第二数据源的位置。确定所述第二数据源的位置包括:接收表示所述位置的输入。

[0007] 所述方法包括:在用户接口中呈现所识别的所述第一数据处理元素、所述第一数据源或这两者的标识符。所述方法包括:接收所识别的所述第一数据处理元素、所述第一数据源或这两者的选择。

[0008] 分析所述第一版本的计算机程序包括:分析向所述第一版本的计算机程序中的数据处理元素至少之一的输入流。

[0009] 所述方法包括:在所述计算机程序的运行时分析所述第一版本的计算机程序。

[0010] 所述计算机程序包括图。生成所述第二版本的计算机程序包括:使所生成的数据源元素位于向所述第二数据处理元素的输入流处。

[0011] 来自所述第二数据源的数据具有与来自所述第一数据源的数据相同的格式。

[0012] 生成数据源元素包括:针对所生成的数据源元素定义覆盖规范。

[0013] 所述方法包括:利用基于来自所述第一数据源的数据的数据来填充所述第二数据源。

[0014] 在所述第一版本的计算机程序中,第三数据处理元素引用位于所述第一版本的计算机程序外部的第一数据目的地。在所述第二版本的计算机程序中,基于所述第三数据处

理元素的第四数据处理元素引用与所述第一数据目的地不同的第二数据目的地。

[0015] 所述方法包括:识别所述第一版本的计算机程序中所包括的第三数据处理元素。所述第三数据处理元素引用位于所述第一版本的计算机程序外部的第一数据目的地。所述方法包括:生成表示与所述第一数据目的地不同的第二数据目的地的输出元素。所述第二版本的计算机程序包括所生成的输出元素和基于所述第三数据处理元素的第四数据处理元素。在所述第二版本的计算机程序中,所述第四数据处理元素引用所生成的输出元素。所述方法包括:确定所述第二数据目的地的位置。确定所述第二数据目的地的位置包括:接收表示所述位置的输入。

[0016] 所述方法包括:执行所述第二版本的计算机程序。

[0017] 执行所述第二版本的计算机程序使得能够对所述计算机程序进行调试。

[0018] 生成所述第二版本的至少一部分的计算机程序包括:生成所述计算机程序的该部分的副本。所述方法包括:修改所述计算机程序的该部分的副本以包括所生成的数据源元素。

[0019] 在一般方面中,一种系统,包括用于利用处理器来分析第一版本的计算机程序的部件。所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素。所述第一数据处理元素引用位于所述第一版本的计算机程序外部的第一数据源。所述系统包括用于生成表示与所述第一数据源不同的第二数据源的数据源元素的部件。所述系统包括用于生成第二版本的计算机程序的部件。所述第二版本的计算机程序包括所生成的数据源元素和基于所述第一数据处理元素的第二数据处理元素。在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的数据源元素。

[0020] 在一般方面中,一种系统,包括处理器,所述处理器连接至存储器,所述处理器和所述存储器被配置为利用所述处理器来分析第一版本的计算机程序。所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素。所述第一数据处理元素引用位于所述第一版本的计算机程序外部的第一数据源。所述处理器和所述存储器被配置为生成表示与所述第一数据源不同的第二数据源的数据源元素。所述处理器和所述存储器被配置为生成第二版本的至少一部分的计算机程序。所述第二版本的计算机程序包括所生成的数据源元素和基于所述第一数据处理元素的第二数据处理元素。在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的数据源元素。

[0021] 在一般方面中,一种用于存储指令的非暂时性计算机可读介质,所述指令用于使计算系统利用处理器来分析第一版本的计算机程序。所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素。所述第一数据处理元素引用位于所述第一版本的计算机程序外部的第一数据源。所述指令使所述计算系统生成表示与所述第一数据源不同的第二数据源的数据源元素。所述指令使所述计算系统生成第二版本的至少一部分的计算机程序。所述第二版本的计算机程序包括所生成的数据源元素和基于所述第一数据处理元素的第二数据处理元素。在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的数据源元素。

[0022] 根据以下的说明书、以及根据权利要求书,本发明的其它特征和优点将变得明显。

附图说明

[0023] 图1是图(graph)的示例。

[0024] 图2是覆盖规范的示例。

[0025] 图3是框图。

[0026] 图4是流程图。

[0027] 图5A～5D是修改后的图的示例。

[0028] 图6～8是框图。

[0029] 图9是流程图。

具体实施方式

[0030] 在测试或调试诸如图等的可执行应用程序时,测试人员或开发人员可能想要使用特殊的一组输入数据来进行测试。在一些示例中,测试人员可能希望对应用程序作出改变。通过在改变之前和之后都使用一致的一组输入数据执行应用程序,可以监测该改变对应用程序所输出的数据的影响。在一些示例中,测试人员可以具有在测试应用程序时要使用的特定一组测试数据,诸如将使得至少执行一次应用程序的所有功能的一组测试数据等。同样,测试人员可能希望将应用程序所输出的数据写入与应用程序将其输出数据写入至的标准目的地不同的特殊目的地。

[0031] 在传统的开发环境中,测试人员手动提供测试所用的期望的一组输入数据并且指定输出数据的目的地。这里,说明用以自动识别用于向应用程序提供输入数据的数据源和用于从应用程序接收输出数据的输出数据宿的方法。所识别的数据源可以被替换数据源(有时称为测试源)自动替换。所识别的输出数据宿可以被替代目的地(有时称为探测器)自动替换。测试源和探测器是插入的示例。

[0032] 在一些示例中,可执行应用程序是基于图的处理。诸如测试源和探测器等的插入是与基于图的处理中的流相关联的对象。测试源可以利用新数据替换通过流的数据(例如,上游数据),使得上游计算不必针对图的每次执行重新运行。例如,测试源可以替换数据源,使得从测试源而不是从数据源向图提供测试数据。探测器可以监测在图执行时通过流的数据,并且可以使得保存该数据以供随后检查或重复使用。例如,探测器可以接收原本将被保存至诸如数据库等的输出数据宿的数据。

[0033] 可以利用作为与图或其它可执行应用程序分开的文件的覆盖规范来定义诸如测试源和探测器等的插入。在一些示例中,可以基于应用程序的自动化分析、例如基于应用程序的数据源和输出数据宿的自动识别,来自动定义插入。

[0034] 覆盖规范中所定义的插入可以在执行期间被添加到应用程序中,而不是成为原始程序的一部分。在编译应用程序时,编译人员考虑覆盖文件并且生成包括插入的可执行应用程序。有时将原始应用程序称为第一版本应用程序并且将包括插入的应用程序称为第二版本应用程序。例如,在基于图的处理的示例中,可以将可执行图从视觉上表示为包括第一版本图的与覆盖规范中所定义的插入对象相组合的组件的第二版本图。在一些示例中,可执行图是shell脚本(外壳脚本)并且没有存储在文件中。在一些示例中,可执行图和图存储在单独文件中。

[0035] 将插入并入第二版本图中没有修改调试中的第一版本图。作为代替,插入定义保

持在单独文件(例如,单独的覆盖规范)中,并且可以在代码生成开始时变为普通图组件以包括在修改后的图中。如此,不存在调试期间无意中破坏原始图的风险。

[0036] 还可以在图中的除数据源和输出数据宿以外的位置处引入插入。这些插入(有时称为内部插入)可以使得测试人员能够在数据流经图时访问该数据。例如,测试人员可能希望在数据从一个组件流向另一组件时验证该数据。测试人员还可能已验证了上游处理正确地起作用,但可能没有验证下游处理。在一些情况下,上游处理可能需要长的时间来执行,这导致测试不高效。如果测试人员可以利用先前验证的数据替换先前验证的上游操作,则可以提高测试效率。在传统的开发环境中,测试人员将需要修改图以添加观察点或者利用预先验证的数据替换上游组件。然而,一旦编辑了图,则测试人员无法确定出没有修改原始图的功能。在一些环境中,测试人员可能缺少编辑图所需的许可。

[0037] 图1示出图100的示例。图100是包括通过流所连接的数据处理组件的计算机程序的视觉表示。连接两个组件的流表示从第一组件输出的记录被传递至第二组件。在第一组件通过流连接至第二组件时,第一组件引用第二组件。

[0038] 诸如数据库(如图所示)、文件、队列、可执行语句(例如,SQL语句)或位于图100外部的另一类型的数据源等的数据源102包括利用图100要处理的一个或多个数据记录。外部是指数据源102的数据没有存储在图100中。数据源102通过流连接至过滤组件103。通常,过滤组件过滤或去除不满足预定标准的记录。在该示例中,过滤组件103使居住于俄亥俄州的客户的数据记录通过并且拒绝其它记录。过滤组件103连接至排序组件104,其中该排序组件104用于按照邮政编码对过滤后的数据记录进行排序。排序组件104连接至复制组件106,其中该复制组件106用于创建数据记录的副本,使得可以采用两个不同的方式来处理数据记录。该复制组件连接至重定格式组件108和按表达式过滤组件110。例如,将按邮政编码进行排序后的居住于俄亥俄州的客户的数据记录的一个实例发送至重定格式组件108,并且将这些数据记录的另一实例发送至按表达式过滤组件110。重定格式组件108将数据记录的格式改变为不同的数据格式,并且按表达式过滤组件110基于与数据记录相关联的表达式来去除数据记录。重定格式组件108和按表达式过滤组件110连接至用于组合所接收到的数据记录的收集组件112,并且该收集组件连接至位于图外部的输出数据宿组件114,诸如数据库(如图所示)、文件、队列或下游处理组件等。外部是指输出数据宿114的数据没有存储在图100中。尽管图100包括许多在组件之间的流,但在该示例中,特别关注数据源102和过滤组件103之间的流116(有时称为源-过滤流116)以及收集组件112和输出数据宿114之间的流118(有时称为收集-输出流118)。

[0039] 图100的测试人员可能希望调试图100以验证其功能。在一些情况下,测试人员可能想要在数据从一个组件流向另一组件时验证数据。在一些情况下,测试人员可能想要绕过图100中的上游组件,而是将数据插入所绕过的组件的位置。在一些情况下,测试人员可能想要使用一致的一组输入数据来测试图100的操作以监测改变该图对该图所输出的数据的影响。在一些情况下,测试人员可能想要使用该测试人员知晓的将使得至少执行一次图的所有功能的一组输入数据来测试图100的操作,由此使得能够完成图的测试。

[0040] 在调试图100时,可以期望避免修改图。例如,测试人员可能不想冒破坏图的功能的风险。在一些示例中,测试人员对图的访问权限可能受限或者可能无权访问图(例如,测试人员可能缺少编辑图所需的许可)。为了在无需修改图100的情况下调试该图,可以使用

覆盖来调试图。在一些示例中,可以例如基于图的自动化分析来自动指定覆盖。可以基于原始图100(有时称为第一版本图)和覆盖规范来生成第二版本的至少一部分的图100。

[0041] 图2示出用于定义覆盖的覆盖规范200的示例。覆盖规范200可以存储在文件中。该文件可以与包含图的文件分开。覆盖规范定义一个或多个插入。插入可以是与图100的流相关联的对象并且可以采用探测器或测试源的形式。

[0042] 探测器在数据通过图100的组件之间的流(例如,沿着从第一组件向第二组件的流或者沿着向输出数据宿的流通过)时收集或监测该数据。例如,在图100执行时数据通过流的情况下,可以监测该数据、可以保存该数据以供随后检查、或者可以保存该数据以供重复使用。覆盖规范可以定义参考携载要收集或监测的数据的流的探测器。探测器指定要收集或监测数据所经由的流。探测器可被配置为报告特定值,或者报告特定值何时在预定范围内或在预定范围外。可以保存通过探测器的数据以供随后分析或使用,例如可以将数据存储在平面文件或关系数据库中。

