****

****

دانشکده فنی و مهندسی

کارشناسی ارشد مهندسی کامپیوتر نرم‌افزار

گروه مهندسی کامپیوتر و فناوری اطلاعات

**گزارش درس سمینارعادی**

**موضوع:**

**کاهش مصرف انرژی (ابر سبز) در مراکز داده های ابری با کمک الگوریتم های فرا ابتکاری و تنظیم کاهش مهاجرت ماشین مجازی**

**نگارش:**

**سید علی محترمی**

**استاد راهنما:**

**دکتر سید علی رضوی ابراهیمی**

بهمن-اسفند 1399

**چکیده**

رشد صعودي دانش و ارتباطات سبب شده است سبک محاسباتی نوینی با عنوان رایانش ابري پدید آید. یکی از بزرگترین مزیتهاي مورد توجه تأمین کنندگان زیرساخت ابر، بحث مالی با به حداقل رسانیدن هزینه ها، تضمین قرارداد سطح سرویس و حداکثر سوددهی میباشد. در این راستا مدیریت انرژي در مراکز داده ابري، موضوعی مهم و قابل توجه براي رسیدن به هدف مذکور است. یکی از راه هاي کاهش مصرف انرژي، آزاد سازي میزبان هاي بیکار و راه دیگر کاهش تعداد مهاجرت ماشین هاي مجازي می باشد. بدین منظور، یکی از چالشهاي مطرح، انتخاب روش جایگذاري ماشین هاي مجازي مهاجرت داده شده، بر روي میزبان مناسب است. در این جزوه، با معرفی الگوریتم کاهش تعداد مهاجرت و الگوریتم جایگذاري بر اساس حد آستانه مناسب، به کاهش مصرف انرژي در مراکز داده ابري پرداخته شده است. این الگوریتم با ارث بري از الگوریتم اولین انتخاب و مرتب کردن نزولی میزبان ها و ماشین هاي مجازي براساس رتبه آنها، که این رتبه براساس بار کاري بدست می آید، عمل جایگذاري را انجام میدهد. راهکار ارائه شده در نرم افزار کلودسیم شبیه سازي شده است. نتایج حاصل از شبیه سازي حاکی از کاهش تعداد مهاجرت ماشین مجازي، افزایش بهره وري مهاجرت وکاهش مصرف انرژي می باشد.

**کلمات کلیدی:** محاسبات ابري سبز، کاهش انرژي مصرفی مراکز داده، تجمیع میزبان ها، کاهش تعداد مهاجرت

[عنوان](#_Toc413125201) صفحه

صفحه

[فصـل اول 1](#_Toc64848111)

[مقـدمـه 1](#_Toc64848112)

[1-1 مقدمه 2](#_Toc64848113)

[1-2 روش تجزیه و تحلیل اطلاعات 4](#_Toc64848114)

[فصـل دوم 6](#_Toc64848115)

[ادبیات تحقیق 6](#_Toc64848116)

[2-1مقدمه 7](#_Toc64848117)

[2-2رایانش ابری 9](#_Toc64848118)

[2-3عناصر زیرساخت رایانش ابري 11](#_Toc64848119)

[2-4 مدل های استقرار ابر 12](#_Toc64848120)

[2-5 معماری ابر 13](#_Toc64848121)

[6-2 مزایاي رایانش ابري از دید زیر ساخت 14](#_Toc64848122)

[2-7مراکز داده ابري 15](#_Toc64848123)

[2-8 مدل هاي کاهش مصرف انرژي 16](#_Toc64848124)

[ 2-8-1 مدیریت استاتیک و مدیریت پویاي توان 18](#_Toc64848125)

[ 2-8-2مدل مصرف انرژي 18](#_Toc64848126)

[ ۲-۸-۳ سطح سیستم عامل 20](#_Toc64848127)

[ ۲-۸-۴ سطح مجازي سازي 21](#_Toc64848128)

[ ۲-۸-۵ سطح مراکز داده 22](#_Toc64848129)

[۲-۹ نتیجه گیري 24](#_Toc64848130)

[فصـل سوم 25](#_Toc64848131)

[پیشینه تحقیق 25](#_Toc64848132)

[1-3 مقدمه 26](#_Toc64848133)

[ 3-1-1 کاهش مصرف انرژي در گره ها در محیط ناهمگن 26](#_Toc64848134)

[ 3-1-2 تجمیع انرژي آگاه براي محاسبات ابري 26](#_Toc64848135)

[ 3-1-3 تخصیص بهینه گره 27](#_Toc64848136)

[ 3-1-4 مدیریت برق 27](#_Toc64848137)

[ 3-1-5 قدرت و عملکرد مدیریت 28](#_Toc64848138)

[ 3-1-6 تخصیص منبع با استفاده از خوشه هاي مجازي 29](#_Toc64848139)

[ 3-1-7چارچوب آگاه از توان و انرژي 29](#_Toc64848140)

[**3-1-7-1اولین انتخاب به صورت کاهشی** 30](#_Toc64848141)

[ 3-1-8ابر سبز : انرژي کارآمد و مدیریت منابع 30](#_Toc64848142)

[**3-1-8-1 سیاست تخصیص** 31](#_Toc64848143)

[**3-1-8-2بهترین انتخاب به صورت کاهشی** 31](#_Toc64848144)

[**3-1-8-3سیاست انتخاب ماشین هاي مجازي** 32](#_Toc64848145)

[3-2 نتیجه گیري 35](#_Toc64848146)

[فصـل چهارم 36](#_Toc64848147)

[روش کار 36](#_Toc64848148)

[4-1مقدمه 37](#_Toc64848149)

[4-2روش کار 37](#_Toc64848150)

[ 1-2-4مدل تجمیع گره هاي فیزیکی 38](#_Toc64848151)

[ 4-2-2 مدل مهاجرت ماشین مجازي 40](#_Toc64848152)

[ 4-3-2مدل تخصیص ماشین مجازي 40](#_Toc64848153)

[4-3 نتیجه گیري 41](#_Toc64848154)

[فصـل پنجم 42](#_Toc64848155)

[کدهاشبیه سازی ها 42](#_Toc64848156)

[5-1مقدمه 43](#_Toc64848157)

[5-2نتیجه گیری: 59](#_Toc64848158)

[منابع : 60](#_Toc64848159)

**فهرست اشکال**

[عنوان](#_Toc413125201) صفحه

|  |  |
| --- | --- |
| تصویر 2-1 : بررسی گوگل ازمقبولیت سیستم های کلاستری،توری وابری تا سال 2013  تصویر 2-2 : بکپارچه سازی کلیه خدمات ارائه شده ابر از دیدکاربر  تصویر 2-۳ : مدل های استقرار ابر  تصویر 2-4 : معماری لابه های ابر  تصویر 2-5 : ابر وتاثیرآن برمحیطزیست  تصویر 2-6 : مصرف انرژی دریک گره | 7  8  13  14  16  17 |
| تصویر 2-7 : تکنیک های مدیریت قدرت | 19 |
| تصویر 2-8 : مدیریت توان درسطح سیستم عامل | 21 |
| تصویر 2-9 : مدیریت توان در سطح مراکز داده ها | 23 |
| تصویر 3-1 : تخصیص ماشین های میزبان به ماشین های مجازی با استفاده از روشFFD | 30 |
| تصویر 3-2 : تخصیص ماشین های میزبان به ماشین های مجازی با روش BFD | 32 |
| تصویر 3-3 : ساختار سیستم انرژی کارامد برای محاسبات ابری سبز | 33 |

**فهرست جداول**

[عنوان](#_Toc413125201) صفحه

|  |  |
| --- | --- |
| جدول 3-۱ : مقایسه تحقیقات پیشین در سطح مراکز داده ها | 35 |

**فهرست علائم اختصاری**

|  |
| --- |
| NIST: National Institute of Standards and Technology |
| IaaS: Infrastructure as a Service |
| PaaS: Platform as a Service |
| SaaS: Software as a Service |
| SLA: Service Level Agreement |
| CDC: Cloud Data Centers |
| ICT: Information and Communication Technologies |
| EPA: US Environmental Protection Agency |
| DPM: Dynamic Power Management |
| SPM: Static Power Management |
| DVFS: Dynamic Voltage and Frequency Scaling |
| DCD: Dynamic Component Deactivation |
| DPS: Dynamic Performance Scaling |
| VMM: Virtual Machine Monitor | |
| DFS: Dynamic Frequency Scaling | |
| LLC: Limited Lookahead Control | |
| FFD: First-Fit Decreasing | |
| PABFD: Power Aware Best Fit Decreasing | |
| BFD: Best Fit Decreasing | |
| MMT: Minimization of Migrations Time | |
| HPG: Highest Potential Growth | |
| NAS: network attached storage | |

فصـل اول

مقـدمـه

1-1 مقدمه

در دنیاي امروز غیرممکن است بتوان تصور کرد فردي بدون رایانه زندگی را سپري کند. درواقع همه افراد و در همه سنین روزانه از رایانه استفاده میکنند و رایانه ها در همه مشاغل و مراودات نقش کلیدي بازي میکنند. رشد صعودي دانش و ارتباطات و از همه مهمتر شبکه گسترده اینترنت باعث شده است کارشناسان فناوري اطلاعات پس از سالها کار با شبکه هاي اینترنت و ایجاد سبکهاي محاسباتی متفاوت، تصمیم به ساخت سبک محاسباتی نوینی به عنوان پردازش ابري [[1]](#footnote-1)، رایانش ابري و یا محاسبات ابري بگیرند (1)

مفاهیم ابتدایی این فناوري در سال 1960 توسط پروفسور جان مک کارتی بیان شد و در سال 2006 براي اولین بار توسط آمازون ارائه گردید و بدین ترتیب در سالهاي بعد شرکتهاي دیگر به ارائه خدمات خود بر مبناي این فناوري روي آوردند(2)

تعاریف بسیاري از این روش محاسباتی شده است. تعریف موسسه ملی استاندارد و فناوري اطلاعات [[2]](#footnote-2) که رایان ابري را مدلی براي فراهم کردن دسترسی آسان بر اساس تقاضاي کاربر از طریق شبکه به مجموعه اي از منابع رایانشی قابل تغییر و پیکربندي (مانند شبکه، گره ها ، فضاي ذخیره سازي،برنامه هاي کاربردي و...)که این دسترسی بتواند با کمترین نیاز به مدیریت منابع و یا نیاز به دخالت مستقیم فراهمکننده سرویس به سرعت فراهمشده، معرفی کرده است(3). فلسفه استفاده از ابر بدین دلیل است که اینترنت مانند ابر در همه جا وجود دارد و نماد انتزاعی براي نمایش نقاط مرزي بین فراهم کنندگان ابرمی باشد. استفاده از رایانش ابري سبب کاهش هزینه ها، ظرفیت ذخیره سازي نامحدود، سهولت در استفاده، مقیاس پذیري[[3]](#footnote-3) ، قابلیت حمل آسان وغیره شده است(1).

با ظهور ابر و رشد درخواست از ساختار آن باعث افزایش چشمگیر مصرف انرژي در مراکز داده شده است. این مصرف بالاي انرژي سبب افزایش هزینه ها و کاهش میزان سود براي فراهم کننده ابر شده است. با افزایش دما، کاهش عمر تجهیزات، کاهش قابلیت اطمینان و افزایش دي اکسید کربن، نقض کیفیت سرویس و بدنبال آن نقض توافق نامه سطح سرویس بوجود خواهد آمد و دلایل فوق از نتایج افزایش مصرف انرژي در مراکز داده بوده است (1و8)بنابراین در جهت تحقق رایانش ابري سبز[[4]](#footnote-4) باید با کاهش انرژي در مراکز داده تأثیر مخرب عوامل فوق را در محاسبات ابري کاهش داد. یکی از بهترین روش هاي به کارگرفته شده براي کاهش مصرف انرژي، تکنیک تجمیع ماشین هاي مجازي و گره ها است (4و8)در این روش که در محیط هاي ابري و مجازي شده به کار می رود، بار کاري (ماشین مجازي) چندین گره فیزیکی بر روي یک گره قرار گرفته و گره کم بار خاموش یا هایبرنت می شود.

بمنظور تجمیع دو چالش وجود دارد: 1 -انتخاب بهترین ماشین فیزیکی براي قرار دادن ماشین مجازي بر روي آن بگونه اي که حداکثر منابع ماشین فیزیکی و کمترین هدر رفتگی منابع را داشته باشیم. 2 -اگر عمل نگاشت به درستی انجام نشود منجر به افزایش تعداد مهاجرت ماشین مجازي خواهد شد که افزایش مصرف انرژي را بدنبال دارد چرا که در عمل مهاجرت، پردازنده و پهناي باند درگیر انتقال صفحات حافظه از گره مبدأ به گره مقصد خواهند شد (4)

بنابراین انتخاب بهترین ماشین فیزیکی براي تخصیص منابع موجود و همچنین کاهش تعداد مهاجرت ماشین مجازي به منظور کاهش مصرف انرژي امري ضروري و هدف این تحقیق می باشد. هدف از تحقیق حاضر کاهش مصرف انرژي با رویکرد حفظ کیفیت سرویس دهی در محیط هاي رایانش ابري در لایه زیر ساخت به عنوان سرویس با کاهش تعداد مهاجرت می باشد. این تحقیق سعی دارد با استفاده ازبهینه سازي برنامه ریزي تخصیص ماشین هاي مجازي به ماشین هاي میزبان، میزبان هاي اضافی را خاموش و یا هایبرنت کند تا در مصرف انرژي صرفه جویی گردد و همین طور با کاهش تعداد مهاجرت ماشین مجازي نه تنها در مصرف انرژي کاهش صورت می گیرد بلکه در کیفیت سرویس دهی و جلوگیري از نقض قرارداد سطح سرویس نیز تأثیر بسزایی خواهد داشت.

1-2 روش تجزیه و تحلیل اطلاعات

براي آزمایش اهداف کاربردي در محیط ابر مانند کاهش تعداد مهاجرت ماشین مجازي و کاهش انرژي مصرفی در مراکز داده ابر، به دلیل صرف هزینه و زمان زیاد، نمی توان از محیط ابر واقعی استفاده نمود. بنابراین بهترین راه حل، استفاده از ابزار شبیه سازي است که امکان تست را در محیط کاملا کنترل شده و تکرار پذیر و در زمان اندك و هزینه کم امکان پذیر می سازد. یکی از شبیه سازهاي رایج در این زمینه، نرم افزار کلودسیم[[5]](#footnote-5) می باشد که تحقیقات تیم توسعه در زمینه ابر تأثیرهاي مهمی بر قابلیتهاي این شبیه ساز داشته است (5,4,3,2)از جمله امکانات کلودسیم می توان به پشتیبانی از انواع مدل سازي سیاست تخصیص منابع و شبیه سازي زیر ساخت ابر و مدیریت سرویس هاي مجازي اشاره نمود.

برتري این شبیه ساز نسبت به شبیه سازهاي دیگر مانند گریدسیم و سیم گرید تفکیک لایه هاي مورد نیاز که در س رایانش ابري استفاده می شود (مانند لایه هاي زیر ساخت به عنوان سرویس ، بستر به عنوان سرویس و نرم افزار به عنوان سرویس). علاوه بر این در شبیه سازهاي مذکور پشتیبانی چندانی از منابع مجازي و مدل سازي مراکز داده با گره هاي محاسباتی فراوان صورت نگرفته است. با توجه به موارد فوق، ابزار استفاده شده در این تحقیق نرم افزار کلودسیم است که چارچوب شبیه ساز عمومی را بمنظور مدلسازي، شبیه سازي و ارزیابی راهکار پیشنهادي در زیر ساخت ابر فراهم می کند

ساختار تحقیق سایر بخش هاي تحقیق حاضر بدین شکل تنظیم شده است: ابتدا در فصل دوم به تعریف اصول و مفاهیم پایه ي موضوع می پردازد. سپس در فصل سوم به راهکارهاي موجود جهت کاهش مصرف انرژي پرداخته می شود و آنها از جنبه هاي مختلف با هم مقایسه می شوند. فصل پنجم به راه اندازي ابزار کلودسیم می پردازد و درفصل آخر به کدهاي به کارگرفته شده در شبیه ساز پرداخته می شود.

