云-端融合下的端设备能耗优化

移动应用技术的快速发展让我们能在移动设备上从事越来越复杂的各类应用，然而这一过程中电池技术的停滞不前限制了移动应用，这也是影响用户体验的最重要因素。在电池技术本身的研究之外，软件技术层面的研究也尝试对设备能耗进行优化。传统意义上通过代码优化、漏洞检测和消除等技术实现的能耗优化仅能在一定程度上缓解。云端融合技术安模式和相应技术平台为此问题提供了一个解决方案。

端设备能耗问题

从1996年Palm公司发布具有128K内存、16MHz CPU的Palm Pilot个人数字助理（PDA）产品，仅经过二十年，市场上到处可见运行着Android、iOS和Windows Phone等系统的移动设备已具有和桌面型电脑相当的硬件配置了。移动终端的计算能力增长的成百上千倍，其上的应用软件从当初的简单的文本、数值存取处理逐步演化为各类复杂信息、娱乐和通信等应用。在用户期待以移动终端设备替换传统桌面和膝上型设备之时，发展相对缓慢的电池技术限制了这一趋势的进一步实现。设备电力续航条件的限制使得应用无法以不及开销的方式运行消耗设备电力资源，这在一定程度上影响了应用性能和相应的用户体验，并且，不恰当的设计更会因为过度消耗电能而进一步影响用户体验。

针对这一问题，相关研究从应用开发运行相关的x工程和技术的角度尝试优化应用能耗。例如Ding Li等人的工作[14]从编码的角度出发，提出一些可以降低能耗的编程经验，并且对一些被推荐的编程经验（例如Android开发者网站上给出的一些建议[15]）进行实验验证，为应用开发者提供降低能耗的指导建议。Mario Linares-Vásquez等人的工作[16]将外部设备测出的能耗数据与执行路径对应起来，统计每个API调用的能耗，并提出了避免使用能耗高的API，而尽量使用功能相同的能耗低的API，来降低应用能耗的方法。Android系统自身也在针对应用能耗进行优化。例如Android运行时Dalvik和ART，Dalvik采用即时编译策略，在程序运行时将字节码翻译成机器码来执行，这样运行时开销就增加了，能耗也就增加了；新的运行时ART采用预编译策略，在程序安装时就将字节码翻译成机器码，降低了运行时开销，也就降低了能耗。

部分工作着重优化能耗较高的设备部件的使用。例如针对高能耗的GPS调用，Zhenyun Zhuang等人的工作[12]也提出了相应的应用能耗优化方法。比如利用能耗更低的基于无线网络的位置感知技术来代替GPS调用；利用加速度传感器来判断用户的运动状态，如果用户是静止的，则限制不必要的GPS调用；将来自不同应用的GPS调用请求进行同步以减少GPS的调用次数；调整GPS调用的请求参数，等等。最终达到了降低98%的GPS调用的能耗的效果。针对高能耗的Wi-Fi接入点扫描，Kyu-Han Kim等人的工作[13]也提出了根据用户的运动状态和Wi-Fi接入点的密度来调整Wi-Fi接入点扫描的间隔，最终达到了降低79%的Wi-Fi接入点扫描的次数的效果。屏幕能耗占应用总能耗的一大部分，降低屏幕能耗能在很大程度上降低应用的能耗。而如今许多Android设备都配有OLED屏幕，其能耗不仅受到屏幕亮度，还受到显示的内容的颜色的影响，在屏幕亮度相同的条件下，显示暗一点的颜色，如黑色、灰色等，要比显示亮一点的颜色，如白色、黄色等的能耗更低。南京大学发表的工作[10]就据此对Android系统能耗配置文件中的屏幕能耗模型做出了改进。另外，Mian Dong等人的工作[11]指出，由于许多Web应用的背景色都是白色，这并不是一个节省能耗的做法，于是他们调整Web应用的配色方案，让面积更大的背景色变为黑色，以此来降低Web应用的能耗。当然这种方法不限于Web应用，许多背景色是白色的Android应用都可以针对屏幕能耗做出改进。

