

基于支持向量机的移动 Web 浏览性能优化研究

高 岭^{1),2)} 任 杰³⁾ 王 海¹⁾ 郑 杰¹⁾ 魏泽玉¹⁾

¹⁾(西北大学信息科学与技术学院 西安 710127)

²⁾(西安工程大学计算机科学学院 西安 710600)

³⁾(陕西师范大学计算机科学学院 西安 710119)

摘 要 随着网页内容和功能的丰富以及用户体验需求的提升,移动 Web 浏览中的计算能耗与日俱增. DVFS (Dynamic Voltage and Frequency Scaling) 技术在同构多核架构中取得较好的节能效果,但在异构多核架构中,由于系统默认的调度策略没有充分利用低功耗核心,导致高性能核心的工作频率和时间往往高于实际需求,浪费大量电能. 而面向异构平台的 Linux HMP (Heterogeneous Multi-Processing) 技术也没有充分利用异构平台的特性,无法对能效进行有效的提升. 针对上述问题,该文面向移动异构平台,提出了一种基于支持向量机的 CPU 配置预测模型. 首先选取 500 个热门网站主页,分析其主页面的架构 (HTML) 及样式 (CSS) 信息,进行特征选择;遍历不同 CPU 配置渲染网页,记录获得最优加载时间、能耗及 EDP 对应 CPU 配置;最后在线下利用支持向量机自主挖掘网页特征同最优配置的内在关系,以此构建移动异构平台的 CPU 资源调度预测模型. 该模型通过分析网页特征,根据不同的优化目标,为渲染引擎分配合适的处理器资源. 实验结果显示,同目前最先进的一种线性回归预测模型相比,该文提出的 CPU 资源调度模型在加载时间、能耗和 EDP 上的性能得到显著提升.

关键词 移动 Web 浏览优化; Web 负载特征; 支持向量机; 异构多核处理器; 资源调度策略

中图法分类号 TP393

DOI号 10.11897/SP.J.1016.2018.02077

Optimize Mobile Web Browsing Based on Support Vector Machine

GAO Ling^{1),2)} REN Jie³⁾ WANG Hai¹⁾ ZHENG Jie¹⁾ WEI Ze-Yu¹⁾

¹⁾(School of Information and Technology, Northwest University, Xi'an 710127)

²⁾(School of Computer Science, Xi'an Polytechnic University, Xi'an 710600)

³⁾(School of Computer Science, Shaanxi Normal University, Xi'an 710119)

Abstract With the growing demand for high-performance mobile browsers and the increasing complexity of websites, the Web browsing consumes more and more energy to satisfy such requirements. The system default governor DVFS (Dynamic voltage and frequency scaling) technology performs well in homogeneous multi-core platform while it cannot make full use of low-power cores in heterogeneous architecture. For example, in order to provide a good user experience, the DVFS prefers to schedule the task on the high-performance cores (big core) with high frequency and takes extra time on activity state which causes much energy waste. Although the Linux HMP (Heterogeneous Multi-processing) scheduler is designed for the heterogeneous architecture, it cannot capture the characteristics of different workloads. The HMP performs better than DVFS on energy efficiency, but it still cannot take advantage of heterogeneous architecture therefore, leaving a huge space to optimize. In response to these problems, this paper proposes a predictive model for CPU configurations based on Support Vector Machine to

收稿日期:2017-02-28;在线出版日期:2017-11-29. 本课题得到国家自然科学基金(61373176,61572401,61672426,61701400)、中央高校基本科研业务专项资金(GK201803063)和中国博士后面上项目(2017M613188)资助. 高 岭,男,1964 年生,博士,教授,中国计算机学会(CCF)高级会员,主要研究领域为网络安全管理和嵌入式网络服务. E-mail: gl@nwwu.edu.cn. 任 杰,男,1988 年生,博士,讲师,中国计算机学会(CCF)会员,主要研究方向为移动计算. 王 海,男,1977 年生,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为服务计算. 郑 杰,男,1985 年生,讲师,主要研究方向为移动通讯. 魏泽玉,女,1993 年生,硕士研究生,主要研究方向为嵌入式系统.

optimize Web browsing on heterogeneous mobile platform. Firstly, it characterizes the web-page workload by conducting the feature selection of the structures (HTML) and styles (CSS) of top 500 popular websites' land-page. Then it schedules the rendering process on the big or little core with different clock frequencies. After that it records the best performing configuration for three different optimization goals (load-time, energy consumption and energy delay product-EDP). Finally, it builds up the predictive models off line by using the SVM (Support Vector Machine) to find the correlations between the selected features and the best configurations of different metrics. Meanwhile, it implements the predictive model as a web browser extension to schedule the rendering engine which is designed to run on the predicted core at specified frequency by analyzing features of web-page. The results show that the SVM-based models make significant improvement on load time, energy consumption and EDP as compared to the state-of-the-art linear regression predictive model.

Keywords mobile web browsing optimization; web workload characterizing; support vector machine; heterogeneous multi-core processor; resource scheduling strategy

1 引言

受益于 Web 技术的飞速发展(HTML5、CSS3、JavaScript)和灵活的开发方式“one size fits all”,基于 Web 的 Hybrid App(Baidu、Facebook)和 Web App(qq、Financial Times)的移动应用市场份额正在迅速增长^①. 为了丰富用户的上网体验,降低网页加载时间,浏览器性能不断提升以应对越来越复杂的网页结构,最终导致浏览过程对资源和电量的大量消耗^[1-2]. 调查显示,73%的移动网民抱怨糟糕的 Web 浏览体验^②,过慢的网页加载速度和高功耗的工作状态不仅影响用户体验,同样限制着浏览器的进一步发展.

网络是影响移动 Web 浏览体验的一个重要因素. 近年来主要通过增加网络带宽,缓存浏览内容,预取网页数据^[3]以及重组浏览器下载和计算顺序^[4]来降低网络对 Web 加载性能的影响. 除此之外, CPU 是提升移动 Web 浏览性能的另一个瓶颈^[5]. 移动浏览器由解析器、渲染引擎(如 Webkit, Gecko)和 JavaScript 引擎组成(如 V8, TraceMonkey),主要工作都依赖 CPU 完成^[6-7]. 现有研究主要通过并行处理 Web 计算任务来提升浏览体验. 例如 Chromium 多进程架构和网页并行加载算法^[8]. 目前,优化浏览器的相关研究有效地提升了 Web 浏览速度,但:(1)只有极少的研究考虑到 Web 浏览过程中的高能耗问题;(2)多数 Web 优化研究是在同构系统上进行;(3)优化目标单一(性能、功耗).

