



(12) 发明专利申请

(10) 申请公布号 CN 117573459 A

(43) 申请公布日 2024. 02. 20

(21) 申请号 202210927989.5

(22) 申请日 2022.08.03

(71) 申请人 华为技术有限公司

地址 518129 广东省深圳市龙岗区坂田华为总部办公楼

(72) 发明人 汪峰来 陆志浩 郑弦

(74) 专利代理机构 北京中博世达专利商标代理有限公司 11274

专利代理师 申健

(51) Int. Cl.

G06F 11/30 (2006.01)

G06F 11/34 (2006.01)

G06F 18/24 (2023.01)

G06F 18/214 (2023.01)

G06F 11/07 (2006.01)

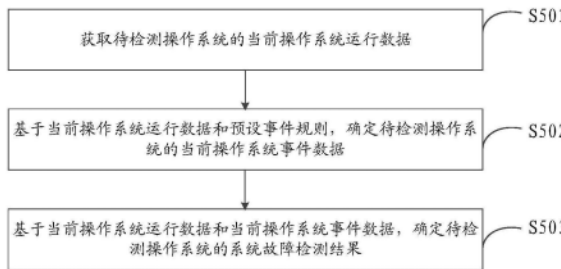
权利要求书3页 说明书33页 附图11页

(54) 发明名称

一种故障检测方法、装置、电子设备及存储介质

(57) 摘要

本申请提供一种故障检测方法、装置、电子设备及存储介质,涉及计算机技术领域,用于快速、准确地确定待检测操作系统的系统故障检测结果。该故障检测方法包括:在获取待检测操作系统的当前操作系统运行数据后,电子设备可以基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。其中,当前操作系统事件数据包括:待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据和当前操作系统的进程事件或线程事件关联的上下文信息。后续,电子设备可以基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果。



1. 一种故障检测方法,其特征在于,应用于电子设备,包括:

获取待检测操作系统的当前操作系统运行数据;

基于所述当前操作系统运行数据和预设事件规则,确定所述待检测操作系统的当前操作系统事件数据;所述当前操作系统事件数据包括:所述待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据和所述当前操作系统的进程事件或线程事件关联的上下文信息;

基于所述当前操作系统运行数据和所述当前操作系统事件数据,确定所述待检测操作系统的系统故障检测结果。

2. 根据权利要求1所述的故障检测方法,其特征在于,所述电子设备包括:系统层处理模块、应用层处理模块和网络层处理模块;所述确定所述待检测操作系统的系统故障检测结果,包括:

所述系统层处理模块确定所述待检测操作系统的系统故障检测结果;

所述方法还包括:

所述系统层处理模块向所述应用层处理模块和所述网络层处理模块发送所述系统故障检测结果;

所述应用层处理模块基于所述系统故障检测结果,确定所述电子设备的应用故障检测结果;

所述网络层处理模块基于所述系统故障检测结果,确定所述电子设备的网络故障检测结果。

3. 根据权利要求1所述的故障检测方法,其特征在于,所述基于所述当前操作系统运行数据和所述当前操作系统事件数据,确定所述待检测操作系统的系统故障检测结果,包括:

基于预设状态规则,对所述当前操作系统事件数据添加状态标识;所述状态标识包括:正常标识和异常标识;

将第一数据输入到预先训练好的故障检测模型中,以得到所述系统故障检测结果;所述第一数据包括所述当前操作系统运行数据和添加所述状态标识后的当前操作系统事件数据;所述故障检测模型为根据所述待检测操作系统的历史操作系统数据训练得到的;所述历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据。

4. 根据权利要求3所述的故障检测方法,其特征在于,还包括:

获取所述历史操作系统数据;

基于预设故障识别算法和所述历史操作系统数据,训练得到所述故障检测模型。

5. 根据权利要求4所述的故障检测方法,其特征在于,所述预设故障识别算法包括:分类算法和参数优化算法;所述基于预设故障识别算法和所述历史操作系统数据,训练得到所述故障检测模型,包括:

基于所述预设状态规则,对所述历史操作系统事件数据添加所述状态标识;

基于所述分类算法和第二数据,对待训练模型进行训练,以得到待调整模型;所述第二数据包括所述历史操作系统运行数据和添加所述状态标识后的历史操作系统事件数据;所述待训练模型包括:待训练的应用特征分类模型和待训练的故障分类模型;

基于所述历史操作系统运行数据和所述参数优化算法,对所述待调整模型进行参数调整,以得到所述故障检测模型;所述故障检测模型包括:应用特征分类模型和故障分类模

型。

6. 根据权利要求5所述的故障检测方法,其特征在于,所述将第一数据输入到预先训练好的故障检测模型中,以得到所述系统故障检测结果,包括:

将所述第一数据输入到所述应用特征分类模型中,以得到应用特征故障检测结果;

将所述第一数据中,与所述应用特征故障检测结果对应的系统数据输入到所述故障分类模型中,以得到所述系统故障检测结果。

7. 根据权利要求1所述的故障检测方法,其特征在于,还包括:

根据所述当前操作系统运行数据和预设依赖规则,构建应用依赖关系;所述应用依赖关系用于表示所述待检测操作系统中各进程或线程及各应用实例之间的相互依赖关系;

根据所述应用依赖关系,构建事件传播关系;所述事件传播关系用于表示所述待检测操作系统中应用事件之间的传播关系;所述应用事件为所述待检测操作系统中的进程或线程及应用实例对应的应用事件;

根据所述事件传播关系,确定引起所述系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程。

8. 根据权利要求7所述的故障检测方法,其特征在于,所述根据所述当前操作系统运行数据和预设依赖规则,构建应用依赖关系,包括:

根据所述当前操作系统运行数据和所述预设依赖规则,构建应用依赖图;所述应用依赖图包括多个应用节点以及应用节点之间的边;所述多个应用节点中的主应用节点用于表示所述待检测操作系统中应用实例的主线程;所述多个应用节点中的从应用节点用于表示:所述待检测操作系统中应用实例的从线程和所述待检测操作系统中应用实例之间的依赖实例;所述多个应用节点中的第一应用节点与第二应用节点之间的边用于表示所述第一应用节点与所述第二应用节点之间存在依赖关系。

9. 根据权利要求8所述的故障检测方法,其特征在于,所述根据所述应用依赖关系,构建事件传播关系,包括:

根据所述应用依赖图,构建事件传播图;所述事件传播图包括与所述多个应用节点一一对应的多个事件节点以及事件节点之间的边;所述多个事件节点用于表示所述待检测操作系统中应用事件对应的应用实例;所述多个事件节点中的第一事件节点与第二事件节点之间的边用于表示所述第一事件节点对应的第一事件与所述第二事件节点对应的第二事件之间存在传播关系。

10. 根据权利要求9所述的故障检测方法,其特征在于,所述根据所述事件传播关系,确定引起所述系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程,包括:

确定与故障事件节点之间具有边的传播事件节点;所述故障事件节点为故障事件对应的事件节点;所述故障事件为所述系统故障检测结果对应的故障事件;

将所述传播事件节点中的传播起始事件节点对应的事件确定为所述根因事件,以及将所述传播起始事件节点对应的应用实例确定为所述根因应用实例,和/或,将所述传播起始事件节点对应的线程确定为所述根因进程或线程。

11. 根据权利要求1-10任一项所述的故障检测方法,其特征在于,引起所述系统故障检测结果中的系统故障包括以下单个类型的故障或组合类型的故障:

网络输入输出IO故障、磁盘输入输出IO故障、调度故障、内存故障、进程或线程故障、文件系统故障、磁盘故障、中央处理器CPU故障和容器故障。

12.一种电子设备,其特征在于,所述电子设备包括:

存储器;

通信接口;

一个或多个处理器;

其中,所述存储器中存储有一个或多个计算机程序,所述一个或多个计算机程序包括指令,当所述指令被所述电子设备执行时,使得所述电子设备执行如权利要求1-11中任一项所述的故障检测方法。

13.一种计算机可读存储介质,其特征在于,包括计算机指令,当所述计算机指令在电子设备上运行时,使得所述电子设备执行如权利要求1-11中任一项所述的故障检测方法。

14.一种计算机程序产品,包括指令,其特征在于,当所述指令在电子设备上运行时,使得所述电子设备执行如权利要求1-11中任一项所述的故障检测方法。

一种故障检测方法、装置、电子设备及存储介质

技术领域

[0001] 本申请涉及计算机技术领域,尤其涉及一种故障检测方法、装置、电子设备及存储介质。

背景技术

[0002] 大规模云服务中的云服务器在系统架构上设计了大量的可靠性、可用性等机制。在这些机制的作用下,云服务器可以容忍一些系统故障继续运行。此时,在云服务器的应用性能管理(Application Performance Monitoring,APM)层,云服务器检测到的云服务应用是无故障的。但是,在云服务器的互联网技术基础设施管理(Internet Technology Infrastructure Monitoring,ITIM)层,即操作系统(Operating System,OS)层,云服务器会检测到云服务的操作系统可能存在故障隐患。这种以APM层、ITIM层等不同检测视角造成的检测差异称作灰度故障。

[0003] 目前,在检测灰度故障时,云服务器通常通过“层层下钻”的方式进行故障诊断,即云服务器依次判断APM层的应用程序(Application,APP)是否故障、网络性能管理与诊断(Network Performance Monitoring and Diagnostics,NPMD)层的网络功能是否故障,以及ITIM层的操作系统是否故障。

[0004] 而在判断ITIM层的操作系统是否故障时,由于云服务器只能获取到操作系统的中央处理器(central processing unit,CPU)占用率、当前磁盘空间大小等基础信息,因此,通用的故障检测方法还需要结合人工经验,进一步根据上述基础信息确定操作系统是否故障,耗时耗力,效率较低。

发明内容

[0005] 本申请提供一种故障检测方法、装置、电子设备及存储介质,涉及计算机技术领域,用于快速、准确地检测操作系统中的故障。

[0006] 为达到上述目的,本申请采用如下技术方案:

[0007] 第一方面,本申请提供一种故障检测方法,应用于电子设备,包括:

[0008] 在获取待检测操作系统的当前操作系统运行数据后,电子设备可以基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。其中,当前操作系统事件数据包括:待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据和当前操作系统的进程事件或线程事件关联的上下文信息。后续,电子设备可以基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果。

[0009] 上述当前操作系统运行数据为待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据。也就是说,这种数据一般仅仅表示一个进程事件或线程事件的指标(例如CPU利用率等),因此,当前操作系统运行数据为指标类型的数据。而上述当前操作系统事件数据不仅包括当前操作系统的进程事件或线程事件运行过程中产生的数

据,还包括当前操作系统进程事件或线程事件关联的上下文信息。也就是说,这种数据不仅可以表示一个进程事件或线程事件的指标,还可以表示这个进程事件或线程事件的事件内容(例如该线程事件为主线程事件),和/或,事件状态(例如该线程事件为异常状态事件)等,因此,当前操作系统事件数据为事件类型的数据。

[0010] 在这种情况下,电子设备在检测操作系统层级的故障时,不仅可以获取到指标类型的当前操作系统运行数据,还可以基于当前操作系统运行数据确定事件类型的当前操作系统事件数据。后续,电子设备进行故障检测的过程中,相比通用技术中仅通过基础信息(即指标类型的当前操作系统运行数据)结合人工经验进行故障检测的方法,本申请实施例提供的故障检测方法可以基于当前操作系统运行数据和当前操作系统事件数据,快速确定待检测操作系统的系统故障检测结果,无需人工经验的介入,提高了故障检测的效率。

[0011] 其次,由于当前操作系统事件数据包括当前操作系统进程事件或线程事件关联的上下文信息,因此,电子设备在进行故障检测的过程中,相比通用技术仅通过基础信息结合人工经验进行故障检测,本申请实施例提供的故障检测方法可以挖掘当前操作系统事件数据中的上下文信息,从而基于上下文信息,获取进程事件或者线程事件的上下文关联事件的相关内容,进而根据这些内容结合当前操作系统运行数据,准确的确定系统故障检测结果,提高了故障检测的准确度。

[0012] 再次,由于当前操作系统运行数据和当前操作系统事件数据具备细粒度(可以精细到待检测系统的线程事件或进程事件)、宽范围检测数据(不仅包括指标类型的数据,还包括事件类型的数据)的特点,因此,本申请实施例提供的故障检测方法也可以检测到更细粒度的故障检测结果(例如具体线程事件或进程事件的故障检测结果),进一步提高了故障检测的准确度。

[0013] 此外,本申请实施例提供的故障检测方法无需在电子设备的操作系统层级添加各种故障检测工具,仅需操作系统自身具备的数据采集功能和数据处理功能,便可以快速检测故障,实现了基于操作系统轻量级的故障检测。

[0014] 在一种可能的实现方式中,电子设备包括:系统层处理模块、应用层处理模块和网络层处理模块;上述确定待检测操作系统的系统故障检测结果的方法具体包括:系统层处理模块确定待检测操作系统的系统故障检测结果。

[0015] 本申请实施例提供的故障检测方法还包括:系统层处理模块可以向应用层处理模块和网络层处理模块发送系统故障检测结果。相应的,应用层处理模块基于系统故障检测结果,确定电子设备的应用故障检测结果。相应的,网络层处理模块基于系统故障检测结果,确定电子设备的网络故障检测结果。

[0016] 由上可知,电子设备中的系统层处理模块可以在确定系统故障检测结果后,向应用层处理模块和网络层处理模块发送系统故障检测结果,以使得应用层处理模块基于系统故障检测结果,确定电子设备的应用故障检测结果,以及网络层处理模块基于系统故障检测结果,确定电子设备的网络故障检测结果,从而实现了系统层处理模块分别与应用层和网络层之间通过松耦合的方式,快速、准确的确定待检测操作系统的灰度故障的故障检测结果,提高了故障检测效率。

[0017] 在一种可能的实现方式中,上述基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果的方法具体包括:基于预设状态规则,对当

前操作系统事件数据添加状态标识,并将第一数据输入到预先训练好的故障检测模型中,以得到系统故障检测结果。其中,状态标识包括:正常标识和异常标识;第一数据包括当前操作系统运行数据和添加状态标识后的当前操作系统事件数据;故障检测模型为根据待检测操作系统的历史操作系统数据训练得到的;历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据。

[0018] 由上可知,由于故障检测模型为根据待检测操作系统的历史操作系统数据训练得到的,历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据,因此,电子设备确定待检测操作系统的系统故障检测结果时,可以通过故障检测模型,快速、准确的确定系统故障检测结果,提高了故障检测效率。

[0019] 在一种可能的实现方式中,该故障检测方法还包括:获取历史操作系统数据,并基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型。

[0020] 由上可知,电子设备可以基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型,以便于后续通过故障检测模型,快速、准确的确定系统故障检测结果,提高了故障检测效率。

[0021] 在一种可能的实现方式中,预设故障识别算法可以包括:分类算法和参数优化算法;上述基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型的方法具体包括:基于预设状态规则,对历史操作系统事件数据添加状态标识,并基于分类算法和第二数据,对待训练模型进行训练,以得到待调整模型。其中,第二数据包括历史操作系统运行数据和添加状态标识后的历史操作系统事件数据;待训练模型包括:待训练的应用特征分类模型和待训练的故障分类模型。后续,可以基于历史操作系统运行数据和参数优化算法,对待调整模型进行参数调整,以得到故障检测模型;故障检测模型包括:应用特征分类模型和故障分类模型。

[0022] 由上可知,由于预设故障识别算法可以包括分类算法和参数优化算法,因此,分类算法可以用于对故障检测模型的初始模型进行分类训练,参数优化算法可以对分类训练过程中的模型进行参数调整,从而得到故障检测模型,以便于后续通过故障检测模型,快速、准确的确定系统故障检测结果,提高了故障检测效率。

[0023] 在一种可能的实现方式中,上述将第一数据输入到预先训练好的故障检测模型中,以得到系统故障检测结果的方法具体包括:将第一数据输入到应用特征分类模型中,以得到应用特征故障检测结果,并将第一数据中,与应用特征故障检测结果对应的系统数据输入到故障分类模型中,以得到系统故障检测结果。

[0024] 由上可知,由于故障检测模型包括应用特征分类模型和故障分类模型,因此,电子设备可以依次根据应用特征分类模型和故障分类模型确定不同的应用特征,以及每个应用特征对应的故障分类,提高了故障检测的准确度。

[0025] 在一种可能的实现方式中,该故障检测方法还包括:

[0026] 根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系,以及根据应用依赖关系,构建事件传播关系。其中,应用依赖关系用于表示待检测操作系统中各进程或线程及各应用实例之间的相互依赖关系;事件传播关系用于表示待检测操作系统中应用事件之间的传播关系;应用事件为待检测操作系统中的进程或线程及应用实例对应的应用事件。后续,可以根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,

以及根因应用实例和/或根因进程或线程。

[0027] 由上可知,在确定系统故障检测结果后,为了对待检测操作系统中发生故障的事件或应用进行根因定位,电子设备还可以构建应用依赖关系和事件传播关系,以此来进一步的实现故障根因定位。

[0028] 在一种可能的实现方式中,上述根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系的方法具体包括:根据当前操作系统运行数据和预设依赖规则,构建应用依赖图;应用依赖图包括多个应用节点以及应用节点之间的边;多个应用节点中的主应用节点用于表示待检测操作系统中应用实例的主线程;多个应用节点中的从应用节点用于表示:待检测操作系统中应用实例的从线程和待检测操作系统中应用实例之间的依赖实例;多个应用节点中的第一应用节点与第二应用节点之间的边用于表示第一应用节点与第二应用节点之间存在依赖关系。

[0029] 由上可知,电子设备可以以应用依赖图的显示形式,构建应用依赖关系,以便于后续可以方便、快捷的根据应用依赖图进行根因定位。

[0030] 在一种可能的实现方式中,上述根据应用依赖关系,构建事件传播关系,包括:根据应用依赖图,构建事件传播图;事件传播图包括与多个应用节点一一对应的多个事件节点以及事件节点之间的边;多个事件节点用于表示待检测操作系统中应用事件对应的应用实例;多个事件节点中的第一事件节点与第二事件节点之间的边用于表示第一事件节点对应的第一事件与第二事件节点对应的第二事件之间存在传播关系。

[0031] 由上可知,电子设备可以以事件传播图的显示形式,构建事件传播关系,以便于后续可以方便、快捷的根据事件传播图进行根因定位。

[0032] 在一种可能的实现方式中,上述根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程的方法具体包括:

[0033] 确定与故障事件节点之间具有边的传播事件节点,并将传播事件节点中的传播起始事件节点对应的事件确定为根因事件,以及将传播起始事件节点对应的应用实例确定为根因应用实例,和/或,将传播起始事件节点对应的线程确定为根因进程或线程。其中,故障事件节点为故障事件对应的事件节点;故障事件为系统故障检测结果对应的故障事件。

