Android定制系统中的性能优化策略与实现

胡亚军[1,2]

(1. 清华大学深圳国际研究生院，广东 深圳 518055；2. 长城汽车股份有限公司，河北 保定 071000)

摘要： 随着Android智能终端功能的不断增强，用户对智能设备的性能要求日益提高。Android操作系统，凭借其开放性和可定制性，为用户提供了丰富的个性化选项。然而，定制过程中的优化对于保证系统流畅运行和良好用户体验至关重要。为探讨Android定制系统中的性能体验测量与优化策略，分析了Android系统性能体验的基本概念和测量方法，指出常见的性能问题，如内存泄漏、中央处理器(CPU)过载和输入/输出(I/O)阻塞等，结合用户体验感知，提出三个关键的优化策略，包括系统启动时间、应用启动时间和运行流畅度等。通过实验评估，验证了这些优化策略在提升系统性能方面的有效性。结果表明，通过系统化的性能优化，可以显著提升Android定制系统的性能，增强用户体验。

关键词： Android定制系统；性能优化；用户体验；资源管理；系统流畅性

中图分类号： TP312 **文献标志码：** A **文章编号：** 1671－1807(2024)24－0294－08

在移动互联网时代，Android智能产品已成为人们日常生活中不可或缺的设备。随着应用需求的不断增长，用户对智能产品性能的期望值也在不断提升。Android操作系统，以其开放源代码的特性和广泛的设备兼容性，占据了全球智能终端市场的主导地位。然而，随着应用功能的日益复杂化，Android设备面临着严峻的性能挑战，尤其是在定制系统中，开发者为了提供更加个性化的用户体验，往往会引入额外的功能和界面改动，这些改动在丰富用户体验的同时，也可能带来性能上的负担。性能优化是提升Android定制系统用户体验的关键。一个流畅、响应迅速的系统能够显著提高用户的满意度和忠诚度。因此，研究和实现有效的性能优化策略对于Android定制系统至关重要。

由于Android系统的版本碎片化、硬件的多样化和App生态的复杂性，Android设备的触控响应延迟和卡顿现象通常比iOS系统更为严重。尽管流畅性是用户的主观感受，近年来，一些专家学者以智能手机和智能汽车作为载体，对Android定制系统性能评价指标和优化策略进行了研究。李建伟[1]从用户体验的角度，对智能手机的软硬件构成进行了深入研究，把性能分为面向硬件的基准性能、面向软件的响应时延和界面操作的流畅度。张博涵[2]在对Android开发中的流畅性优化方案进行研究时，将Android的流畅性分为启动速度、绘制速度和响应速度。赵子渊[3]通过技术性能指标、滑动流畅度、响应速度、启动速度进行智能手机的自动化性能测试，其中技术性能指标包括液晶显示屏(liquid crystal display，LCD)的刷新速率、随机存取存储器(RAM)占用率、只读存储器(ROM)使用状况等。尤增显[4]通过用户调研，认为响应性能、显示性能、稳定性和易用性是用户视角下的流畅度的影响因素。Ng和Dietz[5]认为触摸交互系统的关键性能指标包括帧速率和响应时间。LI等[6]认为用户感知的手机性能包括准确性、一致性、响应性和流畅度。谈笑[7]认为智能手机的交互操作性能包括准确性、跟手性、响应性和流畅性。李丹等[8]认为启动速度和界面运行流畅性是智能座舱体验评价的关键因素。晏江华等[9]通过响应时间和流畅度来测评智能座舱的性能，其中响应时间包括开机时间、应用启动时间、路径规划时间、语音识别及唤醒时间等，流畅度包括划屏、界面切换、导航和语音交互等。李盈盈[10]认为智能座舱的客观体验指标主要由响应速度、流畅度和故障率构成。郁淑聪等[11]将智能座舱的系统性能分解为功能满意度、信息处理能力和显示性能三个二级指标。他们从不同维度尝试对Android定制系统的性能指标进行定义，虽然指标不尽相同，也没有对各个指标对用户体验的

收稿日期： 2024-07-22

作者简介： 胡亚军(1980—)，男，湖南衡阳人，博士，总工程师，研究方向为汽车智能化技术、智能产品用户体验、Android定制化系统架构与关键技术等。

影响程度以及如果测量给出方案,但是在开机时间即系统启动时间、应用启动时间和响应时间等方面仍存在广泛的共识。基于这个共识,相关专家学者尝试通过不同的方法对 Android 定制系统的性能进行优化。胡鼎等[12]通过优化 Bootloader、Linux 内核和 RamDisk,将 Android 车载倒车系统启动时间缩短了 14 s 左右。陈禹樵和隋浩[13]从启动流程、布局渲染和代码规范三个方面对 Android 应用进行优化,对于新应用的开发和旧应用的重构和优化具有一定的借鉴作用,但是 Android 应用种类繁多、场景多变,仅仅依靠这三个方面的优化明显不够,需要有更多的优化手段和创新方法来提升 Android 应用的启动速度。刘凯等[14]提出一种基于双操作系统的应用快速启动和双备份方案,通过同时在 QNX 和 Android 系统中备份关键应用程序,来保证倒车影像等的快速启动,该方案对于保障核心应用的运行效果显著,但是受限于硬件资源的限制,难以大范围推广。