本文针对移动 Web 浏览过程中的能耗问题,基于 big, LITTLE 异构处理器平台,构建高效灵活的 CPU 动态调度模型. 该模型通过支持向量机(Support Vector Machine, SVM)对 500 个热门网页架构(HTML)和样式(CSS)进行线下学习而得. 根据当前网络状态及网页负载情况,利用 big, LITTLE 异构平台内置高性能处理器和低功耗处理器的优势,为 Web 浏览过程动态分配 CPU 资源. 旨在提升用户浏览体验,降低网页加载功耗. 实验结果表明,相比于 Linux HMP 调度策略,本文提出的基于 SVM 的 CPU 调度模型,能大幅降低 Web 浏览能耗,提升浏览性能.

2 相关工作

移动 Web 浏览优化一直是广受关注的研究热点, Meyerovich 等人^[7]通过并行渲染引擎中的布局算法提升浏览速度. Mai 等人^[8]开发的 Adrenaline 原型插件通过并行网页渲染过程提升浏览器性能. 但早期 Web 浏览优化方案很少考虑能耗需求. 不考虑能耗的性能提升,会大幅缩短移动设备的续航时间,同样会影响用户体验. Thiagarajan 等人^[2]对网页渲染过程的能耗进行测量,建议重新组织 JavaScript 文件加载顺序,删除不必要的 CSS 规则. 同时发现不同网页加载能耗及处理时间的差异性源于网页结构和组件的不同. Zhao 等人^[4]分析 3G 无线接

① The 2017 Future of Apps, <http://info.inspirehub.com/the-future-of-web-apps-whitepaper>

② How Loading Time Affects Your Bottom Line, <https://blog.kissmetrics.com/loading-time>

口的工作特点,针对移动浏览器下载网页的高能耗问题,在网页加载过程中重组浏览器的计算顺序,让 3G 无线接口长时间处于低功耗状态. 在异构多核平台中,Bui 等人^[9]针对 Chromium 多进程架构,提出自适应内容绘画技术(ACP)以降低渲染频率. 同时提出应用辅助调度技术,让浏览器在大小核之间调度内部线程,降低能耗. 但该调度技术只是简单的基于绘画速度来调整线程,并没有充分考虑浏览器不同进程的特点和网页负载情况.

鉴于移动平台的多样性,本文提出针对不同平台进行自动构建优化模型的策略. 目前,基于机器学习的预测模型在系统优化方面应用广泛,如预测功耗^[10],程序优化^[11-13],自组织并行策略^[14-15],任务调度,生成基准^[16-17],预测服务质量^[18]以及集中通讯流量以降低能耗^[19-20],但尚未有通过机器学习预测移动 Web 浏览过程中的最优处理器配置的研究.

3 背景介绍

3.1 big, LITTLE 异构多核心架构

ARM 公司开发的 big, LITTLE 异构多核心架构,将高性能核心(big)和低功耗核心(LITTLE)相结合,旨在保证性能的同时,优化任务的执行功耗. 例如,网页加载过程的初始化渲染或游戏场景切换,这类 CPU 密集型任务总是伴随着诸如用户滚动阅读、等待游戏玩家输入等低强度任务. 以及一些轻量级任务,如打字、发送 E-mail. 这些都可以通过对 big, LITTLE 异构核心的合理调度来达到降低功耗,提升性能的目标. 官方报告显示^①,big, LITTLE 在软实时环境中,最高可节能 75%. 另外,可通过对任务的高度并行处理提升 40%的性能. 本文使用的移动异构平台 Odroid XU3,配置了高性能(Cortex-A15)和低功耗(Cortex-A7)CPU 核心. Cortex-A15 拥有 15 到 24 级流水线,运行速度快,功耗较大. Cortex-A7 配置 8 到 10 级别流水线,处理轻量级任务,功耗较低^[21].

3.2 Web 浏览器及工作流程

Chromium 多进程浏览器隶属于 Google Chrome 的开源项目,多进程架构可以避免单个进程漏洞引发的系统崩溃,同时可以利用多核处理器的并行策略提升浏览性能. 如图 1 所示,Chromium 由一个浏览器主进程(Browser process)和多个渲染进程(Renderer)组成,每一个网页标签分配一个独

立的渲染进程. 浏览器进程负责 UI 的运行、标签和插件的管理、控制所有网络的接入及资源的分配,它使用 IPC 管道(Inter-process Communication)同渲染进程进行通讯. 渲染进程通过渲染引擎解析 HTML 文件、渲染和布局可视化组件,工作流程如图 2 所示. Renderer 通过 Browser Process 从网络层获取网页数据后,创建一个 HTML 解析线程(Parser)分析经过浏览器转码后的字符串,经过词法分析,安全处理等,将 HTML 中的标签(tag)转换为文档对象模型(Document Object Model)节点,然后调用 Renderer 构建 DOM 树. 当在解析过程中发现<script>标记时,传统策略是阻塞解析,JavaScript 线程执行脚本代码,直到完成后返回 Parser 继续执行,该方法严重影响浏览性能. 新的策略为防止阻塞产生的延迟,Parser 调用其它线程(预解析器)继续扫描后续代码,判断是否有需要加载的资源,如有则并行进行资源加载. 但对 DOM 树的修改仍然是由主解析器进行操作. 与此同时,Renderer 解析 CSS 文件样式信息,构建渲染树. 渲染树中包含可视化组件的颜色、尺寸等样式信息. 在 DOM 树和渲染树构建完成后,对所有结点进行布局,提供可视结点在屏幕上的坐标信息. 最后遍历渲染树进行画图,在屏幕上根据规则显示每一个可视化组件. 由此可见,复杂的网页渲染过程无疑增加了处理器的计算强度,进而加重了处理器的能耗负担.

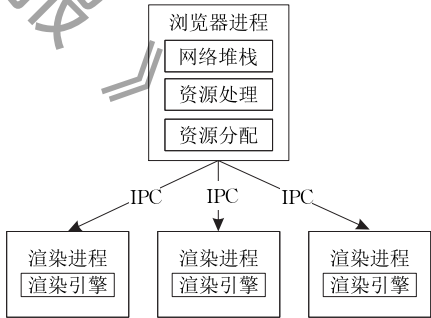


图 1 Chromium 浏览器架构

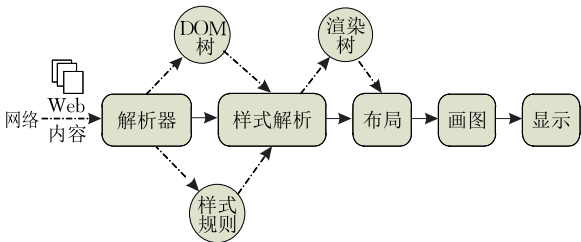


图 2 Chromium 渲染工作流程

① big, LITTLE 处理, <https://www.arm.com/zh/products/processors/technologies/biglittlprocessing>. php

3.3 Linux HMP 调度策略

Linux HMP 调度策略会充分利用大核的高性能优势,将高优先级 CPU 密集型前台任务安排在大核上执行, big, LITTLE 间的迁移模式通过将任务从小核的最高频率转换到大核上实现 DVFS 的扩展. 任务迁移大概需要 $15\mu\text{s}$, DVFS 每隔 50ms 就会对系统和单核性能进行一次评估, 然后花费 $100\mu\text{s}$ 调整频率和电压. 而 CPU 核间任务的迁移时间要小于 DVFS 改变配置的时间, 更比 50ms 的评估时间短. HMP 策略依然沿用 DVFS 策略, 但调度器会意识到存在不同的 CPU 核心, 并将任务分配到大小核上. 尽管 HMP 考虑了整体性能, 将任务动态分配在大小核上, 但实验结果显示^[22], HMP 默认调度策略仍以性能为最高优先级, 大量的任务依然会长时间的以高频运行在大核上, 造成不必要的能耗浪费.