[0034] 由上可知,电子设备可以根据应用依赖图和事件传播图,快速、准确的确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因线程。

[0035] 在一种可能的实现方式中,引起系统故障检测结果中的系统故障包括以下单个类型的故障或组合类型的故障:网络输入输出IO故障、磁盘输入输出IO故障、调度故障、内存故障、进程或线程故障、文件系统故障、磁盘故障、中央处理器CPU故障和容器故障。

[0036] 由上可知,电子设备可以确定各种单个类型或者组合类型的故障的系统故障检测结果,从而可以检测到各种类型的系统灰度故障。

[0037] 第二方面,本申请提供一种故障检测装置,包括:获取单元和处理单元;获取单元,用于获取待检测操作系统的当前操作系统运行数据;处理单元,用于基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据;当前操作系统事件数据包括:待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据和当前操作系统的进程事件或线程事件关联的上下文信息;处理单元,还用于基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结

果。

[0038] 在一种可能的实现方式中,电子设备包括:系统层处理模块、应用层处理模块和网络层处理模块。系统层处理模块用于确定待检测操作系统的系统故障检测结果;系统层处理模块还用于向应用层处理模块和网络层处理模块发送系统故障检测结果;应用层处理模块用于基于系统故障检测结果,确定电子设备的应用故障检测结果;网络层处理模块用于基于系统故障检测结果,确定电子设备的网络故障检测结果。

[0039] 在一种可能的实现方式中,处理单元,具体用于:基于预设状态规则,对当前操作系统事件数据添加状态标识;状态标识包括:正常标识和异常标识;将第一数据输入到预先训练好的故障检测模型中,以得到系统故障检测结果;第一数据包括当前操作系统运行数据和添加状态标识后的当前操作系统事件数据;故障检测模型为根据待检测操作系统的历史操作系统数据训练得到的;历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据。

[0040] 在一种可能的实现方式中,获取单元,还用于获取历史操作系统数据;处理单元,还用于基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型。

[0041] 在一种可能的实现方式中,预设故障识别算法包括:分类算法和参数优化算法;处理单元,具体用于:基于预设状态规则,对历史操作系统事件数据添加状态标识;基于分类算法和第二数据,对待训练模型进行训练,以得到待调整模型;第二数据包括历史操作系统运行数据和添加状态标识后的历史操作系统事件数据;待训练模型包括:待训练的应用特征分类模型和待训练的故障分类模型;基于历史操作系统运行数据和参数优化算法,对待调整模型进行参数调整,以得到故障检测模型;故障检测模型包括:应用特征分类模型和故障分类模型。

[0042] 在一种可能的实现方式中,处理单元,具体用于:将第一数据输入到应用特征分类模型中,以得到应用特征故障检测结果;将第一数据中,与应用特征故障检测结果对应的系统数据输入到故障分类模型中,以得到系统故障检测结果。

[0043] 在一种可能的实现方式中,处理单元,还用于根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系;应用依赖关系用于表示待检测操作系统中各进程或线程及各应用实例之间的相互依赖关系;处理单元,还用于根据应用依赖关系,构建事件传播关系;事件传播关系用于表示待检测操作系统中应用事件之间的传播关系;应用事件为待检测操作系统中的进程或线程及应用实例对应的应用事件;处理单元,还用于根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程。

[0044] 在一种可能的实现方式中,处理单元,具体用于:根据当前操作系统运行数据和预设依赖规则,构建应用依赖图;应用依赖图包括多个应用节点以及应用节点之间的边;多个应用节点中的主应用节点用于表示待检测操作系统中应用实例的主线程;多个应用节点中的从应用节点用于表示:待检测操作系统中应用实例的从线程和待检测操作系统中应用实例之间的依赖实例;多个应用节点中的第一应用节点与第二应用节点之间的边用于表示第一应用节点与第二应用节点之间存在依赖关系。

[0045] 在一种可能的实现方式中,处理单元,具体用于:根据应用依赖图,构建事件传播图;事件传播图包括与多个应用节点一一对应的多个事件节点以及事件节点之间的边;多

个事件节点用于表示待检测操作系统中应用事件对应的应用实例；多个事件节点中的第一事件节点与第二事件节点之间的边用于表示第一事件节点对应的第一事件与第二事件节点对应的第二事件之间存在传播关系。

[0046] 在一种可能的实现方式中,处理单元,具体用于:确定与故障事件节点之间具有边的传播事件节点;故障事件节点为故障事件对应的事件节点;故障事件为系统故障检测结果对应的故障事件;将传播事件节点中的传播起始事件节点对应的事件确定为根因事件,以及将传播起始事件节点对应的应用实例确定为根因应用实例,和/或,将传播起始事件节点对应的进程或线程确定为根因进程或线程。

[0047] 在一种可能的实现方式中,引起系统故障检测结果中的系统故障包括以下单个类型的故障或组合类型的故障:网络输入输出IO故障、磁盘输入输出IO故障、调度故障、内存故障、进程或线程故障、文件系统故障、磁盘故障、中央处理器CPU故障和容器故障。

[0048] 第三方面,本申请提供一种电子设备,可以包括:处理器和用于存储处理器可执行指令的存储器;其中,处理器被配置为执行所述指令,以实现上述第一方面中任一种可能的实现方式中的故障检测方法。

[0049] 第四方面,本申请提供一种计算机可读存储介质,计算机可读存储介质上存储有指令,当所述计算机可读存储介质中的指令由电子设备的处理器执行时,使得所述电子设备能够执行上述第一方面中任一种可能的实现方式中的故障检测方法。

[0050] 第五方面,本申请提供一种计算机程序产品,该计算机程序产品包括计算机指令,当计算机指令在电子设备的处理器上运行时,使得电子设备的处理器执行如第一方面中任一种可能的实现方式中的实现方式所述的故障检测方法。

[0051] 应当理解的是,以上的一般描述和后文的细节描述仅是示例性和解释性的,并不能限制本申请。

[0052] 可以理解地,上述各个方面所提供的故障检测装置、电子设备、计算机可读存储介质以及计算机程序产品均应用于上文所提供的故障检测方法,因此,其所能达到的有益效果可参考上文所提供的故障检测的方法中的有益效果,此处不再赘述。

附图说明

[0053] 图1为本申请实施例提供的一种灰度故障的场景示意图;

[0054] 图2为通用技术提供的故障检测方法的流程示意图;

[0055] 图3为本申请实施例提供的一种故障检测系统的结构示意图;

[0056] 图4为本申请实施例提供的一种电子设备的内部模块结构框图;

[0057] 图5为本申请实施例提供的一种故障检测方法的流程示意图一;

[0058] 图6为本申请实施例提供的一种故障检测方法的流程示意图二;

[0059] 图7为本申请实施例提供的一种故障检测方法的流程示意图三;

[0060] 图8为本申请实施例提供的一种故障检测方法的流程示意图四;

[0061] 图9为本申请实施例提供的一种故障检测方法的流程示意图五;

[0062] 图10为本申请实施例提供的一种故障检测方法的流程示意图六;

[0063] 图11为本申请实施例提供的一种故障检测方法的流程示意图七;

[0064] 图12为本申请实施例提供的一种故障检测方法的流程示意图八;

- [0065] 图13为本申请实施例提供一种故障检测方法的流程图九;
- [0066] 图14为本申请实施例提供一种故障检测方法的流程图十;
- [0067] 图15为本申请实施例提供一种故障检测方法的流程图十一;
- [0068] 图16为本申请实施例提供一种故障检测方法的流程图十二;
- [0069] 图17为本申请实施例提供一种故障检测方法的流程图十三;
- [0070] 图18为本申请实施例提供一种故障检测方法的流程图十四;
- [0071] 图19为本申请实施例提供一种故障检测方法的流程图十五;
- [0072] 图20为本申请实施例提供一种故障检测装置的结构示意图;
- [0073] 图21示出了本申请实施例提供一种终端的结构示意图;
- [0074] 图22示出了本申请实施例提供一种服务器的结构示意图。

具体实施方式

[0075] 下面将结合本申请实施例中的附图,对本申请实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅仅是本申请一部分实施例,而不是全部的实施例。基于本申请中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其它实施例,都属于本申请保护的范围。

[0076] 本申请中字符“/”,一般表示前后关联对象是一种“或者”的关系。例如,A/B可以理解为A或者B。

[0077] 术语“第一”、“第二”仅用于描述目的,而不能理解为指示或暗示相对重要性或者隐含指明所指示的技术特征的数量。由此,限定有“第一”、“第二”的特征可以明示或者隐含地包括一个或者更多个该特征。在本实施例的描述中,除非另有说明,“多个”的含义是两个或两个以上。

[0078] 此外,本申请的描述中所提到的术语“包括”和“具有”以及它们的任何变形,意图在于覆盖不排他的包含。例如包含了一系列步骤或模块的过程、方法、系统、产品或设备没有限定于已列出的步骤或模块,而是可选地还包括其他没有列出的步骤或模块,或可选地还包括对于这些过程、方法、产品或设备固有的其它步骤或模块。

[0079] 另外,在本申请实施例中,“示例性的”、或者“例如”等词用于表示作例子、例证或说明。本申请中被描述为“示例性的”或“例如”的任何实施例或设计方案不应被解释为比其它实施例或设计方案更优选或更具优势。确切而言,使用“示例性的”、或者“例如”等词旨在以具体方式呈现概念。

[0080] 在对本申请提供的故障检测方法进行详细介绍之前,先对本申请涉及的相关要素、应用场景、实施环境进行简单介绍。

[0081] 首先,对本申请涉及的相关要素进行简单介绍。

[0082] 指标数据(metrics):操作系统中一些可进行聚合计算的原子型数据。

[0083] 示例性的,指标数据可以是CPU占用情况、系统内存占用、接口响应时间、接口响应每秒查询率(queries per second,QPS)等。

[0084] 又一示例性的,指标数据还可以是用于表示任务队列当前深度的度量值,在元素入队或出队时可以被更新。

[0085] 又一示例性的,指标数据还可以是用于表示超文本传输协议(Hyper Text

Transfer Protocol, HTTP) 请求个数的计数器, 在新请求到来时进行累加。

[0086] 这些指标数据都是根据时间序列存储的数据值, 可以在一段时间内进行一些求和、求平均、百分位等聚合计算, 并用于后续操作系统的数据分析与整理。

[0087] 日志数据(logging): logging是操作系统运行时发生的一个个事件的记录, 可以用于记录操作系统的离散事件, 以及为反馈排查问题提供详细的信息。

[0088] 链路数据(Tracing): 用于记录消息请求范围内的调用信息。链路数据是排查系统性能问题的利器。链路数据不仅有利于梳理接口及服务间调用的关系, 还有助于排查慢请求产生的原因或异常发生的原因。

[0089] 示例性的, 链路数据可以记录一次远程方法调用的执行过程和耗时。

[0090] 又一示例性的, 在微服务中一般有多个调用信息, 如从最外层的网关开始, A服务调用B服务, 调用数据库、缓存等。在操作系统中, 链路数据可以清楚展现某条调用链中从主调方到被调方内部所有的调用信息。

[0091] 无状态数据(Stateless): 即本申请实施例中的操作系统运行数据, 用于表示操作系统中, 一次操作对应的数据。本申请实施例中, 电子设备中部署有用于采集数据的采集探针。采集探针采集到的操作系统运行数据必须全部来自于无状态数据所携带的信息以及可以被所有数据所使用的公共信息。

[0092] 有状态数据(Stateful): 即本申请实施例中的操作系统事件数据, 用于表示操作系统中, 有数据存储功能的数据。这种数据可以存储该数据对应的进程事件或线程事件的事件内容(例如该线程事件为主线程事件), 和/或, 事件状态(例如该线程事件为异常状态事件)等。本申请实施例中, 采集探针采集到的操作系统事件数据一般包括该操作系统事件数据的相关信息, 即上下文信息。

[0093] 数据采集探针在采集有状态数据和无状态数据时, 可以通过确定两个来自相同标签的数据在操作系统内是否具备上下文关系, 来进行状态化的判断, 即具备上下文关系的数据为有状态数据, 不具备上下文关系的数据为无状态数据。

[0094] 如背景技术所描述, 大规模云服务中的云服务器在系统架构上设计了大量的可靠性、可用性等机制。在这些机制的作用下, 云服务器可以容忍一些系统故障继续运行。此时, 在云服务器的APM层, 云服务器检测到的云服务应用是无故障的。但是, 在云服务器的ITIM层, 即OS层, 云服务器会检测到云服务的操作系统可能存在故障隐患。这种以APM层、ITIM层等不同检测视角造成的检测差异称作灰度故障。

[0095] 灰色故障通常包括网络输入输出(Input/Output, IO)故障(例如随机丢包)、磁盘IO故障(例如产生磁盘碎片)、内存抖动/泄漏故障、CPU调度/干扰故障、容量压力故障等造成操作系统的性能下降的非致命异常故障。据统计分析, 灰度故障是大规模云服务生产环境中影响最大的一类故障。

[0096] 图1示出了本申请实施例提供的一种灰度故障的场景示意图。如图1所示, 在APM层, APM层的故障检测模块(observation) 1可以获取APM层的应用程序(application, APP) 1、APP 2、APP 3等APP的检测报告(probe report)。

[0097] 相应的, 在NPMD层, NPMD层的observation 2可以获取NPMD层的网络设备(network devices)的检测报告。

[0098] 相应的, 在ITIM层, ITIM层的observation 3可以获取ITIM层的OS 1、OS 2、OS 3等

OS的检测报告。

[0099] 由于大规模云服务中的云服务器在系统架构上设计了大量的可靠性、可用性等机制,因此,APM层的observation 1基于获取的检测报告确定的故障检测结果与ITIM层基于获取的检测报告确定的故障检测结果不同,即APM层、ITIM层等不同检测视角造成的检测差异即为灰度故障。

[0100] 目前,在检测灰度故障时,智能运维(Artificial intelligence for IT Operations,AIOps)的故障检测方法是以APM层为中心,自上而下(Top-Down)的技术体系进行故障检测。

[0101] 具体的,通用的故障检测方法以APM层为视角,以感知APM层的APP的服务等级指标(Service Level Indicator,SLI)为入口,“层层下钻”的方式进行故障诊断,即云服务器依次判断APM层的APP是否故障、NPMD层的网络功能是否故障,以及ITIM层的操作系统是否故障。

[0102] 在这种情况下,通用技术通常使用数十种管理和监控工具,通常是APM工具、NPMD工具和无数特定于孤岛的ITIM工具的组合。在这种情况下,因为APM工具、NPMD工具和ITIM工具不相关,即它们本质上使用的计算机语言不同,导致这些工具无法获得有关其他层级正在发生的事件的事件信息,只能实现APM层、NPMD层、ITIM层各自层级的检测与分析。

[0103] 而在判断ITIM层的操作系统是否故障时,由于云服务器只能获取到操作系统的CPU利用率、当前磁盘空间大小等基础信息,因此,通用的故障检测方法还需要结合人工经验,进一步根据上述基础信息确定操作系统是否故障,耗时耗力,效率较低。

[0104] 图2示出了通用技术中,一种灰度故障检测方法的流程示意图。如图2所示,通用的灰度故障检测方法具体包括:

[0105] S1、APM层的检测模块检测APP的应用质量是否异常。

[0106] 其中,APP的应用质量异常具体包括APP的性能下降、用户体验不佳(例如使用卡顿)等。

[0107] S2、NPMD层的检测模块检测网络设备是否有网络问题。

[0108] S3、获取ITIM层的通用检测数据。

[0109] 可选的,获取ITIM层的通用检测数据时,可以通过ITIM层的数据采集模块获取,也可以人工获取。

[0110] S4、基于APM层的检测结果、NPMD层的检测结果、ITIM层的检测结果和运维专家经验确定灰度故障检测结果。

[0111] S5、基于灰度故障检测结果和运维专家经验确定故障诊断结果。

[0112] 由上可知,通用的故障检测方法还需要结合人工经验,进一步根据上述基础信息确定操作系统是否故障,耗时耗力,效率较低。

[0113] 针对上述问题,本申请提供一种故障检测方法,应用于电子设备,包括:在获取待检测操作系统的当前操作系统运行数据后,电子设备可以基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。后续,电子设备可以基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果。

[0114] 上述当前操作系统运行数据为待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据。也就是说,这种数据一般仅仅表示一个进程事件或线程事

件的指标(例如CPU利用率等),因此,当前操作系统运行数据为指标类型的数据。而上述当前操作系统事件数据不仅包括当前操作系统的进程事件或线程事件运行过程中产生的数据,还包括当前操作系统进程事件或线程事件关联的上下文信息。也就是说,这种数据不仅可以表示一个进程事件或线程事件的指标,还可以表示这个进程事件或线程事件的事件内容(例如该线程事件为主线程事件),和/或,事件状态(例如该线程事件为异常状态事件)等,因此,当前操作系统事件数据为事件类型的数据。

[0115] 在这种情况下,电子设备在检测操作系统层级的故障时,不仅可以获取到指标类型的当前操作系统运行数据,还可以基于当前操作系统运行数据确定事件类型的当前操作系统事件数据。后续,电子设备进行故障检测的过程中,相比通用技术中仅通过基础信息(即指标类型的当前操作系统运行数据)结合人工经验进行故障检测的方法,本申请实施例提供的故障检测方法可以基于当前操作系统运行数据和当前操作系统事件数据,快速确定待检测操作系统的系统故障检测结果,无需人工经验的介入,提高了故障检测的效率。

[0116] 其次,由于当前操作系统事件数据包括当前操作系统进程事件或线程事件关联的上下文信息,因此,电子设备在进行故障检测的过程中,相比通用技术仅通过基础信息结合人工经验进行故障检测,本申请实施例提供的故障检测方法可以挖掘当前操作系统事件数据中的上下文信息,从而基于上下文信息,获取进程事件或者线程事件的上下文关联事件的相关内容,进而根据这些内容结合当前操作系统运行数据,准确的确定系统故障检测结果,提高了故障检测的准确度。

[0117] 再次,由于当前操作系统运行数据和当前操作系统事件数据具备细粒度(可以精细到待检测系统的线程事件或进程事件)、宽范围检测数据(不仅包括指标类型的数据,还包括事件类型的数据)的特点,因此,本申请实施例提供的故障检测方法也可以检测到更细粒度的故障检测结果(例如具体线程事件或进程事件的故障检测结果),进一步提高了故障检测的准确度。

[0118] 此外,本申请实施例提供的故障检测方法无需在电子设备的操作系统层级添加各种故障检测工具,仅需操作系统自身具备的数据采集功能和数据处理功能,便可以快速检测故障,实现了基于操作系统轻量级的故障检测。

