(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国 际 局

(43) 国际公布日
2019 年 4 月 18 日 (18.04.2019)

WIPO | PCT

(10) 国际公布号
WO 2019/072200 A1

(51) 国际专利分类号:
G06F 9/50 (2006.01) *G06F 9/48* (2006.01)

(21) 国际申请号: PCT/CN2018/109753

(22) 国际申请日: 2018 年 10 月 11 日 (11.10.2018)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
201710953233.7 2017年10月13日 (13.10.2017) CN

(71) 申请人: 华 为 技 术 有 限 公 司 (**HUAWEI TECHNOLOGIES CO., LTD.**) [CN/CN]; 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。

(72) 发明人: **陈秋林 (CHEN, Qiulin)**; 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。 **陈寒冰 (CHEN, Hanbing)**; 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。 **亢治 (KANG, Zhi)**; 中国广东省深圳市龙岗区坂田华为总部办公楼, Guangdong 518129 (CN)。

(81) 指定国 (除另有指明，要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL,

WO 2019/072200 A1

(54) **Title:** METHOD FOR MANAGING RESOURCE, AND TERMINAL DEVICE

(54) 发明名称: 资源管理的方法及终端设备

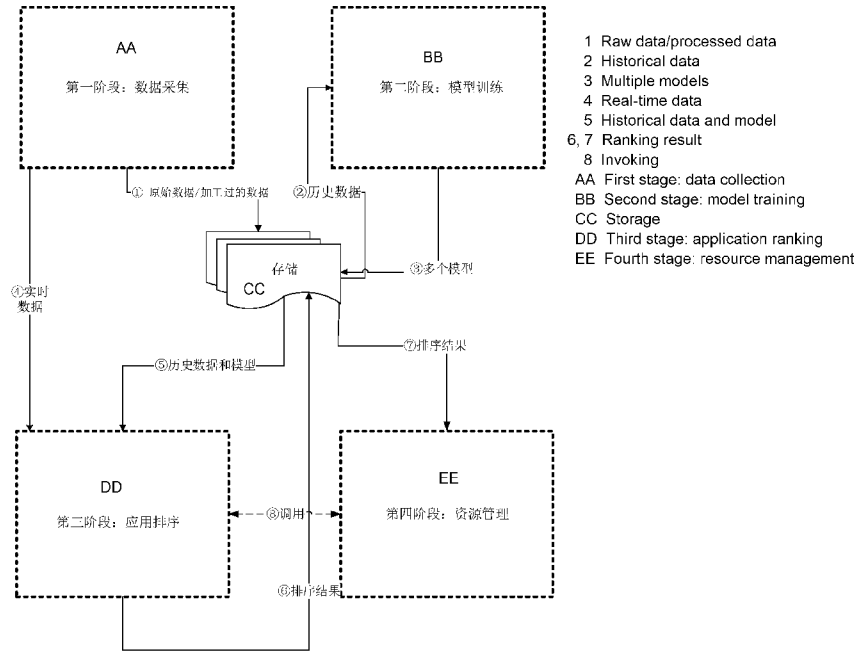

图 4

(57) **Abstract:** Provided are a method for managing a resource in a computer system, and a terminal device. The method comprises: acquiring data, the data comprising application timing characteristic data relevant to a current foreground application, and the data further comprising at least one piece of the following real-time data: the system time of a computer system, current state data of the computer system and current position data of the computer system; according to the at least one piece of the real-time data, selecting a target machine learning model, which matches the real-time data, from multiple machine learning models; inputting the acquired data

[见续页]

SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW。

(84) 指定国(除另有指明，要求每一种可提供的地区保护)：ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:

— 包括国际检索报告(条约第21条(3))。

into the target machine learning model so as to rank multiple applications installed on the computer system in terms of importance; and executing resource management according to a result of the importance ranking. The method improves the accuracy of recognizing the importance of an application, so that an object for resource allocation/pre-reservation or resource recycle is more accurate, thus improving the effectiveness of resource management.

(57) 摘要：本申请提供一种在计算机系统中管理资源的方法以及终端设备。该方法包括：获取数据，所述数据包括与当前的前台应用相关的应用时序特征数据，所述数据还包括以下实时数据中的至少一种：所述计算机系统的系统时间、所述计算机系统的当前状态数据和所述计算机系统的当前位置数据；根据所述实时数据中的至少一种从多个机器学习模型中选择与所述实时数据匹配的目标机器学习模型；将获取的所述数据输入所述目标机器学习模型以对所述计算机系统上安装的多个应用进行重要性排序；根据所述重要性排序的结果执行资源管理。该方法提高应用的重要性识别准确度，从而使得资源分配/预留或资源回收的对象更准确，进而提升资源管理的有效性。

资源管理的方法及终端设备

技术领域

本申请涉及计算机操作系统领域，尤其涉及一种对操作系统中部署的应用进行重要性排序以及根据重要性排序进行资源管理的方法、装置及系统等。

背景技术

计算机操作系统，包括智能手机操作系统，首要任务就是为运行在其上的各种应用维护资源分配。计算机操作系统视角的资源，包含多种类型，如以时间片为表示形式的处理器（例如中央处理器 CPU）计算资源，以内存页为表现形式的内存资源，以带宽为表现形式的输入输出 (Input/Output，I/O)资源等。当支撑用户当前操作的资源无法及时供给，操作系统就会表现出卡顿。因此，影响操作系统卡顿与否的核心因素是资源调度策略，而资源调度策略是否合理最重要的是识别重要应用和非重要应用，已尽可能达到资源的合理分配。

以智能手机操作系统 Android 为例，应用可划分为前台应用、后台应用和未启动的应用，用户能够感受到卡顿的场景通常是在对前台应用的使用过程中，所以前台应用相对更重要。因此，从用户的角度出发，Android 系统出现卡顿的主要原因就是：应用的无序执行，导致前台应用或服务被使用时需要的资源得不到保证。现有的方法能够识别前台应用和后台应用，并适当为前台应用分配更多资源，但这种分配相对固定。但是在实时场景下，尤其是当资源不足的时候，如何为前台应用或者重要应用临时提供更多资源，以保证这些重要应用的运行，就涉及到如何释放一些被当前非重要应用占用的资源的问题，在解决这个问题的过程中，释放哪些应用的资源以便于临时为重要应用让路就成为了亟待解决的问题。

发明内容

本申请提供资源管理方法以及应用该方法的终端设备等，该方法中包括应用重要性排序方法和资源调度方法，能够识别当前场景下应用的重要性，进而据此实现资源调度，为更重要的应用保证资源供给，从而一定程度上避免系统卡顿，从而提升用户体验。

第一方面，本申请提供一种资源管理方法。该方法可以应用于计算机系统，例如终端设备。终端设备获取数据，该数据包括与当前的前台应用相关的应用时序特征数据以及以下实时数据中的至少一种：所述计算机系统的系统时间、所述计算机系统的当前状态数据和所述计算机系统的当前位置数据。终端设备根据所述实时数据中的至少一种从多个机器学习模型中选择与所述实时数据匹配的目标机器学习模型，这里的多个机器学习模型对应不同的应用使用规律。终端设备将获取的所有数据输入所述目标机器学习模型，通过该目标机器学习模型对所述计算机系统上安装的多个应用进行重要性排序。该重要性排序的结果可以作为该终端设备执行资源管理的决策因素之一。确定目标机器学习模型也可以使用除上述实时数据之外获取到的其他数据。

设置多种机器学习模型，基于采集到的数据在这多种机器学习模型中确定出与终端设备当前场景最相关的机器学习模型，并根据该机器学习模型确定应用的重要性排序，使得应用重要性的排序结果更符合终端设备的当前场景，即重要性排序的实时准确性更高。

另外，与现有技术采集单一或少量数据相比，本申请提供的应用排序是基于采集到的

多种与终端设备的使用相关的数据，数据的多样化也能够提高应用排序的准确性。

在一些实施例中，应用时序特征数据用于表征多个应用被使用的时间顺序的数据。具体的，应用时序特征数据可以包括最近被使用的 k1 个应用，所述前台应用的 k2 个最大可能的前续应用以及 k3 个最大可能的后续应用，其中，k1、k2 和 k3 均为正整数。

在一些实施例中，终端设备根据所述计算机系统的系统时间确定所述计算机系统当前所处的时间段；根据所述计算机系统当前所处的时间段从对应关系中确定与所述计算机系统当前所处的时间段对应的目标机器学习模型，所述对应关系包括多个时间段以及该多个时间段分别对应的多个机器学习模型。

在一些实施例中，终端设备根据所述计算机系统的当前位置数据确定所述计算机系统当前所处的语义位置，然后根据所述计算机系统当前所处的语义位置从对应关系中确定与所述计算机系统当前所处的语义位置对应的目标机器学习模型，所述对应关系包括多个语义位置以及该多个语义位置分别对应的多个机器学习模型。

以上为确定目标机器学习模型的两种方式。多个目标机器学习模型分别对应用户使用应用的多种使用规律。多种使用规律的划分可以根据一个维度划分，例如系统时间或当前位置数据，也可以根据多个维度划分。

在一些实施例中，终端设备根据所述实时数据中的至少两种确定目标机器学习模型。与前两种实现方式不同的是，在这些实施例中，目标机器学习模型对应的使用规律是多个维度划分的结果，例如用户使用规律从时间和地理位置两个维度被划分四种：工作时间且地点为公司、工作时间且地点为出差（非公司）、非工作时间且地点为家、以及非工作时间且地点为娱乐场所（非家）。这四种使用规律呈现出各自不同的特点，所以分别对应一个机器学习模型。终端设备根据实时数据确定一个与当前最符合的机器学习模型。

本申请不对“使用规律”的衡量方法做限定。所谓使用规律不同，即用当前的衡量方法来看不同的使用规律呈现出不同的特征。

在一些实施例中，终端设备还可以根据应用使用历史预测当前场景下用户经常使用的应用个数 N，然后在重要性排序的基础上确定出排序在前的 N 个应用。这样终端设备根据在执行资源管理的时候就可以将这个 N 个应用作为重要应用，在某些情况下可以为这 N 个应用预留资源，或采取一些措施保护这个 N 个应用。

虽然有所有应用的重要性排序结果，但是终端设备执行资源管理的时候选取部分应用，到底选取几个是困难的问题，通过预测用户经常使用的应用个数 N，能够让终端设备执行资源管理的时候目的性更强，更明确。而且这 N 个应用确实是重要性很高的应用，这使得资源管理的合理性更强。

在一些实施例中，终端设备确定重要性排序在前的 N 个应用（或更少或更多），为确定的应用预留资源，或临时冻结剩余的其他应用，或为每个 CPU 创建一个 vip 队列，所述 vip 队列中包括确定出来的这些应用的任务（进程或线程），所述 vip 队列中各个人物的执行优先于所述 CPU 的其他执行队列。

通过本申请提供的方法进行应用排序，需要收集应用使用的历史数据，但是对于新安

装不久的应用，有可能因为应用使用的历史数据过少而导致排序靠后，但这并不能准确代表该新安装应用的真正重要性。

第二方面，本申请还提供以一种新安装应用的重要性排序方法，相当于对新安装应用进行一个重要性的补偿。终端设备根据新安装应用的权重对所述新安装应用进行重要性排序，选取其中排序在前的 N2 个新安装应用，其中所述新安装应用安装到所述计算机系统上的时间小于预设的第二阈值。相应的，终端设备在执行资源管理的时候也可以考虑新安装应用的排序。例如，在执行资源预留的时候在应用数量有限的情况下即考虑前述提到的重要性排序结果中排序在前的部分应用，也考虑一下新安装应用排序结果中排序在前的新安装应用。其他类型的资源管理也类似。

这样就避免了在资源管理时疏忽新安装应用中对用户比较重要的应用的，进一步提高了资源管理的有效性。

在一些实施例中，终端设备根据使用可能性权重和时间衰减权重计算每个新安装应用的得分，得分高的新安装应用的重要性高于得分低的新安装应用的重要性。其中，所述使用可能性权重用于反映新安装应用最近有没有被使用；所述时间衰减权重用于反应当前时间距离应用安装时间的时间差。

第三方面，本申请还提供一种数据采集方法以及根据采集到的数据执行模型训练的方法，可以用来支持其他实施例中提供的应用排序以及资源管理等。

终端设备采集并存储应用数据和所述计算机系统的相关数据，其中，所述应用数据包括所述应用的标识，所述应用被使用的时间，所述计算机系统的相关数据包括以下数据中的至少一种：在所述应用被使用的时间所述计算机系统的时间、状态数据和位置数据。

进一步的，终端设备根据过去一段时间内采集并存储的所述应用数据计算多个应用的应用时序特征数据；将所述应用数据，或所述应用数据和所述计算机系统的相关数据输入分类模型，例如熵增模型，以获得与应用被使用的规律相关的多个分类，其中任意两个分类分别对应的两种规律存在不同；分别针对所述多个分类中的每个分类训练机器学习模型，所述机器学习模型用于实现应用的重要性排序，所述训练的输入包括所述应用被使用的时间、所述应用时序特征数据，以及所述计算机系统的相关数据中的至少一种。

在一些实施例中，模型训练的过程也可以在服务器端执行。终端设备将采集到的数据发送给服务器，由服务器执行模型训练。训练后的模型可以存储在本地，也可以返回给终端设备。若模型存储在服务器侧，那么终端设备可以进行重要性排序时向服务器申请模型。进一步的，重要性排序也可以在服务器端执行，终端设备只需存储排序结果或使用时向服务器申请排序结果。

第四方面，本申请还提供一种资源管理方法，可以应用于计算机系统，例如终端设备。终端设备监测到特定事件时，临时冻结部分应用，直至特定时间段结束，然后解冻被冻结的全部或部分应用。所述特定事件为指示资源需求量升高的事件，例如，应用的启动事件、拍照事件、图库缩放事件、滑动事件以及亮/灭屏事件等。

特定时间段结束仅是解冻的其中一种条件，另一种条件可以是检测到所述特定事件结

束。

进一步的，可能存在一些紧急事件发生，这些事件发生在所述特定时间段结束和所述特定事件结束之前，一旦发生这类型的紧急事件，那么需要提前解冻与紧急事件相关的应用。

在瞬时资源需求量大时，通过临时冻结部分重要性不那么高的应用，释放部分资源，可以保证那些为用户所感知且资源需求量较大的应用的资源供给，从而避免这部分应用出现卡顿的现象，提升用户体验。临时冻结仅是将应用冻结较短的一段时间，接着就释放，而现有技术中通常是将长期不用的应用长期冻结，并在应用被请求时再解冻。虽然都是冻结，但至少在使用场景和冻结的具体方式上存在不同。

在一些实施例中，终端设备通过设置定时器实现临时冻结，所述定时器的时长被设置为所述特定时间段。这种方式代码改动量较小。

在一些实施例中，所述临时冻结的部分应用包括所有后台应用，或所有位于后台且用户不可感知的应用。在另一些实施例中，所述临时冻结的部分应用包括重要性低的应用，其中应用的重要性根据应用的历史使用情况、机器学习算法以及系统的当前场景数据获得。具体的，应用的重要性可根据前述提供的重要性排序的方法获得，选取其中重要性排序靠后的应用执行临时冻结。

第五方面，本申请还提供另一种资源管理方法，可以应用于计算机系统，例如终端设备。该终端设备包括多个物理核，每个物理核都对应一个第一队列和一个第二队列，所述第一队列和第二队列中分别包括一个或多个待所述物理核执行的任务。至少一个物理核执行如下方法：获取并执行所述第一队列中的任务，直至所述第一队列中所有的任务都执行完毕，再获取并执行所述第二队列中的任务。

将重要任务放到一个额外的执行优先性更高的队列中，物理核先执行这些重要任务，从而保证重要任务的资源供给。

在一些实施例中，监测所述第一队列中是否存在等待时间超过特定阈值的任务，若存在，将所述任务移动到另一个物理核对应的第一队列中。

由于 linux 操作系统的限制，实时任务一般不允许从一个物理核移动到另一物理核，这里的重要任务指的是非实时任务。在监测到存在等待超时的重要任务时，将等待超时的重要任务移动到另一个空闲的物理核的第一队列中，从而避免重要任务等待时间过久造成的卡顿现象。

在一些实施例中，所述第一队列中的任务包括重要任务（或称关键任务）以及所述重要任务依赖的任务。所述重要任务为对影响用户体验的任务，或者说用户可感知的任务，或者所重要性高的应用的任务，其中应用的重要性根据应用的历史使用情况、机器学习算法以及系统的当前场景数据获得。任务之间的依赖关系例如为数据依赖、锁依赖或 binder 服务依赖等。

因为重要任务的执行依赖于它所依赖的任务的执行，所以将重要任务及其依赖的任务都放到执行优先性更高的第一队列中，能够进一步提高重要任务的执行速度。

第六方面，针对本申请提供的每种方法，本申请还提供一个相应的装置，该装置包括用于实现方法的各个步骤的模块。该模块的实现可以是软件、软硬结合或硬件。

第七方面，本申请还提供一种终端设备，包括处理器和存储器，所述存储器用于存储计算机可读指令，所述处理器用于读取所述存储器中存储的所述计算机可读指令实现本申请提供的任意一种或多种方法。

第八方面，本申请还提供一种存储介质，具体可以为非易失性存储介质，用于存储计算机可读指令，当一个或多个处理器执行该计算机可读指令时实现本申请提供的任意一种或多种方法。

第九方面，本申请还提供一种计算机程序产品，该产品中包括计算机可读指令，当一个或多个处理器执行该计算机可读指令时实现本申请提供的任意一种或多种方法。

在一些实施例中，本申请中并没有限定重要性“高”或“低”具体是多高或多低，因为本领域技术人员可以理解，在不同情况下有不同的需求。对本申请提供的排序方法而言，应用的“重要性”为应用被用户使用的可能性，可能性越大，重要性越高。但在其他一些实施例中，例如应用资源管控，应用的“重要性”可以根据当下资源管控的状况决定，比如应用的重要性可以与用户是否感知到该应用相关，等等。另外，本申请提供的两种资源管理方法可以依赖于本申请提供的排序方法，也可以不依赖。

解决计算机操作系统资源无序调度的关键就是要系统能够实时的、准确的感知应用的重要性，并根据应用重要性排序实施最优的资源管理策略，确保系统资源物尽其用。简言之，操作系统要站在用户的视角，充分识别用户的使用需求并按需进行资源供给，例如，对于用户正在使用的，充分保障，对于用户将要使用的，提前准备，对于用户当前最不关注的，例如无效自启或关联启动的应用等，充分回收。

本申请提供的应用排序方法，通过实时采集计算机设备（例如智能终端）的多种信息，并分别训练与应用使用规律相关的不同分类下的多种机器学习模型，重要性排序时选择最符合用户使用规律的机器学习模型对应用的重要性进行实时地预测，提高了应用重要性的识别准确度。

进一步的，在识别应用重要性的基础上，进行合理有效地资源管理，从而保障了重要应用的资源供给，提升了计算机设备使用的流畅度，进而提升了用户的使用体验。

附图说明

为了更清楚地说明本申请提供的技术方案，下面将对附图作简单地介绍。显而易见地，下面描述的附图仅仅是本申请的一些实施例。

图 1 为一种终端设备的逻辑结构示意图；

图 2 为一种终端设备中部署的操作系统的逻辑结构示意图；

图 3 为感知管理装置的部分模块的逻辑结构示意图；

图 4 为资源管理方法的概要示意图；

图 5 为资源管理方法中数据采集方法的流程示意图；

图 6 为资源管理方法中模型训练方法的流程示意图；

图 7 为根据一个或多个维度划分应用使用规律的原理和效果的示意图；

图 8 为资源管理方法中应用实时排序方法的流程示意图；

图 9 为资源管理方法中临时冻结方法的流程示意图；

图 10 为就绪队列第一状态的示例图；

图 11 为就绪队列第二状态的示例图；

图 12 为每个 cpu 对应的两个队列的示例图；

图 13 为任务移动方法的流程示意图；

图 14 为任务移动涉及到的队列变化的示例图。

具体实施方式

为了方便理解本申请的实施例，首先在此介绍本申请实施例描述中会引入的几个要素。

操作系统：是管理计算机硬件与软件资源的计算机程序，同时也是计算机系统的内核与基石。操作系统需要处理如管理与配置内存、决定系统资源供需的优先次序、控制输入与输出设备、操作网络与管理文件系统等基本事务。操作系统也提供一个让用户与系统交互的操作界面。 操作系统的型态非常多样，不同机器安装的操作系统可从简单到复杂，可从手机的嵌入式系统到超级计算机的大型操作系统。许多操作系统制造者对它涵盖范畴的定义也不尽一致，例如有些操作系统集成了图形用户界面（graphic user interface, GUI），而有些仅使用命令行界面，而将 GUI 视为一种非必要的应用程序。终端操作系统一般认为是运行在手机、平板电脑、销售终端等终端上的操作系统，例如目前主流的 Android 或 iOS。