4 支持向量机

4.1 线性支持向量机

支持向量机^[20]用于分类与回归中分析数据的监督式学习模型. 原始最大分隔超平面线性分类算法, 即 SVM 线性分类算法的前身由 Vapnik 于 1963 年提出, 该算法寻找分类超平面使两类分隔间隙最大, 最终转换为凸二次规划问题予以求解.

对训练数据 $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)$, 进行分类学习, 其中 \mathbf{x}_i 是 p 维向量, $y_i = 1/-1$, 正负 1 代表 \mathbf{x}_i 属于两类中的哪一类. SVM 即是寻找可以将 \mathbf{x}_i 明确划分为 $y_i = 1$ 或 $y_i = -1$ 的超平面的学习算法, 超平面定义为距离两个类最远距离的分界线, 定义如等式:

$$\mathbf{w} \cdot \mathbf{x} = 0 \quad (1)$$

\mathbf{w} 是超平面的法向量, 参数 $b/\|\mathbf{w}\|$ 表示超平面沿着法向量 \mathbf{w} 方向的偏移.

硬边界: 如果训练数据线性可分, SVM 将会选择两个平行的超平面将数据分为两类, 两个超平面之间的距离越远越好. 由这两个超平面界定的区域叫做“边界”, 最大边界超平面就是在两个平行超平面中间位置同两个超平面平行的超平面, 定义如下:

$$\mathbf{w} \cdot \mathbf{x} - b = 1 \text{ 和 } \mathbf{w} \cdot \mathbf{x} - b = -1 \quad (2)$$

两个超平面间的距离为 $2/\|\mathbf{w}\|$, 因此最大化超平面的间距只能通过最小化 $\|\mathbf{w}\|$, 为了避免训练点进入“边界”, 对于所有的训练点, 定义如下限制:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i - b &\geq 1, \text{ if } y_i = 1 \\ \mathbf{w} \cdot \mathbf{x}_i - b &\leq -1, \text{ if } y_i = -1 \end{aligned} \quad (3)$$

对于 $1 \leq i \leq n$, 式(3)重新定义如下

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1 \quad (4)$$

寻找最优超平面的过程, 即在式(4)的限制下, 最小化 $\|\mathbf{w}\|$, 其中, $\|\mathbf{w}\|$ 和 b 确定了分类器.

软边界: 对于一些无法严格把点区分开的情况(噪声), 即线性不可分. 为解决这类问题, SVM 引入了铰链损失函数, 让 SVM 能容忍这种错误, 定义如下:

$$\max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i - b)) \quad (5)$$

若满足式(4)即 \mathbf{x} 位处边界两侧, 等式 5 为 0. 对于位处边界内的数据点, 函数值同边界距离成比例. 目标是使式(6)最小:

$$\left[\frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i - b)) \right] + C \|\mathbf{w}\|^2 \quad (6)$$

具体做法就是在目标函数添加错误累加项, 添加系数 C , 控制对错误的容忍度, 并且添加错误容忍度的约束. 如果 C 值够小, 软边界同硬边界相同.

4.2 非线性支持向量机

对于线性不可分情况, 由 Boser 等人^[24]于 1992 年提出在最大分割超平面中应用核函数 $K(x, z) = \langle \varphi(x), \varphi(z) \rangle$. 将原有输入空间映射到其它特征空间, 将非线性问题转化为线性问题, 代替 SVM 线性分类中的点积表示法. 据泛函的有关理论, 函数 $K(x, z)$ 满足 Mercer 条件时它就对应某一变换空间的内积. 常用的核函数有以下四种.

(1) 线性核函数, 主要用于线性可分情况, 具有参数少, 训练速度快的特点:

$$K(x, z) = \mathbf{x}^T \mathbf{z} + c \quad (7)$$

(2) 多项式核函数, 主要用于正交归一化数据, 参数 d 表示映射维度, 当 d 过大时会产生数据过拟合问题:

$$K(x, z) = (\mathbf{a}\mathbf{x}^T \mathbf{z} + c)^d \quad (8)$$

(3) 径向基函数, 具有很强的抗干扰能力, 由于其很强的局部性, 其参数决定了函数作用范围随着参数 γ 的减小而减弱:

$$K(x, z) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2), \gamma > 0 \quad (9)$$

(4) Sigmoid 核函数, 源于神经网络, 采用该核函数时, 支持向量机本质是一种多层感知神经网络:

$$K(x, z) = \tanh(\mathbf{a}\mathbf{x}^T \mathbf{z} + c) \quad (10)$$

5 移动 Web 浏览能耗问题

移动系统通过动态电压频率调整技术降低功

耗. 该动态技术根据工作负载调节 CPU 的运行电压和频率. 但 DVFS 主要应用于同构多核平台, 在异构平台中不能达到很好的节能效果. 而由 ARM 开发的 Linux HMP 依然倾向于将性能放在首位, 并没有充分利用异构平台的大小核优势.

图 3 所示为 Chromium 加载 www. amazon. cn 时渲染进程在 Cortex-A15、浏览器进程在 Cortex-A7 运行时的能耗. 横坐标表示大小核的频率组合, 其中 Cortex-A15 频率范围 0.2~2.0 GHz, Cortex-A7 频率范围 0.2~1.4 GHz, 调频粒度 0.1 GHz, 共 247 种组合(如 b9-l4 表示 Cortex-A15 为 0.9 GHz, Cortex-

A7 为 0.4 GHz). 如图所示, 通过 Linux HMP 加载 Amazon 的能耗为 5.674 W, 而最低能耗配置 b9-l4 耗能 0.8121 W. 图 4~图 6 分别展示了加载 www. baidu. com 和 www. amazon. cn 时, 系统默认调度策略 Linux HMP 的能耗(图 4)、时间(图 5)和 EDP (EDP=能耗×时间, 图 6)以及可达到的最优值. 实验结果显示, 相比于默认调度策略, baidu 和 amazon 的最优配置分别可节能 63%、85%, 加载速度提升为 57%、23%, EDP 优化可达 52%、64%. 由此可知, 当前异构多核平台上的 Linux HMP 策略在 Web 浏览性能上存在很大的提升空间.