[0119] 下面将结合附图对本实施例的实施方式进行详细描述。

[0120] 本申请实施例提供的故障检测方法可以应用于故障检测系统。图3示出了本申请实施例提供的一种故障检测系统的结构示意图。

[0121] 如图3所示,本申请实施例提供的故障检测系统包括云终端301、网络设备302和云服务器303。其中,网络设备302分别与云终端301和云服务器303通信连接。其中,云终端301又叫虚拟终端,它是基于可靠、高速的网络通信、海量存储的实现以及多进程或线程、高效率CPU技术的前提下,利用远程虚拟化技术实现客户终端设备的管理解决方案,通过将远程终端设备硬件、运行软件及硬盘数据等分离成不同的传输层次,构成主机与客户端的动态架构,从而实现集中管理和动态分组分权管理的统一化管理平台。本申请实施例中,云终端301可以请求云服务器303提供的服务资源,并向用户提供对应的云服务。

[0122] 云服务器是一种简单高效、安全可靠、处理能力可弹性伸缩的计算服务。本申请实施例中,云服务器303可以向云终端301提供服务资源,以使得云终端301向用户提供对应的云服务。

[0123] 图3中的云终端301可以是指向用户提供语音和/或数据连通性的设备,具有无线连接功能的手持式设备、或连接到无线调制解调器的其他处理设备。云终端301可以经无线接入网(radio access network,RAN)与一个或多个核心网进行通信。无线终端可以是移动终端,如移动电话(或称为“蜂窝”电话)和具有移动终端的计算机,也可以是便携式、袖珍式、手持式、计算机内置的或者车载的移动装置,它们与无线接入网交换语言和/或数据,例如,手机、平板电脑、笔记本电脑、台式电脑、上网本、个人数字助理(personal digital assistant,PDA)。

[0124] 图3中的网络设备302可以是无线通信的基站或基站控制器等。在本申请实施例中,所述基站可以是全球移动通信系统(global system for mobile communication,GSM),码分多址(code division multiple access,CDMA)中的基站(base transceiver station,BTS),宽带码分多址(wideband code division multiple access,WCDMA)中的基站(node B),物联网(internet of things,IoT)或者窄带物联网(narrow band-internet of things,NB-IoT)中的基站(eNB),未来5G移动通信网络或者未来演进的公共陆地移动网络(public land mobile network,PLMN)中的基站,本申请实施例对此不作任何限制。

[0125] 图3中的云服务器303可以是服务器集群(由多个服务器组成)中的一个服务器,也可以是该服务器中的芯片,还可以是该服务器中的片上系统,还可以通过部署在物理机上的虚拟机(virtual machine,VM)实现,本申请实施例对此不作限定。

[0126] 在一种可以实现的方式中,本申请实施例提供的故障检测方法主要用于检测云服务器303操作系统层级的灰度故障。

[0127] 在又一种可以实现的方式中,本申请实施例提供的故障检测方法也可以用于检测云终端301操作系统层级的灰度故障,还可以用于检测其他具有操作系统层级、网络层级和应用层级的电子设备灰度故障,本申请对此不作限定。

[0128] 为了便于描述,本申请实施例以检测云服务器303操作系统层级的灰度故障为例进行说明。

[0129] 如图3所示,云服务器303可以包括APM层、NPMD层和ITIM层。

[0130] APM层的observation 1可以获取APM层的APP 1、APP 2、APP 3等APP的检测报告,并基于该检测包括确定APM层的应用故障检测结果。

[0131] 相应的,NPMD层的observation 2可以获取NPMD层的网络设备的检测报告,并基于该检测包括确定NPMD层的网络故障检测结果。

[0132] 本申请实施例中,ITIM层除了提供通用的基础设施检测能力,还具备操作系统灰度故障识别与诊断能力。

[0133] 也就是说,ITIM层的observation 3可以根据本申请实施例提供的故障检测方法,获取ITIM层的OS 1、OS 2、OS 3等OS的检测报告,并基于该检测包括确定ITIM层的故障检测结果。

[0134] 后续,ITIM层的observation 3可以向APM层的observation 1、NPMD层的observation 2发送ITIM层的系统故障检测结果,以使得APM层的observation 1、NPMD层的observation 2结合ITIM层的系统故障检测结果,确定灰度故障。

[0135] 相应的,APM层的observation 1也可以向NPMD层的observation 2、ITIM层的observation 3发送APM层的应用故障检测结果,以使得NPMD层的observation 2、ITIM层的

observation3结合APM层的应用故障检测结果,确定灰度故障。

[0136] 相应的,NPMD层的observation 2也可以向APM层的observation 1、ITIM层的observation3发送NPMD层的网络故障检测结果,以使得APM层的observation 1、ITIM层的observation3结合NPMD层的网络故障检测结果,确定灰度故障。

[0137] 这样,本申请实施例提供的故障检测方法可以使得APM层、NPMD层、ITIM层的孤岛检测工具相互融合。将通用技术中,以应用为视角自上而下的智能运维的故障检测方法,与本申请实施例提供的以操作系统为视角自下而上主动识别、定位灰度故障的方案相结合,减少人工介入,提高运维效率。

[0138] 图4是本申请实施例提供的电子设备的内部模块结构框图。该电子设备可以采集操作系统中的指标数据、日志数据、链路数据等数据,并基于指标数据、日志数据、链路数据等信息确定事件(Events)等数据,然后进行数据分析识别灰度故障。后续,通过构建应用依赖图及事件传播图进行根因推理,实现对操作系统健康度、灰度故障、根因定位的检测。

[0139] 该电子设备可以是图3中的云服务器303,也可以是云终端301,还可以是具有操作系统层级、网络层级和应用层级的电子设备,本申请实施例对此不作限定。

[0140] 如图4所示,本申请实施例提的电子设备的内部模块可以包括:数据采集模块、灰度故障识别模块、根因分析模块、检测模块和数据存储模块。

[0141] 其中,数据采集模块用于采集操作系统中的指标数据、日志数据、链路数据等数据(即本申请实施例中的操作系统运行数据),并基于指标数据、日志数据、链路数据等数据生成事件数据(即本申请实施例中的操作系统事件数据)。

[0142] 事件数据用于表示基于指标数据、日志数据、链路数据等数据生成的实时异常事件。

[0143] 灰度故障识别模块用于基于指标数据、日志数据、链路数据等数据生成的事件数据,以及多维长周期时序数据(即多个时间段下的指标数据、日志数据、链路数据等数据)的关联分析,完成灰度故障的识别。

[0144] 根因分析模块用于通过静态的服务部署方式生成应用依赖图,并基于领域知识及事件动态生成事件传播图,并基于构建好的应用依赖图和事件传播图对识别到的灰度故障进行根因推理。

[0145] 检测模块用于提供对操作系统健康度、异常事件、故障根因的展示功能或数据接口。

[0146] 数据存储模块用于存储数据采集模块采集到的数据、灰度故障识别模块识别出的灰度故障以及根因分析模块确定的故障根因,并向检测模块提供检测模块所需的数据。

[0147] 基于上述电子设备中的各个模块的功能,下面对本申请实施例提供的技术方案进行详细说明。

[0148] 本申请实施例提供的故障检测方法可以应用于电子设备。

[0149] 下面结合附图对本申请实施例提供的故障检测方法进行详细介绍。

[0150] 图5示出了本申请实施例提供一种故障检测方法。如图5所示,该故障检测方法具体包括:

[0151] S501、电子设备获取待检测操作系统的当前操作系统运行数据。

[0152] 具体的,在进行灰度故障检测的过程中,电子设备可以基于两个触发条件触发故

障检测流程。

[0153] 第一个触发条件为电子设备主动触发故障检测流程。在这种情况下,电子设备可以直接获取待检测操作系统的当前操作系统运行数据,从而开始故障检测。

[0154] 可选的,电子设备主动触发故障检测流程可以是预先设定有定时任务。在当前时刻满足定时任务设定的时间时,电子设备可以直接获取待检测操作系统的当前操作系统运行数据,从而开始故障检测。

[0155] 可选的,上述定时任务可以是周期性的任务。在这种情况下,电子设备可以周期性的获取待检测操作系统的当前操作系统运行数据,从而开始故障检测,从而提高电子设备的待检测操作系统的故障检测效率。

[0156] 第二个触发条件为电子设备被动触发故障检测流程。在这种情况下,电子设备可以响应于接收到的故障检测指令,获取待检测操作系统的当前操作系统运行数据,从而开始故障检测。

[0157] 可选的,上述故障检测指令可以是APM层或者NPMD层发送的,也可以是响应于用户的故障检测操作生成的,本申请实施例对此不作限定。

[0158] 在一种可以实现的方式中,操作系统运行数据可以是待检测操作系统中的指标数据、日志数据、链路数据等数据。指标数据、日志数据、链路数据等数据又称为无状态数据,可以表示当前操作系统运行过程中产生的指标类型的数据。

[0159] 关于指标数据、日志数据、链路数据等数据的描述可以参考申请涉及的相关要素的简单介绍,在此不再赘述。

[0160] 当前操作系统运行数据是指当前时间段的系统运行数据。电子设备在当前时间段进行故障检测时,需要获取当前时间段的系统运行数据,从而保证准确的检测到在当前时间段的系统灰度故障。

[0161] 在一种可以实现的方式中,若电子设备想要检测历史时间段的系统灰度故障,则可以获取历史时间段的系统运行数据。

[0162] 在一种可以实现的方式中,结合图3,电子设备可以调用数据采集模块,获取待检测操作系统的当前操作系统运行数据。

[0163] S502、电子设备基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。

[0164] 其中,当前操作系统事件数据包括:待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据和当前操作系统的进程事件或线程事件关联的上下文信息。

[0165] 在一种可以实现的方式中,操作系统事件数据可以是待检测操作系统中的事件数据。事件数据又称为有状态数据,可以表示当前操作系统的进程事件或线程事件过程中产生的数据和当前操作系统的进程事件或线程事件关联的上下文信息。关于事件数据的描述可以参考申请涉及的相关要素的简单介绍,在此不再赘述。

[0166] 在一种可以实现的方式中,预设事件规则可以是根据人为经验或者领域知识预先建立好的。在获取到当前操作系统运行数据后,电子设备可以基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。

[0167] 也就是说,当前操作系统运行数据可以是单点的运行数据,当前操作系统事件数

据可以根据一个周期内的多个运行数据确定的事件数据。

[0168] 示例性的,电子设备每五秒检测一次待检测操作系统的CPU利用率,那在当前时间段内的一分钟可以检测12次CPU利用率,从而得到12个CPU利用率检测结果。在这种情况下,上述12个CPU利用率检测结果即为12个当前操作系统运行数据。

[0169] 预设事件规则包括:CPU利用率均高于预设数值时为故障事件。当这一分钟的前40秒的CPU利用率均高于预设数值时,电子设备可以获取前40秒中的8个CPU利用率检测结果,并基于这8个CPU利用率检测结果确定这一分钟的前40秒为CPU故障事件,即当前操作系统事件数据。

[0170] 在一种可以实现的方式中,当前操作系统事件数据还可以包括当前时间段内的漏洞修复、软件升级等系统活动类型的事件。

[0171] 又一示例性的,预设电子设备获取到的当前操作系统运行数据包括:待检测操作系统在当前时刻的CPU利用率为第一利用率、待检测操作系统在当前时刻的运行线程为线程A。预设事件规则包括:在同一时刻下,待检测操作系统的CPU利用率为A线程的CPU利用率。在这种情况下,电子设备可以基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据包括:线程A在当前时刻的运行过程中占用的CPU利用率为第一利用率。

[0172] 此外,电子设备还可以获取到关于线程A在当前时刻的运行过程中的上下文信息,例如线程B向线程A发送的运行指令等。

[0173] 在一种可以实现的方式中,结合图3,电子设备可以调用数据采集模块,基于获取到的当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。

[0174] 在又一种可以实现的方式中,结合图3,如图6所示,待检测操作系统中包括数据面和管理面。

[0175] 其中,数据面中的数据是指APM层的APP在ITIM层运行时,对ITIM层的各种系统资源使用的数据,可以直接反应APP的各种指标与使用情况。

[0176] 数据面可以包括多个线程节点(例如图6中的节点1、节点n、节点...),每个节点可以包括多个容器。每个容器可以通过系统调用层连接内核层。内核层的管理模块可以基于扩展的伯克利包过滤器(extended Berkeley Packet Filter,eBPF)技术,开发数据面探针模块(Probe Agent)。

[0177] 在故障检测的过程中,数据采集模块可以基于Probe Agent从数据面中获取指标数据、日志数据、链路数据等无状态数据以及事件数据中的有状态数据,并将获取到的数据存储到数据存储模块中。

[0178] 管理面中的数据是指ITIM层中的OS在运行时,OS的管理数据,例如OS版本升级事件数据、磁盘扩容事件数据。

[0179] 在故障检测的过程中,数据采集模块可以基于管理面的事件代理模块(event Agent)从管理面中事件数据中的活动类的事件数据,并将获取到的数据存储到数据存储模块中。

[0180] S503、电子设备基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果。

[0181] 在一种可以实现的方式中,电子设备在获取到当前操作系统运行数据和当前操作系统事件数据后,可以通过预先训练好的故障检测模型,确定待检测操作系统的系统故障检测结果,具体检测过程可以参考下文实施例的描述。

[0182] 在又一种可以实现的方式中,电子设备中可以预先存储有多种类型的系统数据(包括系统运行数据和操作系统事件数据)对应的系统故障检测结果。在获取到当前操作系统运行数据和当前操作系统事件数据后,电子设备可以读取系统数据与系统故障检测结果,从而确定待检测操作系统在当前时间段的系统故障检测结果。

[0183] 其中,多种类型的系统数据(包括系统运行数据和操作系统事件数据)对应的系统故障检测结果可以是电子设备根据历史时间段的系统数据和历史时间段的系统故障检测结果确定的,也可以是根据领域知识预先创建的,本申请实施例对此不作限定。

[0184] 在一种可以实现的方式中,引起系统故障检测结果中的系统故障包括以下单个类型的故障或组合类型的故障:网络IO故障、磁盘IO故障、调度故障、内存故障、进程或线程故障、文件系统故障、磁盘故障、CPU故障和容器故障。这样,电子设备可以检测到各种单个类型或者组合类型的故障的系统故障检测结果。

[0185] 上述当前操作系统运行数据为待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据。也就是说,这种数据一般仅仅表示一个进程事件或线程事件的指标(例如CPU利用率等),因此,当前操作系统运行数据为指标类型的数据。而上述当前操作系统事件数据不仅包括当前操作系统的进程事件或线程事件运行过程中产生的数据,还包括当前操作系统进程事件或线程事件关联的上下文信息。也就是说,这种数据不仅可以表示一个进程事件或线程事件的指标,还可以表示这个进程事件或线程事件的事件内容(例如该线程事件为主线程事件),和/或,事件状态(例如该线程事件为异常状态事件)等,因此,当前操作系统事件数据为事件类型的数据。

[0186] 在这种情况下,电子设备在检测操作系统层级的故障时,不仅可以获取到指标类型的当前操作系统运行数据,还可以基于当前操作系统运行数据确定事件类型的当前操作系统事件数据。后续,电子设备进行故障检测的过程中,相比通用技术中仅通过基础信息(即指标类型的当前操作系统运行数据)结合人工经验进行故障检测的方法,本申请实施例提供的故障检测方法可以基于当前操作系统运行数据和当前操作系统事件数据,快速确定待检测操作系统的系统故障检测结果,无需人工经验的介入,提高了故障检测的效率。

[0187] 其次,由于当前操作系统事件数据包括当前操作系统进程事件或线程事件关联的上下文信息,因此,电子设备在进行故障检测的过程中,相比通用技术仅通过基础信息结合人工经验进行故障检测,本申请实施例提供的故障检测方法可以挖掘当前操作系统事件数据中的上下文信息,从而基于上下文信息,获取进程事件或者线程事件的上下文关联事件的相关内容,进而根据这些内容结合当前操作系统运行数据,准确的确定系统故障检测结果,提高了故障检测的准确度。

[0188] 再次,由于当前操作系统运行数据和当前操作系统事件数据具备细粒度(可以精细到待检测系统的线程事件或进程事件)、宽范围检测数据(不仅包括指标类型的数据,还包括事件类型的数据)的特点,因此,本申请实施例提供的故障检测方法也可以检测到更细粒度的故障检测结果(例如具体线程事件或进程事件的故障检测结果),进一步提高了故障检测的准确度。

[0189] 此外,本申请实施例提供的故障检测方法无需在电子设备的操作系统层级添加各种故障检测工具,仅需操作系统自身具备的数据采集功能和数据处理功能,便可以快速检测故障,实现了基于操作系统轻量级的故障检测。

[0190] 在一种可以实现的方式中,在包括大量的云终端和云服务器的大规模云场景下,由于云服务器的服务部署的复杂性,可能导致云服务器的节点(即云服务器的进程或线程)多、链路长。在这种情况下,考虑到时间累积效应,可能造成进程事件或线程事件对应的数据传播效应滞后。如果按照通用的故障检测方法,可能导致故障检测效率低下。本申请实施例可以结合当前操作系统运行数据和当前操作系统事件数据,基于当前操作系统运行数据和当前操作系统事件数据具备细粒度(可以精细到待检测系统的线程事件或进程事件)、宽范围检测数据(不仅包括指标类型的数据,还包括事件类型的数据)的特点,挖掘当前操作系统事件数据中的上下文信息,从而可以快速、准确的确定待检测操作系统的系统故障检测结果。

[0191] 示例性的,在实际应用场景下,通过数据统计,基于通用的故障检测方法需要0.5天-7天的时间,才能确定待检测操作系统的系统故障检测结果。而本申请实施例结合当前操作系统运行数据和当前操作系统事件数据,仅需0.5小时-72小时,便可以快速、准确的确定待检测操作系统的系统故障检测结果。相比通用技术以“天”级的故障检测速度,本申请实施例提供的故障检测方法可以使得故障检测速度缩短到“分钟”级,提高了故障检测的效率。

[0192] 在一种可以实现的方式中,电子设备可以包括:系统层处理模块、应用层处理模块和网络层处理模块。在这种情况下,结合图5,如图7所示,上述S503中,电子设备基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果的方法具体包括:

[0193] S701、系统层处理模块基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果。

[0194] 为了提高灰度故障的检测效率,在确定待检测操作系统的系统故障检测结果后,电子设备可以结合APM层和/或NPMD层的故障检测结果,进一步的确定灰度故障。在这种情况下,本申请实施例提供的故障检测方法还包括:

[0195] S702、系统层处理模块向应用层处理模块和网络层处理模块发送系统故障检测结果。

[0196] 具体的,由于系统故障检测结果是待检测操作系统的系统层的故障检测结果,因此,为了结合应用层的应用故障检测结果,和/或,网络层的网络故障检测结果进一步判定电子设备的灰度故障,系统层处理模块可以向应用层处理模块和网络层处理模块发送系统故障检测结果。