应用：也叫应用软件或应用程序，是一种计算机程序，被设计来为用户实现一组相关联的功能、任务或活动。应用在操作系统上部署，具体可以与操作系统的系统软件（例如操作系统）绑定部署，例如系统级别的应用（或称系统服务），也可以独立部署，例如目前常见的文字处理应用（例如 Word 应用）、网页浏览器应用、多媒体播放应用、游戏应用等。

系统资源：本申请中指的是计算机系统内部的资源，包括但不限于内存资源、处理资源和 I/O 资源中的任意一种或多种。对资源的管理包括很多种实现，例如通过对一些应用执行关闭、冻结或压缩等实现资源回收，或者通过拒绝应用启动实现资源的保留，或者通过预加载应用的方式实现资源为该应用的预留。

前台应用、后台应用以及应用的前后台切换：前台应用是运行在前台的应用的简称。后台应用是运行在后台的应用的简称。例如，当前启动了 Word 应用和网页浏览器应用，但是当前用户正在写一封有些操作系统通过两个列表分别管理这两种应用。在 Linux 系统中，当一个应用从前台切换到后台或从后台切换到前台时，会触发一个前后台切换事件，系统可通过监控该前后台切换事件来感知应用的前后台切换。在一些实施例，后台应用由分为后台不可感知应用和后台可感知应用，后台可感知应用是指某些应用即使运行在后台也可以被用户感知到，例如音乐播放应用或导航应用，二者即使后台运行，用户还是能听到音乐或导航的声音。

前续应用和后续应用：一个应用的前续应用是该应用被使用（可以理解为被切换到前台）之前被使用的应用，一个应用的后续应用是该应用被使用之后被使用的应用。应用被

使用意味着应用启动或应用从后台切换到前台，应用启动也可以理解为应用（从关闭）切换到前台。

应用时序特征数据：用于表征多个应用被使用的时间顺序的数据，例如最近被使用的应用有哪些、一个应用的前续应用可能是哪个应用、或者一个应用的后续应用可能是哪个应用等。

系统状态数据：或简称为状态数据，用于表示计算机设备或操作系统本身状态的信息。更具体的，可理解为设备中内置的组件或外接组件的状态数据，例如网络连接状态，或耳机、充电线等外接设备的连接状态等。

位置数据和语义位置数据：位置数据是广义的概念，任意表示位置的信息都可以认为是位置数据，例如经纬度。 将经纬度等较为精确的位置数据赋予实际意义，例如“家”、“公司”、或“娱乐场所”等，这就是语义位置数据。语义位置数据也属于位置数据的一种。

位置数据和系统状态数据也可以统一理解为设备的场景数据。

应用重要性：应用被使用的概率，或者理解为应用被切换到前台的可能性。本申请中对应用重要性的排序，即是对应用被使用概率的排序，这种排序的基础是根据应用使用历史以及一些其他信息的对应用使用概率的预测。

本申请中的“多个”若无特殊说明，表示的是两个或两个以上。本申请中提到的 “数据”或“信息”并没有限定存储方式或格式。

本申请提供的方法主要应用于终端设备，该终端设备（通常为移动终端）也可称之为用户设备（user equipment, UE）、移动台（mobile station, MS）、移动终端（mobile terminal）等，可选的，该终端可以具备经无线接入网（radio access network, ran）与一个或多个核心网进行通信的能力，例如，终端可以是移动电话（或称为“蜂窝”电话）、或具有移动性质的计算机等，例如，终端还可以是便携式、袖珍式、手持式、计算机内置的或者车载的移动装置。举例来说，终端设备可以是蜂窝电话、智能电话、膝上型计算机、数字广播终端、个人数字助理、便携式多媒体播放器、导航系统等。应理解的是，除了移动终端以外，本申请任意实施例提供的方法也可以应用于固定终端，例如个人电脑、销售终端（point of sale, POS）、或自动取款机等；或者也可以应用于服务器等非终端类型的计算机系统。

下面将参照附图更详细地描述涉及本申请的终端设备。需要说明的是，后置用语“单元”和 “模块”仅仅是为了描述的方便，并且这些后置用语不具有相互区分的含义或功能。

请参阅图 1，为本实施例应用的一种终端设备的结构示意图。如图 1 所示，终端设备 100 包括无线通信模块 110、传感器 120、用户输入模块 130、输出模块 140、处理器 150、音视频输入模块 160、接口模块 170、存储器 180 以及电源 190。

无线通信模块 110 可以包括至少一个能使终端设备 100 与无线通信系统之间或终端设备 100 与该终端设备 100 所在的网络之间进行无线通信的模块。例如，无线通信模块 110 可以包括广播接收模块 115、移动通信模块 111、无线因特网模块 112、局域通信模块 113 和位置 (或定位）信息模块 114。

广播接收模块 115 可以经由广播信道从外部广播管理服务器接收广播信号和/或广播相关信息。广播信道可以包括卫星信道和地面信道。广播管理服务器可以是生成并发送广播信号和/或广播相关信息的服务器，或者可以是接收预先生成的广播信号和/ 或广播相关信

息并将它们发送至终端设备 100 的服务器。广播信号不仅可以包括电视广播信号、无线电广播信号和数据广播信号，而且可以包括电视广播信号和无线电广播信号的组合的形式的信号。广播相关信息可以是关于广播信道、广播节目或广播服务提供商的信息，甚至可以通过移动通信网络来提供。在通过移动通信网络来提供广播相关信息的情况下，广播相关信息可以由移动通信模块 111 接收。广播相关信息可以以多种形式存在。例如，广播相关信息可以以数字多媒体广播（digital multimedia broadcasting, DMB)系统的电子节目指南（electronic program guide, EPG)的形式存在，或者以手持数字视频广播（digital video broadcast-handheld, DVB-H)系统的电子服务指南（electronic service guide, ESG)的形式存在。广播接收模块 115 可以使用各种广播系统来接收广播信号。更具体地说，广播接收模块 115 可以使用诸如地面数字多媒体广播（multimedia broadcasting-terrestrial, DMB-T)、 卫星数字多媒体广播(digital multimedia broadcasting-satellite，DMB-S)、媒体前向链路(media forward link only，MediaFLO)、DVB-H 和综合业务地面数字广播（integrated services digital broadcasting-terrestrial, ISDB-T)的数字广播系统来接收广播信号。广播接收模块 115 可以从上述数字广播系统以外的提供广播信号的广播系统接收信号。通过广播接收模块 115 接收到的广播信号和/或广播相关信息可以存储在存储器 180 中。

移动通信模块 111 可以向移动通信网络上的基站、外部终端和服务器中的至少一方发送无线电信号，或者可以从它们中的至少一方接收无线电信号。根据文本/多媒体消息的接收和发送，信号可以包括语音呼叫信号、视频电话呼叫信号和多种格式的数据。

无线因特网模块 112 可以对应于用于无线接入的模块，并且可以包括在终端设备 100 中或从外部连接到终端设备 100。可以使用无线 LAN(WLAN 或 Wi-Fi)、全球微波接入互操作性（world interoperability for microwave access，WiMAX)、高速下行链路分组接入（high speed downlink packet access, HSDPA)等作为无线因特网技术。

局域通信模块 113 可以对应于用于局域通信的模块。此外，可以使用蓝牙(Bluetooth)、射频识别（radio frequency identification, RFID)、 红外数据协会（infrared data association, IrDA)、超宽带（ultra wide band, UWB)和/或 ZigBee 作为局域通信技术。

位置信息模块 114 可以确认或获得移动终端 100 的位置。通过使用全球导航卫星系统（global navigation satellite system ,GNSS)，位置信息模块 114 可以获得位置信息。GNSS 是描述围绕地球旋转并向预定类型的无线导航接收器发送参考信号以使得无线电导航接收器可以确定他们在地球表面上的位置或靠近地球表面的位置的无线电导航卫星系统。GNSS 可以包括美国的全球定位系统（global positioning system，GPS)、欧洲的伽利略系统、俄国的全球轨道导航卫星系统、中国的罗盘系统和日本的准天顶卫星系统等。

GPS 模块是位置信息模块 114 的代表性示例。GPS 模块 114 可以计算关于一个点或对象与至少三个卫星之间的距离的信息和关于测量得到距离信息时的时间的信息，并且可以对获得的距离信息应用三角测量法以在预定时间根据经度、纬度和高度获得关于一个点或对象的三维位置信息。还可以使用利用三个卫星来计算位置和时间信息并利用另一卫星来校正计算出的位置和时间信息的方法。另外，GPS 模块 114 可以实时地连续计算当前位置并使用定位或位置信息来计算速度信息。

传感器 120 可以感测终端设备 100 的当前状态，诸如终端设备 100 的打开/闭合状态、

终端设备 100 的位置、用户是否与终端设备 100 接触、终端设备 100 的方向、和终端设备 100 的加速/减速，并且传感器 120 可以生成用于控制终端设备 100 的操作的感测信号。例如，在滑盖式电话的情况下，传感器 120 可以感测滑盖式电话是打开的还是闭合的。此外，传感器 120 可以感测电源 190 是否供电和/或接口单元 170 与外部装置是否连接。传感器 120 具体可以包括姿态检测传感器、接近传感器等。

用户输入模块 130，用于接收输入的数字信息、字符信息或接触式触摸操作/非接触式手势，以及接收与终端设备 100 的用户设置以及功能控制有关的信号输入等。触控面板 131，也称为触摸屏，可收集用户在其上或附近的触摸操作（比如用户使用手指、触笔等任何适合的物体或附件在触控面板 131 上或在触控面板 131 的操作），并根据预先设定的程式驱动相应的连接装置。可选的，触控面板 131 可包括触摸检测装置和触摸控制器两个部分。其中，触摸检测装置检测用户的触摸方位，并检测触摸操作带来的信号，将信号传送给触摸控制器；触摸控制器从触摸检测装置上接收触摸信息，并将它转换成触点坐标，再送给该处理器 150，并能接收处理器 150 发来的命令并加以执行。例如，用户在触控面板 131 上用手指单击一个应用图标，触摸检测装置检测到此次单击带来的这个信号，然后将该信号传送给触摸控制器，触摸控制器再将这个信号转换成坐标发送给处理器 150，处理器 150 根据该坐标和该信号的类型（单击或双击）执行对该应用的打开操作。

触控面板 131 可以采用电阻式、电容式、红外线以及表面声波等多种类型实现。除了触控面板 131，输入设备 130 还可以包括其他输入设备 132，其他输入设备 132 可以包括但不限于物理键盘、功能键（比如音量控制按键、开关按键等）、轨迹球、鼠标、操作杆、薄膜开关（dome switch)、、滚轮（jog wheel)和拨动开关（jog switch)等中的一种或多种。

输出模块 140 包括显示面板 141，用于显示由用户输入的信息、提供给用户的信息或终端设备 100 的各种菜单界面等。可选的，可以采用液晶显示器（liquid crystal display，LCD）或有机发光二极管（organic light-emitting diode, OLED）等形式来配置显示面板 141。在其他一些实施例中，触控面板 131 可覆盖显示面板 141 上，形成触摸显示屏。

另外，输出模块 140 还可以包括音频输出模块 142、告警器 143 以及触觉模块 144 等。

音频输出模块 142 可以在呼叫信号接收模式、电话呼叫模式或记录模式、语音识别模式和广播接收模式中输出从无线通信单模块 110 接收到的音频数据，或输出存储在存储器 180 中的音频数据。音频输出模块 142 可以输出与在终端设备 100 中执行的功能相关的音频信号，诸如，呼叫信号接收音、消息接收音。音频输出模块 142 可以包括接收器、扬声器、蜂鸣器等。音频输出模块 142 可以通过耳机插孔输出声音。用户可以通过将耳机连接到耳机插孔来收听声音。

告警器 143 可以输出用于指示终端设备 100 的事件的发生的信号。例如，当接收到呼叫信号、接收到消息、输入键信号或输入触摸时，可以生成告警。告警器 143 还可以输出与视频信号或频信号不同形式的信号。例如，通过振动指示事件的发生的信号。

触觉模块 144 可以生成用户可以感觉到的各种触觉效果。触觉效果的一个示例是振动。还可以控制由触觉模块 144 所生成的振动的强度和/或模式。例如，不同的振动可以组合地或顺序地输出。触觉模块 144 可以生成多种触觉效果，除了振动以外，还可以为相对于皮肤表面垂直移动的针列的刺激效果、通过喷孔或吸孔形成的空气喷力效果或空气吸力效果、

摩擦皮肤的刺激效果、电极接触的刺激效果、使用静电力的刺激效果、和利用能够吸热或放热元件再现热或冷的效果等多种效果中的一种或多种。触觉模块 144 不仅可以通过直接接触来发送触觉效果，而且可以允许用户通过用户的手指或手臂的肌觉来感觉触觉效果。终端设备 100 可以包括多个触觉模块 144。

处理器 150 可以包括一个或多个处理器，例如，处理器 150 可以包括一个或多个中央处理器，或者包括一个中央处理器和一个图形处理器。当处理器 150 包括多个处理器时，这多个处理器可以集成在同一块芯片上，也可以各自为独立的芯片。一个处理器可以包括一个或多个物理核，其中物理核为最小的处理模块。

音视频输入模块 160，用于输入音频信号或视频信号。音视频输入模块 160 可以包括摄像头 161 和麦克风 162。摄像头 161 可以处理图像传感器在视频电话模式或拍摄模式中获得的静止图像或运动图像的图像帧。处理后的图像帧可以显示在显示面板 141 上。

经过摄像头 161 处理的图像帧可以存储在存储器 180 中或者可以通过无线通信模块 110 发送到外部设备。终端设备 100 还可以包括多个摄像头 161。

麦克风 162 可以在呼叫模式、记录模式或语音识别模式中接收外部音频信号，并将接收的音频信号处理为电子音频数据。该音频数据可以接着变换为可以通过移动通信模块 111 发送到移动通信基站的形式，并在呼叫模式中输出。麦克风 162 可以采用各种噪声消除算法(或噪声抵消算法)，以消除或降低在接收外部音频信号时所产生的噪声。

接口模块 170 可以用作连接到终端设备 100 的外部设备的通路。接口模块 170 可以接收来自外部设备的数据或电力并将数据或电力发送到终端设备 100 的内部组件，或者向外部设备发送终端设备 100 的数据。例如，接口模块 170 可以包括为有/无线头戴式耳机端口、外部充电器端口、有线/无线数据端口、存储卡端口、用于连接具有用户识别模块的设备的端口、音频 I/O 端口、视频 I/O 端口和/或耳机端口。

接口模块 170 还可以与用户识别模块连接，用户识别模块是存储用于验证使用终端设备 100 的权利的信息的芯片。包括用户识别模块的识别设备可以制造成智能卡的形式。因而，识别设备可以经由接口模块 170 与终端设备 100 连接。

接口模块 170 还可以是在终端设备 100 与外部托架连接时将来自外部托架的电力提供给终端设备 100 的通路，或者是将用户通过托架输入的各种命令信号发送给终端设备 100 的通路。从托架输入的各种命令信号或电力可以用作确认终端设备 100 是否被正确地安装在托架中的信号。

存储器 180 存储计算机程序，该计算机程序包括操作系统程序 182 和应用程序 181 等。典型的操作系统如微软公司的 Windows，苹果公司的 MacOS 等用于台式机或笔记本的系统，又如谷歌公司开发的基于 Linux 的安卓（Android）系统等用于移动终端的系统。处理器 150 用于读取存储器 180 中的计算机程序，然后执行计算机程序定义的方法，例如处理器 150 读取操作系统程序 182 从而在该终端设备 100 上运行操作系统以及实现操作系统的各种功能，或读取一种或多种应用程序 181，从而在该终端设备上运行应用。

操作系统程序 182 中包含了可实现本申请任意实施例提供的方法的计算机程序，从而使得处理器 150 读取到该操作系统程序 182 并运行该操作系统后，该操作系统可具备本申请提供的应用实时排序功能和/或资源管理功能等。

存储器 180 还存储有除计算机程序之外的其他数据 183，例如本申请采集获得的应用信息、训练得到的模型以及实时排序的结果等信息，还例如临时存储输入/输出的数据(例如，电话簿数据、消息、静态图像和/或运动图像)，当对触摸屏施加触摸输入时输出的各种模式的振动和声音有关的数据等。

存储器 180 可以是以下类型中的一种或多种：闪速（flash)存储器、硬盘类型存储器、微型多媒体卡型存储器、卡式存储器(例如 SD 或 XD 存储器)、随机存取存储器（random access memory, RAM)、静态随机存取存储器（static RAM, SRAM)、只读存储器（read only memory, ROM)、电可擦除可编程只读存储器（electrically erasable programmable read-only memory，EEPROM)、可编程只读存储器(programmable ROM, PROM)、磁存储器、磁盘或光盘。

在其他一些实施例中，存储器 180 也可以是因特网上的网络存储设备，终端设备 100 可以对在因特网上的存储器 180 执行更新或读取等操作。

电源 190 可以在处理器 150 的控制下接收外部电力和内部电力， 并且提供终端设备 100 的各个组件的操作所需的电力。

各个模块的连接关系仅为一种示例，本申请任意实施例提供的方法也可以应用在其它连接方式的终端设备中，例如所有模块通过总线连接。

本申请提供的方法可用硬件或软件来实现。在硬件实现方式下，可以使用专用集成电路（application specific integrated circuit, ASIC)、数字信号处理器（digital signal processor, DSP)、可编程逻辑器件（programmable logic device, PLD)、现场可编程门阵列（field programmable gate array, FPGA)、处理器、控制器、微控制器和/或微处理器等电子单元中的至少一个来实现本申请的实施方式。在软件实现方式下，诸如过程和功能的实施方式可以使用执行至少一个功能和操作的软件模块实现。软件模块可以以任意适当的软件语言编写的软件程序来实现。软件程序可以存储在存储器 180 中，并由处理器 150 读取并执行。

图 2 以安卓（Android）系统为例，介绍本申请提供的方法的一种实现方式。如图所示，典型的 Android 系统 200 包括应用 210、应用框架 220、系统运行库和安卓运行环境 230 以及内核 240。Android 是基于 Linux 开发的，所以此内核 240 为 Linux 内核。

应用 210 包括浏览器、媒体播放器、游戏应用、文字处理应用等各种应用，有些是系统自带的应用，有些是用户根据需要安装的应用。本申请重点关注这些应用被用户使用的情况，从而为用户最可能使用的应用保障资源供给，避免用户视角的应用卡顿现象。

应用框架 220 包括安卓服务 224，安卓服务包括安卓系统提供的多种系统服务，供其他模块调用使用。例如电源管理器 224-a、通知管理器 224-b、连接管理器 224-c、包管理器 224-d、位置管理器 224-e、有线访问管理器 224-g、蓝牙设备 224-h、视图系统 224-i 等。这些管理器均可以参考现有技术中安卓系统提供的模块实现，本申请不再详述。

系统运行库和安卓运行环境 230 包括系统运行库 231 和 Android 运行环境 232。系统运行库 231，也叫程序库，包含一些 C/C++库，这些库能被 Android 系统中不同的组件使用。

Android 运行环境 232 包括核心库和 Dalvik 虚拟机。该核心库提供了 Java 编程语言核心库的大多数功能。每一个 Android 应用程序都在它自己的进程中运行，都拥有一个独立的 Dalvik 虚拟机实例。Dalvik 被设计成一个可以同时高效运行多个虚拟系统的设备。Dalvik

虚拟机依赖于 Linux 内核 240 的一些功能，比如线程机制和底层内存管理机制等。

Android 的核心系统服务依赖于 Linux 内核 240，如安全性，内存管理，进程管理， 网络协议栈和各种驱动模型。 Linux 内核 240 也同时作为硬件和软件之间的抽象层。另外，Android 还对 Linux 内核做了部分修改，主要涉及两部分修改：