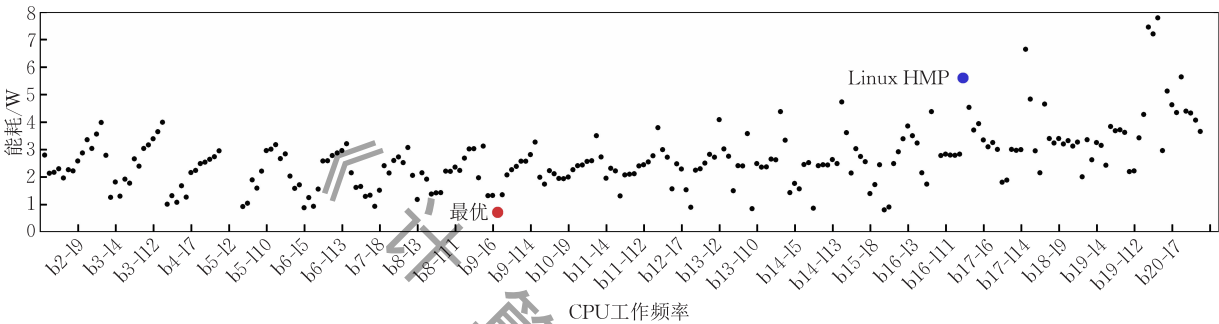


图 3 HMP 和最优配置加载 baidu 和 amazon 的时间对比

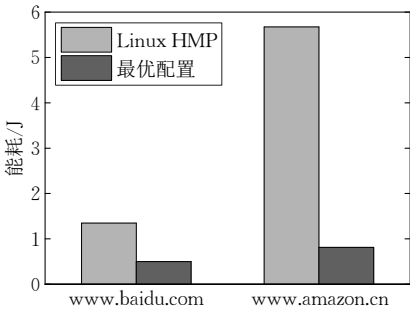


图 4 HMP 和最优配置加载 baidu 和 amazon 的能耗对比

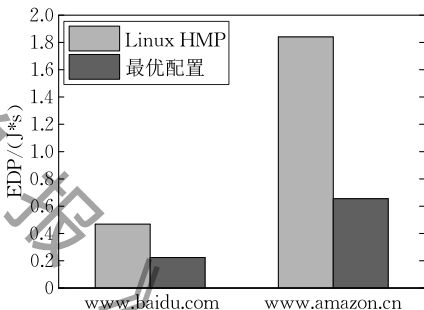


图 6 HMP 和最优配置加载 baidu 和 amazon 的 EDP 对比

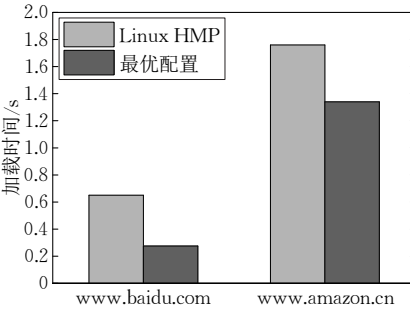


图 5 HMP 和最优配置加载 baidu 和 amazon 的时间对比

另外, 本文使用 amazon 最低能耗配置加载另外 20 个热门网页, 相比于各自网页最优能耗配置, amazon 的最优配置平均浪费能耗 57%(图 7). 说明单一的 CPU 配置不能满足所有网页的能效需求. 因此, 基于移动异构平台的网页浏览过程, 需要一种更行之有效的 CPU 资源调度策略来满足用户对不

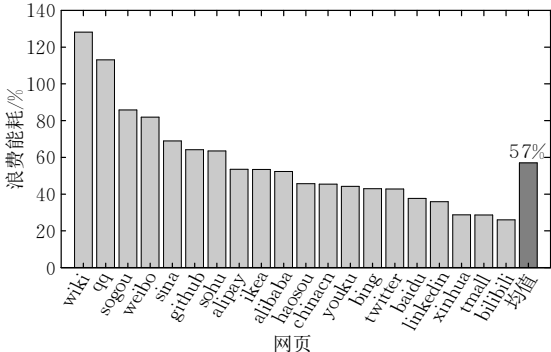


图 7 amazon 最优能耗配置加载 20 个热门网页的能耗浪费情况

6 基于支持向量机的 CPU 调度模型

本文针对移动 Web 浏览提出的 CPU 调度模

型,通过分析不同网页负载(网页架构和网页样式等信息),根据不同优化目标(加载时间、能耗和 EDP),预测满足优化需求的最优配置对网页进行渲染. 工作流程如图 8 所示:(1) 首先,在网页解析期间抽取网页特征;(2) 然后将特征值输入预测器;(3) 输出值为渲染网页的 CPU 核心和运行频率,最后由调度器完成处理器调频及进程调度工作. 本文



图 8 工作流程

6.1 网页特征信息

实验表明,由于网页结构(HTML)和网页样式(CSS)的差异(图 9、图 10 显示了 500 个网页的 DOM 节点数及网页大小分布情况),Web 渲染时间和能耗也大不相同^[5],由此 Web 浏览过程中的最优能耗,加载时间,EDP 的 CPU 配置也各不相同. 如图 8 所示,Web 浏览资源调度优化模型的第一步为选择 928 个特征作为机器学习初始输入特征(表 1). 本文通过自主开发的 Chromium 扩展遍历 500 个网页样本,采集每个网页的特征信息,最终获得如 DOM 树结构(节点数量、树深度和 HTML 标签)决定了网页的复杂度和布局情况;样式信息决定了如何渲染网页元素(字体和颜色等);网页内容越多,渲染时间越长.

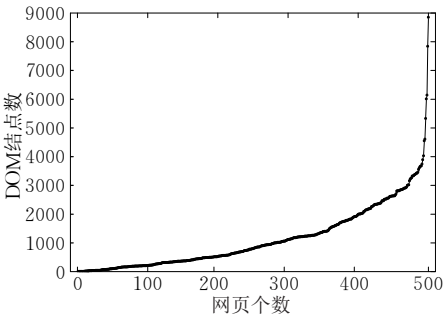


图 9 500 个热门网页的 DOM 节点数

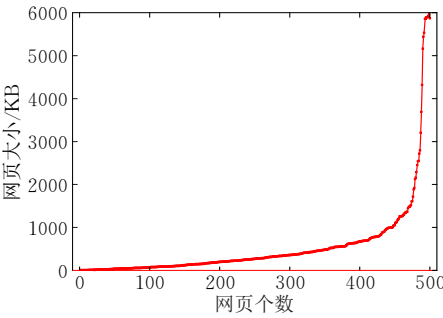


图 10 500 个热门网页的网页大小

选择基于径向函数的支持向量机(Support Vector Machine, SVM)作为最优配置预测器,通过线下学习建立预测模型. 与线性回归、马尔科夫链、K 临近值、决策树等一系列其它建模方法进行实验评估,其中 SVM 的预测精度最高,且 SVM 可以对线性及非线性问题进行建模,由此使用 SVM 作为建模方法.