[0197] S703、应用层处理模块基于系统故障检测结果,确定电子设备的应用故障检测结果。

[0198] 具体的,在接收到系统层处理模块发送的系统故障检测结果后,应用层处理模块可以基于系统故障检测结果,确定电子设备的应用故障检测结果。

[0199] 在一种可以实现的方式中,应用层处理模块可以基于系统层处理模块发送的系统故障检测结果,确定与该系统故障检测结果对应的应用是否故障,并在确定该应用故障时,

将该应用故障确定为应用故障检测结果。

[0200] 在又一种可以实现的方式中,应用层处理模块可以先基于应用层的故障检测工具,确定应用层的应用故障。然后,应用层处理模块结合检测到的应用故障和系统层处理模块发送的系统故障检测结果,进一步的确定电子设备的网络故障检测结果。

[0201] S704、网络层处理模块基于系统故障检测结果,确定电子设备的网络故障检测结果。

[0202] 具体的,在接收到系统层处理模块发送的系统故障检测结果后,网络层处理模块可以基于系统故障检测结果,确定电子设备的网络故障检测结果。

[0203] 在一种可以实现的方式中,网络层处理模块可以基于系统层处理模块发送的系统故障检测结果,确定与该系统故障检测结果对应的网络设备是否故障,并在确定该网络设备故障时,将该网络设备故障确定为网络故障检测结果。

[0204] 在又一种可以实现的方式中,网络层处理模块可以先基于网络层的故障检测工具,确定网络层的网络设备故障。然后,网络层处理模块结合检测到的网络设备故障和系统层处理模块发送的系统故障检测结果,进一步的确定电子设备的网络故障检测结果。

[0205] 通用技术中,电子设备在检测灰度故障时,应用层处理模块需要先检测APM层的应用故障检测结果。然后,在确定APM层的应用故障检测结果之后,网络层处理模块再确定NPMD层的网络故障检测结果。最后,系统层处理模块获取ITIM层的操作系统运行数据,结合人工经验确定最终的灰度故障检测结果。

[0206] 由上可知,通用技术的灰度故障检测方法中,APM层、NPMD层和ITIM层之间是紧耦合的,电子设备需要依次确定每个层级的故障检测结果,从而确定最终的灰度故障检测结果。而本申请中,ITIM层分别与APM层和NPMD层之间是松耦合的。也就是说,APM层、NPMD层和ITIM层之间可以独立检测当前层级的故障检测结果。相比通用技术,本申请实施例取消了层层下钻的步骤,提高了故障检测效率。

[0207] 其次,ITIM层的系统层处理模块可以提供独立的数据采集、灰度故障识别、根因分析能力,并在确定系统故障检测结果后,可以向应用层处理模块和网络层处理模块发送系统故障检测结果,以使得应用层处理模块基于系统故障检测结果,确定电子设备的网络故障检测结果,以及网络层处理模块基于系统故障检测结果,确定电子设备的网络故障检测结果,相比通用的故障检测方法需要APM层、NPMD层和ITIM层依次确定故障检测结果,本申请实施例提供的故障检测方法可以使得APM层、NPMD层和ITIM层同步确定相应的层级对应的故障检测结果,并结合各个层级的检测结果,确定灰度故障的检测结果,提高了故障检测的效率。

[0208] 此外,通用的故障检测方法中,APM层、NPMD层和ITIM层确定相应层级对应的故障检测结果后,需要通过人工经验进行灰度故障的判断。即各个层级对应的故障检测结果独立输出到人工检测对应的输出平台,各个层级对应的故障检测结果无结合处理,这就导致了各个层级对应的故障检测结果形成了孤岛数据。而本申请实施例可以以ITIM层的待检测操作系统为视角,自下而上主动识别、定位灰度故障,与以APM层为视角自上而下的通用的智能运维故障检测方法结合,可以消除数据孤岛,进一步提高了灰度故障检测的准确度。

[0209] 在一种可以实现的方式中,为了快速、准确的确定待检测操作系统的系统故障检测结果,电子设备可以基于训练好的故障检测模型,确定待检测操作系统的系统故障检测

结果。在这种情况下,本申请实施例提供的故障检测方法可以包括:电子设备训练得到故障检测模型的流程(简称为“故障检测模型训练”流程),以及电子设备根据故障检测模型确定待检测操作系统的系统故障检测结果的流程(简称为“故障检测”流程)。

[0210] 下面先对“故障检测模型训练”流程进行描述。

[0211] 如图8所示,本申请实施例提供的“故障检测模型训练”流程具体包括:

[0212] S801、电子设备获取历史操作系统数据。

[0213] 其中,历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据。

[0214] 历史操作系统运行数据是指历史时间段的系统运行数据。历史操作系统事件数据是指历史时间段的系统事件数据。

[0215] 历史操作系统事件数据包括:待检测操作系统中,历史操作系统的进程事件或线程事件运行过程中产生的数据和历史操作系统的进程事件或线程事件关联的上下文信息。

[0216] 电子设备可以先获取历史时间段的系统运行数据(即历史操作系统运行数据),然后基于历史操作系统运行数据和预设事件规则,确定待检测操作系统的历史操作系统事件数据。

[0217] 电子设备获取历史时间段的系统运行数据,以及基于历史操作系统运行数据和预设事件规则,确定待检测操作系统的历史操作系统事件数据的描述,可以参考S501中,电子设备获取待检测操作系统的当前操作系统运行数据,并基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据的相关描述,在此不再赘述。

[0218] 电子设备在训练得到故障检测模型时,可以基于历史时间段的系统运行数据和操作系统事件数据进行模型训练。

[0219] 关于历史时间段的系统运行数据的描述可以参考S501中,关于当前时间段的系统运行数据的相关描述,在此不再赘述。

[0220] S802、电子设备基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型。

[0221] 在一种可以实现的方式中,为了提高故障检测模型的模型精度,电子设备可以获取与历史操作系统运行数据对应的历史时间段的原始数据。然后,电子设备可以对这些原始数据执行预处理操作,以得到历史操作系统运行数据,为后续训练得到故障检测模型提供准确的基准数据。

[0222] 其中,预处理操作用于对原始数据进行数据清洗,以得到完整、平滑且无噪声的当前操作系统运行数据。

[0223] 可选的,上述预处理操作可以包括:缺失补齐、异常去噪、指标平滑等操作,本申请实施例对此不作限定。

[0224] 电子设备训练得到故障检测模型时,可以基于共轭梯度法和历史操作系统数据,训练得到故障检测模型,也可以基于梯度下降法和历史操作系统数据,训练得到故障检测模型,还可以基于其他通用的故障识别算法和历史操作系统数据,训练得到故障检测模型,本申请实施例对此不作限定。

[0225] 由上可知,电子设备可以基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型,以便于后续通过故障检测模型,快速、准确的确定系统故障检测结果,提高了故障检测效率。

[0226] 在一种可以实现的方式中,预设故障识别算法可以包括分类算法和参数优化算法。分类算法用于对故障检测模型的初始模型进行分类训练。参数优化算法可以对分类训练过程中的模型进行参数调整,从而得到故障检测模型。在这种情况下,结合图8,如图9所示,上述S802中,电子设备基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型的方法具体包括:

[0227] S901、电子设备基于预设状态规则,对历史操作系统事件数据添加状态标识。

[0228] 其中,状态标识包括:正常标识和异常标识。

[0229] 具体的,在确定待检测操作系统的历史操作系统事件数据后,为了训练得到精度较高的故障检测模型,电子设备可以基于预设状态规则,对历史时间段的历史操作系统事件数据添加状态标识。

[0230] 可选的,预设状态规则可以是根据领域知识创建的。

[0231] 示例性的,预设状态规则可以包括:线程A在运行过程中,CPU利用率大于预设阈值时,确定线程A的状态标识为异常标识;线程A在运行过程中,内存占用空间大于预设内存大小时,确定线程A的状态标识为异常标识。

[0232] 在一种可以实现的方式中,由于不同的系统事件对应的预设状态规则可能不同,因此,电子设备可以基于预设状态规则,对历史操作系统事件数据中的部分操作系统数据添加状态标识。

[0233] 示例性的,当预设状态规则包括:线程A在13点-14点的运行过程中,CPU利用率大于70%时,确定线程A的状态标识为异常标识,以及,线程A在18点-19点的运行过程中,CPU利用率大于80%时,确定线程A的状态标识为异常标识时,若线程A在15点的运行过程中,CPU利用率为75%,则无法准确的为线程A的操作系统事件数据添加状态标识。在这种情况下,电子设备可以仅对历史操作系统事件数据中,除线程A的操作系统事件数据以外的部分操作系统数据添加状态标识。

[0234] 在一种可以实现的方式中,电子设备还可以基于领域知识,对历史操作系统事件数据添加相关的解释信息。

[0235] 结合上述示例,预设历史操作系统事件数据为:线程A在13点-14点的运行过程中,CPU利用率大于70%,并且该历史操作系统事件数据的状态标识为异常标识。在这种情况下,电子设备还可以基于领域知识,对历史操作系统事件数据添加相关的解释信息:因为线程A在13点-14点的运行过程中,线程A中出现了多个冗余代码,从而导致CPU利用率大于70%,状态异常。

[0236] 这样,通过基于领域知识,对历史操作系统事件数据添加相关的解释信息,可以使得训练得到的故障检测模型输出的系统故障检测结果具备可解释性。

[0237] S902、电子设备基于分类算法和第二数据,对待训练模型进行训练,以得到待调整模型。

[0238] 具体的,故障检测模型可以是一种多分类模型。在这种情况下,电子设备基于分类算法和第二数据,对待训练模型进行训练,以得到待调整模型。

[0239] 其中,第二数据包括历史操作系统运行数据和添加状态标识后的历史操作系统事件数据;待训练模型包括:待训练的应用特征分类模型和待训练的故障分类模型。

[0240] 在一种可以实现的方式中,上述分类算法可以是K最近邻(k-Nearest Neighbor,

KNN) 分类算法,也可以是贝叶斯分类器算法,也可以是逻辑回归算法,还可以是其他用于分类的算法,本申请实施例对此不作限定。

[0241] 应用特征分类模型用于确定待检测操作系统的系统故障的应用特征。

[0242] 可选的,上述应用特征可以包括:CPU密集型、IO密集型、周期特征等。

[0243] CPU密集型也叫计算密集型,指的是操作系统的硬盘、内存性能相对CPU要好很多。此时,操作系统运作大部分的状况是CPU完全加载,CPU要读或者写硬盘或者内存时,硬盘或者内存存在很短的时间就可以完成。

[0244] IO密集型指的是操作系统的CPU性能相对硬盘、内存要好很多,此时,系统运作,大部分的状况是CPU在等I/O(硬盘/内存)的读/写操作,此时CPU加载率并不高。IO密集型的程序一般在达到性能极限时,CPU利用率仍然较低。这可能是因为任务本身需要大量I/O操作,而管道链路不顺畅,没有充分利用处理器能力。

[0245] 周期特征指的是操作系统的进程或线程或应用具有周期性的特征。

[0246] 故障分类模型用于确定待检测操作系统发生系统故障的应用特征的故障类型。

[0247] 可选的,上述故障类型可以包括:状态类型故障、功能类型故障、性能类型故障等。

[0248] 在一种可以实现的方式中,为了训练得到精度较高的故障检测模型,电子设备可以获取与历史操作系统运行数据对应历史时间段的原始数据。然后,电子设备可以对这些原始数据执行预处理操作,以得到历史操作系统运行数据,为后续训练得到故障检测模型提供准确的基准数据。

[0249] 其中,预处理操作用于对原始数据进行数据清洗,以得到完整、平滑且无噪声的当前操作系统运行数据。

[0250] 可选的,上述预处理操作可以包括:缺失补齐、异常去噪、指标平滑等操作,本申请实施例对此不作限定。

[0251] S903、电子设备基于历史操作系统运行数据和参数优化算法,对待调整模型进行参数调整,以得到故障检测模型。

[0252] 其中,故障检测模型包括:应用特征分类模型和故障分类模型。

[0253] 具体的,在对待调整模型进行模型训练的过程中,需要对待调整模型的参数进行调整优化。在这种情况下,电子设备可以基于历史操作系统运行数据和参数优化算法,对待调整模型进行参数调整,以得到故障检测模型。

[0254] 参数优化算法用于使得待训练模型的输出精度达到预设精度,通过将预设精度参数化,采用参数优化算法,不断的调整待训练模型的参数,使得待训练模型的输出结果不断接近参数化的目标值。

[0255] 在一种可以实现的方式中,参数优化算法可以包括:梯度下降算法、动量优化算法、自适应学习率优化算法等。

[0256] 在一种可以实现的方式中,上述待训练模型的输出精度可以包括:查全率、查准率、F1分数(F1 Score)等。

[0257] 相应的,参数化的目标值可以包括:查全率对应的预设数值、查准率对应的预设数值、F1分数对应的预设数值。

[0258] 由上可知,由于预设故障识别算法可以包括分类算法和参数优化算法,因此,分类算法可以用于对故障检测模型的初始模型进行分类训练,参数优化算法可以对分类训练过

程中的模型进行参数调整,从而得到故障检测模型,以便于后续通过故障检测模型,快速、准确的确定系统故障检测结果,提高了故障检测效率。

[0259] 在一种可以实现的方式中,上述“故障检测模型训练”流程可以是基于离线场景训练完成的,也可以是基于在线使用场景训练完成的。

[0260] 离线场景是指在电子设备可以在获取历史操作系统数据后,将历史操作系统数据迁移到实验室场景下,并在实验室场景下基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型。后续,在使用故障检测模型时,可以基于在线迁移算法,将离线场景下训练得到的故障检测模型迁移到在线使用场景下进行故障检测。

[0261] 在线使用场景是指在电子设备可以在获取历史操作系统数据后,直接基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型。后续,在使用故障检测模型时,可以直接使用故障检测模型进行故障检测。

[0262] 图10示出了本申请实施例提供的一种基于离线场景训练得到故障检测模型的流程示意图。结合图4,如图10所示,该“故障检测模型训练”流程可以包括:

[0263] S1001、数据预处理。

[0264] 具体的,电子设备的数据采集模块可以采集历史时间段的无状态数据(即本申请实施例中的历史操作系统运行数据)和历史时间段的有状态数据(即本申请实施例中的历史操作系统事件数据)。

[0265] 然后,电子设备的灰度故障识别模块可以对无状态数据做缺失补齐、异常去躁、指标平滑等操作,为后续步骤提供准确的基准数据。

[0266] S1002、数据标签化。

[0267] 具体的,由于有状态数据包含上下文信息,因此,电子设备的灰度故障识别模块可以为有状态数据添加应用正常/异常等状态的标签,标签的内容可以基于领域知识自定义。

[0268] 其次,电子设备还可以基于领域知识,对有状态数据添加相关的解释信息。

[0269] S1003、故障检测模型训练。

[0270] 具体的,电子设备可以基于无状态数据和数据标签化后的有状态数据进行模型训练,从而生成应用特征分类模型和故障分类模型。

[0271] S1004、故障识别算法评估。

[0272] 具体的,电子设备可以通过计算应用特征分类模型和故障分类模型的查准率、查全率、F1-Score指标,评估应用特征分类模型和故障分类模型是否达到训练收敛条件。

[0273] S1005、故障检测模型生成。

[0274] 具体的,在故障识别算法评估后,电子设备可以生成离线场景下的故障检测模型,供测试验证或线上迁移训练使用。

[0275] 下面再对“故障检测”流程进行描述。

[0276] 结合图5,如图11所示,上述S503中,电子设备基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果的方法具体包括:

[0277] S1101、电子设备基于预设状态规则,对当前操作系统事件数据添加状态标识。

[0278] 其中,状态标识包括:正常标识和异常标识。

[0279] 具体的,在确定待检测操作系统的当前操作系统事件数据后,为了使得故障检测模型可以准确的确定待检测操作系统的系统故障检测结果,电子设备可以基于预设状态规

则,对当前操作系统事件数据添加状态标识。

[0280] 电子设备基于预设状态规则,对当前操作系统事件数据添加状态标识的相关描述,可以参考S901中,电子设备基于预设状态规则,对历史操作系统事件数据添加状态标识的相关描述,在此不再赘述。

[0281] S1102、电子设备将第一数据输入到预先训练好的故障检测模型中,以得到系统故障检测结果。

[0282] 其中,第一数据包括当前操作系统运行数据和添加状态标识后的当前操作系统事件数据;故障检测模型为根据待检测操作系统的历史操作系统数据训练得到的;历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据。

[0283] 在一种可以实现的方式中,为了提高故障检测模型输出系统故障检测结果的准确度,电子设备可以获取与当前操作系统运行数据对应当前时间段的原始数据。然后,电子设备可以对这些原始数据执行预处理操作,以得到当前操作系统运行数据,为后续确定系统故障检测结果提供准确的基准数据。

[0284] 其中,预处理操作用于对原始数据进行数据清洗,以得到完整、平滑且无噪声的当前操作系统运行数据。

[0285] 可选的,上述预处理操作可以包括:缺失补齐、异常去躁、指标平滑等操作,本申请实施例对此不作限定。

[0286] 由上可知,由于故障检测模型为根据待检测操作系统的历史操作系统数据训练得到的,历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据,因此,电子设备确定待检测操作系统的系统故障检测结果时,可以通过故障检测模型,快速、准确的确定系统故障检测结果,提高了故障检测效率。

[0287] 在一种可以实现的方式中,由S902可知,待训练模型包括:待训练的应用特征分类模型和待训练的故障分类模型,因此,电子设备可以通过训练好的应用特征分类模型和故障分类模型确定系统故障检测结果。在这种情况下,电子设备将第一数据输入到预先训练好的故障检测模型中,以得到系统故障检测结果的方法具体包括:

[0288] 电子设备将第一数据输入到应用特征分类模型中,以得到应用特征故障检测结果。

[0289] 其中,应用特征故障检测结果可以包括:CPU密集型的应用特征故障检测结果、IO密集型的应用特征故障检测结果、周期特征的应用特征故障检测结果等。

[0290] 电子设备将第一数据中,与应用特征故障检测结果对应的系统数据输入到故障分类模型中,以得到系统故障检测结果。

[0291] 其中,系统故障检测结果可以包括:CPU密集型的应用出现状态类型的故障、CPU密集型的应用出现功能类型的故障、CPU密集型的应用出现性能类型的故障、IO密集型的应用出现状态类型的故障、IO密集型的应用出现功能类型的故障、IO密集型的应用出现性能类型的故障、周期特征的应用出现状态类型的故障、周期特征的应用出现功能类型的故障、周期特征的应用出现性能类型的故障。

