ابزارهای یادگیری ماشین مبتنی بر ابر برای نرم افزارهای پیشرفته داده های بزرگ

چکیده

ما ابزارهای یادگیری ماشین مبتنی‌بر ابر را برای افزایش برنامه‌های کاربردی داده‌های بزرگ پیشنهاد می‌کنیم، که در آن ایده‌ی اصلی پیش‌بینی حجم کار "بعدی" در برابر هدف زیرساخت ابر از طریق یک رویکرد مبتنی بر گروه‌های نوآورانه است که اثر بخشی طبقه‌بندی‌های مختلف شناخته شده به منظور افزایش طیف دقت طبقه بندی نهایی ترکیب می‌کند، که درحال حاضر بسیار به زمینه‌ای خاص از داده‌های بزرگ مربوط است. به اصطلاح مشکل طبقه بندی حجم کار در جهت بهبود بهره وری و قابلیت اطمینان برنامه‌های کاربردی داده‌ی بزرگ مبتنی بر ابر نقش حیاتی ایفا می‌کند. اجرای عاقلانه‌ی روش ما نهادهای ابر را مستقر می‌کند که روش طبقه‌بندی در بالای ماشین‌های مجازی، که یک نمایش مناسب برای داده‌های بزرگ مبتنی بر ابراست. ارزیابی مقدماتی و تجزیه و تحلیل، به وضوح منافع حاصل از طبقه بندی چارچوب را تایید می‌کند.

1. مقدمه

در این مقاله، ابزارهای یادگیری ماشین مبتنی‌بر ابر، برای افزایش برنامه های کاربردی داده های بزرگ پیشنهاد می‌کنیم (به‌عنوان مثال، [29]، [6]، [17])، که در آن ایده اصلی پیش‌بینی حجم کار "بعدی" با هدف زیرساخت ابر از طریق گروه های نوآورانه است (به‌عنوان مثال، [35]) روش ترکیب اثر طبقه بندی‌های مختلف به منظور افزایش طیف دقت طبقه بندی نهایی شناخته شده است، که در حال حاضر به زمینه‌ای خاص از داده‌های بزرگ مربوط می‌شود (به‌عنوان

مثال، [16]). به اصطلاح مشکل طبقه بندی حجم کار نقش مهمی در جهت بهبود بهره وری و نمایش قابلیت اطمینان برنامه های کاربردی داده های بزرگ مبتنی بر ابر ایفا می‌کند (به‌عنوان مثال، [47]، [48]). اجرای عاقلانه‌ی این روش، نهادهای ابر را به منظور طبقه بندی توزیع شده در بالای ماشین های مجازی مستقر می‌کند (به‌عنوان مثال، [23])، که نشان دهنده‌ی وسیله ای مناسب برای داده های بزرگ مبتنی بر ابر است.

تکنولوژی مجازی سازی در محیط های محاسباتی مدرن مانند محاسبات ابری [8]، [11]، [18]، [7] و مزارع سرور [41]، [19] نقش اساسی دارد. با اجرای تعدادی ماشین مجازی در سخت افزار یکسان، مجازی سازی به ما اجازه می‌دهد تا استفاده‌ی بهینه‌ای از منابع سخت افزاری دردسترس انجام گیرد. علاوه بر این، مجازی سازی مزایایی همچون امنیت، قابلیت اطمینان، مقیاس پذیری و مدیریت منابع را برای ما به ارمغان می‌آورد (به‌عنوان مثال، [9]، [42]، [10]). مدیریت منابع در محیط مجازی می‌تواند با طبقه بندی حجم کار کاربرد های مجازی انجام شود (به عنوان مثال، [50]). در نتیجه، خصوصیات حجم کار به‌طور گسترده در طول گذشته مورد بررسی قرار گرفته و تحقیقاتی زیادی انجام شده است (به‌عنوان مثال، [12]، [4]). اخیرا، برخی از کارها به سمت خصوصیات حجم کار در محیط‌های مرکز داده انجام شده است [21]. از سوی دیگر، مدل‌سازی حجم کار و پیش‌بینی در محیط‌های مجازی سازی شده در [20] [22]، [3] بیان شده‌اند، درحالی که حفظ تعادل حجم کار مجازی شده در [24]، [46] نشان داده شده است.

از نقطه نظر روش، طبقه بندی حجم کار، یک کار حیاتی است که قبلا ذکر شد و با جمع‌آوری معیارهای مناسب در طول اجرای برنامه‌های کاربردی مرجع همراه است و در حال اجرای یک الگوی طبقه‌بندی بر روی داده‌های جمع آوری شده است، که باعث می‌شود میان طبقات مختلف تفاوت قائل شویم. در سطح پایه، حجم کار می‌تواند به‌عنوان CPU متمرکز و یا I / O متمرکز باشد. در [27]، هو و همکارانش برنامه‌ریزی‌های نامتقارن ماشین مجازی بر اساس سطح پایه‌ی طبقه‌بندی را انجام دادند. در سطح نهایی، حجم‌کار می‌تواند به عنوان CPU متمرکز، حافظه متمرکز، دیسک خواندن/ نوشتن متمرکز و شبکه متمرکز باشد. ژائو و همکارانش [50] یک مدل طبقه‌بندی بر اساس چندین سطح طبقه‌بندی را توصیف کردند. در [49]، ژانگ و همکارانش مشکل انتخاب خودکار معیارهایی که بهترین دقت در

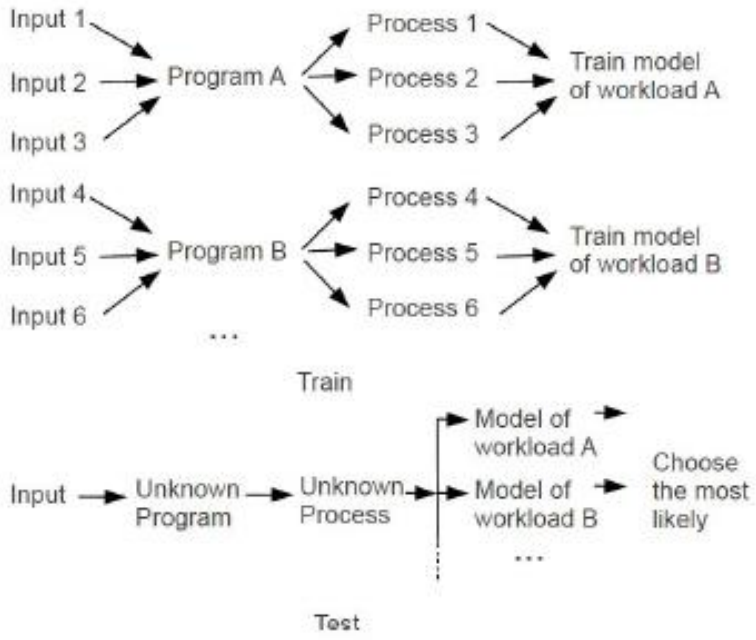
طبقه‌بندی را ارائه می‌کنند، نشان داده است. همچنین، حجم‌کاری را که می‌تواند با توجه به منابع حافظه به عنوان سیگنال طبقه‌بندی شود مورد مطالعه قرار داده است، که می‌تواند با استفاده از پارامترهای طیفی مورد تجزیه و تحلیل قرار گیرد (به‌عنوان مثال، [39]، [31]). نتایج در آماده‌سازی ماشین‌ها و در شبیه‌سازی نشان می‌دهد که طبقه‌بندی مدل پنهان مارکوف (HMM) [5] می‌تواند برای مدل ارجاع به حافظه ایجاد و توسط فرآیندهای در حال اجرا مدیریت شود.