Binder (IPC)驱动器：提供有效的进程间通信，虽然 Linux 内核本身已经提供了这些功能，但 Android 系统很多服务都需要用到该功能，为了某种原因实现了自己的一套。

电源管理：主要用于省电。因为 Android 系统是为移动终端，例如智能手机设计的，低耗电是一个重要目的。

Android 系统以软件代码的形式存储在存储器 180 中，处理器 150 读取并执行该软件代码以在终端设备 100 上实现该系统提供的各项功能，其中包括本实施例提供的功能。

本实施例涉及的改进主要在应用框架 220。如图所示，除安卓原有的系统服务之外，本实施例的应用框架 220 还包括感知管理装置，该感知管理装置包括数据采集模块 221、应用排序模块 222、以及决策执行模块 223。

应理解的是，感知管理装置也可以作为 Android 的系统服务提供给其他组件使用。其中的模块 221-223 可以分别做三个系统服务，也可以适当合并或进一步细分。

数据采集模块 221 采集终端设备 100 的场景数据，该场景数据包括设备所处的位置数据（例如 GPS 位置数据）、设备中内置的组件或外接组件的状态数据等。该状态数据例如可以是显示面板布局状态、网络连接状态、耳机或充电线连接状态、摄像头/音视频组件/传感器的状态等。可参考图 1 了解终端设备 100 的多种组件。这里的“组件”即包括硬件也包括软件。

数据采集模块 221 还采集与应用的使用相关的应用数据，例如应用类型、应用名称、应用被使用的时间等。采集的数据直接或经过处理后被存储到存储器 180 或发送给其他模块。

应用排序模块 222 根据数据采集模块 221 实时采集的数据和机器学习模型，确定实时场景下所有应用的重要性排序结果。在实现实时重要性排序之前，应用排序模块 222 还根据数据采集模块 221 采集的历史数据进行训练获得上述机器学习模型。

决策执行模块 223：根据应用排序模块 222 输出的应用的实时重要性排序制定系统资源如何管理的决策，并直接执行或调用其他模块执行相应的资源管理措施。应理解的是，在具体实现中，决策执行模块 223 在并非在做所有资源管理决策时都需要应用的实时重要性排序。

需要说明的是，当以上模块需要存储中间数据或最终数据时，需要存储功能，这些存储功能可以由独立的存储模块实现，也可以融合在三个模块中分别实现。

在其他一些实施例中，上述模块 221-223 也可以在系统运行库和安卓运行环境 230 或内核 240 实现，也可以部分在应用框架 220 实现，部分在其他层次实现。

如图 3 所示，应用排序模块 222 包括两大子模块：模型训练模块 310 和实时排序模块 320。

模型训练模块 310 根据数据采集模块 221 采集的历史数据训练出应用于应用实时排序的模型，并将其存储到存储器 180 中，供实时排序模块 320 做实时的应用排序时调用。具

体的，模型训练模块 310 主要训练三个模型：维度划分模型、排序模型以及 N 值预测模型。本申请中的“模型”是广义的概念，参数、公式、算法或对应关系等都可以认为是模型。

维度划分模型用于对应用使用规律进行分类，该分类可以是一维分类也可以是多维分类。“维度”是使用规律分类的依据，例如时间、空间等。以时间为例，以一天 24 小时为划分依据，将应用使用规律划分为工作时间段和非工作时间段两个分类。具体的，根据当前的系统时间和训练获得的维度划分模型确定终端当前属于工作时间段还是非工作时间段。这里实际反映的即是用户当前处于工作状态还是非工作状态，在这两种状态下用户对应用的使用规律呈现出不同的特点。进一步的，维度还可以是多维，比如时间维度和地点维度。例如工作时间段又划分为公司办公、出差办公；非工作时间段又划分为家和出游等。为方便理解， 这里的举例采用了带有语义的描述，例如“办公”、 “出游”等，应理解的是具体实现中并不以此为限。

N值预测模型是用来确定特定分类下用户经常使用的应用的个数的。N的值为正整数。以时间维度分类为例，N 值反映的是特定时间段内用户经常使用的应用的个数。

实时排序模块 320 是一个实时处理模块，它能够根据数据采集模块 221 实时采集的的数据和模型训练模块 310 训练的模型确定应用的实时重要排序。实时排序模块 320 利用模型训练模块 310 提供的功能完成三个功能：维度划分预测 321、应用排序预测 322 以及 N 值获取 323。

应理解的是，模型训练模块 310 和实时排序模块 320 内的每个功能都可以看做是一个功能模块或单元。

如图 3 所示，数据采集模块 221 可能会对采集到的全部或部分数据进行清洗/加工等处理，再存储到存储器 180 中或提供给实时排序模块 320。

需要说明的是，以上几个模块或功能并非在所有实施例中都要应用，在本申请的一些实施例中，仅选择部分功能实现也可以。

下面介绍本实施例提供的应用排序方法以及资源管理方法，亦即前述模块或单元的具体实现。

如图 4 所示，为本实施例提供的应用排序方法以及资源管理方法的全局方案示意图。该图示出了四个主要方法流程，包括第一阶段数据采集、第二阶段模型训练、第三阶段应用排序以及第四阶段资源管理。在描述的过程中，为了给出方案的全局概貌，将按照以上顺序进行描述，但应理解的是，在方案的具体实现中，这四个阶段并非全然是串行执行的。

终端设备运行之后，先经过第一阶段即数据采集的过程，采集的主要是用户使用应用的数据以及终端设备的系统状态数据、位置数据等信息，这些信息可以反应用户对应用的使用规律，用于在将来预测用户对应用的使用概率。采集到的数据部分会直接存储，部分可能需要加工后再存储（①）。第一阶段的执行主体是数据采集模块 221。

经过一段时间的数据采集之后，进行第二阶段的模型训练。针对排序模型本申请采用的是机器学习的训练方法，一般需要一定量的历史数据输入（②），但是具体需要采集多长时间才启动第一次训练，本申请不做限定。机器学习模型的训练过程就是模型建立过程，

实质是一个参数学习与调优的过程。对模型进行训练，便是模型参数的学习更新。训练出来的机器学习模型或者模型参数会被存储起来，用于第三阶段排序使用。

在排序模型的训练之前基于机器学习算法或其他算法将采集的历史数据从一个维度或多个维度分为多类，多个类别中的应用使用规律呈现出不同的特征，针对这多个类别分别训练各自的排序模型。因此第二阶段一个重要的输出就是多个分类以及多个分类分别对应的多个机器学习模型（或模型参数）（③）。第二阶段的执行主体是模型训练模块 310。进一步的，模型训练模块 310 还根据 N 值预测算法计算用户在特定分类下经常使用的应用个数 NB 并存储起来用于后续资源管理。 N 值预测的过程也可以放到第三阶段或第四阶段。

终端设备可周期性启动训练过程，或者在指定的时间点上启动训练过程，或者按采集到数据简单不匹配原则启动训练过程等。

第二阶段模型训练完成之后就可以进行第三阶段应用排序。第三阶段根据实时数据（④）第二阶段输出的机器学习模型（⑤）来进行应用的排序。除了这两项之外，还需要一些历史数据（⑤）的辅助。实时数据包括实时采集到的当前应用以及当前应用的使用时间、终端设备的当前时间（即当前的系统时间）、终端设备的当前状态数据和终端设备的当前位置数据等其中的一种或多种，而有些数据，例如多个应用被使用的时间顺序则还需要历史数据中记载的多个应用的使用时间来计算获得。第三阶段的执行主体是实时排序模块 320，输出的是所有安装在终端设备上的应用的排序结果（⑥）。

第三阶段应用排序的启动可以是周期性的，也可以是事件触发性的。其中，触发事件可以是应用切换、应用安装、应用卸载、终端设备状态变化等，这些对用户来说都可能造成应用重要性的变化。这种情况下排序后的结果可以存储起来，供第四阶段资源管理需要的时候使用。另外，触发事件也可以是第四阶段资源管理触发的请求事件，即在资源管理的调用下执行应用排序。

第四阶段资源管理，可以访问并使用存储器中存储的最新的排序结果（⑦），也可以根据需求实时调用（⑧）第三阶段应用排序获得排序结果。然后，根据应用的重要性排序对资源进行管理。

下面示例性地分别介绍这四个阶段的详细实现方式。

请参考图 5 为第一阶段数据采集过程的示意图。首先数据采集模块 221 设置事件监听机制（S501），用于监听关键事件是否发生，若检测到关键事件（S502），则触发数据采集过程（S503）。采集的数据可以直接存储（S505），也可以经过清洗/加工（S504）后再存储。

数据清洗(Data cleaning)指的是计算机对数据进行重新审查和校验的过程，目的在于删除重复信息、纠正存在的错误，并提供数据一致性。数据加工指的是计算机对数据进行类型、格式等变化，或对数据进行数学变换等处理。

一方面，数据采集模块 221 采集终端设备 100 的场景数据和应用使用相关的应用数据历史数据，采集到的数据经过清洗/加工后存储到存储器 180 中，作为历史数据供模型训练模块 310 训练生成各种模型。具体的，数据采集模块 221 实时监控终端设备的状态、应用使用情况等，当条件满足时采集数据并存储。例如，数据采集模块 221 实时监控和缓存用户行为状态（也可以理解为终端设备行为状态，例如运动或静止）、应用使用状态、系统状

态和位置状态的改变。当发生应用前后台切换或其他类型的关键事件之后，将切到前台的应用包名、当前时间以及上述状态的最新数据等信息记录到持久化存储介质中。

另一方面，数据采集模块 221 还在需要排序时实时采集数据，采集的数据作为实时数据成为实时排序模块 320 的输入，以便于实时排序模块 320 对应用进行实时排序。

举例来说，数据采集模块 221 采集的数据以及采集方式如下表所示。应理解的是，在一些实施例中，可以根据需要酌情减少或增加一些数据，本实施例只是举例，无意以此表中的数据为限。

表 1

数据类别	采集方式
时间	获取当前系统时间
应用切换到前台的时间	通过采集当前系统时间或获取应用切换时的时间戳获得。
前台应用的包名	使用包管理器 224-d（PackageManager）所提供的getApplicationInfo方法来获取应用包名。 采集涉及模块：PackageManager
蓝牙连接状态	获取设备所绑定的蓝牙设备列表，判断是否存在处于连接状态的蓝牙设备。也可监听BluetoothDevice.ACTION_BOND_STATE_CHANGED来被动获取。 采集涉及模块：蓝牙设备224-h（BluetoothDevice）
网络连接状态	使用 CONNECTIVITY_SERVICE 系统服务所提供的getActiveNetworkInfo 方法获取当前的网络状态。也可通过注册ConnectivityManager.CONNECTIVITY_ACTION 的 Action 来被动获取。 采集涉及模块：连接管理器224-c（ConnectivityManager）
通知列表状态	使用系统服务 NOTIFICATION_SERVICE 所提供的getActiveNotifications方法获取当前状态条中驻留的通知信息列表，遍历上述列表使用getPackageName方法获取通知信息所关联的应用包名。也可通过NotificationListenerService的方式被动获取。 采集涉及模块：通知管理器224-b（NotificationManager）.
充电线连接状态	通过向电源管理器（BatteryManager）发送特定 Intent（ACTION_BATTERY_CHANGED），来获取当前是否连接充电线。也可通过注册android.intent.action.ACTION_POWER_CONNECTED、android.intent.action.ACTION_POWER_DISCONNECTED 来被动获取。 采集涉及模块：电源管理器224-a（BatteryManager）
耳机线连接状态	监听系统广播（ACTION_HEADSET_PLUG），当连接状态发生

	改变时缓存最新状态。 采集涉及模块：有线访问管理器224-g（WiredAccessoryManager）
GPS信息	监听位置变化（LocationListener），当位置发生变化时缓存最新的位置数据。 采集涉及模块：位置管理器224-e(LocationManager）
应用类型	使用系统服务PackageManager所提供的getApplicationInfo方法来获取应用包名的flags和hwFlags属性，结合这两者属性来判断应用的类型（也叫内置类型）。 采集涉及模块：**PackageManager**
应用布局信息	访问桌面布局数据库，查询包名所对应应用在桌面的布局信息。注：用户使用第三方桌面应用时，该方法无法获取，因此需要判断当前系统桌面应用包名。 采集涉及模块：视图系统224-i（**View System**）

Android 系统提供了函数接口来获取一些数据，表 1 中描述了这些数据的一种获取方法所涉及到的模块（例如 PackageManager）或函数（例如 getActiveNotifications），可参考图 2，本领域技术人员可以根据需要使用其他获取方式，本申请并不限制。除了 Android 系统之外，在其他系统中所调用的模块和函数可能是不一样的，本申请也不以 Android 系统为限。

表 1 中前台应用指的就是当前运行在前台的应用，这种应用被认为是当前用户正在使用的、对用户而言相对重要的应用。有些应用运行在后台用户也会使用，比如音乐播放引用，对此类型应用的采集时间可以放到此类应用启动，即第一次切换到前台的时刻。

监测到关键事件时触发以上数据采集过程，此处的关键事件可以包括以下事件中的一种或多种：前后台切换事件、应用的安装事件、应用的卸载事件、系统状态发生变化引起的通知事件或地理位置信息发生变化引起的通知事件。其中，地理位置信息发生变化也可以是先识别语义地理位置是否发生变化，例如是否从家变化到了办公地，如果语义地理位置发生变化再启动数据采集。关键事件不局限于以上示例。概括来说，关键事件主要取决于需要采集的信息，如果监控到需要采集的信息可能发生变化，就可能需要启动上述数据采集过程。

采集的数据中应用切换到前台的时间有可能就是当前系统时间，比如监测到前后台切换事件时执行数据采集，那么采集的当前系统时间就可以认为是当前前台应用切换的时间。如果在其他事件的触发下执行数据采集，那么可以通过当前前台应用在切换时记录的一个用于表示切换时间的时间戳来获取应用切换到前台的时间。该时间戳就是应用在被切换到前台时利用当时的系统时间记录的。

本申请中“应用切换到前台的时间“有时也称为 “应用被使用的时间”。

采集的数据从最后的使用方式上又可以分为两类：1）直接可使用的数据，例如表 1 中示例的蓝牙/网络/耳机/充电线连接状态、通知列表状态和应用类型等数据。这些数据通过特征提取就可以当成训练模型的输入参数直接使用。 2）需要经过加工的数据，主要有两类：与应用时序特征相关的信息和 GPS 位置信息，这些数据需要进一步加工转换成其它的

形式作为参数输入。下面主要就第 2）类作详细介绍。

GPS 位置信息，需要针对语义位置进行聚类，例如家和办公地。当一旦发生这两种地理位置的转换，系统将会推送一条语义位置信息发生变更的广播。在其他一些实施例中，GPS 位置信息也可以直接使用。

应理解的是，数据加工过程可以发生在存储之前，也可以在模型训练使用该数据之前再做加工。

与应用时序特征相关的信息主要包括前台应用的包名和应用切换到前台的时间。通过收集这两类信息的历史记录（可以认为是用户使用应用的轨迹记录），进一步可以统计得到以下三类信息：a）最近被使用的 k1 个应用，b）当前前台应用的 k2 个最大可能的前续应用以及，c）当前前台应用的 k3 个最大可能的后续应用。k1、k2 和 k3 的取值可以相等也可以不相等。a）b）c）三类信息统称为应用时序特征数据。

a)是当前前台应用被使用之前用户最近使用的 k1 个应用，可以直接根据存储的历史数据中各个应用以及各个应用的使用时间获得。例如，假设 k1=2，当前应用使用时间是 18:00，根据各个应用的使用时间确定在当前应用使用时间之前最近使用应用的时间分别是 15:45，15:30，那么 k1 个应用就是 15:45 和 15:30 使用的这两个应用。在其他一些实施例中，k1 也可以包括当前前台应用。

b）和 c）则需要从用户使用应用的历史轨迹里利用数学方法统计出应用的多阶关联关系。下面就多阶关联关系的统计模型作描述。

本实施例用矩阵 U[M*M]来表示终端设备 100 上安装的 M 个应用的关联关系。U [i*j] 表示从应用 i 直接切换到应用 j，这种切换的次数，其中 i 和 j 均为小于或等于 M 的正整数。例如，在某一刻发生了应用 i 切换到应用 j 的操作，那么 U [i*j] 这条记录加 1。

需要说明的是，这里针对的是前台应用，即当前位于前台的应用从应用 i 直接变成了应用 j，换句话说，用户使用应用 i 接着又使用了应用 j，这即是应用 i 直接切换到应用 j。

进一步的，应用间的跳转次数也可以考虑在内。应用 i 直接切换到应用 j 说明二者可能强相关，但应用 i 经过几次跳转之后切换到应用 j 也能体现二者之间一定可能性的关联。基于这种实现，可以增加一个跳转次数参数，例如发生过应用 i 经过 d 次跳转切换到应用 j，则 U[i,j][d]这条记录加 1。跳转次数太多，说明应用之间的关联关系很弱，可以不考虑，所以可以设置最多 D 次跳转 (例如 D 取值为 5)。

那么，应用 i 与应用 j 的关联矩阵中每个元素 $U'[i,j]$ 就可以被定义为：

$$U'[i,j]=\alpha U[i,j][0]+\beta U[i,j][1]+\ldots+\gamma U[i,j][D] \qquad (1)$$

基于上述描述方法：$U'[,j]$（矩阵的第 j 列）则可表示从其它应用跳转到应用 j 的可能性；$U'[i,]$（表示矩阵的第 i 行）则可表示其它应用从应用 i 跳转过来的可能性。从这个矩阵中可以获取到一个应用最大可能的 k2 个前续应用和最大可能的 k3 个后续应用。例如，当前前台应用为应用 v, 则可以从 $M'[,v]$ 中依次挑选出最大的 k2 个值，这 k2 值对应的行值就是应用 v 最大可能的 k2 个前续应用；从 $U'[v,]$ 中依次挑选出最大的 k3 个值，这 k3 值对

应的列值就是应用 v 最大可能的 k3 个后续应用。

应用时序特征数据和其他采集的数据一起存储到存储器 180 中，供下一阶段模型训练使用。在应用实时排序的过程中也会用到上述方法去获得前台应用对应的应用时序特征数据。

请参考图 6 为第二阶段模型训练的过程示意图。经过一段时间的数据采集之后，存储器 180 中已经存储了一些数据，这些数据反映了应用使用的规律。这种规律需要通过机器学习的方式形成可重复利用的模型。 模型训练过程即是做这个。如图所示，从存储器 180 中读取数据（S601），对数据进行特征提取(S602)等操作，然后训练维度划分模型，即将用户对应用的使用习惯按照一个或多个维度进行分类（S603）。分类之后针对不同的分类分别训练排序模型（S604），并将训练结果存储到存储器中，供第三阶段应用排序使用。进一步的，该阶段还可以进行用户经常使用的应用个数 N 的预测(S605)。下面分别对涉及的三个模型算法进行介绍。

a）训练维度划分模型（S603）

用户使用应用的习惯在某些特维度上可能存在断裂，例如在时间维度上，用户工作时间段和非工作时间段（也可称之为休息时间段）使用应用的习惯就会有很大差异，这种情况下如果笼统地构造一个排序模型去对应用进行排序，那么就可能出现极大的偏差。训练维度划分模型即是探索在某些维度上去对用户习惯作更精细的划分，消除这种断裂带来的误差，进一步提升排序的精准度。

对于一个或多个维度的不同的分类，可通过对原始数据应用一些划分算法获得一个划分结果。如图 7 所示，原始数据从维度 X 上被划分为三种分类。进一步的，每种分类还可以在维度 Y 上再划分出更多分类，不再详述。这也是下面将介绍的熵增算法的基本原理。

本实施例将以时间维度为例，将时间划分为工作时间段和非工作时间段两类。以一天时间为准，方法就是找到一个或者两个时间点，能够将用户的工作和非工作时间区分开来。具体实现上又可以分为：1）单线划分，以[0:00~x]、[x~24:00]形式的划分。只有一个变量 x； 2）双线划分，以[0:00~x_1]、[x_1~x_2]、[x_2~24:00]形式的划分，需要两个变量 x_1 和 x_2。下面以双线划分为例。