[0292] 由上可知,由于故障检测模型包括应用特征分类模型和故障分类模型,因此,电子设备可以依次根据应用特征分类模型和故障分类模型确定不同的应用特征,以及每个应用特征对应的故障分类,提高了故障检测的准确度。

[0293] 图12示出了本申请实施例提供的一种基于预先训练得到故障检测模型进行故障检测的流程示意图。结合图4,如图12所示,该“故障检测模型训练”流程可以包括:

[0294] S1201、数据预处理。

[0295] 具体的,电子设备的数据采集模块可以采集当前时间段的无状态数据(即本申请实施例中的当前操作系统运行数据)和当前时间段的有状态数据(即本申请实施例中的当前操作系统事件数据)。

[0296] 然后,电子设备的灰度故障识别模块可以对无状态数据做缺失补齐、异常去躁、指标平滑等操作,为后续步骤提供准确的基准数据。

[0297] S1202、数据标签化。

[0298] 具体的,由于有状态数据包含上下文信息,因此,电子设备的灰度故障识别模块可以为有状态数据添加应用正常/异常等状态的标签,标签的内容可以基于领域知识自定义。

[0299] 其次,电子设备还可以基于领域知识,对有状态数据添加相关的解释信息。

[0300] S1203、故障检测模型训练或加载离线场景下的故障检测模型。

[0301] 在一种可以实现的方式中,电子设备可以基于无状态数据和数据标签化后的有状态数据进行模型训练,从而生成应用特征分类模型和故障分类模型。

[0302] 在又一种可以实现的方式中,电子设备可以基于在线迁移算法,从离线场景下加载预先训练好的故障检测模型。

[0303] S1204、灰度故障检测。

[0304] 具体的,电子设备通过应用特征分类模型,按照应用特征定义的应用阈值,基于偏离度、持续时间等因素进行故障检测。

[0305] S1205、灰度故障事件生成。

[0306] 具体的,在通过应用特征分类模型进行灰度故障检测后,可以进一步通过故障分类模型,确定最终的灰度故障事件(即本申请实施例中的系统故障检测结果),例如应用资源使用异常、应用性能异常、应用状态异常等。

[0307] 其中,灰度故障事件对应的检测结果可以包括故障事件的异常属性,例如发生的时间、位置、类型等。

[0308] 其中,故障事件的类型包括故障事件的资源使用、性能、状态等。

[0309] 在一种可以实现的方式中,在确定系统故障检测结果后,为了对待检测操作系统中发生故障的事件或应用进行根因定位,电子设备还可以构建应用依赖关系和事件传播关系,以此来进一步的实现故障根因定位。

[0310] 在这种情况下,结合图5,如图13所示,本申请实施例提供的故障检测方法还包括:

[0311] S1301、电子设备根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系。

[0312] 其中,应用依赖关系用于表示待检测操作系统中各进程或线程及各应用实例之间的相互依赖关系。

[0313] 在一种可以实现的方式中,预设依赖规则可以是当第一进程或线程与第二进程或线程之间存在执行先后顺序的关系时,确定第一进程或线程与第二进程或线程之间具有依赖关系;也可是当第一进程或线程需要基于第二进程或线程的运行结果才能实现运行时,确定第一进程或线程与第二进程或线程之间具有依赖关系;还可以是其他类型的规则,本

申请实施例对此不作限定。

[0314] 当前操作系统运行数据中包括各进程或线程及各应用实例对应的运行数据。在获取当前操作系统运行数据后,电子设备可以根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系。

[0315] 在一种可以实现的方式中,电子设备可以基于包括节点和边的应用依赖图的形式,构建应用依赖关系。在这种情况下,电子设备根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系的方法具体包括:

[0316] 电子设备根据当前操作系统运行数据和预设依赖规则,构建应用依赖图。

[0317] 其中,应用依赖图包括多个应用节点以及应用节点之间的边;多个应用节点中的主应用节点用于表示待检测操作系统中应用实例的主线程;多个应用节点中的从应用节点用于表示:待检测操作系统中应用实例的从线程和待检测操作系统中应用实例之间的依赖实例;多个应用节点中的第一应用节点与第二应用节点之间的边用于表示第一应用节点与第二应用节点之间存在依赖关系。

[0318] 在一种可以实现的方式中,主节点与主节点、主节点和从节点间存在单向或者双向直接依赖关系,从节点间存在间接依赖或没有依赖关系。

[0319] 应用实例的从线程可以包括代理应用实例(proxy)等。应用节点的属性可以包含时间、状态、资源使用等信息。

[0320] S1302、电子设备根据应用依赖关系,构建事件传播关系。

[0321] 具体的,由于应用事件的执行主体为待检测操作系统中的进程或线程及应用实例,因此,在确定应用依赖关系后,电子设备可以根据应用依赖关系,构建事件传播关系。

[0322] 其中,事件传播关系用于表示待检测操作系统中应用事件之间的传播关系。应用事件为待检测操作系统中的进程或线程及应用实例对应的应用事件。

[0323] 在一种可以实现的方式中,电子设备可以基于包括节点和边的事件传播图的形式,构建事件传播关系。在这种情况下,电子设备根据应用依赖关系,构建事件传播关系的方法具体包括:

[0324] 电子设备根据应用依赖图,构建事件传播图。

[0325] 其中,事件传播图包括与多个应用节点一一对应的多个事件节点以及事件节点之间的边;多个事件节点用于表示待检测操作系统中应用事件对应的应用实例;多个事件节点中的第一事件节点与第二事件节点之间的边用于表示第一事件节点对应的第一事件与第二事件节点对应的第二事件之间存在传播关系。

[0326] 具体的,事件节点属于应用节点。

[0327] 在一种可以实现的方式中,事件节点对应的事件包括异常事件、有状态事件以及管理面的事件(例如应用升级、系统打补丁等)。事件节点的属性可以包括时间、事件类型、异常等级等信息。

[0328] 在一种可以实现的方式中,有依赖关系的节点中发生的事件之间存在单向、双向、无传播关系。

[0329] S1303、电子设备根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程。

[0330] 具体的,在确定事件传播关系后,电子设备可以根据事件传播关系,确定引起系统

故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程,实现了从多维度、多类型数据进行关联根因分析的效果,解决了通用的故障检测方法中,故障根因分析困难、定位粒度不精细等技术问题。

[0331] 在一种可以实现的方式中,电子设备可以基于应用依赖图和事件传播图,快速、准确的确定根因事件,以及根因应用实例和/或根因进程或线程。在这种情况下,电子设备根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程的方法具体包括:

[0332] 电子设备确定与故障事件节点之间具有边的传播事件节点。

[0333] 其中,故障事件节点为故障事件对应的事件节点;故障事件为系统故障检测结果对应的故障事件。

[0334] 电子设备将传播事件节点中的传播起始事件节点对应的事件确定为根因事件,以及将传播起始事件节点对应的应用实例确定为根因应用实例,和/或,将传播起始事件节点对应的进程或线程确定为根因进程或线程。

[0335] 具体的,电子设备可以将传播事件节点中的传播起始事件节点对应的事件确定为根因事件,以及将传播起始事件节点对应的应用实例确定为根因应用实例,和/或,将传播起始事件节点对应的进程或线程确定为根因进程或线程。

[0336] 这样,本申请提供的故障检测方法可以在确定应用依赖图和事件传播图后,基于事件传播图的传播关系,分析定位根因事件,从而定位到进程/线程级应用及对应的根因事件,并基于事件传播图的传播关系逐事件节点分析定位异常事件的根因,输出应用依赖路径和事件传播路径,实现了从多维度、多类型数据进行关联根因分析的效果,解决了通用的故障检测方法中,故障根因分析困难、定位粒度不精细等技术问题。

[0337] 在一种可以实现的示例中,如图14所示,待检测操作系统中的主节点1中可以包括:应用11和应用12。主节点2中可以包括:应用21和应用22。主节点n中可以包括:应用n1和应用n2。应用11用于执行事件a。应用12用于执行事件b。应用21用于执行事件c。应用22用于执行事件d。应用n1用于执行事件m。应用n2用于执行事件n。电子设备可以获取待检测操作系统中,应用11、应用12、应用21、应用22、应用n1和应用n2的当前操作系统运行数据。接着,电子设备可以根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系。

[0338] 示例性的,当前操作系统运行数据中包括:应用11在当前时间段执行了事件a,应用12在当前时间段执行了事件b。预设依赖规则包括:在当前时间段同时执行事件的应用具有依赖关系。在这种情况下,电子设备可以确定应用11与应用12具有应用依赖关系,并基于应用11与应用12具有应用依赖关系,确定事件a和事件b具有事件传播关系。

[0339] 基于同样的方法,电子设备可以确定应用11与应用22具有应用依赖关系,应用22与应用n1具有应用依赖关系,应用n1与应用21具有应用依赖关系。

[0340] 相应的,电子设备可以确定事件d和事件m具有事件传播关系,事件m和事件c具有事件传播关系。

[0341] 图15示出了本申请实施例提供的又一种故障检测方法的流程示意图。如图15所示,该故障检测方法的步骤具体包括:

[0342] S1501、应用层处理模块检测APP的应用质量是否异常。

[0343] S1502、网络层处理模块检测网络设备是否有网络问题。

[0344] S1503、系统层处理模块获取待检测操作系统的当前操作系统运行数据,以及基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据。

[0345] 具体的,系统层处理模块采集的当前操作系统运行数据可以包括:指标数据、日志数据、链路数据等无状态数据。

[0346] 系统层处理模块确定的当前操作系统事件数据可以包括事件类型的有状态数据。

[0347] S1504、系统层处理模块基于获取到的当前操作系统运行数据和确定的当前操作系统事件数据,以及多维长周期时序数据的关联分析,完成灰度故障的识别。

[0348] S1505、系统层处理模块构建应用依赖关系和事件传播关系,并根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例和/或根因进程或线程。

[0349] 需要说明的是,本申请实施例对于S1501、S1502和S1503-S1505之间先后执行顺序不作限定。

[0350] 也就是说,本申请实施例可以将应用层处理模块检测APP的应用质量是否异常。网络层处理模块检测网络设备是否有网络问题、系统层处理模块检测待检测操作系统的系统故障检测结果之间是松耦合的关系。相比通用的故障检测方法,本申请实施例提供的故障检测方法取消了层层下钻的步骤,ITIM层提供独立的数据采集、灰度故障识别、根因分析能力,并可以将系统故障检测结果主动推送到APM层和NPMD层,以使得应用层处理模块和网络层处理模块根据系统故障检测结果准确的确定灰度故障。

[0351] 图16示出了本申请实施例提供的一种网络IO类异常事件类型的应用依赖图和事件传播图。

[0352] 如图16所示,待检测操作系统中的主节点1中可以包括:应用市场从节点和搜索微服务从节点。

[0353] 主节点2中可以包括:负一屏(nginx)从节点。

[0354] 主节点n中可以包括:负一屏服务从节点和第一远程字典服务集群(redis cluster server)从节点。

[0355] 主节点n+1中可以包括:第二远程字典服务集群从节点。

[0356] 应用市场从节点对应的APM层的事件包括:a.1、高频搜索结果空异常事件。

[0357] 搜索微服务从节点对应的APM层的事件包括:a.2、应用版本升级事件。

[0358] 主节点1对应的APM层的事件包括:a.3、CPU高负载事件。

[0359] 负一屏从节点对应的APM层的事件包括:a.4、微服务响应超时事件。

[0360] 负一屏从节点对应的ITIM层的事件包括:b.1、N-T2链路往返时间(round-trip time,RTT)异常事件。

[0361] 负一屏服务从节点对应的APM层的事件包括:a.5、新闻公告信息获取异常事件。

[0362] 负一屏服务从节点对应的ITIM层的事件包括:b.2、T2-R2链路RTT异常事件。

[0363] 第二远程字典服务集群从节点对应的ITIM层的事件包括:b.3、第二远程字典服务网络丢包异常事件。

[0364] 结合图16,图17示出了本申请实施例提供的一种网络IO类异常事件类型的故障检测方法流程示意图。

[0365] S1701、应用层处理模块检测APP异常事件。

[0366] 具体的,应用层处理模块检测到应用性能下降异常,对应节点2上的a.4异常事件(微服务响应超时),同时也检测到节点1的a.1(高频搜索结果空异常)、a.2(应用版本升级)、节点n的a.5异常事件(新闻公告信息获取异常)。

[0367] S1702、网络层处理模块未检测到网络异常事件。

[0368] S1703、系统层处理模块采集节点1、节点2、…、节点n+1的网络I/O、磁盘I/O、调度、内存、进程/线程、文件系统、磁盘、CPU及容器数据。

[0369] S1704、系统层处理模块识别节点2上的b.1异常事件,节点n上的b.2异常事件,节点n+1的b.3异常事件。

[0370] 具体的,系统层处理模块可以识别到节点2上的b.1异常事件(节点2的Nginx到节点n的负一屏服务RTT时延异常),节点n上的b.2异常事件(节点n的负一屏服务到节点n+1的第二远程字典服务集群RTT时延异常),节点n+1的b.3异常事件(第二远程字典服务集群存在网络丢包异常)。

[0371] S1705、系统层处理模块确定根因定位结果。

[0372] 具体的,应用依赖关系可以包括:节点1的应用市场依赖于节点1的搜索微服务;节点2的Nginx服务依赖于节点1的应用市场;节点2的Nginx依赖于节点n的负一屏服务;节点n的负一屏服务依赖于第一远程字典服务集群;节点n的负一屏服务依赖于节点n+1的第二远程字典服务集群。

[0373] 根据事件传播关系确定根因定位结果的具体过程为:由于节点n的负一屏服务依赖于节点n+1的第二远程字典服务集群,所以b.3可以传播到b.2,由于节点2的Nginx依赖于节点n的负一屏服务,所以b.2可以传播到b.1,其中b.3异常事件的根因是由于网络协议栈缓冲器(buffer)队列满导致,所以b.1由b.3导致,根因即为b.3异常事件的根因。

[0374] 图18示出了本申请实施例提供的一种混合类灰度故障的应用依赖图和事件传播图。

[0375] 混合类灰度故障指是在异常事件传播路径中存在网络I/O、磁盘I/O、调度、内存、进程/线程、文件系统、磁盘、CPU及容器等不同类的异常事件。

[0376] 如图18所示,待检测操作系统中的主节点1中可以包括:应用市场从节点和搜索微服务从节点。

[0377] 主节点2中可以包括:负一屏(nginx)从节点。

[0378] 主节点n中可以包括:负一屏服务从节点和第一远程字典服务集群(redis cluster server)从节点。

[0379] 主节点n+1中可以包括:第二远程字典服务集群从节点。

[0380] 应用市场从节点对应的APM层的事件包括:a.1、高频搜索结果空异常事件。

[0381] 搜索微服务从节点对应的APM层的事件包括:a.2、应用版本升级事件。

[0382] 主节点1对应的APM层的事件包括:a.3、CPU高负载事件。

[0383] 负一屏从节点对应的APM层的事件包括:a.4、微服务响应超时事件。

[0384] 负一屏从节点对应的ITIM层的事件包括:b.1、N-T2链路往返时间(round-trip time,RTT)异常事件。

[0385] 负一屏服务从节点对应的APM层的事件包括:a.5、新闻公告信息获取异常事件。

[0386] 负一屏服务从节点对应的ITIM层的事件包括:b.2、T2-R2链路RTT异常事件。

[0387] 第二远程字典服务集群从节点对应的ITIM层的事件包括:b.3、第二远程字典服务CPU被干扰(CPU异常)事件。

[0388] 结合图18,图19示出了本申请实施例提供的一种混合类灰度故障的故障检测方法流程示意图。

[0389] S1901、应用层处理模块检测APP异常事件。

[0390] 具体的,应用层处理模块检测到应用性能下降异常,对应节点2上的a.4异常事件(微服务响应超时),同时也检测到节点1的a.1(高频搜索结果空异常)、a.2(应用版本升级)、节点n的a.5异常事件(新闻公告信息获取异常)。

[0391] S1902、网络层处理模块未检测到网络异常事件。

[0392] S1903、系统层处理模块采集节点1、节点2、...、节点n+1的网络I/O、磁盘I/O、调度、内存、进程/线程、文件系统、磁盘、CPU及容器数据。

[0393] S1904、系统层处理模块识别节点2上的b.1异常事件,节点n上的b.2异常事件,节点n+1的b.3异常事件。

[0394] 具体的,系统层处理模块可以识别到节点2上的b.1异常事件(节点2的Nginx到节点n的负一屏服务RTT时延异常),节点n上的b.2异常事件(节点n的负一屏服务到节点n+1的第二远程字典服务集群RTT时延异常),节点n+1的b.3异常事件(第二远程字典服务集群存在CPU被干扰异常)。

[0395] S1905、系统层处理模块确定根因定位结果。

[0396] 具体的,应用依赖关系可以包括:节点1的应用市场依赖于节点1的搜索微服务;节点2的Ngnix服务依赖于节点1的应用市场;节点2的Ngnix依赖于节点n的负一屏服务;节点n的负一屏服务依赖于第一远程字典服务集群;节点n的负一屏服务依赖于节点n+1的第二远程字典服务集群;节点n+1的第二远程字典服务集群与Spark Worker双向依赖。

[0397] 根据事件传播关系确定根因定位结果的具体过程为:由于节点n的负一屏服务依赖于节点n+1的第二远程字典服务集群,所以b.3可以传播到b.2,由于节点2的Ngnix依赖于节点n的负一屏服务,所以b.2可以传播到b.1,其中b.3异常事件的根因是由于内存集群计算平台(Spark Worker)的批处理业务抢占CPU导致,所以b.1由b.3导致,根因即为b.3异常事件的根因。

[0398] 需要说明的是,本申请实施例对于S1901、S1902和S1903-S1905之间先后执行顺序不作限定。

[0399] 可以理解的,在实际实施时,本申请实施例所述的故障检测设备可以包含有用于实现前述对应故障检测方法的一个或多个硬件结构和/或软件模块,这些执行硬件结构和/或软件模块可以构成一个故障检测设备。

[0400] 本领域技术人员应该很容易意识到,结合本文中所公开的实施例描述的各示例的算法步骤,本申请能够以硬件或硬件和计算机软件的结合形式来实现。某个功能究竟以硬件还是计算机软件驱动硬件的方式来执行,取决于技术方案的特定应用和设计约束条件。专业技术人员可以对每个特定的应用来使用不同方法来实现所描述的功能,但是这种实现不应认为超出本申请的范围。

[0401] 基于这样的理解,本申请实施例还对应提供一种故障检测装置,可以应用于故障检测设备。图20示出了本申请实施例提供的故障检测装置的结构示意图。如图20所示,该故

障检测装置可以包括:获取单元2001和处理单元2002;