در پژوهش پیشنهادی ما، فاز طبقه‌بندی به‌صورت زیر است. ابتدا، در یک محیط مجازی برخی از برنامه‌های مرجع را اجرا می‌کنیم (در این کار، از معیار شناخته شده‌ی 2006SPEC CINT [40] استفاده می‌کنیم)، و پس از اجرا آن، برخی از ویژگی‌ها را با استفاده از رابط‌های برنامه کاربردی از مانیتور ماشین مجازی استخراج می‌کنیم. با این ویژگی‌های جمع‌آوری شده، یک مدل از حجم کار برای هر یک از برنامه‌ها با توجه به الگوریتم‌های یادگیری ماشین ارائه می‌کنیم. برنامه‌های ناشناخته در محیط یکسانی اجرا می‌شوند، و ویژگی‌های خود را به مدل‌های مرجع حجم کار، جهت پیدا کردن این باور که حجم کار ناشناخته می‌تواند با هر یک از مدل‌ها همراه شود، می‌دهند. در نهایت، نظرات به‌دست آمده با استفاده از الگوریتم‌های طبقه‌بندی مختلف با استفاده از قانون Dempster-Shafer [36] به‌منظور استخراج یک طبقه‌بندی با کیفیت بالاتر ادغام می‌شوند.

به‌طورخاص، ما در میان بارهای برنامه تبعیض قائل می‌شویم. در واقع، ما معیار 2006SPEC را تحت سیستم عامل مجازی اجرا می‌کنیم و برخی از ویژگی‌ها را از طریق مانیتور ماشین مجازی جمع‌آوری می‌کنیم. با استفاده از الگوریتم‌های یادگیری ماشین یک مدل برای هر حجم کار توسعه می‌دهیم. حجم کار ناشناخته در میان مدل‌های مختلف طبقه‌بندی می‌شود. طبقه‌بندی در میان حجم کار در حال اجرا در مجازی‌سازی کاربردهای بالقوه‌ی جالبی ارائه می‌دهد. برای مثال، اگر معیارها مناسب انتخاب شوند، ممکن است ویژگی‌های اصلی فرآیندهای در حال اجرا در ماشین مجازی مشخص شوند. احتمال دیگر این است که بدانید که چه فرآیندهایی هنگام اجرای مشتری مورد نیاز است. دیگر برنامه‌های کاربردی ممکن در این مورد، تشخیص نرم‌افزارهای مخرب [26] می‌باشد. در این رابطه، فرآیندهای در حال

اجرا می‌توانند برای دیدن حجم کار تحت نظارت قرار گیرند اگر حجم کار یکسان باشد و یا در طول زمان تغییر کند. ارزیابی مقدماتی و تجزیه و تحلیل تجربی به‌وضوح مزایای ناشی از چارچوب طبقه بندی ارائه شده را تایید می‌کند.

2. اصول عملیاتی

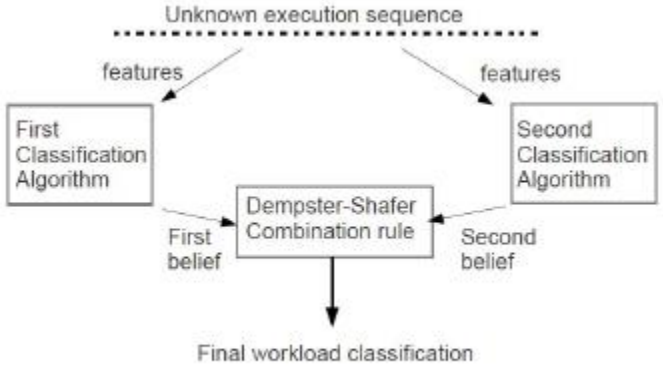


شکل 1: آموزش و آزمون مدل حجم کار

آموزش و آزمایش مراحل الگوریتم طبقه‌بندی در شکل 1 توصیف شده است. ایده‌ی پشت آموزش، استفاده از دنباله اجراهای مختلف، تولید شده توسط برنامه، زمانی که ورودی‌های مختلف را برای آموزش مدل حجم کار برنامه می‌گیرد. از سوی‌دیگر، هنگامی که دنباله اجرای ناشناخته‌ای به یک مدل حجم کار وارد می‌شود، احتمال دارد که حجم کار از دنباله‌ی ناشناخته شبیه به مدل تولید شده باشد.

ما طبقه‌بندی حجم کار را با چهار طبقه‌بند انجام دادیم، شبکه‌های عصبی (به‌عنوان مثال، [34])، مدل‌های پنهان مارکوف، نزدیکترین k همسایه [2] وARMA [28].

ما دو آزمون طبقه‌بندی انجام دادیم: ابتدا مدل‌های حجم کار را با همان شش معیار مورد استفاده برای استخراج مدل تست کردیم. بااین‌حال، داده‌های ورودی از داده‌های مورد استفاده در آموزش متفاوت بود، در نتیجه فرآیندها همیشه متفاوت هستند. در مرحله دوم، شش معیار دیگر برای محاسبه‌ی شباهت با مدل‌های حجم کار استفاده شد.



شکل 2: تلفیق داده از دو طبقه‌بند

طبقه‌بندی‌های مختلف و ویژگی‌ها می‌تواند در یک چارچوب ادغام داده به منظور بهبود دقت طبقه‌بندی در نظر گرفته شود، همان‌گونه که در شکل 2 گزارش شده است.

همان‌گونه که در پایان این مقاله نشان داده شده است، این طبقه‌بندی با بالاترین کیفیت را می‌توان برای پیدا کردن یک دسته حجم کار ناشناخته استفاده کرد.

3. معیار SPEC 2006

CINT 2006 [40] معیار CPU متمرکز، بر یک سیستم پردازنده، زیر سیستم حافظه و کامپایلر تاکید می‌کند. SPEC، CPU2006 را برای ارائه یک مقایسه از عملکرد محاسبه متمرکز طراحی. کرده است. تمام معیارها به‌عنوان کد منبع ارائه شده است. دوازده برنامه‌ی گنجانده شده در مجموعه معیارها را می‌توان در کلاس‌های زیر با توجه به قابلیت‌ها گروه‌بندی کرد: کلاس کامپایلر، کلاس بازی، کلاس فشرده‌سازی، کلاس محاسبات علمی، کلاس بهینه‌سازی.