首先从存储器 180 读取数据。这里用到的数据主要有应用包名和应用被使用的时间。

将一天时间划分成 24 个时间段，每个时间段包括 1 小时。根据读取的数据统计每个时间段内用户使用应用的情况。本实施例中用二维矩阵 S 来统计任意时间段内应用的使用情况， 例如 $S[h][i]$ 就表示了在 h 时间段，应用 i 总共被使用的次数。从而，一段时间 $[h_1,h_2]$ 内，应用 i 的应用总共被使用的次数 $Sum_{[h_1,h_2]}(i)$，以及该时间 $[h_1,h_2]$ 内应用 i 被使用的次数占所有应用被使用的次数的占比 $f_{[h_1,h_2]}(i)$ 可分别由下面公式（2）和（3）两式计算获得：

$$Sum_{[h_1,h_2]}(i)=\sum_{h\in[h_1,h_2]}S[h][i] \quad (2)$$

$$f_{[h_1,h_2]}(i)=Sum_{[h_1,h_2]}(i)/\sum_i Sum_{[h_1,h_2]}(i) \quad (3)$$

$f_{[h_1,h_2]}(i)$也可以称之为应用 i 在该时间$[h_1,h_2]$内被使用的频率。

从而，该段时间$[h_1,h_2]$内用户使用应用的信息熵$E_{[h_1,h_2]}$定义如下：

$$E_{[h_1,h_2]}=\sum_i f_{[h_1,h_2]}(i)*(-1)*\log_2 f_{[h_1,h_2]}(i) \quad (4)$$

这样，x_1和x_2的一次双线划分从而产生的熵计算如下：

$$E(x_1,x_2)=f(0,x_1)*E_{[0,x_1]}+f(x_1,x_2)*E_{[x_1,x_2]}+f[x_2,23]*E_{[x_2,23]} \quad (5)$$

其中$f(x_1,x_2)$为[x_1, x_2]这一段时间内所有应用的使用次数在全天所有应用的使用次数中的占比，计算如下：

$$f(x_1,x_2)=\sum_{h\in[x_x,x_x]}\sum_i S[h][i]/\sum_{h\in[0,23]}\sum_i S[h][i] \quad (6)$$

从而，问题的求解最终被转化成了寻找两个划分时间x_1和x_2，使得这种划分下的熵值最小，即：

$$\arg\min E(x_1,x_2) \quad (7)$$

举例来说，最后得出的结果可能是x_1和x_2分别为 9:00 和 18:00，那么三段分别是[0:00~9:00]非工作，[9:00~18:00]工作，[18:00~24:00]非工作，用表 2 表示为：

表 2

[0:00~9:00]	非工作
[9:00~18:00]	工作
[18:00~24:00]	非工作

在本实施例中将[0:00~9:00]和[18:00~24:00]均认为是非工作时间段，针对非工作时间段训练了一个排序模型。在其他一些实施例中，也可以针对[0:00~9:00]和[18:00~24:00]分别训练一个排序模型。另外，需要说明的是“工作”和“非工作”的语义是根据当前大多数人的生活习惯给予的，实际上该方法的意义是反映了应用被使用的历史呈现出两种不同的规律，本申请并不限定一定要采用“工作”和“非工作”这种语义划分。

b）训练排序模型（S604）

终端设备上应用的重要性和应用的使用概率相关，使用概率越高，重要性越大。应用的使用概率在系统的不同状态下又可能不同。例如，网络连接的状态下，各个应用的使用概率和网络断开的状态下各个应用的使用概率是不同的，这就导致网络连接和断开情况下应用重要性排序是不同的。所以在对应用进行排序时需要考虑当前系统状态。

需要说明的是，网络连接或断开状态仅是举例，与终端设备相关的其他状态也可以在考虑范围之内，比如耳机是否连接，蓝牙是否连接、充电线是否连接或终端设备当前的位置等。

概括来说，训练排序模型主要包括以下几步。

读取数据。从存储器 180 中读取数据。这里训练需要用到的数据主要包含：应用被使用的时间、应用时序特征信息、系统状态信息和语义位置信息。应用时序特征信息在前面已经详述内容和获取过程。系统状态信息主要包括：耳机/充电线/网络的连接状态。语义位置信息例如可以包括家和办公室等。

特征提取。通过统计、识别和/或矢量化等手段，将读取的数据转换成机器学习算法可处理的特征向量/特征矩阵。具体包括：s1）应用被使用的时间和应用时序特征信息：使用矢量化、时间片和工作日/非工作日识别方法将应用被使用的时间和应用时序特征信息转换成向量。例如，应用被使用的时间被分成两部分，日期和时间段，将周一，周二，……周日分别映射为 0，1，……，6，时间段则是分段映射，比如[9:00~18:00]是一段，所有这个时段的时间映射为同一个数字。s2）系统状态信息：从读取的系统状态信息中抽取特征，使用离散/枚举化方法将系统状态信息转换成向量。s3）语义位置信息：将读取的语义位置信息进行编码从而转换成向量。 s4）归一化处理，使用最大/最小值法对数值进行[0,1]归一化，并使用白化方法对数值进行方差归一化。s6）特征矩阵组合，将各个维度的数据合并为特征矩阵，提供给算法训练模块。

模型训练与存储。将数据划分成多组，分别进行模型训练获得多个模型。数据的划分也可以发生在特征提取之前，对数据划分之后分别进行特征提取。举例来说，按照本申请前述实施例提供的时间维度上的划分，将周一至周五[9:00~18:00]的数据和周一至周五[0:00~9:00] [18:00~24:00]的数据分别训练两个模型，并将生成的排序模型（机器学习模型参数信息）和对应的时间段存储到数据库中去。周六和周日全天的数据可以作为非工作时间段去训练。

存储结果的一个例子如下：

表 3

[9:00~18:00] （工作时间段）	机器学习模型 1 的参数信息
[0:00~9:00] [18:00~24:00] （非工作时间段）	机器学习模型 2 的参数信息

下面以机器学习算法 softmax 为例详细介绍一下训练过程。选择 Softmax 算法的原因是该算法是增量式的学习算法。增量式学习算法意味着训练过程可保证在原先学习的模型参数基础上，融合新训练样本的特征数据，对参数进行进一步的调优，即无需保留原有的样本数据，这样可减少数据的存储量，节省存储空间。

通过统计、识别和/或矢量化等手段，将获取的应用时序特征信息、系统状态信息等数据转换成 softmax 算法可处理的特征向量 s=（s1,s2,.....sW）， W 为特征向量的数量（或长度）。应用集合使用一个向量表示 a= (a1,a2,.....aM)，aj 表示第 j 个应用，M 表示手机上安装的应用个数。目的是求出排序函数（即排序模型）$h_\theta(s)$，当然求出排序函数的前提是求出参数θ。

因此，根据 softmax 算法，排序函数定义为

$$h_\theta(s) = \frac{1}{\sum_{i=1}^{M} e^{\theta_i s}} \begin{bmatrix} e^{\theta_1 s} \\ \vdots \\ e^{\theta_M s} \end{bmatrix} \qquad (8)$$

此模型的代价函数为

$$J(\theta) = -\frac{1}{k}\left[\sum_{i=1}^{k}\sum_{j=1}^{M} 1\{y^{(i)} = j\} log \frac{e^{\theta_j x^{(i)}}}{\sum_{l=1}^{M} e^{\theta_l x^{(i)}}}\right] + \frac{\lambda}{2}\sum_{i=1}^{M}\sum_{j=1}^{W}\theta_{ij}^2 \qquad (9)$$

其中，k 表示训练输入的样本数量，M 表示类别数目，即应用个数，W 表示特征向量 s 的长度。λ表示权重（$\lambda > 0$）。1{.} 是示性函数，其取值规则为：

$$1\{\text{表达式为真}\} = 1$$
$$1\{\text{表达式为假}\} = 0$$

$y^{(i)} = j$ 标识第i个样本中应用包名在向量a中的标识是j。因此，若第i个样本中应用包名在向量a中的标识是j为真，$1\{y^{(i)} = j\}$=1，否则，$1\{y^{(i)} = j\}$=0。

模型训练采用梯度下降法或者牛顿法最小化$J(\theta)$，求出θ。其中梯度下降法方法如下：

$$\theta_j^{'} = \theta_j - \alpha\nabla_{\theta_j}J(\theta) \quad (10)$$

$$\nabla_{\theta_j}J(\theta) = -\frac{1}{k}\sum_{i=1}^{k}\left[x^{(i)}(1\{y^{(i)}=j\} - \frac{e^{\theta_j x^{(i)}}}{\sum_{l=1}^{M} e^{\theta_l x^{(i)}}})\right] + \lambda\theta_j \quad (11)$$

α是一个固定参数，j 表示迭代次数。

本实施例通过每24小时进行一次训练，不断更新识别模型，使得越来越接近用户的真实使用习惯。

通过以上模型训练求出参数θ之后，就得知了排序函数$h_\theta(s)$，也就是在 中实时排序时会用到的排序函数。将求得的信息按照表3的形式存储到存储器180中，供实时排序时使用。

c）N 值预测算法（S605）

N 的值为正整数，反映的是某一个特定分类内用户经常使用的应用的个数，例如一个特定时间段内（比如工作时间段）用户经常使用的应用的个数。N 值预测算法用于确定 N 的值。N 值可以用于资源管理，因为这 N 个应用是用户经常使用的，所以资源管理的时候可以适当保护这 N 这个应用的资源。

依照前述，本实施例是以时间为维度将用户使用规律做了两个分类，如表 3 所示。那么，对 N 值的计算也可以以时间为维度进行同样的分类。输出结果如表 4 所示：工作时间段和非工作时间段分别计算一个 N 值。这样 N 值的预测更准确。

在其他一些实施例中，也可以所有分类只计算一个 N 值，或计算 N 值的分类方式和表 3 的分类方式不同。在另外一些实施例中，也可以采用非时间维度进行 N 值的计算，比如位置维度，举例来说，与家相关的数据计算一个 N 值，与办公地相关的维度计算一个 N 值。

N 值可以是预先计算存储起来，例如如表 4 存储，也可以资源管理需要 N 值的时候实时调用 N 值预测算法获得 N 值。

表 4

[9:00~18:00] （工作时间段）	机器学习模型 1 的参数信息	N 值
[0:00~9:00] [18:00~24:00] （非工作时间段）	机器学习模型 2 的参数信息	N 值

N 值预测算法的一种实现方式：获取前一天或多天统一时间段的应用的使用记录，统计每个应用的使用次数，然后根据使用次数从大到小排序，依次找到最大的 V 个应用，若

该 V 个应用的使用次数之和能达到该时间段所有应用被使用的次数之和的 90%，则 N=V。

N 值预测算法的另一种实现方式：二阶滑动平均。第 t 天特定时间段（或其他维度的某种分类）上需要保护的应用个数 N_t 是第 t-1、t-2 天同样时间段上需要保护的应用个数的平均值加上方差的加权，如公式（12）所示：

$$N_t = \frac{(N_{t-1} + N_{t-2})}{2} + \alpha * \sigma \qquad (12)$$

α为权重系数，σ是数据中所有 N_i 的方差值。α取值例如为 0.005。第 1 天和第 2 天的 N 值可以使用前一种实现方式计算 N 值。

按照公式（12）每次计算一个 N 值都要计算所有历史 N_i 的方差值σ，这样需要存储下所有历史的 N_i。本实施例中为了减少历史 Ni 的存储量，使用增量式计算法来评估所有 Ni 的方差值：

$$\sigma^2 = \frac{P[\sigma_H^2 + (\overline{X} - \overline{H})^2] + Q[\sigma_A^2 + (\overline{X} - \overline{A})^2]}{P + Q} \qquad (13)$$

其中，$\overline{X}$ 是所有数据的平均值，σ_H^2 是历史数据方差，$\overline{H}$ 是历史数据平均值，P是历史数据的个数，σ_A^2 是新增数据的方差，$\overline{A}$ 是新增数据的平均值，Q是新增数据的个数。

请参考图 8 为第三阶段应用实时排序过程的过程示意图。

实时排序模块 320 设置事件监听机制，检测触发实时排序的关键事件（S801）。监测到关键事件之后（S802），触发数据采集模块 221 采集实时数据（S803），该实时数据可以包括终端设备的系统时间、终端设备的当前状态数据和终端设备的当前位置数据。

关键事件主要包含以下几种：1）前后台切换；2）应用的安装和卸载；3）系统的状态发生变化：即耳机/网络的连接状态发生变化；4）收到情景智能广播通知，即通知语义地理位置(公司/家)发生了变化，等。

根据采集到的终端设备的系统时间和前述维度划分模型的训练结果（表 2），确定终端设备当前所处的时间维度上的分类，即为工作或非工作时间段（S804）。根据排序模型的训练结果(表 3)，选择相关分类的排序模型或仅获得模型参数，然后根据历史数据以及实时采集的数据等，预测出实时场景下所有应用的重要性排序（S805）。这里的历史数据就是历史采集并存储的数据，具体可参考表 1。

应理解的是，待输入排序模型的数据可能需要经过预处理，比如与当期前台应用相关的时序特征数据需要将采集到的实时数据和部分历史数据进行统计后获得，如果模型需要输入语义位置数据，那还需要通过聚类将实时采集到的终端设备位置信息转化为语义位置数据或根据预设的 GPS 位置区间与语义位置的对应关系获得相应的语义位置数据。需不需

要预处理，以及做怎样的预处理根模型训练的时候匹配即可，本申请对此并不做具体限定。

需要说明的是，表 2 和表 3 仅是方便理解的举例，具体实现过程中，二者可以合并为一个表，即一个对应关系，而根据系统当前时间可以一次性确定出排序模型或模型参数。

需要说明的是，本实施例在 S804 分类使用的是实时采集的系统时间来分类，在其他实施例中，如果分类的维度不同，也可以根据需要利用实时采集的其他数据和/或历史数据来分类。

最后实时排序模块 320 将排序的结果存储（S806）起来，供第四阶段资源管理时查询使用。在其他一些实施例中，第三阶段的排序也可以是第四阶段实时调用时才发生，然后实时返回排序结果。

进一步的，图 8 中未示出，排序之后还可以通过 N 值预测算法或通过预先存储的对应关系（如表 4）获得该时间段内的用户经常使用的应用的个数 N。 该步骤不是必须的，可以在资源管理真正需要 N 值时再实时计算。

针对用户近期才安装到手机上的新应用，在利用上述方法对应用排序的时候，可能会因为采集到的数据过少而导致对应用的重要性评估过低。因此本申请进一步还提出一种对新安装应用的补充机制，通过计算一个分值 Score 对新安装应用进行排序。

排序新安装应用主要考虑两方面因素：1）使用可能性权重，本实施例中也可以称之为 LRU 权重：查看当前新安装的应用有没有出现在 LRU (last recently used)列表中，以判断新安装的应用是否在最近被使用过，若出现，则 *LRU* 为 1，不出现 *LRU* 为 0；2）时间衰减权重：计算当前时间距离新安装应用安装时间的时间差衰减权重。LRU 列表中存储了最近使用的多个应用的标识，例如列表长度为 8，则按照时间顺序存储最近使用的 8 个应用，每插入一个新使用的应用，就删除一个最久之前使用的应用。

基于这两种因素，定义新安装应用 Score 如下：

$$Score = \alpha_1 \times LRU + \alpha_2 \times e^{-t} \qquad (14)$$

其中，α_1为 LRU 的权重系数， α_2为时间衰减的权重系数，t 为当前时间距离应用安装时间的时间差的离散值。计算所有新安装应用的 Score，并进行排序。

在其他一些实施例中，也可以采用其它形式来确定新安装的应用是否在最近使用过。例如，记录新安装应用上一次的使用时间，然后判断上一次使用时间距离当前时间的时间差是否小于某个阈值等。

在需要进行 N 个用户经常使用的应用的推荐时，通过前述 softmax 算法的排序结果推荐排序在前的 N-x 个，通过新安装应用的排序结果推荐 x 个，两种推荐方法一共推荐 N 个应用，作为当前系统最重要的应用，以用于接下来的资源管理。

其中，x 个新安装应用可以依次选出的 Score 最大的 x 个应用，而 x 的取值是多少可以根据需要设定，例如为 2。或者设置条件：被推荐的新安装应用的 Score 必须大于某个阈值

（例如 0.5）和/或推荐个数最多为 2。具体如何设置条件，本申请不做限定。

下面介绍第四阶段系统资源管理的实施例，下面介绍的方法可以部分或全部由决策执行模块 223 实现。

系统资源管理的主要目的是保证前台应用或重要性高的应用的资源供给。本实施例提供一种应用进程的临时冻结（也可以称之为瞬时冻结）方法，用以在资源需求量较大时，临时冻结一些不重要的进程，以为重要进程的使用提供更充分的资源，避免因为资源不足造成应用卡顿而影响用户体验。

进程冻结通常指的是将进程置为不可中断的睡眠状态或停止状态，并放入等待队列的过程。“进程”可以理解为正在运行的应用的实例，在有些描述中，二者可以被等同理解。

在传统的台式计算机操作系统内，冻结技术主要用来在系统休眠（hibernate）或挂起（suspend）的时候，将进程列表中所有进程的状态都置于睡眠或停止状态，并保存所有进程的上下文到硬盘上，这样在从休眠或挂起恢复时，系统就可以通过解冻所有被冻结的进程，把自己恢复到之前的运行状态。

针对终端操作系统，不同的终端操作系统对冻结技术的使用不太相同。部分终端操作系统采用伪后台技术，即在应用进程切换到后台且在后台运行一段时间后，冻结该进程，直到用户重新将该应用切换到前台，才解冻该进程继续运行。Android 操作系统在内存不足时，OOM（out of memory）或 LMK（low memory killer）模块会触发内存回收通过杀进程的方式来回收内存，但这样导致被杀应用重新打开的时间变长，同时用户原先的状态也无法保存，降低了用户体验，为解决该问题引入冻结技术。具体的，系统在内存不足时（例如内存剩余量低于某个阈值）先冻结进程，把该进程的上下文交换到 ROM 中，并释放内存，这样在下次重新打开该应用时，通过解冻该进程来继续运行，既避免了应用被杀丢失原先的状态，也降低了应用重启的时间，提高了用户体验。

总之，现有的冻结技术主要用在对长期不用的进程的处理，或是对资源的长久释放，直到有用户对该进程有需求才解冻。

本实施例提出一种瞬时冻结技术，通过对其他应用进程的临时冻结，使用临时让路的方式来保证对重要应用或重要场景所需资源的及时给足，并在资源需求降低后，重新自动恢复解冻这些被冻结的应用进程，在提高该应用的处理和响应速度的同时，又避免了另一些应用被长期冻结后，导致的体验下降（如下载被长期冻结后，就无法继续下载了），提高了用户体验。

当某些对资源瞬时需求量较大的特定事件（例如应用的启动、activity 切换、拍照、滑动、亮/灭屏、图库缩放等）被触发时，为了避免非重要应用的执行对这些特定事件造成影响，强制要求临时冻结非重要应用，并在关键事件完成后，根据应用的重要性排序，恢复部分被临时冻结的应用的运行。

这里的“特定事件”一般都与重要应用相关，重要应用比如前台应用或后台可感知应用，后台可感知应用例如为在后台运行的音乐播放应用。被临时冻结的非重要应用可以根据需要选择，例如，根据前述实施例提供的应用的重要性排序结果选择除前 N 个应用之外的其他所有应用进行临时冻结（这里的 N 为前述实施例中计算出的用户经常使用的应用的

个数），或选择所有后台应用进行临时冻结，或者根据需要选择其他当前判定非重要或预测一段时间内非重要的应用。

如图 9 所示，检测函数监测特定事件（S901），通过活动管理器服务（Activity Manager Service, AMS）的打点上报函数上报特定事件(S902)，临时冻结部分应用（S903）。在调用临时冻结的函数时设置冻结时间，例如 1.5s，这个冻结时间可以是预先设置的，不能改变，也可以是用户可配置的。冻结时间通过定时器实现，检测到定时器到期（S904），然后解冻应用（S907）。

另一方面，也可以监听特定事件的执行，当检测到特定事件结束（S905）之后，解冻应用。这里的特定事件结束的检测也可以通过活动管理器服务的打点上报函数上报的方式来实现。步骤 S904 和 S905 可以理解为两种正常解冻的条件，实现其中任意一种或两种都可以。