性能优化不仅涉及代码层面的优化,如内存管理、中央处理器(CPU)调度和输入/输出(I/O)操作,也包括系统层面的优化,如资源调度、进程管理和电源管理等。本文的目的在于深入探讨 Android 定制系统中的性能优化策略,并提出具体的优化技术与实现方法。本文首先分析了 Android 系统中常见的性能问题,并基于这些问题提出了一系列针对性的优化策略;随后,详细介绍了这些优化策略的实现方法,并通过实验评估了优化措施的效果。

1 Android 定制系统概述

在智能操作系统领域,Android 以其开源特性和基于 Linux 的架构处于领先地位,为定制和优化提供了广泛的可能性。本文深入探讨了 Android 定制系统的基本概念和架构基础,并强调性能优化在提升用户体验中的关键作用。Android 是一种多才多艺的操作系统,最初设计用于触摸屏移动设备,如今已超越其原始范畴,扩展到电视、汽车系统和可穿戴设备等多种设备。

系统的架构分为四个明确的层次。应用层:通过电话、短信和浏览器等应用程序直接吸引用户;应用框架层:提供一系列服务和组件,对应用开发至关重要;运行时层:由 Android 运行时支持,继 Android 5.0 之后取代 Dalvik 虚拟机;Linux 内核层:作为基础操作系统层,负责硬件抽象、驱动程序和管理内存。

Android 设备的普及引发对个性化和精细化用户体验需求的不断增长,催生定制系统的出现。这些系统使制造商和开发者能够根据特定需求调整 Android 平台,包括用户界面的改进、性能的提升、功能的增加或减少以及安全性的增强。定制通常是通过修改系统源代码实现的,允许对各种组件进行选择性增强或替换。常见的定制方法包括对启动器和系统用户界面进行美学和功能上的改变,调整 CPU 和内存管理以提高运行效率,集成额外的应用程序或服务,以及加强系统安全协议。

性能优化在 Android 定制系统中显得尤为重要,因为定制过程可能会引入新的性能瓶颈。性能增强策略是多方面的,包括减少内存泄漏、策略性地分配 CPU 资源、优化 I/O 操作以减少延迟,以及提高用户界面(user interface,UI)渲染效率,减少过度绘制和布局重排。

总之,虽然 Android 的定制系统为用户提供了丰富的个性化选项,但同时也带来了性能优化的挑战。深入理解 Android 的架构框架和定制系统的特点对于有效实施性能优化策略至关重要,从而将用户体验提升到一个新的高度。

2 性能优化策略

2.1 Android 系统性能体验的基本概念及测量方法

性能体验,即流畅性,指的是用户界面操作交互的流畅程度。智能终端的交互技术从过去功能机时代的按键发展到智能机时代的触屏,流畅性便随之而来,成为智能终端用户体验的重要组成部分,流畅性的好坏通常代表用户对终端性能的总体评价,将直接影响到用户使用智能产品的心情。由于 Android 系统版本的碎片化、硬件的多样化和 APP 生态的复杂性,加之 Android 是运行虚拟机上,触控响应的延迟和卡顿都比 iOS 系统严重,一直以来,相对于 iOS,Android 系统的流畅性一直为用户所诟病。流畅性是一个用户的主观感受,一直没有一个客观可靠的指标来进行测量。

狭义的流畅性即帧率,也就是一秒钟渲染的画面数,本文使用流畅度指标来衡量,而用户在使用智能手机的过程中所感受到的性能水平,通过对国内外学者的研究成果进行归纳和提炼,广义上的流畅包括硬件的基准性能和软件的交互操作性能。其中硬件的基准性能包括 CPU、IO、内存和图形处理性能,软件的交互操作性能包括响应时间、滑动流畅度,硬件配置是跟智能手机价格成正向关联关系,分配 20%的权重,软件交互是体验优化和改进

的重心，也是用户场景体验的集中表现，占80%的权重。响应时间表示的是用户完成某项任务时手机反应的快与慢，根据Shane[15]的调查表明，程序终止和响应慢是导致用户卸载应用程序、差评的主要原因，这两项都与响应时间相关，滑动的流畅度则跟画面的切换有关，分别分配60%和20%的权重。为了偏于测试取值，把响应时间进一步拆分为纯净环境和重载环境下的响应时间两个指标，纯净环境是一个理想值，也可以成为软件基准值，占20%，重载环境下能更大程度上反映用户的容忍度，占40%（表1），工程体验指数通过这4个测量指标来反映智能手机的流畅性，对于流畅性的度量转化为这4个测量指标的用户体验测试。