表 1 网页特征

结构(HTML)	HTML 标签及数量
	HTML 属性及数量
	DOM 节点数
	DOM 树深度
样式(CSS)	CSS 特性及数量
	CSS 规则及数量
	CSS 选择器模式及数量
其它	网页大小

特征提取. 为了获取 DOM 树特征,本文开发的 Chromium 扩展首先遍历 DOM 树获取每一个 DOM 元素的索引,然后通过 Chromium API document.getElementById()采集节点信息. 为了获得 CSS 样式特征,扩展使用 document.styleSheets()抽取 CSS 规则,包含选择器和一些声明对象.

特征选择. 为了训练出更准确的预测器,训练样本数量往往要比特征数量大一个数量级. 鉴于本文训练样本数量小于 500 个网页,需要删除部分特征(如影响小及相关性高的特征)以对特征进行筛选. 本文通过以下三步对 928 个特征做进一步的筛选. 第一步,删除不常用特征,例如 HTML 标签 DFN, EM, 第二步,删除所有网页共有且数量相同的特征,它们对渲染时间或者处理器选择的差异性上影响较小,如 HTML 标签或属性(<def><body>). 第三步,删除相关度高的特征,为了获取属性间的相关性,构建相关系数矩阵(correlation coefficient matrix),相关值在-1 到+1 之间,相关值越靠近+1/-1 则相关性越强,本文删除相关度绝对值大于 0.75 的属性. 由此筛选出 73 个特征作为机器学习的输入特征,如表 2 所示.

特征归一化处理. 在将特征值输入到学习算法之前,为了均衡不同特征对分类结果的影响程度,通过线性归一法对特征值进行归一化处理(式(11)).

表 2 73 个网页特征

# HTML 标签	a, b, br, button, div, h1, h2, h3, h4, i, iframe, li, link, meta, nav, img, noscript, p, script, section, span, style, table, tbody
# HTML 属性	alt, async, border, charset, class, height, content, href, media, method, onclick, placeholder, property, rel, role, style, target, type, value, background, cellpadding, cellspacing, width, xmlns, src
# 样式选择器	class, descendant, element, id
# 样式规则	background. attachment/clip/color/image, background. repeat. x/y, font. family/size/weight, color, display, float, background. size, background. border. image. repeat/slice/source/width,
其它信息	DOM tree depth, # DOM nodes, # style rules, size of the webpage (Kilobytes)

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{11}$$

然后将处理好的特征值输入 SVM 分类算法构建预测模型。在测试集进行分类预测时,也需先提取特征值进行归一化处理,然后输入分类模型进行分类预测。

6.2 SVM 建模

生成训练数据:本文使用 400 个网页训练 SVM 模型,这些网页来自 alexa 提供的全球最热门的网站。网站类型包含了电商、搜索引擎、新闻等。在训练前,将 500 个网页下载并存储在本地服务器和 Odroid XU3 的 RAM 中。样本类别标签通过遍历 CPU 配置,分别进行网页加载,采集加载时间和功耗信息,标记对应网页的最优能耗配置,最小延迟配置和最优 EDP 配置。不同优化目标的最优配置如表 3 所示(加粗体为渲染进程所在核心及运行频率)。

表 3 不同优化目标的最优配置

能耗		加载时间		EDP	
A15	A7	A15	A7	A15	A7
0.8	0.6	1.6	1.2	1.3	0.4
0.9	0.6	1.7	1.2	1.4	0.4
1.0	0.6	1.8	1.2	1.5	0.4
0.4	1.1	1.9	1.2	0.6	1.2
0.4	1.2	—	—	0.6	1.3
0.4	1.3	—	—	0.6	1.4

构建模型:特征值和最优配置组对输入监督学习算法。学习算法会发现二者之间的联系并输出 SVM 模型。由于本节有三个不同的优化目标,因此针对不同的优化目标分别构建 SVM 模型。训练过程在线下完成,一次部署,训练模型及部署需要一周的时间。

SVM 预测模型工作流程如图 11 所示,步骤如下:

- (1) 对 400 个训练样本进行特征提取;
- (2) 遍历 CPU 配置,标记每个网页的最优配置(最优能耗配置、加载时间配置和 EDP 配置);

- (3) 将步骤(1),(2)所得信息输入 SVM-RBF 分类算法进行学习;
- (4) 基于不同优化目标(能耗、加载时间和 EDP)构建预测模型。至此,线下学习过程结束;
- (5) 分别对另外 100 个测试样本进行特征提取,输入步骤(4)的预测模型;
- (6) 模型输出分类结果,即预测最优配置;
- (7) 将预测的 CPU 配置输入 CPU 调度器中(调度器为系统后台服务进程),调度进程,加载网页。

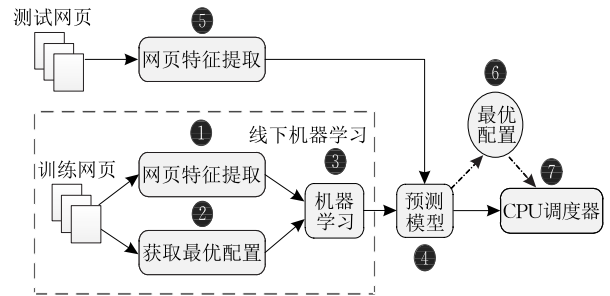


图 11 线下构建 SVM 分类模型及线上基于网页负载的 CPU 调度流程图

6.3 实验对比 SVM 和决策树预测精确度

本文分别使用 SVM 和决策树两种分类算法构建最优能耗配置预测模型,对比两种模型预测准确度。首先利用 R 的 rpart 包分析 400 个网页特征和分类标签,自动建立决策树,另外 100 个网页作为测试样本,实验结果显示,分类准确率仅为 42.7%。为了避免过拟合问题,进一步对决策树进行裁剪,此时分类准确率降为 39%。

SVM 是全局分类算法,通常会使用所有属性构建非重叠分区。针对本文特征数量较少,且训练样本量一般的情况,选择径向基函数 RBF(Radio Basis Function)作为 SVM 核函数。惩罚因子 C 和核函数宽度 γ 是影响 SVM-RBF 模型分类准确率的两个重要参数。C 指定错分数据的代价, γ 是 RBF 函数特有的内核参数,小 γ 值会产生低偏差高方差的分类结果,大 γ 值会产生高偏差低方差的分类结果。本文使用 R 的 e1071 包构建 SVM 模型,通过内置 SVM.tune() 函数即网格搜索法确定分类最佳的 C 和 γ 参数值。实验显示,最优能耗配置预测模型分类准确率可达 88.9%(C=100, $\gamma=0.01$)。最快加载时间和最优 EDP 分类器的分类准确度分别为 88% 和 85%。实验结果显示,对于没有给出准确预测结果的网页,其性能和最优性能相差不大,相比于 Linux HMP,错分网页在加载时间(14%),能耗(11%)以及 EDP(36%)三个度量标准下依然有所提升。

