دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده‌ی مهندسی کامپیوتر و فناوری اطلاعات

بهینه‌سازی مصرف انرژیِ کامپیوترهای همراه با توجه به رفتار کاربر

نگارنده: سینا مهدی پور

sinamahdipour@aut.ac.ir

استاد راهنما: سعید شیری

shiry@aut.ac.ir

زمستان ۱۳۹۵

چکیده

تلفن‌های همراه امروزی دارای قطعات سخت افزاری بسیار قدرتمند و برنامه‌های بسیار پیچیده‌ای هستند که با استفاده از این قطعات، امکانات جذابی را در اختیار کاربران خود قرار داده و به جزیی جدا نشدنی از زندگی ما تبدیل شده‌اند. این پیشرفت و پیچیدگی بی‌هزینه نبوده و بزرگ‌تر شدن نمایشگرها در کنار افزایش قابل توجه تعداد پیکسل‌ها، افزایش تعداد و انواع حسگرها و درگاه‌های ارتباطی مثل Wi-Fi، Bluetooth، UMTS و LTE و همین‌طور وجود برنامه‌های قدرتمند و سنگین، همگی به مصرف بسیار بیشتر انرژی توسط گوشی‌های موبایل ختم شده‌اند. در چنین شرایطی ابداع روش‌های جدید مدیریت و بهینه‌سازی مصرف انرژی بسیار کاربردی و مفید خواهد بود.

هرچند که برای بدست آوردن حداکثر میزان بهینه‌سازی به روش‌ها و راهکارهایی در هر دو سطح سخت افزاری و نرم افزاری نیاز است اما در این مقاله، منظور فوق به وسیله‌ی روشی نرم افزاری به دست خواهد آمد. اکثر روش‌های موجود ذخیره‌ی انرژی توجهی به الگوی استفاده‌ی هر کاربر و همین‌طور تغییر نیازهای یک برنامه ندارند؛ اما هدف ما استفاده از همین اطلاعات برای رسیدن به نتیجه‌ی مطلوب است. به کارگیری حجم وسیع داده‌های جمع‌آوری شده از حسگرها می‌تواند در تشخیص تغییر محتوای کاربر، فهم عادات وی و تشخیص نیازهای برنامه‌ها موثر باشد. استفاده‌ی درست و هوشمندانه از این اطلاعات می‌تواند منجر به پیشرفت خوبی در کاهش مصرف انرژی شود.

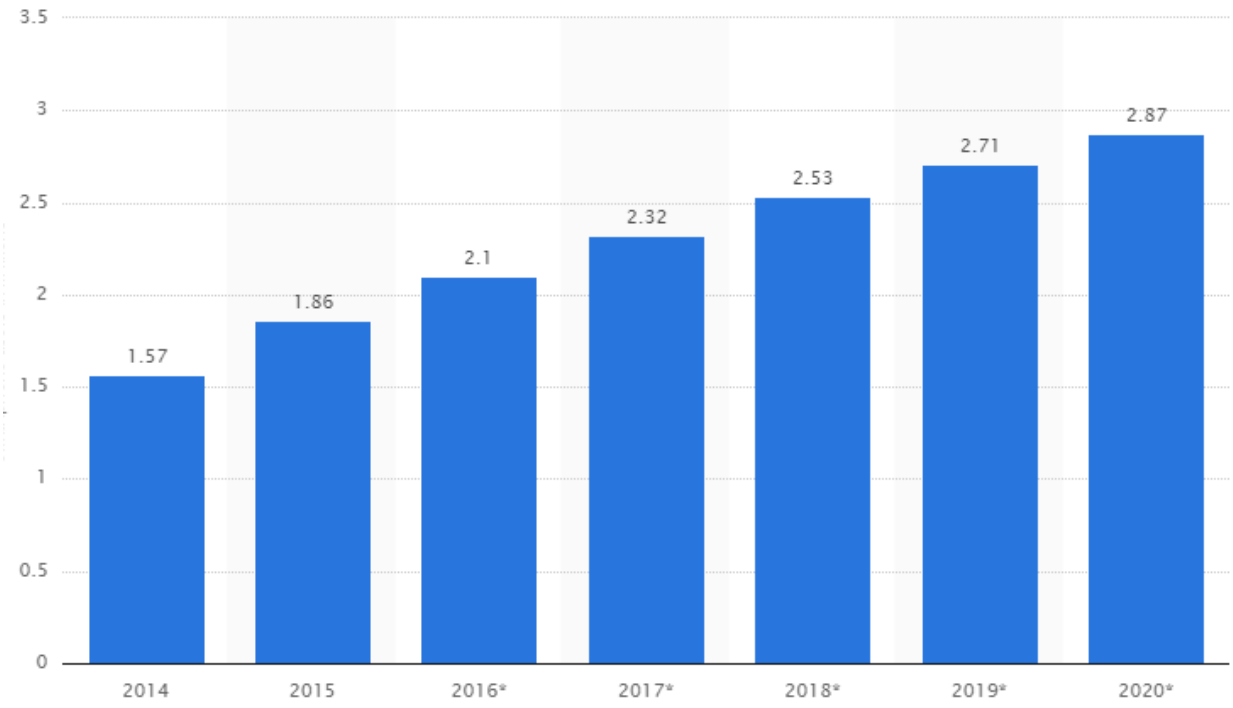
در این مقاله ابزاری برای تحلیل و بررسی تعاملات برنامه‌ها و کاربر برای فهم چگونگی پراکندگی و فراوانی استفاده‌ی کاربر از قطعات سخت افزاری در جهت بهینه‌سازی مصرف انرژی، ارائه می‌شود. برای این منظور از روش‌های یادگیری ماشین (machine learning) جهت تشخیص و دسته بندی اطلاعات مربوط به رفتار و عادات کاربر استفاده می‌شود. با این ابزار نرم افزاری برای مدیریت قطعاتی که مصرف انرژی قابل توجهی دارند توسعه داده خواهد شد که قابلیت پیش بینی میزان استفاده از برنامه‌ها در آینده را هم دارد. در نتیجه می‌توان فرکانس کاری پردازنده، میزان روشنایی صفحه، اتصالات بی‌سیم (مثل Wi-Fi، LTE و غیره) و میزان صدای گوشی را با در نظر گرفتن نیازهای حال و آینده‌ی کاربر و برنامه‌ها کنترل کرده و به نتایج بهینه رسید.

کلمات کلیدی: گوشی هوشمند، پیش بینی ترتیب اجرای برنامه‌ها، الگویابی، توان مصرفی، یادگیری ماشین، محتوای دستگاه، مدیریت انرژی سیستم عامل، حسگرها

فهرست

۱ مقدمه

گوشی‌های تلفن همراه و دستگاه‌های ارتباطی امروزه در زندگی همه‌ی ما به بخشی جدا نشدنی و بسیار پر اهمیت تبدیل شده‌اند. گوش کردن به موسیقی، تماشای یک مسابقه‌ی فوتبال، پیدا کردن آدرس‌ها، گشت و گذار در اینترنت، مطالعه و برقراری ارتباط با دنیای مجازی و دوستان و آشنایان تنها بخش کوچکی از موارد استفاده‌ی ما از تلفن‌های همراه هستند که زندگی ما را به آن‌ها وابسته کرده‌اند. در چند سال گذشته و با ارائه و پیشرفت روزافزون گوشی‌های هوشمند و ساخت نرم افزارهای بسیار کاربردی توسط برنامه‌نویسان، این وابستگی به شدت افزایش نیز یافته است. در جوامع مدرن امروزی بسیاری از اشخاص بیش از یکی از این دستگاه‌های الکترونیکی (گوشی تلفن همراه، رایانه‌های قابل حمل، تبلت، فبلت و غیره) را داشته و در طول روز از آن‌ها استفاده می‌کند. در نتیجه‌ی مطالعاتی که انجام شده، در پایان سال ۲۰۱۳ میلادی ۶ درصد از جمعیت جهان دارای یک تبلت، ۲۰ درصد آن‌ها دارای یک رایانه‌ی قابل حمل، و ۲۲ درصد از آن‌ها یک گوشی هوشمند تلفن همراه داشته‌اند[۱]. تعداد کاربران گوشی‌های هوشمند برای مثال در سال ۲۰۱۵ به ۱٫۸۶ میلیارد نفر رسیده است. طبق پیش‌بینی ۶۵ درصد جمعیت ایالات متحده‌ی امریکا در سال ۲۰۱۷ دارای یک گوشی تلفن هوشمند خواهند بود[۲].



شکل ۱ نمودار تعداد کاربران گوشی‌های هوشمند در سال‌های ۲۰۱۴ تا ۲۰۲۰ (پیش بینی) [۲]

پیشرفت‌های چند سال اخیر در دنیای گوشی‌های همراه بسیار چشم‌گیر بوده و اضافه شدن گستره‌ی وسیعی از تجهیزات سخت افزاری چون انواع حسگرها، منابع محاسباتی و حافظه‌های گوناگون -که خود منشاء زاده شدن سرویس‌ها و نرم افزارهای قدرتمندی مانند برنامه‌هایی که از موقعیت جغرافیایی شما استفاده می‌کنند یا شبکه‌های اجتماعی که امکاناتی گسترده را در اختیار شما قرار می‌دهند، هستند- همگی باعث دامن زدن به مسئله‌ی مصرف انرژی توسط این دستگاه‌ها شده‌اند. نسل‌های آینده‌ی این دستگاه‌های قابل حمل بیش از پیش به سخت افزارهای جدید و قدرتمند تجهیز خواهند شد. وجود چندین هسته‌ی پردازشی، پردازنده‌های گرافیکی، حافظه‌ی بزرگ، حافظه‌های نهان و تعداد زیادی از راه‌های ورودی/خروجی و پروتکل‌ها و امکانات ارتباطی بخشی از این مواردند. برای مثال گوشی تلفن همراه Galaxy S6 شرکت سامسونگ که در سال ۲۰۱۵ به بازار عرضه شد نسبت به نخستین گوشی این خانواده در سال ۲۰۱۰، دارای سه برابر حسگر بیشتر و هشت برابر هسته‌ی پردازشی بیشتر است. پس از طرفی هر روزه بر تعداد و قدرت سخت افزارهای به کار گرفته شده افزوده می‌شود و از طرف دیگر برنامه‌ها سنگین‌تر شده و منابع بیشتری را در اختیار خود می‌گیرند.

امروزه تمام گوشی‌های هوشمند برای تامین قابلیت حمل از باتری استفاده می‌کنند. باتری استفاده شده در اکثر این گوشی‌ها از نوع لیتیوم یون است که در مقایسه با سایر انواع باتری‌ها نسبت به فضایی که اشغال می‌کنند بازده‌ی بیشتری دارند؛ اما متاسفانه تکنولوژی ساخت باتری‌ها مانند سایر قطعات سخت افزاری پیشرفت نکرده است و در استفاده معمولی از آن‌ها مجبوریم بیش از یک بار در روز آن‌ها را شارژ کنیم. مصرف انرژی دستگاه را می‌توان به روش دستی و با مدیریت قطعات سخت افزاری مثل GPS، 3G، WiFi و غیره با خاموش و روشن کردن آن‌ها کاهش داد اما این راهکاری برای حل مسئله نیست و هوشمند بودن گوشی مورد نظر را زیر سوال می‌برد. اکثر کاربران این مسائل را آزار دهنده یافته و به دنبال گوشی‌هایی واقعا هوشمند با قابلیت شارژدهی بیشتر هستند. از این رو موضوع کاهش مصرف انرژی همواره مسئله‌ای مهم در دستگاه‌های قابل حمل بوده است. از آنجا که باتری‌ها می‌توانند مقدار مشخص و ثابتی از انرژی را در خود ذخیره کنند، زمان قابل استفاده‌ی عملیاتی گوشی برای هر کاربر در هر سیکل شارژ محدود است. این زمان عملیاتی از فاکتورهای مهم برای کاربران در انتخاب گوشی‌ست و شرکت‌های تولیدکننده‌ی تلفن‌های هوشمند را بر آن داشته تا راه حل‌هایی برای افزایش آن بیابند. تحقیقات برای دست‌یابی به این مهم در زمینه‌های مختلف مثل سیستم، معماری مدارها، پردازنده‌ها، حافظه‌ها، صفحات نمایش، زیر سیستم‌های بی‌سیم و نرم افزار در حال انجام است اما تکنولوژی و دانش فنی موجود در کنار پیش‌بینی‌های علمی حاکی از آن است که در حال حاضر بهترین راه و تنها روش جایگزین برای افزایش طول عمر باتری‌ها، کاهش مصرف انرژی در سطح سخت افزار و طراحی و توسعه‌ی نرم افزارها و سیستم‌های عامل با بازده‌ی انرژی بهینه است. در این گزارش ما به روشی نرم افزاری برای کاهش مصرف انرژی گوشی تلفن همراه هوشمند می‌پردازیم.

به طور کلی روش‌های موجود برای کاهش مصرف در سطح نرم افزاری به شش گروه تقسیم می‌شوند [۳]: سیستم‌های عامل آگاه از میزان مصرف انرژی، اندازه‌گیری و مدل‌سازی توان و مصرف انرژی، ارتباط کاربر با برنامه‌ها و منابع محاسباتی، بهینه‌سازی رابط‌ها و درگاه‌های بی‌سیم، بهینه‌سازی حسگرها و کاهش بار محاسباتی. در ادامه به بررسی مختصر هر یک از این راهکارها می‌پردازیم.

- سیستم‌های عامل آگاه از میزان مصرف انرژی:

اصلی‌ترین پرسش در افزایش بازده از نظر مصرف انرژی این است که چه کسی مسئول مدیریت انرژی‌ست؛ برنامه‌ها یا سیستم عامل؟ پاسخ درست هر دو گزینه است. در سطح سیستم عامل ایده‌ی اصلی کاهش مصرف انرژی با یکتا سازی منبع و مدیریت انرژی با تلاش برای رسیدن به حداکثر سود با نفوذ و همکاری سیستم عامل است. در واقع لازمه‌ی کاهش مصرف و مدیریت انرژی، آگاهی و برخورداری از دیدگاهی درست نسبت به چگونگی تقاضای منابع توسط کاربران و برنامه‌هاست.

- اندازه‌گیری و مدل‌سازی توان و مصرف انرژی:

جهت طراحی سیستمی آگاه به میزان مصرف انرژی، اطلاع از چگونگی استفاده‌ی قطعات سخت افزاری از انرژی الزامی‌ست. طراحان سخت افزار ویژگی‌هایی را به قطعات اظافه کرده‌اند تا بتوانند به طور پویا مصرف انرژی خود را با توجه به کارایی و قدرت اجرایی مورد نیاز تغییر دهند. این ویژگی‌ها در اختیار برنامه‌نویسان قرار می‌گیرند؛ اما استفاده‌ی بهینه از آن‌ها نیازمند درک درستی از معنای تصمیم گیری طراحی آن‌ها در بهینگی مصرف انرژی‌ست. از این رو اندازه‌گیری و مدل‌سازی توان مورد استفاده توسط قطعات سخت افزاری و تحلیل آن‌ها راهکاری مناسب خواهد بود.

- ارتباط کاربر با برنامه‌ها و منابع محاسباتی:

افزایش زمان شارژدهی باتری پس از هر سیکل شارژ به شدت به فهمی درست از چگونگی ارتباط کاربر با منابع و باتری نیاز دارد. هر سیستم آگاه به میزان مصرف انرژی باید از اینکه چه زمانی، کجا و چگونه کاربر انرژی باتری را استفاده می‌کند و اینکه احتمال شارژ کردن باتری در چه مواقعی بیشتر است، مطلع باشد. مسئله به دو بخش تقسیم می‌شود: یک بخش تشخیص دوره و زمان شارژ و دشارژ باتری توسط کاربر است که سیستم عامل و کاربر هردو باید قادر به شناسایی محدودیت‌های آینده از نظر توان باقی مانده و اولویت بندی کارها باشند که برای انجام آن بدون تحت تاثیر قرار دادن تجربه‌ی کاربری کاربر نیاز داریم تا بدانیم کاربر چگونه با دستگاه و منابع آن برخورد می‌کند و چگونه به انرژی نیاز دارد.

- بهینه‌سازی رابط‌ها و درگاه‌های بی‌سیم:

رابط‌های بی‌سیم از بزرگ‌ترین مصرف کنندگان انرژی در گوشی‌های تلفن همراه هستند. روش‌های متعددی برای بهینه‌سازی بیشتر در هر لایه‌ی پشته‌ی درگاه و همینطور در بین لایه‌ها، با سود بردن از

حالت‌های متفاوت توان وجود دارد. البته این روش‌ها به زیر ساخت‌هایی در سیستم عامل، برنامه‌ها و شبکه‌ی مورد نظر نیاز دارند. برای مثال تکنولوژی‌های جدید مثل LTE روی کاهش هزینه‌ی انتقال بیت نسبت به گذشته هدف‌گذاری می‌کنند [۴]. پیشرفت‌های خوبی در این زمینه مثل بلوتوث کم مصرف [۵] و WiFi کم مصرف [۶] حاصل شده است.