[0402] 获取单元2001,用于获取待检测操作系统的当前操作系统运行数据。例如,结合图5,获取单元2001用于执行S501。

[0403] 处理单元2002,用于基于当前操作系统运行数据和预设事件规则,确定待检测操作系统的当前操作系统事件数据;当前操作系统事件数据包括:待检测操作系统中,当前操作系统的进程事件或线程事件运行过程中产生的数据和当前操作系统的进程事件或线程事件关联的上下文信息。例如,结合图5,处理单元2002用于执行S502。

[0404] 处理单元2002,还用于基于当前操作系统运行数据和当前操作系统事件数据,确定待检测操作系统的系统故障检测结果。例如,结合图5,处理单元2002用于执行S503。

[0405] 在一种可以实现的方式中,电子设备包括:系统层处理模块、应用层处理模块和网络层处理模块;

[0406] 系统层处理模块用于确定待检测操作系统的系统故障检测结果。例如,结合图7,系统层处理模块用于执行S701。

[0407] 系统层处理模块还用于向应用层处理模块和网络层处理模块发送系统故障检测结果。例如,结合图7,系统层处理模块用于执行S702。

[0408] 应用层处理模块用于基于系统故障检测结果,确定电子设备的应用故障检测结果。例如,结合图7,应用层处理模块用于执行S703。

[0409] 网络层处理模块用于基于系统故障检测结果,确定电子设备的网络故障检测结果。例如,结合图7,网络层处理模块用于执行S704。

[0410] 在一种可以实现的方式中,处理单元2002,具体用于:

[0411] 基于预设状态规则,对当前操作系统事件数据添加状态标识;状态标识包括:正常标识和异常标识。例如,结合图11,处理单元2002用于执行S1101。

[0412] 将第一数据输入到预先训练好的故障检测模型中,以得到系统故障检测结果;第一数据包括当前操作系统运行数据和添加状态标识后的当前操作系统事件数据;故障检测模型为根据待检测操作系统的历史操作系统数据训练得到的;历史操作系统数据包括:历史操作系统运行数据和历史操作系统事件数据。例如,结合图11,处理单元2002用于执行S1102。

[0413] 在一种可以实现的方式中,获取单元2001,还用于获取历史操作系统数据。例如,结合图8,获取单元2001用于执行S801。

[0414] 处理单元2002,还用于基于预设故障识别算法和历史操作系统数据,训练得到故障检测模型。例如,结合图8,处理单元2002用于执行S802。

[0415] 在一种可以实现的方式中,预设故障识别算法包括:分类算法和参数优化算法;处理单元2002,具体用于:

[0416] 基于预设状态规则,对历史操作系统事件数据添加状态标识。例如,结合图9,处理单元2002用于执行S901。

[0417] 基于分类算法和第二数据,对待训练模型进行训练,以得到待调整模型;第二数据包括历史操作系统运行数据和添加状态标识后的历史操作系统事件数据;待训练模型包括:待训练的应用特征分类模型和待训练的故障分类模型。例如,结合图9,处理单元2002用于执行S902。

[0418] 基于历史操作系统运行数据和参数优化算法,对待调整模型进行参数调整,以得到故障检测模型;故障检测模型包括:应用特征分类模型和故障分类模型。例如,结合图9,处理单元2002用于执行S903。

[0419] 在一种可以实现的方式中,处理单元2002,具体用于:

[0420] 将第一数据输入到应用特征分类模型中,以得到应用特征故障检测结果;

[0421] 将第一数据中,与应用特征故障检测结果对应的系统数据输入到故障分类模型中,以得到系统故障检测结果。

[0422] 在一种可以实现的方式中,处理单元2002,还用于根据当前操作系统运行数据和预设依赖规则,构建应用依赖关系;应用依赖关系用于表示待检测操作系统中各进程或线程及各应用实例之间的相互依赖关系。例如,结合图13,处理单元2002用于执行S1301。

[0423] 处理单元2002,还用于根据应用依赖关系,构建事件传播关系;事件传播关系用于表示待检测操作系统中应用事件之间的传播关系;应用事件为待检测操作系统中的进程或线程及应用实例对应的应用事件。例如,结合图13,处理单元2002用于执行S1302。

[0424] 处理单元2002,还用于根据事件传播关系,确定引起系统故障检测结果中的系统故障的根因事件,以及根因应用实例,和/或根因进程或线程。例如,结合图13,处理单元2002用于执行S1303。

[0425] 在一种可以实现的方式中,处理单元2002,具体用于:

[0426] 根据当前操作系统运行数据和预设依赖规则,构建应用依赖图;应用依赖图包括多个应用节点以及应用节点之间的边;多个应用节点中的主应用节点用于表示待检测操作系统中应用实例的主线程;多个应用节点中的从应用节点用于表示:待检测操作系统中应用实例的从线程和待检测操作系统中应用实例之间的依赖实例;多个应用节点中的第一应用节点与第二应用节点之间的边用于表示第一应用节点与第二应用节点之间存在依赖关系。

[0427] 在一种可以实现的方式中,处理单元2002,具体用于:

[0428] 根据应用依赖图,构建事件传播图;事件传播图包括与多个应用节点一一对应的多个事件节点以及事件节点之间的边;多个事件节点用于表示待检测操作系统中应用事件对应的应用实例;多个事件节点中的第一事件节点与第二事件节点之间的边用于表示第一事件节点对应的第一事件与第二事件节点对应的第二事件之间存在传播关系。

[0429] 在一种可以实现的方式中,处理单元2002,具体用于:

[0430] 确定与故障事件节点之间具有边的传播事件节点;故障事件节点为故障事件对应的事件节点;故障事件为系统故障检测结果对应的故障事件;

[0431] 将传播事件节点中的传播起始事件节点对应的事件确定为根因事件,以及将传播起始事件节点对应的应用实例确定为根因应用实例,和/或,将传播起始事件节点对应的进程或线程确定为根因进程或线程。

[0432] 在一种可以实现的方式中,引起系统故障检测结果中的系统故障包括以下单个类型的故障或组合类型的故障:

[0433] 网络输入输出IO故障、磁盘输入输出IO故障、调度故障、内存故障、进程或线程故障、文件系统故障、磁盘故障、中央处理器CPU故障和容器故障。

[0434] 如上所述,本申请实施例可以根据上述方法示例对故障检测设备进行功能模块的

划分。其中,上述集成的模块既可以采用硬件的形式实现,也可以采用软件功能模块的形式实现。另外,还需要说明的是,本申请实施例中对模块的划分是示意性的,仅仅为一种逻辑功能划分,实际实现时可以有另外的划分方式。例如,可以对应各个功能划分各个功能模块,也可以将两个或两个以上的功能集成在一个处理模块中。

[0435] 关于上述实施例中的故障检测装置,其中各个模块执行操作的具体方式、以及具备的有益效果,均已经在前述方法实施例中进行了详细描述,此处不再赘述。

[0436] 本申请实施例还提供一种电子设备。该电子设备可以是终端,该终端可以是手机、电脑等用户终端。图21示出了本申请实施例提供的终端的结构示意图。

[0437] 该终端可以是上述故障检测装置,包括至少一个处理器61,通信总线62,存储器63以及至少一个通信接口64。

[0438] 处理器61可以是一个处理器(central processing units,CPU),微处理单元,ASIC,或一个或多个用于控制本申请方案程序执行的集成电路。作为一个示例,结合图20,故障检测设备中的处理单元2002实现的功能与图21中的处理器61实现的功能相同。

[0439] 通信总线62可包括一通路,在上述组件之间传送信息。

[0440] 通信接口64,使用任何收发器一类的装置,用于与其他设备或通信网络通信,如服务器、以太网,无线接入网(radio access network,RAN),无线局域网(wireless local area networks,WLAN)等。作为一个示例,结合图20,故障检测设备中的获取单元2001实现的功能与图21中的通信接口64实现的功能相同。

[0441] 存储器63可以是只读存储器(read-only memory,ROM)或可存储静态信息和指令的其他类型的静态存储设备,随机存取存储器(random access memory,RAM)或者可存储信息和指令的其他类型的动态存储设备,也可以是电可擦可编程只读存储器(electrically erasable programmable read-only memory,EEPROM)、只读光盘(compact disc read-only memory,CD-ROM)或其他光盘存储、光碟存储(包括压缩光碟、激光碟、光碟、数字通用光碟、蓝光光碟等)、磁盘存储介质或者其他磁存储设备、或者能够用于携带或存储具有指令或数据结构形式的期望的程序代码并能够由计算机存取的任何其他介质,但不限于此。存储器可以是独立存在,通过总线与处理单元相连接。存储器也可以和处理单元集成在一起。

[0442] 其中,存储器63用于存储执行本申请方案的应用程序代码,并由处理器61来控制执行。处理器61用于执行存储器63中存储的应用程序代码,从而实现本申请方法中的功能。

[0443] 在具体实现中,作为一种实施例,处理器61可以包括一个或多个CPU,例如图21中的CPU0和CPU1。

[0444] 在具体实现中,作为一种实施例,终端可以包括多个处理器,例如图21中的处理器61和处理器65。这些处理器中的每一个可以是一个单核(single-CPU)处理器,也可以是一个多核(multi-CPU)处理器。这里的处理器可以指一个或多个设备、电路、和/或用于处理数据(例如计算机程序指令)的处理核。

[0445] 在具体实现中,作为一种实施例,终端还可以包括输入设备66和输出设备67。输入设备66和输出设备67通信,可以以多种方式接受用户的输入。例如,输入设备66可以是鼠标、键盘、触摸屏设备或传感设备等。输出设备67和处理器61通信,可以以多种方式来显示信息。例如,输出设备61可以是液晶显示器(liquid crystal display,LCD),发光二极管

(light emitting diode,LED)显示设备等。

[0446] 本领域技术人员可以理解,图21中示出的结构并不构成对终端的限定,可以包括比图示更多或更少的组件,或者组合某些组件,或者采用不同的组件布置。

[0447] 本申请实施例还提供一种电子设备,该电子设备例如可以是服务器。图22示出了本申请实施例提供的服务器的结构示意图。该服务器可以是故障检测装置。该服务器可因配置或性能不同而产生比较大的差异,可以包括一个或一个以上处理器71和一个或一个以上的存储器72。其中,存储器72中存储有至少一条指令,至少一条指令由处理器71加载并执行以实现上述各个方法实施例提供的故障检测方法。当然,该服务器还可以具有有线或无线网络接口、键盘以及输入输出接口等部件,以便进行输入输出,该服务器还可以包括其他用于实现设备功能的部件,在此不做赘述。

[0448] 本申请还提供了一种包括指令的计算机可读存储介质,所述计算机可读存储介质上存储有指令,当所述计算机可读存储介质中的指令由计算机设备的处理器执行时,使得计算机能够执行上述所示实施例提供的故障检测方法。例如,计算机可读存储介质可以为包括指令的存储器63,上述指令可由终端的处理器61执行以完成上述方法。又例如,计算机可读存储介质可以为包括指令的存储器72,上述指令可由服务器的处理器71执行以完成上述方法。

[0449] 可选地,计算机可读存储介质可以是非临时性计算机可读存储介质,例如,所述非临时性计算机可读存储介质可以是ROM、RAM、CD-ROM、磁带、软盘和光数据存储设备等。

[0450] 本申请还提供了一种计算机程序产品,该计算机程序产品包括计算机指令,当所述计算机指令在故障检测设备上运行时,使得所述故障检测设备执行上述图5-图19任一附图所示的故障检测方法。

[0451] 本领域技术人员在考虑说明书及实践这里公开的发明后,将容易想到本申请的其它实施方案。本申请旨在涵盖本申请的任何变型、用途或者适应性变化,这些变型、用途或者适应性变化遵循本申请的一般性原理并包括本申请未公开的本技术领域中的公知常识或惯用技术手段。说明书和实施例仅被视为示例性的,本申请的真正范围和精神由下面的权利要求指出。

[0452] 通过以上的实施方式的描述,所属领域的技术人员可以清楚地了解到,为描述的方便和简洁,仅以上述各功能模块的划分进行举例说明,实际应用中,可以根据需要而将上述功能分配由不同的功能模块完成,即将装置的内部结构划分成不同的功能模块,以完成以上描述的全部或者部分功能。上述描述的系统,装置和单元的具体工作过程,可以参考前述方法实施例中的对应过程,在此不再赘述。

[0453] 在本申请实施例各个实施例中的各功能单元可以集成在一个处理单元中,也可以是各个单元单独物理存在,也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现,也可以采用软件功能单元的形式实现。

[0454] 所述集成的单元如果以软件功能单元的形式实现并作为独立的产品销售或使用时,可以存储在一个计算机可读存储介质中。基于这样的理解,本申请实施例的技术方案本质上或者说对通用技术做出贡献的部分或者该技术方案的全部或部分可以以软件产品的形式体现出来,该计算机软件产品存储在一个存储介质中,包括若干指令用以使得一台计算机设备(可以是个人计算机,服务器,或者网络设备等)或处理器执行本申请各个实施

例所述方法的全部或部分步骤。而前述的存储介质包括：快闪存储器、移动硬盘、只读存储器、随机存取存储器、磁碟或者光盘等各种可以存储程序代码的介质。

[0455] 以上所述,仅为本申请实施例的具体实施方式,但本申请实施例的保护范围并不局限于此,任何在本申请实施例揭露的技术范围内的变化或替换,都应涵盖在本申请实施例的保护范围之内。因此,本申请实施例的保护范围应以所述权利要求的保护范围为准。