表1 Android终端性能测量方法

评价指标	测量指标	权重/%
流畅性	纯净环境响应时间	20
	重载环境响应时间	40
	流畅度	20
	基准性能	20
计算公式	流畅性=纯净环境响应时间得分×20%+重载环境响应时间得分×40+流畅度×20%+基准性能×20%	

Android终端性能可以通过三个维度进行评估和测量，因为基准性能主要由硬件决定，所以性能体验优化主要针对响应时间和流畅度进行，其中响应时间根据测试环境的不同，可以分为纯净环境响应时间和重载环境响应时间，如果从优化的角度来看，最关键的指标包括系统启动时间和应用启动时间。

2.2 性能问题识别与优化策略

在Android定制系统的开发过程中，性能优化是确保用户体验和系统流畅性的关键环节。以下内容将对Android定制系统中的性能优化策略进行精炼概述，并提出实现方法。

性能问题的识别是优化的前提。内存泄漏、UI渲染效率低下、CPU使用率过高、I/O性能瓶颈以及响应速度等问题，均可通过Android Studio提供的工具进行监测和分析。例如，LeakCanary和MAT用于检测内存泄漏，而CPU Monitor和Android Profiler则用于监控CPU使用情况和I/O性能。

本文的性能优化策略是从系统启动速度、应用启动速度和运行流畅度这三个方面进行性能优化。对Android系统启动速度的优化主要通过优化系统启动的顺序以及基于休眠唤醒的快启技术，除此之外，还可以通过优化Bootloader、内核剪裁等技术直接优化Linux内核的启动和加载时间。对Android应用启动速度的优化主要通过运用AI预加载技术，通过改变应用启动流程，变串行为部分并行，实施异步布局加载，从而可以大幅缩短应用启动时间。对Android运行流畅度的优化重点是解决低内存的资源分配问题，需要对Android系统的低内存管理策略进行优化。

3 性能优化技术实现

3.1 优化系统启动速度

要优化Android系统的启动速度，需要先研究Android系统的启动过程，Android系统的启动包括Bootloader、Linux Kernel和Android三个阶段。Bootloader和Linux Kernel的启动耗时一般都在10 s以内，而Android应用系统的启动占了整个过程的80%时间消耗。包括谷歌在内的很多公司及组织都对Android系统的启动时间的优化进行过研究，然而由于Android应用系统本身的复杂性，导致整个优化工作陷入瓶颈，收效甚微。本文的研究方向是参考个人电脑的休眠唤醒的模式尝试跳过Android应用系统的启动过程，从而缩短开机时间，即Android快启技术，Android系统快速启动的流程如图1所示。

相对于Android正常启动的Bootloader、Kernel再到Android应用系统，快启技术将Android应用系统的启动转变为直接读取保存在内存中的系统镜像，并恢复系统。整个开机时间从正常启动的Bootloader启动时间、Kernel启动时间与Android应用系统启动时间之和，转化为Bootloader、Kernel与系统恢复的时间之和，经测试，采用快启技术，较正常启动可以优化60%的开机时间。同时，系统恢复的时间占比最大，超过50%，Bootloader的启动时间最短，一般在2 s以内，为了给用户提供更好的开机体验，对读取镜像并恢复系统的过程和内核初始化进行进一步优化。

如图1所示，内核的初始化的核心任务就是对重要的子系统进行初始化以及探测外部设备驱动，前者对于系统的正常运行不可或缺且速度快，后者由于智能终端的外部设备众多，且快慢不一，占用了内核初始化时间的70%以上。经研究，读取镜像只需要加载Flash的驱动即可，其他的驱动在这个阶段可以直接忽略，于是对内核初始化的优化只需要在快启环节剔除设备驱动的探测即可以大幅提升开机时间。