- بهینه‌سازی حسگرها:

نرم افزارهای آگاه از موقعیت جغرافیایی گوشی همراه به سرویس‌هایی پرکاربرد و مورد علاقه در سیستم-های قابل حمل تبدیل شده‌اند. یک گوشی موبایل حسگرهایی مثل GPS، موقعیت یابی تحت شبکه و شتاب دهنده‌های موقعیت با وضوح و توان درخواستی متفاوت دارد که در نتیجه مصالحه‌ای بین دقت و مصرف انرژی به وجود خواهد آمد. این گروه از راه‌کارها بر به حداقل رساندن مصرف انرژی در استفاده‌ی مداوم از حسگرها در سطح نرم افزاری تمرکز می‌کنند.

- کاهش بار محاسباتی:

محاسبات ابری در حال گشایش امکاناتی جدید در سیستم‌های قابل حمل مثل تلفن‌های همراه به روش-هایی متعدد است. کاهش بار محاسباتی و انتقال آن به فضای ابری می‌تواند در افزایش زمان شارژدهی باتری و توان اجرایی و محاسباتی در سیستم‌های با منابع محدود بسیار مؤثر باشد و سیستم‌های عامل نوین بیش از پیش بر سرویس‌های آنلاین در حال اجرا در فضای ابری اتکا می‌کنند. اجرای از راه دور اجازه می‌دهد تا محاسبات از گوشی تلفن تغذیه شده توسط باتری به ماشین‌های قدرتمند با منابع تغذیه‌ی وسیع در آن سوی اینترنت منتقل شوند. به وضوح عواملی چون وضعیت شبکه بر این راهکارها مؤثر خواهند بود.

در بین کارهای پیشین انجام شده و مرتبط با این گزارش تعداد اندکی از تغییر پویای نیازهای کاربران و برنامه‌ها برای مدیریت منابع سیستم استفاده کرده‌اند. با این حال سعی می‌کنیم در این بخش به آن‌ها اشاره کرده و تفاوت‌های موجود را بیان کنیم. از نخستین کارهای انجام شده در این موضوع [۷] است. نویسندگان در این پژوهش منفعت حاصل از مطالعه‌ی فعالیت‌های در لحظه‌ی کاربر را برای مشخص کردن مصرف انرژی و کنترل توسعه‌ی بهینه‌سازی توان مصرفی، نشان دادند. هم‌چنین صفحه‌ی نمایش و پردازنده را پر مصرف‌ترین بخش‌های گوشی یافتند.

دیگر کارها مانند [۸] اهمیت مطالعه‌ی رفتار کاربر برای بهینه‌سازی مصرف انرژی را نشان داده و بر ارتباط این دو تاکید کردند. در [۹] نویسندگان روشی برای دخیل کردن تجربه‌ی کاربری کاربر در راهکارهای بهینه‌سازی مصرف انرژی ارائه دادند. آن‌ها یک کنترل کننده‌ی جدید برای فرکانس پردازنده توسعه دادند تا سرعت کلاک را هنگام اجرا بتوانند تغییر بدهند. این کنترل کننده درک کاربر از زمان پاسخ برنامه در لحظه‌ی اجرا را تحلیل کرده

و با استفاده از این اطلاعات، فرکانس پردازنده را تغییر می‌داد. این روش باعث ۶۵,۵ درصد بهبود بازده در گوشی-های اندرویدی شد.

برخی پژوهش‌ها از روش‌های یادگیری ماشین برای گروه بندی برنامه‌ها و فعالیت‌های کاربر استفاده کرده‌اند. با تمرکز بر مصرف WiFi، روش ارائه شده در [۱۰] بین برنامه‌ها برای اولویت بندی رده‌ی استفاده‌ی آن‌ها از شبکه دست به انتخاب می‌زد و تنها به برنامه‌های با اولویت بالا اجازه‌ی استفاده از شبکه را می‌داد.

در مقایسه با تمام کارهای گذشته روش پیشنهادی ما ویژگی‌های اصلی زیر را دارا می‌باشد:

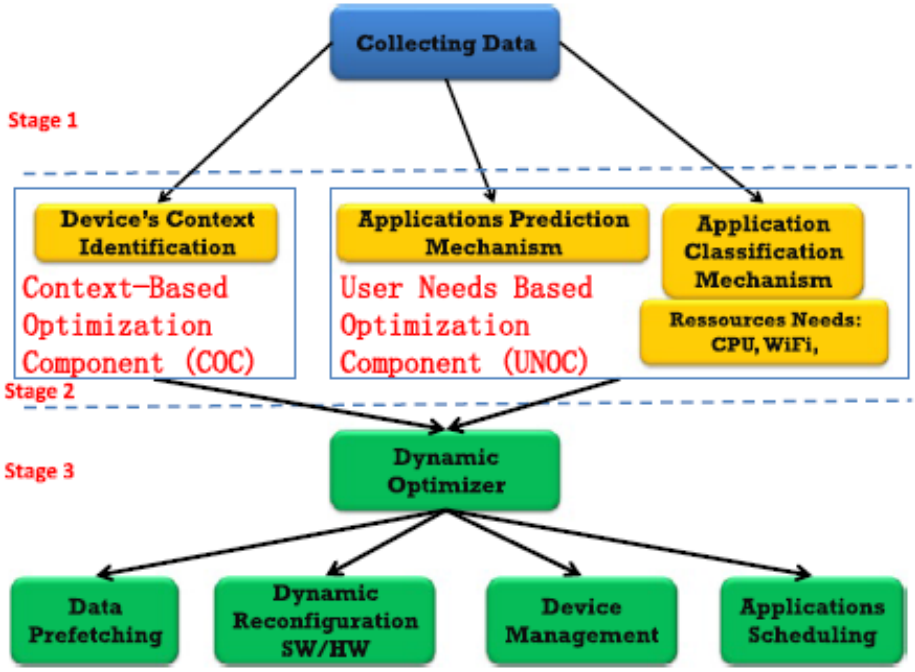
- روش ذخیره‌ی انرژی بیش از یک قطعه را شامل می‌شود (پردازنده، GPS، WiFi و درخشندگی).
- استفاده از چهارچوبِ با معماریِ قسمت بندی شده
- تنظیمات در لحظه‌ی اجرا با توجه به پیش بینی‌ها
- حفظ هدف اصلی (کاهش مصرف انرژی) در مدیریت درخشندگی صفحه نمایش

روشی که در این تحقیق به آن پرداخته شده است را می‌توان در جایی بین گروه اول و سوم قرار داد. همانطور که پیش‌تر بیان شد، یک لازمه‌ی مهم مدیریت مؤثر و کارای انرژی، فهم درستی از چگونگی و محل مصرف انرژی در دستگاه و همینطور میزان مصرف هر بخش از آن است. طراحان سیستم باید الگوهای ارتباط کاربر با بخش‌های مختلف دستگاه را در نظر بگیرند و بتوانند به طور واضح تاثیر هر بهینه‌سازی را بر روی تجربه‌ی کاربری کاربر درک کنند. هدف در روش پیشنهادی این گزارش به دست آوردن، ذخیره و پردازش اطلاعات مربوط به عادات و رفتار کاربر در کنار اطلاعات مربوط به تغییر نیازهای برنامه‌ها به وسیله‌ی محاسبات و داشته‌های حاصل از حسگرهای دستگاه برای کاهش مصرف انرژی‌ست. نکته‌ی کلیدی در روش ما برای ذخیره‌ی انرژی نفوذ و کسب اطلاعات محتوایی کاربر و رفتار و عادات وی برای پیش‌بینی برنامه‌هایی که اجرا می‌کند و بهبود سیاست‌های مدیریت انرژیِ سیستم عامل است. اصلی‌ترین بخش‌های این کار به طور زیر خلاصه می‌شوند:

۱- از حسگرهایی قدرتمند برای جمع‌آوری و پویش مجموعه‌ی بزرگ داده‌ها برای پیدا کردن الگوهای مصرف محتوا استفاده می‌کنیم. هم‌چنین از مقیاس‌هایی برای اندازه‌گیری نیازهای کاربر و مشخص کردن عادات وی استفاده می‌کنیم. شناسایی محتوا و کنش‌ها و واکنش‌های مربوطه‌ی کاربر، تصمیم گیری برای کاهش انرژی اختصاص یافته به منابع بی‌استفاده را در برخی موارد ممکن می‌سازد.
۲- روشی جدید برای گروه بندی و تشخیص برنامه‌های اجرا شده و یافتن ترتیب پر تکرار اجرای برنامه‌ها ارائه می‌شود. بر این اساس می‌توان پیش‌بینی کرد که چه برنامه‌ای با احتمال زیاد در آینده اجرا خواهد

شد. با پیش‌بینی توسعه یافته و آگاهی از نیازهای هر برنامه، قادر خواهیم بود تا منابع در اختیار را تنظیم کرده و بهینه‌سازی‌هایی چون مقیاس‌گذاری پویای فرکانس ولتاژ (DVFS) [۱۱]، پیش واکشی داده و مدیریت دستگاه بدون تاثیر منفی بر رضایت کاربر دست بیابیم. این کارها باعث کاهش مصرف انرژی در کل سیستم می‌شوند.

روند کلی روش پیشنهادی ما در شکل ۱ به نمایش در آمده است. تصویر چکیده‌ای از چند مرحله‌ی کلیدی از روش ما را نشان می‌دهد. مرحله‌ی اول شامل جمع آوری داده در مورد رفتار کاربر و محتوای دستگاه مثل برنامه‌های در حال اجرا، پردازش‌های پس‌زمینه، موقعیت دستگاه، درخشندگی محیط و تاریخ و زمان است.



شکل ۲ بررسی اجمالی روش ارائه شده. تعمیم مراحل روش پیشنهادی

این اطلاعات در مرحله‌ی بعدی طبق سه گام زیر مورد استفاده قرار می‌گیرند:

- طبقه بندی آفلاین برنامه‌ها از نظر منابع
- روش پیش بینی برنامه‌ها
- شناسایی آنلاین محتوای دستگاه

در مرحله‌ی سوم فعال کننده‌ی بهینه سازِ پویا از خروجی مرحله‌ی دوم استفاده می‌کند تا عملیاتی چون مدیریت دستگاه و زمان‌بندی برنامه‌ها را در جهت کاهش مصرف انرژی انجام دهد. ما یک چهارچوب کلی داریم که شامل دو بخش اصلی می‌شود:

- مؤلفه‌ی بهینه‌سازیِ بر پایه‌ی محتوا (COC): بر پایه‌ی محتوای دستگاه و داده‌های حسگرها
- مؤلفه‌ی بهینه‌سازیِ بر پایه‌ی نیازهای کاربر (UNOC): بر پایه‌ی کنش‌های کاربر و گروه بندی برنامه‌ها

در ادامه در فصل ۲ به معماری چهارچوب این گزارش شامل COC و UNOC، در فصل ۳ به توضیح و تشریح COC، در فصل ۴ به شرح UNOC، در فصل ۵ به آزمایش‌های انجام شده ونتایج آن‌ها، در فصل ۶ به کارهای مرتبط و در فصل ۷ به جمع بندی و نتیجه گیری خواهیم پرداخت.

۲ معماری چهارچوب استفاده شده

برای به دست آوردن اهداف بیان شده در این تحقیق، ما یک چهارچوب خاص برای بهینه‌سازی مصرف انرژی و بهبود بخشیدن به مدیریت انرژی توسط سیستم عامل طراحی کردیم. این چهارچوب دارای دو مؤلفه است: بهینه‌سازیِ بر پایه‌ی محتوا (COC) و بهینه‌سازیِ بر پایه‌ی نیازهای کاربر (UNOC). این دو از سنسورها و داده‌های متفاوت استفاده کرده و به طور هم‌زمان و موازی اجرا می‌شوند. یک نکته‌ی کلی این است که با توجه به نوع دستگاه مد نظر (گوشی تلفن همراه، تبلت، لپ‌تاپ و غیره)، برخی کارایی‌های این دو ممکن است در دستگاهی مناسب‌تر باشد. مثلا راهکار ارائه شده‌ی مدیریت نور نمایشگر که وابسته به محل دستگاه است بیشتر برای لپ‌تاپ‌ها و دستگاه‌های عمدتا ساکن کاربرد دارد. سایر راهکارهای بهینه‌سازی ارائه شده شامل مدیریت نور نمایشگر با توجه به درخشندگی محیط، مؤلفه‌ی نیازها و عادات کاربر و مدیریت سطح میکروفن با توجه به سر و صدای محیط در همه‌ی دستگاه‌های قابل حمل با وجود یک سیستم عامل و لایه‌ی کاربر قابل استفاده هستند. به تبع آن، گوشی‌های هوشمند، تبلت‌ها و لپ‌تاپ‌ها می‌توانند از این روش‌ها سود ببرند. معماری عملی این دو مؤلفه را در شکل ۳ (COC) و شکل ۴ (UNOC) می‌بینیم که در واقع معماری ارائه شده در شکل ۲ را پیاده‌سازی می‌کنند.

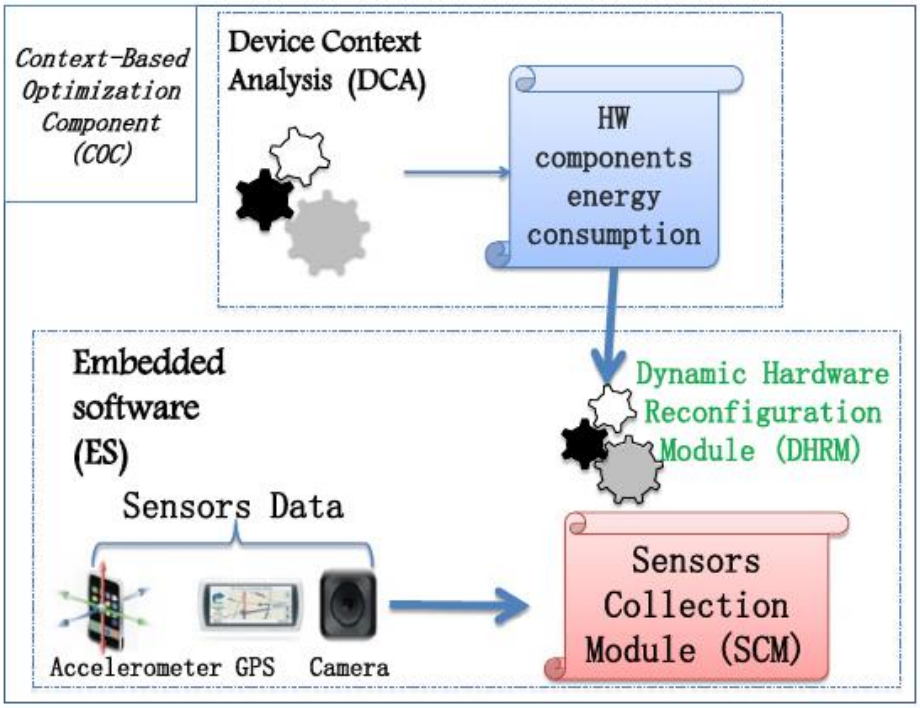
۲ – ۱ بهینه‌سازیِ بر پایه‌ی محتوا (COC)

وظیفه‌ی این مؤلفه جمع‌آوری داده‌ها از سنسورهای جاسازی شده، موقعیت دستگاه، درخشندگی محیط، سر و صدای محیط و غیره است. این داده‌ها نشانگر محتوای دستگاه هستند که همان طور که در شکل ۳ مشخص شده برای اجرای سیاست‌های مختلف از جمله تنظیم نور نمایشگر و حجم صدای بلندگو به کار گرفته می‌شوند. COC در دو فاز کار می‌کند:

۱- تحلیل محتوای دستگاه: در این بخش برخی آزمایش‌های مقدماتی انجام شده و محتوای دستگاه تشخیص داده می‌شود. بین همه‌ی سنسورهای موجود، مرتبط‌ترین‌ها آن‌ها انتخاب می‌شوند تا اطلاعات مد نظر را در اختیار ما بگذارند. این داده‌ها برای توسعه‌ی فاز دوم مورد استفاده قرار می‌گیرند.

۲- نرم افزار جاسازی شده: در این بخش ما فعالیت اجزای مختلف سیستم را به طور لحظه‌ای کنترل می‌کنیم. با توجه به محتوا، ممکن است این اجزا تاثیر زیادی روی انرژی مصرفی داشته باشند. این کنترل با بهره برداری از داده‌های حاصل از حسگرهای جاسازی شده انجام می‌شود.