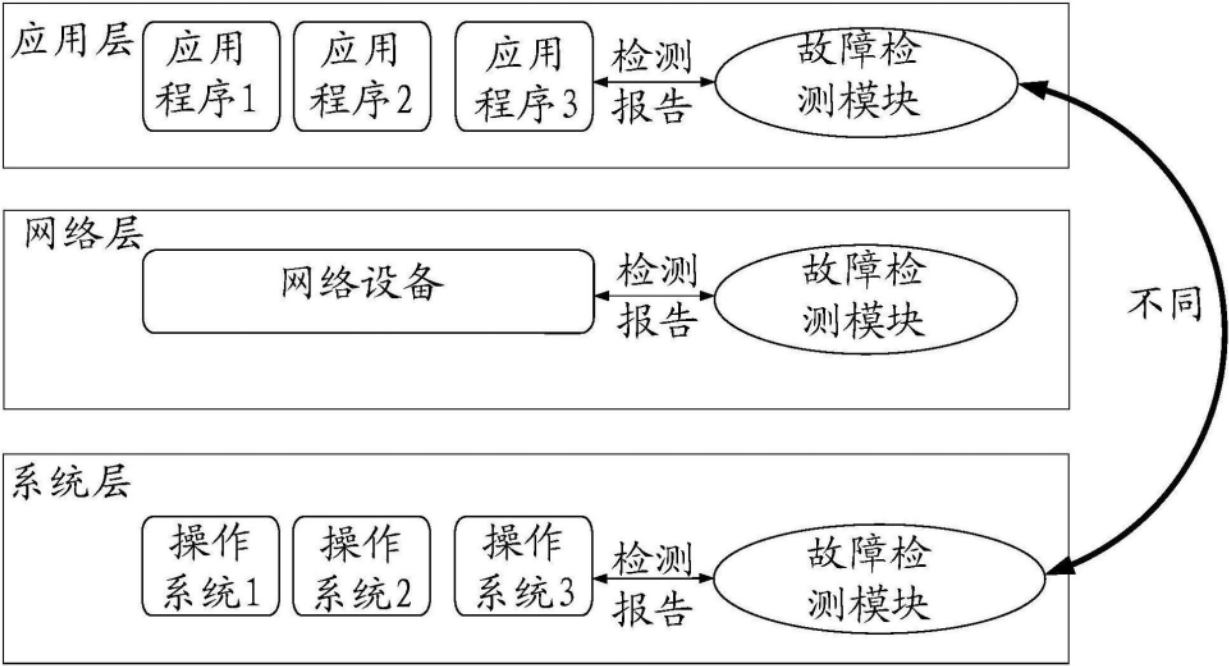


图1

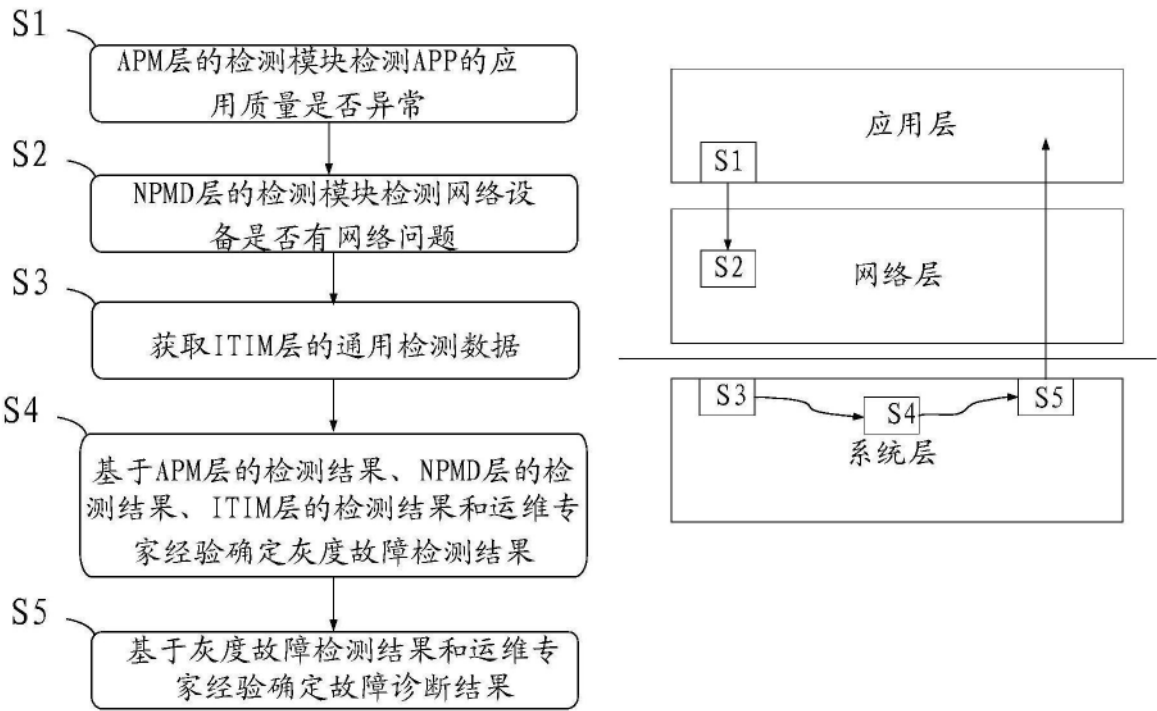


图2

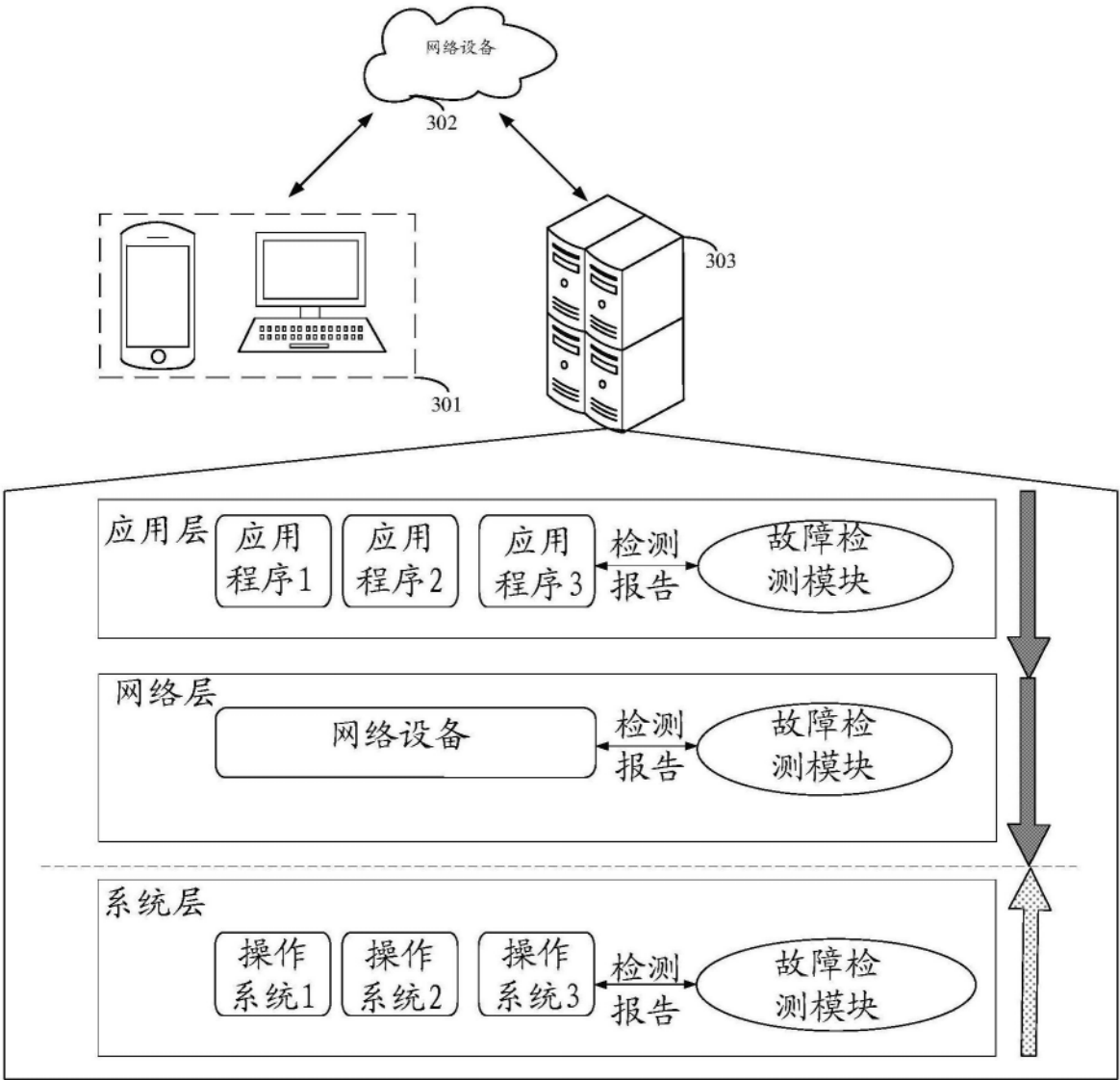


图3

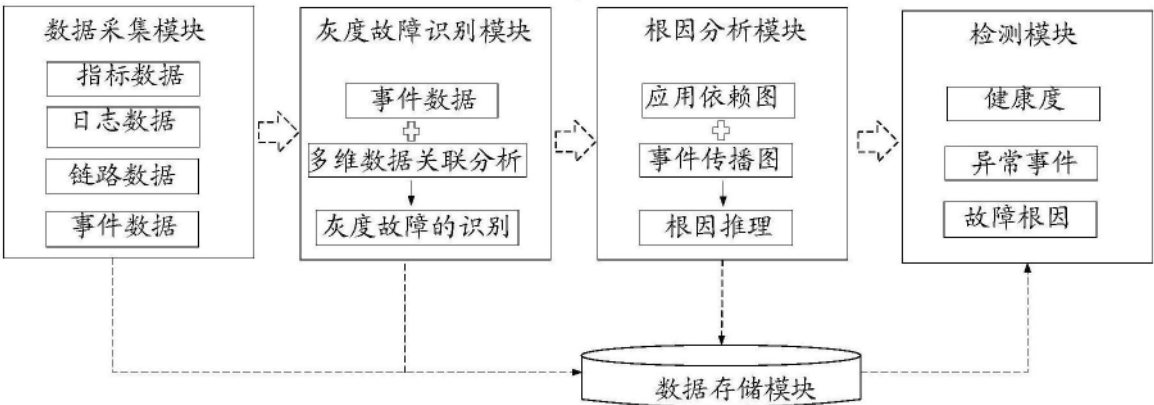


图4

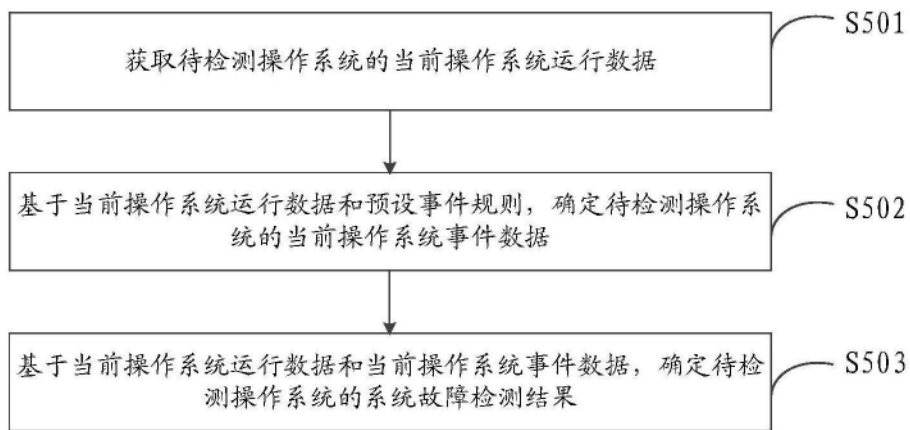


图5

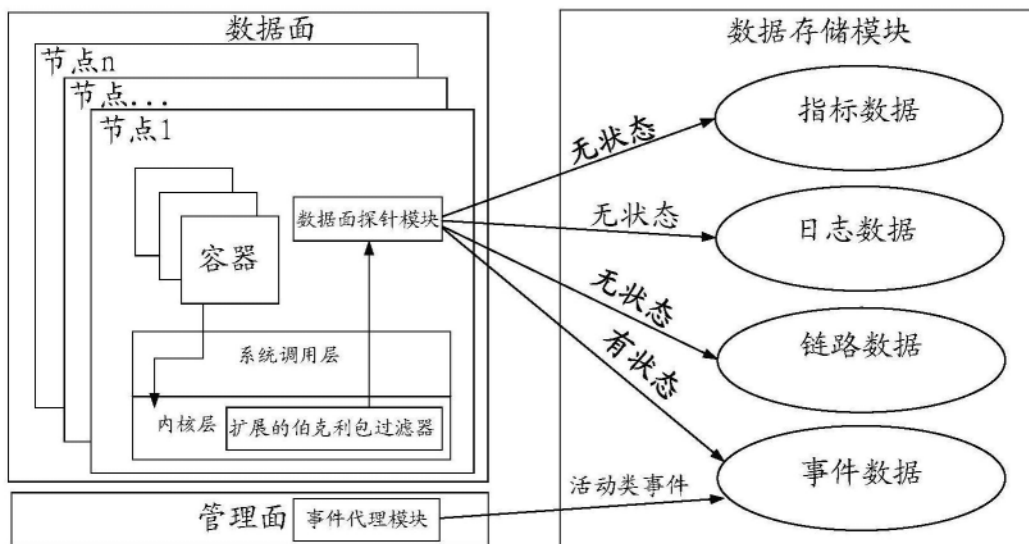


图6

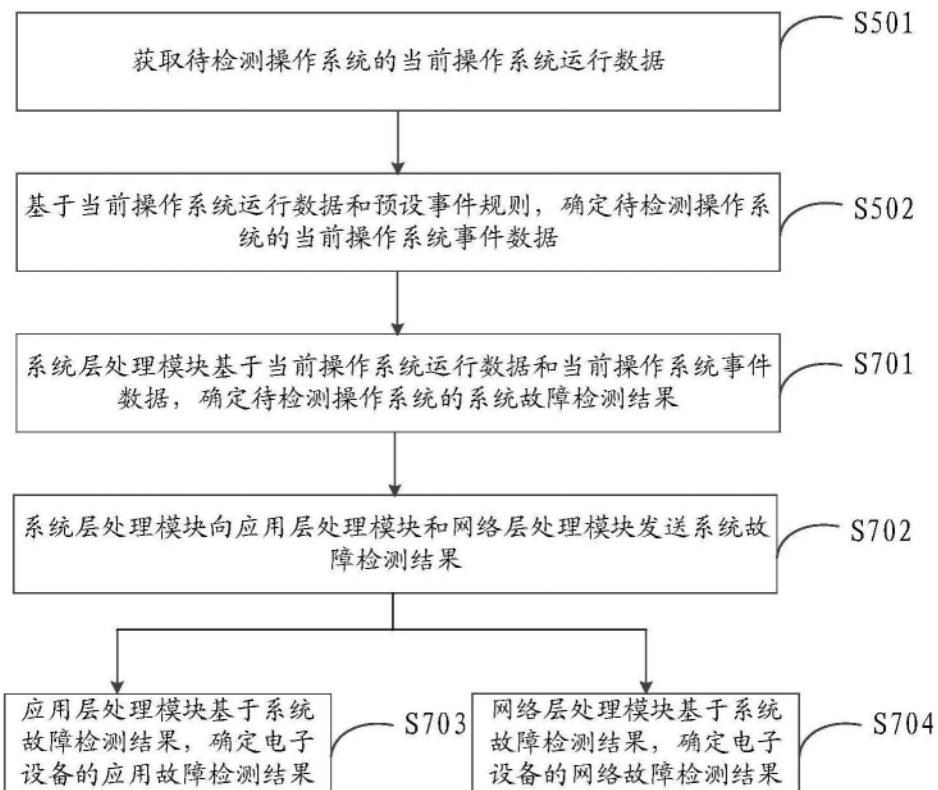


图7

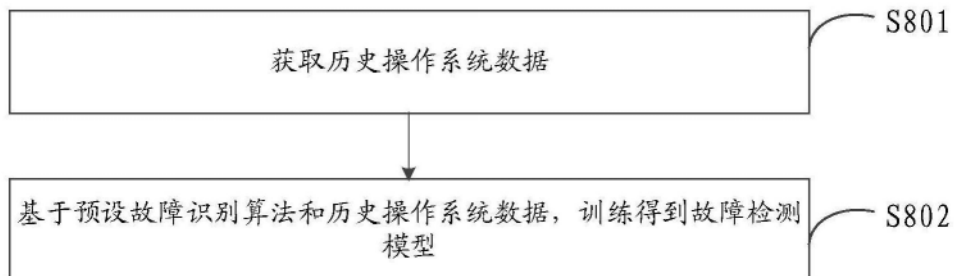


图8

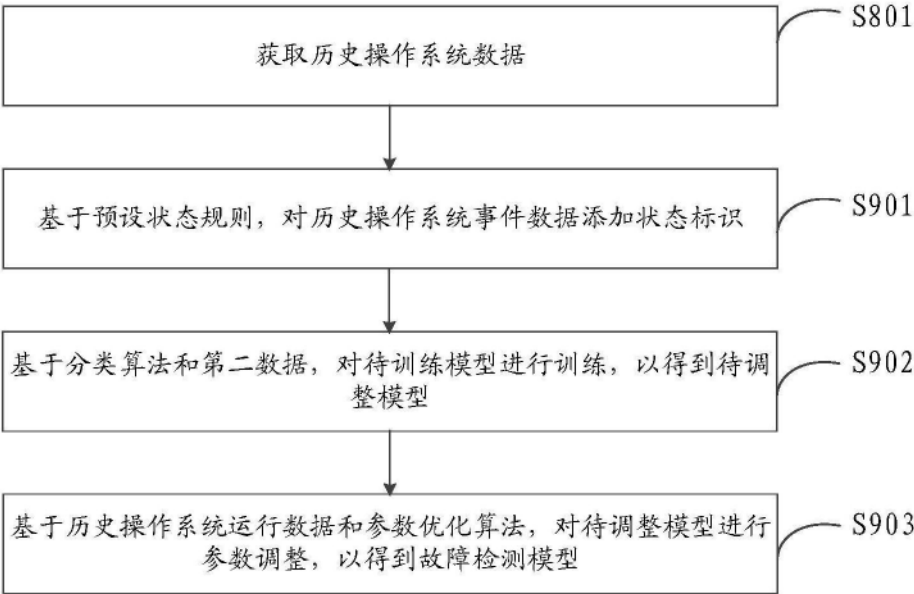


图9

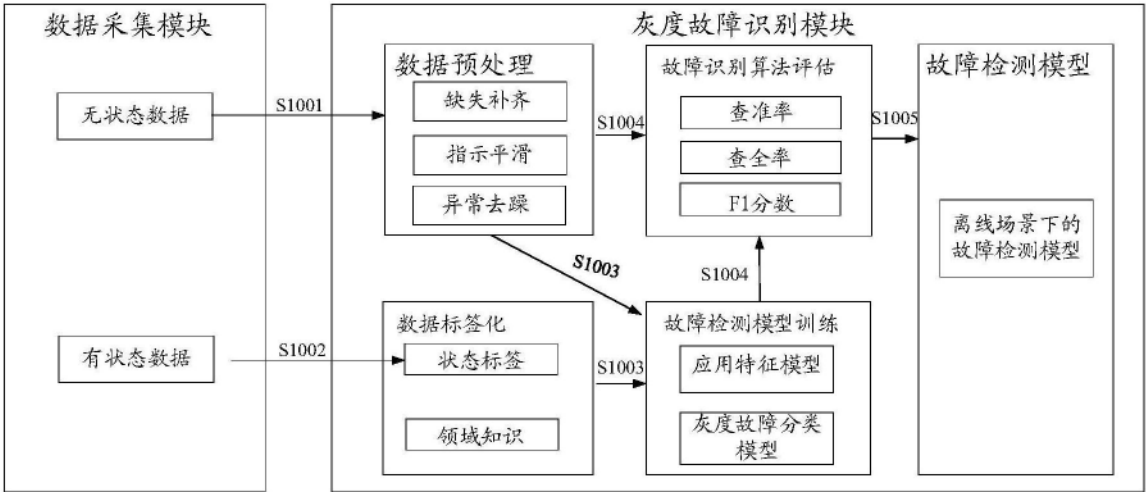


图10

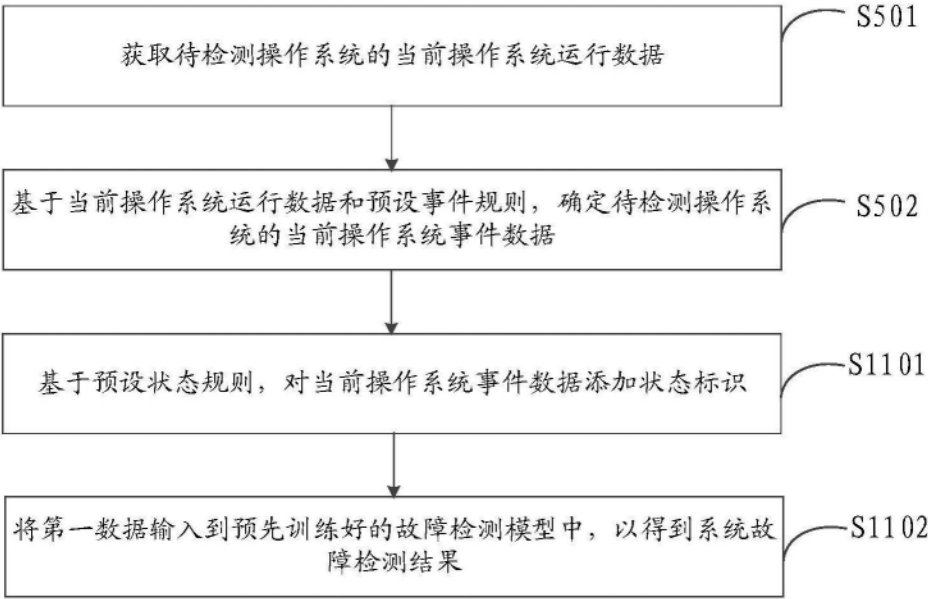