فصـل دوم

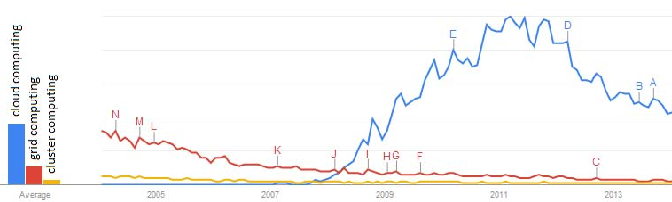
ادبیات تحقیق

در این فصل تعاریف و مفاهیم پیش زمینه اي را که براي شناخت دقیق مسئله در تحقیق ضروري است، توضیح میدهیم

2-1مقدمه

سیرتکاملی محاسبات به گونه اي است که میتوان آن را پس از آب، برق، گاز و تلفن به عنوان عنصراساسی پنجم فرض نمود. درچنین حالتی، کاربران سعی میکنند، بر اساس نیازهایشان و بدون توجه به اینکه یک سرویس در کجا قرار دارد و یا چگونه تحویل داده میشود، به آن دسترسی یابند. نمونه هاي متنوعی از سیستمهاي محاسباتی ارائه شده است، که سعی دارند چنین خدماتی را به کاربرانارئه دهند(3)

برخی ازاین سیستم هاي محاسباتی عبارتنداز: محاسبات خوشه اي[[6]](#footnote-6) ، محاسبات توري[[7]](#footnote-7) واخیرا ًمحاسبات ابري[[8]](#footnote-8) که از آن به عنوان رایانش ابري نیز یاد میشود. محبوبیت این سه رویکرد محاسباتی، از دید موتور جستجوي گوگل مورد ارزیابی قرارگرفته است، که نتیجه ي آن در شکل2-1 نمایش داده شده است، حاکی از آن است که محبوبیت رایانش ابري، پس از ظهور مفاهیم اولیه آن در سال 2007 ،بافاصله زیادي نسبت به سایر رویکردهاي محاسباتی در حال افزایش است.(54)

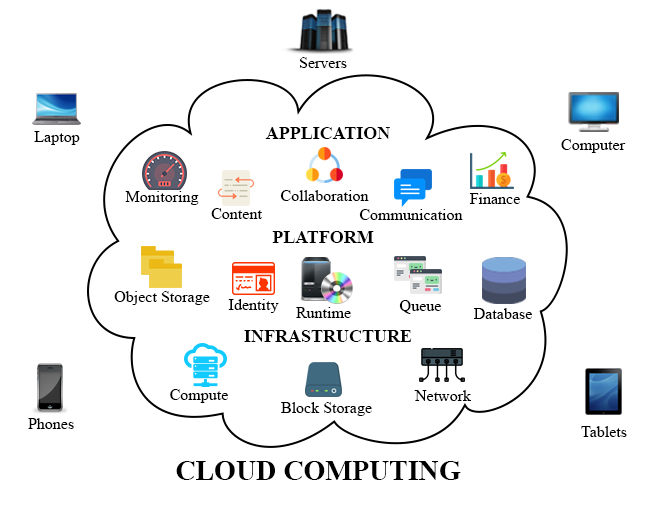


شکل 2-1: بررسی گوگل از مقبولیت سیستمهاي کلاستري، توري و ابري تا سال2013

دنیاي محاسبات به سرعت به سمت توسعه نرم افزارهایی پیش میرود که به جاي اجرا بر روي رایانه هاي منفرد، به عنوان یک سرویس در دسترس میلیونها مصرف کننده قرار داده میشوند. از این نقطه نظر، رایانش ابري از دید کاربران نهایی ساختاري شبیه به یک توده ابر دارد که به واسطه آن میتوانند به برنامه هاي کاربردي از هرجایی از دنیا دسترسی داشته باشند.(55)

اما رایانش ابري از دید فراهم کنندگان منابع زیرساخت، میتواند با کمک ماشین هاي مجازي شبکه شده، به عنوان یک روش جدید براي ایجاد پویاي نسل جدید مراکز داده، مورداستفاده قرار گیرد تا بتوانند یک زیرساخت قابل انعطاف براي ارائه انواع مختلف خدمات محاسباتی و ذخیره سازي در اختیار داشته باشند(10)

دررایانش ابري از دید ارائه سرویس و برنامه هاي کاربردي تلاش بر این است، که خدمات اینترنتی به صورت یک رایانه واحد در اختیار تمام کاربرانی که به آن متصل هستند، قرار بگیرد و فنّاوري هایی نظیر وب دو از عوامل مهم در نیل به این هدف هستند. در این رویکرد جدید، میتوان از لایه هاي مختلف و قابل انعطاف ارائه شده در ابر استفاده کرد.(10) در حقیقت حرکت در ابر به سمتی پیش میرود، که دیگر براي کاربر فرقی نمیکند، در حال استفاده از کدام سایت است. کاربر کل اینترنت را همانند یک رایانه شخصی در خدمت خود مشاهده میکند. شکل زیر یکپارچه سازي خدمات ابر از دید کاربر است(5).



شکل 2-2 :یکپارچه سازي کلیه خدمات ارائه شده ابر از دید کاربر

کاملاً واضح است که در چنین محیطی کار با سایت هاي مختلف، همانند کار با برنامه هاي کاربردي مختلف در یک رایانه شخصی میباشد و گرایش به سمتی است که دیگر کاربر مرزي را بین این خدمات احساس نکند و بتواند با همان درجه آزادي که در رایانه شخصی خود دارد، در اینترنت نیز به فعالیت بپردازد. اما اگر بخواهیم به چنین سطحی از انعطاف پذیري در ارائه خدمات کاربردي به کاربران دست پیدا کنیم، نیاز به زیرساختی قابل انعطاف خواهیم داشت که توسط فراهم کنندگان منابع زیرساخت میتوان به آن دست پیدا کرد(55).

2-2رایانش ابری

براي شناخت بهتر رایانش ابري از دید زیرساخت، ابتدا نگاهی به سیرتکاملی سیستم هاي محاسباتی از ابتدا تا کنون می اندازیم، تا بتوانیم جایگاه آن را در بین دیگر سیستم ها تشخیص دهیم. اگر به نسل اول سیستم هاي محاسباتی نگاه کنیم، ما در ابتدا با یک سیستم بسیار بزرگ مواجه بودیم، که کاربران از طریق یک ترمینال واحد به آن دسترسی داشته اند. کم کم این سیستم ها کوچک تر شدند و با توان پردازشی بیشتر و قیمت کمتر، به صورت رایانه هاي شخصی در اختیار همه کاربران قرار گرفتند. سپس این امکان فراهم شد که با اتصال مجموعه اي از این سیستم هاي کوچک، شبکه اي با توان پردازشی بیشتر فراهم شود، تا پاسخگوي نیاز هاي کاربران شود. اما نیاز هاي پردازشی در حال افزایش بود و نیاز به سیستم هاي محاسباتی قویتر، بنابراین تعداد بسیاري از این شبکه ها بصورت اختصاصی در سرتاسر اینترنت به یکدیگر متصل شدند و شبکه هاي محاسبات توري را بوجود آوردند.(54) در این بین مشاهده شد، که میلیون ها کاربر در اینترنت وجود دارند که در اکثر اوقات از تمام توان رایانه خود استفاده نمی کنند و سیستم محاسباتی دیگري شکل گرفت، تا کاربرانی که تمایل داشته باشند، زمان هاي بیکار سیستم خود را براي کارهاي محاسباتی عام المنفعه هدیه کنند. بنابراین تعداد بسیارزیادي منبع محاسباتی کوچک در شبکه اي تحت عنوان محاسبات داوطلبانه به هم پیوستند و توان پردازشی عظیمی را بوجود آوردند.( 54)

اما هنوز منابع بسیار زیاد دیگري در سازمان ها و مراکز داده اینترنتی وجود داشت که تمام ظرفیت آنها بطورکامل بکارگرفته نشده بود. این منابع نمی توانستند در شبکه محاسبات توري بصورت اختصاصی بکارگرفته شوند، زیرا براي آنها وظیفه دیگري تعریف شده بود. در عین حال امکان استفاده از آنها در شبکه داوطلبانه هم وجود نداشت، چون فلسفه وجودي آنها، کاربردهاي تجاري بود. به این ترتیب رویکرد جدیدي شکل گرفت که بتوان با استفاده از فناوري هاي مجازي سازي این منابع را بصورت قابل انعطاف و پویا براي کاربردهاي مختلف مورد استفاده قرار داد و از تمام ظرفیت آن ها بطور موثر استفاده کرد. این فناوري رایانش ابري در لایه زیرساخت نام داشت که امکان استفاده از منابع محاسبات و ذخیره سازي را بصورت یک سرویس بر حسب نوع نیاز فراهم می آورد. در حقیقت با ایجاد یک لایه انتزاعی بر روي کلیه منابع فیزیکی خود( به کمک مجازي سازي) امکان مدیریت پویاي منابع فیزیکی حاصل می شود(55).

بنابراین رایانش ابري از دید زیرساخت، به گونه اي به سیستم هاي توزیع شده و موازي اطلاق می گردد که مجموعه اي از رایانه هاي مجازي را که به یکدیگر متصل هستند، شامل می شود. این رایانه ها بطور پویا عرضه شده و به عنوان یک یا چند منبع محاسباتی یکپارچه براساس توافقات سطح سرویس ارائه می شوند.(54)

این توافقات در طول انجام مذاکرات سرویس دهندگان و مصرف کنندگان برقرار میگردند. رایانش ابري سعی دارد نسل جدیدي از مراکز داده اي را، با ارائه کردن سرویس ها و خدمات در ماشین هاي مجازي شبکه شده به صورت پویا، به گونه اي ممکن سازد که ارائه دهندگان خدمات کاربردي بتوانند، سرویس ها و برنامه هاي کاربردي را با انعطاف پذیري و سهولت بیشتري ارائه کنند و کاربران نیز بتوانند از هر جایی از دنیا به برنامه هاي کاربردي دسترسی داشته باشند(1و2و3)

در تکنولوژي رایانش ابري، کاربران می توانند از طریق ابزارهاي مختلف( نظیر رایانه هاي شخصی، رایانه هاي همراه و تلفن همراه و...) به برنامه ها، فضاهاي ذخیره سازي، پردازش، از طریق سرویس هاي ارائه شده توسط رایانش ابري، دسترسی داشته باشند. به این ترتیب منابع به جاي قرار گیري در سمت کاربر، در سمت سرورها قرار می گیرند و خدمات فراهم شده از طریق رایانش ابري همانند خدمات عمومی (آب، برق و تلفن و ...) بر اساس سطح استفاده افراد پرداخت می شود(53)

2-3عناصر زیرساخت رایانش ابري

مجازي سازي[[9]](#footnote-9) این فناوري که مربوط به دهه 70 هست جهت جلوگیري از افزایش تعداد سرویس دهنده ها توسط کارشناسان به منظور کاهش هزینه ها به وجود آمد و در دستگاههاي کامپیوتري به معنی پیاده سازي یک طرح واقعی با شبیه سازي تعریف میشود و به طورکلی امکان تخصیص منابع سختافزاري براي چندین سیستم عامل (مختلف) را امکانپذیر میکند(6و8)

· ماشین مجازي[[10]](#footnote-10) ماشین مجازي به دستگاه ی گفته میشود که اجراي سیستم مجازي را بر عهده دارد و هر ماشین مجازي خود یک کامپیوتر است با همان مشخصات و با یک سیستم عامل میهمان ولی جدا از کامپیوتر والد هست و به همین دلیل عملکرد یک ماشین مجازي بر فعالیت دیگر ماشینها تأثیري نخواهد داشت(8,6)

فوق ناظر[[11]](#footnote-11) یک برنامه سطح پایین است که براي فراهم کردن دسترسی منابع سیستم به ماشینهاي مجازي، مورداستفاده قرار میگیرد و باعث میشود که ماشینهاي مجازي از دید یکدیگر پنهان بمانند، به طوري که هر ماشین مجازي تصور میکند، تمام منابع را به تنهایی در اختیار خود گرفته است(53)

پردازش شبکه اي[[12]](#footnote-12) این فناوري با استفاده از زیرساختهاي ارتباطی و شبکه هاي کامپیوتري، امکان دسترسی به انواع منابع از راه دور را میدهد. این شبکه داراي چندین ریزپردازنده، هزاران کامپیوتر شخصی و ایستگاه کاري در سراسر دنیا هست و بااتصال این منابع امکان اجراي پردازشهاي بسیار عظیم را امکان میسازد، اما با پردازش ابري تفاوتهاي عمدهاي دارد.(32) در پردازش شبکه اي تخصیص منابع محدود و از پیش بر اساس قراردادي بین سازمان و مدیر دامنه تعیین شده است. بنابراین منابع محدودي را در اختیار دارد. اما در پردازش ابري تخصیص منابع بر اساس تأمین در صورت درخواست هست یعنی براي کاربر پرداخت درازاي استفاده صورت میپذیرد.(32) در پردازش شبکه اي تمامی دستگاههاي شبکه باید از سیستم عامل ها و نرم افزارهاي یکسان و مشابهی استفاده کنند، اما در پردازش ابري محدودیتی در این زمینه وجود ندارد(32)

پردازش شبکه اي براي تعداد اندکی از کاربران که پردازش هاي بزرگی دارند ولی پردازش ابري براي هر تعداد کاربر با هر نوع درخواستی هست. اما وجه تشابه هردو استفاده از فنّاوري مجازي سازي هست.

وب دو

وب دو پدیدهاي است که در نحوه استفاده از فناوري و طراحی سایت ها در اینترنت مطرح شده است. سایت ها یا خدمات اینترنتی که امکان تبادل اطلاعات را بین کاربران فراهم میکنند، یا به آنها اجازه تولید یا دستکاري در اطلاعات را میدهند معمول وب دو تلقی میشوند(54).

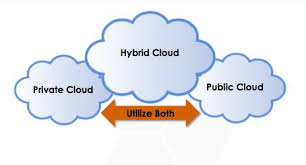
2-4 مدل های استقرار ابر

ابر عمومی[[13]](#footnote-13) در این مدل فراهم کننده ابر منابع را به صورت اشتراکی به کاربران اجاره میدهد و مشابه صنعت برق و تلفن براي کاربران صورتحساب میفرستد. مانند خدمات مایکروسافت و گوگل(54و47)

ابر خصوصی ابر خصوصی براي استفاده انحصاري یک سازمان به کار میرود. مزیت ابرهاي خصوصی امنیت بیشتر هست مانند ابر خصوصی آمازون.

ابر ترکیبی[[14]](#footnote-14) این ابر همانطور که از اسمش پیداست، تشکیلشده از یک یا چند ابر عمومی و خصوصی، که هرکدام با حفظ هویتش با ترکیب شدن به عنوان یک واحد عمل میکند.