[0043] 在一些示例中,探测器可以参考从图100的组件向诸如文件或数据库等的输出数据宿的流。通过在图100的调试期间沿着向数据宿的流配置探测器,该探测器接收从图100输出的数据。例如,每次在调试模式中执行图100时,输出数据可被探测器接收到并且写入文件,使得可以比较或评价来自各种图执行的输出数据。在一些示例中,自动识别输出数据宿并且自动指定覆盖,以定义供所识别的输出数据宿之前的插入用的探测器。

[0044] 在一些示例中,探测器可以参考从图100的上游组件向下游组件的流。通过在图100的调试期间沿着向下游组件的流配置探测器,该探测器接收原本将被下游组件接收到的数据,由此防止下游组件执行。例如,测试人员可能希望监测下游组件之前的图处理的结果。例如,下游组件可以具有对图的外部产生影响的功能,例如下游组件可以向该下游组件处理信用卡记录的各人发送文本消息。在图的调试期间,测试人员可能希望禁用对图的外部产生影响的组件。

[0045] 测试源将数据插入图100中的图100的两个组件之间的特定流处。覆盖规范可以定义参考如下流的测试源,其中该流携载要利用来自测试源的数据替换的数据。在一些示例中,测试源利用新数据替换通常将通过流的数据。在一些方案中,测试源可被配置为读取先前保存的数据,并且将该数据传递至下游组件。在一些示例中,测试源将数据插入图100中的来自数据源(诸如数据库或文件等)的流处。测试源可以插入具有与数据源原本将提供的数据相同的格式的数据。在一些示例中,自动识别数据源并且自动指定覆盖,以定义要替换所识别的数据源的测试源。

[0046] 在一些示例中,图100的直到特定点为止(例如,直到特定组件为止)的执行的结果可能以前已被验证。换句话说,可能已验证了直到特定点为止的上游处理功能。在这些情况下,上游组件在每次图100执行时重新处理功能这一做法可能效率低。测试源可以将数据(例如,先前验证的数据)插入图中的该特定点处。这样,可以绕过图100的先前执行的整个部分。

[0047] 图2示出用于定义覆盖的覆盖规范200的示例。覆盖规范200可以包括一个或多个插入定义。在该示例中,覆盖规范200包括一个测试源定义201和一个探测器定义213。覆盖规范200从指定插入定义可以对应于的图的3行标题开始。该标题之后是测试源定义201,其中该测试源定义201包括名称202、上游端口204、下游端口206、插入类型208、原型路径210

和布局参数212。

[0048] 测试源定义201的上游端口204引用测试源要被插入图100中的流的正上游的组件的输出端口。流的上游的组件是输出端口数据被输出到流上的组件。在图2的示例中,测试源定义201的上游端口204指向数据库102的输出。测试源定义201的下游端口206引用测试源要插入到图100中的流的正下游的组件的输入端口。流的下游的组件是从该流接收到输入端口数据的组件。在图2的示例中,测试源定义的下游端口206指向过滤组件103的输入。因而,该示例中的测试源定义201表示要在数据库102的输出和过滤组件103的输入之间的流中配置测试源,使得测试源所提供的数据可以替换来自数据库102的输入数据。

[0049] 插入类型208定义插入是测试源还是探测器。值“0”定义测试源,并且值“1”定义探测器。由于该插入是测试源,因此插入类型108的值是“0”。

[0050] 原型路径210表示插入的类型。在该示例中,由于该插入是测试源,因此原型路径210指定输入文件组件。原型路径210指向包含用于定义特定类型的插入的代码的文件。布局参数212定义包含测试源将包含的数据的源文件的位置。在一些示例中,该位置是文件路径。源文件中的数据要替换通常将通过由上游端口204和下游端口206定义的流的数据。即,在将测试源应用于图100时,过滤组件103接收源文件中的数据而不是从数据库102接收数据。

[0051] 源文件包含具有与测试源的下游的组件原本将接收到的数据相同的格式的数据。在一些示例中,源文件中的数据可以与测试源的上游的数据源(例如,数据库)中的数据相同。例如。可以将来自数据库102的数据记录复制到源文件中。在一些示例中,数据源表示诸如SQL查询等的可执行语句。在这些示例中,可以执行SQL查询并且可以将查询执行的结果存储在源文件中。在一些示例中,可以从除数据源以外的某处获得源文件中的数据。例如,可以生成源文件中的数据,从而确保对特定数据(例如,特定范围的值)进行处理以进行图100的完整调试。在一些示例中,即使数据源中的数据改变,源文件中的数据也保持相同,由此使得能够利用一致的一组输入数据继续调试。

[0052] 在一些示例中,源文件中的数据可以与图100的正常执行期间将通过流的数据相同,但通过使用测试源插入数据,上游组件可以避免处理。例如,诸如复制组件106等的上游组件可能要求大量的系统资源来处理数据,或者与数据流图100中的其它组件相比可能需要相对较长的时间来处理数据。如此,可以将已知数据(例如,在正常执行期间将通过流的相同数据)插入流中以节省时间或节约系统资源。

[0053] 测试源定义201之后是探测器定义213,其中该探测器定义213包括名称214、上游端口216、下游端口218、插入类型220和原型路径222。

[0054] 探测器定义213的上游端口216引用探测器要被插入图100中的流的正上游的组件的输出端口。在图2的示例中,探测器定义213的上游端口216指向收集组件112的输出。探测器定义213的下游端口218引用探测器要被插入图100中的流的正下游的组件的输入端口。在图2的示例中,探测器定义213的下游端口218指向输出数据宿组件114。因而,该示例中的探测器定义213表示要在收集组件112的输出和输出数据宿组件114之间的流中配置探测器,使得探测器接收原本将被写入输出数据宿组件的数据。

[0055] 探测器定义213的插入类型220定义插入是测试源还是探测器。值“1”表示探测器。由于该插入是探测器,因此插入类型220的值是“1”。

[0056] 原型路径222表示插入的类型。在该示例中,由于该插入是探测器,因此原型路径222指定输出文件组件。原型路径222指向包含用于定义特定类型的插入的代码的文件。

[0057] 在一些示例中,要利用探测器监测的数据被存储在系统自动创建的文件中。该文件可以存储在系统所确定的位置中。探测器监测通过由上游端口216和下游端口218所定义的流的数据。即,在将探测器应用于图100时,监测从收集组件112的输出通向输出数据宿组件114的输入的数据并且将该数据存储在系统自动创建的文件中。在一些示例中,可以在存储该数据之前监测该数据。该文件能够接收具有探测器定义所引用的组件(在该示例中为外部数据宿组件114)将接收的相同格式的数据。

[0058] 在一些示例中,作为图的自动化分析的结果,可以利用覆盖规范来定义一个或多个插入。例如,可以进行图100的自动化分析以识别诸如数据库、文件或其它类型的数据源等的任何数据源。所识别的数据源中的一个或多个数据源可被测试源自动替换。被替换的数据源是指将测试源插入该数据源的正下游的流中,使得将来自该测试源的数据而不是将来自该数据源的数据提供至下游组件。同样,图100的自动化分析可以识别诸如数据库、文件或其它类型的输出数据宿等的任何输出数据宿。所识别的输出数据宿中的一个或多个输出数据宿可被探测器自动替换。被替换的输出数据宿是指将探测器插入该输出数据宿的正上游的流中,使得来自上游组件的数据被探测器而不是被该输出数据宿接收到。还可以使用图100的自动化分析来识别诸如特定类型的组件(例如,执行对图100的外部产生影响的特定类型的组件)等的其它组件。

[0059] 参考图3,分析引擎300自动分析图100以识别数据源302和输出数据宿304。例如,分析引擎300可以访问针对图100的各节点的参数和连接。如果给定节点不存在传入连接,则分析引擎300将该节点识别为数据源。同样,如果给定节点不存在传出连接,则分析引擎300将该节点识别为输出数据宿。为了访问并分析图的各节点,分析引擎“遍历”图的所有连接。在一些示例中,直到运行时(例如,在处理为了调试目的而开始时)为止,图100未被实例化或参数化。分析引擎300可以在运行时进行自动化分析以识别图100中的数据源和输出数据宿。

[0060] 分析引擎300将数据源302和输出数据宿304的标识符发送至插入引擎306,其中该插入引擎306确定数据源和输出数据宿中的哪些数据源和输出数据宿要被测试源和探测器分别替换。在一些示例中,测试人员308提供要被测试源和探测器替换的数据源和输出数据宿的列表310。可以将列表310作为文件、数据库或采用其它格式来提供。例如,测试人员308可以将他期望频繁地改变的任何数据源包括在列表310上。通过利用测试源替换这种数据源,测试人员308可以确保可使用一致的输入数据来测试图。

[0061] 插入引擎306将各个所识别的数据源302和输出数据宿304与列表310上的数据源和输出数据宿进行比较。该插入引擎针对列表310上出现的任何数据源302或输出数据宿304创建覆盖规范312。在一些示例中,利用分析引擎300将诸如上游端口和下游端口等的针对覆盖规范312的参数提供至插入引擎306。在一些示例中,插入引擎306访问图100以获得相关参数。

[0062] 为了针对测试源创建覆盖规范312,插入引擎306利用数据填充源文件。在一些示例中,插入引擎306利用从数据源302复制得到的数据填充将替换特定数据源302的测试源所用的源文件。在一些示例中,数据源302包括诸如SQL语句等的可执行表达式,并且插入引

擎306执行该可执行表达式并利用执行结果填充源文件。在一些示例中,插入引擎306可以经由用户接口314向测试人员308提示源文件所用的数据。例如,插入引擎306可以向测试人员308呈现所识别的数据源302的列表,使得测试人员308可以选择所识别的数据源302中的哪些数据源要被测试源替换。测试人员308还可以指定要包括在该测试源所用的源文件中的数据。在一些情况下,测试人员308可以识别包括该测试源所用的数据的文件的位置(例如,路径)。在一些情况下,测试人员308可以指示插入引擎306生成作为原始数据源302中的数据的副本的源文件。在一些情况下,测试人员308可以指示插入引擎306执行原始数据源302中所包括的或者与原始数据源302相关联的诸如SQL语句等的可执行表达式。在一些情况下,测试人员308可以使得针对测试源的源文件生成数据。例如,测试人员308可以提供将使得至少执行一次图中的每个功能的一组数据(诸如真实数据或所生成数据等)。

[0063] 为了针对探测器创建覆盖规范312,插入引擎306确定输出数据要被存储的文件的位置。在一些示例中,该位置是默认(例如,由系统架构师)设置的。在一些示例中,插入引擎306可以经由用户接口314提示测试人员308指定输出数据文件所用的位置。

[0064] 参考图4,在用以自动定义插入的一般方法中,接收要被测试源和探测器分别替换的数据源和输出数据宿的列表(400)。例如,该列表可以由测试人员基于他关于数据源和输出数据宿的知识或者基于调试的目标或目的来提供。在一些示例中,该列表还可以包括要替换该列表上所包括的数据源和输出数据宿的文件的诸如位置和文件名等的标识符。

[0065] 例如利用处理器自动分析图,以识别该图中的数据源、输出数据宿或这两者(402)。特别地,不具有传入连接的组件被识别为数据源,并且不具有传出连接的组件被识别为输出数据宿。例如,分析各组件以识别该组件的传入连接和传出连接,并且来自各组件的各连接被实现直到相邻组件以识别该组件的传入连接和传出连接。这样,可以分析图的所有组件。在一些示例中,可以在运行时(例如,在图被参数化之后)自动进行该分析。在一些示例中,可以自动地且动态地(例如,在图正运行时)进行该分析。例如,可以在图的执行期间解析特定参数时进行动态分析。在一些示例中,图被接收到短期存储器中,其中利用处理器从该短期存储器中分析图以识别数据源或输出数据宿。