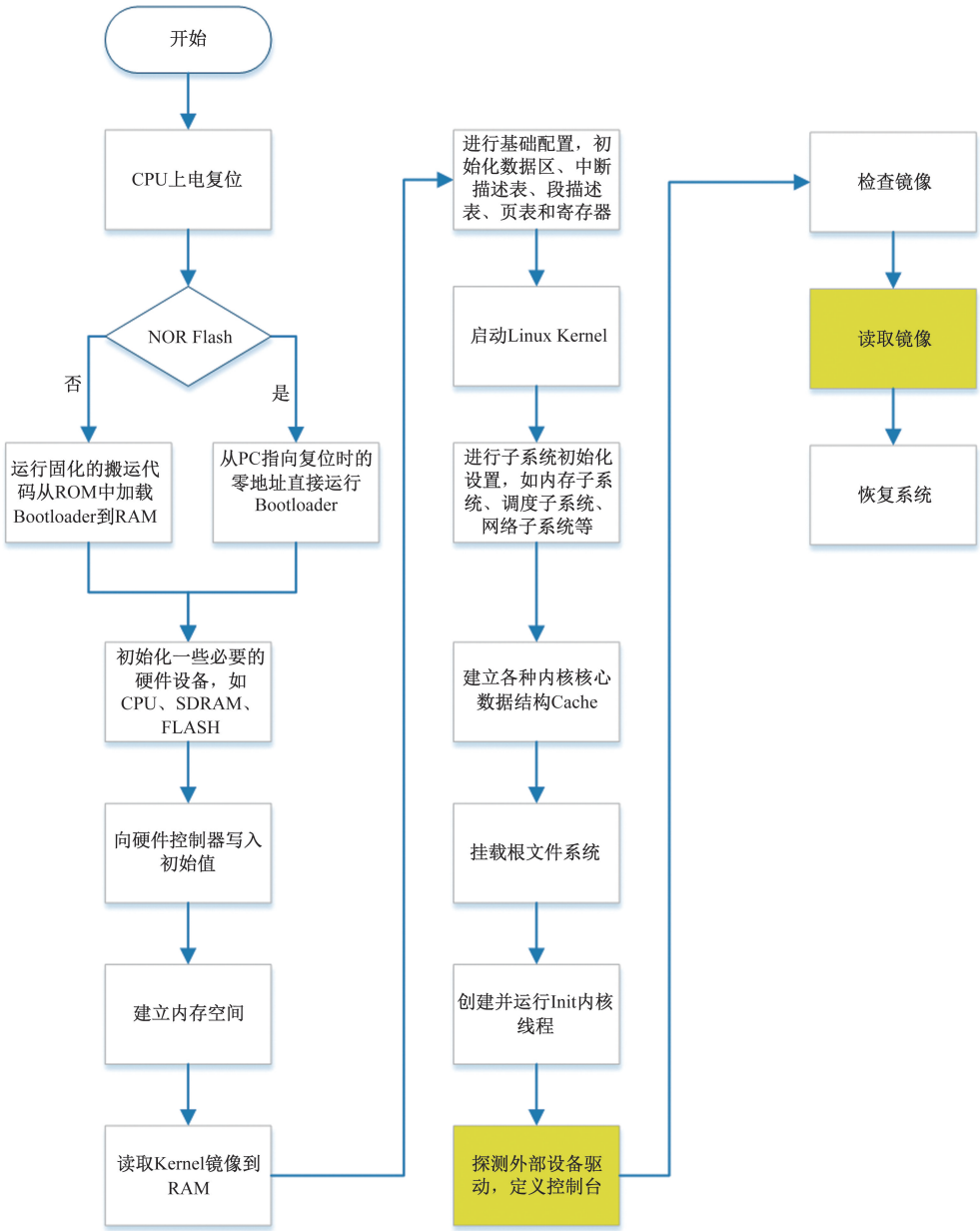

图 1　Android 系统快速启动流程

读取镜像并恢复系统的过程包括读取和恢复两个步骤,一般来说,恢复系统的时间非常快,通常不到 1 s,于是优化的重心落在了读取镜像上。读取镜像的时间主要取决于镜像本身的大小以及 Flash 的读取速度,然而 Flash 的读取速度由硬件决定,与系统无关,对读取镜像并恢复系统的进一步优化措施就是缩小镜像的大小,在读取速度一定的情况下,镜像文件越小,则启动速度越快。经研究分析,构成系统镜像的内容包括系统关键部分和其他非关键部分,前者包括内核占用的内存和 Android 的 Native Service、Dalvik 虚拟机、Zygote 等关键进程,很明显这部分没有优化空间,后者一般指可回收的用户应用程序。因此,对镜像大小的优化只需要将后者从镜像中切除即可大幅缩小。

3.2　优化 Android 应用启动速度

一个安卓应用的正常启动流程一般是串行的,从手指按下、抬起、创建与绑定线程、加载资源、布局计算、UI 渲染、加载资源到播放动画,这些环节所消耗的时间之和即为该应用的启动时间。在安卓应用的正常启动流程基础上,采用了点击预加载、资源预加载、进程预加载技术,大幅提升了应用的启动速度。