图11

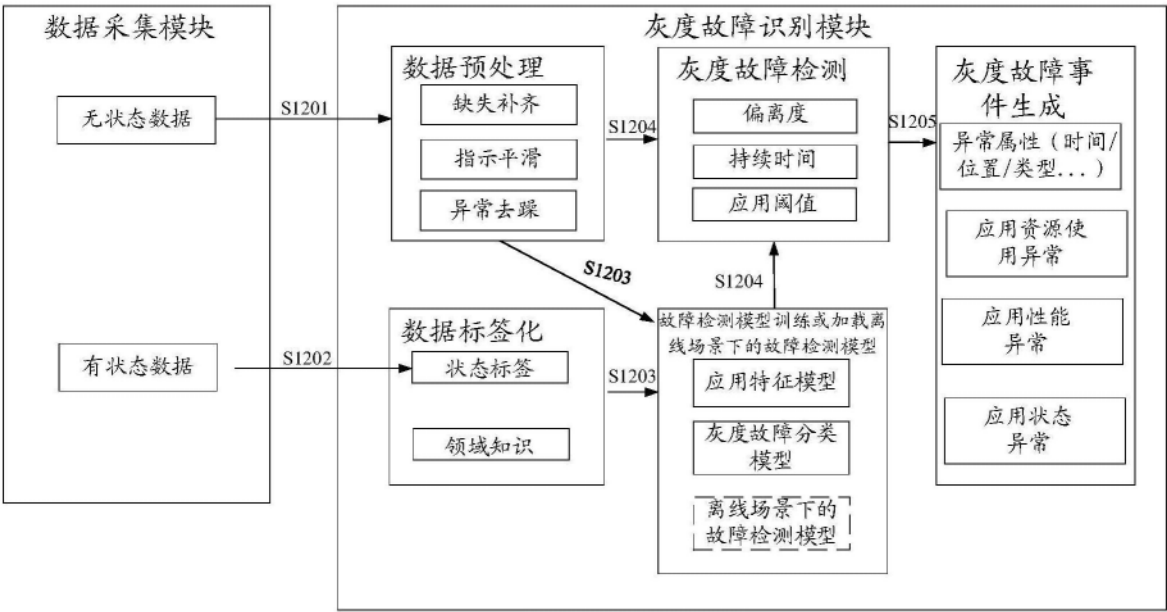


图12

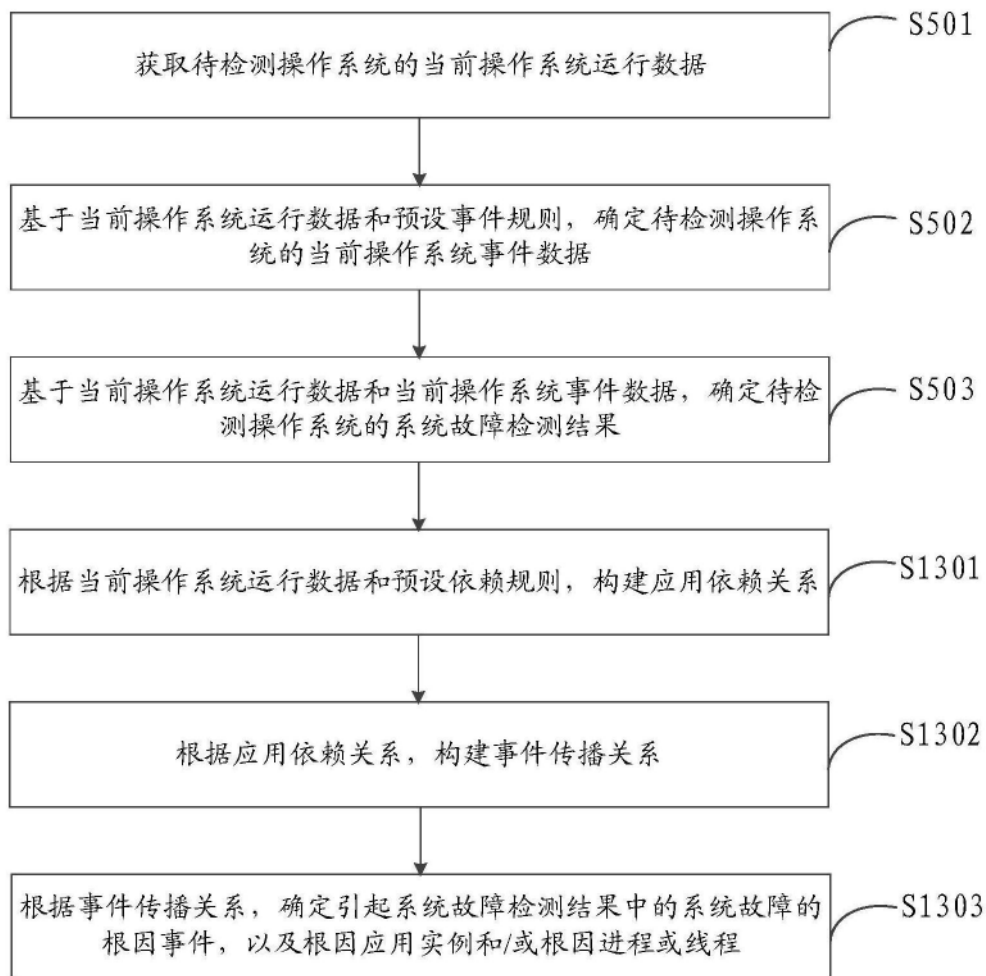


图13

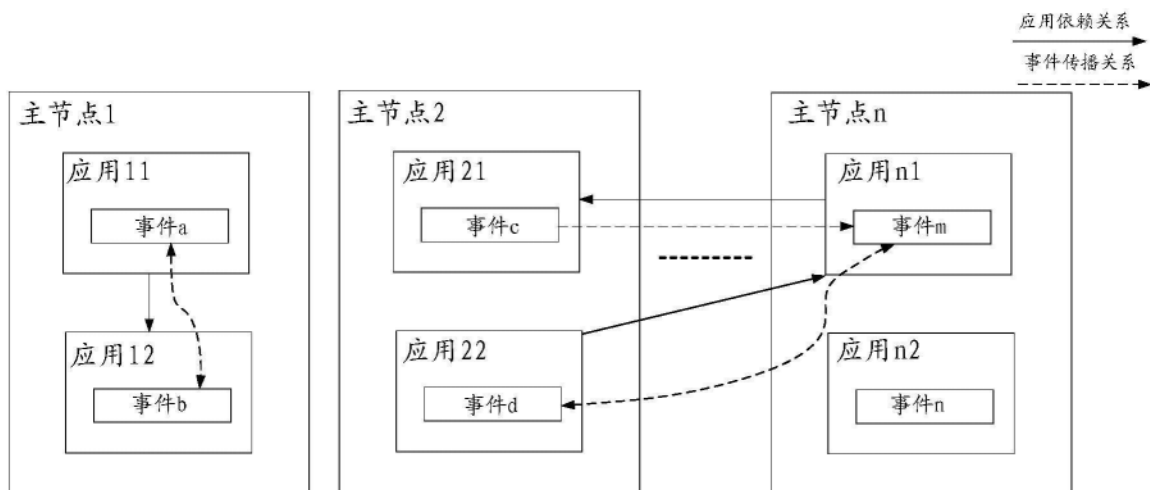


图14

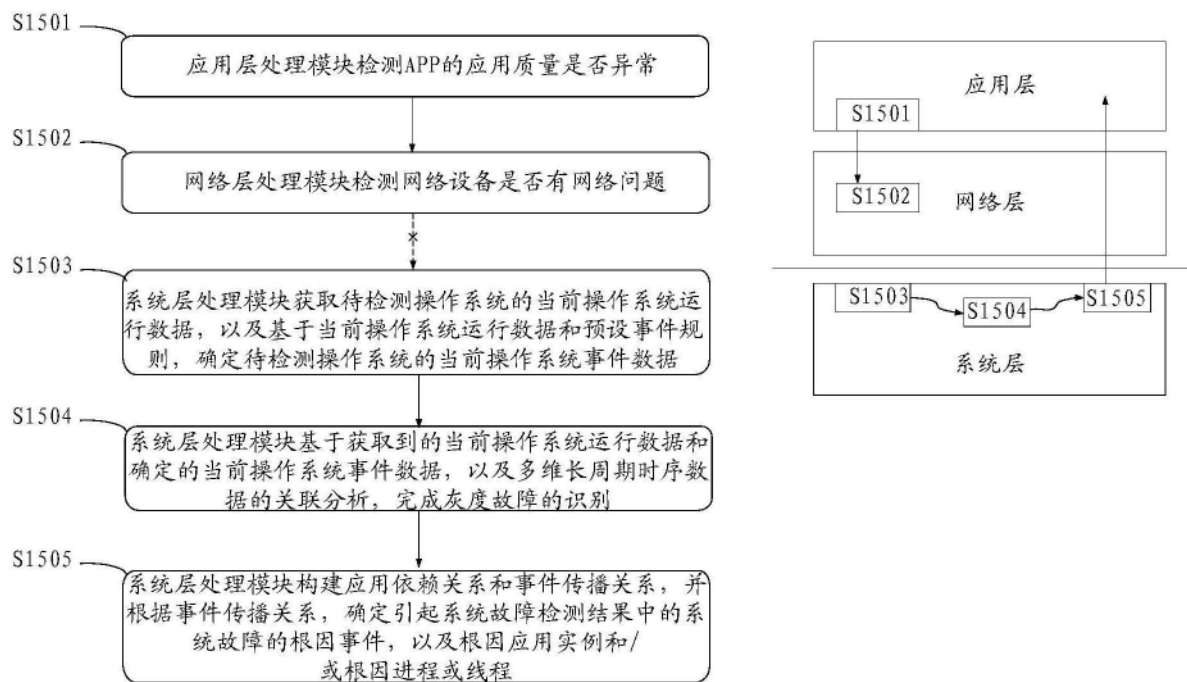


图15

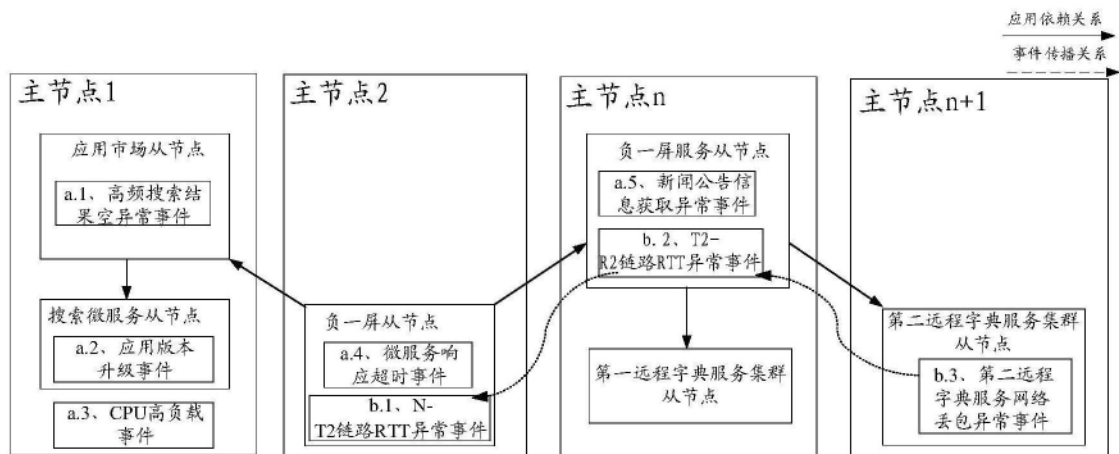


图16

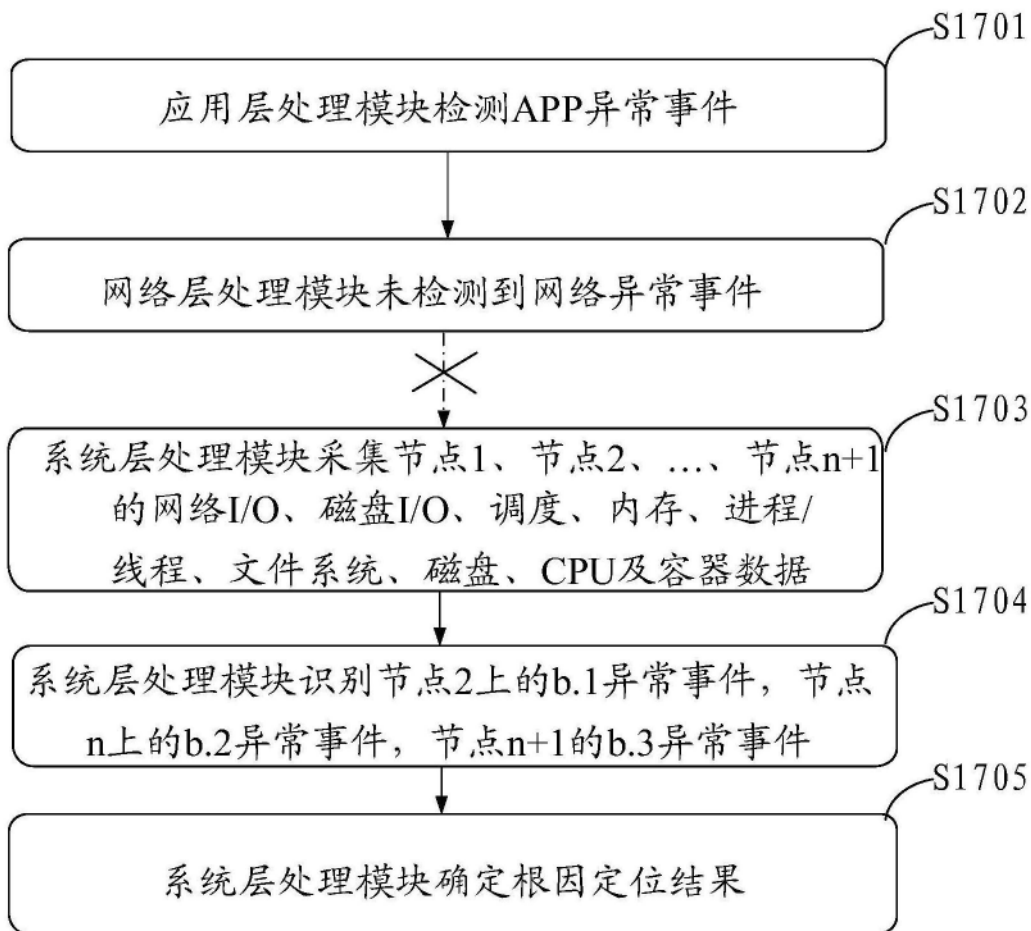


图17

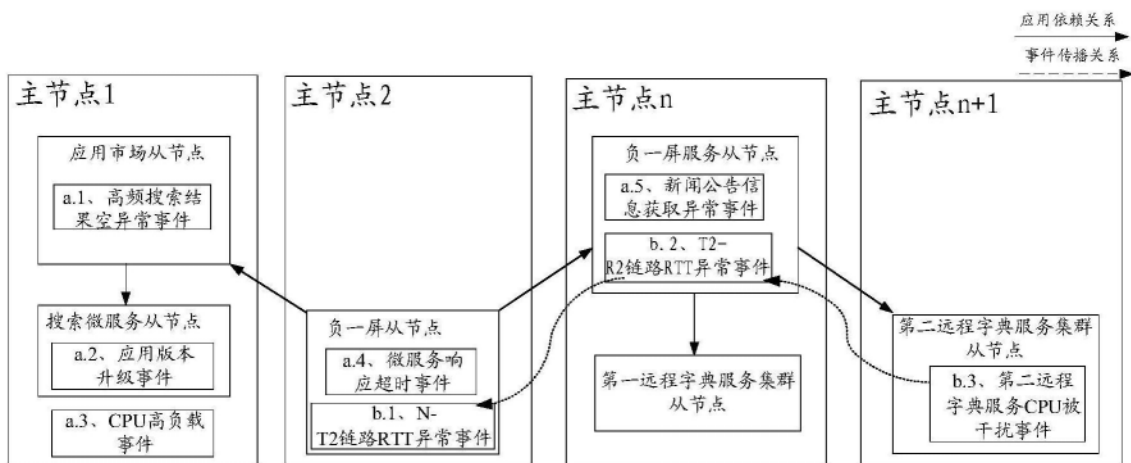


图18

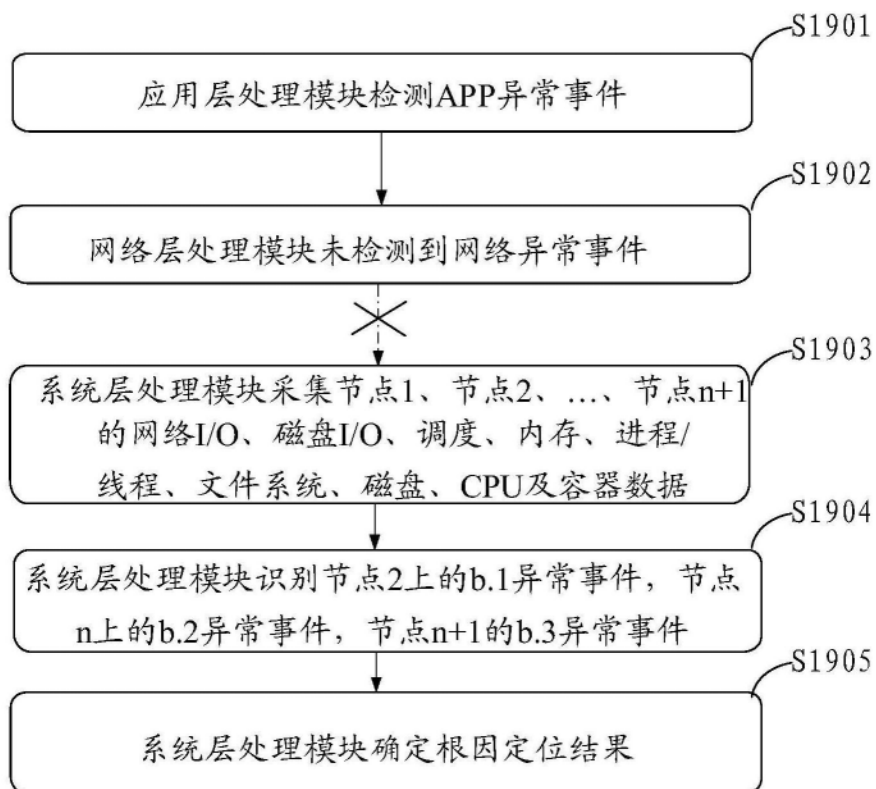


图19

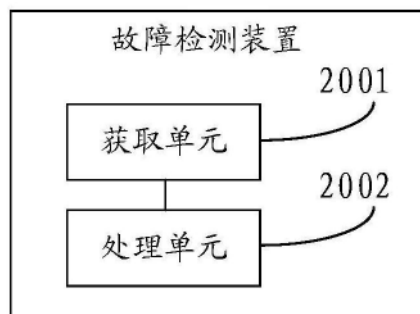


图20

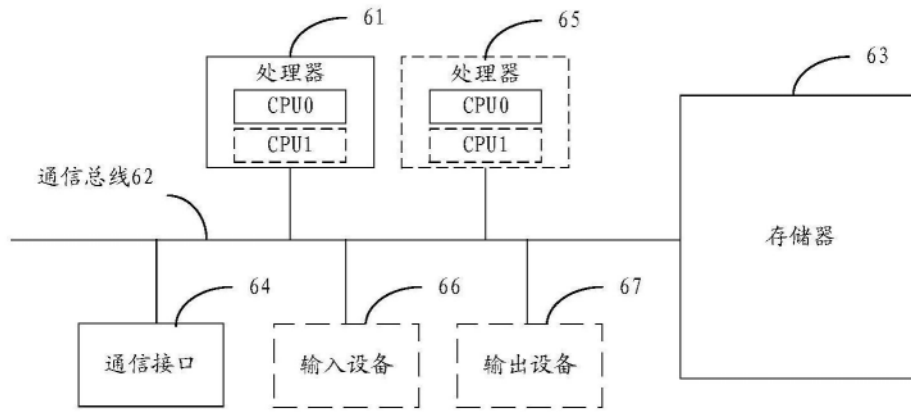


图21



图22