[0066] 自动将所识别的数据源和输出数据宿与列表上的数据源和输出数据宿进行比较(404)。针对列表上所包括的各个所识别的数据源或输出数据宿定义覆盖规范(406)。在执行图之前,编译器可以将图编译成可执行图。作为编译的一部分,编译器考虑覆盖规范200。例如,编译器可以接受覆盖规范200作为输入。对一个或多个插入进行处理并且将该一个或多个插入以各自与覆盖规范200中所包含的插入定义相对应的对象的形式插入图中。可以将插入对象连同第一版本图100中所包括的数据处理组件一起在(图5A所示的)第二版本图500中表示。插入对象可以通过定向流连接至数据处理组件或其它插入。然而,覆盖规范200或存储该覆盖规范的文件保持与包含图的文件分开。即,尽管插入对象以及第一版本图100中所包括的数据处理组件可以出现在第二版本图500中,但包含第一版本图100的文件不包括插入定义。插入对象有时被简称为插入。

[0067] 在一些示例中,测试人员没有供给要替换的数据源和输出数据宿的初始列表。相反,自动分析图并且经由用户接口向测试人员呈现与图相关联的所有数据源和输出数据宿的列表。测试人员可以选择数据源和输出数据宿中的要被插入替换的一个或多个数据源和输出数据宿。测试人员可以提供要替换列表上所包括的数据源和输出数据宿的文件的诸如

位置和文件名等的标识符,或者可以提供用于生成插入所用的源文件的指令。

[0068] 可以使用单执行模式(Single-Execution Mode)和保存状态模式(Saved-State Mode)的至少两个模式其中之一来执行覆盖规范中所定义的插入。

[0069] 图6示出用于在单执行模式中执行插入定义的示例系统。在该示例中,客户端602生成或引用第一版本图604和用于定义插入的覆盖文件606(例如,覆盖规范)。例如,覆盖文件606可以是图2的覆盖规范200。然后,利用编译器608编译图604。编译器608考虑覆盖文件606并且创建第二版本图。第二版本图是可执行的并且包括覆盖文件606所定义的插入。然后,可以执行第二版本图。在一些示例中,编译和执行同时发生。如果要再次执行第二版本图,则重复该处理,其中该处理包括重新指定、重新编译图604并且重新执行第二版本图。从可执行图的一次执行起直到可执行图的下次执行为止,没有保存信息。

[0070] 图5A示出第二版本图500的示例。第二版本图500是图的视觉表示。在该示例中,第二版本图500与图1的第一版本图100相似,并且已被修改为包括插入。第二版本图500包括图2的覆盖规范200中所定义的插入的表示。测试源插入502与测试源定义201相对应,并且探测器插入504与探测器定义213相对应。在该示例中,在编译器编译图100时生成插入。尽管图5A示出第二版本图500,但原始的第一版本图100保持未被修改。

[0071] 在数据源102的输出和过滤组件103的输入之间配置测试源插入502,其中流116已位于第一版本图100中。该插入的位置是基于测试源定义中的上游端口204和下游端口206(图2)。在执行第二版本图500的情况下,数据没有如第一版本图100的情况那样从数据源102流向过滤组件103。作为代替,来自利用测试源502的测试源定义201中的布局参数212所标识的源文件的数据流向过滤组件103。

[0072] 在收集组件112的输出和输出数据宿组件114的输入之间的流118中配置探测器插入504。该插入的位置是基于探测器定义213中的上游端口216和下游端口218(图2)。在执行第二版本图500的情况下,利用探测器插入504来监测并存储从收集组件112流向输出数据宿组件114的数据。如上所述,为了再次执行可执行图,对图进行重新指定和重新编译,并且要重新执行可执行图。从可执行图的一次执行起直到可执行图的下次执行为止,没有保存信息。例如,如果要再次执行可执行图,则将利用相同的数据重新填充探测器插入504。

[0073] 在图5A的示例中,沿着流118流动的数据被探测器插入504和输出数据宿组件114这两者接收到。参考图5B,在一些示例中,第二版本图500′可以包括探测器插入504′,其中该探测器插入504′使向输出数据宿组件114的流中断,使得数据被探测器插入504′接收到而没有流向输出数据宿组件114。

[0074] 在图5A和5B的示例中,第二版本图500、500′包括测试源插入502(或502′)和探测器插入504(或504′)这两者。在一些示例中,第二版本图可以包括多个测试源插入和多个探测器插入。参考图5C,在一些示例中,第二版本图500″可以包括一个或多个测试源插入502″,但不包括探测器插入。参考图5D,在一些示例中,第二版本图500″′可以包括一个或多个探测器插入504″′,但不包括测试源插入。

[0075] 图7示出用于利用保存状态管理器708在保存状态模式中执行插入定义的示例系统。在该示例中,客户端702生成或引用图704和用于定义插入的覆盖文件706(例如,覆盖规范)。例如,覆盖文件706可以是图2的覆盖规范200。保存状态储存库710由保存状态管理器708和编译器712来管理。保存状态管理器708还可以识别保存状态数据位于保存状态储存

库710内的何处。利用编译器712对图704进行编译。编译器712考虑覆盖文件706并且创建包括覆盖文件706所定义的插入的第二版本图。然后,可以执行该第二版本图。在一些示例中,编译和执行同时发生。保存状态模式与单执行模式的不同之处在于:保存状态模式使得能够在执行之间保存信息的同时,多次执行可执行图。例如,参考图5A,如果使用保存状态模式来执行图2的覆盖规范200中所定义的插入,则在第二次执行期间可以不必重新填充在第一次执行第二版本图时所填充的探测器插入504。在一些示例中,由于探测器插入504实质是将数据插入到图中的收集组件112的输出和输出数据宿组件114的输入之间的流处,因此在第二次执行时探测器插入504可以在内部变换为测试源。

[0076] 可以驻留在保存状态管理器目录中的保存状态管理器708管理保存状态。可以保存在保存状态储存库710中的信息的示例包括与探测器插入有关的信息、与测试源插入有关的信息、与覆盖文件706有关的信息和与图组件相关联的参数(例如,属性)等。

[0077] 在一些示例中,在执行可执行图时,仅执行图的特定部分。即,仅执行图的特定组件。在一些示例中,执行比图的所有组件少的组件。可执行图可以仅执行将会影响插入的组件。例如,可执行图可以仅执行图中的所定义的探测器插入监测并存储数据所需的部分。在一些示例中,可能不必执行最下游探测器的下游的组件。在一些示例中,第二版本图是整个原始图的第二版本。在一些示例中,第二版本图是整个原始图的仅一部分的第二版本、例如该图的仅与所定义插入有关的部分的第二版本。

[0078] 在一些示例中,在第一次执行可执行图时填充探测器。在执行之间,可以改变图组件中的一个或多个图组件的参数。组件的参数定义了组件如何工作。可以追踪与组件相关联的参数,使得编译器712可以确定在组件中何时发生参数变化。最后值表追踪与图组件相关联的参数。在运行可执行图时,将最后值表与组件的当前参数进行比较以判断在运行之间任何参数是否发生改变。在发生参数变化的情况下,该变化可能会或者可能不会影响探测器所存储的数据。编译器712判断组件和变化是否影响下次执行可执行图时探测器将存储的数据。如果探测器将存储的数据受到该变化影响,则在下次执行可执行图期间,重新执行改变后的组件以及驻留于该改变后的组件和探测器之间的流上的组件。换句话说,如果改变后的组件影响组件的执行并且该组件影响探测器所存储的数据,则再次执行该改变后的组件和探测器之间的组件。如果探测器将存储的数据不会受到该变化影响,则可以不必重新执行任何组件。

[0079] 图8示出可以使用调试技术的示例数据处理系统800。该系统800包括数据源802,其中该数据源802可以包括诸如存储装置或者至线上数据流的连接等的一个或多个数据源,其中该一个或多个数据源各自可以以各种格式(例如,数据库表、电子表格文件、非结构文本(flat text)文件或大型机所使用的原本格式)中的任何格式来存储或提供数据。执行环境804和开发环境818例如可以在诸如某个版本的UNIX操作系统等的适当的操作系统的控制下安装在一个或多个通用计算机上。例如,执行环境804可以包括包含使用多个中央处理单元(CPU)或多个处理器内核的计算机系统的结构的多节点并行计算环境,可以是本地的(例如,诸如对称多处理(SMP)计算机等的多处理器系统)或本地分布式的(例如,作为集群所连接的多个处理器或大规模并行处理(MPP)系统)、或者远程或远程分布式的(例如,经由局域网(LAN)和/或广域网(WAN)连接的多个处理器)、或者它们的任何组合。

[0080] 执行环境804从数据源802读取数据并且生成输出数据。提供数据源802的存储装

置相对于执行环境804可以是本地的,例如存储在连接至安装有执行环境804的计算机的存储介质(例如,硬盘驱动器808)上,或者相对于执行环境804可以是远程的,例如安装在经由(例如云计算基础设施所提供的)远程连接与安装有执行环境804的计算机进行通信的远程系统(例如,大型机110)上。数据源802可以包含测试源定义(例如,图2的测试源定义201)中所定义的数据。即,测试源定义201的布局参数212可以指向数据源802中的源文件的位置。

[0081] 输出数据可被存储回至数据源802中或者存储回至执行环境804可访问的数据存储系统816中,或者被使用。开发环境818也可以访问数据存储系统816,其中在开发环境818中,开发人员820能够开发、调试并测试图。在一些实现中,开发环境818是用于开发作为图的应用程序的系统,其中这些图包括顶点(表示数据处理组件或数据集),并且这些顶点通过顶点之间的定向流(directed flow)(表示工作元素(即,数据)的流)相连接。例如,在标题为“Managing Parameters for Graph-Based Applications”的美国公开号2007/0011668中更详细地说明了这种环境。

[0082] 简单的数据流图可以处理通过流连接至诸如过滤组件等的数据处理元素的输入数据集。过滤组件通过流连接至输出数据集。数据集例如可以包括用于提供数据(例如,输入数据集)或接收数据(例如,输出数据集)以供数据流图所进行的计算用的文件或数据库表。可以将数据流图中利用“流”表示的数据的流组织成离散的数据元素。例如,这些元素可以包括来自被组织成记录(或行)和字段(或列)的数据集的记录。描述与记录中的值相对应的字段和数据类型的序列的元数据被称为“记录格式”。图中的组件和数据集具有用于连接至流的输入和/或输出端口。流的“源端”与输入数据集的输出端口并且与过滤组件的输出端口相连接。流的“宿端”与过滤组件的输入端口并且与输出数据集的输入端口相连接。数据集或组件的输入或输出端口与元数据(诸如流入端口或从端口流出的数据所用的记录格式等)相关联。可以使用与形成“子图”的流互连的其它组件来实现图中所使用的组件。

[0083] 在标题为“EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS”的美国专利5,966,072中说明了用于执行这种基于图的计算的系统。根据该系统所制作的图提供用于将信息输入至图组件所表示的个别处理或从图组件所表示的个别处理获得信息的方法、用于在处理之间移动信息的方法以及用于定义处理的运行顺序的方法。该系统包括用于从任何可用方法中选取处理间通信方法的算法(例如,与图的流相对应的通信路径可以使用TCP/IP或UNIX域套接字或者使用共享存储器在处理之间传递数据)。

[0084] 开发环境818包括用于存储源代码的代码储存库822。在一些示例中,可以由例如经由用户接口有权访问开发环境的开发人员820来开发源代码和覆盖规范(例如,图2的覆盖规范200)。在一些示例中,源代码和覆盖规范例如是由上述的分析引擎300和插入引擎306自动确定的。在一些示例中,图和覆盖规范可以存储在代码储存库822中。在一些示例中,图存储在代码储存库822中,并且覆盖规范存储在单独的覆盖储存库824中。