در این کار، مدل‌های حجم کار از شش معیار مشتق شده است، بانام‌های401.bzip،403.gcc2،458.sjeng.، 471.omnetpp.، 400.perlbench و 462. .libquantum. در آزمایش اول، مدل‌های مشتق شده با همان شش معیار تست شده است. مهم است که توجه داشته باشید که داده‌های ورودی از داده‌های مورد استفاده در آموزش متفاوت هستند، در نتیجه توالی اجرا همیشه متفاوت است. در آزمایش دوم، شش معیار دیگر برای ارزیابی شباهت برای ارزیابی مدل‌های حجم کار استفاده شده است.

معیارهای مورد استفاده تنها کسری از برنامه‌های کاربردی مشابه هستند، چرا که I / O و فعالیت حافظه از دست رفته است. بنابراین، نتایج گزارش شده به عنوان یک دیدگاه اولیه از کل در نظر گرفته شده است و تنها در حجم کار متمرکز کامپیوتر است.

توضیح این‌که چطور داده‌های ورودی به معیارها به منظور ایجاد گزارش نتایج قابل تکرار سازمان‌دهی می‌شود، مهم است. SPEC شش ورودی مختلف برایbzip ، نه برای gcc و سه برای perlbench. معیار sjeng تنها یک ورودی دارد؛ دو ورودی دیگر برای sjeng از اولین موقعیت‌های شطرنج از chess.html به‌دست می‌آید که از WWW.DOWNSCRIPTS.COM/CHESSDATABASE ASP.NET-SCRIPT.HTML دانلود می‌شود. به‌طور مشابه، omnetpp تنها یک ورودی توسط SPEC دارد و ورودی‌های دیگر از اولین مثال شبکه گزارش شده در HTTP://INET.OMNETPP.ORG/DOC/INET/NEDDOC/INDEX.HTML. به‌دست می‌آید. درنهایت، libquantum از زوج اعداد اضافی زیر داده شده است: (159، 15) و (1413، 17).

تمام معیارها حداقل سه ورودی دارند که برای آموزش استفاده می‌شوند. ورودی‌های اضافی برای تست از روش مشابه به‌دست می‌آید.

4. تنظیم ماشین مجازی

زیرساخت‌های مجازی‌سازی مورد استفاده در این کار توسط نرم افزارVirtualBox [43] ارائه شده است نرم افزار VirtualBox برنامه مجازی‌سازی منبع باز است که مجموعه‌ای غنی از رابط‌های برنامه کاربردی فراهم می‌کند و

معیارهای مختلف برنامه‌ی مجازی را ارائه می‌دهد. مجموعه‌ای کامل از API های موجود در شرح داده شده است. SDK، که با VirtualBox ارائه شده، به اشخاص ثالث اجازه می‌دهد تا برنامه‌های کاربردی در تعامل با VirtualBox را توسعه دهند. VirtualBox در سطوح طراحی شده است. در پایین VMM. را توضیح داده‌ایم. VMM قلب مجازی‌سازی است، عملکرد ماشین‌های مجازی را نظارت می‌کند و امنیت و عدم وجود درگیری بین ماشین مجازی و میزبان را فراهم می‌کند. ماژول‌های وجود دارد که قابلیت‌های اضافی، به‌عنوان مثال، سرور RDP(پروتکل از راه دور دسکتاپ) را ارائه می‌کنند. سطح API بالاتر از این بلوک‌های اساسی نمایش داده شده است.

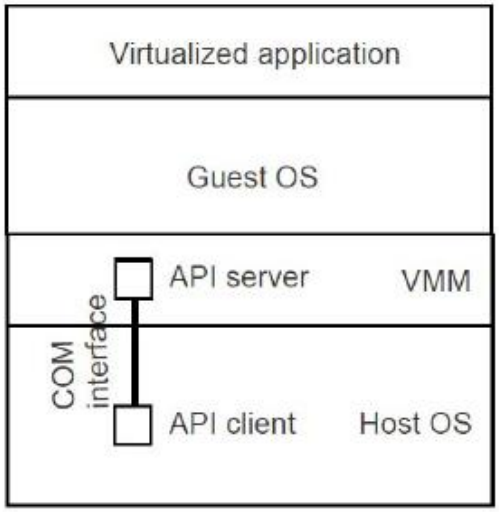
نرم افزار VirtualBox با یک وب‌سرویس همراه است، یک بار در حال اجرا، به عنوان سرور HTTP عمل می‌کند، اتصالات SOAPرا می‌پذیرد [38] و آنها را پردازش می‌کند. رابط کاربری در یک فایل زبان توصیف وب‌سرویس (WSDL) توصیف شده است [45]. نوشتن برنامه‌های مشتری در هر زبان برنامه‌نویسی ممکن است که ابزاری برای پردازش فایل‌های WSDL، مانند جاوا فراهم کرده ،++ C ، PHP NET، پایتون، پرل فراهم کرده باشد. علاوه بر جاوا و پایتون، SDK شامل کتابخانه‌هایی آماده برای استفاده است. API با استفاده از مدل شیء گرا (COM) به‌عنوان یک مدل مرجع پیاده‌سازی شده است. در ویندوز، استفاده از Micrisoft COM طبیعی است. در میزبان‌های دیگر، که COM وجود ندارد، XPCOM که یک برنامه رایگان برای اجرای COM است، استفاده می‌شود.

با وجود مزایای متعدد وب سرویس API، ما از روش COM استفاده کردیم به دلیل این‌که سربار کمتری دارد و نرخ داده‌ی بالاتری دارد. ما یک تبادل اطلاعات انجام دادیم و معلوم شد که وب سرویس قادر است، به‌طورمتوسط 5.96 MS اطلاعات جمع‌آوری کند درحالی‌که با استفاده از داده COM می‌تواند 0.49 MS جمع‌آوری کند. دیگر ویژگی‌های جالب، نگرانی مجموعه‌ای از داده‌های آماری در منابع مورد استفاده است. API توابع زیرا را فراهم می‌کند:

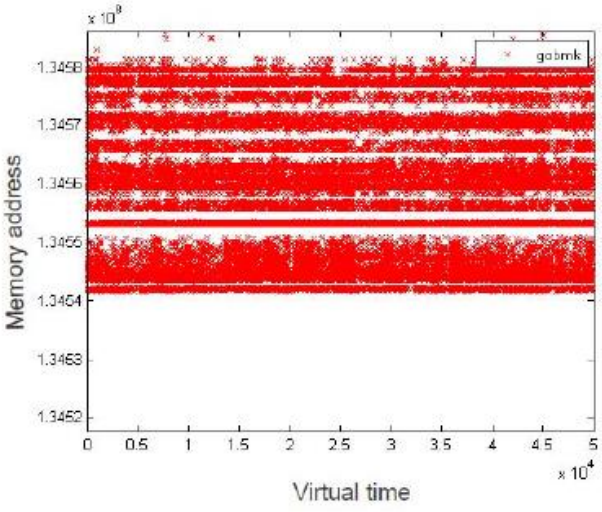
- تعیین می‌کند که کدام گروه‌ها از شاخص‌های مورد علاقه ما هستند (CPU، RAM، شبکه، دیسک).
- تنظیم محدوده اندازه‌گیری (حداقل فاصله زمانی 1)؛
- تنظیم اندازه فریم برای آمار (حداقل، حداکثر، میانگین).
- ساخت پرس و جوهایی در مورد استفاده از یک منبع واحد

5. معیارها

با توجه به معیارها، یک سیستم مالکیت با توجه به دیاگرام شکل 3 توسعه دادیم. مالکیت توسط میزبان هدایت می‌شود؛ تمام دستورات و داده‌های به‌دست آمده از رابط کاربری COM استفاده می‌کند. یک رابط وب برای VMM وجود دارد اما آن از آنچه در بالا بیان شد آهسته‌تر است.

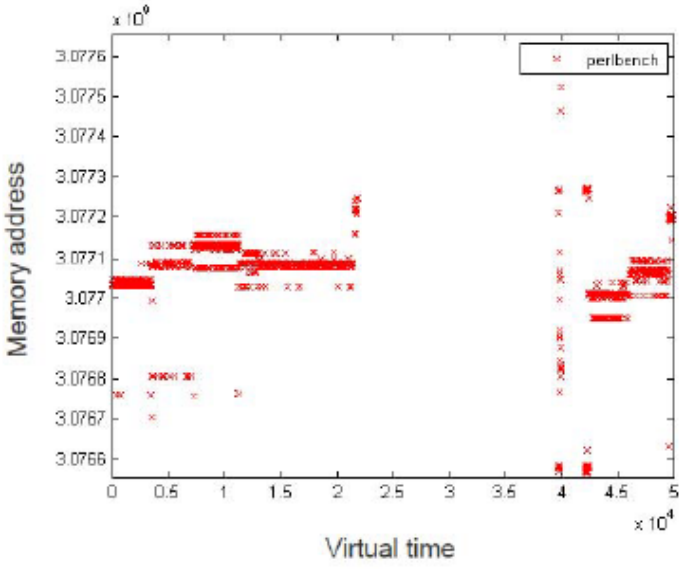


شکل 3: سیستم اکتساب



شکل 4: منابع حافظه تولید شده توسط معیار gobmk

مقادیر اندازه‌گیری برای توصیف حجم کار از دو نوع، یعنی منابع حافظه و تقاضای منبع مورد استفاده قرار می‌گیرد. هر دو مقادیر با استفاده از نرم افزار VirtualBox وکلاس‌های API VMM جمع‌آوری. شده است منابع حافظه، آدرس تولید شده توسط فرآیندهای مجازی هستند.API IMachineDebugger VMM می‌تواند ارزش شمارنده‌های برنامه مربوط به دستورالعمل در هر 0.5 میلی‌ثانیه را جمع‌آوری کند. در شکل 4، یک تکه از منابع حافظه در حال جمع‌آوری در طول اجرای مجازی از یک معیارSPEC (gobmk) است.



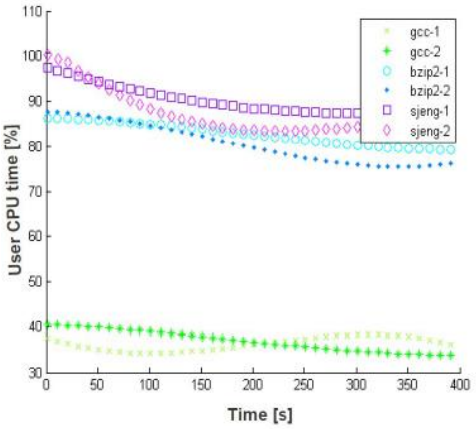
شکل 5: منابع حافظه تولید شده توسط معیار perlbench

در شکل 5، یک تکه از حافظه‌ی مرجع برای دیگر معیارSPEC گزارش شده است(perlbench).

داده‌ها در یک فرمت <address><time stamp> جمع‌آوری شده است. با استفاده از API IPerformanceCollector ویژگی تقاضای منابع تولید شده توسط فرآیندهای مجازی را جمع‌آوری می‌کنیم.

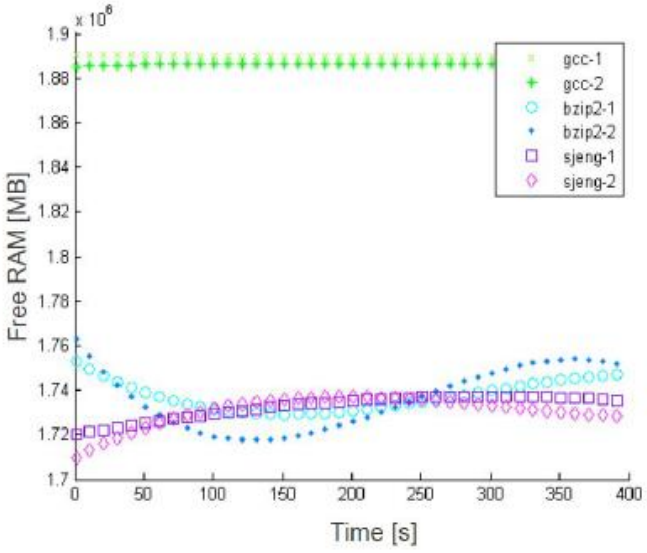
ویژگی‌های تقاضای منابع به شرح زیر است:

- پردازنده‌ی مورد استفاده در حالت کاربر.
- پردازنده‌ی مورد استفاده در حالت سیستم.
- تکه شدن حافظه (حافظه آزاد/کل حافظه).



شکل 6: زمان کاربر ویژه CPU برای برنامه‌های مختلف در دو اجرای متفاوت

در شکل 6، بخشی از 400 S از CPU استفاده می‌شود وقتی‌که حالت کاربر در طول اجرای فرآیندهای مجازی جمع‌آوری می‌کند.



شکل 7: ویژگی RAM آزاد برای برنامه‌های مختلف در دو اجرای مختلف

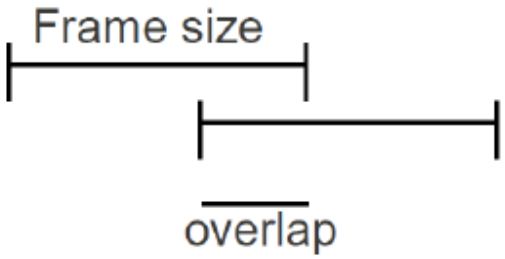
در شکل 7، میزان RAM آزاد حافظه موجود در طول اجرای مجازی فرآیند گزارش شده است.