解冻的时候可以解冻全部应用，也可以仅解冻部分应用。选择需要被解冻的应用可以根据应用的重要性排序选择重要性相对较高的部分应用进行解冻，其他的因为重要性很低，可能很长一段时间还是不会被用户使用，所以可以继续冻结。

在特定事件完成之前，并且指定的定时时长未到，但发生了指示被冻结应用运行环境改变的事件(S906)，例如被冻结的应用切换成了前台、被冻结的应用退出、被冻结的应用收到 binder 异步消息、被冻结的应用的通知消息被用户点击等，则该被冻结的应用可以提前解冻(S907)。这里“指示被冻结应用运行环境改变的事件”一般与某一被冻结应用有关，该事件的发生通常会使所述某一被冻结应用的重要性突然提高，需要及时被解冻。同样的，对指示被冻结应用运行环境改变的事件的监测也可以由前述的活动管理器服务的打点上报函数来实现。指示被冻结应用运行环境改变的事件可能为以下事件中的任意一项：1）应用置前台；2）应用退出；3）应用重安装或者更新版本；4）动态壁纸应用被设置为动态壁纸；5）应用 widget 添加到桌面；6）即时通讯（instance message, IM）/短消息服务(short message service, SMS)/邮件（Email）类应用有网络数据包到达；7）IM/SMS/Email 类应用从无网络到网络连接上；8）其它应用访问冻结应用的提供者（provider）或者服务（service）时；9）系统或其它应用通过 binder 同步调用冻结进程； 10）解锁后，检测到在桌面上有 widget 的应用； 11）运动开始，之前冻结的使用 GPS 和 Sensor 的应用；12）冻结应用处理耳机按键；13）被临时冻结的应用收到 binder 异步消息；14）点击通知栏进入冻结应用；15) 进入灭屏状态等。

需要说明的是，以上 1）-15）事件有的涉及到具体应用，那就意味着需要将该应用解冻（如果该应用在冻结状态的话），有的不涉及具体应用，那就选取部分或全部被冻结的应用解冻，例如进入 15）检测到系统进入灭屏状态，意味着可能没有用户可见的前台应用需要保护了，因此可以解冻所有被冻结的应用，以便于它们尽快继续运行。

可见，除了两种正常解冻的方式外，本实施例还提出一种紧急或提前解冻的方式。在特定事件完成之前，并且指定的定时器时长未到，但发生了被冻结应用运行环境改变的事件，则该被冻结的应用可以被提前解冻，以避免因为冻结影响该应用的正常使用。

用户连续操作（如滑动、或连续启动应用等）将会连续触发临时冻结。为了保证一些后台应用在连续瞬时冻结的操作下有机会运行，本实施例还提出了一种周期性冻结的方法。

在对应用进行冻结操作之前，会先检测所述应用被解冻后的运行时间。对于解冻后累计运行时长已达 t1 秒的应用（t1 是个预先设置的时间长度值，例如 10s），可以连续实施临时冻结。而对于解冻后累计运行时长不足 t1 的应用可以周期实施临时冻结，对这种应用先冻结 t2 秒然后再解冻 t3 秒，直至满足连续冻结条件。这里 t2，t3 都是预先设置时间长度值(t2<t1，t3<t1)，例如:t2=1.5s，t3=3s。通过这样一种周期性解冻的方法，能够很好的保证一些后台应用在前台长期交互操作时能够得到一些运行的机会，不会出现后台一些重要应用长期得不到运行的情况，甚至出现某些应用被连续冻结后导致异常的发生(例如下载被长期冻结后，就无法继续下载了)。在检测到指示所述被冻结应用运行环境改变的事件后，无论所述应用有无被解冻，该应用的解冻后累计运行时长都会被清 0，这样就可以保证各后台应用在接收到所述运行环境改变的事件后都能够运行一段时间，在优先保障前台应用的瞬时高资源需求外，也保证了后台应用的运行和体验。

本申请另一个实施例提供一种重要任务的调度方法，这里的任务可以是进程，也可以是线程。

理想状态下，每个任务都能从 cpu 那里获得相同的时间片，并且同时运行在该 cpu 上，但实际上一个 cpu 同一时刻只能运行一个任务。也就是说，当一个任务占用 cpu 时，其他任务就必须等待。完全公平调度器（completely fair scheduler, CFS）调度算法是现有的 Linux 系统中的一种较为通用的调度算法。CFS 调度算法为了实现公平，必须惩罚当前正在运行的任务，以使那些正在等待的进程下次被调度。具体实现时，CFS 通过每个任务的虚拟运行时间（virtual run time, vruntime）来衡量哪个任务最值得被调度。CFS 中的就绪队列是一棵以 vruntime 为键值的红黑树，vruntime 越小的进程越靠近整个红黑树的最左端。因此，调度器每次选择位于红黑树最左端的那个任务，该任务的 vruntime 最小。

需要说明的是，本实施例中提到的“cpu”指的是计算机设备中的最小处理单元，也可以称之为处理核，或简称为核。

如图 10 所示，针对该二叉树以及二叉树上的任务 T1-任务 T6，cpu 执行任务的顺序为 T1-T3-T5-T2-T4-T6。假设任务 T1 为其中的重要任务。

vruntime 是通过任务的实际运行时间和任务的权重（weight）计算出来的，是一个累加值。在 CFS 调度器中，将任务优先级这个概念弱化，而是强调任务的权重。一个任务的权重越大，则说明这个任务更需要运行，因此它的虚拟运行时间就越小，这样被调度的机会就越大。

由于 CFS 调度算法的公平性，即便把重要任务的权重设置的很大，也会出现重要任务被执行完一次之后，等待至少一个其他任务运行的情况。任务 T1 执行一次之后，任务 T1 的 vruntime 值变大，CFS 调度任务 T3 之后任务 T1 因为 vruntime 值变大所以被插入到队列末尾，如图 11 所示。那么任务 T1 则要等到任务 T3-T5-T2-T4-T6 全部被执行一遍后才能再次被执行。图 11 为一个示例，任务 T1 被执行一次之后，也可能重新入队到其他位置，但是无论怎样，CFS 为了实现公平，总会让任务 T1 出现等待的情况。因为任务 T1 是对用户而言的重要任务，那么这样的等待将有可能造成系统卡顿。

本实施例提出一种重要任务的调度方法，除前述就绪队列之外，为每个 cpu 都再新建一

个运行队列（下称 vip 队列），其结构和前述就绪队列类似。将重要任务放到该 vip 队列中。

重要任务指的是对用户体验影响较大的任务，可以包括前述实施例中提到的重要性排序在前的 N 个应用的所有线程（或进程）；或者重要任务包括这 N 个应用的线程中的关键线程，例如 UI 线程和渲染线程；或者重要任务包括当前前台应用的 UI 线程和渲染线程等关键线程。

重要任务又分为静态和动态两种，静态重要任务的识别一般发生用户态，例如前台应用的用户界面（user interface，UI）线程和渲染（render）线程。静态重要任务一般在应用重要性改变的时候才取消其重要性。动态重要任务为静态重要任务所依赖的重要任务，其识别一般发生在内核态，一旦依赖解除，其重要性就取消。动态重要任务包括静态重要任务直接依赖的任务，还可以包括间接依赖的任务。

这里的依赖可以为数据依赖、锁依赖、binder 服务依赖或控制流依赖等。数据依赖即 B 任务的执行必须依赖于 A 任务的输出；锁依赖即 B 任务的执行需要 A 任务释放的某种锁；控制流依赖为 B 任务在执行逻辑上必须等待 A 任务执行完才能执行；binder 服务依赖指的是任务 A 调用一个 binder 函数（类似远程过程调用），要求任务 B 完成某项功能，并返回运行结果，这样任务 A 对任务 B 产生了 binder 服务依赖，属于控制流依赖的一种具体实例。

以 linux 系统为例，介绍一种依赖检测关系的具体实现过程。在任务控制块 task_struct 中添加两个字段：static_vip 和 dynamic_vip，分别用来表示静态和动态重要任务标志值。static_vip=1，表示该任务为静态重要任务；dynamic_vip 不等于 0，表示该任务为动态重要任务。对于动态重要任务，由于一个任务可能会被多个其他任务同时依赖，并且依赖的原因可以相同或不同，如 mutex 互斥锁、rwsem 读写信号量、binder 服务依赖等，所以我们对 dynamic_vip 这个字段进行划分，分成 3 个或更多类型，分别表示 mutex，rwsem 和 binder 依赖，更多类型不在详述，也可以预留几个类型以后扩展。每种类型使用 8 位存储。这样在每次调用对应的依赖函数时，该字段对应的区块值加 1，在每次依赖函数完成后，该值相应的减 1，直到所有字段都等于 0，再取消该任务的重要属性，重新放入就绪队列中运行。

下面以 mutex 锁和 binder 依赖为例，详细说明其步骤：

重要任务 A 中调用 mutex_lock 函数获取一个互斥锁，在该函数中会检测锁的状态，如果获取失败，则表明该锁已被其他任务获取了，当前任务 A 被挂起，进入睡眠状态，我们在其中添加了代码，用来获取当前该锁的持有者（任务 B），并且对该任务结构的字段 dynamic_vip 的值对应位加 1，然后再把该任务 B 移到 vip 队列中进行调度运行。当任务 B 释放该锁 mutex_unlock 时，判断其 task_struct 中的 dynamic_vip 的相应值，并减 1，如果为 0，则把该任务从 vip 队列中移除。其他锁的运行过程类似。

重要任务 A 调用普通任务 B 的 binder 函数时，在内核的 binder 驱动中的 binder_thread_write 函数中，会先找到任务 B 的任务结构 task_struct，并设置其 dynamic_vip 的值，然后把相应的函数 id 和参数传递给任务 B，并唤醒任务 B 开始运行，然后重要任务 A 等待其返回运行结果；当任务 B 通过 binder_thread_read 函数完成相应的功能后，在返回运行结果到任务 A 前，检查 dynamic_vip 的值，如果为 0，则把该任务从 vip 队列中移除。

按照上述方式识别出多个重要任务后，将这些任务插入 vip 队列。插入原则可以和前述就绪队列一样，即根据每个任务的 vruntime 的值插入。如图 12 示例，vip 队列中包括两个

重要任务 T1 和 T7，当前 T7 的 vruntime 小于 T1 的 vruntime.

每次获取下一个需运行的任务时，cpu 先检查 vip 队列中是否有任务需要运行，如有则选取该队列中的任务运行，如该队列为空，才选取就绪队列中的任务运行。这样就保证了重要任务都能在其他非重要任务之前运行，为重要任务实现了类似插队的功能。

进一步的，当 vip 队列中的任务数较多时，为了避免 vip 队列中的任务排队等待，可以适时检查当前各 cpu 的 vip 队列中的任务是否有延迟。若存在延迟的任务，则确定该任务是否可移动，若可移动则移动该任务到另一个空闲的 cpu 的 vip 队列中。通过这样的方式实现了重要任务的迁移（或者称搬家），保证了重要任务的及时运行，避免了卡顿，进一步提高了用户体验。

具体的，参考图 13，内核在时钟中断到来（S1301）时检查当前 cpu 对应的 vip 队列中是否有任务等待时长超过阈值（例如 10ms），即判断是否有延迟任务（S1302）。 如果有任务的等待时长超过该阈值，再检查该任务的数据和/或指令是否还存在于缓存（cache）中，即判断该任务是否可移动(S1303)。若该任务的数据和/或指令全部没有存在于或部分没有存在于缓存中（cache 是否 hot），则在当前 cpu 所在的 cpu 簇（cluster）中选择一个 vip 队列中没有任务等待，也没有实时任务的目的 cpu(S1304)，将该任务迁移到该目的 cpu(S1305)。任务的迁移可以通过调用内核提供的函数 migration 就可以实现。检测 cache 是否 hot 可以通过调用内核提供的函数 task_hot 实现。

具体实现的时候，可以一次性识别出所有可迁移的任务，然后执行迁移；也可以依次处理 vip 队列中的任务。

一个任务的等待时长为该任务当前时间与入队时间的时间差，前提是该任务这中间从未被运行过；或者一个任务的等待时长为当前时间与上一次被运行的时间的时间差。

检测一个队列中的任务的等待时间是否超过阈值通常需要先获取队列的锁，在时钟中断到来时实现上述方法，可一定程度上避免死锁。

移动一个任务时可以将该任务移动到处理效率比原 cpu 高的 cpu 上。通过这样的方式可以进一步提高重要任务的处理效率。例如，当前终端设备 8 个核（cpu0-cpu7）一般分为两个等级，4 个小核和 4 个大核，那么迁移的时候如果原始 cpu 为小核，那可以选择一个大核作为目的 cpu，将重要任务迁移过去，参考图 14 所示。

通过为每个 cpu 创建一个 vip 队列，并将重要任务放到该队列中，使得重要任务优先于就绪队列中的其他任务执行，可以保证重要任务的执行效率，由于重要任务和系统的卡顿体验相关，从而可以一定程度上避免出现用户可感知的系统卡顿，提升终端设备的用户体验。

需要说明的是，前述实施例中提出模块或模块的划分仅作为一种示例性的示出，所描述的各个模块的功能仅是举例说明，本申请并不以此为限。程序设计人员可以根据需求合并其中两个或更多模块的功能，或者将一个模块的功能拆分从而获得更多更细粒度的模块，以及其他变形方式。

以上描述的各个实施例之间相同或相似的部分可相互参考。

以上所描述的装置实施例仅仅是示意性的，其中所述作为分离部件说明的模块可以是或者也可以不是物理上分开的，作为模块显示的部件可以是或者也可以不是物理模块，即

可以位于一个地方，或者也可以分布到多个网络模块上。可以根据实际的需要选择其中的部分或者全部模块来实现本实施例方案的目的。另外，本申请提供的装置实施例附图中，模块之间的连接关系表示它们之间具有通信连接，具体可以实现为一条或多条通信总线或信号线。本领域普通技术人员在不付出创造性劳动的情况下，即可以理解并实施。

以上所述，仅为本申请的一些具体实施方式，但本申请的保护范围并不局限于此。

权 利 要 求

1.一种在计算机系统中管理资源的方法，其特征在于，包括：

获取数据，所述数据包括与当前的前台应用相关的应用时序特征数据，所述数据还包括以下实时数据中的至少一种：所述计算机系统的系统时间、所述计算机系统的当前状态数据和所述计算机系统的当前位置数据；

根据所述实时数据中的至少一种从多个机器学习模型中选择与所述实时数据匹配的目标机器学习模型；

将获取的所述数据输入所述目标机器学习模型以对所述计算机系统上安装的多个应用进行重要性排序；

根据所述重要性排序的结果执行资源管理。

2. 如权利要求 1 所述的方法，其特征在于，获取的所述数据包括所述计算机系统的系统时间；

所述选择与所述实时数据匹配的目标机器学习模型包括：

根据所述计算机系统的系统时间确定所述计算机系统当前所处的时间段；

根据所述计算机系统当前所处的时间段从对应关系中确定与所述计算机系统当前所处的时间段对应的目标机器学习模型，所述对应关系包括多个时间段以及该多个时间段分别对应的多个机器学习模型。

3. 如权利要求 1 所述的方法，其特征在于，获取的所述数据包括所述计算机系统的当前位置数据；

所述选择与所述实时数据匹配的目标机器学习模型包括：

根据所述计算机系统的当前位置数据确定所述计算机系统当前所处的语义位置；

根据所述计算机系统当前所处的语义位置从对应关系中确定与所述计算机系统当前所处的语义位置对应的目标机器学习模型，所述对应关系包括多个语义位置以及该多个语义位置分别对应的多个机器学习模型。

4. 如权利要求 1-3 任意一项所述的方法，其特征在于，还包括：

采集并存储应用数据和所述计算机系统的相关数据，其中，所述应用数据包括所述应用的标识，所述应用被使用的时间，所述计算机系统的相关数据包括以下数据中的至少一种：在所述应用被使用的时间所述计算机系统的时间、状态数据和位置数据。

5. 如权利要求 4 所述的方法，其特征在于，还包括：

根据过去一段时间内采集并存储的所述应用数据计算多个应用的应用时序特征数据；

将所述应用数据，或所述应用数据和所述计算机系统的相关数据输入分类模型以获得与应用被使用的规律相关的多个分类，所述多个分类为一维分类或多维分类，其中任意两个分类分别对应的两种规律存在不同；

分别针对所述多个分类中的每个分类训练机器学习模型，所述机器学习模型用于实现应用的重要性排序，所述训练的输入包括所述应用被使用的时间、所述应用时序特征数据，

所述训练的输入还包括所述计算机系统的相关数据中的至少一种。

6. 如权利要求 4 或 5 所述的方法,其特征在于,在监测到以下事件中的一种或多种时,启动所述采集步骤:前后台切换事件、应用的安装事件、应用的卸载事件、所述计算机系统的状态数据发生变化引起的通知事件或所述计算机系统的位置数据发生变化引起的通知事件。

7. 如权利要求 5 或 6 所述的方法,其特征在于,所述多个分类为从时间维度上划分的多个分类。

8. 如权利要求 7 所述的方法,其特征在于,所述多个分类包括工作时间段和非工作时间段。

9. 如权利要求 1-8 任意一项所述的方法,其特征在于,包括:

在监测到以下事件中的一种或多种时,启动所述获取数据的步骤:前后台切换事件、应用的安装事件、应用的卸载事件、所述计算机系统的状态数据发生变化引起的通知事件或所述计算机系统的位置数据发生变化引起的通知事件。

10. 如权利要求 1-9 任意一项所述的方法,其特征在于,在根据所述重要性排序的结果执行资源管理之前,还包括:

获取需要保护的应用的个数 N,所述 N 值满足如下条件:在过去被使用的次数最多的 N 个应用的使用次数占所有应用在该段时间内被使用次数之和的比例大于预设的第一阈值,其中所述 N 为大于 0 且小于或等于 M 的整数;

所述根据所述重要性排序的结果执行资源管理包括:根据所述 N 和所述重要性排序的结果执行资源管理。

11. 如权利要求 10 所述的方法,其特征在于,所述根据所述 N 和所述重要性排序的结果执行资源管理包括:

从所述重要性排序的结果中识别排序在前的 N1 个应用,并对所述 N1 个应用或剩余的其他应用执行资源管理,其中 N1 为小于或等于 N 的正整数。

12. 如权利要求 10 所述的方法,其特征在于,还包括:

根据新安装应用的权重对所述新安装应用进行重要性排序,选取其中排序在前的 N2 个新安装应用,其中所述新安装应用安装到所述计算机系统上的时间小于预设的第二阈值;

相应的,所述根据所述 N 和所述重要性排序的结果执行资源管理包括:

从所述重要性排序的结果中识别排序在前的 N-N2 个应用,对所述 N-N2 个应用和所述 N2 个新安装的应用执行资源管理,或对剩余的其他应用执行资源管理。

13. 如权利要求 12 所述的方法,其特征在于,所述根据新安装应用的权重对所述新安装应用进行重要性排序包括:

根据使用可能性权重和时间衰减权重计算每个新安装应用的得分,得分高的新安装应用的重要性高于得分低的新安装应用的重要性;

其中,所述使用可能性权重用于反映新安装应用最近有没有被使用;所述时间衰减权重用于反应当前时间距离应用安装时间的时间差。

14. 如权利要求 1-13 任意一项所述的方法，其特征在于，所述根据所述重要性排序的结果执行资源管理包括以下管理措施中的任意一种或多种：

为识别出来的重要性高的应用的预留资源；

临时冻结识别出来的重要性低的应用，直至特定时间结束；和

为每个 CPU 创建一个 vip 队列，所述 vip 队列中包括重要性高的应用的任务，所述 vip 队列中各个任务的执行优先于所述 CPU 的其他执行队列。

15. 如权利要求 1-14 任意一项所述的方法，其特征在于，所述应用时序特征数据包括最近被使用的 k1 个应用，所述前台应用的 k2 个最大可能的前续应用以及 k3 个最大可能的后续应用，其中，k1、k2 和 k3 均为正整数。