6.4 模型部署

图 12 显示了模型的工作流程,在解析阶段(占总渲染时间的 1%^[7]),扩展首先抽取并归一化特征值,利用 SVM 分类器预测最优处理器配置.然后将预测值传输到调度器进行调频和任务调度.抽取特征值,预测以及调整处理器频率总耗时小于 20 ms,在接下来的试验中都将其包含在内.随着 DOM 树的构建过程,Web 特征值可能发生变化,进而影响预测结果.为了确保本文的方法可以适应这样的变化,当 DOM 树节点数目变化超过 30%时,重新抽取特征值,触发再预测和配置动作,实验结果显示,这样的情况鲜有发生.以 baidu 为例,在现实生活中当用户的手机处于低电量状态时,同时用户需要搜索关键信息,这样的使用场景时常发生.此时应基于最优能耗的优化目标,预测处理器配置.首先从网页中获取表 2 列出的 73 个特征值信息,并进行归一化处理,表 4 列出了 baidu 部分非 0 特征值,同时显示了归一化前后的特征值的变化.然后将处理后的特征值输入线下训练好的 SVM 模型中,根据输入值,模型输出最优能耗配置信息 $\langle A15, 0.9-0.4 \rangle$,表明在以能耗为优化目标时,加载 baidu 的最佳配置是渲染进程在 A15 处理器上以 900 MHz 运行, A7 以 400 MHz 运行.最后将这个预测值传输到运行时调

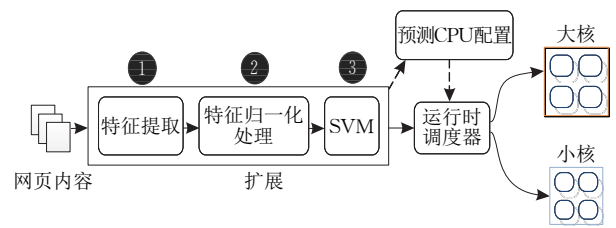


图 12 SVM 模型运行时预测及处理器调度流程

表 4 baidu 部分非 0 特征

特征	初始值	归一化值
# DOM 节点数	754	0.084
树深度	13	0.285
规则数	645	0.063
网页大小	2448	0.091
# div	131	0.026
# h4	28	0.067
# href	148	0.074
# src	36	0.053
# li	52	0.031
# link	10	0.040
# script	3	0.015
# background. color	218	0.058
# class	995	0.045
# descendant	4454	0.0168
# id	4	0.007

度器中,对硬件平台进行相应的配置.实验结果显示,相比于 Linux HMP 调度器,SVM 模型可以节能 58%.

7 实验评估

7.1 实验平台

本文实验平台是基于 big. LITTLE 异构处理器的 Odroid XU3 移动开发板.该处理器架构在移动终端市场已有大量应用(例如, Galaxy S4),具有一定的通用性.相关配置参数如表 5 所示.开发板运行 Ubuntu 14.04 LTS 操作系统.此外,Odroid XU3 内置功耗传感器,实时监控处理器的功耗,通过自主开发的功耗采集程序在后台实时获取功耗信息,采样间隔 0.1 s.交叉编译 Chromium 浏览器(版本: 48.0)使之在 ARM 开发板上运行.开发 Chromium 扩展采集网页特征信息.

表 5 Odroid XU3 配置

	型号	时钟频率/GHz	核心数
大核	Cortex-A15	0.2~2.0	4
小核	Cortex-A7	0.2~1.4	4
GPU	Mali-T628	533	8

7.2 性能评估及开销

性能评估:本文对每一个网页在不同配置下的运行状况进行采样分析,多次运行,计算几何平均值.运行次数取决于采样结果是否在 95%的置信区间内,上下置信界限相差 5%以内.并修改 Chromium 渲染引擎以记录加载时间.实验数据去除了浏览器启动及关闭所需时间.同时,自主开发的轻量级脚本程序,每秒从能耗传感器中采样 10 次.

模型开销:图 13 显示了模型运行过程中的开销分布情况,SVM 预测模型的开销甚小,在整个渲染过程中占时小于 1%(20 ms).其中任务迁移开销最大(将渲染进程从一个核迁移到另外一个核上运行),

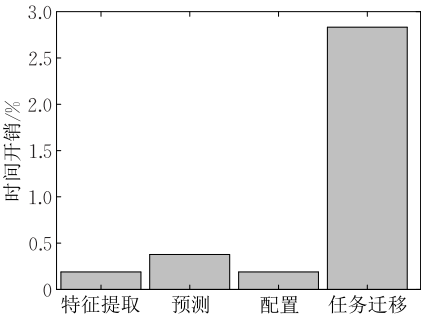


图 13 SVM 模型预测及调度开销

平均耗时 15 ms, 包含初始化硬件环境等一系列工作, 例如缓存预热、特征提取、结果预测、设置处理器频率总耗时低于 5 ms, 开销可忽略不计。

7.3 网页分布

图 14~16 分别显示了不同优化目标下, 网页渲染工作在各自最优处理器配置上的分布情况。当加载时间为优化目标时, 如表 3 所示, 全部网页都选择 A15 核心进行网页渲染工作(最低频率 1.6 GHz)。进一步分析实验结果, 近半网页在 A15 处理器以 1.9 GHz 的工作频率运行, 剩余网页分布在 A15 的 1.6~1.8 GHz 间, 由于 CPU 热保护机制(CPU throttling), 一些网页在较低频率下进行渲染表现更佳。图 15 显示了能耗最优的处理器配置分布情况, 结构简单的网页(占比 30%)受益于低功耗的 A7 小核。对于选择 A15 进行渲染的网页, 其运行频率也远低于加载时间最优配置运行频率。对于图 16 显示的 EDP 来说, 一些