شکل زیر نماي کلی از مدل هاي ابررا نشان می دهد:



شکل 2-3 :مدل هاي استقرار ابر

2-5 معماری ابر

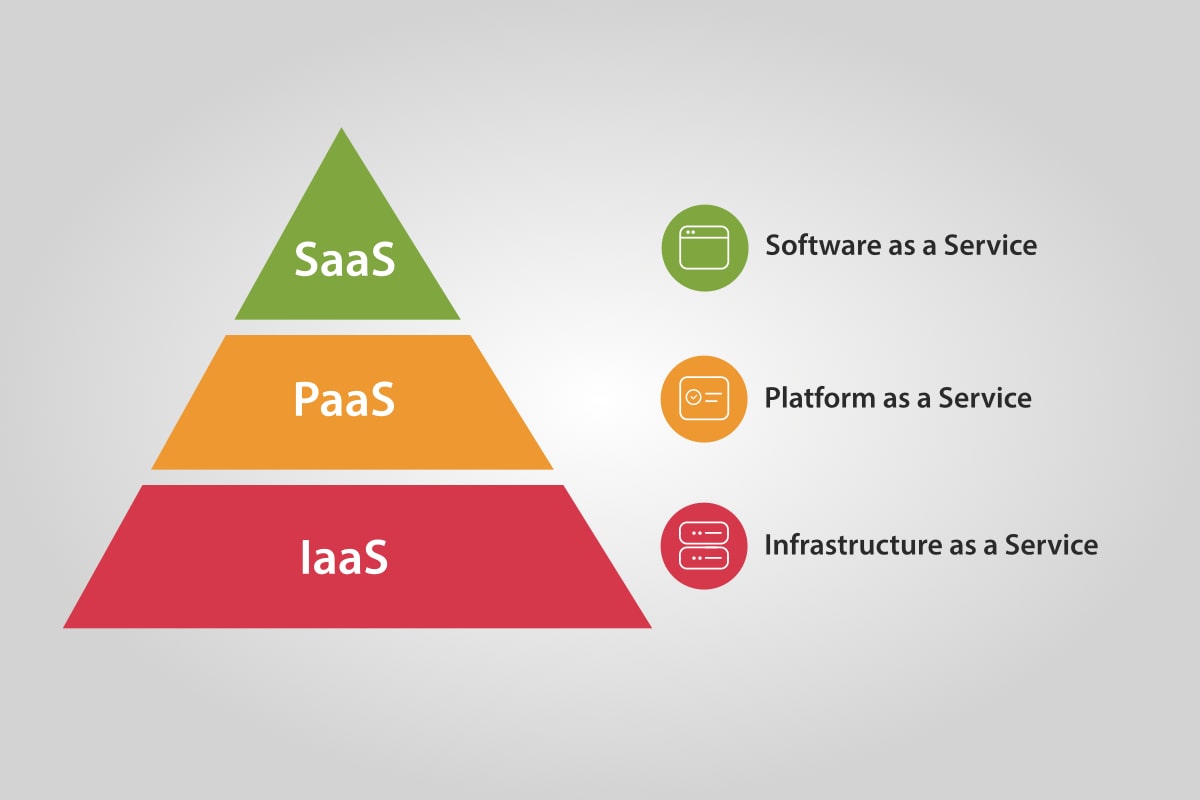
زیر ساخت به عنوان سرویس[[15]](#footnote-15)

ارائه یک بستر مجازي شده در قالب سرویس توسط فراهم کننده ابر هست که این بستر میتواند فضاي ذخیره سازي گره ها، اجزاي شبکه و ماشین مجازي باشد. برخی سازمانها که این خدمات را ارائه میدهند مانند آمازون و گوگل(8)

سکو به عنوان سرویس[[16]](#footnote-16) این لایه در بالاي لایه زیرساخت قرار دارد. در این سرویس امکان ساخت لایه هاي بالاتر به وجود میآید. این لایه شامل میان افزار، امکانات تبادل پیام و تنظیم اتصال هست. درواقع محیطی را براي توسعه دهنده ي وب براي ایجاد یک برنامه کاربردي ایجاد میکند. مانند گوگل اپ (8)

نرم افزار به عنوان سرویس[[17]](#footnote-17) در این مدل نرم افزار و سخت افزار محاسباتی به عنوان سرویس در اختیار مشتري قرار میگیرد. یعنی میتوان به طورکلی اینطور تعریف کرد که تمامی نرم افزارهایی که تاکنون از مغازه ها تهیه میکردیم اکنون آنها را در ابر و با استفاده از اینترنت و با یک واسط، که در سراسر دنیا به آنها دسترسی خواهیم داشت. مانند سرویس جیمیل از شرک گوگل(8و60)

در شکل زیر معماري چندلایه اي ابر نشان داده شده است



شکل 2-4 : معماري لایه هاي ابر

6-2 مزایاي رایانش ابري از دید زیر ساخت

به منظور استفاده حداکثر از رایانش ابري، توسعه دهندگان باید بتوانند ساختار برنامه هاي خود را بگونه اي اصلاح کنند که به بهترین شکل توسط رویکردمعماري رایانش ابري پشتیبانی شود. مزایاي استقراربرنامه ها با استفاده از رایانش ابري شامل کاهش زمان اجرا و زمان پاسخ، حداقل کردن ریسک آماده سازي و ارائه زیرساخت فیزیکی، کاهش هزینه ورود و افزایش امکان نوآوري می باشد(18و1)

شاید بزرگترین مزیت، مورد توجه تأمین کنندگان زیرساخت ابر، بحث مالی باشد که در برگیرنده هزینه هاي سرمایه گذاري است. براي مثال اگر شما یک سرویس دهنده خریداري کنید، یک هزینه سرمایه گذاري انجام داده اید، زیرا در ابتدا هزینه را پرداخت کرده اید و سپس ممکن است پس از 2 تا 3 سال نتیجه سرمایه گذاري خود را بدست آورید(2)

از دیدگاه یک ارائه دهنده ابر مسأله مهم این است که با به حداقل رساندن هزینه هاي عملیاتی و تضمین قرارداد سطح سرویس، به حداکثر سود دهی برسند که در همین راستا مدیریت انرژي در مراکز داده ابري تبدیل به یک موضوع بسیار مهم و قابل توجه براي رسیدن به این هدف می باشد(2).

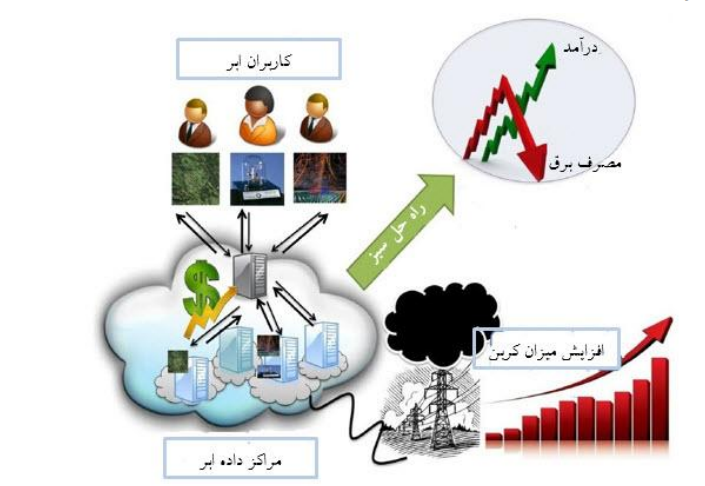
2-7مراکز داده ابري[[18]](#footnote-18)

مراکز داده ابر با نوع سنتی خود کاملا متفاوت است. این مراکز تشکیل شده از صدها یا هزاران شبکه کامپیوتري که به منظور ذخیره سازي و توزیع قدرت و داراي تجهیزات شبکه، تهویه و زیر ساخت خنک کننده ها می باشد. ابرها نیاز به مصرف انرژي بالایی دارند، براي مثال یک مرکز داده معمولی با 1000قفسه نیاز به 10 مگاوات قدرت دارد که از هزینه هاي عملیاتی آنهم بیشتر خواهد شد. از سال 1990 تا به امروز مصرف برق دوبرابر شده است و پیش بینی می شود تا سال 2040 تقریبا افزایش 2,2 %را در هر سال بدنبال داشته باشد (1) این افزایش دما کاهش عمر تجهیزات، تولید گازهاي گلخانه اي، کاهش قابلیت اطمینان، نقض کیفیت سرویس و بدنبال آن نقض توافق نامه سطح سرویس بدنبال خواهد داشت(1) از دیگر مسائل جدي که تأمین کننده ابر باید به آن توجه نماید این است که، با مصرف بسیار انرژي الکتریکی در مراکز داده ابري با این مقیاس بزرگ، دي اکسید کربن تولید می شود که اثر مخرب بر اتمسفر (تأثیر گازهاي گلخانه اي) و همچنین اثر مخرب بر محیط زیست دارد(1)

در ماه آوریل 2007 گارتنر تخمین زده است که صنعت فناوري اطلاعات و ارتباطات حدود 2 %از کل گازهاي گلخانه اي جهان را تولید می کند که با صنعت حمل و نقل هوایی برابر است و طبق گزارش منتشر شده توسط اتحادیه اروپا کاهش در حجم انتشار گاز دي اکسید کربنتا 15-%30 %قبل از سال 2020 براي حفظ افزایش دماي جهانی لازم و ضروري می باشد(11,1).

براساس گزارش آژانس حفاظت ازمحیط زیست آمریکا سال 2007 در مراکز داده ابري آمریکا در حدود 5,1درصد 1 از کل انرژي، که هزینه اي حدود 5,4میلیارددلار است، مصرف می شود که این مصرف بالاي انرژي منجر به انتشار کربن بسیار بالا که درحدود80-116 مگاتن در هر سال برآورد شده، می شود(1) با توجه به محبوبیت روبه رشد استفاده از محاسبات ابري و افزایش آگاهی مردم در سراسر جهان به سمت استفاده از منابع سازگار با محیط زیست محققان را وادار کرده است به تدبیر در مورد مفاهیم ابر سازگار با محیط زیست به نام محاسبات ابري سبز[[19]](#footnote-19) با کاهش مصرف انرژي و کاهش انتشار گاز دي اکسید کربن(4,3,2)

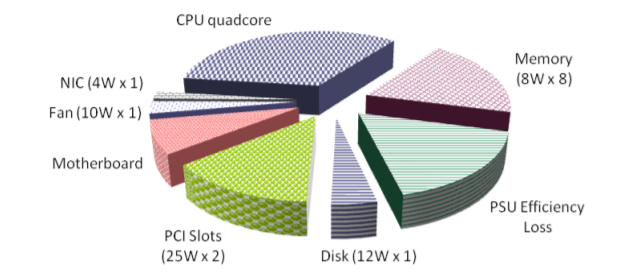
شکل زیرتأثیر ابر بر محیط زیست را نشان داده است:



شکل 2-5 :ابرو تأثیر آن بر محیط زیست

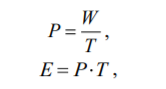
2-8 مدل هاي کاهش مصرف انرژي

در شکل زیر میزان مصرف انرژي در یک گره نشان داده شده است، همان طور که نمایان است میزان مصرف انرژي در مراکز داده ابري ارتباط مستقیم یا خطی با میزان مصرف پردازنده دارد و این دلیلی می شود تا بیشترین تمرکز بر این قطعه باشد.



شکل 2-6 :مصرف انرژي در یک گره

براي درك هرچه بهتر مدیریت انرژي لازم است ابتدا دو اصطلاح توان(برق) و انرژي را تعریف کنیم: توان و انرژي با تعریف کارانجام شده توسط سیستم معرفی می شوند. توان نرخ انجام کار در یک دوره زمانی وانرژي کل کارانجام گرفته که بیش از یک دوره زمانی است. توان و انرژي در(2-1)تعریف شده اند:



تفاوت این دو مهم است، به عبارتی کاهش مصرف توان از طریق کاهش هزینه هاي زیرساخت ابر، هزینه هاي ژنراتوربرق و سیستم خنک کننده صورت می گیرد، درحالی که کاهش انرژي کلی و در برگیرنده کاهش قبوض برق می باشد

مصرف انرژي می تواند به طور موقت از طریق مدیریت پویاي توان[[20]](#footnote-20) کاهش بیابد با تکنیک DPM ،یا به طور دائم تکنیک SPM ،که این تکنیک با استفاده از قطعات سخت افزاري بسیارکارآمد با استفاده از مدیریت استاتیک توان[[21]](#footnote-21) مانند پردازنده، ذخیره سازي دیسک و منابع تغذیه مصرف انرژي را بهینه سازي می کند در مقابل تکنیک پویا با استفاده از منابع نرم افزاري به این هدف دست پیدا می کند(5,4)

* 2-8-1 مدیریت استاتیک و مدیریت پویاي توان

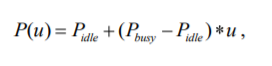
مصرف استاتیک توان یا نشت توان ، جریانی که در حال حاضر در هر مدار فعال ایجاد می شود و مستقل از نرخ 1 ساعت است. کاهش این توان نیاز به بهبود طراحی سیستم هاي سطح پایین دارد[5,4 .[ مصرف پویاي توان عمدتا نیاز به یک سناریوي کاربردي خاص دارد و وابسته به ساعت می باشد و آنرا به صورت زیرمعرفی می کنیم:



aیک فعالیت سوئیچینگ است وc خازن فیزیکی وv ولتاژ تغذیه وf فرکانس ساعت است. ارزش هايa,c با طراحی سطح پایین سیستم تعیین می شوند و کاهش پویا ترکیبی از ولتاژ و فرکانس، که به این تکنیک [[22]](#footnote-22)DVFS گفته می شود. ایده اصلی این روش در مقیاس پایین عملکرد پردازنده می باشد، زمانی که از آن به طور کامل استفاده نمی شود پس تکنیک DVFS یعنی کاهش ولتاژ و فرکانس پردازنده می باشد(5و4)

* 2-8-2مدل مصرف انرژي

یک رابطه خطی بین استفاده از پردازنده و مصرف کل یک گره وجود دارد. گره در حالت غیر فعال و زمانی که به طورکامل استفاده شده است، در زیر رابطه خطی نشان داده شده است:

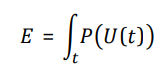


که در آن P برآورد مصرف برق می باشد و Pidle مصرف یک گره بیکارو pbusy انرژي مصرف شده توسط یک گره زمانی که به طور کامل استفاده شود و U استفاده از پردازنده می باشد و همچنین در(2-4 (مدل غیر خطی نیز نشان داده شده است(4و6)

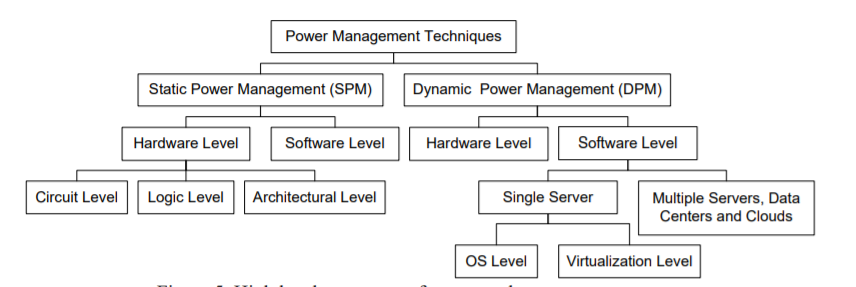


r یک پارامتر کالیبراسیون است که مربع خطا را به حداقل می رساند و به صورت تجربی بدست می آید. یا در در رابطه زیر(4)و این رابطه که در آن K ،کسري از توان مصرف شده توسط گره بیکار وpMax ،ماکزیمم توان مصرف شده هنگامی که گره به طور کامل استفاده می شود، در 250 فرض می شود(5).

و تابع انرژي کل مصرفی توسط گره هاي فیزیکی در(2-6 (تعریف می شود که (t(U تابعی از بهره برداري پردازنده در واحد زمان می باشد(4)



شکل زیرمدل های مدیریت توان درسطح نمودار درختی نشان می دهد

  
شکل 2-7 : تکنیک هاي مدیریت قدرت

همانطور که در شکل فوق نشان داده شده است تکنیک هاي DPM در سطح مجازي سازي، در سطح مراکز داده و همینطوردر سطح سیستم عامل اعمال می شوند که در سطح مراکز داده با تکنیک هاي غیر فعال کردن قطعات پویا و عملکرد مقیاس پذیري پویا عمل می کند.

غیر فعال کردن قطعات پویاDCD[[23]](#footnote-23) : قطعاتی از سیستم که عملکرد مقیاس پذیري را پشتیبانی نمی کنند نیاز به تکنیکی دارند تا زمانی که بیکار هستند، غیر فعال شوند. به طور مثال براي صرفه جویی در توان در مواقعی نیاز به خاموش کردن منبع تغذیه می باشد(4,6,5).