ایده‌ی اصلی این راه‌کار این است که با مورد توجه قرار دادن اطلاعات محتوایی می‌توان مصرف انرژی را کاهش داده و آن را ذخیره کرد. این هدف با نظارت بر حسگرهای موجود در دستگاه قابل دست‌یابی است. داده‌های حسگرها پردازش شده و به فرصت‌های کاهش مصرف انرژی مرتبط می‌شوند.



شکل ۳ معماری مؤلفه‌ی بهینه سازی بر پایه‌ی محتوا(مؤلفه‌ی اول)

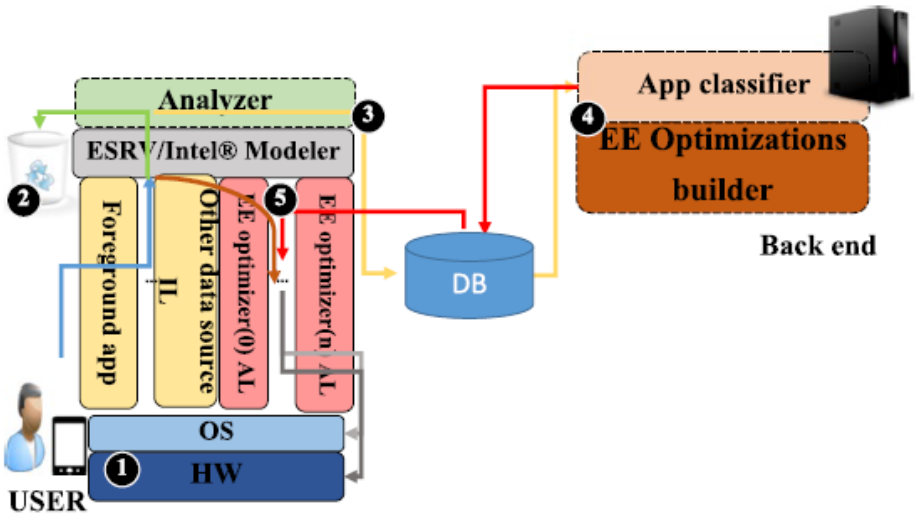
۲ – ۲ بهینه‌سازیِ بر پایه‌ی نیازهای کاربر (UNOC)

این بخش برای استفاده از نیازها و عادات کاربر در جهت بهینه‌سازی مصرف انرژی ساخته شده است. ساختار و معماری کلی این مؤلفه در شکل ۴ نشان داده شده است. این جزء در پنج گام زیر پیاده‌سازی می‌شود:

۱- راه‌کار جمع آوری داده: اطلاعات مربوط به رفتار کاربر و میزان استفاده از دستگاه جمع آوری می‌شوند.
۲- پردازش داده‌های جمع آوری شده به وسیله‌ی تحلیل‌گر برای ضمانت حفظ حریم شخصی کاربر با ناشناس کردن داده‌ها.
۳- ذخیره‌سازی داده‌ها در یک پایگاه داده.
۴- این گام خود به دو بخش تقسیم می‌شود:

- بارگذاری داده‌های جمع آوری شده در جز back-end به منظور پردازش آن‌ها توسط روش‌هایی مثل طبقه بندی برنامه‌ها، پیش‌بینی برنامه‌ها و پروفایل کردن رفتار کاربر.
- ساخت قوانین بهینه‌سازی.

۵- اعمال قوانین به دست آمده به فعال کننده‌ی بهینه‌سازی برای پیاده‌سازی یک روش بهینه‌سازی خاص برای هر جزء سخت افزاری دستگاه.



شکل ٤ معماری بهینه سازی بر پایه‌ی نیازهای کاربر (مؤلفه‌ی دوم)

در این گزارش بر روی راه‌کار جمع آوری داده، طبقه بندی برنامه‌ها، پیش‌بینی برنامه‌ها و روش بهینه‌سازی به شکل زیر تمرکز می‌کنیم:

۱- راه‌کار جمع آوری داده: همان گام اول اشاره شده است. در این گام مجموعه‌ی بزرگی از داده‌ها را شامل برنامه‌های در حال اجرا، تاریخ، ساعت، زمان سپری شده توسط هر برنامه و فرآیندهای پس زمینه جمع آوری می‌کنیم.

۲- این بخش خود شامل مراحل زیر است:

- طبقه بندی آفلاین برنامه‌ها با توجه به نیاز آن‌ها به Wi-Fi و پردازنده. در حال حاضر تنها بر همین دو جز (Wi-Fi و پردازنده) تمرکز می‌کنیم. این دو جز پر مصرف‌ترین اجزای دستگاه‌های قابل حمل هستند اما این طبقه بندی می‌تواند شامل روشنایی صفحه، میکروفن، GPS و غیره نیز بشود.
- متوسط زمان اجرای هر برنامه محاسبه می‌شود. این مرحله هم به طور آفلاین انجام می‌شود اما پایگاه داده را می‌توان به طور دوره‌ای بروز کرد.
- روش پیش‌بینی برنامه‌ها.

۳- تمام مراحل ترکیب شده و توسط فعال کننده‌ی بهینه‌سازی که اتصالات Wi-Fi و فرکانس پردازنده را برای بهینه‌سازی مصرف انرژی مدیریت می‌کند، مورد استفاده قرار می‌گیرند.

در ادامه دو مؤلفه‌ی ذکر شده را شرح می‌دهیم.

۳ بهینه‌سازیِ بر پایه‌ی محتوا (COC)

یک بخش مهم از مدیریت انرژی داشتن درکی درست از این مهم است که چگونه، چه زمانی و کجا کاربر با دستگاه تعامل داشته و تقاضای استفاده از منابعی چون نور صفحه، حجم صدا، اتصالات و غیره را دارد. COC بر اساس محتوای دستگاه و کارهای کاربر پایه ریزی شده است. محتوای دستگاه با موقعیت، نور محیط و سر و صدای محیطش تعریف می‌شود. روشنایی صفحه و میزان صدای بلندگو به ترتیب و با توجه به موقعیت دستگاه (عادی یا غیر عادی، درخشندگی محیط و سر و صدای محیط) کنترل می‌شوند. برای انجام این کار سیاست‌هایی بر داده‌های حاصل از حسگرها اعمال می‌شوند تا میزان مصرف انرژی را کاهش دهند. ماژول مجموعه‌ی حسگرها (SCM) و ماژول بازپیکربندی پویای سخت افزار (DHRM) برای رسیدن به اهداف COC توسعه داده شدند. در دو بخش زیر به توضیح چگونگی مدیریت روشنایی و صدا توسط SCM و DHRM می‌پردازیم.

۳ – ۱ مدیریت روشنایی صفحه بر اساس موقعیت دستگاه

۳ – ۱ – ۱ ماژول مجموعه‌ی حسگرها (SCM)

وظیفه‌ی این ماژول جمع آوری داده‌ها از حسگرهای جاسازی شده برای تشخیص مناسب‌ترین وضع قرارگیری دستگاه است. در گوشی‌های هوشمند امروزی حسگرهای متعددی مثل شتاب‌سنج، حسگر نور محیط (ALS)، حسگر جهت، شیب‌سنج، قطب‌نما، گردش‌نما (gyrometer) و حسگر موقعیت جغرافیایی وجود دارند. برای فهمیدن این که کدام حسگرها مرتبط‌تر از سایرین هستند برخی آزمایشات مقدماتی انجام شده است. در ابتدا مقادیر حسگرها را در موقعیت‌های مختلف وضع قرارگیری دستگاه (حالت عادی، شیب‌دار، متغیر و پرحرکت و غیره) جمع آوری می‌کنیم. مقادیر حسگرها را در چند موقعیت قرارگیری متفاوت مقایسه می‌کنیم تا مرتبط‌ترین‌ها را انتخاب کنیم. حسگرهایی که مقادیر بسیار متفاوتی را در موقعیت‌های متفاوت نشان می‌دهند نادیده گرفته می‌شوند. حسگرهای زیر موجودند:

- درخشندگی محیط: حسگر نور محیط (ALS).
- جهت: شیب‌سنج، قطب‌نما.

- حرکت: شتاب‌سنج، گردش‌نما
- موقعیت مکانی: GPS
- سر و صدای محیط: میکروفن

حسگرهای زیر را انتخاب می‌کنیم:

۱- برای درخشندگی و سر و صدای محیط: از حسگرهای ALS و میکروفن استفاده می‌کنیم زیرا این دو تنها حسگرهایی هستند که این اطلاعات مورد نظر را فراهم می‌کنند.

۲- برای جهت: شیب‌سنج را انتخاب کرده ایم چون داده‌های به دست آمده از این حسگر حاوی اطلاعات بیشتری نسبت به قطب‌نماست و علاوه بر آن داده‌های حاصل از قطب‌نما وابسته به شدت میدان مغناطیسی هستند.

۳- برای حرکت: هردو حسگر شتاب‌سنج و گردش‌نما دارای مقیاس‌های سه بعدی در محورهای x، y و z هستند. در مثال زیر انحراف معیار را مقایسه می‌کنیم:

- Accelerometer(x,y,x) = (0.57,0.37,0.52)
- Gyroscope(x,y,z) = (89.42,57.80,54.95)

فراوانی نرمال شده حساسیت را نشان می‌دهد. مقادیر با حساسیت بالا حاوی اطلاعات بیشتری هستند. با استفاده از این مقایسه، حسگر gyroscope را برای حرکت انتخاب کردیم.

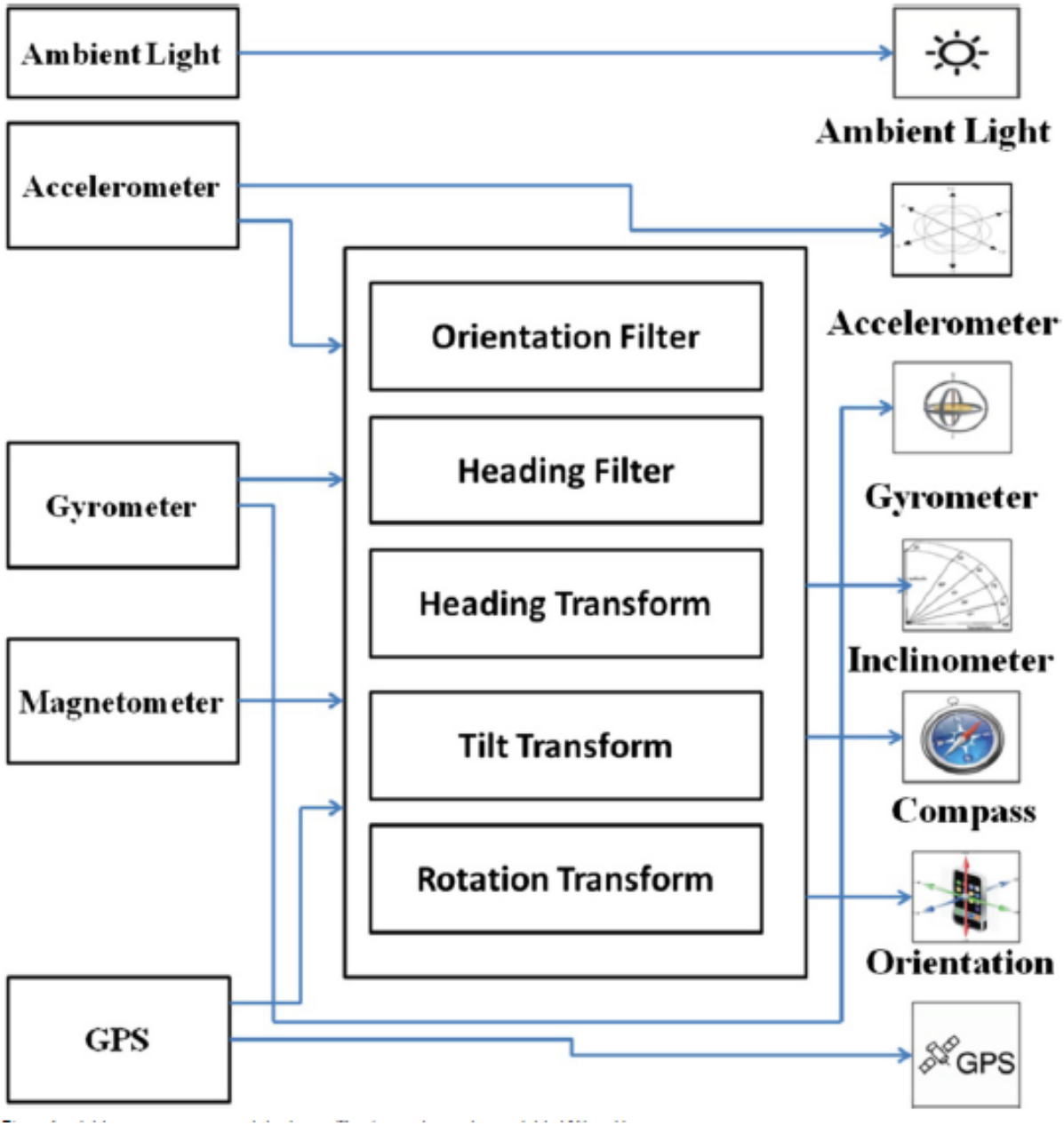
۴- از جمع‌آوری داده‌های مربوط به موقعیت مکانی به علت محرمانه بودن آن‌ها صرف نظر می‌کنیم.

SCM با استفاده از اطلاعات حاصل از حسگرها، در هنگامی که دستگاه در حالت ایستاده قرار دارد، کالیبره می‌شود (مندرج می‌شود). پس از آن اطلاعات حسگرها به طور لحظه‌ای جمع آوری می‌شوند. به این شکل اگر دستگاه کج شود، اطلاعات شیب آن در آنِ واحد در SCM بروز می‌شود. در پایان داده‌ها برای استفاده‌ی ماژول بازپیکربندی پویای سخت افزار (DHRM) در حافظه ذخیره می‌شوند. این بازپیکربندی مداوم بوده و در پس‌زمینه انجام می‌شود. هربار که مقدار یک حسگر تغییر می‌کند، SCM مقدار جدید را توسط یک حافظه‌ی به اشتراک گذاشته شده (shared memory) بروز می‌کند و سپس DHRM بهینه‌سازی را بر میزان روشنایی صفحه اعمال می‌کند. در شکل ۵ حسگرهای منطقی و HW و در شکل ۶ روند فرآیند SCM نشان داده شده است.

۳ – ۱ – ۲ ماژول بازپیکربندی پویای سخت افزار (DHRM)

وظیفه‌ی این ماژول مدیریت قطعات سخت افزاری با توجه به موقعیت دستگاه که در بخش ۴ – ۱ – ۱ به آن اشاره شد است. داده‌های موجود در حافظه‌ی فرّارِ به اشتراک گذاشته شده، برای اعمال راه‌کار تصمیم گیری

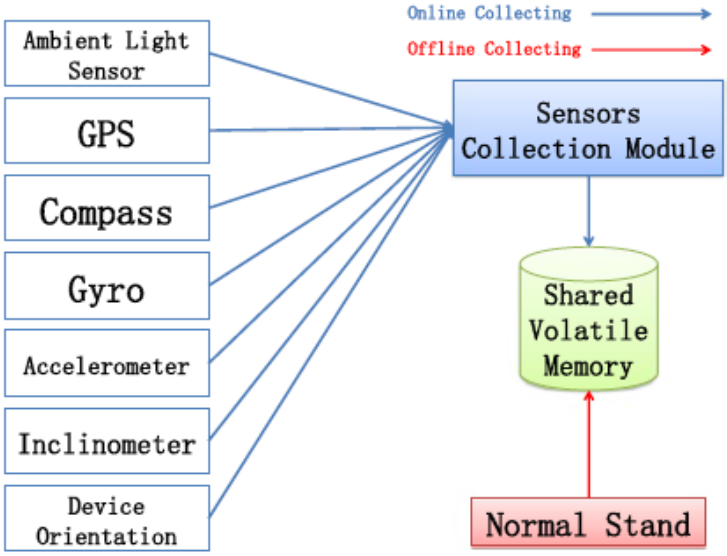
منطقی وارد این ماژول می‌شوند. این ماژول درایور صفحه نمایش دستگاه را کنترل کرده و میزان روشنایی آن را همان طور که در شکل ۷ می‌بینیم تغییر می‌دهد.