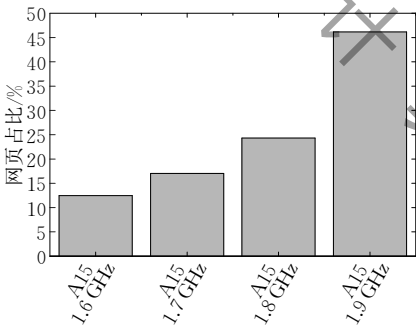


图 14 最优加载时间处理器配置分布

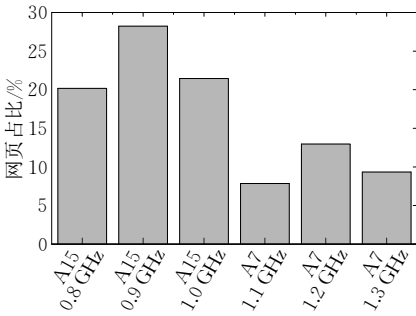


图 15 最低能耗处理器配置分布

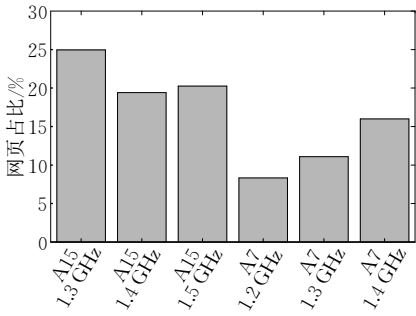


图 16 最优 EDP 处理器配置分布

网页在小核 A7 上表现更优, 运行频率倾向于中间值, 这是因为 EDP 是对能耗和加载时间的权衡。

7.4 SVM 模型同 WS 的实验比较

本文将 SVM 模型同 Zhu 等人^[5]的研究工作进行比较, Zhu 等人提出的 WS(Web-aware Scheduling mechanism)利用线性回归模型预测 Web 加载时间和能耗, 由此确定运行渲染进程的配置。图 17~图 19 显示了 500 个网页在不同优化目标下 SVM 模型和 WS 相对于 Linux HMP 的优化程度。基于不同优化目标, 不同网页的性能提升各不相同, 其中 min-max 条表明不同网页可以获得的性能提升范围。

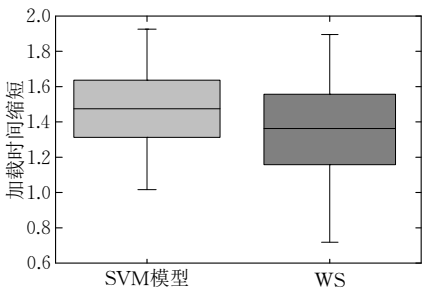


图 17 SVM 模型与 WS 加载时间优化程度对比

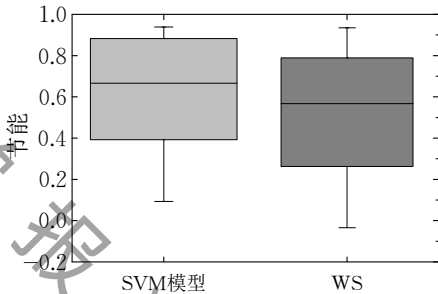


图 18 SVM 模型与 WS 节能对比

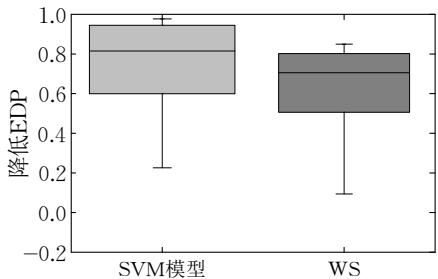


图 19 SVM 模型与 WS 的 EDP 优化程度对比

加载时间: 图 17 显示了当最快加载速度为优化目标时, SVM 模型和 WS 的性能提升对比。由图可知, 相比于 Linux HMP, WS 加载速度平均提升 1.34x, 但由于预测错误, 会大大降低某些网页的加载速度(1.26x)。相比而言, SVM 模型表现更优, 最高可提升加载速度 1.92x, 平均提升加载速度 1.45x。

能耗: 图 18 显示了以能耗为优化目标时两种方

法的性能对比图. 在该场景下, 两种自适应策略 (SVM 模型, WS) 通过频率的调整都能达到很好的节能效果. 结果显示, 相比于 Linux HMP, WS 平均节能 57.6% (最高可节能 85%). 但本文的 SVM 预测模型表现更佳, 平均节能 63.5% (最高可节能 93%). 更重要的是, 由于 SVM 模型能更好的分析网页负载特点, 将渲染工作移植到合适的处理器核心以合适的频率运行, 所以在所有测试网页中, SVM 预测模型相比于 Linux HMP 渲染网页所需能耗更低. 在一些网页测试中, 由于 WS 将本该在小核上运行的进程迁移到了大核上, 这样的错分会导致能耗过高的问题, 同时影响 WS 的性能表现.

EDP: 图 19 显示了当最小 EDP 值为优化目标时不同策略的表现, 即在不显著延长加载时间的情况下, 降低能耗. 相比于 Linux HMP 调度器, 两种自适应策略都表现出了更高的性能. WS 平均降低 EDP 69% (最高可达 84%). SVM 模型平均降低 EDP 81% (最高可达 95%). 相比于 WS, SVM 模型平均降低 38% 的 EDP 值.

由此可知, 两种自适应优化策略都有效提升了移动 Web 浏览时性能, 但相比于本文提出的 SVM 预测模型, 线性回归模型 WS 表现欠佳, 主要原因是: (1) 对于即将加载的网页, 当现有配置无法满足预测结果时, WS 一律调度大核以最高频率加载网页. 该方法在一些网络环境中, 会严重增加电池负担; (2) 线性回归模型仅可以发现网页负载特征同处理器配置间的线性关系, 会大大降低对一些网页配置的预测精度. 本文提出的方法通过动态调度系统所有处理器核心, 以及针对线性和非线性行为进行建模来解决上述问题.

8 结 论

Web 浏览已成为智能移动用户日常活动, 当前浏览器主要通过系统默认调度策略为网页渲染工作分配资源. 由于系统往往不了解单个网页的负载情况和用户需求, 无法分配合适的处理器资源, 导致能耗和性能的损失. 目前, 国内外鲜有针对移动 Web 浏览的优化研究. 本文面向 ARM big, LITTLE 移动多核异构平台, 提出了一种基于支持向量机的最优 CPU 配置预测模型. 该模型的核心是通过机器学习算法, 根据当前网页负载和不同优化目标 (能耗, 加载时间, EDP) 预测渲染引擎的处理器配置 <核心, 运行频率>, 达到提升移动 Web 浏览性能

的目标. 实验结果显示, 相比于系统默认的 Linux HMP, 以及目前最先进的线性回归预测模型 (WS), 本文提出的 SVM 模型在能耗、加载时间和 EDP 三个度量指标中表现最优.

致 谢 感谢陕西省新型网络安全保障与服务工程实验室的大力支持!