16. 如权利要求 1-15 任意一项所述的方法，其特征在于，所述当前位置数据为语义位置数据。

17. 如权利要求 1-16 任意一项所述的方法，其特征在于，所述当前状态数据为以下数据中的一种或多种：表示网络连接或网络断开的数据，表示耳机连接或断开的数据，表示充电线连接或断开的数据，以及表示蓝牙连接或断开的数据。

18.一种终端设备，其特征在于，所述终端设备包括处理器和存储器，所述存储器用于存储计算机可读指令，所述处理器用于读取所述存储器中存储的所述计算机可读指令执行如权利要求 1-17 任意一项所述的方法。

19.一种临时冻结应用的方法，其特征在于，包括：

在检测到特定事件时，临时冻结部分应用；

当特定时间段结束或监测到所述特定事件结束时，解冻被冻结的全部或部分应用，或者，当检测到一个或多个被冻结的应用的运行环境发生改变时，解冻所述一个或多个被冻结的应用。

20.如权利要求 19 所述的方法，其特征在于，所述特定事件为指示资源需求量升高的事件。

21. 如权利要求 19 或 20 所述的方法，其特征在于，所述特定事件包括以下事件中的至少一种：应用的启动事件、拍照事件、图库缩放事件、滑动事件以及亮/灭屏事件。

22. 如权利要求 19-21 任意一项所述的方法，其特征在于，所述临时冻结为通过设置定时器实现临时冻结，所述定时器的时长被设置为所述特定时间段。

23. 如权利要求 19-22 任意一项所述的方法，其特征在于，所述临时冻结的部分应用包括所有后台应用；或者所述临时冻结的部分应用包括重要性低的应用，其中应用的重要性根据应用的历史使用情况、机器学习算法以及系统的当前场景数据获得。

24. 如权利要求 19-23 任意一项所述的方法，其特征在于，所述检测到一个或多个被冻结的应用的运行环境发生改变包括检测到如下事件中的任意一种或多种：被冻结的应用切换到前台、被冻结的应用退出、被冻结的应用收到异步消息以及被冻结的应用的通知消息

被用户点击。

25.一种终端设备，其特征在于，所述终端设备包括处理器和存储器，所述存储器用于存储计算机可读指令，所述处理器用于读取所述存储器中存储的计算机可读指令执行如权利要求 19-24 任意一项所述的方法。

26.一种计算机系统中执行任务的方法，其特征在于，所述方法应用于计算机系统，所述计算机系统包括多个物理核，每个物理核对应第一队列和第二队列，所述第一队列和第二队列中分别包括一个或多个待所述物理核执行的任务，至少一个物理核执行如下方法：

获取并执行所述第一队列中的任务，直至所述第一队列中所有的任务都执行完毕，再获取并执行所述第二队列中的任务。

27. 如权利要求 26 所述的方法，其特征在于，还包括：

监测所述第一队列中是否存在等待时间超过特定阈值的任务，若存在，将所述任务移动到另一个物理核对应的第一队列中。

28. 如权利要求 26 或 27 所述的方法，其特征在于，所述将所述任务移动到另一个物理核对应的第一队列中包括：在确定所述任务可移动后再将所述任务移动到所述另一个物理核对应的第一队列中。

29. 如权利要求 26-28 任意一项所述的方法，其特征在于，所述第一队列中的任务包括重要任务以及所述重要任务依赖的任务。

30. 如权利要求 29 所述的方法，其特征在于，所述重要任务为对影响用户体验的任务。

31. 如权利要求 29 或 30 所述的方法，其特征在于，所述重要任务为重要性高的应用的任务，其中应用的重要性根据应用的历史使用情况、机器学习算法以及系统的当前场景数据获得。

32. 如权利要求 29-31 任意一项所述的方法，其特征在于，所述依赖包括以下依赖关系中的至少一种：数据依赖、锁依赖以及 binder 服务依赖。

33.一种终端设备，其特征在于，所述终端设备包括处理器和存储器，所述存储器用于存储计算机可读指令，所述处理器用于读取所述存储器中存储的计算机可读指令执行如权利要求 26-32 任意一项所述的方法。

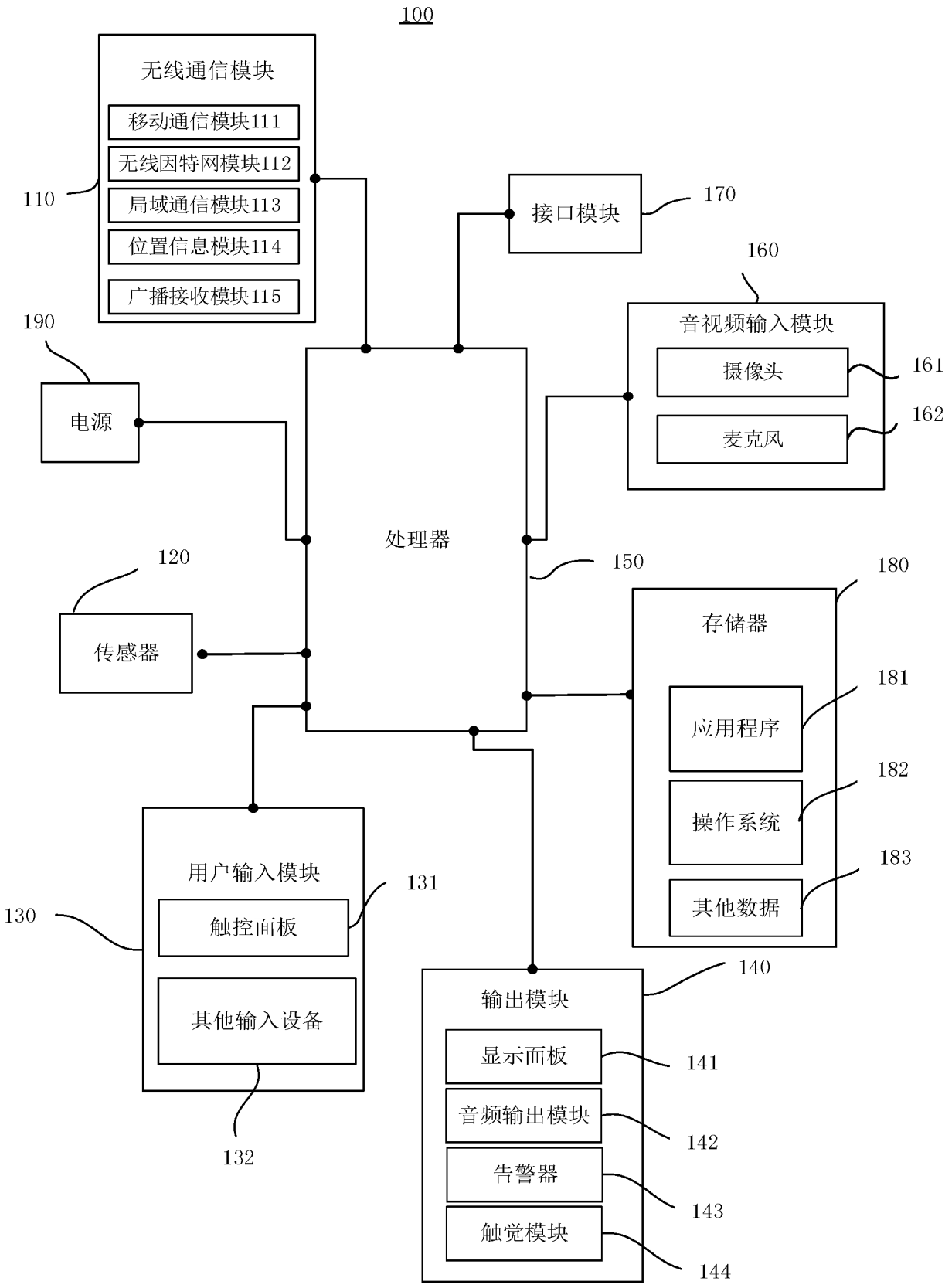
100
无线通信模块
移动通信模块111
无线因特网模块112
局域通信模块113
位置信息模块114
广播接收模块115
110
接口模块
170
160
音视频输入模块
摄像头
161
麦克风
162
190
电源
处理器
150
120
传感器
180
存储器
181
应用程序
182
操作系统
183
其他数据
用户输入模块
131
130
触控面板
其他输入设备
132
140
输出模块
显示面板
141
音频输出模块
142
告警器
143
触觉模块
144

图 1

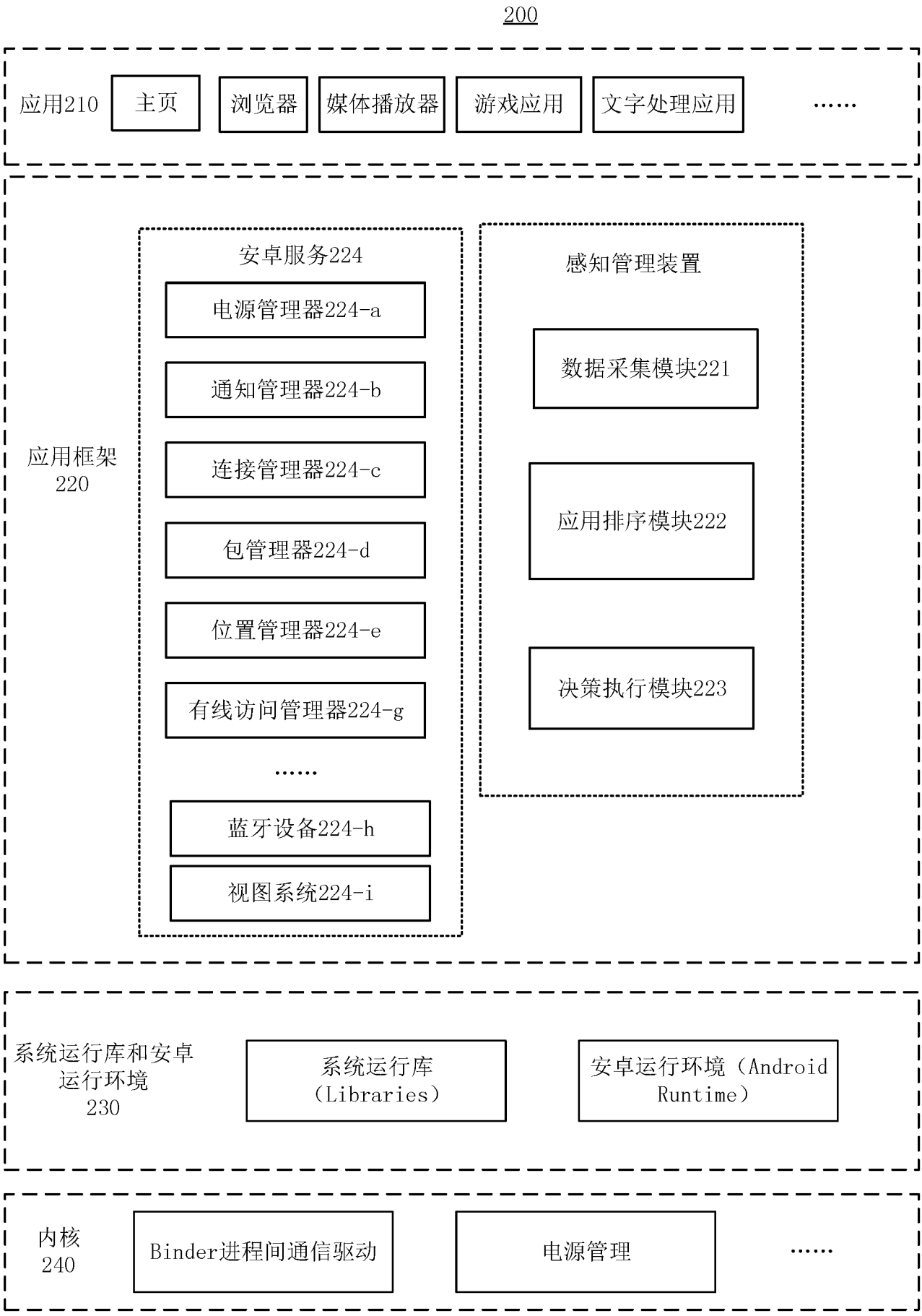
200
应用210
主页
浏览器
媒体播放器
游戏应用
文字处理应用
……
应用框架
220
安卓服务224
电源管理器224-a
通知管理器224-b
连接管理器224-c
包管理器224-d
位置管理器224-e
有线访问管理器224-g
……
蓝牙设备224-h
视图系统224-i
感知管理装置
数据采集模块221
应用排序模块222
决策执行模块223
系统运行库和安卓运行环境
230
系统运行库（Libraries）
安卓运行环境（Android Runtime）
内核
240
Binder进程间通信驱动
电源管理
……

图 2

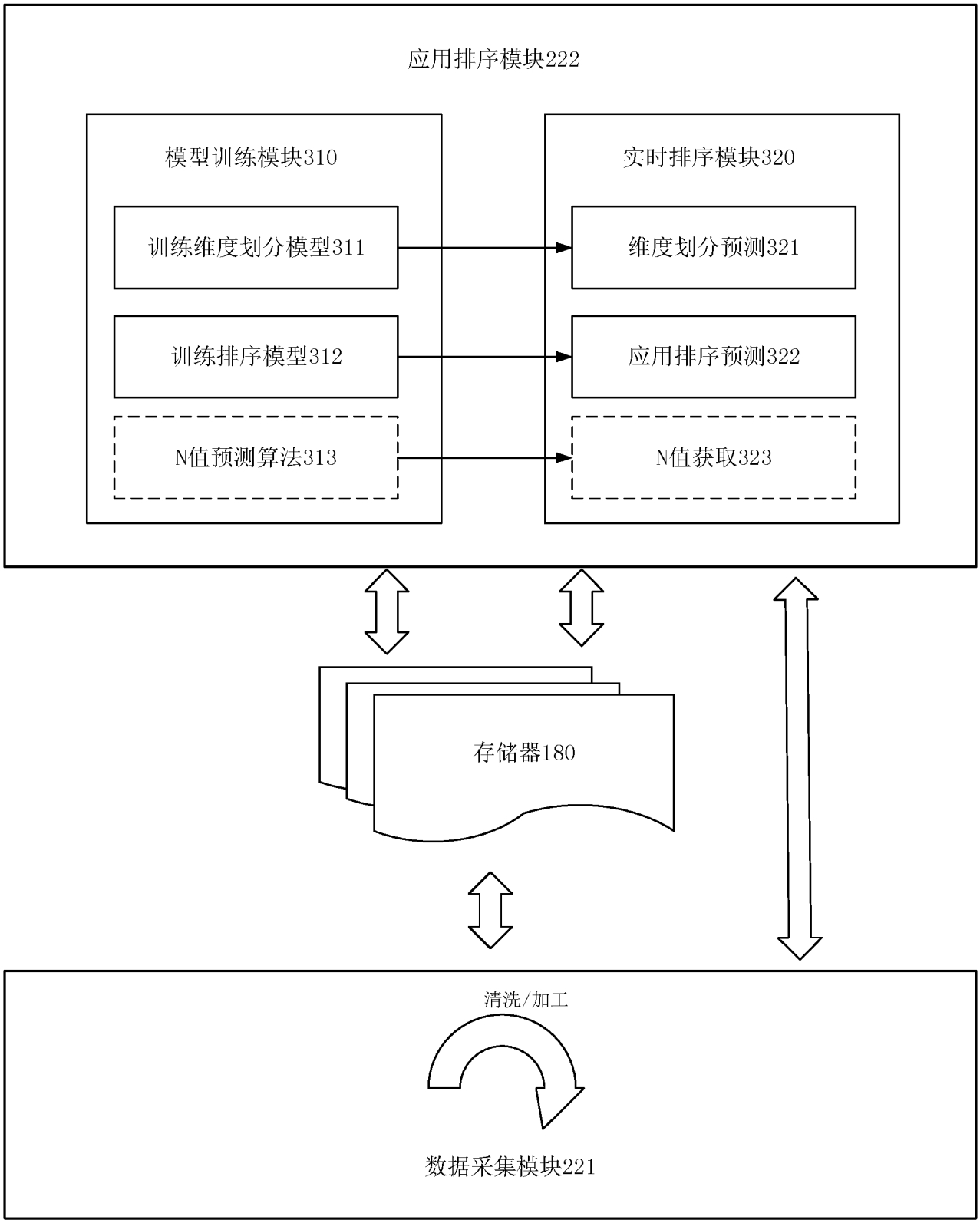
应用排序模块222
模型训练模块310
训练维度划分模型311
训练排序模型312
N值预测算法313
实时排序模块320
维度划分预测321
应用排序预测322
N值获取323
存储器180
清洗/加工
数据采集模块221

图 3

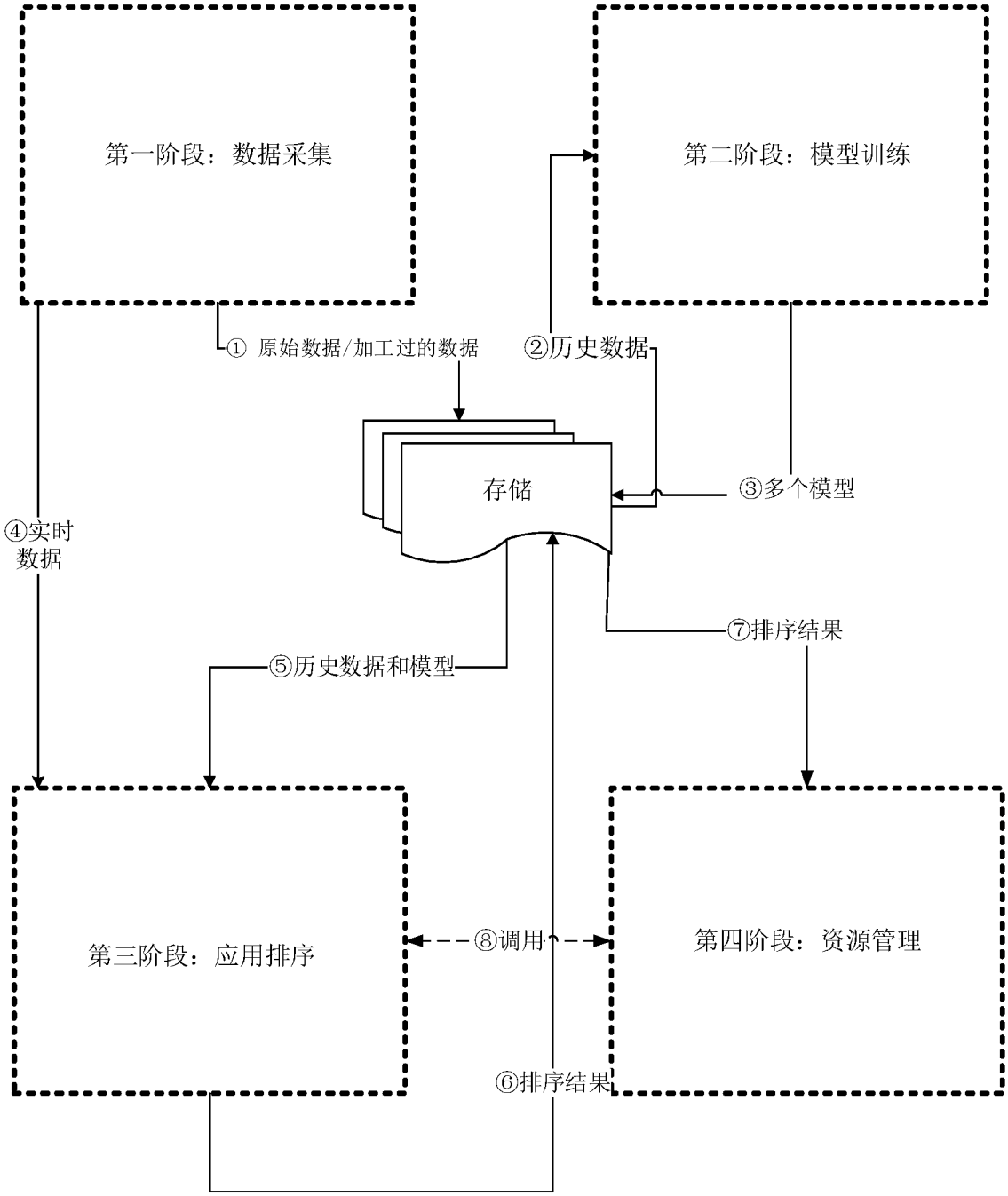
第一阶段：数据采集
第二阶段：模型训练
① 原始数据/加工过的数据
②历史数据
存储
③多个模型
④实时数据
⑤历史数据和模型
⑦排序结果
第三阶段：应用排序
⑧调用
第四阶段：资源管理
⑥排序结果