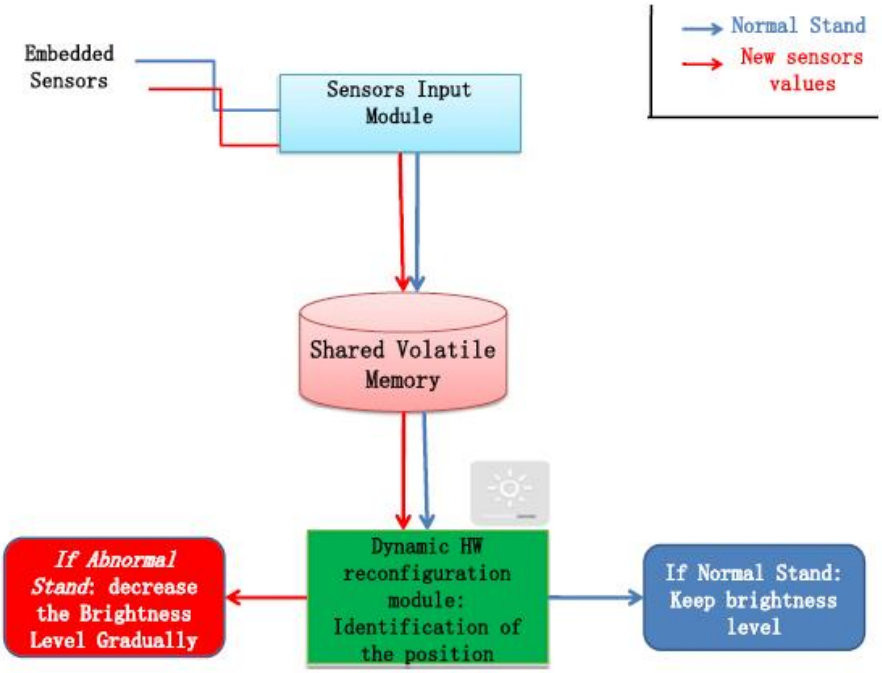
شکل ۵ حسگرهای موجود در گوشی‌های همراه. حسگرهای منطقی و HW موجود

DHRM مقدار جدید حسگر را با مقدار آن در حالت عادی مقایسه و سپس با توجه به این مقایسه، میزان روشنایی صفحه را به مناسب‌ترین میزان برای کاربر تنظیم می‌کند. برای مثال اگر مقدار حسگر ژیروسکپ از محدوده‌ی مجاز بیشتر شود، ماژول یک بهینه‌سازی خاص را بر روشنایی صفحه با کاهش میزان آن اعمال می‌کند. درخشندگی محیط نیز برای تنظیم درخشندگی صفحه نمایش مورد استفاده قرار می‌گیرد. وقتی محیط بسیار

پرنور باشد، روشنایی صفحه زیاد شده و برعکس. در مدیریت روشنایی صفحه نمایش ۳۰ درصد فرصت کاهش توان مصرفی بین حالت بیشترین و کمترین میزان روشنایی وجود دارد.



شکل6 جمع آوری داده‌های حسگرها در SCM



شکل ۷ مدیریت روشنایی توسط DHRM

ماژول DHRM بر اساس داده‌های به دست آمده توسط SCM یکی از چهار حالت قرار گیری دستگاه را انتخاب می‌کند. مثالی از هر یک را در زیر می‌بینیم:

- سخت برای دیدن: لرزش دستگاه برای دیدن درست آن بسیار مهم است.
- حرکت ملایم: از حرکات کوچک تا ملایم مثل بازی کردن.
- حالت عادی: دستگاه در جایی برای لحظه‌ای رها شده است.
- کج غیر عادی: قرار گرفتن دستگاه با زاویه‌های مثبت و منفی ۹۰ درجه بدون حرکت.

هر حالت با توجه به مقیاس حسگرها تشخیص داده می‌شود. DHRM روشنایی صفحه‌ی را، همان طور که در جدول ۱ می‌بینیم، با توجه به حالت تشخیص داده شده روی مقدار مربوط تنظیم می‌کند.

جدول 1 روشنایی صفحه (SB) بر اساس موقعیت دستگاه در DHRM

Threshold	Category	State	SB
Gyro average movement sum ≥ 80	Motion	Hard-To-Watch	20 %
Gyro average movement sum > 80 and gyro average movement sum > 8	Motion	Mild-motion	30 %
Gyro average movement sum < 8 and inclinometer sum ≥ 60	Motionless	Abnormal-tilt	20 %
Gyro average movement sum > 8 and inclinometer sum < 60	Motionless	Normal-stand	Relative to ALS

وقتی دستگاه در حالت عادی قرار دارد، تنها درخشندگی محیط را برای تنظیم روشنایی صفحه مد نظر قرار می‌دهیم (شکل ۸).

۳ – ۲ مدیریت صدا بر اساس سر و صدای محیط

مشابه روش بخش ۴ – ۱ می‌خواهیم حجم صدای بلندگوی دستگاه را با توجه به میزان سر و صدای محیط مدیریت کنیم. باز دو ماژول اصلی داریم: جمع کننده‌ی سر و صدای محیط (ANC) که سر و صدا را اندازه گرفته و در اختیار ماژول دوم قرار می‌دهد؛ و ماژول کنترل صدا (SCM) که حجم صدای بلندگو را با توجه به اطلاعات دریافتی از ماژول اول تنظیم می‌کند.

برای مثال اگر میزان سر و صدای محیط زیاد باشد ماژول SCM حجم صدای دستگاه را زیاد می‌کند و برعکس. این تغییر به صورت پویا و لحظه‌ای انجام می‌شود و علاوه بر آن برای جلوگیری از آزار کاربر افزایش یا کاهش ناگهانی به صورت تدریجی اعمال می‌شود تا کاربر نسبت به آن آگاه نبوده و احساس ناراحتی نکند.

lux	state	brightness
100000	direct sunlight	100
31622.8	direct sunlight	100
28183.8	overcast outdoors	100
100000	overcast outdoors	100
8912.51	bright indoors	100
1778.28	bright indoors	100
1584.89	bright indoors	93
1412.54	bright indoors	90
1258.93	bright indoors	86
1122.02	bright indoors	75
1000	bright indoors	65
891.251	bright indoors	55
794.328	normal indoors	50
707.946	normal indoors	46
630.957	normal indoors	46
562.341	normal indoors	43
501.187	normal indoors	43
446.684	normal indoors	40
398.107	normal indoors	40
354.813	normal indoors	40
316.228	normal indoors	36
281.838	dim indoors	36

lux	state	brightness
251.189	dim indoors	36
223.872	dim indoors	33
199.526	dim indoors	33
177.828	dim indoors	30
112.202	dim indoors	30
100	dim indoors	25
79.4328	dim indoors	25
70.7946	dim indoors	25
39.8107	dim indoors	25
35.4813	dim indoors	25
31.6228	dim indoors	25
28.1838	dim indoors	20
25.1189	dim indoors	20
22.3872	dim indoors	20
19.9526	dim indoors	20
12.5893	dim indoors	20
11.2202	dim indoors	20
10	dim indoors	20
8.91251	darkness	20
7.07946	darkness	20
6.30957	darkness	15

شکل ۸ تنظیم روشنایی صفحه بر اساس درخشندگی محیط. برخی مقادیر درخشندگی محیط (lux) و میزان روشنایی مرتبط با آن

۴ بهینه‌سازیِ بر پایه‌ی نیاز کاربر (UNOC)

۴ – ۱ راه‌کار طبقه بندی

در روش پیشنهادیِ این گزارش طبقه بندی به صورت آفلاین انجام شده و سپس در حین بهینه‌سازی مورد استفاده قرار می‌گیرد. برنامه‌ها بر اساس میزان مصرف‌شان از پردازنده و Wi-Fi گروه بندی می‌شوند. برای Wi-Fi طبقه بندی به صورت دو حالته (روشن یا خاموش) و برای پردازنده بر اساس بیشترین فرکانس پردازنده است. در هر دو مورد قبل از تنظیم و تغییر منابع، فعال کننده‌ی بهینه‌سازی لیست همه‌ی پردازش‌ها را چک می‌کند تا تداخلی به وجود نیاید.

۴ – ۱ – ۱ طبقه بندی بر اساس استفاده از Wi-Fi

هدف در این طبقه بندی مدیریت رابط بی‌سیم بر اساس نیاز های برنامه‌های در حال اجراست. برای رسیدن به طبقه بندی آفلاین، ما تعدادی آزمایش مقدماتی انجام دادیم. در ابتدا نرخ استفاده از اینترنت با جمع زدن نرخ دانلود و آپلود در حالی که هیچ برنامه یا فرآیندی در پس‌زمینه در حال اجرا نبود، اندازه‌گیری شد. حداقل آن به علت مدیریت اتصالات در سیستم عامل ویندوز شرکت Microsoft در حالی که برنامه‌ای در حال اجرا نیست برابر با ۱۰ کیلوبایت به دست آمد. در گام بعدی برنامه‌هایی را که می‌خواستیم آن‌ها را طبقه بندی کنیم جداگانه اجرا کرده و پهنای باند مصرفی را اندازه می‌گیریم. وقتی که جمع نرخ دانلود و آپلود در حال اجرای یک برنامه کمتر از ۱۰ کیلوبایت باشد، فرض می‌کنیم که نیازی به اتصال Wi-Fi نیست و برعکس. جدول ۲ نتیجه‌ی این طبقه‌بندی را برای سه مثال نشان می‌دهد.

جدول 2 طبقه بندی بر اساس مصرف Wi-Fi

Apps	Internet rate	Download	Upload	Class
Messenger	32.55	12.93	19.62	On
Youtube	366.09	325.81	40.28	On
2048	1.56	1.02	0.54	Off

حال نیاز به Wi-Fi ارزیابی شده و هر برنامه در کلاس روشن یا خاموش قرار گرفته است. البته باید کامل‌ترین تصمیم بر اساس شرایط فعلی دستگاه گرفته شود تا وقتی که اتصال Wi-Fi قطع می‌شود موجب نارضایتی کاربر نشود.

۴ – ۱ – ۲ طبقه بندی بر اساس نیاز به پردازنده

سیستم عامل ویندوز ۸٫۱ فرکانس پردازنده را به طور خودکار مدیریت می‌کند اما در برخی موارد فرکانس تنظیم شده توسط آن با فرکانس مورد نیازِ برنامه‌ها و کاربر هم‌خوانی ندارد. برای بهبود این مدیریت تصمیم گرفتیم که برنامه‌ها را بر اساس فرکانس پردازنده‌ی مورد نیاز طبقه بندی کنیم.

در پیاده‌سازی فعلی به طور دلخواه ۳ مرز (۸۰۰ مگاهرتز، ۱٫۲۵ گیگاهرتز و ۱٫۷۵ گیگاهرتز) که ۴ طبقه را به وجود می‌آورند در نظر گرفته شده است. ۴ طبقه به شرح زیر هستند:

کلاس C1: برنامه‌هایی که به فرکانس پردازشی زیر ۸۰۰ مگاهرتز نیاز دارند؛ مثل برنامه‌های ویرایش متن (Word و ...) و بازی‌های ساده

کلاس C2: برنامه‌هایی که به فرکانس پردازشی بین ۸۰۰ مگاهرتز تا ۱٫۲۵ گیگاهرتز نیاز دارند؛ مثل مرورگرها (Chrome و ...).

کلاس C3: برنامه‌هایی که به فرکانس پردازشی بین ۱٫۲۵ تا ۱٫۷۵ گیگاهرتز نیاز دارند؛ مثل بازی‌های سنگین (۲۰۴۸ و ...).

کلاس C4: برنامه‌هایی که به فرکانس پردازشی بالای ۱٫۷۵ گیگاهرتز نیاز دارند؛ مثل برنامه‌های شبیه-سازی، سنتز، پردازش تصویر و غیره که البته در طی آزمایشات این گزارش برنامه‌ای جز این کلاس قرار نگرفته است.

برای قرار دادن یک برنامه در یک طبقه، میزان بکارگیری پردازنده و فرکانس مورد نیاز آن اندازه‌گیری می‌شود. به طور دقیق‌تر میانگین (m)، انحراف معیار (e) و نسبت فرکانس پردازشی استفاده شده به حداکثر فرکانس (f) برای طبقه بندی برنامه‌ها انتخاب شده‌اند؛ به این شکل که احتمال تعلق برنامه به کلاس Ci محاسبه می‌شود. طبقه بندی ما به کمک یک گروه بند Bayesian انجام شده است. هرچند که روش‌های بسیار متعددی مثل درخت تصمیم گیری، روش‌های قانون محور مثل پس‌رفت منطقی (logistic regression)، پس‌رفت خطی، NaiveBayes، ماشین حمایت بردار (SVM)، k-نزدیک‌ترین همسایه (k-NN) و شبکه‌ی عصبی مصنوعی (ANN) برای این کار وجود داشت اما ما به علل زیر روش Naïve Bayesian را انتخاب کردیم.

- سادگی پیاده‌سازی.
- سرعت در بررسی و طبقه بندی.
- به داده‌های اندکی برای پیش‌بینی پارامترها نیاز دارد (مانند مورد ما در این گزارش).
- عدم حساسیت به ویژگی‌های نامربوط که حتی وقتی فرض NB برقرار نیست به نتایج خوب می‌رسد.

- مانند روش پس‌رفت منطقی (logistic regression) زودتر از مدل متمایز کننده همگرا می‌شود که نتیجتاً انرژی مصرفی بررسی آن کمتر خواهد بود.

گروه بند ما به شکل زیر عمل می‌کند:

$$\mathrm{P(Ci|m,e,f)} = \max 1 \le j \le 4 \; P(Cj|m,e,f)$$

احتمال یک برنامه برای قرار گرفتن در کلاس Ci به ازای m، e و f:

$$\mathrm{P(Ci|m,e,f)} = P(Ci) * \frac{P(m,e,f|Ci)}{P(m,e,f)}$$

$$\mathrm{P(m,e,f)} = P(m|Ci) * P(e|Ci) * P(f|Ci)$$

$$\mathrm{P(Ci|m,e,f)} = \sum_{i=1}^{4} P(m|Ci) * P(e|Ci) * P(f|Ci)$$

با محاسبه‌ی احتمالات مختلف، توزیع m، e و f را توزیع گاوسی در نظر می‌گیریم؛ برای مثال با توجه به میانگین (u) و واریانس نسبت (σ^2) داریم:

$$\mathrm{P(Ci|m,e,f)} = \frac{1}{\sqrt{2 * \pi * \sigma^2}} * \exp(-\frac{(f-u)^2}{2 * \sigma^2})$$

جدول ۳ نتایج طبقه بندی را برای سه برنامه‌ی نمونه نشان می‌دهد. با مد نظر قرار دادن کلاس هر برنامه، فعال کننده‌ی بهینه‌سازی مناسب‌ترین حداکثر فرکانس را انتخاب می‌کند. به وضوح فعال کننده‌ی بهینه‌سازی منابعی که توسط برنامه‌های پس‌زمینه در حال استفاده هستند را قطع نمی‌کند (مثل Wi-Fi برای برنامه‌ی Skype).

پس از طبقه بندی آماری، در قسمت‌های بعدی در مورد جمع آوری داده و راه‌کار پیش‌بینی صحبت می‌کنیم.

جدول3 طبقه بندی بر اساس نیاز به پردازنده

Apps	Avg. usage	Stand dev.	Ratio freq.	Class
Chrome	25.88	7.65	0.71	Medium
Foxit	7.45	4.07	0.55	Low
2048	46	10.58	0.87	High

۴ – ۲ جمع آوری داده و پیش‌بینی

پارامترهای مربوط به کاربران مختلف مثل شغل، نوع زندگی و جنسیت با هم تفاوت دارند. این موارد را برای رسیدن به یک بهینه‌سازی خوب مصرف انرژی برای هر کاربر باید مد نظر قرار داد. در واقع ترتیب برنامه‌ها (سناریو) مطابق نیازهای هر کاربر مشخص، متفاوت هستند. ایده‌ی روش ما این است که برنامه‌های باز شده توسط کاربر را بر طبق روز هفته، ساعت و فرآیندهای پس‌زمینه تحلیل کنیم.