[0085] 代码储存库822和覆盖储存库824其中之一或这两者可以与编译器826进行通信。编译器826可以将第一版本图和覆盖规范(例如,图2的覆盖规范200)编译成可执行的第二版本图828。例如,编译器可以接受覆盖规范作为输入。对一个或多个插入进行处理并且将该一个或多个插入以各自与覆盖规范中所包含的插入定义相对应的对象的形式插入图中。可以利用修改图(例如,图5A的第二版本图500)来从视觉上表示第二版本图828。可以在第二版本图500中表示插入对象。

[0086] 开发环境818可以包括测试执行环境830,其中该测试执行环境830用于执行第二版本图828。例如,一旦利用编译器826对图进行了编译,则可以执行第二版本图828。执行第二版本图828可以包括执行与第二版本图828的组件、插入和定向流相关联的计算作为组件之间的数据流(例如,工作元素或数据记录)。在一些示例中,测试执行环境830在无需修改代码储存库822中所存储的第一版本图的源代码或者覆盖储存库824中所存储的源代码的情况下,执行第二版本图828。测试执行环境830可以是经由开发环境818的接口可访问的,或者可以具有其自己的接口。该接口可被配置为显示与执行有关的信息。该接口还可被配置为显示与插入有关的信息(例如,探测器正监测并保存的数据、或者测试源正插入的数据)。测试执行环境830可以允许开发人员820多次执行第二版本图828并且在执行之间修改第二版本图828的多个方面。

[0087] 在一些示例中,开发人员指导图的插入和编译。例如,开发人员820从代码储存库822中选择图1的第一版本图100。开发人员820还从覆盖储存库824中选择图2的覆盖规范200。在一些示例中,代替选择覆盖规范200,开发人员820可以从覆盖储存库824内的各种覆盖规范中选择插入定义。开发人员820指示编译器826基于第一版本图100和覆盖规范200来对第二版本图828进行编译。

[0088] 在一些示例中,插入可以是自动插入的。例如,如上所述,例如通过识别不具有传入连接或不具有传出连接的组件来自动识别图100中的数据源和输出数据宿。自动将所识别的数据源和输出数据宿与在图100的调试期间要被插入替换的数据源和输出数据宿的列表进行比较。例如,该列表可以由开发人员820提供。根据该列表来针对图100的数据源和输出数据宿自动创建覆盖规范。然后,自动编译第二版本图。

[0089] 在一些示例中,在执行之后,开发人员820可以评价输出到探测器插入504中的数据。在必要的情况下,开发人员820可以仍然使用来自测试源502的相同输入数据,来对第一版本图100作出改变并且重新执行改变后的第一版本图100。通过使输入数据在图的多轮执行内保持相同,开发人员820可以将从改变后的第一版本图输出的数据与先前输出的数据进行比较,以判断第一版本图是否正按照期望进行。

[0090] 在一些示例中,在执行期间或在执行之后,开发人员820可以观察与第二版本图828的执行及其组件、插入和流有关的信息。例如,返回参考图5A,开发人员820可以观察到探测器插入504所监测并存储的数据不正确或出乎意料。开发人员820可以从覆盖储存库824中选择用于定义要插入在重定格式组件108和收集组件112之间的流中的探测器的探测器插入定义以及用于定义要插入在按表达式过滤组件和收集组件112之间的流中的探测器的探测器插入定义。开发人员820可以分析这两个探测器所收集到的数据,以判断是来自重定格式组件108的数据还是来自按表达式过滤组件110的数据正导致从收集组件112输出的数据不正确。

[0091] 继续该示例,假定开发人员820判断为来自按表达式过滤组件110的数据不正确。代替调试按表达式过滤组件110的上游的所有组件(例如,过滤组件103、排序组件104和复制组件106)以判断不正确数据的原因,开发人员820可以选择将正确数据(例如,预期从按表达式过滤组件110输出的数据)插入按表达式过滤组件110和收集组件112之间的流中。开发人员820可以从覆盖储存库824中选择用于定义要插入在按表达式过滤组件110和收集组件112之间的流中的测试源的测试源插入定义。如果开发人员820关注对按表达式过滤组件

110的下游的第二版本图500的组件进行调试,则该方法可以是适当的。

[0092] 在一些示例中,覆盖规范没有被作为文件永久地存储在代码储存库822或覆盖储存库824中。相反,通常将包括在覆盖文件中的信息(例如,插入定义)是由开发人员820(例如,经由用户接口)所开发的、或者是由分析引擎300和插入引擎306自动确定的并暂时存储在存储器中。然后,将覆盖信息传递至编译器(例如,图6的608)或保存状态管理器(例如,图7的708)。

[0093] 图9是示出调试过程900的流程图。接收到第一版本图(例如,图1的图100)(902)。例如,第一版本图可被接收到处理器可访问的短期存储器中。第一版本图100包括组件和流。组件表示针对数据记录所进行的操作,并且流表示组件之间的数据记录的流。这些组件可以包括图1所示的过滤组件103、排序组件104、复制组件106、重定格式组件108、按表达式过滤组件110和收集组件112。这些流可以包括图1所示的复制-重定格式流116和收集-输出数据宿流118。

[0094] 接收到用于定义一个或多个插入的覆盖规范(904)。在一些示例中,覆盖规范是从开发人员或测试人员接收到的。在一些示例中,例如如上所述,覆盖规范是自动定义的。覆盖规范可以是图2所示的覆盖规范200。覆盖规范可以包括一个或多个插入定义(例如,测试源定义201和探测器定义213)。插入定义可以包括(测试源定义所用的)名称、上游端口、下游端口、插入类型、原型路径和布局参数。所定义的各个插入可以与图100的流相关联。插入可以采取探测器或测试源的形式。采用图的组件的形式的插入的示例包括图5A的测试源插入502和探测器插入504。例如,测试源插入502与图100的数据库-过滤流116相关联,并且探测器插入504与图100的收集-输出数据宿流118相关联。

[0095] 生成各自与所定义的插入其中之一相对应的一个或多个对象(906)。这些对象可以是图的组件。对象的示例包括图5A的测试源插入502和探测器插入504。例如,测试源插入502与图100的复制-重定格式流116相关联,并且探测器插入504与图100的收集-输出数据宿流118相关联。

[0096] 生成第二版本的至少一部分的图(908),其中该图包括图100的该部分的组件和流中的至少一些组件和流以及所生成的一个或多个对象。在一些示例中,第二版本图是原始图100的被修改为包括图100的该部分的组件和流中的至少一些组件和流以及所生成的一个或多个对象的副本。可以利用修改后的图(例如,图5A的第二版本图500)来从视觉上表示第二版本图。将各对象插入与对应于该对象的所定义插入相关联的流处。例如,参考第二版本图500,将测试源插入502插入复制-重定格式流116中,并且将探测器插入504插入收集-输出数据宿流118中。尽管所生成的插入对象以及图100的数据处理组件可以出现在第二版本图500中,但第一版本图100(或包含第一版本图100的文件)没有被修改。

[0097] 尽管描述了可以对图和覆盖规范进行编译以创建包括覆盖文件所定义的插入的第二版本图的编译器(例如,图6的编译器608和图7的编译器712),但在一些实施例中,没有对图和覆盖规范进行编译。例如,可以在无需编译的情况下直接执行图和覆盖规范。解释器可以通过将各语句翻译成已编译成机器代码的一个或多个子例程的序列来直接执行图和覆盖规范。

[0098] 尽管描述了采用探测器和测试源的形式的插入,但在一些实施例中,插入可以采用其它形式。插入可以广泛地用于将数据注入图的给定点并从图的给定点提取数据。例如,

插入可被设计成监测通过图的流的数据的质量。如果数据质量低于阈值,则用户可以接收自动警报。在美国申请序列号14/715,904中可以发现关于插入的更多说明。

[0099] 此外,尽管在图的上下文中描述了插入,但在一些实施例中,可以将插入与其它可执行应用程序相结合地使用。例如,可以通过针对通用的可执行应用程序的自动化分析来识别该应用程序所用的数据源和输出数据宿。所识别的数据源和输出数据宿可以被测试源和探测器分别替换。这样,可执行应用程序可以处理来自测试源的数据并向探测器输出数据。该结构对于测试或调试可执行应用程序而言可以是有用的。

[0100] 上述的调试方法可以使用执行适当软件的计算系统来实现。例如,该软件可以包括在一个或多个编程或可编程计算系统(可以具有诸如分布式、客户端/服务器或网格等的各种架构)上执行的一个或多个计算机程序中的过程,其中该一个或多个编程或可编程计算系统各自包括至少一个处理器、至少一个数据存储系统(包括易失性和/或非易失性存储器和/或存储元件)、(用于使用至少一个输入装置或端口接收输入、并且用于使用至少一个输出装置或端口提供输出的)至少一个用户接口。该软件可以包括例如提供与图的设计、结构和执行相关的服务的更大程序的一个或多个模块。该程序的模块(例如,图的元素)可以实现为符合数据储存库中所存储的数据模型的数据结构或其它有组织的数据。

[0101] 可以将软件设置在诸如(例如,利用通用或专用计算系统或装置可读取的)CD-ROM或其它计算机可读介质等的有形非暂时性介质上、或者经由网络的通信介质(例如,以编码在传播信号中的形式)传递至执行该软件的计算系统的有形非暂时性介质。可以在专用计算机上、或者使用诸如协处理器或现场可编程门阵列(FPGA)或专用集成电路(ASIC)等的专用硬件来进行该处理的一部分或全部。可以以利用不同的计算元件来进行软件所指定的计算的不同部分的分布式方式来实现该处理。优选将这种计算机程序各自存储在通用或专用可编程计算机可访问的存储装置的计算机可读存储介质(例如,固态存储器或介质、或者磁性或光学介质)上或者下载至该计算机可读存储介质,以在利用计算机读取存储装置介质以进行这里所述的处理的情况下配置计算机并使该计算机进行工作。本发明的系统还可被视为作为配置有计算机程序的有形非暂时性介质来实现,其中如此配置成的介质使计算机以特定的预定义方式进行工作,以进行这里所述的处理步骤中的一个或多个。

[0102] 已经说明了本发明的多个实施例。然而,应当理解,上述说明意图例示而并非限制由所附权利要求书的范围所限定的本发明的范围。因此,其它实施例也在所附权利要求书的范围内。例如,可以在没有背离本发明的范围的情况下进行各种变形。另外,上述步骤中的一部分可以是与顺序无关的,因而可以以与所描述的顺序不同的顺序来进行。

[0103] 例如,除上述特征外或者作为上述特征的替代,说明以下实施例:

[0104] 实施例1涉及一种方法,该方法包括利用处理器来分析第一版本的计算机程序。所述分析包括识别所述第一版本的计算机程序中所包括的第一数据处理元素。所述第一数据处理元素引用位于所述第一版本的计算机程序外部的第一数据源。所述方法还包括生成表示与所述第一数据源不同的第二数据源的数据源元素。所述方法还包括生成第二版本的计算机程序。所述第二版本的计算机程序包括所生成的数据源元素和基于所述第一数据处理元素的第二数据处理元素。在所述第二版本的计算机程序中,所述第二数据处理元素引用所生成的数据源元素。