应用预加载技术的前后对比图如图 2 所示,预加载的核心就是改变应用的串行启动流程为部分并行。

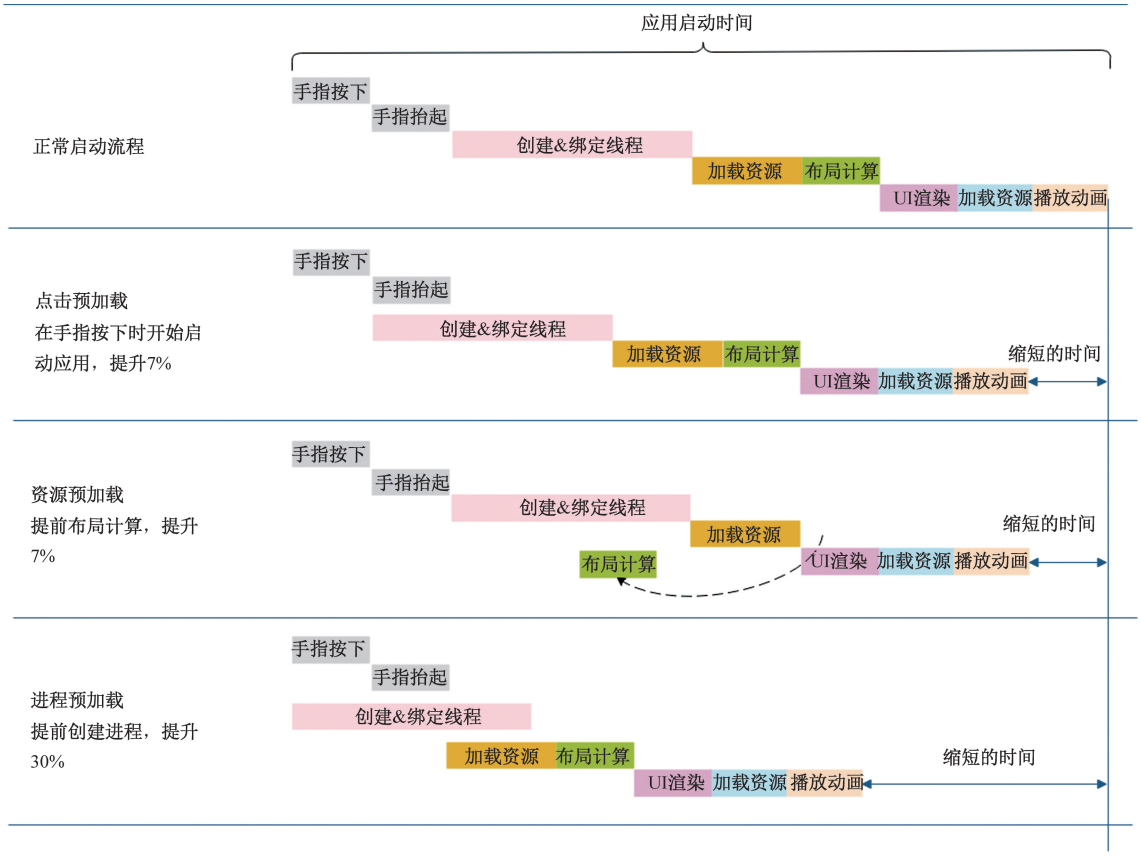

图 2　预加载技术方案

3.3　优化 Android 系统流畅度

卡顿的产生可以分为两类，一类是系统层面，另一类是应用层面。前者常见的原因由 Surface Flinger 主线程调用超时、后台活动进程太多、主线程处于 Runnable 状态迟迟无法调度、系统层面的应用程序管理服务(activity manager service，AMS)锁和窗口管理服务(window manager service，WMS)锁导致系统任务阻塞、Layer 过多导致 Surface Flinger Layer 的计算超时；后者常见的原因有主线程执行时间过长、后台繁忙导致主线程 Binder 超时、应用界面复杂导致 Web View 性能不足、帧率和刷新率不相符导致画面不顺畅等。

经过对 Google Vitals 有关通用应用界面卡顿标准和腾讯 Perf Dog Jank 有关游戏应用界面卡顿标准的研究，程序员无法通过平均帧率来判断 Android 系统是否卡顿，而只有当与用户交互的应用或游戏出现丢帧时，才可以确定系统或应用的何处出现了卡顿，因此，可以借助 sumpsys 工具去获取丢帧的信息，再通过卡顿问题定位工具如 gfxinfo、Systrace 或 Perfetto、Layout Inspect、Block Canary、CPU 性能剖析器和图形处理器(graphic processing unit，GPU)渲染模式分析工具来锁定具体的卡顿原因。

针对系统层面产生的卡顿，无论是主线程调用超时或无法调度、后台进程太多还是系统锁、计算超时，都与系统可用内存低息息相关，因此，系统层面的优化核心是解决低内存的资源分配问题，需要对 Android 系统的低内存管理策略进行优化。

缺省情况下，Android 按照进程的重要性，将进程分为六个类别[16]，如图 3 所示，重要性依次降低。

前台进程(Forground)：目前正在显示的进程以及屏幕上显示的一些系统进程，如呼叫、全局搜索等。

可见进程(Visible)：有一些不在前台，但用户还是看得见的进程，比如输入法，虽然这部分进程不在前台，但是和平时使用的有很大的关系。

服务进程(Server)：目前运行的一些服务，包括

图 3　Android 进程管控优先级

主、次服务，但主服务不可能被进程管理所终止，如拨号等，因此低内存下的进程管控主要谈次服务，如联系人内部存储。

后台进程（Hidden）：如文件管理器、计算器，当程序显示在屏幕时，它们所运行的进程即为前台进程，当用户按下 Home 键，一旦回到主桌面，就变成一个后台进程。后台进程的管理是进程管控的核心和重心，其策略有很多种，大体上可以分为两类：一种是激进的方式，比如程序一到后台就马上终止，这样可以使程序的运行效率和表现达到最大化，但因为没有程序缓存，无法加速程序的再次或者反复启动速度；另一种是保守的方式，即尽可能多地保留后台程序，虽然会因占用过多的内存空间而影响在行程序的运行速度，但再次启动以运行过的程序时，速度较首次会有明显的提升，这里就需要结合实际的场景找到一个最佳平衡点。