۴ – ۲ – ۱ کاوش کاربر برای جمع آوری داده و پردازش زمان

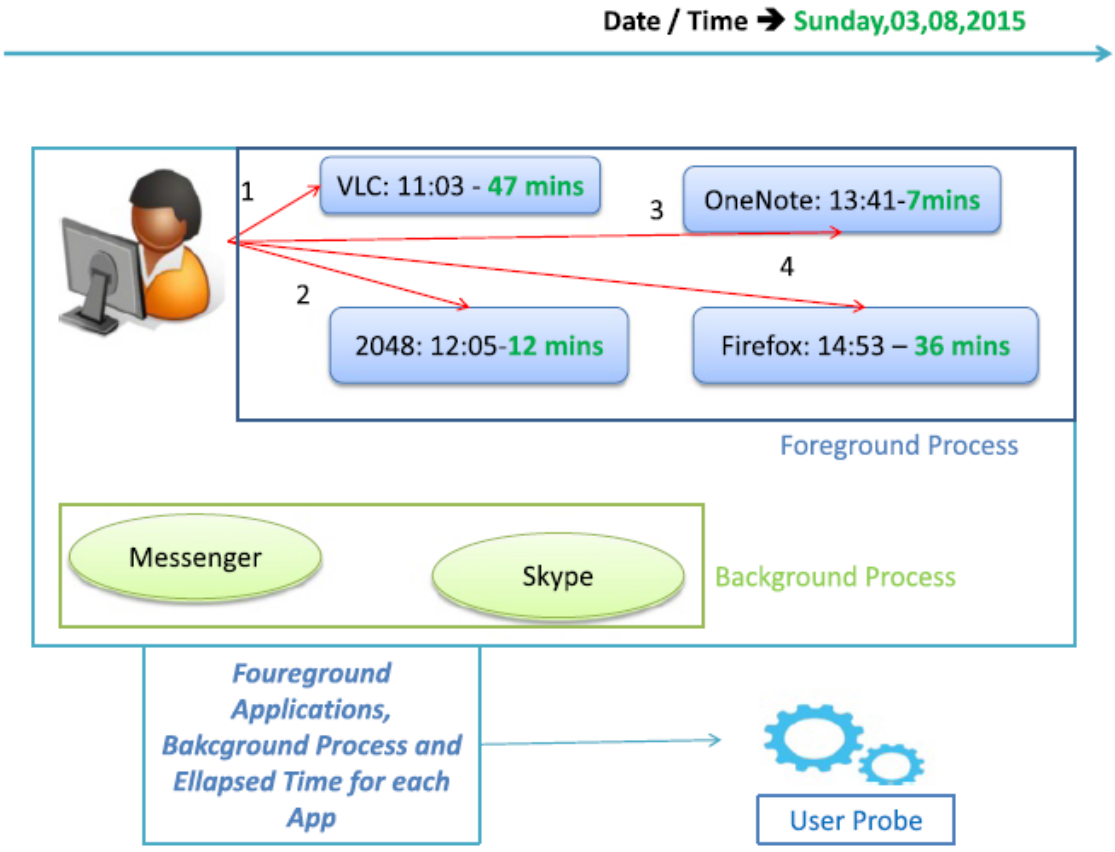
کاوش کاربر وابسته به برنامه‌های مختلف باز شده توسط او در یک بازه‌ی بزرگ زمانی‌ست. پارامترهای اصلی زمان و تاریخ هستند. فرض می‌کنیم کاربر رفتارهای متفاوتی بین روزهای عادی هفته و تعطیلات آخر هفته و همین طور بین دوره‌های متفاوت یک روز دارد. برای مثال برنامه‌های اجرا شده در صبح دوشنبه در محل کار با برنامه‌های اجرا شده در شب شنبه با هم فرق دارند. علاوه بر این انتظار می‌رود که این رفتارها از یک کاربر تا کاربر دیگر نیز متفاوت باشند؛ برای مثال با توجه به شغل‌شان.

کاوش کاربر به صورت لحظه‌ای و در حالت اجرا انجام می‌شود. هر برنامه‌ی باز شده توسط کاربر، سایر فرآیندهای پس‌زمینه، روز هفته، تاریخ و زمان جمع آوری می‌شود. شکل ۹ خلاصه‌ی عملکرد کاوش کاربر را نشان می‌دهد. هر بار که کاربر برنامه‌ی جدید را باز می‌کند، زمان سپری شده روی برنامه‌ی قبلی محاسبه شده و در یک پایگاه داده ذخیره می‌شود تا میانگین زمان اجرای هر برنامه با در نظر گرفتن سایر پارامترهای اشاره شده به دست بیاید.

۴ – ۲ – ۲ پیش‌بینی برنامه‌های آینده

همان طور که بیان شده است ایده‌ی اصلی روش ما ارائه‌ی یک راه مدیریت انرژیِ شخصی‌سازی شده برای سیستم‌های قابل حمل است که روش عادی و استاندارد مدیریت انرژی توسط سیستم عامل را بهبود ببخشد. هدف پیش‌بینی برنامه‌هایی‌ست که در آینده اجرا خواهند شد تا به کمک آن مصرف انرژی را با تنظیم روند مصرف منابع سیستم در سناریوهای خاص کاهش دهیم.

برای پیش‌بینی برنامه‌های آینده فرض می‌کنیم که آن‌ها به برنامه‌هایی که در حال حاضر در حال اجرا شدن هستند و برخی داده‌های موقت بستگی دارند. ما باید بفهمیم که کاربر عادت دارد که چه برنامه‌هایی را به ترتیب و پشت هم اجرا کند. برای این منظور از تکنیک‌های استخراج الگوی ترتیبی (SPM) بین داده‌هایی که از نظر زمانی به هم مرتبط هستند استفاده می‌کنیم. بین روش‌های متعدد SPM که در [۱۳] بررسی شده‌اند ما یکی از ساده‌ترین و رایج‌ترین روش‌ها یعنی *الگوی ترتیبی تعمیم داده شده (GSP)* [۱۴ و ۱۵] را انتخاب کرده ایم.



شکل ۹ کاوش کاربر

الگوریتم GSP برای پیدا کردن ترتیب‌های پر تکرار استفاده می‌شود که نهایتاً ساختارهای ارتباطی یا سببی مربوط به زمان را در یک مجموعه از داده آشکار می‌کند. انگیزه‌ی ما از استفاده از الگوریتم GSP پیدا کردن یک نظم در برنامه‌های اجرا شده در یک روز از هفته، یک زمان خاص و در حالتی که فرآیندهای پس‌زمینه‌ی خاصی در حال اجرا هستند، است. به طور دقیق‌تر موارد پردازش شده با این الگوریتم، برنامه‌های در حال اجرا در یک بازه‌ی زمانی مشخص از یک روز مشخص در طول چند هفته هستند. برای مثال می‌خواهیم بدانیم که آیا یک مجموعه برنامه‌ی خاص با یک ترتیب مشخص هر هفته سه شنبه بین ساعت ۱ تا ۳ بعد از ظهر اجرا می‌شوند. الگوریتم GSP قادر است چنین ترتیب‌هایی را تشخیص بدهد.

یک ترتیب رایج است اگر اجرای آن بیشتر از یک مقدار حداقلی در پایگاه داده ثبت شده باشد. بر اساس الگوریتم Apriori [۶]، GSP با جمع کردن برنامه‌هایی که فرکانس آن‌ها بیشتر از یک مقدار حداقل پشتیبانی است شروع می‌کند تا مجموعه‌ی ترتیب رایج با طول ۱ را بسازد. سپس به صورت مکرر داده‌ها را پویش می‌کند تا تعداد پشتیبانی را جمع آوری کرده و ترتیب‌های مکرر با طول k+1 را از ترتیب‌های مکرر با طول k انتخاب کند. این

فرآیند انقدر تکرار می‌شود تا هیچ ترتیب مکرر یا کاندیدی پیدا نشود. در پایان فرآیند GSP ما تعداد اجرای برنامه‌ها از ترتیب ۱ برنامه‌ای تا ترتیب k برنامه‌ای را خواهیم داشت.

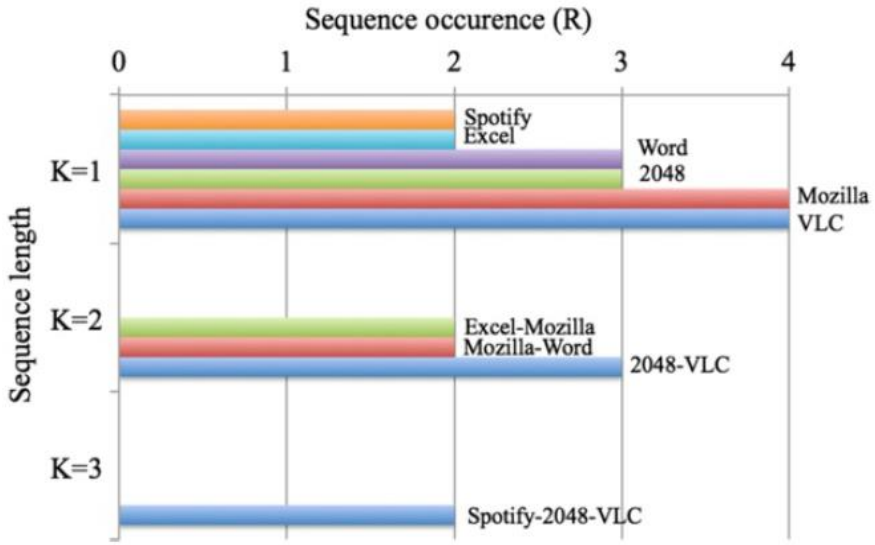
جدول ۴ یک مثال کوچک از داده‌های ورودی فرآیند GSP را نشان می‌دهد. این داده‌ها برنامه‌های اجرا شده در ۴ دوشنبه‌ی پی در پی را نشان می‌دهد. خروجی، همان طور که در شکل ۱۰ می‌بینیم، تعداد اجرای برنامه‌ها از ۹ تا ۱۱ صبح است. «K» طول ترتیب را که تعداد موارد موجود در ترتیب است، نشان می‌دهد. «R» فرکانس تکرار که همان تعداد تکرار این ترتیب است، را نشان می‌دهد.

اگر K = 1، تعداد اجرا (R) تعداد تکرار یک برنامه‌ی مشخص را می‌دهد. می‌بینیم که Mozila و VLC پر تکرارترین برنامه‌ها هستند. اگر K = 2، تعداد اجرا (R) ترتیب‌های تشکیل شده از دو برنامه را می‌دهد؛ برای مثال ترتیب ۲۰۴۸، VLC که سه بار اتفاق افتاده است. اگر K = 3، تعداد اجرا (R) ترتیب‌های تشکیل شده از سه برنامه

جدول4 نمونه داده‌های ورودی GSP

Mon [9–11]	Application sequences
1st week	Excel, Mozilla, Spotify, 2048, VLC
2nd week	Excel, Mozilla, Word, CALC, VLC
3rd week	Notepad, Mozilla, Word, 2048, VLC
4th week	Mozilla, Word, Spotify, 2048, VLC

را می‌دهد؛ برای مثال ترتیب Spotify، ۲۰۴۸، VLC که دو بار اتفاق افتاده است. بر اساس خروجی‌های GSP و داده‌های جمع آوری شده‌ی پویای مربوط به برنامه‌های در حال اجرا، تاریخ و زمان که توسط کاوش کاربر به دست آمده‌اند، برنامه‌هایی که در آینده اجرا خواهند شد را می‌توان پیش‌بینی کرد.



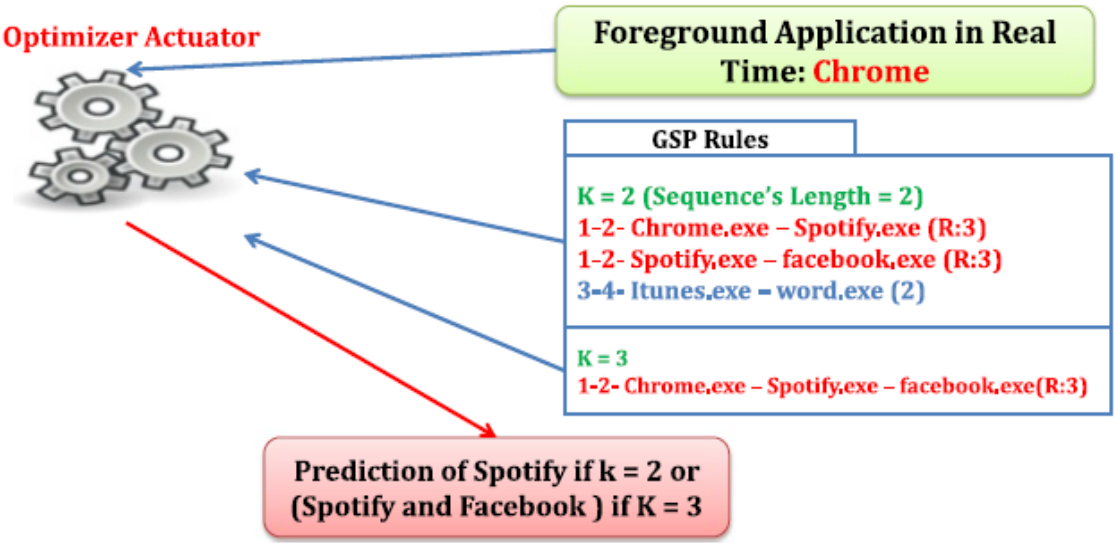
شکل ۱۰ نمونه‌ی خروجی GSP

می‌توان پیش‌بینی‌های متمایزی با توجه به طول ترتیب که همان «K» است، داشت. در زیر این پیش‌بینی‌ها را می‌بینیم:

- با طول ۱ محتمل‌ترین برنامه پیش‌بینی می‌شود (هرچه که برنامه‌ی در حال اجرای فعلی باشد).
- K = 2 نشان می‌دهد که چه برنامه‌ای پس از برنامه‌ی فعلی اجرا خواهد شد؛ مثل ۲۰۴۸ که پس از VLC می‌آید.
- K = 3 دو مورد را نشان می‌دهد. یک این که کدام برنامه پس از ترتیب فعلیِ شاکل از برنامه‌ی در حال اجرا و برنامه‌ی قبلی اجرا خواهد شد. و دوم این که کدام دو برنامه برای اجرا پس از برنامه‌ی فعلی محتمل‌تر خواهند بود.

در این کار به عنوان اولین کار، ما تنها پیش‌بینی بر اساس برنامه‌ی فعلی را مورد مطالعه قرار می‌دهیم که در آن انتخاب کوچک‌ترین K برای پیش‌بینی برنامه‌ی بعدی دقیق‌تر خواهد بود.

هرچند که این انتخاب، فرکانسی که قوانین GSP در آن کار می‌کند را افزایش می‌دهد، باعث مصرف زمان و انرژی می‌شود. حتی اگر این افزایش انرژی قابل توجه و سربار ناچیز باشد، یک نقطه‌ی تعادل وجود خواهد داشت. شکل ۱۱ روش پیش‌بینی با پردازش GSP را نشان می‌دهد.



شکل ۱۱ نمونه‌ی روش پیش‌بینی برنامه‌های آینده

برای تشخیص مناسب‌ترین عامل K، اندازه‌گیری‌ها و مطالعاتی انجام شد تا هزینه‌ی اضافی (سربار) به ازای Kهای مختلف به دست آید. به عنوان مثال ترتیب Mozilla 20 mn، Word 17 mn، VLC 47 mn، 2048 7 mn را انتخاب می‌کنیم. اگر Mozilla برنامه‌ی در حال اجرا در حال حاضر باشد، قوانین GSP باز کردن Word، یا Word، VLC و یا Word، VLC، ۲۰۴۸ را بر اساس K انتخاب شده پیش‌بینی می‌کند. هزینه‌ی اجرای قوانین GSP از نظر انرژی w/s ۲ خواهد بود.

- با انتخاب K = 2 هزینه سربار حدود w/s ۶ خواهد بود.
- با انتخاب K = 4 هزینه سربار حدود w/s ۲ خواهد بود.

تفاوت w/s ۴ی در ۷۱ دقیقه ناچیز است. این تفاوت ما را بر آن داشت تا برای دقت بیشتر و با یک سربار ناچیز K = 2 را انتخاب کنیم.

دقت پیش‌بینی برنامه‌ها با افزایش داده‌های جمع آوری شده افزایش می‌یابد. بنابراین با گذر زمان دقت زیاد می‌شود. عامل دومی که دقت را افزایش می‌دهد، قواعد رفتار کاربر است. در واقع وقتی کاربر برنامه‌های ثابتی را در یک شرایطِ مشخص (که در این گزارش با عنوان محتوا از آن یاد شده است.) اجرا می‌کند، دقت بیشتر می‌شود و برعکس. افزایش این دقت در حال توسعه است.

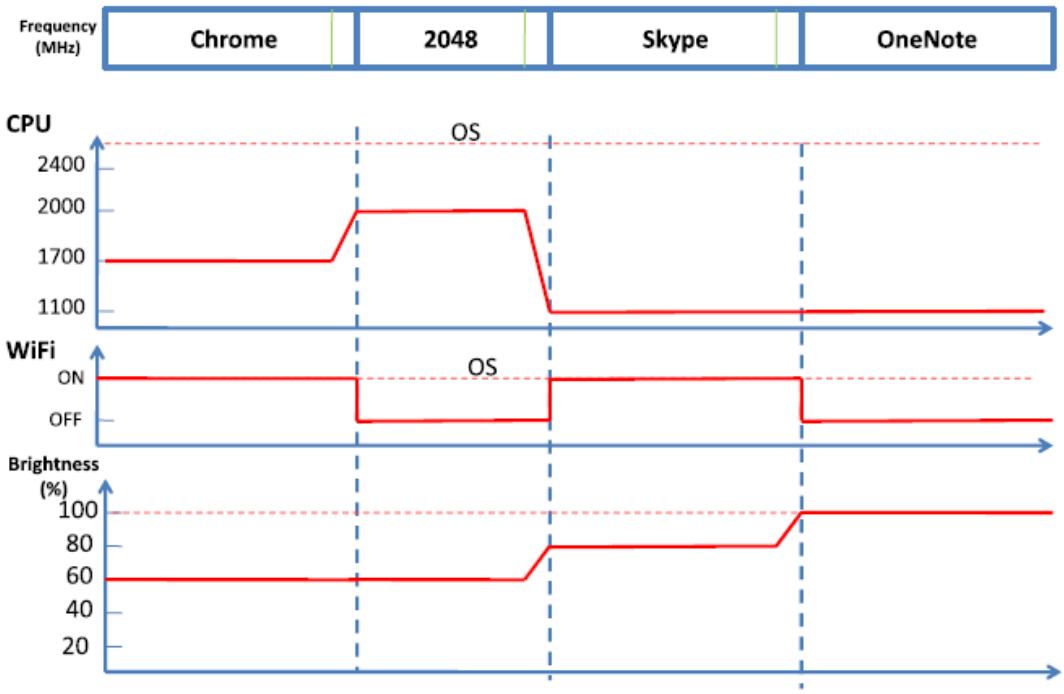
گام بعدی عملی کردن مدیریت انرژی بر اساس فاز پیش‌بینی، زمان سپری شده‌ی اجرای برنامه و یک طبقه-بندی منابع است.