هر منحنی مربوط به اجرای یک فرآیند مختلف مرتبط در ماشین مجازی است. همچنین ویژگی داده‌های منبع مورد تقاضا در یک فرمت time-stamp به‌دست می‌آید.

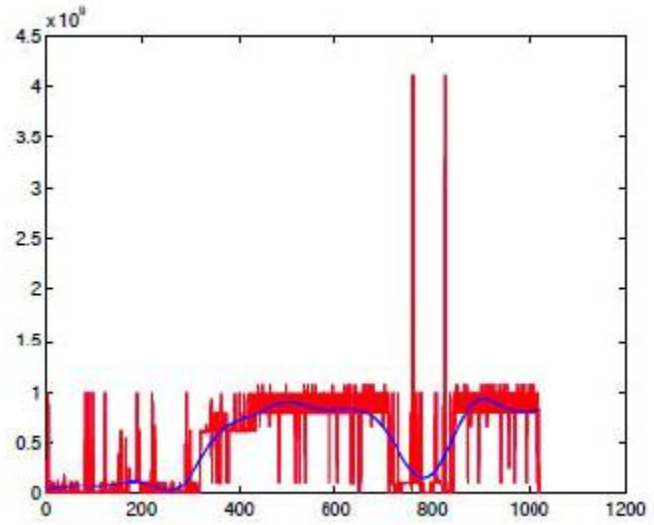
6. روش تجزیه و تحلیل داده

به‌خوبی شناخته شده است که دستورالعمل اولیه از یک کد در حال اجرا نماینده‌ی رفتار برنامه نیست. در واقع اولین میلیاردها دستورالعمل به جز فایل‌های I/O و تخصیص حافظه به عنوان ساختمان داده، قبل از رسیدن به محاسبات واقعی که توسط برنامه انجام می‌شود راه‌اندازی شده است. در این کار از تکنیک‌های کشف حالت برنامه همان‌طوری‌که در [37] توصیف شده برای پیدا کردن شروع حالت پایدار برنامه‌ها استفاده نمی‌کنیم. در عوض، کورکورانه 1 میلیارد دستورالعمل را قبل از شروع تجزیه و تحلیل داده اجرا می‌کنیم.

توالی منابع حافظه را به عنوان سیگنال یک بُعدی و دنباله تقاضای منابع را به عنوان یک سیگنال چند بُعدی در نظر می‌گیریم. به‌طور مشابه، آنچه که در سیگنال پردازش اتفاق می‌افتد، با استفاده از یک توصیف پارامتریک از توالی بیان می‌کنیم. حوادث در این روند، از جمله حلقه یا رفتارهای پی‌درپی برنامه، حوادث مهمی در توالی معیارها و پس از آن در طیف سیگنال تولید می‌کنند. به‌عنوان مثال، حلقه یک طیف حداکثری را معرفی می‌کند درحالی‌که طیف دنباله آدرس‌های پی‌درپی تولید یک جزء طیفی DC می‌کند. علاوه‌براین، توالی به صورت پویا خواص خود را تغییر می‌دهد. بنابه‌این دلایل، از توصیف طیف زمانی کوتاه در دنباله حافظه استفاده می‌کنیم. بنابراین توالی در همپوشانی تجزیه و تحلیل با توجه به اندازه تقسیم می‌شود، همانطور که در شکل 8 گزارش شده است. این قاب‌ها بیشتر به بلوک‌هایی تقسیم شده است. از این رو، به‌عنوان مثال، یک قاب از مراجع حافظه توسط مجموعه‌ای از بلوک‌ها نمایش داده می‌شود. همچنین، دنباله‌ای چند بُعدی از ویژگی‌های منابع مورد تقاضا در هم‌پوشانی فریم‌ها و بلوک‌ها تقسیم شده است.



شکل8: تحلیل فریم



شکل 9: بازسازی سیگنال با دگرگونی معکوس اولین ضرایب DCT

گام بعدی انجام تجزیه و تحلیل طیفی از بلوک‌ها است. در بین پارامترهای طیفی ممکن، یک نمایش تبدیل گسسته‌ی کسینوسی (DCT) انتخاب می‌کنیم. DCT یک عملیات پردازش سیگنال با خواص مهم شناخته شده است [25]. به‌عنوان مثال، برای کاهش افزونگی سیگنال مفید است زیرا به‌عنوان انرژی ممکن در چند ضریب ممکن (تراکم انرژی) قرار گرفته است. اولین ضرایب DCT به‌عنوان ورودی به الگوریتم طبقه‌بندی داده شده است. اثرات حفظ ضرایب DCT اول در شکل 9نشان داده شده است. یک فریم از 1024 منابع حافظه در این شکل رسم شده است. در این فریم یک تبدیل DCT انجام می‌دهیم. اولین ضریب شانزده برای به‌دست آوردن یک بردار ضریب 1،024 با صفر لایه استفاده می‌شود. توسط تبدیل معکوس این دنباله منحنی رسم شده در شکل 9 بدست می‌آید. بدیهی است که حفظ اولین ضرایب موجب صاف شدن قله‌های توالی می‌شود درحالی که هنوز رفتار توالی کلی را نشان می‌دهد و منعکس‌کننده‌ی سیگنال افزونگی کاهش اموال است.

در مورد تقاضای منابع، چون سه نوع از ویژگی‌ها در یک فریم وجود دارد، DCT به‌طور جداگانه در هر ویژگی به‌کار برده می‌شود. در هر مورد، ما تعداد کمی از ضرایب DCT در ویژگی را برای نشان دادن فریم انتخاب می‌کنیم. در این مورد، فریم به‌صورت جداگانه توسط سه جزء بردار توصیف می‌شود، که از هر ضریب DCT برای یک ویژگی تشکیل

شده است. نتیجه این است که هر بلوک از توالی شامل هر دو مراجع حافظه و منابع تقاضا است، که توسط یک عدد صحیح بیان می‌شود.

Frame size	48s
Overlap	50%
Number of DCT coeffs	16
Number of blocks per frame	40
Vector quantization	128 levels

TABLE I. FINAL PARAMETERS SETUP

جدول 1: پارامترهای نهایی SETUP

با شروع از پارامترهای تجزیه و تحلیل اولیه برخی از آزمایش‌های طبقه‌بندی شبکه‌های عصبی را، با استفاده از سه لایه پنهان و 50 سلول عصبی انجام دادیم. تجزیه و تحلیل بهینه نهایی پارامترها در جدول 1 گزارش شده است.

آ. الگوریتم‌های طبقه‌بندی

در این بخش شرح مختصری از الگوریتم‌های یادگیری ماشین که به‌عنوان طبقه‌بندی استفاده شده می‌آوریم.