این تکنیک براساس روش هاي تطبیقی و پیش بینی رفتار سیستم در گذشته و آینده عمل می کند یا بر اساس تعیین پارامتراستاتیک آستانه زمان بیکاري قطعات به منظور کاهش توان و انرژي عمل می کند.

عملکرد مقیاس پذیري پویا[[24]](#footnote-24) DPS برخی قطعات مانند پردازنده این اجازه را می دهند تا با کاهش یا افزایش تدریجی فرکانس با ولتاژ به منظور کاهش توان تنظیم شوند و این رویکرد وقتی مفید واقع می شود که منابع به طور کامل استفاده نشده باشد. روش DVFS نمونه اي از تکنیک DPSمی باشد.

* ۲-۸-۳ سطح سیستم عامل

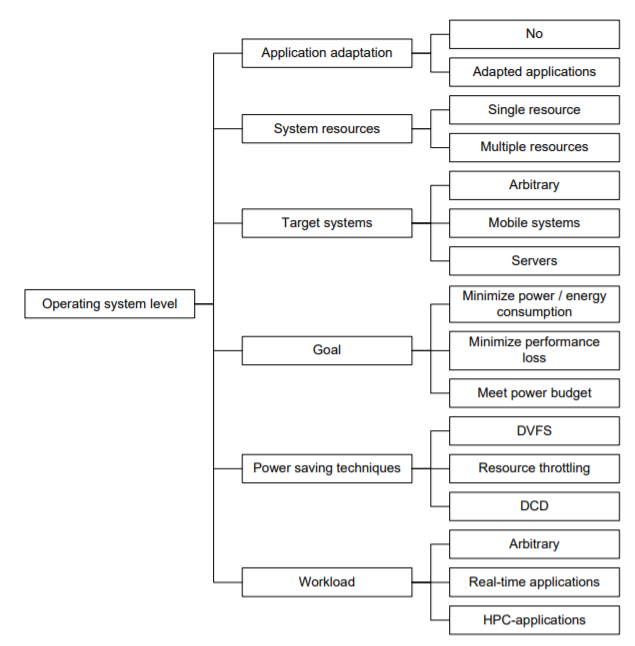
این بخش به بحث مدیریت منابع در سطح سیستم عامل می پردازد. همانطور که در شکل زیر نشان داده شده ما در این سطح ویژگی هایی داریم که ابتدا تعاریفی ازآنها خواهیم داشت: ·

انطباق نرم افزار : آیا سیستم به منظور مدیریت توان، نیاز به اعمال تغییرات در نرم افزاردارد یا نه؟

منابع سیستم: آیا سیستم در بهینه سازي منابع مانند پردازنده تکی عمل می کند یا به صورت چندتایی؟ هدف قرار دادن سیستم : آیا هدف کلی است یا مختص سیستمی خاص می باشد؟

هدف : آیا سیستم مصرف برق یا انرژي را کاهش خواهد داد؟

تکنیک هاي صرفه جویی انرژي مانند DCD,DPS(DVFS)



شکل 2-8 : مدیریت توان در سطح سیستم عامل

* ۲-۸-۴ سطح مجازي سازي

فن آوري که استفاده از منابع سیستم رابهبود می بخشد ودر نتیجه کاهش توان را بدنبال دارد تکنیک مجازي سازي می باشد. مجازي سازي یک لایه انتزاعی است بین سیستم عامل و سخت افزار که به آن مانیتور(فوق ناظر) ماشین مجازي گفته می شود که این ناظر منابع فیزیکی را تقسیم می کند به برش هاي منطقی به نام ماشین مجازي که هر ماشین مجازي می تواند یک سیستم عامل منحصر به فرد براي کاربر خود ایجاد نماید (۴) مجازي سازي این امکان را می دهد تا چندین ماشین مجازي روي یک گره فیزیکی ایجاد شود و از مزیت هاي این روش، توانایی حرکت ماشین مجازي از یک گره فیزیکی به گره دیگر می باشد. به این حرکت مهاجرت ماشین مجازي گفته می شود حال این حرکت می تواند آنلاین یا گرم و در زمان اجرا باشد و یا آفلاین یا سرد باشد.(۴)

مهاجرت ماشین مجازي در زمان اجرا از روش هاي صرفه جویی در انرژي می باشد که با تجمیع ماشین هاي مجازي برروي یک گره فیزیکی و کاهش تعداد گره هاي فیزیکی بیکارو تبدیل آنها به حالت خاموش یا هایبرنت به این هدف می رسد. لازم به ذکر است که یک گره بیکار در حدود 70%از کل انرژي را مصرف می کند(5,4) نکته مهم این است که ناظر ماشین مجازي می تواند به عنوان یک سیستم عامل آگاه به انرژي عمل نموده و با نظارت به عملکرد کلی سیستم و اعمال تکنیک هاي DVFSوDCDبر اجزاي سیستم مصرف انرژي را کاهش دهد(۴).

* ۲-۸-۵ سطح مراکز داده

یک گره بیکار حدود 70 %از مصرف انرژي را در مراکز داده به خود اختصاص می دهد. در حالت استاتیک با خاموش کردن یا به حالت خواب بردن گره در مصرف برق صرفه جویی می شود ولی در حالت پویا با تجمیع حجم کار بر یک گره و غیر فعال کردن گره هاي بیکار، این عمل سبب بهبود استفاده از انرژي خواهد شد. تجمیع یا تثبیت حجم کار ممکن است منجربه تخریب عملکرد برنامه هاي کاربردي کاربران شود و درنتیجه نقض قرارداد سطح سرویس که خود جاي بحث دارد را بدنبال داشته باشد(4 )

ویژگی هاي این سطح درشکل زیراینگونه تعریف می شوند:

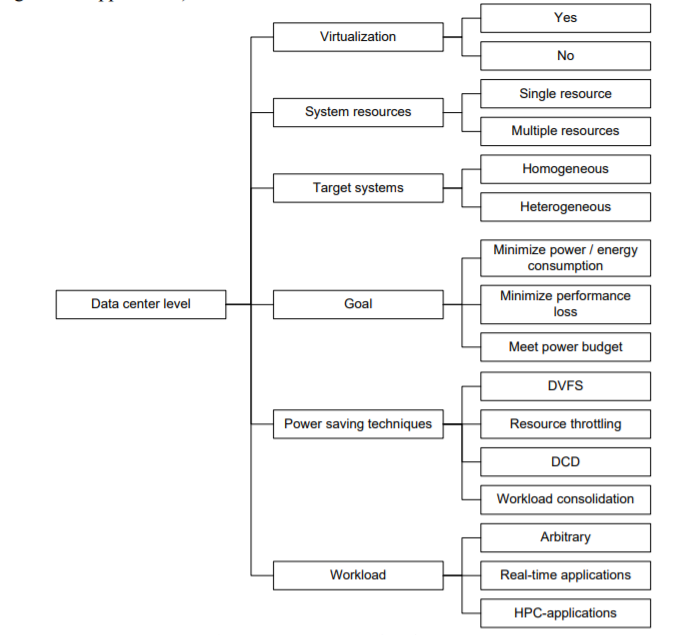
مجازي سازي: آیا این رویکرد به مجازي سازي منابع در مراکز داده می پردازد یا نه؟

هدف سیستم: روي محیط همگن عمل می کند یا محیط ناهمگن ؟

هدف: آیا مصرف انرژي یا برق را کاهش می دهد؟

تکنیک هاي به کارگرفته شده براي رسیدن به این هدف کدامند؟آیا سیستم از تکنیک هاي DPSمانند DVFSاستفاده می کند یا از تکنیک هاي DCDو یا تجمیع VMبه منظور به حداقل رساندن انرژي استفاده می کند(4)

ساختار : آیا مدیریت منابع متمرکز است یا به صورت توزیع شده می باشد.



شکل 2-9 : مدیریت توان در سطح مراکز داده

۲-۹ نتیجه گیري

در این فصل مفاهیم پایه، اصول و تعاریف مورد نیاز براي درك مفاهیم رایانش ابري مانند مفاهیم مربوط به مرکز داده ابري، تکنیک مجازي سازي، مدلهاي کاهش مصرف انرژي و تکنیک هاي مدیریت توان را در سطح سیستم عامل، در سطح مجازي سازي و سطح مراکز داده مورد بررسی قرارگرفت. براي مثال در سطح مرکز داده مشخص شد که در حالت استاتیک با خاموش کردن یا به حالت خواب بردن گره در مصرف برق صرفه جویی می شود ولی در حالت پویا با تجمیع حجم کار بر یک گره و غیر فعال کردن گره هاي بیکار، این عمل سبب بهبود استفاده از انرژي می شود.

فصـل سوم

پیشینه تحقیق

این فصل به مرور کارهاي انجام شده جهت حل مشکل، مصرف انرژي و راهکارهاي انجام شده روي این موضوع می پردازیم.

1-3 مقدمه

به دلیل اهمیت مصرف انرژي در محاسبات در مقیاس بزرگ مانند ابرها و وجود نگرانی هاي بسیار از انتشارکربن، ازبین رفتن قابلیت اطمینان سیستم در بخش هاي زیر پژوهش هاي انجام شده براي مقابله با مشکلات ذکر شده، نشان داده شده است.

* 3-1-1 کاهش مصرف انرژي در گره ها در محیط ناهمگن

هیت و همکاران در(20)مسأله توزیع درخواست کاربران به گره ها مانند وب سرورها از لحاظ کارآمدي انرژي در محیط ناهمگن را مورد بررسی قرار داده اند و براي اولین بار است که در محیط ناهمگن، کاري تحقیقاتی که توزیع حجم کار با در نظر گرفتن انرژي کارآمد، در نظر گرفته می شود. ایده این کار براساس تعیین عملکرد گره هاي ناهمگن براساس توان مصرف شده توسط منابع مانند پردازنده، دیسک سخت برآورد شده است و در مرحله بعد نرم افزارهاي مستقر در گره و نوع درخواستشان از منابع و عملکرد هرکدام و میزان مصرف برق در نظرگرفته می شود و با غیر فعال کردن گره اي که هیچ درخواستی به آن نشده است و هیچ حجم کاري روي آن نمی باشد. پژوهش هاحاکی ازآن است که روش ارائه شده، قادر به کاهش انرژي به میزان 42%می باشند.

* 3-1-2 تجمیع انرژي آگاه براي محاسبات ابري

سریکانتایاه و همکاران د(42) به تجمیع ماشین مجازي در مراکز داده ابري به منظور بهینه سازي انرژي می پردازند و به عنوان نخستین گام در جهت تجمیع به روابط متقابل بین مصرف انرژي و استفاده از منابع می پردازند. میزان مصرف انرژي حاکی ازآن است که وقتی منابع به درستی استفاده نشود منجر به هزینه هاي بالاتر و زمان اجراي طولانی تر ودرنتیجه مصرف انرژي بیشتر خواهد شد و با توجه به نتایج بدست آمده و اظهارداشت نویسندگان پژوهش، هدف از انرژي آگاه با تجمیع حجم کار براي حفظ گره هاي خوب می باشد.

* 3-1-3 تخصیص بهینه گره

گاندي و همکاران در(15) میزان مصرف برق گره ها در ایالات متحده را مورد بررسی قرار دادند و مصرف بیش از 5/1از کل برق در ایالات متحده را و هزینه نزدیک به 5/4 میلیارد دلاري را در مزارع گره ها می بینند و با توجه به افزایش قیمت انرژي بسیاري از صنایع به دنبال راهکاري براي بهترین استفاده از انرژي می باشند. ایده آنها این بود که براي رسیدن به بهترین عملکرد، یک گره باید در بالاترین سطح قدرت خود قرار گیرد و طرح تخصیص بهینه به عوامل بسیاري بستگی دارد.

آنها با استفاده از تکنیک هاي مقیاس پذیري فرکانس پویا[[25]](#footnote-25) ، مقیاس پذیري ولتاژ - فرکانس پویا[[26]](#footnote-26) و ترکیب این دو قدرت گره را به حداکثر رسانده و آزمایشات حاکی از آن است، با توجه به رابطه قدرت با فرکانس و با استفاده از طرح تخصیص بهینه می توان به طور قابل توجهی به کاهش متوسط زمان پاسخ و کاهش مصرف انرژي دست یافت.

* 3-1-4 مدیریت برق

ناتوجی و همکاران در (35)به بررسی مشکل مدیریت منابع در مراکز داده مجازي در مقیاس بزرگ پرداختند و این اولین باربود که تکنیک مدیریت انرژي و توان در زمینه سیستم هاي مجازي استفاده شد. علاوه براین تکنیک مقیاس پذیري سخت افزار وتجمیع ماشین هاي مجازي اولین بار در کنار هم و اعمال روش مدیریت انرژي توسط نویسندگان به نام روش مقیاس پذیري منابع نرم افزاري صورت گرفت. هدف از این روش بهره برداري از ماشین هاي مجازي مهمان[[27]](#footnote-27) می باشد و نویسندگان مدیریت منابع را در دوسیاست محلی و جهانی تقسیم کرده اند. در سطح محلی مدیریت توان ماشین مجازي مهمان در هر دستگاه فیزیکی می باشد و در سیاست جهانی که مدیریت چندین ماشین مجازي را به عهده دارد به تجمیع ماشین هاي مجازي با استفاده از تکنیک مهاجرت به منظور آزاد سازي گره سبک بار و متقابلا صرفه جویی در انرژي می پردازد. آزمایشات حاکیاز آن است که استفاده از روش ارائه شده منجر به هماهنگی مؤثر از ماشین هاي مجازي و سیاست هاي مدیریت انرژي و کاهش مصرف انرژي تا 34 %خواهد شد

رمیا راگاوندرا و همکاران در (36)به مصرف برق و گرماي حاصل شده، به عنوان چالش هاي کلیدي در محیط هاي مرکز داده اشاره می کنند و طبقه بندي آنان نیز در هردو سطح محلی و جهانی می باشد. روش آنها از یک کنترلرماشین مجازي[[28]](#footnote-28) استفاده می کند تا بر کنترل بهینه سازي مصرف برق، بهره برداري از منابع، پیش بینی تقاضاي آینده و تنظیم پردازنده نظارت نماید. این کنترلر، مصرف برق Vms را در سراسرگره فیزیکی کنترل می کند و با تجمیع ماشین هاي مجازي و تعویض گره ها از آماده به کار یا بالعکس به صرفه جویی در انرژي می رسد.

* 3-1-5 قدرت و عملکرد مدیریت

دارا کیوزیک و همکاران در(24 )هدف از کار خود را به حداقل رساندن مصرف برق و جلوگیري از نقض قراردادسطح سرویس می نامند و اینکار را با استفاده از یک کنترلر انجام می دهند به نام کنترلر محدود به نگاه جلو که مدیریت منابع را مانند به اشتراك گذاري پردازنده به ماشین هاي مجازي وتجمیع و مهاجرت آنلاین ماشین هاي مجازي به همراه تغییر حالت گره از روشن به خاموش یا بالعکس به عنوان مکانیزمی در صرفه جویی قدرت اعمال می کنند. نویسندگان این مقاله با ارائه یک مدل ریاضی براي حل مشکل بهینه سازي قدرت نشان دادند که LLC، میزان 26 %در هزینه هاي مصرف برق براي یک دوره 24 ساعته صرفه جویی کرده و به میزان 6,1 %از درخواست ها با نقض SLAمواجه شده اند.