图 4

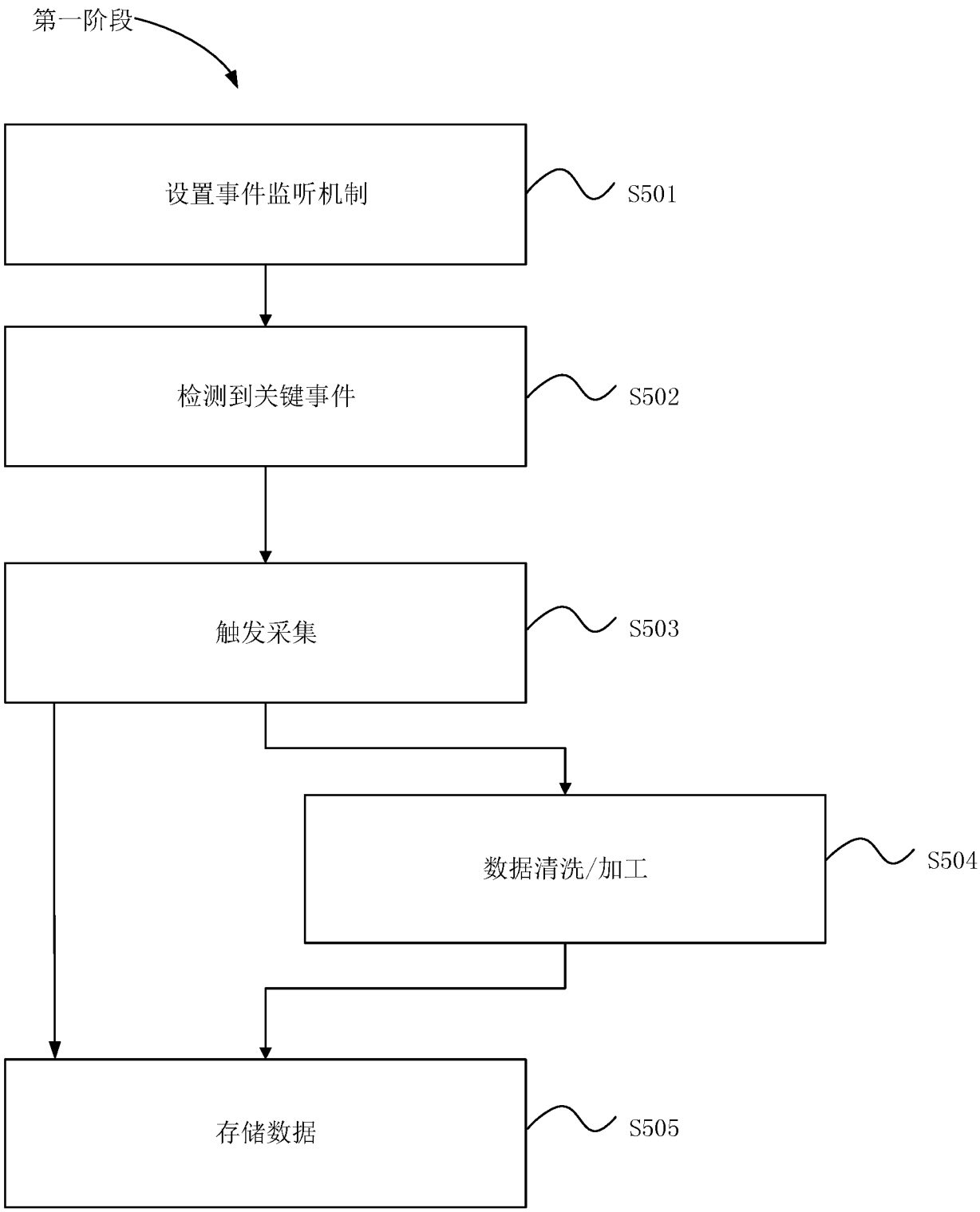
第一阶段
设置事件监听机制
S501
检测到关键事件
S502
触发采集
S503
数据清洗/加工
S504
存储数据
S505

图 5

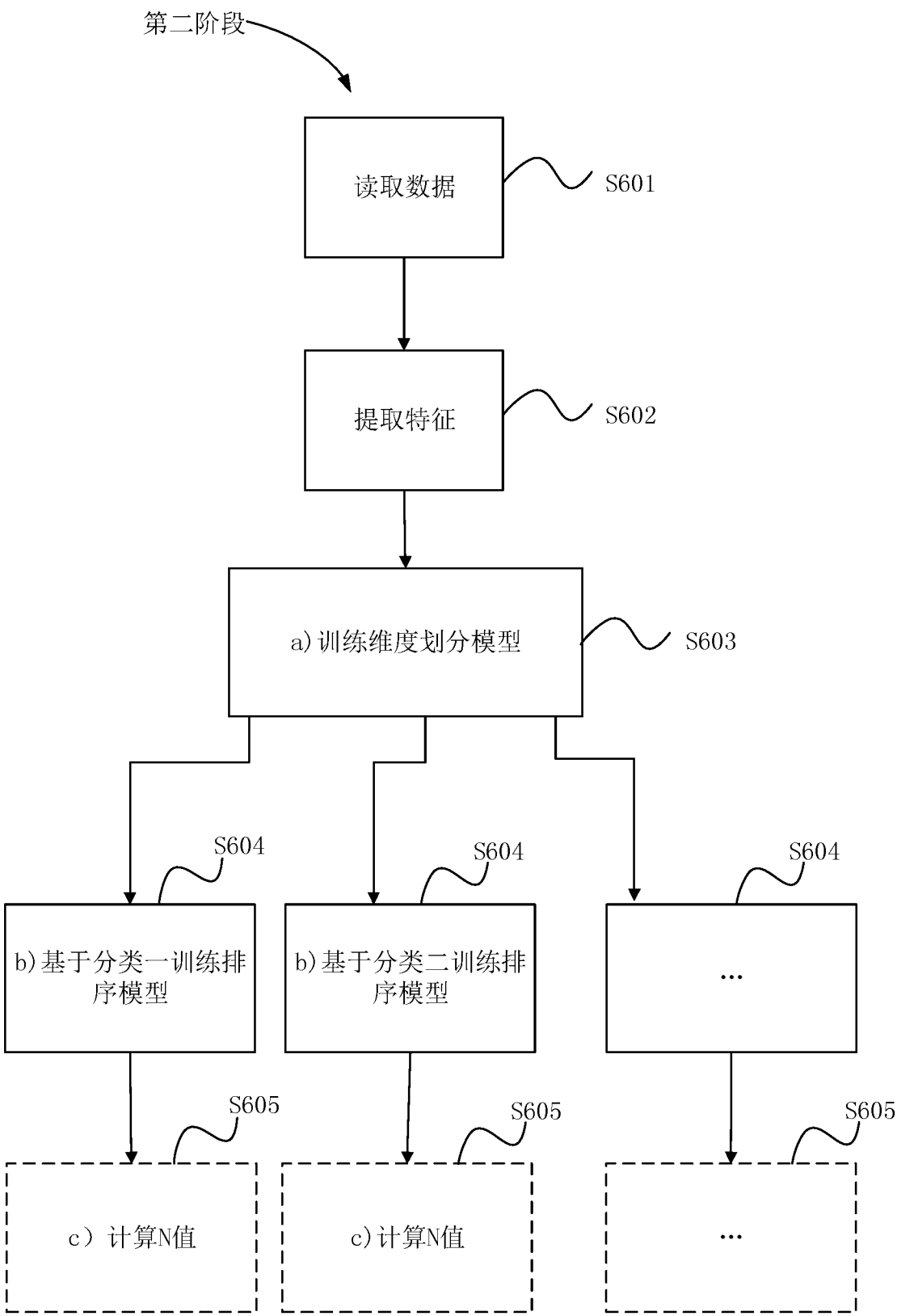
第二阶段
读取数据
S601
提取特征
S602
a) 训练维度划分模型
S603
S604
S604
S604
b) 基于分类一训练排序模型
b) 基于分类二训练排序模型
...
S605
S605
S605
c) 计算N值
c) 计算N值
...

图 6

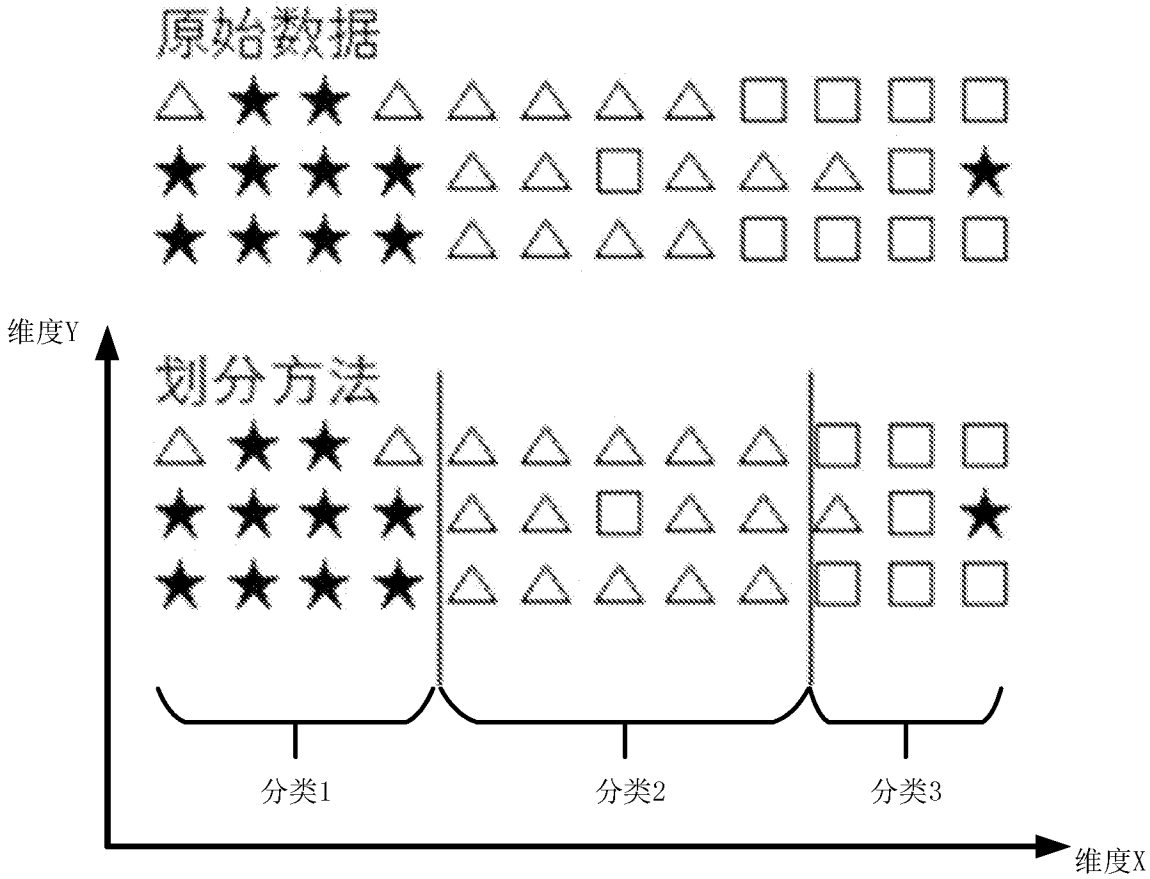

图 7

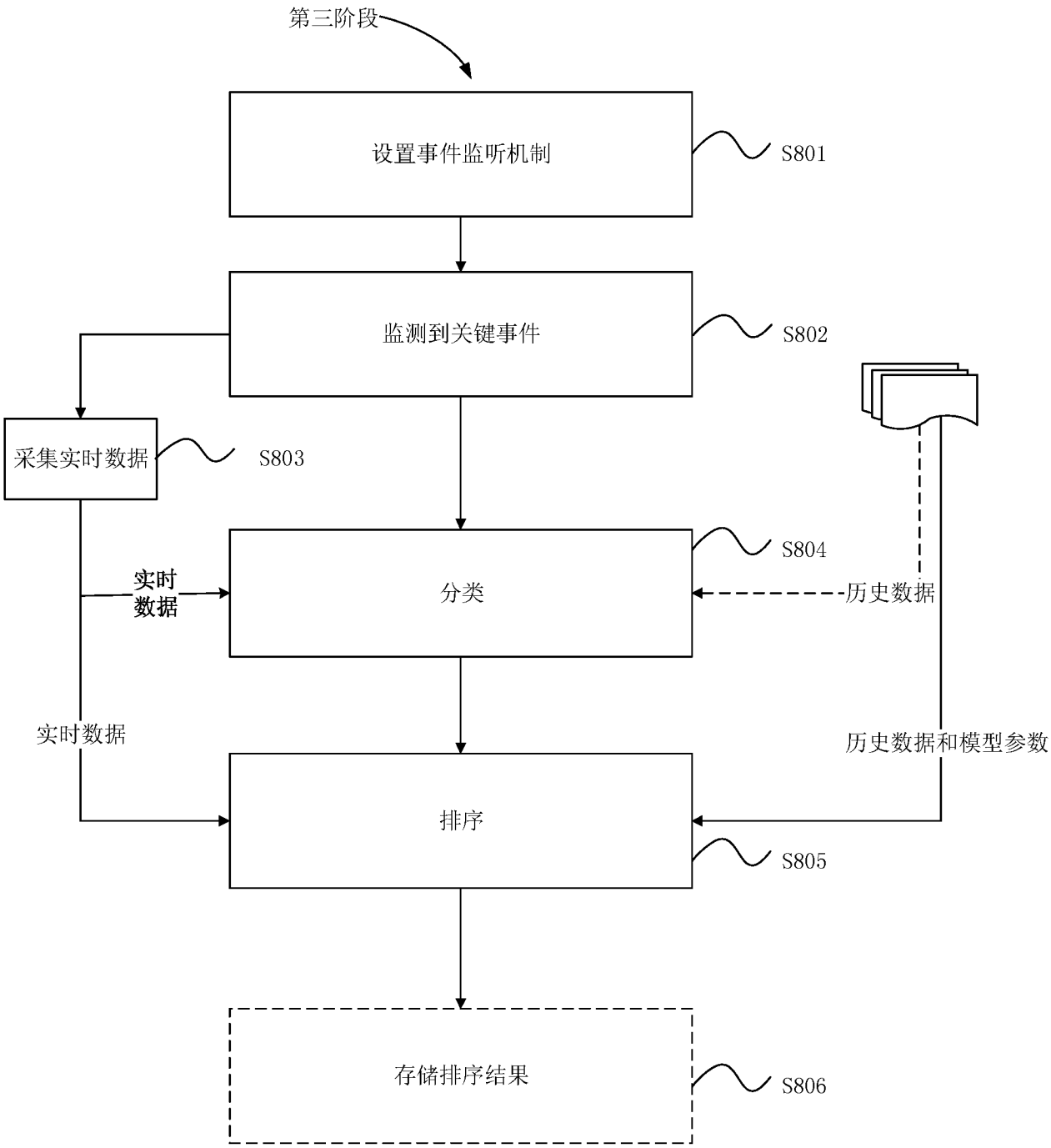
第三阶段
设置事件监听机制
S801
监测到关键事件
S802
采集实时数据
S803
S804
实时数据
分类
历史数据
实时数据
历史数据和模型参数
排序
S805
存储排序结果
S806

图 8

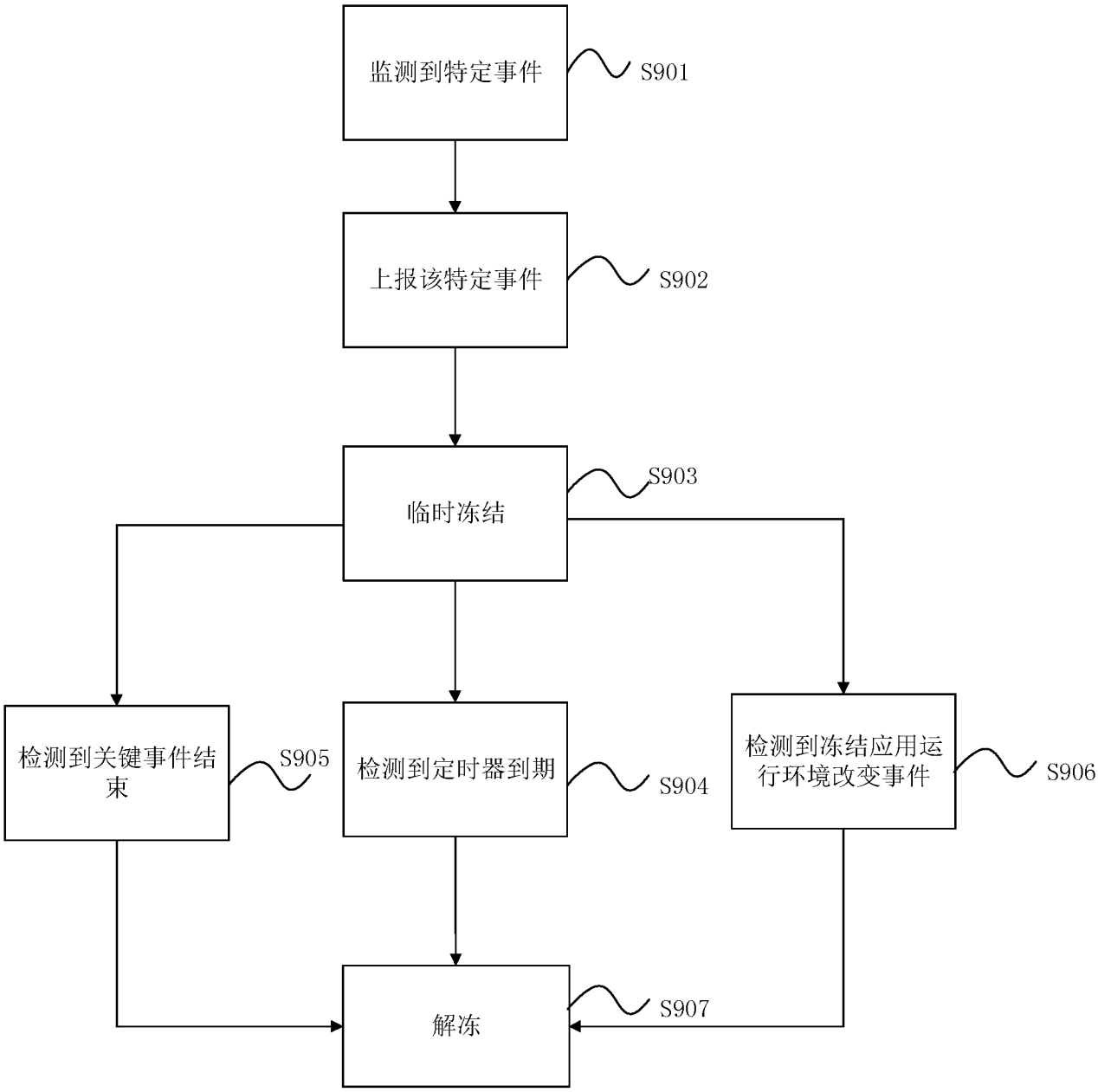
监测到特定事件
S901
上报该特定事件
S902
临时冻结
S903
检测到关键事件结束
S905
检测到定时器到期
S904
检测到冻结应用运行环境改变事件
S906
解冻
S907