1) الگوریتم kاُمین همسایه نزدیک: الگوریتم kاُمین همسایه نزدیک (K-NN) یک الگوریتم ساده‌ی یادگیری ماشین است که از هیچ مدلی در طول مرحله آموزش استفاده نمی‌کند، همان‌گونه که دیگر الگوریتم‌های یادگیری ماشین انجام می‌دهند. K- NN به اصولی استوار است که نمونه در یک مجموعه داده در مجاورت نمونه‌های دیگر است که دارای خواص مشابه هستند. اگر اشیاء با برچسب طبقه‌بندی علامت‌گذاری شوند، اشیاء با رأی اکثریت از همسایگان خود طبقه‌بندی می‌شوند و به کلاسی در میان نزدیکترین همسایگان خود تخصیص می‌یابند. K-NN از فاصله اقلیدسی برای نمایش نزدیکی به همسایه از شی طبقه‌بندی نشده استفاده می‌کند.

2) الگوریتم شبکه‌های عصبی: ما از یک شبکه عصبی با توپولوژی رو به جلو با سه لایه پنهان، که در آن جریان اطلاعات بین ورودی و خروجی یک راه برای خروج دارد استفاده می‌کنیم.

3) الگوریتم مدل پنهان مارکف: در مدل‌های پنهان مارکوف (HMM ها)، خروجی هر یک از حالت‌ها مربوط به توزیع احتمال خروجی به‌جای یک رویداد قطعی است. این است که اگر مشاهدات، توالی گسسته علامت انتخاب شده از یک الفبای محدود باشند، پس از آن برای هر حالت یک توزیع احتمال گسسته وجود دارد که فرایند تصادفی جهت مدل‌سازی را توصیف می‌کند. در HMM ها، دنباله حالات پنهان است و تنها می‌توان از طریق یکی‌دیگر از مجموعه فرآیندهای تصادفی مشاهده کرد. بنابراین، توالی حالات با یک الگوریتم مناسب ، بر اساس معیارهای بهینه‌سازی بهبود می‌یابد. مهم است که توجه داشته باشید، احتمال مشاهدات می‌تواند بردار گسسته و یا مقادیر پیوسته باشد.

4) ARMA: با توجه به توالی مرجع حافظه به عنوان یک سری زمانی از داده M_T، مدل ARMA ابزاری برای درک و شاید، پیش‌بینی مقادیر آتی در این سلسله است. مدل شامل دو بخش است، یک بخش خودبازگشت (AR) و یک بخش حرکت میانگین (MA). بنابراین، مدل است به عنوان مدل $\mathrm{ARMA(p.q)}$ بیان شده است، که در آن p بخش کاهنده و q مرتبه حرکت میانگین است:

$$M_t = c + \epsilon_t + \sum_{i=1}^{p} \varphi_i M_{t-i} + \sum_{i=1}^{q} \psi \epsilon_{t-i}$$

که در آن ψ_iوφ_i پارامترهای مدل، ϵ_t سر و صدا سفید و C ثابت است. طبقه‌بندی با مدل ARMA از مدل خطی تعمیم یافته است.

7. ادغام DEMPSTER-SHAFER

هدف از نظریهDEMPSTER-SHAFER [36]، ادغام اقدامات مختلف شواهد است. در پایه‌ی تئوری یک مجموعه متناهی از فرضیه‌های ممکن است، که می‌گوید $\theta = \{\theta_1 . 000 . \theta_k\}$

آ) تخصیص اساسی باور

تخصیص اساسی باور (BBA) می‌تواند به‌عنوان یک کلیت از یک تابع چگالی احتمال بیان شود. دقیق‌تر بیان کنیم، تخصیص اساسی باور m یک تابع است که یک مقدار در بازه‌ی [0، 1] به هر زیر مجموعه A از θ که شرایط زیر را داشته باشد اختصاص می‌دهد:

$$\sum_{A\subseteq\theta} m(A) = 1.\, m(\,) = 0$$

شایان ذکر است که $m(A)$ از زیرمجموعه A از θ، نه همه‌ی عناصر A پشتیبانی می‌کند. زیرا ما تنها می‌توانیم باور به زیرمجموعه‌ای از θ ، نه به هر فرضیه اختصاص دهیم.

ب) ترکیبی از شواهد

دو باور اساسی$(.)\ M_1$ و $(.)\ M_2$ را مشخص می‌کنیم. بنابراین $(.)\ M_1$ و $(.)\ M_2$ می‌توانند عوضی از $C \subset \theta$ مطابق فرمول زیر باشند[36]:

$$m(c) = m_1 \oplus m_2 = \frac{\sum_{j.k.A_j \cap B_k = c} m_1(A_j) m_2(B_k)}{1 - \sum_{j.k.A_j \cap B_k = 0} m_1(A_j) m_2(B_k)}$$

مخرج یک عامل طبیعی است.

ج) طبق بندی کلاس تنها براساسBBA

اگر K معیار داشته باشیم می‌توانیم از K طبقه‌بندی استفاده کنیم، هر آموزش‌دیده از فرآیندهای تولید شده توسط هر یک از K معیار استفاده می‌کند. هر طبقه‌بند به‌عنوان یک متخصص در ترکیب DS استفاده می‌شود. هدف از این طبقه‌بند پی بردن به معیار فرایند ناشناخته است.

طبقه‌بند C_i احتمال P_i را به‌عنوان خروجی فراهم می‌کند، $i = 1. \ldots, K$، که در آن p_i احتمال این است که فرایند توسط طبقه‌بند تجزیه و تحلیل شود. مجموعه فرضیه با $\theta = \{\theta_1. 000. \theta_k\}$ نشان داده شده است، θ_i رخدادی است که از پردازش معیار iاُم می‌آید، تحت این فرض، احتمال زیر مجموعه $\{\theta_i\}$ برابر است با:

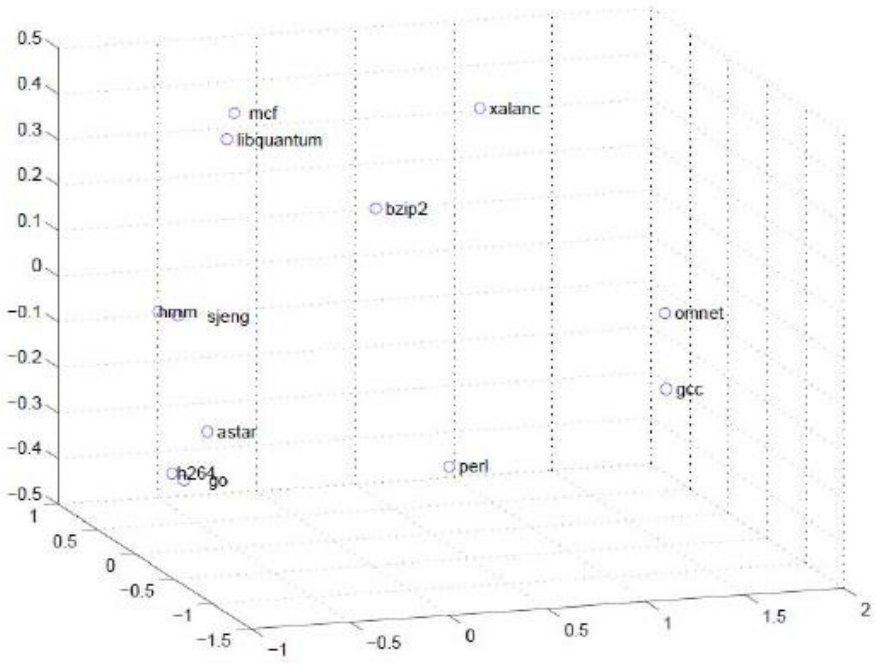
$$m_i(\{\theta_i\}) = p_i$$

$$m_i(c) = \frac{1 - p_i}{2^k - 1}$$

بر اساس این BBA، i امین طبقه‌بندی از مجموعه آموزش دیده از داده‌های تولید شده توسط معیار i و مجموعه‌ای از داده‌ها که توسط معیار i تولید نشده، استفاده می‌کند و آن‌را به‌عنوان خروجی در بازه‌ی [0،1] فراهم می‌کند، که نشان‌دهنده‌ی احتمال جریان ورودی از معیار i است.

8. تجزیه و تحلیل تجربی مقدماتی

همان‌گونه که تاکنون مشخص شده است، با استفاده از ادغام Dempster-Shafer داده‌ها، می‌توانیم یک طبقه بندی با کیفیت بالا با استفاده از طبقه بندی های با کیفیت پایین بسازیم. درحال‌حاضر، نشان می‌دهیم که این طبقه بندی با کیفیت بالا را می‌توان برای پیدا کردن دسته‌ای از حجم کار نا شناخته به‌کار برد. با آزمایش یک توالی اجرای ناشناخته با طبقه بندی آموزش دیده با حجم کار W، می‌توان نشانه‌ای از دنباله اجرای نامعلوم را که به حجم کار w اختصاص داده شده به‌دست آورد. اگر یک دنباله اجرا با حجم کار W با طبقه‌بندی با کیفیت بالا، با طبقه بند آموزش دیده با حجم کار W تست شود، خروجی نزدیک به یک خواهد بود. اگر دنباله اجرا با حجم کار مشابه با W با همان طبقه‌بندی تست شود، خروجی نزدیک به یک خواهد بود و الی آخر. این ویژگی می‌تواند برای اختصاص یک حجم کار دسته‌های دنباله اجرای ناشناخته استفاده شود. آزمایش شرح داده شده در این بخش به بررسی استفاده از فاصله میان معیار از نقطه نظر حجم کار می‌پردازد. شکل 10 یک نمای گرافیکی از فاصله میان حجم کار را نشان می‌دهد.



شکل 10: نمایش 3 بُعدی از فاصله میان برنامه‌ها

این شکل نشان می‌دهد که 429.mcf میتواند در 462.libquantum و 456.hmmer می‌تواند در458.sjeng باشد و حجم کار 445.gobmk و 464.h264ref بسیار به هم نزدیک باشند.

9. نتیجه‌گیری و کارهای آتی

در این مقاله با طبقه بندی برنامه در یک محیط مجازی سروکار داریم، به‌عنوان نمونه، بهبود کارآیی و قابلیت اطمینان برنامه های کاربردی داده‌های بزرگ مبتنی بر ابر. نشان دادیم که، با استفاده از طبقه‌بندی با کیفیت پایین‌تر، می‌توانیم یک طبقه بندی با کیفیت بالاتر با استفاده از الگوریتم های همجوشی داده ها بسازیم. ممکن است چندین کاربرد از این نوع طبقه بندی، از پروفایل کاربر تا تشخیص نرم افزارهای مخرب وجود داشته باشد. پیشرفت واضحی از کار در این مقاله برای استفاده از دیگر معیارها به منظور در برگرفتن دیگر فعالیت های حجم کار توصیف شده است. همچنین، برای بهبود بیشتر ویژگی‌های چارچوب با یکپارچه سازی راه‌حل برای مقابله با جنبه های بزرگ پردازش گسترده داده‌ها برنامه‌ریزی کرده‌ایم، در حجم بالای کار هنوز هم ممکن است، تکنیک های فشرده سازی داده‌ها (به‌عنوان مثال، [15])، روش قطعه شدن (به‌عنوان مثال، [13])، روش حفظ حریم خصوصی (به‌عنوان مثال، [14])، تعریف شده باشد، به‌ویژه، ممکن است با بسیاری از مسائل تشخیص نرم افزارهای مخرب همراه شده باشد.

REFERENCES

[1] P.S.C. Alencar, D.D. Cowan, F. McGarry and R.M. Palmer, Developing a collaborative cloud-based platform for watershed analysis and management, IEEE CollaborateCom 2014, pp. 457–459

[2] N.S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician 46(3), 1992, pp. 175-185

[3] F. Azmandian, M. Moffie, J. G. Dy, J. A. Aslam, D. R. Kaeli, Workload Characterization at the Virtualization Layer, MASCOTS 2011, pp. 63– 72

[4] P.Barford and M.Crovella, Generating representative web workload for network and server performance evaluation, ACM SIGMETRICS Performance Evaluation Review 26(1), 1998, pp. 151–169

[5] L.E. Baum, T. Petrie, Statistical Inference for Probabilistic Functions of Finite State Markov Chains, The Annals of Mathematical Statistics 37(6), 1966, pp. 1554-1563

[6] A. Biswas, S. Majumdar, B. Nandy and A. El-Haraki, Automatic Resource Provisioning: A Machine Learning Based Proactive Approach, IEEE CloudCom 2014, pp. 168–173

[7] S. Bleikertz, C. Vogel, T. Gro, Cloud radar: near real-time detection of security failures in dynamic virtualized infrastructures, ACSAC 2014, pp. 26–35

[8] G. Bruder, F. Steinicke, A. Nchter, Poster: Immersive point cloud virtual environments, 3DUI 2014, pp. 161–162

[9] M. Carlson, Systems and Virtualization Management: Standards and the Cloud. A report on SVM 2013. Journal of Network Systems Management 22(4), 2014, pp. 709–715

[10] Y. Cho, J. Choi, J. Choi, An integrated management system of virtual resources based on virtualization API and data distribution service, CAC 2013, p. 26

[11] I.-H. Chuang, Y.-T. Tsai, M.-F. Horng, Y.-H. Kuo, J.-P. Hsu, A GABased Approach for Resource Consolidation of Virtual Machines in Clouds, ACIIDS 2014, pp. 342–351