* 3-1-6 تخصیص منبع با استفاده از خوشه هاي مجازي

مارك استیل ول و همکاران در (43)به مقایسه الگوریتم هاي تخصیص منبع پرداخته و الگوریتم بین پکینگ[[29]](#footnote-29) در محیط همگن مجازي را به عنوان بهترین الگوریتم تخصیص نام می برند. روند کاري این الگوریتم بدین گونه است که با مرتب کردن نزولی وظایف و پویا گرفتن حجم کار و استفاده از یک ناظر با عنوان مدیر ماشین هاي مجازي به تخصیص منابع می پردازد. مدیر ماشین مجازي امکان مهاجرت ماشین هاي مجازي را درمیان گره هاي فیزیکی می دهد

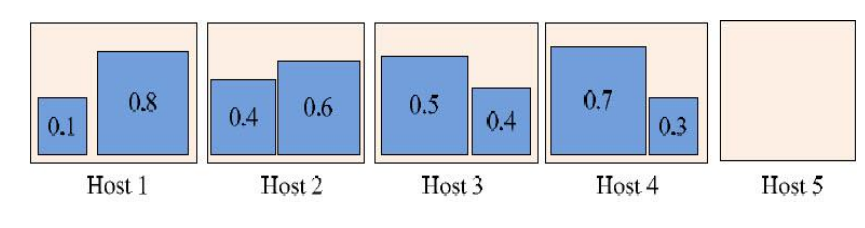
. سانگ و همکاران در (41)در جهت بهبود مصرف انرژي و کاهش آن استفاده درست و کارآمد از منابع را در مراکز داده مجازي مورد مطالعه قرار داده اند. منابعی که در این کار در نظر گرفته می شود پردازنده و حافظه اصلی می باشند. منابع براي اطمینان از کیفیت سرویس دهی براساس الویت هاي برنامه در اختیارش قرار می گیرد. یعنی زمانی که منابع محدود می باشد عملکرد یک برنامه با الویت پایین را عمدا تخریب کرده و منابع را به برنامه اي مهم و با الویت بالا اختصاص می دهد. نویسندگان برنامه ریزي را در سه سطح انجام می دهند: زمانبندي در سطح برنامه براي فرستادن درخواست به تمام ماشین هاي مجازي، زمانبندي در سطح محلی منابع ماشین مجازي در حال اجرا، زمانبندي در سطح جهانی و کنترل زمانبندي بین تمام برنامه هاي کاربران .

* 3-1-7چارچوب آگاه از توان و انرژي

ورما و همکاران در(44)به بررسی و طراحی چارچوبی آگاه از توان در یک محیط مجازي ناهمگن می پردازد و با استفاده از تکنیک هاي سخت افزاري مانند DVFS و تکنیک مجازي سازي به مدیریت انرژي می پردازند و با تعریف یک مدیر جهانی به تخصیص ماشین مجازي جدید و همچنین تخصیص دوباره ماشین مجازي مهاجرت کرده می پردازند و آنان با استفاده از اندازه ماشین هاي مجازي هزینه مهاجرت آنان را بدست می آورند. نویسندگان به مقایسه چندین الگوریتم تعریف شده براي حل مشکل بهینه سازي توان پرداخته و مشکلاتی که این الگوریتم ها دارند مانند الگوریتم بین پکینگ که داراي اشکالاتی بود مانند اندازه متغیر بین و هزینه بسته بندي بین ها، که براي حل مشکل بسته بندي بین، نویسندگان از الگوریتم اولین انتخاب به صورت کاهشی استفاده کردند. نتایج نشان می دهند که استفاده از این چارچوب در حدود 25 %در توان صرفه جویی شده است.

### **3-1-7-1اولین انتخاب به صورت کاهشی**

در این روش ابتدا کالاها به صورت نزولی مرتب شده، سپس با انتخاب بزرگترین کالا، بسته هاي موجود یکی یکی بررسی می شوند تا به اولین بسته اي که بتواند کالاي انتخابی را در خود جاي دهد برسد، سپس کالادر آن قرار می گیرد. در شکل زیر چگونگی این روش نشان داده شده است.



شکل3-1 :تخصیص ماشین هاي میزبان به ماشین هاي مجازي با استفاده از روش FFD

* 3-1-8ابر سبز : انرژي کارآمد و مدیریت منابع

بویا و همکاران در (5)به چالش ها و معماري براي مدیریت انرژي کارآمد در محاسبات ابري سبز و به ارائه راه حل هایی براي محاسبات ابري سبز پرداخته اند که نه تنها می تواند با کاهش انرژي تولید گازهاي گلخانه اي را کاهش دهد و محیط زیست رانجات دهد، بلکه کاهش هزینه هاي عملیاتی را نیز بدنبال خواهد داشت. منابع در چند سطح ارائه می شوند: در اولین سطح، کاربر درخواست خود را به کارگزار[[30]](#footnote-30) براي تأمین منبع مانند پردازنده می دهد و پس از تعیین یک مهلت زمانی[[31]](#footnote-31) از طرف کاربر، کارگزار درخواست کاربر را به تعدادي از مراکز داده می دهد براي تأمین ماشین مجازي، مراکز داده با در اختیار کذاشتن ماشین مجازي براساس مهلت مورد نیاز کاربر این درخواست را برآورده می نمایند.

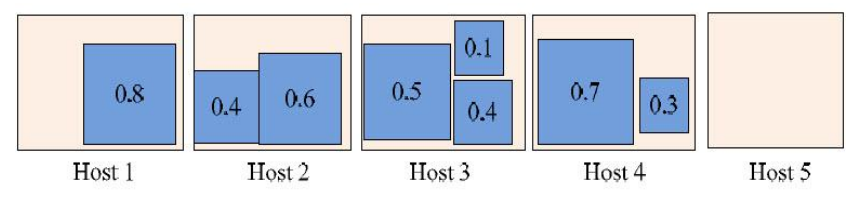
### **3-1-8-1 سیاست تخصیص**

در این پژوهش(5)سیاست تخصیص ماشین مجازي به دو بخش تقسیم می شود: بخش اول، پذیرش درخواست هاي جدید براي ارائه ماشین مجازي و قراردادن ماشین مجازي بر روي گره و در بخش دوم بهینه سازي تخصیص ماشین هاي مجازي فعلی بعد از مهاجرت می باشد.

بدلیل مشکل بسته بندي الگوریتم بین پکینگ، آنان ازاصلاح الگوریتم بهترین انتخاب به صورت کاهشی به منظور جایگذاري ماشین هاي مجازي به گره استفاده کردند، بدین صورت که تخصیص براساس حداقل افزایش مصرف انرژي بعد از این تخصیص، صورت می گیرد. پیچیدگی این الگوریتم m.n می باشد که n تعداد ماشین هاي مجازي و m تعداد میزبان هاي فیزیکی می باشد.

### **3-1-8-2بهترین انتخاب به صورت کاهشی**

در این روش ابتدا پاکت ها به صورت نزولی مرتب می شوند، سپس با انتخاب بزرگترین کالا به جایگذاري آن درون بسته اي با بهترین انتخاب اقدام می نماید، بهترین انتخاب جعبه اي است کهکمترین فضاي باقی مانده پس از جایگذاري را داشته باشد.. در شکل 3-2 چگونگی این روش نشان داده شده است.(32).



شکل 3-2 :تخصیص ماشین هاي میزبان به ماشین هاي مجازي با روش BFD

بر اساس الگوریتم جایگذاري ماشین مجازي، اصلاح الگوریتم بهترین انتخاب به صورت کاهشی ابتدا ماشین هاي مجازي به صورت نزولی و براساس میزان بهره وري پردازنده شان مرتب می شوند و این ماشین هاي مرتب شده به گره هایی تخصیص داده خواهند شد که کمترین افزایش توان را پس از تخصیص ایجاد نمایند(6,5,4)

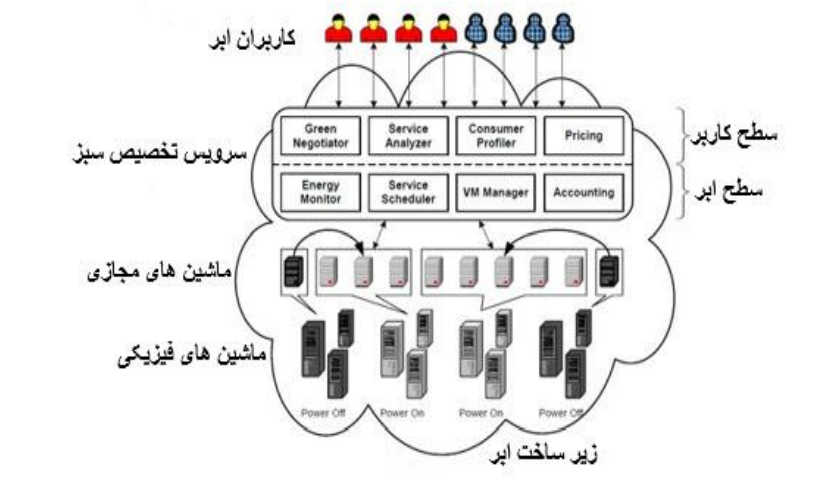
### **3-1-8-3سیاست انتخاب ماشین هاي مجازي**

نویسندگان(5)سه سیاست براي انتخاب ماشین هاي مجازي که مجبور به مهاجرت می شوند تعریف می کنند(2)

سیاست کمترین زمان مهاجرت : کاهش زمان مهاجرت با انتخاب ماشین هاي مجازي براساس کمترین میزان حافظه آنها و به حداقل رساندن سربار مهاجرت .

بالاترین رشد بالقوه[[32]](#footnote-32) : مهاجرت ماشین هاي مجازي که داراي کمترین میزان استفاده از پردازنده هستند به منظور به حداقل رسانیدن نقض قرارداد سطح سرویس.

انتخاب تصادفی : انتخاب تعداد لازم از ماشین هاي مجازي براي مهاجرت با توجه به متغیر توزیع تصادفی . آنان در سیاست انتخاب از روش هاي کران بالا و پایین بهره وري میزبان ها به عنوان معیار انتخاب استفاده کرده به گونه اي که بهره وري کلی پردازنده از مقادیر معین تجاوز نکند. شکل ساختار سیستم کارآمد براي انرژي در زیر ساخت ابر نشان داده شده است:

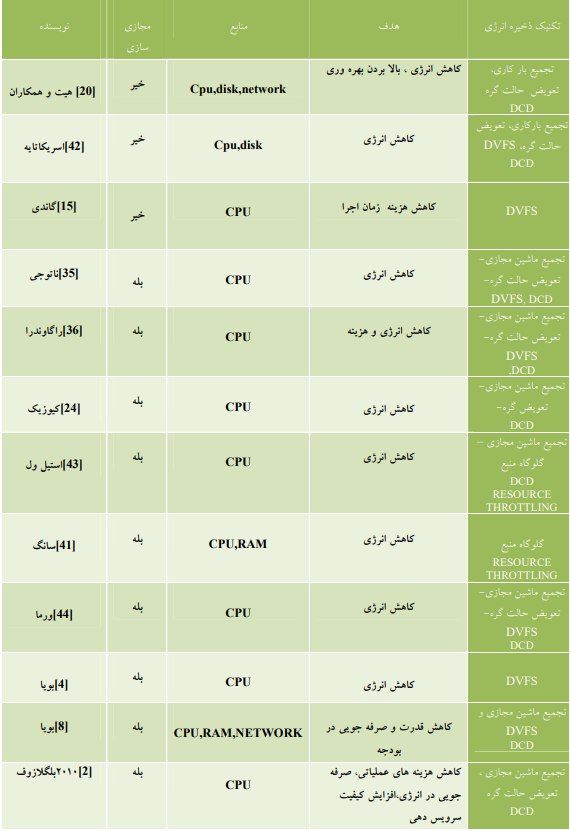


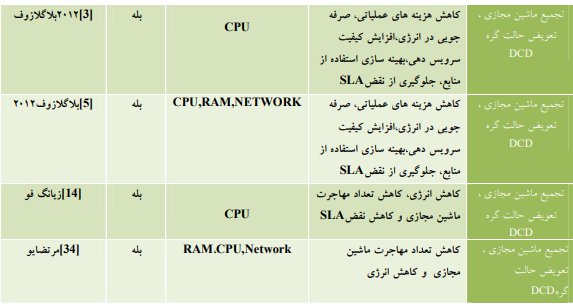
شکل 3-3 : ساختار سیستم انرژي کارآمد براي محاسبات ابري سبز

بلاگلازوف در(2 )یک روش فرااکتشافی براي تخصیص منابع انرژي آگاه و تجمیع ماشین هاي مجازي پیشنهاد داد. آنها یک مجموعه آستانه بالا براي استفاده از پردازنده در یک گره تعیین کردند و سپس در طی یک دوره ثابت آنها را بررسی کردند، اگر استفاده از گره بیش از حد آستانه باشد، گره مشخص می شود و سپس ماشین مجازي براي مهاجرت از گره مذکور انتخاب می شود. بلاگلازوف در کار بعدي خود(5)به جاي آستانه ثابت از یک متغیر استفاده نمود.

زیانگ فو در (14)کارهاي انجام گرفته توسط تیم ابر (3و5)مورد مطالعه قرار داده است و به منظور کاهش مصرف انرژي از سیاست بهبود انتخاب ماشین مجازي که بر اساس استفاده از پردازنده و سیاست تخصیص ماشین مجازي مهاجرت داده شده به یک میزبان با استفاده از روش حداقل ضریب همبستگی (یعنی اینکه آیا با قرار دادن این ماشین مجازي مهاجرت داده شده عملکرد این میزبان دچار تخریب شده و بر روي عملکرد ماشین هاي مجازي موجود در این میزبان اختلال ایجاد می شود) بهره برده است.

عزیز مرتضائو و سانگ یون اه در(34)به تجمیع گره ها با استفاده از الگوریتم مهاجرت ماشین مجازي در محاسبات ابري سبز پرداخته اند و ازآنجاکه مهاجرت یک عمل پر هزینه براي تأمین کننده ابر می باشد، هدف دوم این نویسندگان به حداقل رسانیدن تعداد مهاجرت ها می باشند و با بکارگیري الگوریتم خودشان و مقایسه آن با الگوریتم هاي اکتشافی بین- پکینگ مانند اولین انتخاب به صورت کاهشی و بهبود آن به این مهم دست یافته اند، اما چالش هاي این کار در این است که در حالی که تعداد ماشین مجازي در حال افزایش می باشد، تعداد مهاجرت نیز بسیار افزایش پیدا می کنند و به چگونگی دوباره تخصیص ماشین مجازي مهاجرت داده شده نمی پردازد، همچنین دراین کار به کاهش انرژي پرداخته نشده است. درجدول زیر تحقیقات پیشین در سطح مراکز داده آورده شده است

جدول3-1:مقایسه تحقیقات پیشین در سطح مراکز داده 



3-2 نتیجه گیري

در این فصل با توجه به اهمیت مصرف انرژي در محاسبات ابري در مقیاس بزرگ، به کاهش مصرف انرژي در گره ها در محیط ناهمگن پرداخته شد و روش هاي حل شده در این محیط مورد بحث و بررسی قرار گرفت.

فصـل چهارم

روش کار

این فصل به بیان نوآوري این پژوهش می پردازد.

4-1مقدمه

در این فصل چگونگی روش به کار برده شده، به منظور کاهش تعداد مهاجرت و کاهش انرژي در مراکز داده ابر می پردازد.

4-2روش کار

در محیط ابر مجموعه اي گره فیزیکی وجود دارد، که برنامه هاي کاربردي، در حال اجرا بر روي آنها می باشند. فرض شده است، در این مجموعه پس از مدتی در نحوه چیدمان ماشین هاي مجازي بر روي گره ها پراکندگی دیده شود، به عبارتی تعداد گره هاي بیکار روبه افزایش باشند و براي صرفه جویی در انرژي باید روندي بکارگرفته شود تا با مهاجرت ماشین هاي مجازي از این گره هاي بیکار و تخصیص دوباره این ماشین هاي مجازي به گره هاي مناسب به هدف موردنظر رسید. پس اولین هدف این است که از طریق فناوري مهاجرت ماشین مجازي، با مهاجرت ماشین هاي مجازي، حداقل گره ها استفاده شود. از آنجا که مهاجرت عملی پر هزینه است، دومین هدف کاهش تعداد مهاجرت است. از نتایج اهداف ذکر شده، هدف اصلی یعنی کاهش انرژي مصرفی خواهد بود.