۴ – ۲ – ۳ فعال کننده‌ی بهینه‌سازی

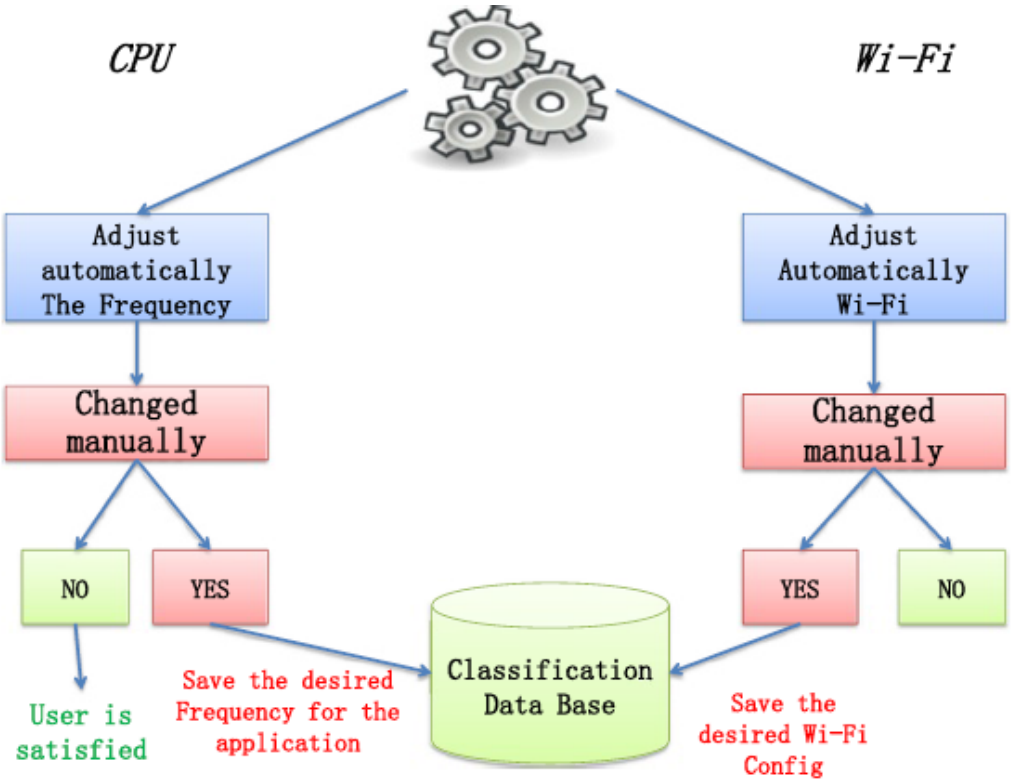
فعال کننده‌ی بهینه‌سازی تنظیم کردن منابع را برای برنامه‌ی «ب» به طور تدریجی پیش از پایان برنامه‌ی «الف» شروع می‌کند تا تجربه‌ی کاربری کاربر تحت تاثیر قرار نگیرد.

در واقع یک تغییر ناگهانی در منابع ممکن است تاثیر زیادی بر رضایت کاربر داشته باشد و ممکن است وی با افزایش دستی منابع مثل درخشندگی صفحه یا غیره به آن پاسخ دهد. در این گزارش برای مدیریت Wi-Fi تنها از فاز طبقه بندی استفاده می‌کنیم. شکل ۱۲ کاربرد پیش‌بینی را نشان می‌دهد.

رضایت کاربر در فعال کننده‌ی بهینه‌سازی مورد توجه است. برای کارایی پردازنده وقتی که فرکانس مدرج شد، فعال کننده‌ی بهینه‌سازی، پذیرش فرکانس ارائه شده را توسط کاربر بررسی می‌کند. اگر فرکانس جدیدی توسط کاربر تنظیم شود، این مقدار جدید ذخیره شده و در دفعه‌ی بعدی اجرای این برنامه در این محتوای مشخص مد نظر قرار داده خواهد شد (شکل ۱۳).



شکل ۱۲ تنظیم منابع. تاثیر فاز پیش‌بینی



شکل ۱۳ پارامتر رضایت کاربر در تنظیم منابع. نمایش چگونگی توجه فعال کننده‌ی بهینه سازی به پارامتر رضایت کاربر

۵ آزمایش‌ها و نتایج

آزمایش‌های ما در این گزارش به دو بخش تقسیم می‌شوند. در بخش اول به ارائه‌ی ابزارها و محیط آزمایشگاهی می‌پردازیم و در بخش دوم نتایج آزمایش را بررسی می‌کنیم. آزمایش‌ها بر روی یک الترابوک با پردازنده‌ی دو هسته‌ای core i7-3667U با فرکانس ۲٫۵ گیگاهرتز و ۴ گیگاهرتز RAM انجام شده‌اند. روش ارائه شده عمومی بوده و بر روی سایر پلتفرم‌ها مثل گوشی‌های هوشمند، تبلت‌ها و لپ‌تاپ‌ها قابل اجرا است. علت انتخاب الترابوک، نمایش امکان پذیری روش ما و سادگی اتصال آن به دستگاه اندازه‌گیری ماست. در این گزارش از دستگاه Yokogawa WT210 [۱۶] برای اندازه‌گیری استفاده شده است.

این الترابوک علاوه بر صفحه‌ی لمسی دارای یک پورت سیم کارت نیز بوده و Windows 8.1 بر روی آن نصب است. وجود فروشگاه ویندوز و قابلیت metro در این نسخه در کنار سایر ویژگی‌های این الترابوک معماری آن را شبیه به تبلت‌ها و گوشی‌های هوشمند کرده است.

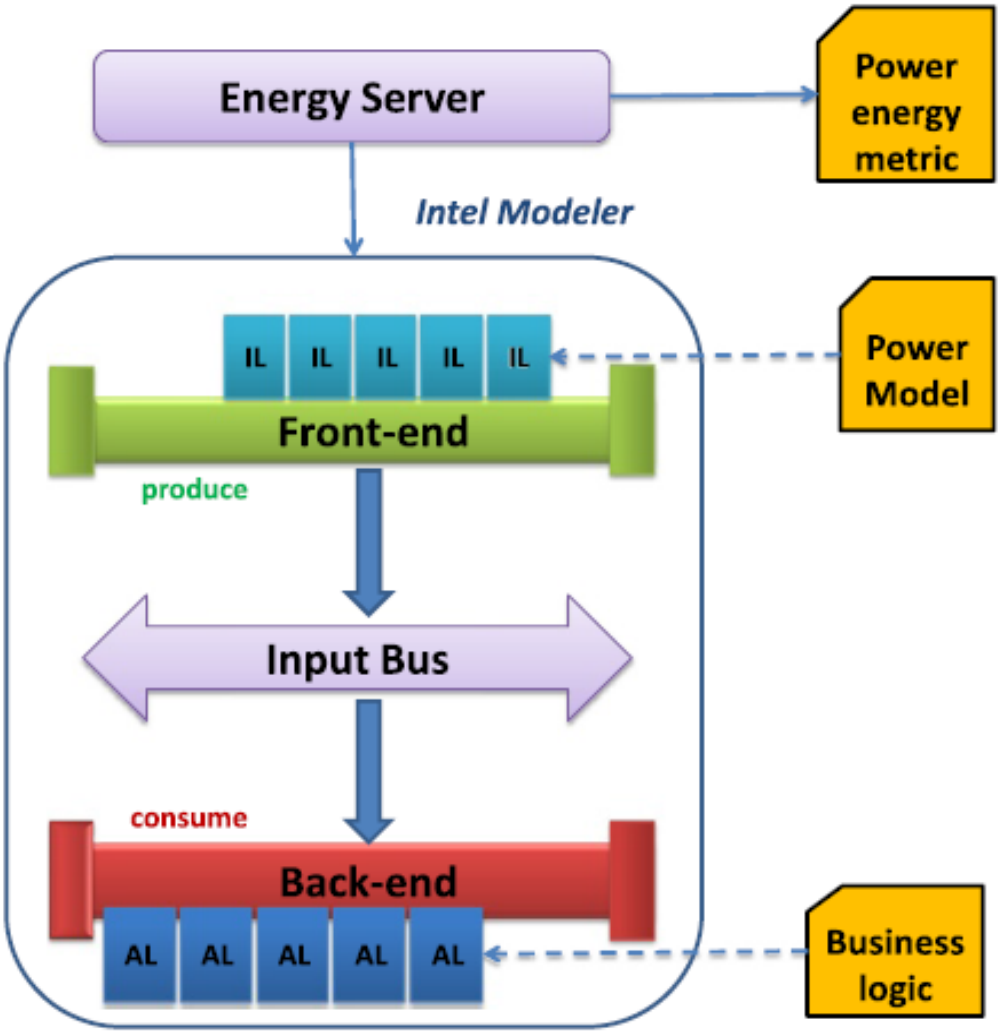
۵ – ۱ ابزارها و محیط آزمایشگاهی

در این بخش به معرفی ابزارها و محیط انجام آزمایش و کاربرد آن‌ها می‌پردازیم. شکل ۱۴ تصویری از آن به دست می‌دهد. ما برای پیاده‌سازی روش خود از چک کننده‌ی انرژی SDKit (iESDK) [۱۷] استفاده کرده‌ایم. این SDK برای اندازه‌گیری و بهینه‌سازی مصرف انرژی نرم افزار طراحی شده است. در این تحقیق از دو مؤلفه‌ی درایور اصلی (سرور انرژی ESRV) و مدل کننده‌ی این SDK سود می‌بریم. مدل کننده، سرویس‌های لازم برای پیاده‌سازی جمع آوری داده و ایده‌های ذخیره‌ی انرژی را فراهم می‌کند. تعدادی ماژول توسعه‌ی جمع‌آوری داده، کتاب‌خانه‌های ورودی‌های .a.k.a (ILs) و کتاب‌خانه‌های فعال کننده‌ها (ALs) توسعه داده شده‌اند. این مدل کننده از سه بخش جلویی (front-end)، عقبی (back-end) و گذرگاه ورودی (IB) تشکیل شده است که در زیر به شرح هر یک می‌پردازیم.

- بخش جلویی داده‌ها را جمع آوری می‌کند: استفاده از پردازنده، روشنایی صفحه، میزان باتری، برنامه-های باز و غیره. هر مقیاس یک ورودی خوانده می‌شود. جمع آوری مقیاس‌های جدید به توسعه‌ی یک IL نیاز خواهد داشت.
- وقتی که داده‌ها جمع شدند، مقیاس‌ها بر روی گذرگاه ورودی قرار داده می‌شوند. هر عنصر متصل به گذرگاه به طور مستقیم به مقیاس‌ها دسترسی دارد. گذرگاه ورودی‌ها رابط اصلی بین بخش جلویی و عقبی است.

- بخش عقبی سرویس‌های هسته مثل واقعه‌نگار، ارتباط خودکار توان به ورودی‌ها، نگهبان و مدیر ارتباطات را فراهم می‌کند. این بخش می‌تواند توسط ALها توسعه داده شود. ALها برای انجام کارهای خاصی چون سیستم عامل پویا یا پیکربندی پلتفرم طراحی شده‌اند؛ معمولا از آن‌ها برای پیاده‌سازی انواع ایده‌های بهینه‌سازی که به طور لحظه‌ای از طریق ورودی‌ها توسط بخش جلویی به ماژول می-رسند استفاده می‌شود.

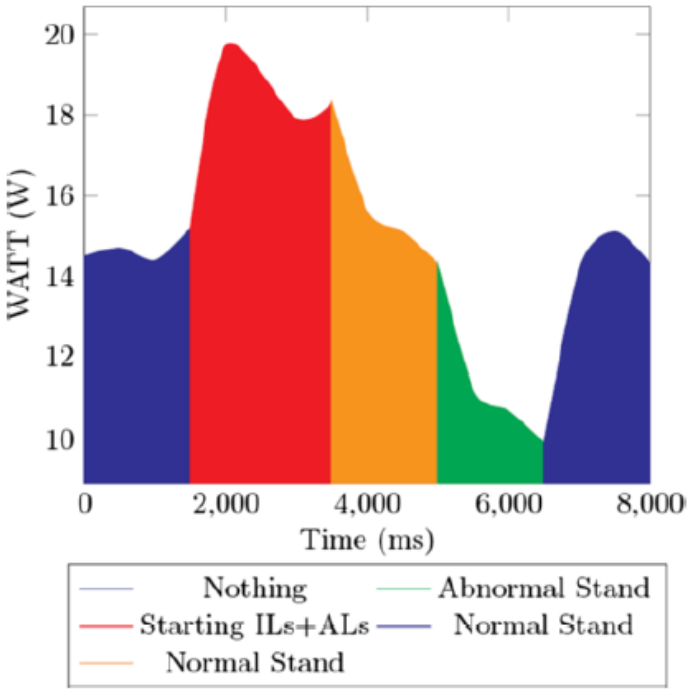
در این گزارش SCM و کاوش‌گر کاربر به عنوان IL و DHRM، GSP و فعال کننده‌ی بهینه‌سازی به عنوان AL پیاده‌سازی شده‌اند. در این آزمایش ما مصرف انرژی کل دستگاه را اندازه می‌گیریم و به یک بخش خاص بسنده نمی‌کنیم.



شکل ۱۴ ESRV- مدل کننده‌ی اینتل

۵ – ۲ ارزیابی مؤلفه‌ی بهینه‌سازی بر اساس محتوا (شرایط) (COC)

هدف از این بخش آزمایش اندازه‌گیری مصرف انرژی الترابوک در هنگام به کار گیری روش ماست. در این گزارش بر میزان روشنایی صفحه تمرکز کرده ایم که نتایج آن در شکل ۱۵ می‌بینیم. همان طور که در شکل ۱۵ می‌بینیم نمودار دارای ۵ فاز متفاوت است. فاز اول که به رنگ آبی کشیده شده است نمایش‌گر انرژی مصرفی دستگاه در حالت بی‌کار بدون بهره گرفتن از روش ارائه شده در این تحقیق است که ۱۴ وات پیش‌بینی شده است. در فاز دوم (رنگ قرمز) شاهد سربار مصرف انرژی به علت به کار انداختن روش ماست که مقدار آن حدود ۵ وات است. فاز سوم (رنگ نارنجی) نشان‌گر حالتی‌ست که الترابوک در موقعیتی مناسب برای کاربر قرار گرفته است که میزان روشنایی صفحه در آن افزایش یافته است. در فاز چهارم (رنگ سبز) می‌بینیم که مصرف انرژی به علت عدم استفاده از پردازنده کاهش یافته و میزان روشنایی صفحه هم کم شده است. توان مصرفی در این فاز حدود ۱۰ وات است. این کاهش میزان روشنایی صفحه ۳۰ درصد ذخیره‌ی انرژی را نتیجه می‌دهد. در فاز آخر (رنگ آبی) میزان روشنایی به علت بازگشت دستگاه به موقعیت معمولی بازنشانی می‌شود.