内容提供方（Content Provider）：没有程序实体，只提供用于其他程序的内容，如通信录内容提供方、通话记录内容提供方等。

空进程（Empty）：在程序退出后，没有任何数据留在内存中运行的进程如 BTE，为了提高下一次启动程序的速度，或者对程序的一些历史信息进行记录，依然会在进程中驻留一个空进程。

Android 是基于 Linux 内核的一种嵌入式操作系统，其内存管理则不同于 Linux 的内存溢出（out of memory，OOM）方式，Linux 的进程一旦活动结束就会结束进程并回收内存，考虑到嵌入式系统中频繁启动的应用程序、有限的 CPU 算力和内存空间，在进程活动结束时，Android 并不会立即终止它们，而是将这些进程保存在内存中，如果手机内存足够，会发现系统可用的内存越来越少，直到一个特别消耗内存的应用启动，这个时候，Android 的内存管理才会启动如图 4 所示的优先级策略，杀掉一些它认为优先级不高和内存阈值太多的应用，即低

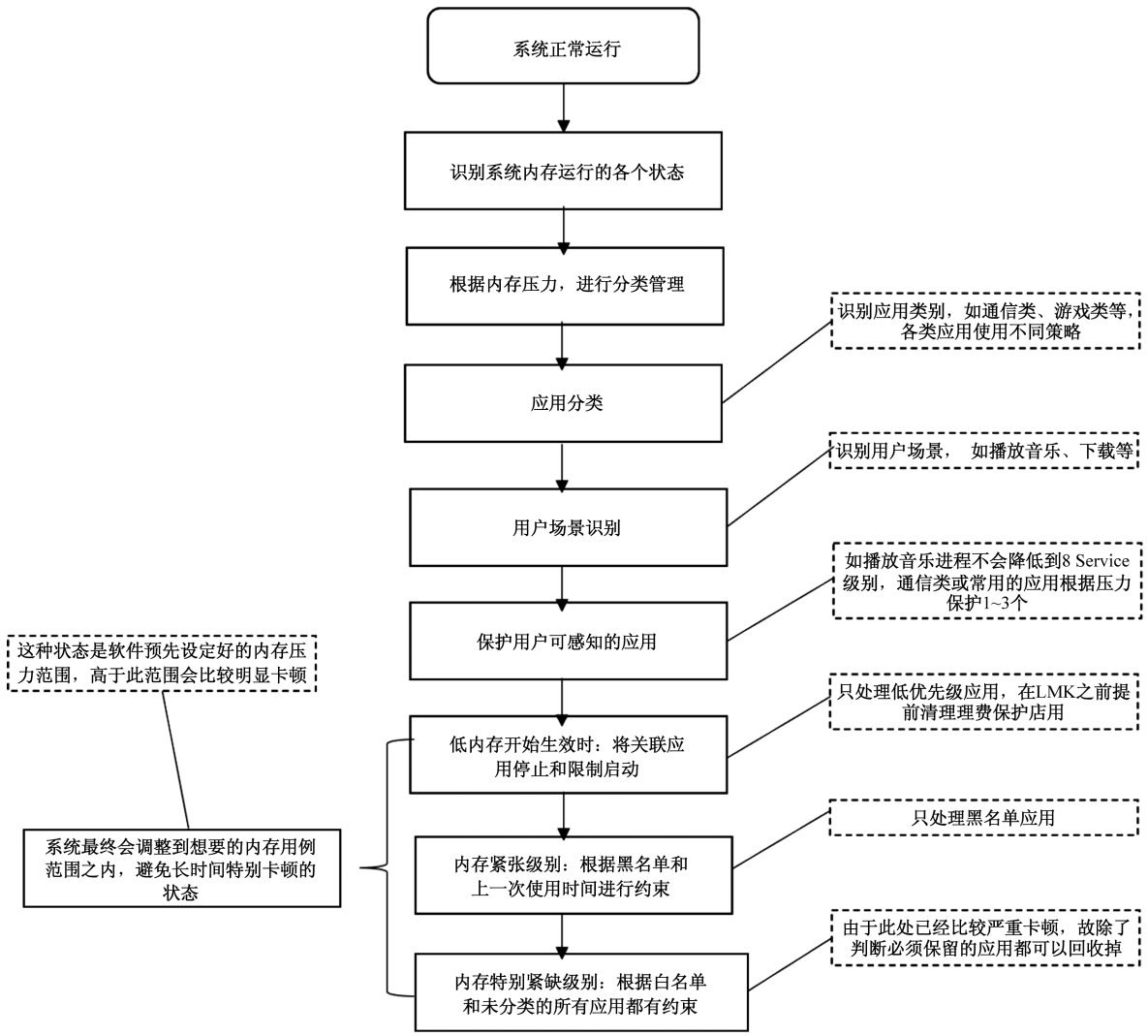

图 4　低内存下的进程管控方案

内存管控策略(low memory killer,LMK),这也是Android 智能终端产品越用越慢的首要原因。LMK 是一种基于 OOM 兼顾嵌入式系统的特点修改的内存管理机制,它不会立即杀死执行完毕的进程,只有在系统内存不足的情况,会选择终止 Bad 进程并释放内存。Bad 进程有两个选择标准:代表进程优先级的 oom_adj 和进程占用内存的大小,优先级不同选择优先级高的,优先级相同则选择占用内存大的。安卓内核每间隔一段时间会检查当前可用内存低于 oom_adj 所需空间内存的门槛值与否,如果是,就杀死 Bad 进程[17]。