ابتدا لازم است براي استفاده از گره ها، میزان آستانه، استفاده از پردازنده گره مشخص شود. چراکه استفاده 100% از پردازنده به تخریب کارائی منجر خواهد شد. انتخاب این مقدار بسیار مهم است، به دلیل آنکه اگر مقدار آستانه بسیار بالا انتخاب شود، برعملکرد ماشین هاي مجازي در حال اجرا در آن گره اثر گذاشته و کارایی آن گره را کاهش می دهد و اگر بسیار پایین انتخاب شود دیگر تجمیعی رخ نخواهد داد. دراین تحقیق با توجه به (5و24)این مقدار بین 50 %تا 90 %انتخاب می شود.

راهکارپیشنهادي از سه کاراکتر پردازنده، حافظه و پهناي باند براي ماشین هاي مجازي و گره ها استفاده می کند

اندازه دیسک درنظر گرفته نمی شود و فرض می شود، که یک شبکه ذخیره سازي متصل به عنوان یک ذخیره ساز اصلی در امتداد خوشه وجود دارد .

ci ظرفیت پردازنده (گیگاهرتز) و mi ظرفیت حافظه به مگابایت است، معرفی می ، که گره ي i با دو بعد ci و mi شود. براي ماشین هاي مجازيj ،نیز دوبعد ظرفیت پردازنده Vcj و ظرفیت حافظه ماشین Vmj درنظر گرفته می شود

. اگر فرض شود، M تا گره و K تا ماشین مجازي وجود دارد عمل تجمیع اینگونه عمل می کند که : {Sm,………,S3,S2,S1}از خوشه Si ،که نشان دهنده حالت گره iام می باشد و مجموعه {ip,…..i2,i1 }که درآن ip………….i2,i1 عضو بازه [k,1]که نشان دهنده شماره شناسایی vm که در حال حاضر در گره i ساکن است.

باید در نظر گرفته شود، که براي هر گره i ، ظرفیت کلی ماشین هاي مجازي روي یک گره نمی تواند از ظرفیت خود گره(ni )بیشتر باشد:



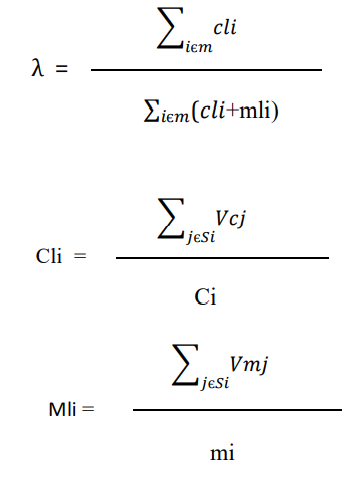
سه معیاري که بر تجمیع گره ها تأثیر گذار می باشند، استفاده از تعداد گره ها، تعداد گره هاي آزاد شده، تعداد مهاجرت ماشین مجازي هستند، که هدف بر معیار اول و سوم می باشد. کاهش معیارها، با کمک ارث بري از الگوریتم اولین انتخاب به صورت کاهشی، با تخصیص ماشین مجازي مهاجرت داده شده و چگونگی مدل تجمیع گره هاي فیزیکی بدست خواهد آمد.

* 1-2-4مدل تجمیع گره هاي فیزیکی

M تا گره براساس رتبه شان با استفاده از رابطه زیربه صورت نزولی مرتب می شوند:



که به ترتیب λ از cli و mli ازروابط زیر بدست خواهد آمد:



گره ها براساس رتبه شان به صورت نزولی مرتب می شوند و آخرین گره از این لیست انتخاب می شود ، یعنی کوچکترین گره از لحاظ رتبه، و بررسی می شود که آیا مهاجرت امکان پذیر است یا نه. این منطق در این الگوریتم گنجانده می شود که اگرحتی یک ماشین مجازي از این گره نتواند مهاجرت داده شود، این گره نمی تواند آزاد شود

به عبارتی اگر m ، لیست تعداد ماشین درحال مهاجرت از یک گره باشد، منطق ایجاب می کند که m تا ماشین باید مهاجرت داده شوند، یا هیچ کدام از ماشین ها ي مذکور نمی توانند مهاجرت داده شوند. بنابراین گره هاي داراي حداقل بارکاري آزاد خواهند شد و از مهاجرت هاي متعدد جلوگیري می شود، چراکه با این روش گره اي انتخاب خواهدشد، که داراي کمترین بارکاري و کم ترین تعداد ماشین مجازي ساکن برآن است. در نتیجه کاهش تعداد مهاجرت را بدنبال خواهد داشت و این مراحل تکرار خواهد شد تا دیگر هیچ مهاجرتی امکان پذیر نباشد

* 4-2-2 مدل مهاجرت ماشین مجازي

ماشین هاي مجازي که روي گره آخر (کمترین بار کاري) در این لیست می باشند، انتخاب می شوند وکاندیداي مهاجرت می شوند. سپس همین لیست انتخابی حاوي ماشین هاي مجازي نیز به صورت نزولی براساس رتبه شان طبق رابطه زیرمرتب می شوند.



براي تخصیص ماشین مجازي از این لیست استفاده می شود. شرایط توقف این الگوریتم به شرح زیر خواهد بود: با تعیین متغیر کل مهاجرت و امکان پذیر بودن مهاجرت تمام ماشین هاي مجازي از گره انتخاب شده و شرط دیگر توقف حلقه، ME0 که حداقل بهره وري می باشد و مقدار آن1,0 درنظرگرفته می شود. یعنی اگر میزان بهره وري مهاجرت طبق رابطهکمتر از 1,0 بدست آمد، آنگاه الگوریتم پایان پذیرد.

* 4-3-2مدل تخصیص ماشین مجازي

ماشین هاي مجازي براساس رتبه شان در لیست مهاجرت قرار می گیرند و منتظر قرار گرفتن بر گره مورد نظر می باشند. این الگوریتم به صورت دوره اي لیستی از ماشین هاي مجازي در حال مهاجرت را و لیستی از گره هاي موجود (به غیر از گره هاي آزاد شده) را می گیرد. براي معرفی پارامترهایش ابتدا تابع هدفی تعریف می شود، بدین صورت که ماشین مجازي بهگره اي داده شود که با اضافه شدن آن ماشین به آن گره، بار کل گره موردنظر از حد بالاي آستانه(یعنی 90%بیشتر نشود.

در پایان با گرفتن توان از گره هاي فیزیکی، میزان انرژي مصرفی بر حسب کیلووات در ساعت با استفاده از رابطه زیربدست آورده می شود.

انرژي = توان مصرفی مراکز داده بر اساس نوع مدل پاور / (1000\*3600

بهره وري مهاجرت = (تعداد گره هاي آزاد شده / تعداد مهاجرت ماشین مجازي) \* 100

4-3 نتیجه گیري

در این فصل روش کار ارائه شد. که با ارث بري از الگوریتم اولین انتخاب به صورت کاهشی و با بدست آوردن رتبه گره هاي موجود ورتبه ماشین هاي مجازي و با در نظر گرفتن حداکثر حد آستانه عمل تخصیص ماشین هاي مجازي مرتب شده در لیست مهاجرت را انجام داد. در پایان میزان مصرف انرژي و میزان بهره وري بدست آمد.

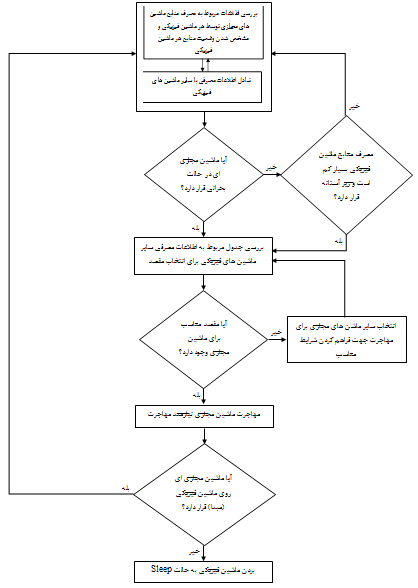
فصـل پنجم

کدهاشبیه سازی ها

در این فصل کدهاي شبیه سازي شده به طور کامل شرح داده می شوند.

5-1مقدمه

در این بخش به تشریح کامل کد نویسی های مربوط به پیاده سازی مهاجرت زنده ماشینهای مجازی در رایانش ابری خواهیم پرداخت. مهاجرت زنده ماشین های مجازی بر اساس انتقال لاگ ها انجام می شود. با انقال لاگ ها، مصرف انرژی ماشین های مجازی مدیریت شده و در حالتی که هر ماشین مجازی از جنبه مصرف باطری یا انرژی در پایین تر از حد آستانه قرار گیرد، کلیه ماشین های مجازی همسایه که دارای بیشترین میزان مصرف انرژی و کمترین فاصله با گره فعلی کاوش شده و سپس ماشین مجازی مناسب انتخاب و مابقی مهاجرت ادامه پیدا می کند. در کل سناریوی پیشنهادی مطابق با فلوچارت زیر می باشد.



با توجه به فلوچارت بالا، کد های پیاده سازی شده به شرح ذیل می باشد:

در تکه کدهای زیر فایلهای کتابخانه ای مربوط به زبان برنامه نویسی جاوا و همچنین کلود سیم جهت بار گذاری مراکز داده، هاست ها و ماشینهای مجازی در ابتدای شبیه سازی بارگذاری می گردد.

package RM\_Balancing\_vms;

import org.cloudbus.cloudsim.examples.power.\*;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import java.text.DecimalFormat;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.LinkedList;

import java.util.List;

import java.util.Map;

import java.util.Scanner;

یکی از مهمترین علت های استفاده از فایلهای کتابخانه کلود سیم در محیط جاوا فراهم نمودن امکان لازم جهت مدیریت و اجرای ایده هایی در محیط رایانش ابر است. در این کلیه فایل های کتابخانه ای مربوط به هاست ها، مراکز داده، ماشین های مجازی، تخصیص منابع به هر ماشین مجازی، بروکر ها، زمانبندی کارها، تعادل بار پویا جهت انتخاب ماشین مجازی بهینه با مصرف انرژی لازم و غیره ایمپورت شده اند.

import org.cloudbus.cloudsim.Cloudlet;

import org.cloudbus.cloudsim.CloudletSchedulerDynamicWorkload;

import org.cloudbus.cloudsim.Datacenter;

import org.cloudbus.cloudsim.DatacenterBroker;

import org.cloudbus.cloudsim.DatacenterCharacteristics;

import org.cloudbus.cloudsim.Host;

import org.cloudbus.cloudsim.HostDynamicWorkload;

import org.cloudbus.cloudsim.HostStateHistoryEntry;

import org.cloudbus.cloudsim.Log;

import org.cloudbus.cloudsim.Pe;

import org.cloudbus.cloudsim.Storage;

import org.cloudbus.cloudsim.Vm;

import org.cloudbus.cloudsim.VmAllocationPolicy;

import org.cloudbus.cloudsim.VmSchedulerTimeSharedOverSubscription;

import org.cloudbus.cloudsim.VmStateHistoryEntry;

import org.cloudbus.cloudsim.power.PowerDatacenter;

import org.cloudbus.cloudsim.power.PowerDatacenterBroker;

import org.cloudbus.cloudsim.power.PowerHost;

import org.cloudbus.cloudsim.power.PowerHostUtilizationHistory;

import org.cloudbus.cloudsim.power.PowerVm;

import org.cloudbus.cloudsim.power.PowerVmAllocationPolicyMigrationAbstract;

import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;

import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;

import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

import org.cloudbus.cloudsim.util.MathUtil;

در این قسمت تمامی توابع و متدهایی که برای عملیات مختلفی نوشته شده اند

public class MainCode {

/\*\*

\* Creates the vm list.

\*

\* @param brokerId the broker id

\* @param vmsNumber the vms number

\*

\* @return the list< vm>

\*/

در این قسمت مجموعه ای از ماشین های مجازی در هر هاست موجود در یک دیتاسنتر تولید می گردند.

public static List<Vm> createVmList(int brokerId, int vmsNumber) {

List<Vm> vms = new ArrayList<Vm>();

for (int i = 0; i < vmsNumber; i++) {

int vmType = i / (int) Math.ceil((double) vmsNumber / Constants.VM\_TYPES);

در قسمت زیر نیز هر ماشین مجازی که تولید می شود به آن یک میزان انرژی مصرفی، یک میزان حافظه رم، سیستم عامل مورد نظر و سایر پیش نیاز های لازم اختصاص داده شده و در لیست ماشین های مجازی اضافه می شود.

vms.add(new PowerVm(i,brokerId,Constants.VM\_MIPS[vmType],Constants.VM\_PES[vmType],Constants.VM\_RAM[vmType],Constants.VM\_BW,Constants.VM\_SIZE,1,"Xen",new CloudletSchedulerDynamicWorkload(Constants.VM\_MIPS[vmType], Constants.VM\_PES[vmType]),Constants.SCHEDULING\_INTERVAL));

}

return vms;

}

در این قسمت یک تعداد هاست تعریف شده و با توجه به تعداد هاست های تعیین شده، میزبانهای مربوطه تولید شده و به لیست میزبان ها اضافه می گردد.

public static List<PowerHost> createHostList(int hostsNumber) {

List<PowerHost> hostList = new ArrayList<PowerHost>();

for (int i = 0; i < hostsNumber; i++) {

int hostType = i % Constants.HOST\_TYPES;

List<Pe> peList = new ArrayList<Pe>();

for (int j = 0; j < Constants.HOST\_PES[hostType]; j++) {

peList.add(new Pe(j, new PeProvisionerSimple(Constants.HOST\_MIPS[hostType])));

}

hostList.add(new PowerHostUtilizationHistory(i,new RamProvisionerSimple(Constants.HOST\_RAM[hostType]),new BwProvisionerSimple(Constants.HOST\_BW),Constants.HOST\_STORAGE,peList,new VmSchedulerTimeSharedOverSubscription(peList),Constants.HOST\_POWER[hostType]));

}

return hostList;

}

در این قسمت سرویس دهنده های بروکر که عملیات را به صورت غیره متمرکز انجام می دهند تعریف می گردد. مراکز داده در واقع  بین تولید کنندگان و مصرف کنندگان منابع یک کارگزار (Broker) قرار گرفته است. پس از احراز هویت، مصرف کنندگان (کاربران) منابع با کارگزاران برای اجرای برنامه های کاربردی خودشان و دسترسی به منابع راه دور اتصال برقرار می کنند. کارگزاران وظیفه کشف، انتخاب و تجمیع منابع، انتقال منابع و برنامه ها، اجراء عملیات، دسترسی راه دور به منابع و گردآوری نتایج حاصل را بر عهده دارد.

public static DatacenterBroker createBroker() {

DatacenterBroker broker = null;

try {

broker = new PowerDatacenterBroker("Broker");

} catch (Exception e) {

e.printStackTrace();

System.exit(0);

}

return broker;

}

/\*\*

\* Creates the datacenter.

\*

\* @param name the name

\* @param datacenterClass the datacenter class

\* @param hostList the host list

\* @param vmAllocationPolicy the vm allocation policy

\* @param simulationLength

\*

\* @return the power datacenter

\*

\* @throws Exception the exception

\*/

در این قسمت دیتاسنترهای لازم تعریف و تولید می گردد. هر دیتاسنتر یا مرکز داده نیز شامل یک سری ویژگی ها و خصوصیات می باشد که عبارتند زا : نام دیتاسنتر، کلاس دیتاسنتر، لیست هاست های موجود در دیتاسنتر، سیاست تخصیص منابع، مدت اجرای مرکز داده. در نهایت میزان مصرف انرژی توسط دیتاسنتر محاسبه و بر گردانده می شود.

public static Datacenter createDatacenter(String name,Class<? extends Datacenter> datacenterClass,List<PowerHost> hostList,VmAllocationPolicy vmAllocationPolicy)

throws Exception

{

در قسمت زیر برای دیتاست تعیین شده پارامترهای مربوطه مشخص شده است. جلوی هر معیار مقدار و توضیح آن نشان داده شده است.