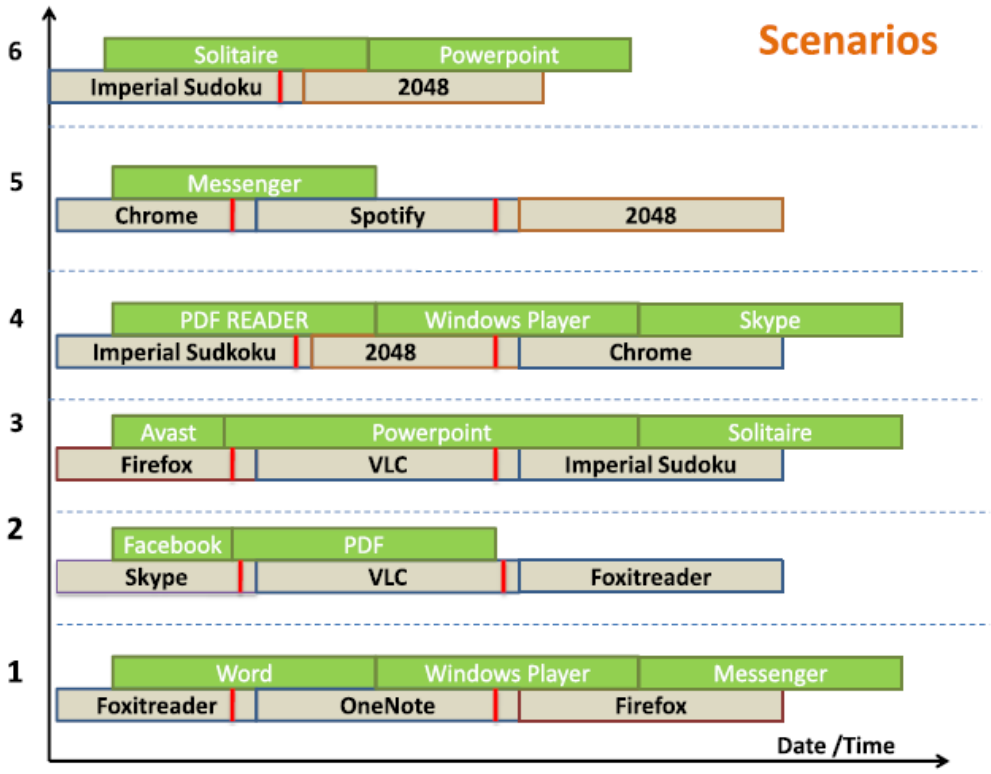


شکل ۱۵ ارزیابی مؤلفه‌ی بهینه سازی بر اساس محتوا. مصرف انرژی در چند موقعیت متفاوت

با به کارگیری مؤلفه‌ی بهینه‌سازی بر اساس محتوا قادر به ذخیره‌ی انرژی به میزان ۳۰ درصد نسبت به سیاست‌های عادی سیستم عامل شده ایم. در بخش بعد نتایج آزمایش‌های انجام شده به منظور ارزیابی کارایی این مؤلفه‌ی بهینه‌سازی را گزارش می‌کنیم.

۵ – ۳ پیش‌بینی و طبقه بندی برنامه‌ها

نتایج آزمایش‌ها بر اساس ترتیب برنامه‌های باز شده توسط ۶ کاربر است. آزمون اول برای آزمایش امکان پذیری روش ارائه شده طراحی شده است. در این بخش برای هر کاربر ترتیب برنامه‌های آینده را پیش‌بینی کرده و آن‌ها را طبقه بندی می‌کنیم. در پایان نتایج و میزان ذخیره‌ی انرژی با به کار گیری روش خود را با حالت استفاده از مدیریت انرژی عادی سیستم عامل مقایسه می‌کنیم. برای هر کاربر ترتیب برنامه‌های باز شده، برنامه‌های پس‌زمینه، زمان سپری شده در هر برنامه و طبقه بندی بر اساس میزان استفاده از پردازنده و Wi-Fi را داریم. برای نشان دادن تفاوت رفتار کاربر و برنامه‌های اجرا شده به عنوان تابعی از روز هفته، ۶ روایت متفاوت را همان‌طور که در شکل ۱۶ می‌بینیم در نظر گرفتیم. برنامه‌های سبز رنگ برنامه‌های پس‌زمینه و برنامه‌های توسی برنامه‌های باز در صفحه هستند. خط قرمز رنگ روی برنامه‌ها نشان دهنده‌ی نقطه‌ی شروع مورد انتظار برای تنظیم منابع برای برنامه‌ی بعدی‌ست که نشان دهنده‌ی ۱۰ درصد باقی مانده‌ی برنامه‌ی فعلی نیز هست. پیش از ارزیابی کارایی ایده‌ی ارائه شده، ابتدا برنامه‌ها را برای هر کاربر طبقه بندی کردیم. برای این کار از اطلاعات ضبط شده در طول دو هفته استفاده کردیم. جدول ۵ طبقه بندی بر اساس میزان استفاده از پردازنده را نشان می‌دهد. مقدار «L» کم مصرف‌ترین کلاس، «M» کلاس متوسط و «H» پرمصرف‌ترین کلاس را نشان می‌دهد. همان‌طور که در این جدول می‌بینیم برنامه‌های ثابت برای کاربران متفاوت با توجه رفتار آن‌ها با آن برنامه در کلاس‌های متفاوتی قرار گرفته‌اند.



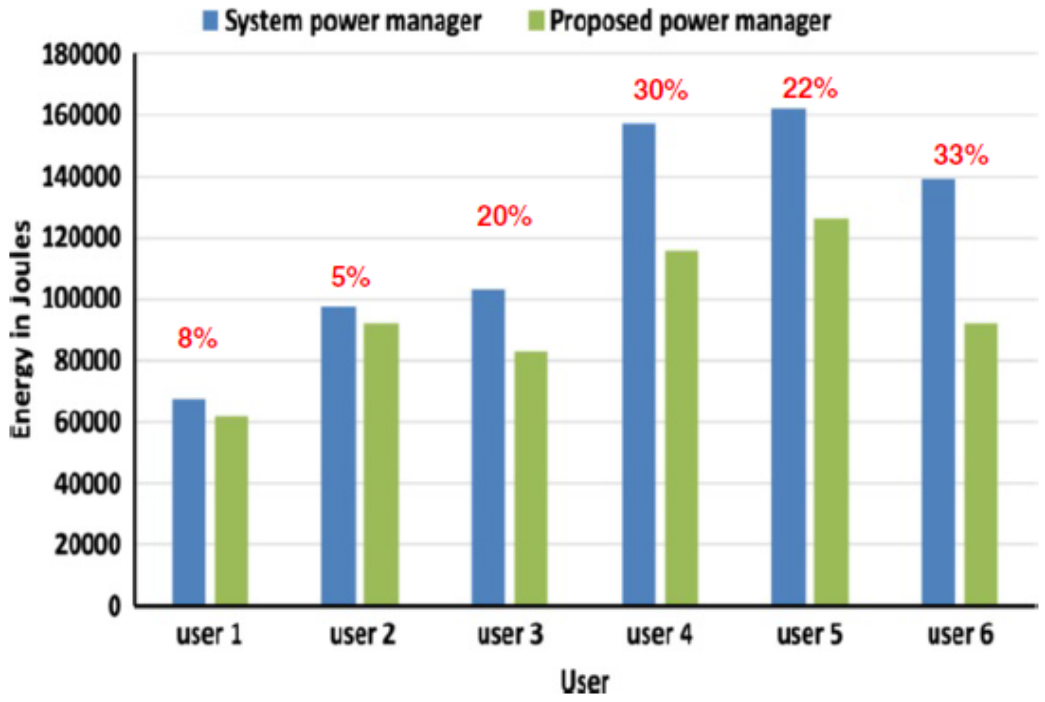
شکل ۱۶ ترتیب برنامه‌های کاربران متفاوت

طبقه بندی برنامه‌ها بر اساس نیاز به پردازنده5 جدول

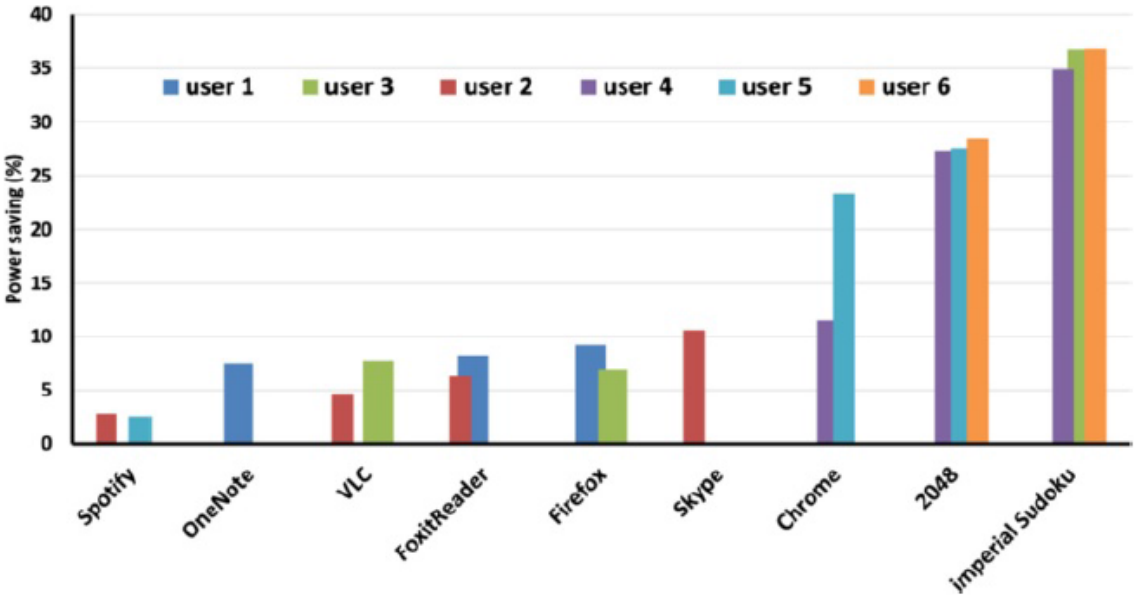
Apps	CPU application class					
	Usr1	Usr2	Usr3	Usr4	Usr5	Usr6
Firefox	*M*		*L*			
Skype		*L*				
Chrome				*L*	*M*	
Spotify		*L*			*L*	
VLC		*L*	*L*			
2048				*H*	*M*	*M*
Sudoku			*L*	*L*		*L*
Foxit	*L*	*L*				
OneNote	*L*					

۵ – ۴ ارزیابی مؤلفه‌ی بهینه‌سازی بر اساس نیاز کاربر (UNOC)

نتایج آزمایش‌های انجام شده نشان می‌دهد که به کار گیری روش ما میزان مصرف انرژی را ۳۳ درصد نسبت به حالت استفاده از مدیریت انرژی عادی سیستم عامل کاهش می‌دهد. شکل ۱۷ تاثیر راه حل ارائه شده بر ذخیره‌ی انرژی برای ۶ کاربر متفاوت را نشان می‌دهد. نتایج نشان دهنده‌ی این است که مقدار انرژی ذخیره شده برای کاربران متفاوت به شدت متفاوت است. این مقدار برای کاربر ۱ و کاربر ۲ به ترتیب ۸ و ۵ درصد هستند. برای کاربر ۳ و ۴ مقدار انرژی ذخیره شده به ترتیب ۱۹ و ۳۰ درصد است. برای کاربر پنجم این مقدار ۲۲ درصد و برای کاربر ۶ برابر با ۳۳ درصد است. این تفاوت‌ها به علت تفاوت برنامه‌های در حال اجرا و روش تعامل هر کاربر با برنامه‌هاست. برای مثال کاربر ۴ و ۵ برنامه‌های ۲۰۴۸ و chrome را اجرا می‌کنند اما ذخیره‌ی انرژی برای کاربر ۴ به اندازه‌ی ۸ درصد بیشتر است. این تفاوت به علت تفاوت کلاس‌های این دو برنامه برای دو کاربر ۴ و ۵ است. نتایج نشان‌گر این است که نوع برنامه‌ی اجرایی دارای اثری قابل توجه بر ذخیره‌ی انرژی‌ست. برنامه‌هایی که نیاز کمی به محاسبات دارند، به طور محدود از اتصالات استفاده می‌کنند و تعاملات کمی با کاربر دارند کم‌ترین قابلیت ذخیره‌ی انرژی را دارند. این موضوع را می‌توان در رفتار کاربر ۲ دید. برنامه‌های VLC، FoxitReader و Spotify که توسط او اجرا شده‌اند، دارای نیاز کمی به محاسبات و اتصالات دارند که کاهش مصرف انرژی را محدود می‌کنند. برای فهم تاثیر هر برنامه بر مصرف انرژی شکل ۱۸ کاهش مصرف انرژی در هر برنامه برای هر کاربر را نشان می‌دهد. نرم افزار Spotify یک برنامه از کلاس نیاز کم به پردازنده است در نتیجه کاهش توان مصرفی در هنگام اجرای این برنامه بسیار کم است (حدود ۲,۵ درصد)؛ اما برنامه‌ی OneNote که خصوصیاتی مشابه Spotify دارد اما در زمینه‌ی اتصالات پر مصرف‌تر است ۷ درصد بهینه‌سازی بیشتر را نتیجه می‌دهد.



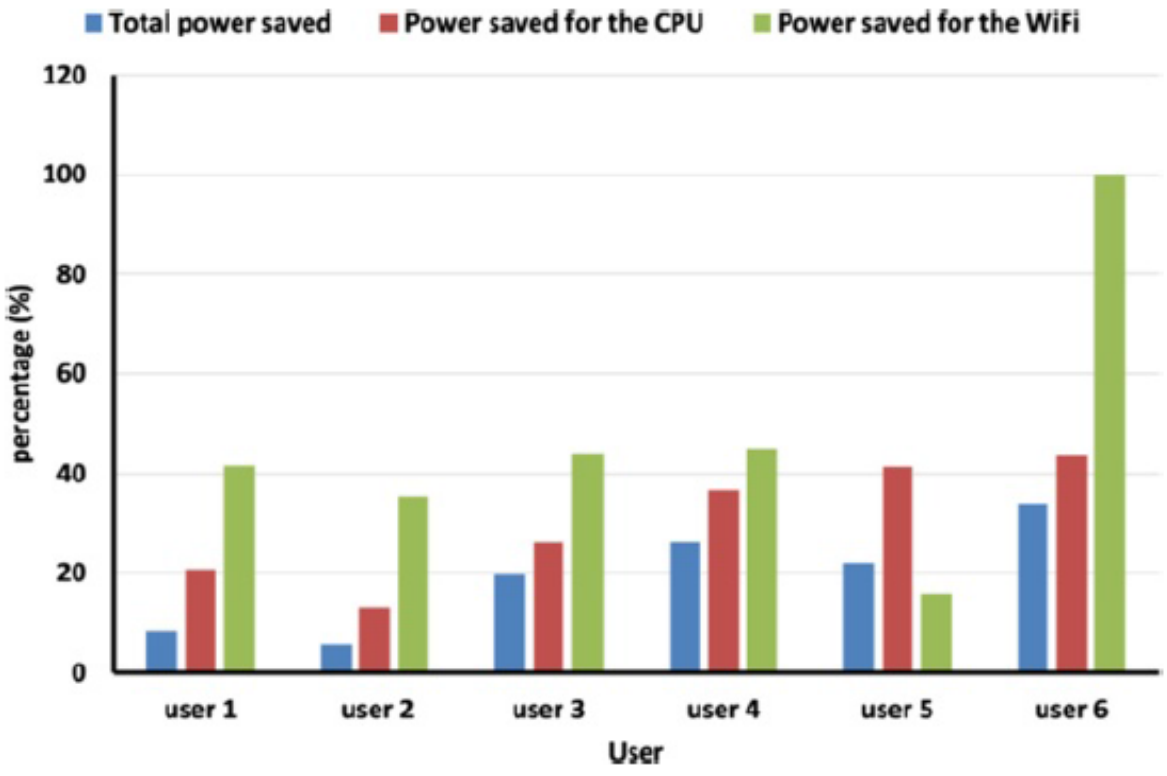
شکل ۱۷ مصرف انرژی هر کاربر



شکل ۱۸ ذخیره‌ی انرژی برای هر برنامه

نتیجه این است که برنامه‌هایی که در کلاس‌های نیاز زیاد به پردازنده قرار دارند در روش ما قابلیت ذخیره مقدار قابل توجه‌یی انرژی را دارند. Imperial Sudoku، ۲۰۴۸ و Google Chrome برنامه‌هایی از این دست هستند. هم‌چنین نتیجه می‌گیریم که ارتباطی بین تعامل با کاربر و مصرف انرژی وجود دارد. برای مثال Chrome وقتی که نیازی کمی به پردازنده دارد حدود ۱۱ درصد مصرف انرژی را کاهش می‌دهد؛ در حالی که وقتی نیاز آن به پردازنده در سطح متوسط قرار می‌گیرد تا ۲۳ درصد کاهش مصرف انرژی خواهیم داشت. شکل ۱۹ درصد کاهش مصرف انرژی توسط روش ارائه شده در این گزارش را نشان می‌دهد. با تحلیل این شکل متوجه می‌شویم:

- ذخیره‌ی انرژی برای Wi-Fi وابسته به مدت زمان سپری شده در یک برنامه‌ی خاص است. کاهش مصرف انرژی حاصل از خاموش کردن Wi-Fi مقدار ثابتی‌ست. با خاموش کردن رابط Wi-Fi کل مصرف انرژی التراوبوک به اندازه‌ی ۱٫۶ وات کاهش پیدا کرد.
- کل کاهش توان مصرفی با مدیریت پردازنده همان طور که پیشتر دیدیم وابسته به برنامه‌های در حال اجرا، رفتار کاربر و الگوی تعامل کاربر با برنامه است.