图 9

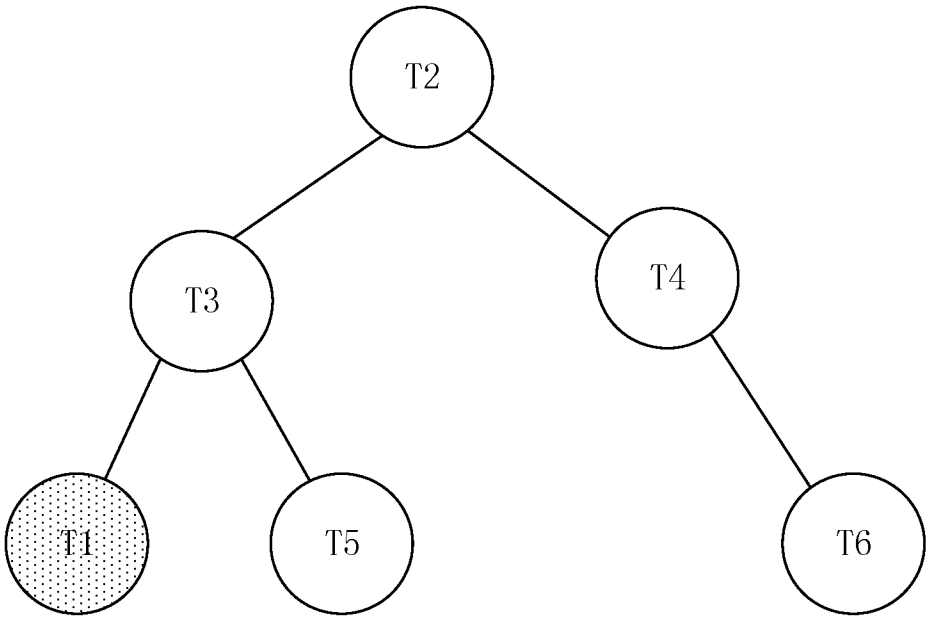

图 10

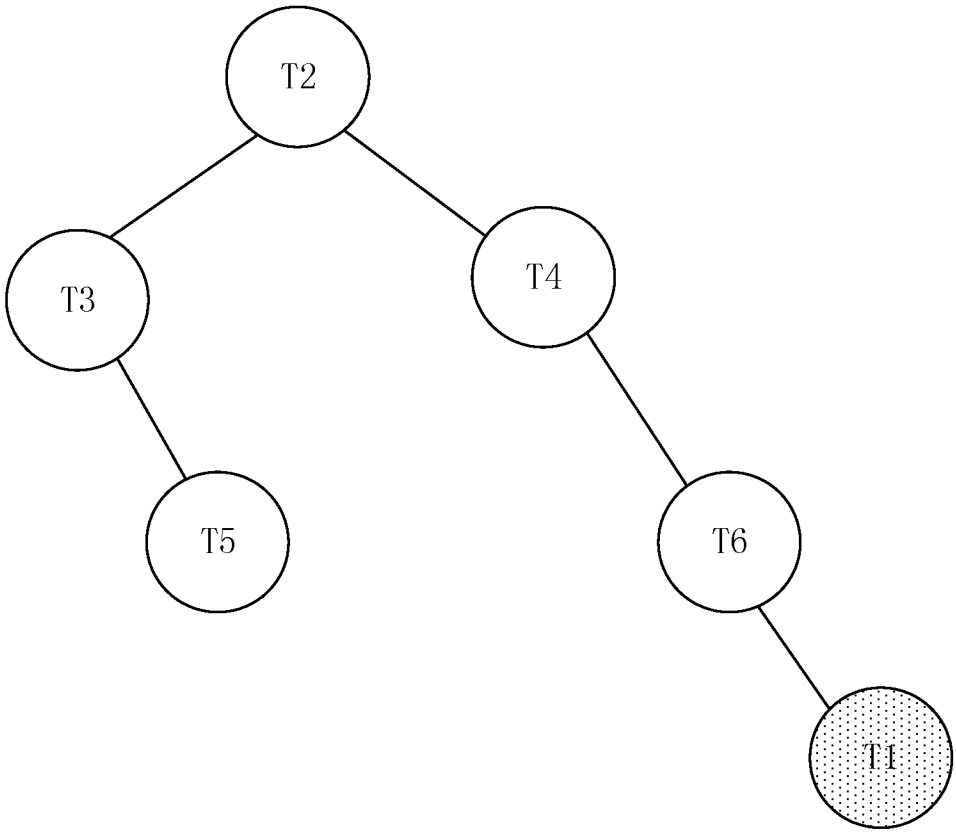

图 11

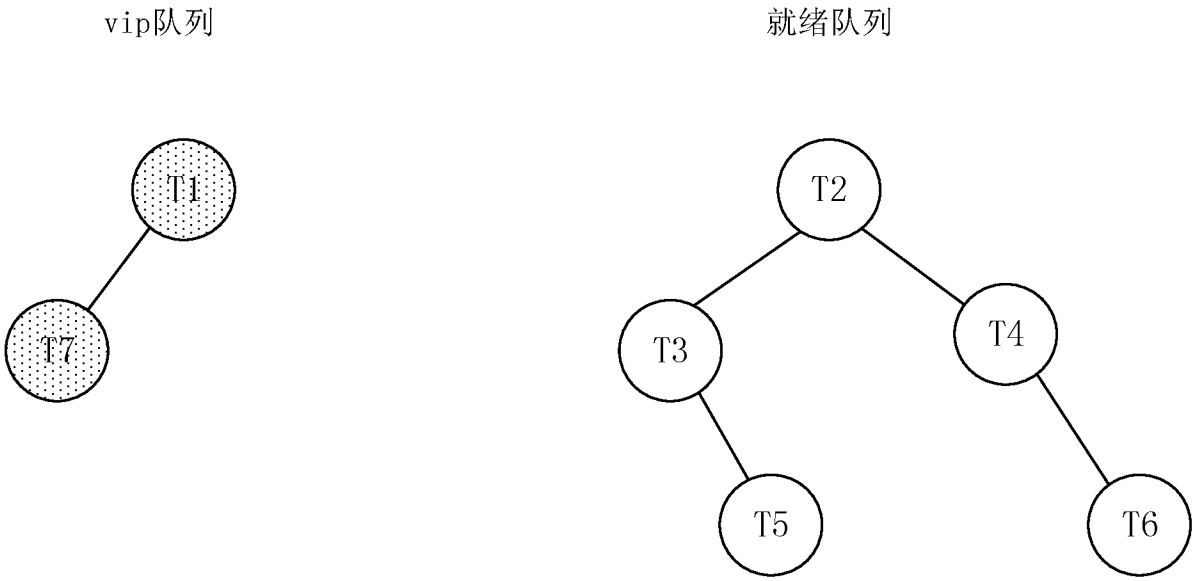

图 12

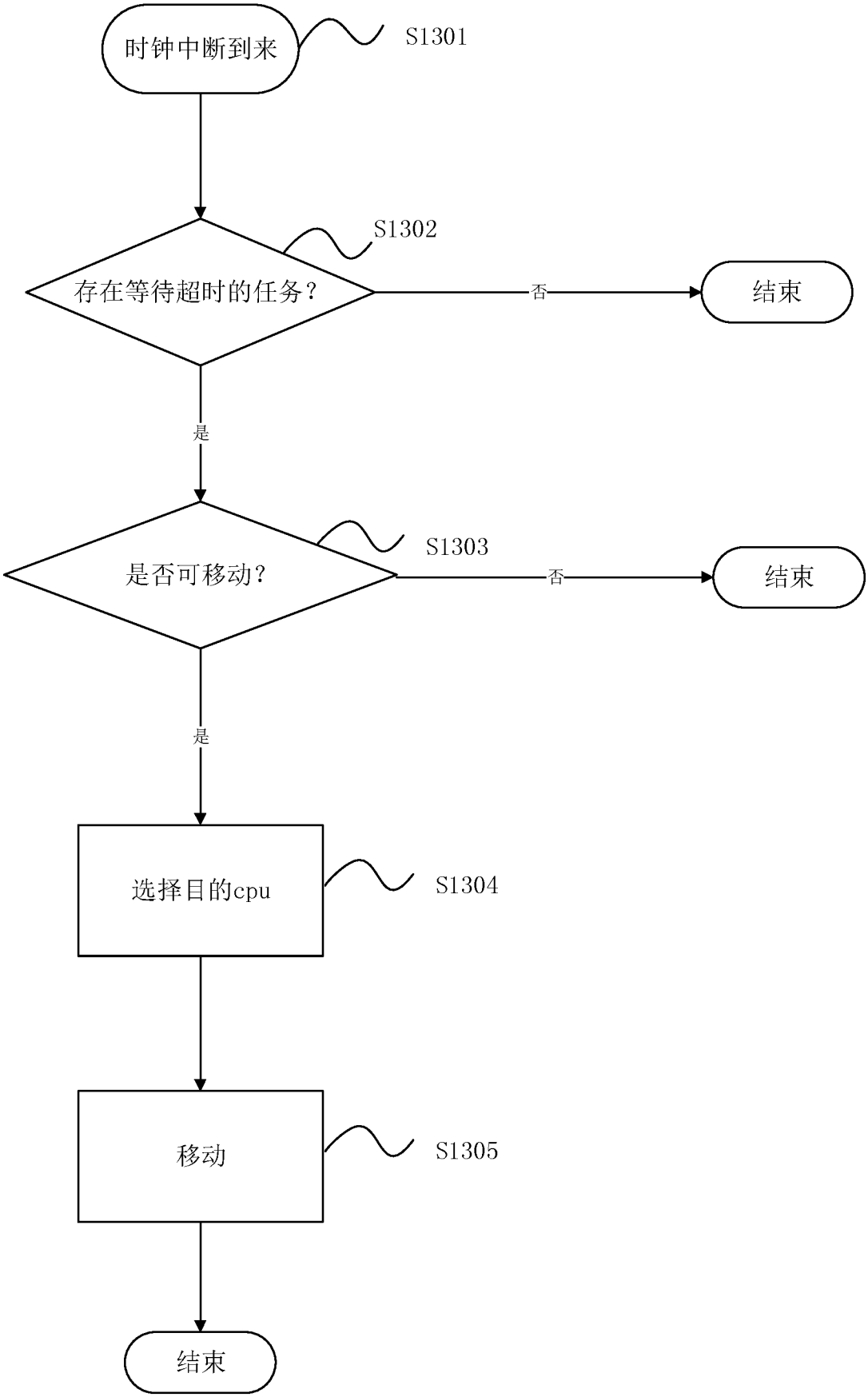
时钟中断到来
S1301
S1302
存在等待超时的任务？
否
结束
是
S1303
是否可移动？
否
结束
是
选择目的cpu
S1304
移动
S1305
结束

图 13

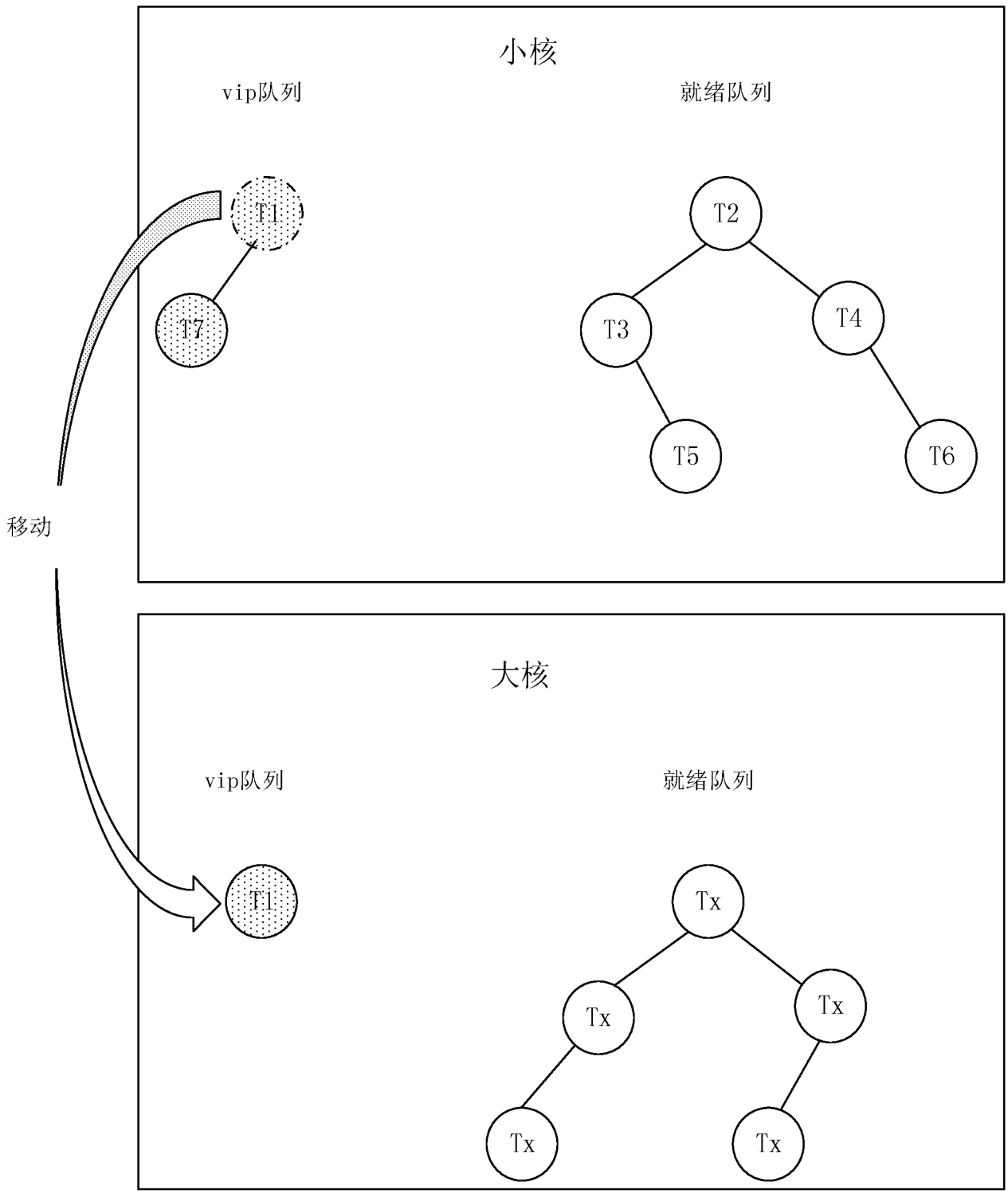
小核
vip队列
就绪队列
T1
T7
T2
T3
T4
T5
T6
移动
大核
vip队列
就绪队列
T1
Tx
Tx
Tx
Tx
Tx

图 14

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2018/109753

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/50(2006.01)i; G06F 9/48(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06Q;G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNKI, CNABS, CNTXT, DWPI, SIPOABS: 资源, 管理, 应用, 程序, 机器学习, 排序, 实时, 时间, 位置, 语义, 冻结, 解冻, 切换, 核, 队列, 任务, 依赖, resource, management, application, program, machine learning, sort+, real time, location, semantics, bind+, freez+, unfreez+, switch, core, queue, task, dependency

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CN 101923382 A (LENOVO (BEIJING) CO., LTD.) 22 December 2010 (2010-12-22) description, paragraphs 33-67	19-25
X	CN 106055399 A (YULONG COMPUTER TELECOMMUNICATION SCIENTIFIC (SHENZHEN) CO., LTD.) 26 October 2016 (2016-10-26) description, paragraphs 54-78	1-18
X	CN 101414270 A (ZHEJIANG UNIVERSITY) 22 April 2009 (2009-04-22) description, pages 5-7	26-33

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
07 January 2019	**21 January 2019**

Name and mailing address of the ISA/CN	Authorized officer
State Intellectual Property Office of the P. R. China **No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing** **100088** **China**	
Facsimile No. **(86-10)62019451**	Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2018/109753

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

[1] The claims comprise three inventions claimed in independent claims 1, 19 and 26, namely, claim 1 sets forth a method for managing resources in a computer system, claim 19 sets forth a method for temporarily freezing an application, and claim 26 sets forth a method for executing tasks in a computer system. The three inventions do not have the same or corresponding technical features, obviously also do not have the same or corresponding specific technical features, do not have any technical correlation, do not fall within a single general inventive concept, therefore do not comply with the requirement of unity of invention and do not comply with PCT Rule 13.1.

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☑ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2018/109753

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN 101923382 A	22 December 2010	CN 101923382 B	16 January 2013
CN 106055399 A	26 October 2016	None	
CN 101414270 A	22 April 2009	None	

国际检索报告

国际申请号 PCT/CN2018/109753

A. 主题的分类

G06F 9/50(2006.01)i; G06F 9/48(2006.01)i

按照国际专利分类(IPC)或者同时按照国家分类和IPC两种分类

B. 检索领域

检索的最低限度文献(标明分类系统和分类号)

G06Q;G06F

包含在检索领域中的除最低限度文献以外的检索文献

在国际检索时查阅的电子数据库(数据库的名称，和使用的检索词（如使用））

CNKI,CNABS,CNTXT,DWPI,SIPOABS:资源,管理,应用,程序,机器学习,排序,实时,时间,位置,语义,冻结,解冻,切换,核,队列,任务,依赖, resource, management, application, program,machine learning,sort+,real time, location, semantics, bind+, freez+, unfreez+,switch,core,queue,task,dependency

C. 相关文件

类 型*	引用文件，必要时，指明相关段落	相关的权利要求
X	CN 101923382 A (联想北京有限公司) 2010年 12月 22日 (2010-12-22) 说明书第33-67段	19-25
X	CN 106055399 A (宇龙计算机通信科技深圳有限公司) 2016年 10月 26日 (2016-10-26) 说明书第54-78段	1-18
X	CN 101414270 A (浙江大学) 2009年 4月 22日 (2009-04-22) 说明书第5-7页	26-33

☐ 其余文件在C栏的续页中列出。

☑ 见同族专利附件。

* 引用文件的具体类型：

"A" 认为不特别相关的表示了现有技术一般状态的文件

"E" 在国际申请日的当天或之后公布的在先申请或专利

"L" 可能对优先权要求构成怀疑的文件，或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件（如具体说明的）

"O" 涉及口头公开、使用、展览或其他方式公开的文件

"P" 公布日先于国际申请日但迟于所要求的优先权日的文件

"T" 在申请日或优先权日之后公布，与申请不相抵触，但为了理解发明之理论或原理的在后文件

"X" 特别相关的文件，单独考虑该文件，认定要求保护的发明不是新颖的或不具有创造性

"Y" 特别相关的文件，当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时，要求保护的发明不具有创造性

"&" 同族专利的文件

国际检索实际完成的日期	国际检索报告邮寄日期
2019年 1月 7日	2019年 1月 21日

ISA/CN的名称和邮寄地址	受权官员
中国国家知识产权局(ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088	王京霞
传真号 (86-10)62019451	电话号码 (86-10)62411718

表 PCT/ISA/210 (第2页) (2015年1月)

国际检索报告

国际申请号
PCT/CN2018/109753

第III栏　缺乏发明单一性的意见(续第1页第3项)

本国际检索单位在该国际申请中发现多项发明，即：

[1] 权利要求书包括独立权利要求1、19、26所要求保护的3项发明，即权利要求1请求保护一种在计算机系统中管理资源的方法，权利要求19请求保护一种临时冻结应用的方法，权利要求26请求保护一种计算机系统中执行任务的方法，上述3项发明不存在相同或相应的技术特征，显然也就不存在相同或相应的特定技术特征，不存在技术关联，不属于一个总的发明构思，因而不满足发明单一性的要求，不符合PCT实施细则13.1的规定。

1. ☐ 由于申请人按时缴纳了被要求缴纳的全部附加检索费，本国际检索报告涉及全部可作检索的权利要求。

2. ☑ 由于无需付出有理由要求附加费的劳动即能对全部可检索的权利要求进行检索，本单位未通知缴纳任何加费。

3. ☐ 由于申请人仅按时缴纳了部分被要求缴纳的附加检索费，本国际检索报告仅涉及已缴费的那些权利要求，具体地说，是权利要求：

4. ☐ 申请人未按时缴纳被要求缴纳的附加检索费。因此，本国际检索报告仅涉及权利要求书中首先提及的发明；包含该发明的权利要求是：

对异议的意见

☐ 申请人缴纳了附加检索费，同时提交了异议书，适用时，缴纳了异议费。

☐ 申请人缴纳了附加检索费，同时提交了异议书，但未在通知书规定的时间期限内缴纳异议费。

☐ 缴纳附加检索费时未提交异议书。

国际检索报告
关于同族专利的信息

国际申请号
PCT/CN2018/109753

检索报告引用的专利文件	公布日（年/月/日）	同族专利	公布日（年/月/日）
CN 101923382 A	2010年 12月 22日	CN 101923382 B	2013年 1月 16日
CN 106055399 A	2016年 10月 26日	无	
CN 101414270 A	2009年 4月 22日	无	