基于 LMK 策略,结合终端用户的实际使用场景和设备特性,提出一种新的低内存管理策略,该策略的进程管控的技术方案如图 4 所示。

应用分类方法是,通过采样终端中的多个应用程序的配置文件的特征,并结合朴素贝叶斯算法实现对待分类的应用程序进行分类,纯软件算法实现。

针对应用层面产生的卡顿,需要具体应用具体分析,一般可以围绕对象分配、垃圾回收、线程调度和 Binder 调用来进行,具体措施包括布局优化、列表优化、主线程耗时优化和内存优化,如图 5 所示。

4 结论与展望

本文深入探讨了 Android 定制系统中的性能优化问题,并提出一系列针对性的优化策略与实现方法。通过实验评估,证实了这些优化措施在提升系统性能方面的有效性。尽管取得了一定成果,但也遇到了如兼容性问题、代码复杂性增加等挑战。这些挑战提示,在未来的工作中需要更加注重优化措施的通用性和易用性。

性能优化是一个不断发展的领域,未来的研究

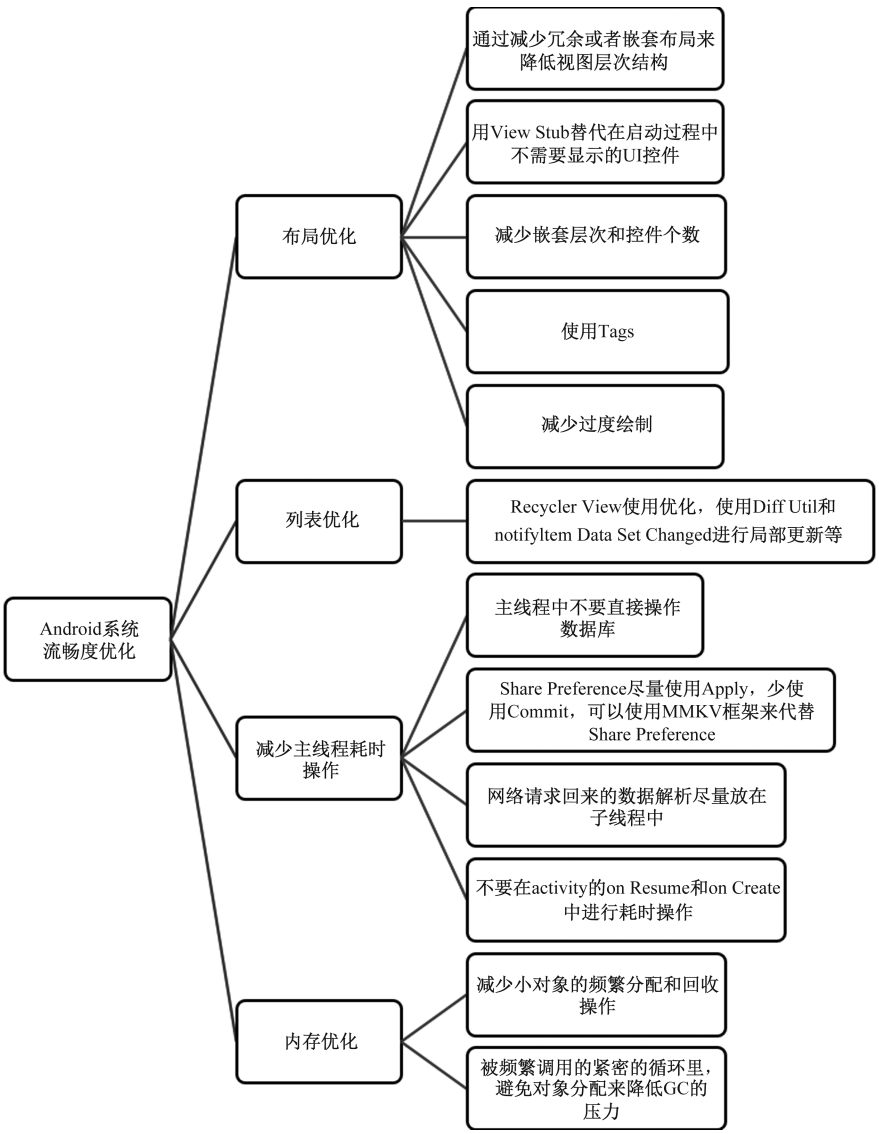

图 5 Android 系统流畅度优化技术

可以探索自动化优化工具的开发、个性化优化策略的应用，以及人工智能（artifical intellegence，AI）和机器学习技术在性能优化中的潜力。基于当前研究结果，未来的工作可以集中在以下几个方向：