شکل ۱۹ میزان کل ذخیره‌ی انرژی به کمک روش ارائه شده

۵ – ۵ ارزیابی هزینه‌ی سربار مؤلفه‌ی بهینه‌سازی بر اساس نیاز کاربر

هزینه‌ی بهینه‌سازی توسط سربار مصرف توان، پردازنده و حافظه اندازه‌گیری شد. این اندازه‌گیری‌ها ارائه شده‌اند تا بدانیم که روش ما بر اساس IL-ها و AL-ها نه تنها بر کارایی سیستم و تجربه‌ی کاربری کاربر تاثیر منفی ندارد، بلکه به کاربر اجازه می‌دهد تا انرژی دستگاه خود را بیشتر از آنچه سیستم عامل به طور عادی انجام می‌دهد، ذخیره کند. هزینه‌های سربار در جدول ۶ نشان داده شده‌اند. سربار مؤلفه‌ی بهینه‌سازی بر اساس نیاز کاربر ناچیز است و تنها چند ثانیه طول می‌کشد. این راه حل را می‌توان بر روی تبلت‌ها یا برای مثال iPhone 6 که مصرف توان آن‌ها ۱۲ وات است به کار برد.

در جدول ۷ برخی نتایج اضافی ارائه شده‌اند. این آزمایش‌ها مقیاس بندی فرکانس برای YouTube و Word را نشان می‌دهند. برای YouTube با استفاده از ۳۰ درصد حداکثر فرکانس پردازنده هیچ تاخیری در پاسخ‌دهی برنامه نخواهیم داشت و این کار ۱۴,۳۴ درصد کاهش انرژی مصرفی را به همراه داشت. در Word با کاهش ۷۰ درصدی فرکانس پردازنده اشکالی در کارایی به وجود نیامد در حالی که مصرف انرژی ۱۰ درصد کاهش یافت. تاکید می‌کنیم در هر دو مورد تجربه‌ی کاربر در استفاده از برنامه‌ها تحت الشعاع قرار نگرفته است.

جدول 6 سربار مؤلفه‌ی بهینه سازی بر اساس نیاز کاربر

Mechanisms	Power overhead	CPU (%)	RAM (MB)	Time (ms)
Prediction (IL)	6	0.3	3.74	2000
Optimize actuator (AL)	3.8	0.3	1.45	1500

جدول 7 نتایج مقیاس بندی پردازنده

App	Time s	CPU freq. (%)	CPU power (W)	Gain (%)	Latency (S)
Youtube	360	100	5.802	14.34	0
Youtube	360	30	4.97	14.34	0
Word	600	100	5.116	7.74	0
Word	600	50	4.72	7.74	0
Word	600	30	4.63	9.64	0

۶ کارهای مرتبط

کارهای بسیاری در زمینه‌ی بهینه‌سازی مصرف انرژی در رایانه‌های قابل حمل انجام شده است. در این بین تعداد بسیار اندکی از آن‌ها به تغییر نیازها و رفتار کاربر و برنامه‌ها برای بهینه‌سازی مصرف از منابع استفاده کرده‌اند. در این بخش ما بر کارهایی که به رفتار و تجربه‌ی کاربر توجه دارند تمرکز می‌کنیم.

یکی از نخستین کارها در این زمینه [۷] است. نویسندگان اهمیت و فایده‌ی مطالعه‌ی فعالیت‌های کاربر برای شخصی‌سازی بهینه‌سازی و کنترل توسعه‌ی بهینه‌سازی توان را نشان داده‌اند. آزمایش‌های آن‌ها بر روی یک گوشی موبایل HTC تفاوت‌های مهمی را بین رفتارهای کاربران مختلف نشان داد. آن‌ها پردازنده و صفحه را پر مصرف‌ترین اجزای دستگاه یافتند.

کارهای دیگر مثل [۸] اهمیت مطالعه‌ی فعالیت‌ها و رفتار کاربر برای بهینه‌سازی توان مصرفی در سیستم‌های قابل حمل را نشان می‌دهند. این مطالعات ارتباط بین مصرف انرژی و فعالیت‌های کاربر را نشان داده‌اند. در [۹] نویسندگان راهکاری را برای اعمال روش‌های بهینه‌سازی مختلف با توجه به تجربه‌ی کاربر ارائه داده‌اند. آن‌ها یک کنترل‌کننده‌ی فرکانس پردازنده طراحی کردند که سرعت کلاک پردازنده را در حین کار تغییر دهد. این کنترل-کننده‌ی ارائه شده زمان پاسخ کاربر و برنامه‌ها را به صورت آنلاین تحلیل و سپس از این اطلاعات برای کنترل فرکانس پردازنده استفاده می‌کند. کاهش مصرف انرژی در پردازنده با این روش به ۶۵٫۵ درصد بیشتر نسبت به کنترل‌کننده‌ی عادی اندروید رسیده است. نویسندگان هم‌چنین از مشخصات تعامل بین کاربر و سیستم برای کاهش انرژی مصرفی سود برده‌اند. آن‌ها از زمان سپری شده بین دو تعامل متوالی برای کاهش روشنایی صفحه در طول این وقفه استفاده کرده‌اند تا مصرف انرژی کلی سیستم را کاهش دهند. نویسندگان در [۱۸، ۱۹] راهکارهای بر پایه‌ی فعالیت‌های کاربر و اطلاعات محتوایی و هم‌چنین تعدادی سیاست مدیریتی برای هر جز سیستم ارائه دادند. در روش آن‌ها فرکانس پردازنده بر اساس حجم کار موجود به طور پویا تغییر می‌کند. هم‌چنین زمان زنده بودن پردازش‌های پس‌زمینه را با توجه به برخی الگوها کاهش دادند. در این راهکار ۲۰ درصد کاهش مصرف در مقایسه با روش‌های تبلیغاتی موجود حاصل شد. هرچند روش آن‌ها کاملا خودکار نبوده و به تغییر در کد منبع برنامه‌های در حال اجرا نیاز داشت.

برخی از کارها از یادگیری ماشین برای دسته‌بندی برنامه‌ها و فعالیت‌های کاربر استفاده کرده‌اند. با هدف مصرف انرژی Wi-Fi، روش ارائه شده در [۱۰] بین برنامه‌ها انتخابی را انجام می‌دهد تا به برنامه‌هایی که تعامل زیادی با شبکه دارند اولویت بیشتری بدهد. برنامه‌ها به کمک دسته‌بند SVM و با توجه به میزان ترافیک شبکه‌ی مصرفی با اولویت‌های کم و زیاد دسته‌بندی می‌شوند. بر این اساس تنها ترافیک ناشی از برنامه‌های با اولویت بالا مجاز خواهند بود تا در مصرف انرژی صرفه‌جویی شود. در [۱۹] نویسندگان فعالیت‌های کاربران را بر اساس میزان

نیاز آن‌ها به روشنایی صفحه دسته‌بندی کردند و سپس این اطلاعات را به اطلاعات محتوایی مربوط کردند. سپس از راهکارهای یادگیری ماشین برای پیش‌بینی درخشندگی مورد نیاز استفاده کردند. در [۲۰] نویسندگان ناظر توان را با یک معماری سرور-مشتری توسعه دادند تا گزارش استفاده از دستگاه‌های اندرویدی را جمع‌آوری کنند. بر اساس الگوهای استفاده، پروفایل‌های ذخیره‌ی انرژی تولید شده و برای پوشش نیازهای هر دستگاهِ داخل سیستم شخصی‌سازی می‌شوند. نتایج آزمایشگاهی نشان داده است که ناظر توان می‌تواند عمر باتری را تا ۹۰ درصد افزایش دهد. هرچند این روش دارای مشکلات حریم شخصی ناشی از استخراج الگوی مصرف است. مقاله‌ی مروری [۲۱] لیستی کاربردی از مطالعات مربوطه را که از شناخت فعالیت‌های کاربر برای ذخیره‌ی انرژی استفاده می‌کنند، در خود آورده است. اکثر مطالعات لیست شده از یادگیری ماشین استفاده می‌کنند اما تعداد اندکی از آن‌ها فعالیت‌های آینده را پیش‌بینی می‌کنند.

در مقایسه با کارهای گذشته، روش ما ویژگی‌های اصلی زیر را دارد:

- روش ذخیره‌ی انرژی بیش از یک قطعه را شامل می‌شود (پردازنده، GPS، WiFi و درخشندگی).
- چهارچوبِ استفاده شده دارای یک معماریِ قسمت بندی شده است. این ماژولار بودن در کنار به کارگیری Intel Energy Checker SDK روش ما را منعطف می‌کند. اضافه کردن یک ورودی جدید برای بهبود و پیشرفت بهینه‌سازی با اضافه کردن داده‌های یک حسگر جدید یا یک فعال‌کننده‌ی جدید برای مدیریت یک سخت‌افزار جدید بسیار آسان خواهد بود.
- تنظیمات در لحظه‌ی اجرا با توجه به پیش بینی‌ها انجام می‌شوند. به علاوه بهینه‌سازی در روش ما هردوی اطلاعات محتوایی و عادات کاربر را مد نظر قرار می‌دهد.
- در مقایسه با روش ارائه شده در [۱۹] هدف اصلی یعنی کاهش مصرف انرژی در مدیریت درخشندگی صفحه نمایش حفظ می‌شود و این هدف پارامتر اصلی در راهکار ارائه شده است.

۷ نتیجه‌گیری

در این مقاله دو روش جدید برای کاهش انرژی مصرفی در سیستم‌های قابل حمل ارائه شد.

در راهکار نخست از محتوای فعلی کاربر استفاده می‌کنیم. در نسخه‌ی فعلی کار، محتوا شامل موقعیت مکانی دستگاه، درخشندگی محیط، و سر و صدای محیط است. دیگر اطلاعات و حسگرها در آینده مدنظر گرفته خواهند شد. روش ما قادر به ۳۰ درصد کاهش مصرف انرژی با هزینه‌ی سربار بسیار کم است که آن هم تنها در طی شروع IL-ها و AL-ها در هنگام بالا آمدن سیستم است (حدود ۱ وات در دو ثانیه). هم‌چنین روش ما اثری بر روی حافظه‌ی ذخیره‌سازی و پردازش‌ها ندارد.

راهکار دوم قدرتمندتر از راهکار اول است اما سربارهایی را در زمینه‌ی پردازش و حافظه به همراه دارد. این راهکار بر پایه‌ی روش‌های استخراج داده و یادگیری ماشین است. بر خلاف روش‌های موجود که نیازها و رفتار کاربران به ندرت مورد توجه قرار می‌گیرند، در راهکار دومِ ما، نه تنها به این موارد توجه می‌شود، بلکه برنامه‌های محتمل به اجرا را از نظر نیاز به منابع دسته‌بندی می‌کنیم و برنامه‌های آینده را پیش‌بینی می‌کنیم.

در مقایسه با روش مدیریت انرژی پیشرفته‌ی ویندوز ۸٫۱ برای برخی شرایط بهبود بهینه‌سازی در روش ما به ۳۰ درصد می‌رسد.

از دیگر نکات می‌توان به در نظر گرفتن رضایت کاربر به عنوان یکی از پارامترهای کنترل بهینه‌سازی اشاره کرد. راهکار پیش‌بینی برنامه‌های آینده را نیز می‌توان در کارهای آینده برای افزایش دقت توسعه داد. ماژول DHRM را می‌توان با توجه به پارامترهای ذکر شده در [۱۹] مثل مقدار انرژی باقی مانده در باتری بهبود بخشید. الگوریتم استفاده شده برای یادگیری ترتیب برنامه‌ها GSP بوده است که بهترین الگوریتم موجود نیست و می‌توان راهکارهای بهتری را برای آن یافت.

از آنجا که برنامه‌ها نیازهای متعدد و متفاوتی در فازهای مختلف اجرای خود دارند، به نظر بهتر خواهد بود که فازهای برنامه‌ها را به جای یک برنامه‌ی جدا در درون خود آن‌ها مورد توجه قرار بدهیم. هم‌چنین شاید بتوان با عمومی کردن برخی قسمت‌های بهینه‌سازی و استفاده از اطلاعات سایر کاربران به کمک اینترنت و توسعه‌ی سرور به بهبود عملکرد بهینه‌سازی رسید.

منابع

[۱] http://uk.businessinsider.com/smartphone-andtablet-penetration-2013-10?r=US&IR=T.

[۲] https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

[۳] Narseo Vallina-Rodriguez and Jon Crowcroft. “Energy Management Techniques in Modern Mobile Handsets”. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, 2012.

[۴] S. Frattasi, H. Fathi, F. Fitzek, R. Prasad, and M. Katz, “Defining 4G technology from the users perspective,” Network, IEEE, vol. 20, no. 1, pp. 35 –41, jan.-feb. 2006.

[۵] Wibree, http://www.bluetooth.com/Pages/Low-Energy.aspx.

[۶] Z. Shelby and C. Bormann, “6lowpan: The wireless embedded internet, wiley,” 2009.

[۷] A Shye, B Scholbrock, G Memik, PA Dinda, in Proceed. of the ACM SIGMETRICS Int. Conf. on Measurement and Modeling of Computer Systems. SIGMETRICS ’10. Characterizing and modeling user activity on smartphones: summary (ACM, New York, NY, USA, 2010), pp. 375–376

[۸] C Bunse, in 28th International Conference on Informatics for Environmental Protection: ICT for Energy Effieciency, EnviroInfo 2014, Oldenburg, Germany, September 10-12, 2014. On the impact of user feedback on energy consumption (ACM, 2014), pp. 759–764

[۹] W Song, N Sung, B-G Chun, J Kim, in Proceed. of the 15thWorkshop on Mobile Computing Systems and Applications. HotMobile ’14. Reducing energy consumption of smartphones using user-perceived response time analysis (ACM, New York, NY, USA, 2014), pp. 20–1206

[۱۰] AJ Pyles, X Qi, G Zhou, M Keally, X Liu, in Proceedings of the 2012 ACM conference on ubiquitous computing. SAPSM: Smart adaptive 802.11 PSM for smartphones (ACM, New York, 2012), pp. 11–20

[۱۱] G Semeraro, G Magklis, R Balasubramonian, DH Albonesi, S Dwarkadas, ML Scott, in Proceedings of the 8th International Symposium on High-Performance Computer Architecture. HPCA ’02. Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling (IEEE Computer Society, Washington, DC, USA, 2002), p. 29

[۱۲] Ismat Chaib Draa, Smail Niar, Jamel Tayeb, Emmanuelle Grislin and Mikael Desertot. “Sensing user context and habits for run-time energy optimization”. *EURASIP journal on Embedded Systems*, Springer, 2016

[۱۳] CH Mooney, JF Roddick, Sequential pattern mining—approaches and algorithms. ACM Comput. Surv. 45(2), 1–39 (2013)

[۱۴] R Srikant, R Agarwal, in Proceed. of the 5th Int. Conf. on EDT: Advances in Database Technology. Mining sequential patterns: generalizations and performance improvements (ACM, 1996), pp. 3–17

[۱۵] Y Hirate, H Yamana, Generalized sequential pattern mining with item intervals. J. Comput. 1(3), 51–60 (2006)

[۱۶] Power Meter Yokogawa WT210. http://www.electro-meters.com/yokogawa/yokogawa-power-meters/wt210/. Accessed 2010

[۱۷] Intel Energy Checker Software Development Kit UserGuide. http://www.greencodelab.fr/content/intel-energy-checker-tutoriel-0. Accessed 2010

[۱۸] SK Datta, C Bonnet, N Nikaein, in Consumer Electronics (ISCE), 2013 IEEE 17th International Symposium On. Power monitor v2: novel power saving android application, (2013), pp. 253–254

[۱۹] M Schuchhardt, S Jha, R Ayoub, M Kishinevsky, G Memik, in Proceed. of the 2014 Int. Conf. on Compilers, Architecture and Synthesis for Embedded Systems. CASES '14. Caped: context-aware personalized display brightness for mobile devices (ACM, New York, NY, USA, 2014), pp. 19–11910

[۲۰] SK Datta, C Bonnet, N Nikaein, in Wireless andMobile Networking Conference (WMNC), 2014 7th IFIP. Personalized power saving profiles generation analyzing smart device usage patterns (IEEE, Vilamoura, Portugal, 2014), pp. 1–8

[۲۱] D Gordon, J Czerny, M Beigl, Activity recognition for creatures of habit. Pers. Ubiquit. Comput. 18(1), 205–221 (2014)