参 考 文 献

- [1] D'Ambrosio S, De Pasquale S, Iannone G, et al. Energy consumption and privacy in mobile Web browsing: Individual issues and connected solutions. *Sustainable Computing: Informatics and Systems*, 2016, 11: 63-79
- [2] Thiagarajan N, Aggarwal G, Nicoara A, et al. Who killed my battery?: Analyzing mobile browser energy consumption// *Proceedings of the 21st International Conference on World Wide Web*. Lyon, France, 2012: 41-50
- [3] Wang Z, Lin F X, Zhong L, et al. How far can client-only solutions go for mobile browser speed?// *Proceedings of the 21st International Conference on World Wide Web*. Lyon, France, 2012: 31-40
- [4] Zhao B, Hu W, Zheng Q, et al. Energy-aware web browsing on smartphones. *IEEE Transactions on Parallel and Distributed Systems*, 2015, 26(3): 761-774
- [5] Zhu Y, Srikanth A, Leng J, et al. Exploiting webpage characteristics for energy-efficient mobile Web browsing. *IEEE Computer Architecture Letters*, 2014, 13(1): 33-36
- [6] Badea C, Haghighat M R, Nicolau A, et al. Towards parallelizing the layout engine of firefox// *Proceedings of the 2nd USENIX Conference on Hot Topics in Parallelism*. California, USA, 2010: 1
- [7] Meyerovich L A, Bodik R. Fast and parallel webpage layout// *Proceedings of the 19th International Conference on World Wide Web*. North Carolina, USA, 2010: 711-720
- [8] Mai H, Tang S, King S T, et al. A case for parallelizing web pages// *Proceedings of the Usenix Conference on Hot Topics in Parallelism*. California, USA, 2012: 2
- [9] Bui D H, Liu Y, Kim H, et al. Rethinking energy-performance trade-off in mobile web page loading// *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. Paris, France, 2015: 14-26
- [10] Shye A, Scholbrock B, Memik G. Into the wild: Studying real user activity patterns to guide power optimizations for mobile architectures// *Proceedings of the IEEE/ACM International Symposium on Microarchitecture*. New York, USA, 2009: 168-178
- [11] Wang Z, O'Boyle M F P. Mapping parallelism to multi-cores: A machine learning based approach// *Proceedings of the ACM Sigplan Symposium on Principles and Practice of Parallel Programming*. North Carolina, USA, 2009: 75-84

- [12] Wang Z, O'Boyle M F P. Partitioning streaming parallelism for multi-cores: A machine learning based approach// Proceedings of the IEEE International Conference on Parallel Architectures and Compilation Techniques (PACT). Vienna, Austria, 2010; 307-318
- [13] Grewe D, Wang Z, O'Boyle M F P. Portable mapping of data parallel programs to OpenCL for heterogeneous systems// Proceedings of the IEEE/ACM International Symposium on Code Generation and Optimization (CGO). Shenzhen, China, 2013; 1-10
- [14] Tournavitis G, Wang Z, Franke B, et al. Towards a holistic approach to auto-parallelization: Integrating profile-driven parallelism detection and machine-learning based mapping. ACM SIGPLAN Notices, 2009, 44(6): 177-187
- [15] Wang Z, Tournavitis G, Franke B, et al. Integrating profile-driven parallelism detection and machine-learning-based mapping. ACM Transactions on Architecture & Code Optimization, 2014, 11(1): 2
- [16] Grewe D, Wang Z, O'Boyle M F P. A workload-aware mapping approach for data-parallel programs// Proceedings of the 6th International Conference on High Performance and Embedded Architectures and Compilers. Heraklion, Greece, 2011; 117-126
- [17] Grewe D, Wang Z, O'Boyle M F P. OpenCL task partitioning in the presence of GPU contention// Proceedings of the International Workshop on Languages and Compilers for Parallel Computing. California, USA, 2013; 87-101
- [18] Berral J L, Gavalda R, Torres J. Adaptive scheduling on power-aware managed data-centers using machine learning// Proceedings of the IEEE/ACM International Conference on Grid Computing (GRID). Lyon, France, 2011; 66-73
- [19] Kan E Y Y, Chan W K, Tse T H. EClass: An execution classification approach to improving the energy-efficiency of software via machine learning. Journal of Systems & Software, 2012, 85(4): 960-973
- [20] Tang Z, Guo S, Li P, et al. Energy-efficient transmission scheduling in mobile phones using machine learning and participatory sensing. IEEE Transactions on Vehicular Technology, 2015, 64(7): 3167-3176
- [21] Kamdar S, Kamdar N. Big. LITTLE architecture: Heterogeneous multicore processing. International Journal of Computer Applications, 2015, 119(1): 35-38
- [22] Zhu Y, Reddi V J. High-performance and energy-efficient mobile web browsing on big/little systems// Proceedings of the IEEE 19th International Symposium on High Performance Computer Architecture (HPCA2013). Shenzhen, China, 2013; 13-24
- [23] Cortes C, Vapnik V. Support-vector networks. Machine Learning, 1995, 20(3): 273-297
- [24] Boser B E, Guyon I M, Vapnik V N. A training algorithm for optimal margin classifiers// Proceedings of the 5th Annual Workshop on Computational Learning Theory. Pennsylvania, USA, 1992; 144-152



GAO Ling, born in 1964, Ph. D., professor. His research interests include network management and embedded system.

REN Jie, born in 1988, Ph. D., lecturer. His research interest is mobile computing.

WANG Hai, born in 1977, Ph. D., associate professor. His research interest is Web service.

ZHENG Jie, born in 1985, Ph. D., lecturer. His research interest is mobile communication.

WEI Ze-Yu, born in 1993, M.S. candidate. Her research interest is embedded system.

Background

Modern mobile browsers can access all websites instantly on a range of devices, so that many mobile users prefer to choose browser to go shopping, get information and so on. Nowadays, Web browsing is an activity that billions of mobile users perform on a daily basis.

On the one hand, the high-quality wireless communication technology and high-performance CPU provide a good user experience and a good platform for the development of mobile browsers. On the other hand, the increasing traffic load and unreasonable resources scheduling of mobile browsers are

undoubtedly increasing the burden of mobile battery as well as hindering the development of the mobile network service market.

Because of the huge markets share of mobile network, the mobile browsers' design and optimization are popular research topics in academic and industry area. At present, the current mobile web browsers rely on the operating system to exploit the underlying architecture, which has no knowledge of the individual web workload and often leads to poor energy efficiency.

This paper describes how to render a webpage and the infrastructure of a modern mobile browser: Chromium. Firstly, we characterize the webpage workload by the webpages' structure (HTML) and the style information (CSS). Secondly, we schedule the renderer on the big or little core with different frequency and record the best loadtime, energy consumption and EDP. Finally, we use the Support Vector Machine to find the correlation between the features and best configuration for different metrics. To recap, we develop a machine learning based approach on the heterogeneous multi-process platform to predict which core with what frequency to run the render process, which is based on different user requirements (load time, energy consumption and EDP value). So we can optimize the web browsing and improve user experience by different personalized optimized models.

We compare two adaptive scheduling approaches (WS

and SVM model) over the system default strategy Linux HMP on a big, LITTLE ARM system. Our preliminary experience with the model is encouraging. The proposed SVM models perform better than WS for different metrics. It achieves over 45% improvement over the Linux default policy across three evaluation metrics: load time, energy consumption and the energy delay product. It consistently outperforms a state-of-the-art webpage-aware scheduling mechanism which predicts the energy consumption and loadtime by linear regression. In future work, we intend to explore further refinement of prediction accuracy, and also to exploit the network delay to perform optimization as well.

The work was supported by the National Science Foundation Grant of China (Nos.61373176, 61572401, 61672426 and 61701400), and the Post-Doctoral Foundation of China (No. 2017M613188).