String arch = "x86"; // system architecture

String os = "Linux"; // operating system

String vmm = "Xen";

double time\_zone = 10.0; // time zone this resource located

double cost = 3.0; // the cost of using processing in this resource

double costPerMem = 0.05; // the cost of using memory in this resource

double costPerStorage = 0.001; // the cost of using storage in this resource

double costPerBw = 0.0; // the cost of using bw in this resource

DatacenterCharacteristics characteristics = new DatacenterCharacteristics(arch,os,vmm,hostList,time\_zone,cost,costPerMem,costPerStorage,costPerBw);

Datacenter datacenter = null;

try {

datacenter = datacenterClass.getConstructor(String.class,

DatacenterCharacteristics.class,VmAllocationPolicy.class,List.class,Double.TYPE).newInstance(name,characteristics,vmAllocationPolicy,

new LinkedList<Storage>(),Constants.SCHEDULING\_INTERVAL);

} catch (Exception e) {

System.exit(0);

}

return datacenter;

}

این تابع قبل از اینکه هر هاست یا میزبان را در حالت sleep یا خاموش شدن قرار دهد، زمانهای انجام کار و پردازش ها را دریافت می کند. در واقع کلیه میزبان های موجود در یک دیتاسنتر کاوش شده و مدت زمان اجرای مربوط به هر هاست را محاسبه و ارزیابی می کند. در این قسمت شرایط بحرانی برای ماشین های مجازی از لحاظ مصرف انرژی بررسی میگردد. در صورتی که از یک حد آستانه کمتر باشد ماشین مجازی در حالت شات دون میرود.

public static List<Double> getTimesBeforeHostShutdown(List<Host> hosts) {

List<Double> timeBeforeShutdown = new LinkedList<Double>();

for (Host host : hosts) {

boolean previousIsActive = true;

double lastTimeSwitchedOn = 0;

for (HostStateHistoryEntry entry : ((HostDynamicWorkload) host).getStateHistory()) {

if (previousIsActive == true && entry.isActive() == false && entry<=10 ) {

timeBeforeShutdown.add(entry.getTime() - lastTimeSwitchedOn);

}

if (previousIsActive == false && entry.isActive() == true) {

lastTimeSwitchedOn = entry.getTime();

}

previousIsActive = entry.isActive();

}

}

return timeBeforeShutdown;

}

/\*\*

\* با استفاده از این متد زمان شروع عملیات در دیتاسنترددریافت مگردد

\*

\* @param vms the vms

\* @return the times before vm migration

\*/

با استفاده از این متد زمانی که عملیات مهاجر زنده و غیره متمرکز ماشین های مجازی صورت می گیرد دریافت شده و در واقع مدت زمان انجام مهاجرت زنده ماشین های مجازی از یک سرور به سرور دیگر را محاسبه می کند. انتقال لاگ ها در این قسمت انجام می شود.

public static List<Double> getTimesBeforeVmMigration(List<Vm> vms) {

List<Double> timeBeforeVmMigration = new LinkedList<Double>();

for (Vm vm : vms) {

boolean previousIsInMigration = false;

double lastTimeMigrationFinished = 0;

در این قسمت اطلاعات سایر ماشین های همجوار به روش حریصانه بررسی شده و در هر حالت ماشینی انتحاب می شود که بیشترین انرژی لازم را داشته و در دسترس نیز باشد.

for (VmStateHistoryEntry entry : vm.getStateHistory()) {

if (previousIsInMigration == true && entry.isInMigration() == false) {

timeBeforeVmMigration.add(entry.getTime() - lastTimeMigrationFinished);

}

if (previousIsInMigration == false && entry.isInMigration() == true) {

lastTimeMigrationFinished = entry.getTime();

}

previousIsInMigration = entry.isInMigration();

}

}

return timeBeforeVmMigration;

}

این تابع کلیه نتایج و عملیاتی را که در هنگام مهاجرت مجازی، تخصیص منابع و کلیه پردازش های انجام شده را نمایش و چاپ می کند.

public static void printResults(PowerDatacenter datacenter,List<Vm> vms,double lastClock,String experimentName,boolean outputInCsv,String outputFolder)

{

Log.enable();

List<Host> hosts = datacenter.getHostList();

int numberOfHosts = hosts.size();

int numberOfVms = vms.size();

در این قسمت زیر معیار های انرژی و زمان مهاجرت و سایر موارد دریافت، محاسبه و چاپ می شود.

double totalSimulationTime = lastClock;

double energy = datacenter.getPower() / (3600 \* 1000);

int numberOfMigrations = datacenter.getMigrationCount();

Map<String, Double> slaMetrics = getSlaMetrics(vms);

در این قسمت که حدود 15 خط است کلا زمانهای مربوط به فعال و غیره فعال شدن یک ماشین مجازی در میزبان، یک میزبان در دیتاسنتر ودر کل یک دیتاسنتر است.

double slaOverall = slaMetrics.get("overall");

double slaAverage = slaMetrics.get("average");

double slaDegradationDueToMigration = slaMetrics.get ("underallocated\_migration");

double slaTimePerActiveHost = getSlaTimePerActiveHost(hosts);

double sla = slaTimePerActiveHost \* slaDegradationDueToMigration;

List<Double> timeBeforeHostShutdown = getTimesBeforeHostShutdown(hosts);

int numberOfHostShutdowns = timeBeforeHostShutdown.size();

double meanTimeBeforeHostShutdown = Double.NaN;

double stDevTimeBeforeHostShutdown = Double.NaN;

if (!timeBeforeHostShutdown.isEmpty()) {

meanTimeBeforeHostShutdown = MathUtil.mean(timeBeforeHostShutdown);

stDevTimeBeforeHostShutdown = MathUtil.stDev(timeBeforeHostShutdown);

}

در این قسمت کلیه زمان ها بررسی می شوند و در صورتی که زمانی مقدار دهی نشده است یا اینکه زمان شروع ماشین مجازی یا سروری مقدار دهی نشده است آن را مقدار دهی می کنند.

List<Double> timeBeforeVmMigration = getTimesBeforeVmMigration(vms);

double meanTimeBeforeVmMigration = Double.NaN;

double stDevTimeBeforeVmMigration = Double.NaN;

if (!timeBeforeVmMigration.isEmpty()) {

meanTimeBeforeVmMigration = MathUtil.mean(timeBeforeVmMigration);

stDevTimeBeforeVmMigration = MathUtil.stDev(timeBeforeVmMigration);

}

if (outputInCsv) {

File folder = new File(outputFolder);

if (!folder.exists()) {

folder.mkdir();

}

File folder1 = new File(outputFolder + "/stats");

if (!folder1.exists()) {

folder1.mkdir();

}

File folder2 = new File(outputFolder + "/time\_before\_host\_shutdown");

if (!folder2.exists()) {

folder2.mkdir();

}

File folder3 = new File(outputFolder + "/time\_before\_vm\_migration");

if (!folder3.exists()) {

folder3.mkdir();

}

File folder4 = new File(outputFolder + "/metrics");

if (!folder4.exists()) {

folder4.mkdir();

}

StringBuilder data = new StringBuilder();

String delimeter = ",";

در این قسمت کلیه زمانهای قبل و بعد از عملیات مهاجرت مقدار دهی، محاسبه و نمایش داده می شوند.

data.append(experimentName).append(delimeter);

data.append(parseExperimentName(experimentName));

data.append(String.format("%d", numberOfHosts)).append(delimeter);

StringBuilder append = data.append(String.format("%d", numberOfVms)).append(delimeter);

data.append(String.format("%.2f", totalSimulationTime)).append(delimeter);

StringBuilder append1 = data.append(String.format("%.5f", energy)).append(delimeter);

data.append(String.format("%d", numberOfMigrations)).append(delimeter);

data.append(String.format("%.10f", sla)).append(delimeter);

StringBuilder append2 = data.append(String.format("%.10f", slaTimePerActiveHost)).append(delimeter);

StringBuilder append3 = data.append(String.format("%.10f", slaDegradationDueToMigration)).append(delimeter);

StringBuilder append4 = data.append(String.format("%.10f", slaOverall)).append(delimeter);

StringBuilder append5 = data.append(String.format("%.10f", slaAverage)).append(delimeter);

// data.append(String.format("%.5f", slaTimePerVmWithMigration) + delimeter);

// data.append(String.format("%.5f", slaTimePerVmWithoutMigration) + delimeter);

// data.append(String.format("%.5f", slaTimePerHost) + delimeter);

data.append(String.format("%d", numberOfHostShutdowns) + delimeter);

data.append(String.format("%.2f", meanTimeBeforeHostShutdown) + delimeter);

data.append(String.format("%.2f", stDevTimeBeforeHostShutdown) + delimeter);

data.append(String.format("%.2f", meanTimeBeforeVmMigration) + delimeter);

data.append(String.format("%.2f", stDevTimeBeforeVmMigration) + delimeter);

در این قسمت نیز سیاست های مهاجرت و تخصیص منابع برای کلیه ماشین های مجازی مقدار دهی می شود.

if (datacenter.getVmAllocationPolicy() instanceof PowerVmAllocationPolicyMigrationAbstract) {

PowerVmAllocationPolicyMigrationAbstract vmAllocationPolicy = (PowerVmAllocationPolicyMigrationAbstract) datacenter

.getVmAllocationPolicy();

زمانهای اجرا در این قسمت مدیریت می شود.

double executionTimeVmSelectionMean = MathUtil.mean(vmAllocationPolicy.getExecutionTimeHistoryVmSelection());

double executionTimeVmSelectionStDev = MathUtil.stDev(vmAllocationPolicy.getExecutionTimeHistoryVmSelection());

double executionTimeHostSelectionMean = MathUtil.mean(vmAllocationPolicy.getExecutionTimeHistoryHostSelection());

double executionTimeHostSelectionStDev = MathUtil.stDev(vmAllocationPolicy.getExecutionTimeHistoryHostSelection());

double executionTimeVmReallocationMean = MathUtil.mean(vmAllocationPolicy.getExecutionTimeHistoryVmReallocation());

double executionTimeVmReallocationStDev = MathUtil.stDev(vmAllocationPolicy.getExecutionTimeHistoryVmReallocation());

double executionTimeTotalMean = MathUtil.mean(vmAllocationPolicy.getExecutionTimeHistoryTotal());

double executionTimeTotalStDev = MathUtil.stDev(vmAllocationPolicy.getExecutionTimeHistoryTotal());

data.append(String.format("%.5f", executionTimeVmSelectionMean)).append(delimeter);

data.append(String.format("%.5f", executionTimeVmSelectionStDev)).append(delimeter);

StringBuilder append6 = data.append(String.format("%.5f", executionTimeHostSelectionMean)).append(delimeter);

StringBuilder append7 = data.append(String.format("%.5f", executionTimeHostSelectionStDev)).append(delimeter);

StringBuilder append8 = data.append(String.format("%.5f", executionTimeVmReallocationMean)).append(delimeter);

data.append(String.format("%.5f", executionTimeVmReallocationStDev)).append(delimeter);

data.append(String.format("%.5f", executionTimeTotalMean)).append(delimeter);

data.append(String.format("%.5f", executionTimeTotalStDev)).append(delimeter);

writeMetricHistory(hosts, vmAllocationPolicy, outputFolder + "/metrics/" + experimentName

+ "\_metric");

}

data.append("\n");

writeDataRow(data.toString(), outputFolder + "/stats/" + experimentName + "\_stats.csv");

writeDataColumn(timeBeforeHostShutdown, outputFolder + "/time\_before\_host\_shutdown/"

+ experimentName + "\_time\_before\_host\_shutdown.csv");

writeDataColumn(timeBeforeVmMigration, outputFolder + "/time\_before\_vm\_migration/"

+ experimentName + "\_time\_before\_vm\_migration.csv");

} else {

Log.setDisabled(false);

Log.printLine();

Log.printLine(String.format("Experiment name: " + experimentName));

Log.printLine(String.format("Number of hosts: " + numberOfHosts));

Log.printLine(String.format("Number of VMs: " + numberOfVms));

Log.printLine(String.format("Total simulation time: %.2f sec", totalSimulationTime));

Log.printLine(String.format("Energy consumption: %.2f kWh", energy));

Log.printLine(String.format("Number of VM migrations: %d", numberOfMigrations));

Log.printLine(String.format("SLA: %.5f%%", sla \* 100));

Log.printLine(String.format(

"SLA perf degradation due to migration: %.2f%%",

slaDegradationDueToMigration \* 100));

Log.printLine(String.format("SLA time per active host: %.2f%%", slaTimePerActiveHost \* 100));

Log.printLine(String.format("Overall SLA violation: %.2f%%", slaOverall \* 100));

Log.printLine(String.format("Average SLA violation: %.2f%%", slaAverage \* 100));

Log.printLine(String.format("Number of host shutdowns: %d", numberOfHostShutdowns));

Log.printLine(String.format(

"Mean time before a host shutdown: %.2f sec",

meanTimeBeforeHostShutdown));

Log.printLine(String.format(

"StDev time before a host shutdown: %.2f sec",

stDevTimeBeforeHostShutdown));

Log.printLine(String.format(

"Mean time before a VM migration: %.2f sec",

meanTimeBeforeVmMigration));

Log.printLine(String.format(

"StDev time before a VM migration: %.2f sec",

stDevTimeBeforeVmMigration));

if (datacenter.getVmAllocationPolicy() instanceof PowerVmAllocationPolicyMigrationAbstract) {

PowerVmAllocationPolicyMigrationAbstract vmAllocationPolicy = (PowerVmAllocationPolicyMigrationAbstract) datacenter

.getVmAllocationPolicy();

double executionTimeVmSelectionMean = MathUtil.mean(vmAllocationPolicy

.getExecutionTimeHistoryVmSelection());

double executionTimeVmSelectionStDev = MathUtil.stDev(vmAllocationPolicy

.getExecutionTimeHistoryVmSelection());

double executionTimeHostSelectionMean = MathUtil.mean(vmAllocationPolicy

.getExecutionTimeHistoryHostSelection());

double executionTimeHostSelectionStDev = MathUtil.stDev(vmAllocationPolicy

.getExecutionTimeHistoryHostSelection());

double executionTimeVmReallocationMean = MathUtil.mean(vmAllocationPolicy

.getExecutionTimeHistoryVmReallocation());

double executionTimeVmReallocationStDev = MathUtil.stDev(vmAllocationPolicy

.getExecutionTimeHistoryVmReallocation());

double executionTimeTotalMean = MathUtil.mean(vmAllocationPolicy

.getExecutionTimeHistoryTotal());

double executionTimeTotalStDev = MathUtil.stDev(vmAllocationPolicy

.getExecutionTimeHistoryTotal());

Log.printLine(String.format(

"Execution time - VM selection mean: %.5f sec",

executionTimeVmSelectionMean));

Log.printLine(String.format(

"Execution time - VM selection stDev: %.5f sec",

executionTimeVmSelectionStDev));

Log.printLine(String.format(

"Execution time - host selection mean: %.5f sec",

executionTimeHostSelectionMean));

Log.printLine(String.format(

"Execution time - host selection stDev: %.5f sec",

executionTimeHostSelectionStDev));

Log.printLine(String.format(

"Execution time - VM reallocation mean: %.5f sec",

executionTimeVmReallocationMean));

Log.printLine(String.format(

"Execution time - VM reallocation stDev: %.5f sec",

executionTimeVmReallocationStDev));

Log.printLine(String.format("Execution time - total mean: %.5f sec", executionTimeTotalMean));

Log.printLine(String

.format("Execution time - total stDev: %.5f sec", executionTimeTotalStDev));

}

Log.printLine();

}

Log.setDisabled(true);

}

/\*\*

\* Parses the experiment name.