[0105] 实施例2涉及实施例1,其中,所述方法还包括:确定所述第二数据源的位置。

[0106] 实施例3涉及前述实施例中的任一项,其中,确定所述第二数据源的位置包括:接收表示所述位置的输入。

[0107] 实施例4涉及前述实施例中的任一项,其中,所述方法还包括:在用户接口中呈现所识别的第一数据处理元素、所述第一数据源或这两者的标识符。

[0108] 实施例5涉及实施例4,其中,所述方法还包括:接收所识别的第一数据处理元素、所述第一数据源或这两者的选择。

[0109] 实施例6涉及前述实施例中的任一项,其中,分析所述第一版本的计算机程序包括:分析向所述第一版本的计算机程序中的数据处理元素至少之一的输入流。

[0110] 实施例7涉及前述实施例中的任一项,其中,所述方法还包括:在所述计算机程序的运行时分析所述第一版本的计算机程序。

[0111] 实施例8涉及前述实施例中的任一项,其中,所述计算机程序包括图。

[0112] 实施例9涉及实施例8,其中,生成所述第二版本的计算机程序包括:使所生成的数据源元素位于向所述第二数据处理元素的输入流处。

[0113] 实施例10涉及前述实施例中的任一项,其中,来自所述第二数据源的数据具有与来自所述第一数据源的数据相同的格式。

[0114] 实施例11涉及前述实施例中的任一项,其中,生成数据源元素包括:针对所生成的数据源元素定义覆盖规范。

[0115] 实施例12涉及前述实施例中的任一项,其中,所述方法还包括:利用基于来自所述第一数据源的数据的数据来填充所述第二数据源。

[0116] 实施例13涉及前述实施例中的任一项,其中,在所述第一版本的计算机程序中,第三数据处理元素引用位于所述第一版本的计算机程序外部的第一数据目的地。在所述第二版本的计算机程序中,基于所述第三数据处理元素的第四数据处理元素引用与所述第一数据目的地不同的第二数据目的地。

[0117] 实施例14涉及前述实施例中的任一项,其中,所述方法还包括:识别所述第一版本的计算机程序中所包括的第三数据处理元素。所述第三数据处理元素引用位于所述第一版本的计算机程序外部的第一数据目的地。所述方法还包括:生成表示与所述第一数据目的地不同的第二数据目的地的输出元素。所述第二版本的计算机程序包括所生成的输出元素和基于所述第三数据处理元素的第四数据处理元素。在所述第二版本的计算机程序中,所述第四数据处理元素引用所生成的输出元素。

[0118] 实施例15涉及实施例14,其中,所述方法还包括:确定所述第二数据目的地的位置。

[0119] 实施例16涉及实施例15,其中,确定所述第二数据目的地的位置包括:接收表示所述位置的输入。

[0120] 实施例17涉及前述实施例中的任一项,其中,所述方法还包括:执行所述第二版本的计算机程序。

[0121] 实施例18涉及前述实施例中的任一项,其中,执行所述第二版本的计算机程序使得能够对所述计算机程序进行调试。

[0122] 实施例19涉及前述实施例中的任一项,其中,生成所述第二版本的至少一部分的计算机程序包括:生成所述计算机程序的该部分的副本。

[0123] 实施例20涉及实施例19,其中,所述方法还包括:修改所述计算机程序的该部分的副本以包括所生成的数据源元素。

[0124] 优先权要求

[0125] 本申请要求2015年5月19日提交的美国专利申请14/715,807的优先权,其中该美国专利申请14/715,807要求2014年11月5日提交的美国临时专利申请序列号62/075,451的优先权,这两个申请的全部内容通过引用包含于此。

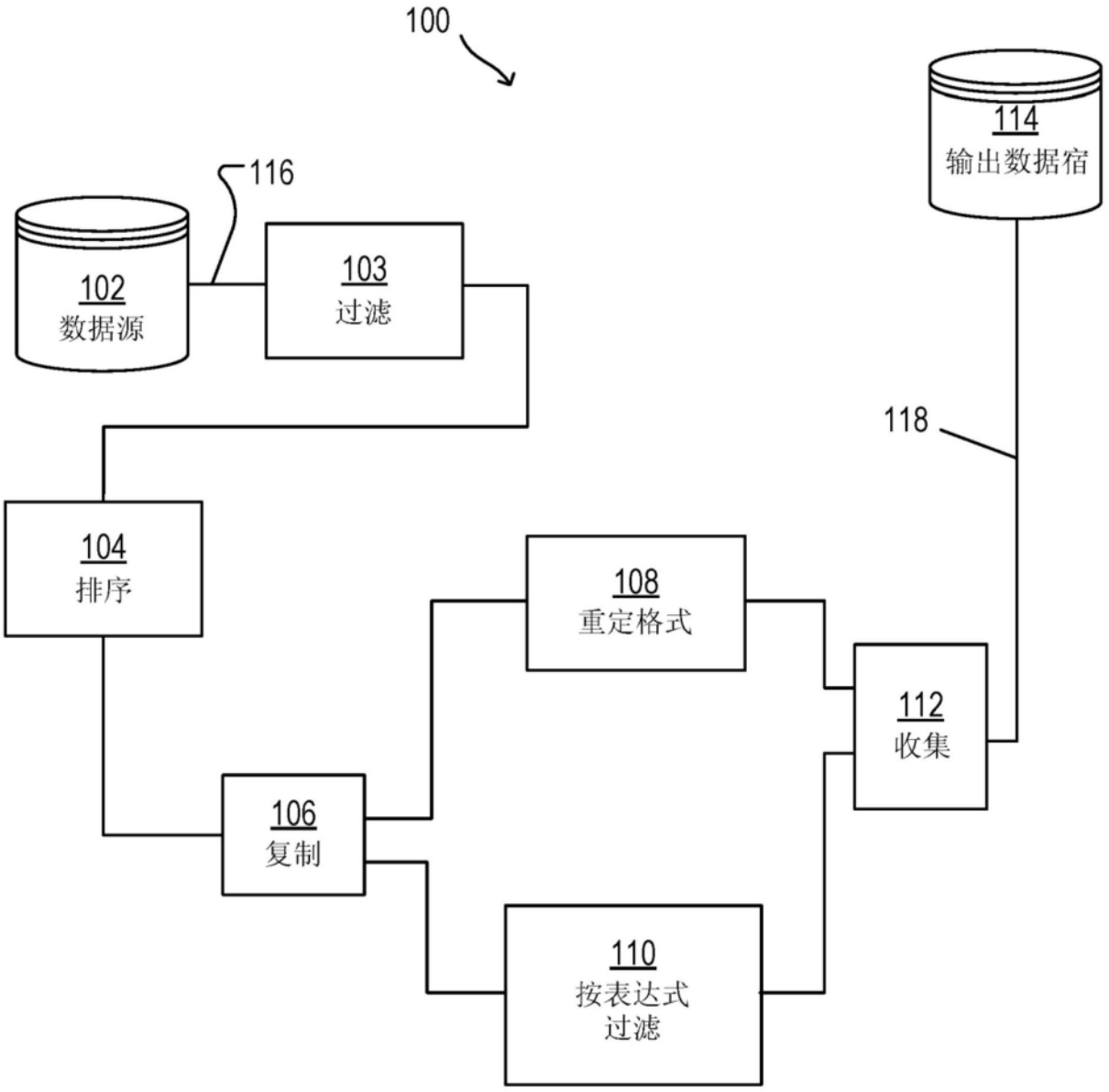

100
114
输出数据宿
116
102
数据源
103
过滤
118
104
排序
108
重定格式
112
收集
106
复制
110
按表达式
过滤

图1

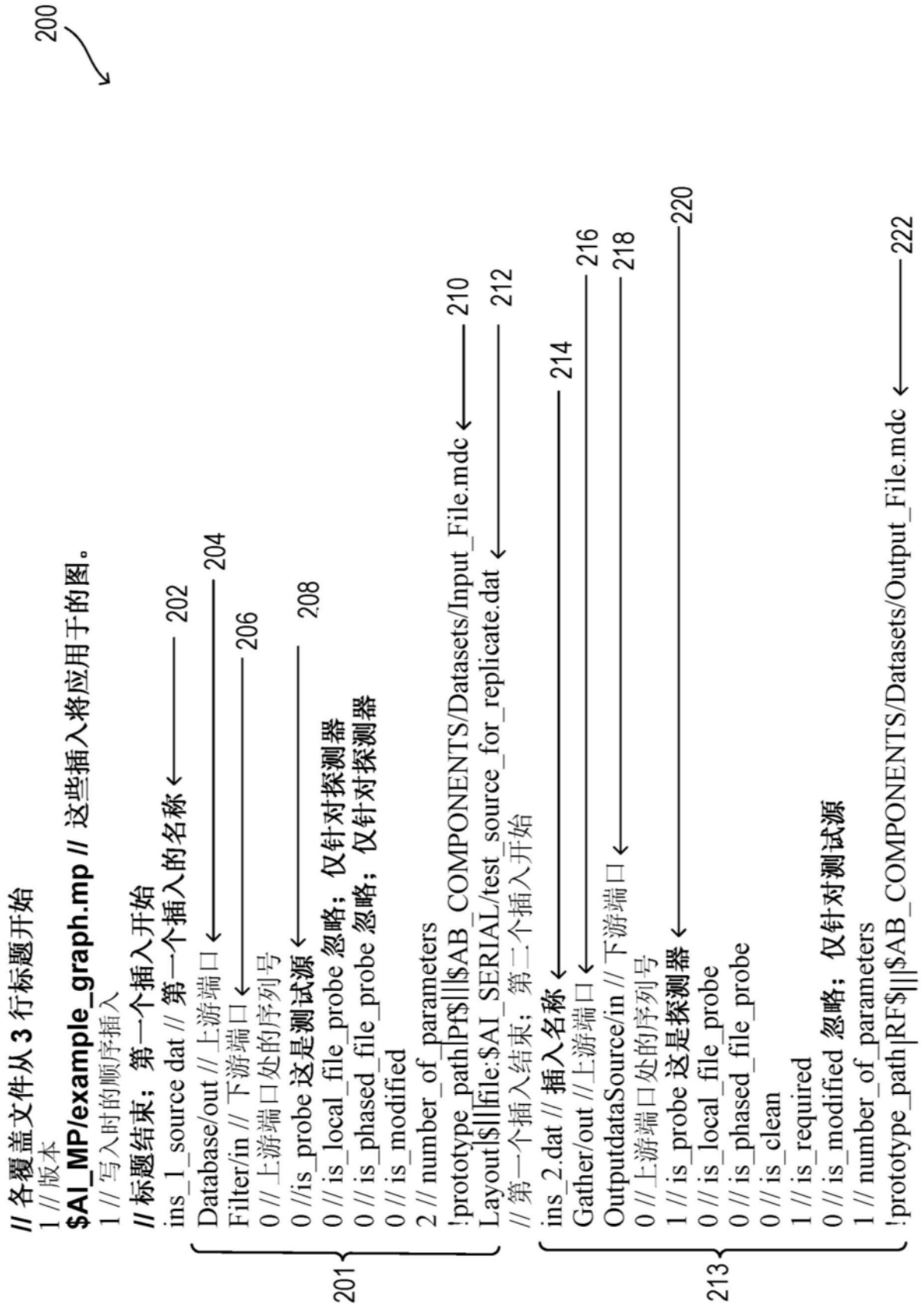

200
// 各覆盖文件从 3 行标题开始
1 // 版本
$AI_MP/example_graph.mp // 这些插入将应用于的图。
1 // 写入时的顺序插入
// 标题结束：第一个插入开始
ins_1_source dat // 第一个插入的名称
202
Database/out // 上游端口
204
Filter/in // 下游端口
206
0 // 上游端口处的序列号
0 //is_probe 这是测试源
208
0 // is_local_file_probe 忽略；仅针对探测器
0 // is_phased_file_probe 忽略；仅针对探测器
0 // is_modified
2 // number_of_parameters
!prototype_path|Pf$|||$AB_COMPONENTS/Datasets/Input_File.mdc
210
Layout|$|||file:$AI_SERIAL/test_source_for_replicate.dat
212
201
// 第一个插入结束；第二个插入开始
ins_2.dat // 插入名称
214
Gather/out // 上游端口
216
OutputdataSource/in // 下游端口
218
0 // 上游端口处的序列号
1 // is_probe 这是探测器
220
0 // is_local_file_probe
0 // is_phased_file_probe
0 // is_clean
1 // is_required
0 // is_modified 忽略；仅针对测试源
1 // number_of_parameters
!prototype_path|RF$|||$AB_COMPONENTS/Datasets/Output_File.mdc
222
213