（1）自动化优化：研究自动化性能优化工具，以减少人工介入，提高优化效率；

（2）个性化优化：根据用户行为模式进行个性化的性能优化，以进一步提升用户体验；

（3）跨版本兼容：开发更通用的优化策略，以支持不同版本的 Android 系统；

（4）AI 与机器学习：探索使用 AI 和机器学习技术预测性能瓶颈并自动调整系统资源。

总体而言，性能优化策略对于提升 Android 定制系统的性能至关重要。通过综合考虑内存管理、CPU 调度和 UI 渲染等多方面因素，可以显著提高用户体验。未来的研究应继续探索新的优化技术和方法，以适应不断变化的市场需求和用户期望。

通过本文的研究，期望能够为 Android 开发者提供实用的优化指导，帮助他们构建更加高效、流畅且节能的移动应用。同时，也期待未来的研究能够在现有基础上进一步推动 Android 性能优化技术的发展。

参考文献

[1] 李建伟. 智能手机用户体验测评系统研究与实现[D]. 北京：北京邮电大学，2014.

[2] 张博涵. 安卓开发中的流畅性优化方案研究[J]. 电子设计工程，2021，29(21)：168-172.

[3] 赵子渊. 面向 Android 手机的自动化性能测试工具的设计与实现[D]. 哈尔滨：哈尔滨工业大学，2017.

[4] 尤增显. 基于用户体验的智能终端流畅度评测研究与实现[D]. 北京：北京邮电大学，2017.

[5] NG A，DIETZ P H. The effects of latency and motion blur on touch screen user experience[J]. Journal of the Society for Information Display，2014，22(9)：449-456.

[6] LI X F，WANG Y，WU J，et al. Mobile OS architecture trends[J]. Intel Technology Journal，2012，4(16)：178-198.

[7] 谈笑. 基于手机交互操作响应性的用户体验研究[D]. 长沙：湖南大学，2019.

[8] 李丹，郑红丽，回姝，等. 智能网联时代汽车智能座舱操作系统的发展[J]. 汽车文摘，2022(5)：1-6.

[9] 晏江华，刘铁山，郑苗苗，等. 智能座舱系统测评技术研究[J]. 中国汽车，2022(8)：51-57.

[10] 李盈盈，刘畅. 汽车智能座舱系统测评技术分析[J]. 汽车测试报告，2022(20)：55-57.

[11] 郁淑聪，孟健，郝斌. 基于驾驶员的智能座舱人机工效测评研究[J]. 汽车工程，2022，44(1)：36-43.

[12] 胡鼎，陈艳，盖志刚，等. 车载 Android 操作系统快速倒车启动技术研究[J]. 企业科技与发展，2018(10)：81-83.

[13] 陈禹樵，隋浩. Android 应用启动速度优化策略研究[J]. 信息与电脑，2021，33(8)：4-7.

[14] 刘凯，夏欢，吴刚，等. 基于双操作系统的应用快速启动和双备份方案[J]. 汽车知识，2024，24(1)：1-5.

[15] SHANES. Survey：exploring the reasons users complain about Apps[EB/OL]. (2012-11-09)[2024-06-30]. www.fiercewirelss. com/developer/survey-exploring-reasons-complain-about-Apps.

[16] 杨丰盛. Android 应用开发揭秘[M]. 北京：机械工业出版社，2010.

[17] 杨丰盛. Android 技术内幕·系统卷[M]. 北京：机械工业出版社，2011.

Performance Optimization Strategies and Implementation in Android Custom Systems

HU Yajun[1,2]

(1. Shenzhen International Graduate School of Tsinghua University，Shenzhen 518055，Guangdong，China；
2. Great Wall Motor Company Limited，Baoding 071000，Hebei，China)

Abstract：With the continuous enhancement of functions in Android smart terminals，users' performance requirements for smart devices are increasingly high. The Android operating system，with its openness and customizability，provides users with a wealth of personalized options. However，optimization during the customization process is crucial to ensure smooth system operation and a good user experience. It is aimed to explore the measurement and optimization strategies of performance experience in Android customized systems. The basic concepts and measurement methods of Android system performance experience were analyzed，common performance issues were pointed out such as memory leaks，CPU overload，and I/O blocking，and in conjunction with user experience perception，three key optimization strategies were proposed，including system startup time，application startup time，and smoothness of operation. Through experimental evaluation，the effectiveness of these optimization strategies in improving system performance was verified. The research results show that through systematic performance optimization，the performance of Android customized systems can be significantly enhanced，thereby strengthening the user experience.

Keywords：Android custom system；performance optimization；user experience；resource management；system smoothness