[12] W.Cime and F.Berman, A comprehensive model of the supercomputer workload, IEEE WWC 2001

[13] A. Cuzzocrea, J. Darmont and H. Mahboubi, Fragmenting very large XML data warehouses via K-means clustering algorithm, International Journal of Business Intelligence and Data Mining 4(3/4), 2009, pp. 301– 328

[14] A. Cuzzocrea and D. Sacca,` Balancing accuracy and privacy of OLAP aggregations on data cubes, ACM DOLAP 2010, pp. 93–98

[15] A. Cuzzocrea, D. Sacca and P. Serafino, ` A Hierarchy-Driven Compression Technique for Advanced OLAP Visualization of Multidimensional Data Cubes, DaWaK 2006, pp. 106–119

[16] A. Cuzzocrea, D. Sacca and J.D. Ullman, ` Big Data: A Research Agenda, IDEAS 2013, pp. 198–203

[17] B. Da Mota, R. Tudoran, A. Costan, G. Varoquaux, G. Brasche, P.J. Conrod, H. Lemaˆıtre, T. Paus, M. Rietschel, V. Frouin, J.-B. Poline, G. Antoniu and B. Thirion, Generic Machine Learning Pattern for Neuroimaging-Genetic Studies in the Cloud, Frontiers in Neuroinformatics 2014(4), 2014

[18] Y. Deng, S. Shen, Z. Huang, A. Iosup, R.W. H. Lau, Dynamic Resource Management in Cloud-based Distributed Virtual Environments, ACM Multimedia 2014, pp. 1209–1212

[19] D. DiFranzo, A. Graves, A farm in every window: a study into the incentives for participation in the windowfarm virtual community, WebSci 2011, p. 14

[20] M. A. El-Refaey, M. A. Rizkaa, Virtual Systems Workload Characterization: An Overview, WETICE 2009, pp. 72–77

[21] D.Gmach, J.Rolia, L.Cherkasova and A.Kemper, Workload analysis and demand prediction of enterprise data center applications, IEEE WWC 2007

[22] G. Goel, R. Ganesan, S. Sarkar, K. Kaup, Workload Analysis for Virtual Machine Placement, IEEE ICPADS 2012, pp.732–737

[23] R.P. Goldberg, Survey of Virtual Machine Research, Computer 7(9), 1974, pp. 34–45

[24] Gutierrez-Garcia, J.O., Ramirez-Nafarrate, A.Policy-based Agents for Virtual Machine Migration in Cloud Data Centers, IEEE SCC 2013

[25] H. S. Hou, D. R. Tretter, M. J. Vogel, Interesting properties of the discrete cosine transform, Journal of Visual Communication and Image Representation 3(1), 1992, pp. 73–83
[26] S.-W. Hsiao, Y.-N. Chen, Y.S. Sun, M.C. Chen, Combining Dynamic Passive Analysis and Active Fingerprinting for Effective Bot Malware Detection in Virtualized Environments, NSS 2013, pp. 699–706
[27] Y. Hu, X. Long, C. Wen, Asymmetric Virtual Machine Scheduling Model Based on Workload Classification, CSSS 2012, pp. 2231–2234
[28] T. Joachims, Text Categorization with Support Vector Machines: Learning with Many Relevant Features, ECML 1998, pp. 137–142
[29] G. Kejela, R.M. Esteves and C. Rong, Predictive Analytics of Sensor Data Using Distributed Machine Learning Techniques, IEEE CloudCom 2014, pp. 626–631
[30] H.-C. Lee, J.-E. Park and M.-J. Lee, C3ware: A Middleware Supporting Collaborative Services over Cloud Storage, The Computer Journal 57(2), 2014, pp. 217–224
[31] C. Mant, Application of resampling and linear spline methods to spectral and dispersional analyses of long-memory processes, Computational Statistics & Data Analysis 51(9), 2007, pp. 4308–4323
[32] A. Moro, E. Mumolo, M. Nolich, Ergodic Continuous Hidden Markov Models for Workload Characterization, ISPA 2009
[33] A. Moro, E. Mumolo, M. Nolich, Workload modeling using pseudo2DHMM, MASCOTS 2009
[34] B.D. Ripley, Neural Networks and Related Methods for Classification, Journal of the Royal Statistical Society - Series B 56(3), 1994, pp. 409– 456
[35] L. Rokach, Ensemble-based Classifiers, Artificial Intelligence Review 33(1-2), 2010, pp. 1–39
[36] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, 1976
[37] T. Sherwood, E. Perelman, G. Hamerly, S. Sair, B. Calder,Discovering and Exploiting Program Phases, IEEE Micro Journal 23(6), 2003, pp. 84–93
[38] SOAP, HTTP://WWW.W3.ORG/TR/SOAP
[39] M.E. Sousa Vieira, A. Surez-Gonzlez, M. Fernndez-Veiga, J.C. LpezArdao, C. Lpez-Garca, Model selection for long-memory processes in the spectral domain, Computer Communications 36(13), 2013, pp. 1436– 1449
[40] Standard Performance Evaluation Corporation, HTTP://WWW.SPEC.ORG/CPU2006/CINT2006/
[41] T. Van Do, Comparison of Allocation Schemes for Virtual Machines in Energy-Aware Server Farms, Computing Journal 54(11), 2011, pp. 1790–1797
[42] N. Vandromme, T. Dandres, E. Maurice, R. Samson, S. Khazri, R.F. Moghaddam, K.K. Nguyen, Y. Lemieux, M. Cheriet, Life cycle assessment of videoconferencing with call management servers relying on virtualization, ICT4S 2014
[43] Virtualbox, HTTP://WWW.VIRTUALBOX.ORG/MANUAL
[44] Virtualbox API, HTTP://WWW.VIRTUALBOX.ORG/SDKREF/INDEX.HTML
[45] WSDL, HTTP://WWW.W3.ORG/TR/WSDL20
[46] L. Xiao, S. Chen, X. Zhang, Dynamic cluster resource allocation for jobs with known and unknown memory demand, IEEE Transactions on Parallel and Distributed Systems 13(3), 2002
[47] P. Xiao, Z. Hu, D. Liu, X. Zhang and X. Qu, Energy-efficiency enhanced virtual machine scheduling policy for mixed workloads in cloud environments, Computers & Electrical Engineering 40(5), 2014, pp. 1650–1665
[48] Y. Xu, Z. Musgrave, B. Noble and M. Bailey, Workload-Aware Provisioning in Public Clouds, IEEE Internet Computing 18(4), 2014, pp. 15–21
[49] J. Zhang and R. J. Figueiredo, Autonomic Feature Selection for Application Classification, IEEE CAC 2006, pp. 43–52
[50] X. Zhao, J. Yin, Z. Chen, S. He, Workload Classification Model for Specializing Virtual Machine Operating System, IEEE CLOUD 2013, pp. 343–350