图2

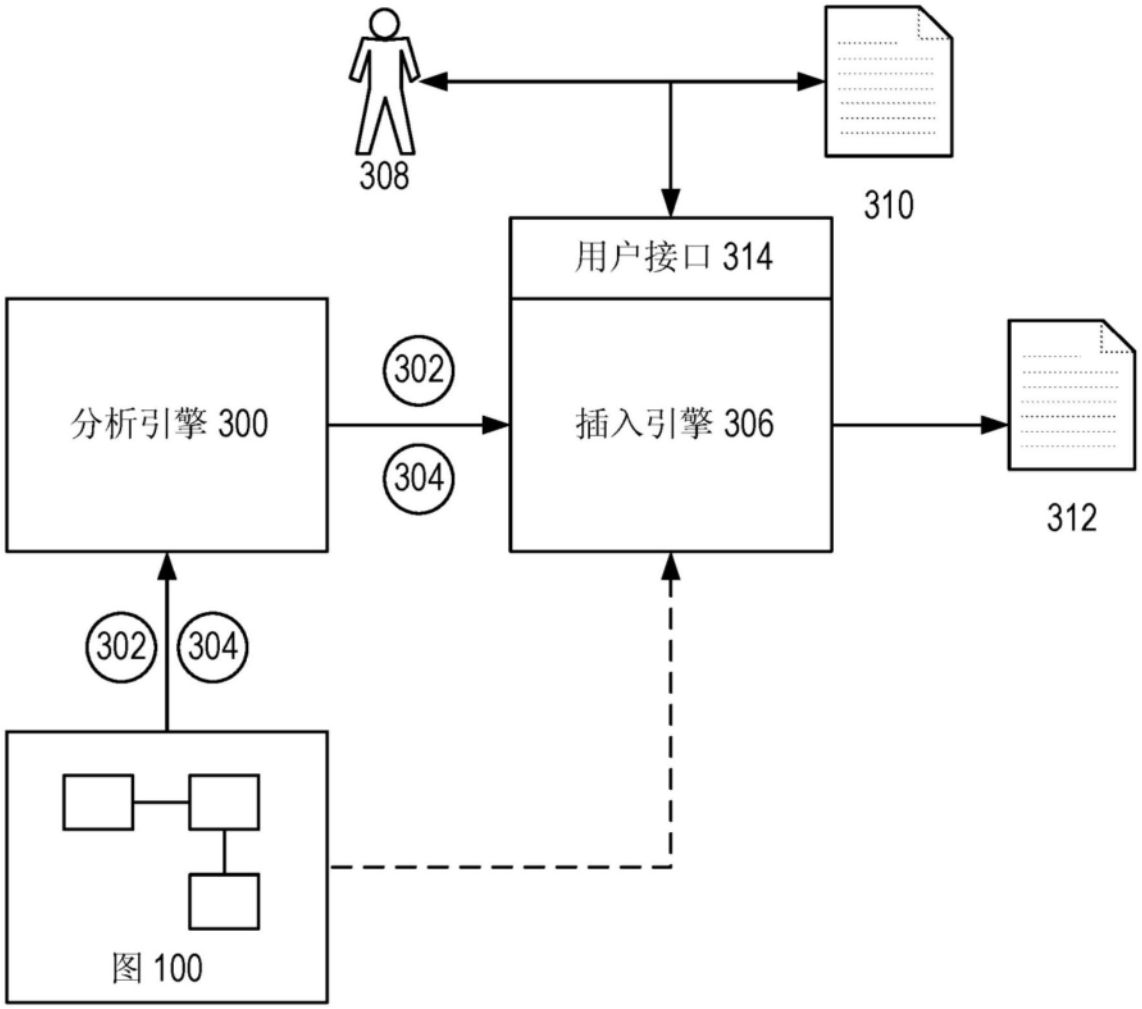


图3

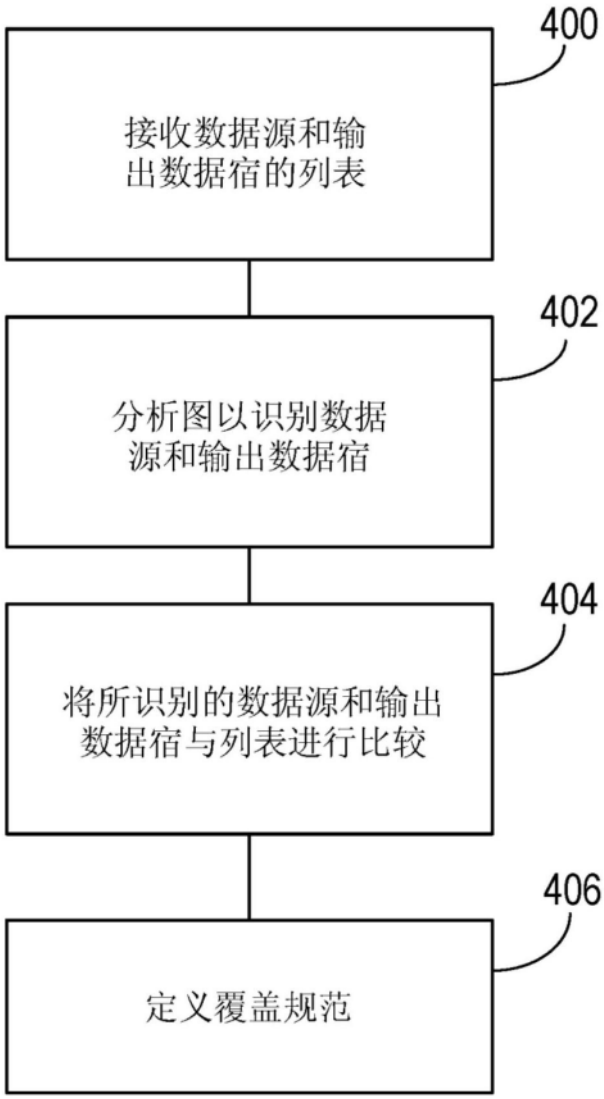

400
接收数据源和输出数据宿的列表
402
分析图以识别数据源和输出数据宿
404
将所识别的数据源和输出数据宿与列表进行比较
406
定义覆盖规范

图4

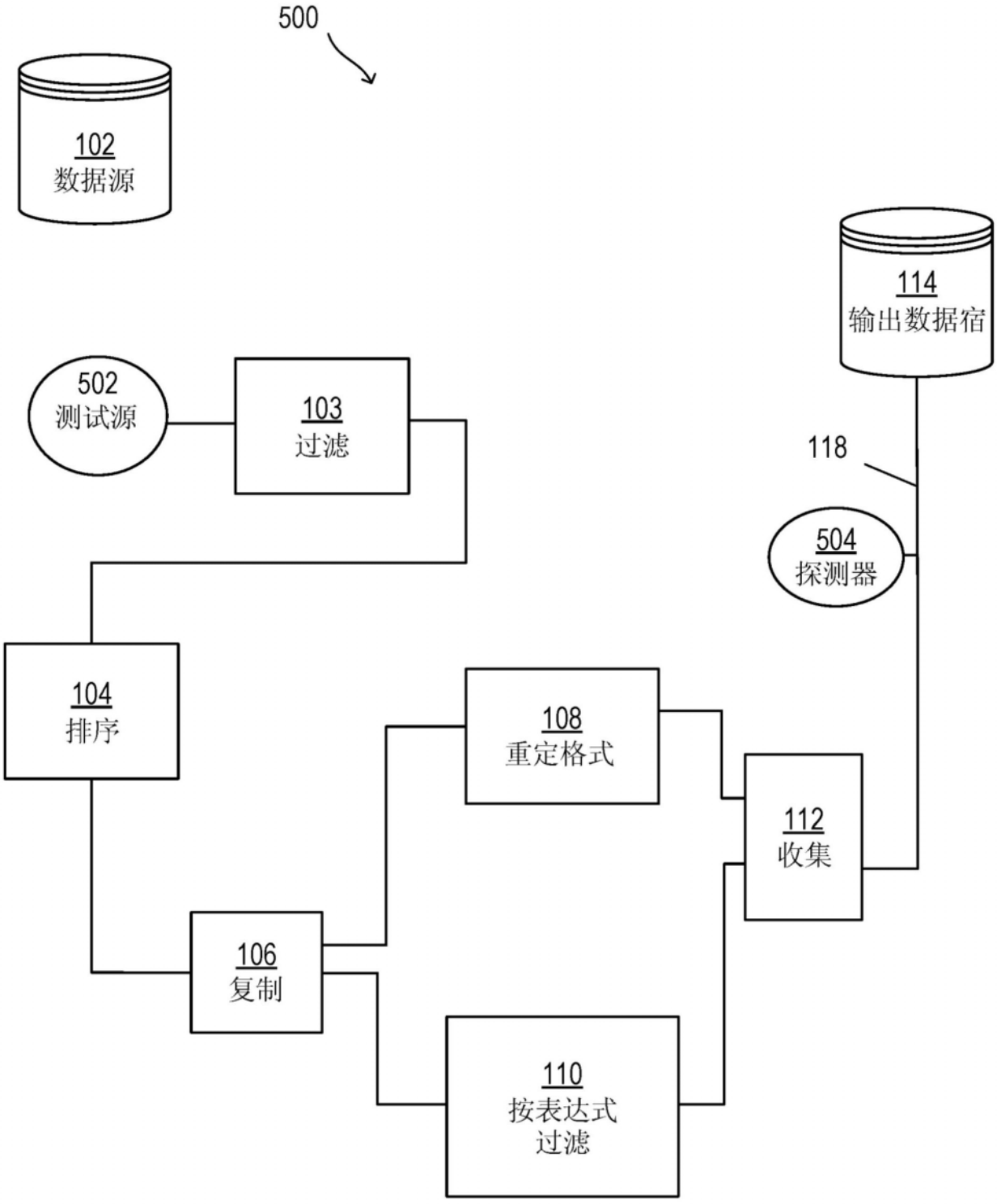

500
102
数据源
114
输出数据宿
502
测试源
103
过滤
118
504
探测器
104
排序
108
重定格式
112
收集
106
复制
110
按表达式
过滤