应用开发中能耗相关的不当设计称为能耗漏洞或能耗Bug，能耗Bug不会影响应用的功能，也不会引起应用崩溃，只会让应用消耗更多的电量，并且用户往往很难发现。部分研究工作通过静态或动态方式进行检测。典型的能耗Bug例如No-sleep Bug。Android系统为了省电，会在用户无操作一段时间后进入休眠状态，但这经常会影响一些应用的功能。于是Android系统提供了WakeLock机制。WakeLock是一种锁机制，只要系统中有应用申请了WakeLock，系统就无法进入休眠状态，直到WakeLock被释放。然而有些应用申请了WakeLock，却忘记将它释放，或由于某些原因没有执行到释放WakeLock的代码，导致系统永远无法进入休眠状态，这样就产生了一个No-sleep Bug。Abhinav Pathak等人的工作[7]通过数据流分析的方法检测可能导致No-sleep Bug的运行路径，并添加释放WakeLock的代码来进行修复。另一类能耗Bug与传感器不当使用相关，某些Android应用会申请一些能耗较高的传感器资源，例如GPS，但在使用完后忘记将其释放，导致后台不断地获取传感器数据，而这些数据又无任何实际用途，造成能耗漏洞。可以看出，Sensor-related Bug与No-sleep Bug的模式非常类似，南京大学发表的工作[8]同样通过数据流分析的方法进行检测并修复。

能耗Bug不仅可以从应用的字节码来检测，还可以从应用的实际能耗行为来分析。例如Abhijeet Banerjee等人的工作[9]利用外部设备测量应用运行过程中的能耗，如果在应用运行之前和运行完毕后，设备的能耗行为不相似，则可以认为执行路径中存在能耗Bug。虽然这是一个利用外部设备的“离线”方法，但由于其目的是检测能耗Bug，因此并无大碍。

能耗漏洞检测、编程优化等,但这些只在一定程度上缓解了设备的能耗问题。

能耗的主要问题是在部件方面消耗的，因此云端融合的基本思路是希望能让终端应用去使用云端设备的耗电部件（CPU、网络、传感器等），相关工作由此展开

云端融合是解决设备能耗问题的一个重要方式，这一技术方式为解决能耗问题提供了各种可能和无限想象

云端融合的能耗优化技术

云端融合技术在工业界产品中已有较为广泛的应用。2011年亚马逊公司在其推出的Fire平板电脑上安装了Silk浏览器，该浏览器的架构体现了较为典型的云-端融合技术特点。当用户选择以“云模式”运行时，Silk将用户输入的URL直接传递到Amazon的EC2云服务，在EC2云服务端完成该URL相关的HTML、CSS、图片、JavaScript等资源。在此过程中EC2云服务还可为Silk浏览器进行资源优化，例如根据客户端分辨率自动进行所下载图片的处理以适配端设备分辨率，并且EC2云服务中还可实现机器学习算法以预测用户将访问的内容，通过对内容预加载实现用户浏览的加速。此外，Silk还可将部分渲染过程相关计算off-load到Amazon的EC2云服务，包括HTML、CSS和JS等文件的解析和渲染树构造等步骤，进一步提升用户渲染过程的效率。

语音识别是当前应用较为广泛的云-端融合技术的应用场景，各类移动应用集成语音识别功能为用户提供友好的文字输入界面。因语音识别技术一般需要通过神经网络算法进行复杂计算后完成语音识别过程，而移动设备本身不具备进行如此重量级计算的资源，因此用户在移动端设备上进行语音采集后编码压缩为数字格式并通过无线网络传输到云服务器，服务器通过将该语音信号和某一统计模型进行比较，猜测最可能且合理的对应文字内容，并将结果返回。而类似Siri等语音助手，还需要进一步将人工智能、人机交互及智能搜索等技术紧密地联系起来，在云端实现对用户所说的话进行理解，并快速寻找答案以语音的方式回答用户。

以在线流媒体方式从云端获取音乐和电影成为当前内容消费的主要形式。这一云资源应用形态也在被用于计算机游戏行业。NVIDIA公司正在开发的GRID云游戏技术在云端服务器内将3D游戏进行渲染和编码，并以流

方式将结果通过有线或无线网络传输到用户的各类设备。游戏软件的升级无需用户进行下载、打补丁甚至重新安装，游戏软件对底层软硬件的升级要求也无需用户进行相应更新。用户个性配置和进度都在云端保存，这一技术使得用户任何时刻都可以在电视、PC、Mac、平板电脑和手机等各类设备上即时开启游戏，享受“游戏即为服务”（Gaming as a Service, GaaS)所带来的乐趣。

计算迁移（Computation offloading）

设备嫁接（Sensor Offloading）