图5A

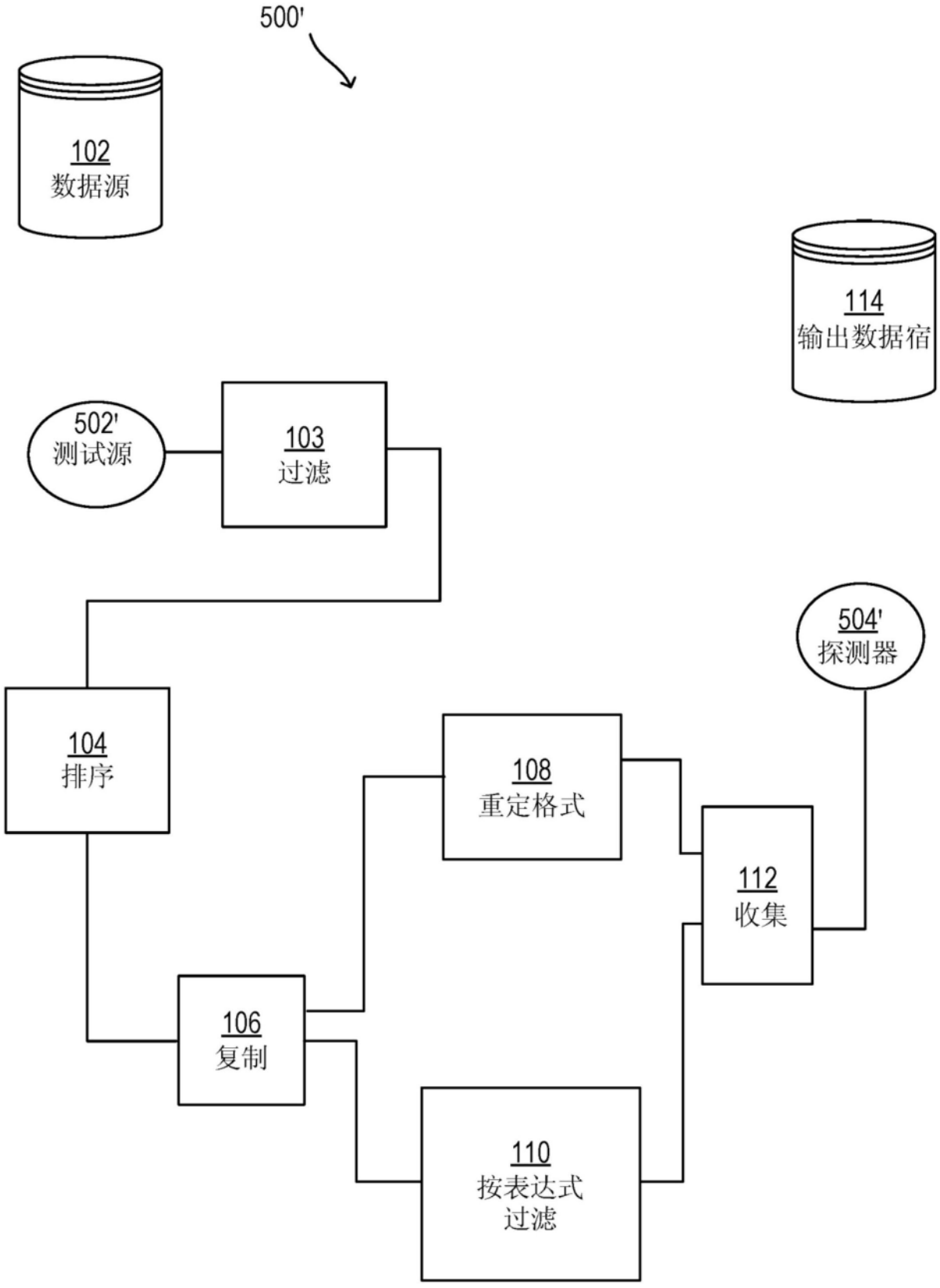

500'
102
数据源
114
输出数据宿
502'
测试源
103
过滤
504'
探测器
104
排序
108
重定格式
112
收集
106
复制
110
按表达式
过滤

图5B

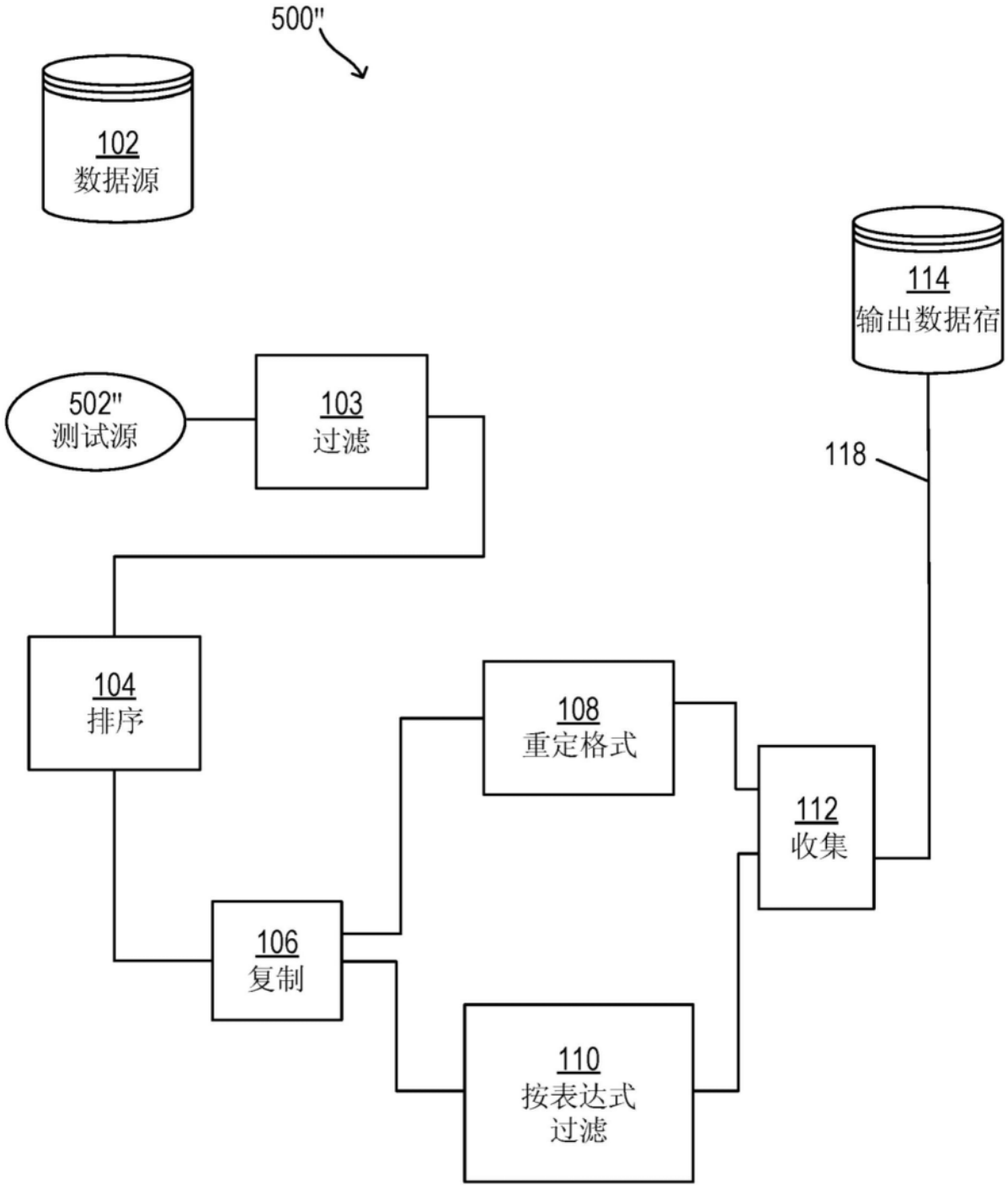

500"
102
数据源
114
输出数据宿
502"
测试源
103
过滤
118
104
排序
108
重定格式
112
收集
106
复制
110
按表达式
过滤

图5C

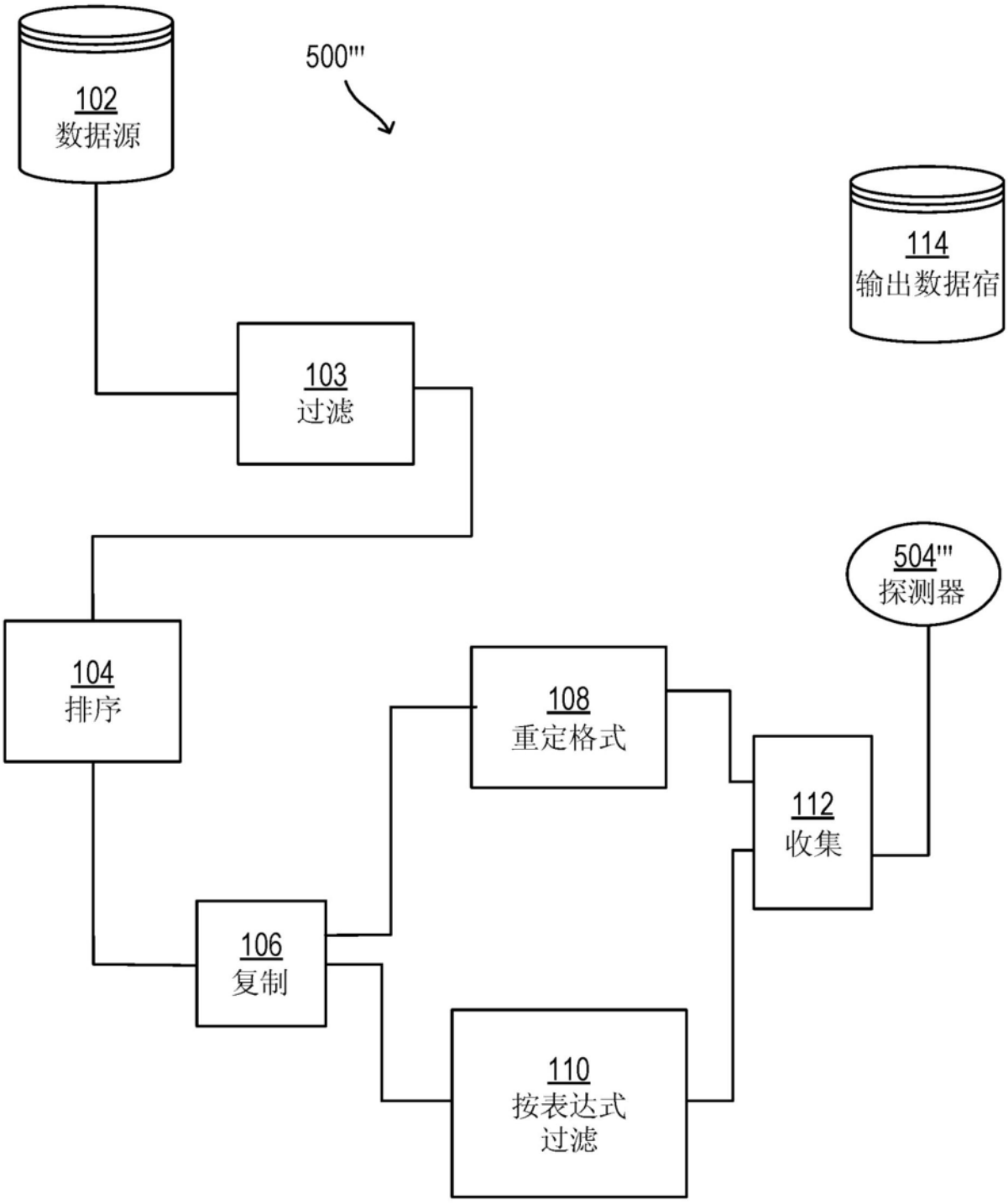

500'''
102
数据源
114
输出数据宿
103
过滤
504'''
探测器
104
排序
108
重定格式
112
收集
106
复制
110
按表达式
过滤

图5D

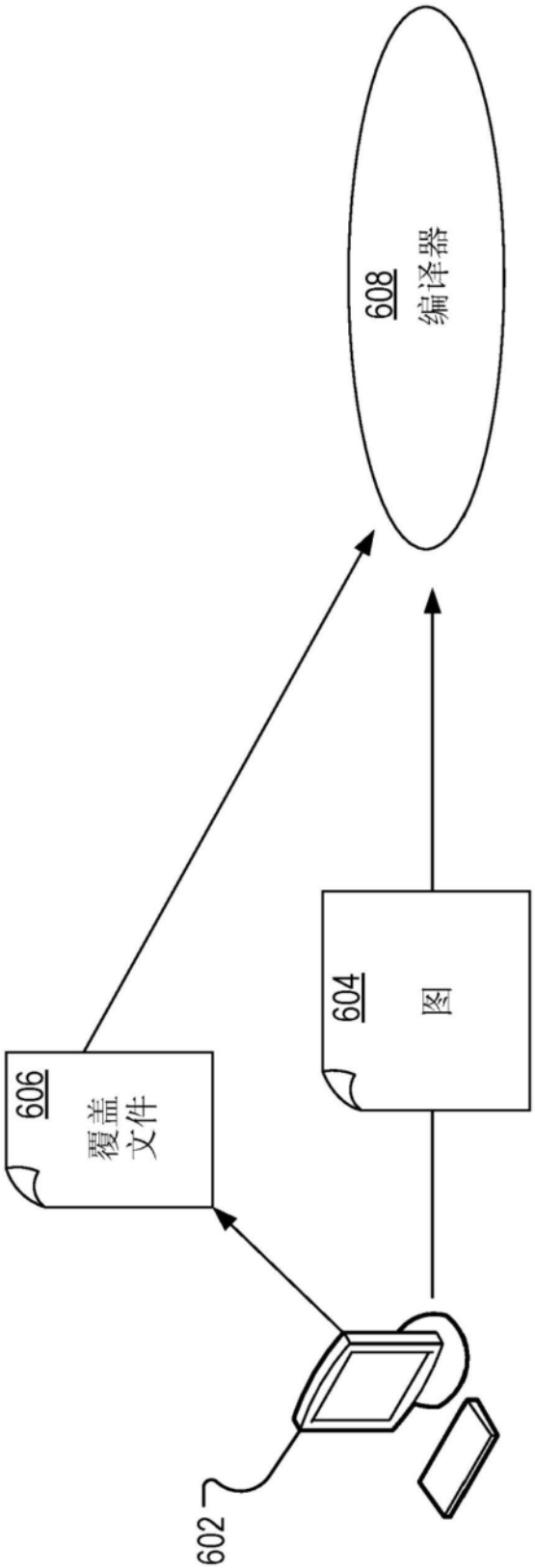


图6

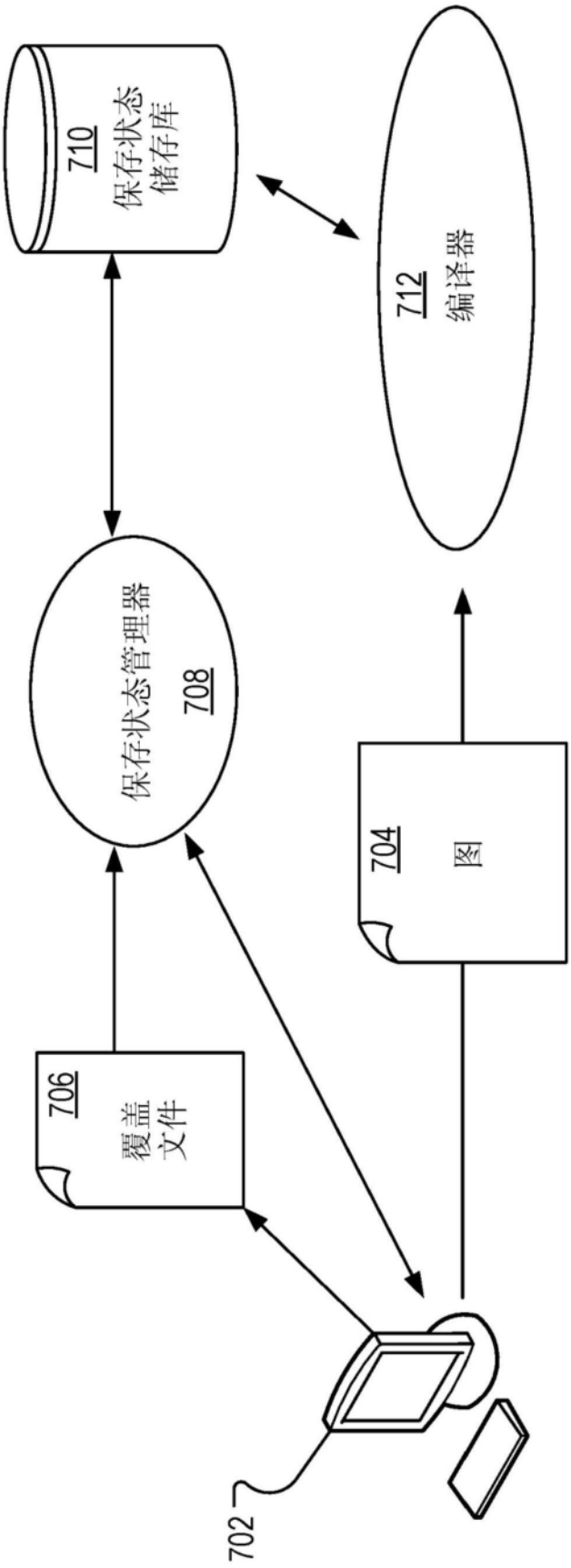


图7

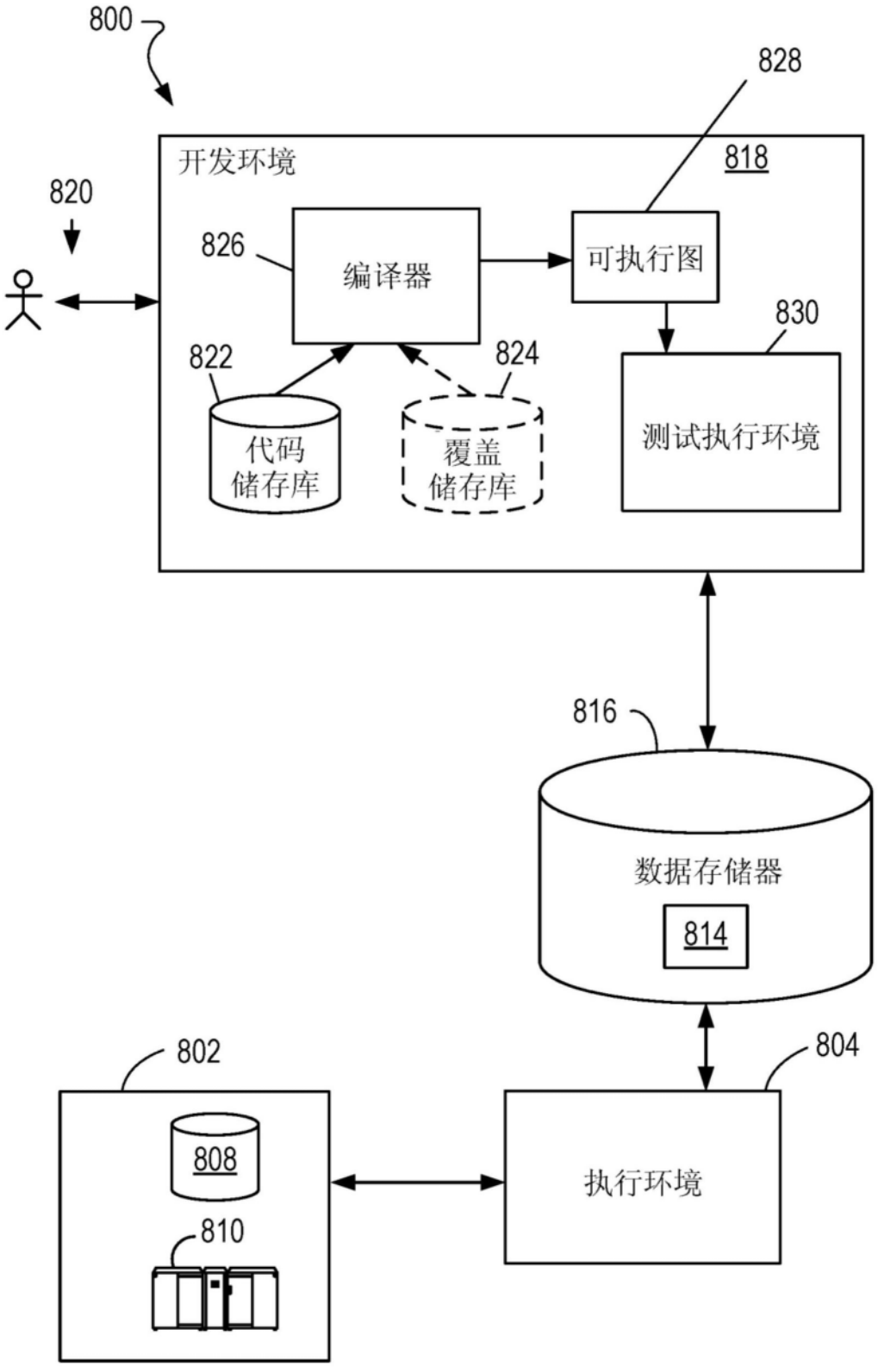

800
828
开发环境
818
820
826
编译器
可执行图
830
822
824
代码
储存库
覆盖
储存库
测试执行环境
816
数据存储器
814
802
804
808
执行环境
810

图8

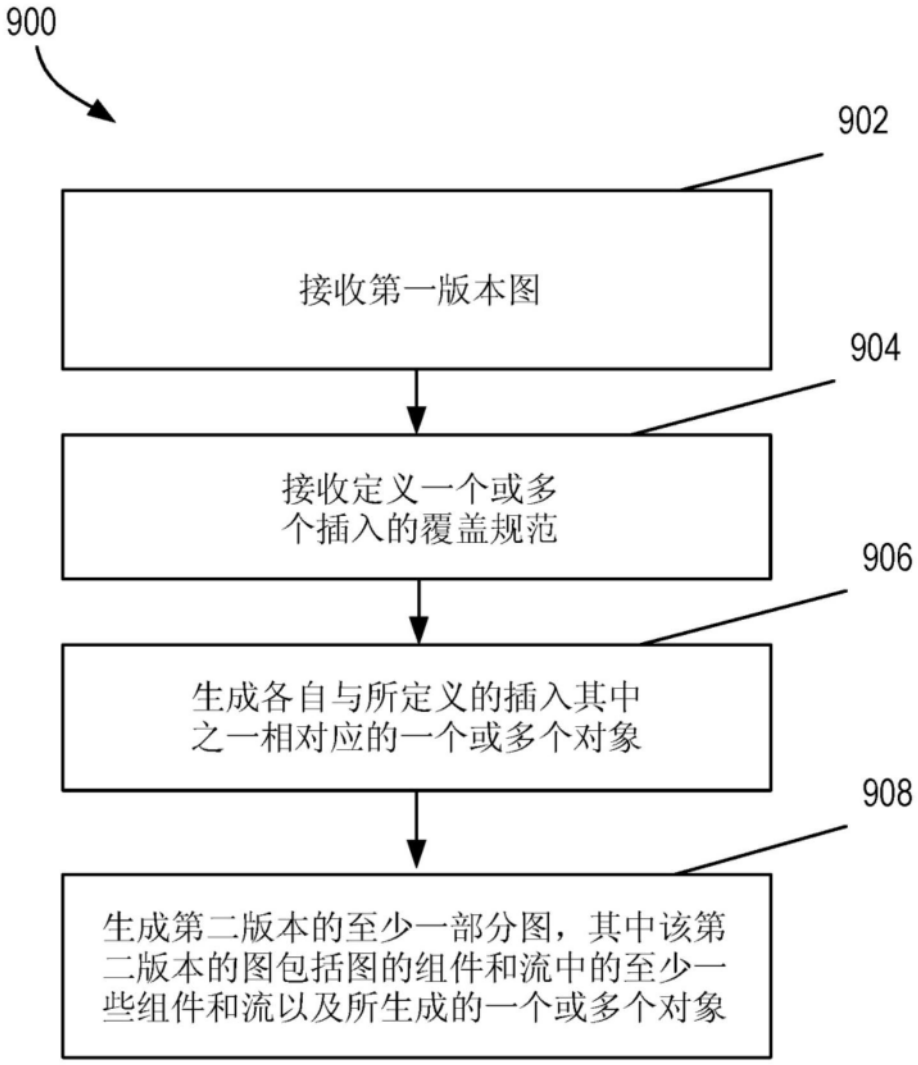

900
902
接收第一版本图
904
接收定义一个或多个插入的覆盖规范
906
生成各自与所定义的插入其中之一相对应的一个或多个对象
908
生成第二版本的至少一部分图，其中该第二版本的图包括图的组件和流中的至少一些组件和流以及所生成的一个或多个对象

图9