\*

\* @param name the name

\* @return the string

\*/

این تابع خروجی ها را در واقع دریافت نموده و در یک فایل ذخیره سازی میکند. زیاد مورد استفاده قرار نمیگیرد این تابع.

public static String parseExperimentName(String name) {

Scanner scanner = new Scanner(name);

StringBuilder csvName = new StringBuilder();

scanner.useDelimiter("\_");

for (int i = 0; i < 4; i++) {

if (scanner.hasNext()) {

StringBuilder append = csvName.append(scanner.next()).append(",");

} else {

csvName.append(",");

}

}

scanner.close();

return csvName.toString();

}

این تابع زمان فعالیت های میزبان ها در مراکز داده را محاسبه نموده و بر میگرداند.

protected static double getSlaTimePerActiveHost(List<Host> hosts) {

double slaViolationTimePerHost = 0;

double totalTime = 0;

for (Host \_host : hosts) {

HostDynamicWorkload host = (HostDynamicWorkload) \_host;

double previousTime = -1;

double previousAllocated = 0;

double previousRequested = 0;

boolean previousIsActive = true;

for (HostStateHistoryEntry entry : host.getStateHistory()) {

if (previousTime != -1 && previousIsActive) {

double timeDiff = entry.getTime() - previousTime;

totalTime += timeDiff;

if (previousAllocated < previousRequested) {

slaViolationTimePerHost += timeDiff;

}

}

previousAllocated = entry.getAllocatedMips();

previousRequested = entry.getRequestedMips();

previousTime = entry.getTime();

previousIsActive = entry.isActive();

}

}

return slaViolationTimePerHost / totalTime;

}

این تابع نیز زمان های اجرای کلیه میزبان ها اعم از هاست های فعال و غیره فعال دریافت می گردد.

protected static double getSlaTimePerHost(List<Host> hosts) {

double slaViolationTimePerHost = 0;

double totalTime = 0;

for (Host \_host : hosts) {

HostDynamicWorkload host = (HostDynamicWorkload) \_host;

double previousTime = -1;

double previousAllocated = 0;

double previousRequested = 0;

for (HostStateHistoryEntry entry : host.getStateHistory()) {

if (previousTime != -1) {

double timeDiff = entry.getTime() - previousTime;

totalTime += timeDiff;

if (previousAllocated < previousRequested) {

slaViolationTimePerHost += timeDiff;

}

}

previousAllocated = entry.getAllocatedMips();

previousRequested = entry.getRequestedMips();

previousTime = entry.getTime();

}

}

return slaViolationTimePerHost / totalTime;

}

این تابع خروجی های تولید شده را به صورت ستونی چاپ می کند. ورودی های این تابع نیز مسیر ذخیره سازی و داده های دریافتی است.

public static void writeDataColumn(List<? extends Number> data, String outputPath) {

File file = new File(outputPath);

try {

file.createNewFile();

} catch (IOException e1) {

e1.printStackTrace();

System.exit(0);

}

try {

BufferedWriter writer = new BufferedWriter(new FileWriter(file));

for (Number value : data) {

writer.write(value.toString() + "\n");

}

writer.close();

} catch (IOException e) {

e.printStackTrace();

System.exit(0);

}

}

این تابع خروجی های تولید شده را به صورت سطری چاپ می کند. ورودی های این تابع نیز مسیر ذخیره سازی و داده های دریافتی است.

public static void writeDataRow(String data, String outputPath) {

File file = new File(outputPath);

try {

file.createNewFile();

} catch (IOException e1) {

e1.printStackTrace();

System.exit(0);

}

try {

BufferedWriter writer = new BufferedWriter(new FileWriter(file));

writer.write(data);

writer.close();

} catch (IOException e) {

e.printStackTrace();

System.exit(0);

}

}

این تابع نیز معیار های مربوطه را در خروجی چاپ می کند.

public static void writeMetricHistory(

List<? extends Host> hosts,

PowerVmAllocationPolicyMigrationAbstract vmAllocationPolicy,

String outputPath) {

// for (Host host : hosts) {

for (int j = 0; j < 10; j++) {

Host host = hosts.get(j);

if (!vmAllocationPolicy.getTimeHistory().containsKey(host.getId())) {

continue;

}

File file = new File(outputPath + "\_" + host.getId() + ".csv");

try {

file.createNewFile();

} catch (IOException e1) {

e1.printStackTrace();

System.exit(0);

}

try {

BufferedWriter writer = new BufferedWriter(new FileWriter(file));

List<Double> timeData = vmAllocationPolicy.getTimeHistory().get(host.getId());

List<Double> utilizationData = vmAllocationPolicy.getUtilizationHistory().get(host.getId());

List<Double> metricData = vmAllocationPolicy.getMetricHistory().get(host.getId());

for (int i = 0; i < timeData.size(); i++) {

writer.write(String.format(

"%.2f,%.2f,%.2f\n",

timeData.get(i),

utilizationData.get(i),

metricData.get(i)));

}

writer.close();

} catch (IOException e) {

e.printStackTrace();

System.exit(0);

}

}

}

این تابع در واقع نتایج نهایی شبیه سازی را مدل و در خروجی چاپ می کند.

public static void printCloudletList(List<Cloudlet> list) {

int size = list.size();

Cloudlet cloudlet;

String indent = "\t";

Log.printLine();

Log.printLine("========== resource management with load balancing in Datacenters OUTPUT ==========");

Log.printLine("Cloudlet ID" + indent+ indent + "STATUS" + indent + "Resource ID" + indent + "VM ID" + indent

+ "Time" + indent + "Start Time Sim" + indent + "Finish Time");

DecimalFormat dft = new DecimalFormat("###.##");

for (int i = 0; i < size; i++) {

cloudlet = list.get(i);

Log.print(indent + cloudlet.getCloudletId());

if (cloudlet.getCloudletStatus() == Cloudlet.SUCCESS) {

Log.printLine(indent + "RM: " + indent + indent

+ cloudlet.getResourceId() + indent

+ cloudlet.getVmId() + indent + dft.format(cloudlet.getActualCPUTime()) + indent

+ dft.format(cloudlet.getExecStartTime()) + indent + indent

+ dft.format(cloudlet.getFinishTime()));

}

}

Log.printLine(" Resource Management In Cloud Is Finished ...");

}

5-2نتیجه گیری:

بنابراین با اجرای شبیه سازی فوق می توان به این نتیجه رسید که استفاده از لاگ ها در انتقالات و مهاجرت های ماشین های مجازی ضمن اینکه با سرعت بیشتری انجام می شود، با ایجاد اختلال در هر ماشین مجازی، سایر ماشین های موجود در هاست می توانند جای گزین شده و بدون اختلال عملیات مهاجرت را انجام دهند.

منابع :

[1]A.Atrey, N. Jain , Iyengar ,"A Study on Green Cloud Computing" International Journal of Grid and Distributed Computing,vol. 6,p. 93-102, (2013).

[2]A.Beloglazov, J. Abawajy , R. Buyya ,"Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing" vol. 28,p.755–768, (2012).

[3]A.Beloglazov, R. Buyya ,"Adaptive Threshold-Based Approach for Energy Efcient Consolidation of Virtual Machines in Cloud Data Centers" ,p.1-6, (2010).

[4]A.Beloglazov, R. Buyya ,"Energy Efficient Resource Management in Virtualized Cloud Data Centers",p.826-831, (2010).

[5]A.Beloglazov, R. Buyya ,"Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Center",vol. 24,p.1397–1420,(2012).

[6]A.Beloglazov, R. Buyya, Y. C. Lee , A. Zomaya ,"A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems" ,vol. 82,p.1-50, (2011).

[7]R.Buyya, A. Beloglazov,"OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds" vol.27(5),p.1310–1333, (2015).

[8]R.Buyya, A. Beloglazov , J. Abawajy ,"Energy-efficient management of data center resources for Cloud computing" ,in Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA),p. 1–12, (2010).

[9]R.N.Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose , R. Buyya ,"CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Software:Practice and Experience,vol. 41,p.23-50, (2011).

[10]M.Cardosa, M. Korupolu ,A. Singh ,"Shares and utilities based power consolidation in virtualized server environments" ,in Proceedings of the 11th IFIP/IEEE International Symposium on Integrated Network Management (IM), p.327-334, (2009).

[11]P.company, www.gartner.com, (2015).[12]S.E.Dashti, A. M. Rahmani ,"Dynamic VMs placement for energy efficiency by PSO in cloud computing",Experimental & Theoretical Artificial,Intelligence,p.1-16, (2015).

[13]H.Duan, Q. Luo, Y. Shi , G. Ma ,"Hybrid Particle Swarm Optimization and Genetic Algorithm for MultiUAV Formation Reconfiguration" , IEEE Computational Intelligence Society,vol. 8(3),p.16 – 27, (2013).

[14]X.FU, C. ZHOU ,"Virtual machine selection and placement for dynamic consolidation in Cloud computingenvironment" ,vol. 9(2),p.322-330, (2015).

[15]A.Gandhi, M. Harchol-Balter, R. Das , C. Lefurgy ,"Optimal power allocation in server farms" ,in Proceedings of the 11th Joint International Conference on Measurement .and Modeling of Computer Systems (SIGMETRICS/Performance),p. 157–168, (2009).

[16]Y.Gaoa, H. Guana, Z. Qia, Y. Houb ,L. Liu ,"A multi-objective ant colony system algorithm for virtual machine placement in cloud computing", Journal of Computer and System Sciences ,vol.79(8),p.1230–1242,(2013).

[17]S.K.Garg, R. Buyya ,"Green Cloud computing and Environmental Sustainability",p. 1-27, (2012).

[18]D.Gmach, J. Rolia, L. Cherkasova ,A. Kemper ,"Resource pool management: Reactive versus proactive or let’s be friends" ,p.2905–2922, (2009).

[19]I.Goiri, J. L. Berral, J. O. Fitó, F. Julià, R. Nou, J. Guitart, R. Gavaldà , J. Torres ,"Energy-efficient and multifaceted resource management for profit-driven virtualized data centers",vol. 28,p.718–731, (2012).

[20]T.Heath, B. Diniz, E. V. Carrera, W. Meira , R. Bianchini ,"Energy conservation in heterogeneous server clusters" ,PPoPP '05 Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming,p. 186-195, (2005).

[21]A.Khosravi, S. K. Garg, R. Buyya ,"Energy and Carbon-Efficient Placement of Virtual Machines in Distributed Cloud Data Centers" ,p.317-328, (2013).

[22]K.H.Kim, A. Beloglazov , R. Buyya ," Power-aware provisioning of Cloud resources for real-time services",in Proceedings of the 7th International Workshop on Middleware for Grids,p. 1–6, (2009).

[23]M.R.V.Kumar, S. Raghunathan ,"Heterogeneity and thermal aware adaptive heuristics for energy efficientconsolidation of virtual machines in Infrastructure clouds", Journal of Computer and System Sciences ,vol.82(2),p. 1-30, (2015).

[24]D.Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy , G. Jiang ,"Power and performance management ofvirtualized computing environments via lookahead control" ,Cluster Computing,vol. 12,p.1–15, (2009).

[25]B.Li, J. Li, J. Huai, T. Wo, Q. Li , L. Zhong ,"EnaCloud: An Energy-saving Application Live Placement Approach for Cloud Computing Environments " ,IEEE,p. 17 – 24, (2009).

[26]C.Lin, "A novel Green Cloud Computing FrameWork For Improving System Efficiency" ,p.2326-2333, (2012).

[27]W.Lin, J. Z. Wang, C. Liang ,D. Qi ,"A Threshold-based Dynamic Resource Allocation Scheme for Cloud Computing" ,vol. 23,p.695–703, (2011).

[28]T.Mastelic, A. Oleksiak, H. Claussen, I. Brandic , J.-M. Pierson ,"Cloud Computing: Survey on Energy Efficiency" Journal ACM Computing Surveys (CSUR),vol. 47(2),p. 33, (2015).

[29]K.Maurya, R. Sinha ,"Energy Conscious Dynamic Provisioning of Virtual Machines using Adaptive Migiration Thresholds in Cloud Data Center",p. 74-82, (2013).

[30]K.maurya, R. sinha ,"Energy Conscious Dynamic Provisioning Of Virtual Machines Using Adaptive Migration Thresholds in Cloud Data center " ,vol. 2(3),p. 74-82, (2013).

[31]Ms.V.Srimathi, Ms.D.Hemalatha, Mr.R.Balachander ,"Green Cloud Environmental Infrastructure", Engineering And Computer Science ,vol.1(3),p. 168-177, (2012).

[32]R.Nathuji, C. Isci , E. Gorbatov, "Exploiting platform heterogeneity for power efficient data centers", in Proceedings of the 4th International Conference on Autonomic Computing (ICAC),p. 5-10,(2007). [33]R.Nathuji, K. Schwan ,"VirtualPower: Coordinated power management in virtualized enterprise systems" ,ACM SIGOPS Operating Systems Review,vol. 41,p. 265–278, (2007).

[34]A. Murtazaev,S.Oh,"Sercon: Server Consolidation Algorithm using Live Migration of Virtual Machines for Green Computing " ,IETE TECHNICAL REVIEW ,vol. 28, p.1-20, (2012).

[35]N.Quang-Hung, P. D. Nien, N. H. Nam, N. H. Tuong , N. Thoai ,"A Genetic Algorithm for Power-Aware Virtual Machine Allocation in Private Cloud", Springer Berlin Heidelberg ,vol.7804, p. 183-191, (2013). [36]R.Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, X. Zhu ,"No “Power” Struggles: Coordinated Multilevel Power Management for the Data Center" ,SIGARCH Computer Architecture News , vol.36, p.48-59, (2008).

[37]S.Rathore, "Efficient Allocation of Virtual Machine in Cloud Computing Environment" , p.59-62, (2012). [38]S.R.Suraj , R.Natchadalingam ,"Adaptive Genetic Algorithm for Efficient Resource Management in Cloud Computing" ,International Journal of Emerging Technology and Advanced Engineering , vol.4(2), p.350-356, (2014).

[39]K.Sammy, R. Shengbing , C. Wilson ,"Energy Efficient Security Preserving VM Live Migration In Data Centers For Cloud Computing", vol.9, (2012).

[40]J.Sekhar, G. Jeba ,"Energy Efficient VM Live Migration in Cloud Data Centers." IJCSN International Journal of Computer Science and Network, vol.2(2),p. 71-75, (2013).

[41]Y.Song, H. Wang ,B. Feng , Y. Li and Y. Sun ,"Multi-tiered on-demand resource scheduling for VM-based data center", in Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid ,p:148-155, (CCGrid 2009).

[42]S.Srikantaiah, A. Kansal , F. Zhao ,"Energy aware consolidation for Cloud computing" ,in Proceedings of the 2008 USENIX Workshop on Power Aware Computing and Systems (HotPower), p. 1-5, (2008). [43]M.Stillwell, D. Schanzenbach , F. Vivien , H. Casanova , "Resource allocation using virtual clusters" ,in Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, p.260–267, (2009). [44]A.Verma , P. Ahuja ,A. Neogi , "pMapper: power and migration cost aware application placement in virtualized systems", n Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, p. 243-264, (2008).

[45]W.Shi, B.Hong , "Towards Profitable Virtual Machine Placement in the Data Center", p. 138-145, (2011). [46]Sh.Wang , Z. Liu, Z. Zheng, Q. Sun , F. Yang , "Particle Swarm Optimization for Energy-AwareVirtual Machine Placement Optimization in Virtualized Data Centers", 19th IEEE International Conference on Parallel and Distributed Systems,p. 102 - 109, (2013). [47]www.cloudbus.org/papers/Cloud-EnvSustainability, "Green Cloud computing and Environmental Sustainability., (2011).

[48] V.Suresh Kumar, M. Aramudhan” Trust Based Resource Selection in Cloud Computing Using Hybrid Algorithm”, I.J. Intelligent Systems and Applications, 2015, p.59-64 .Published Online July 2015 in MECS DOI: 10.5815/ijisa.2015.08.08

[49] Yongqiang Gao , Haibing Guan , Zhengwei Qi , Yang Hou, Liang Liu,” A multi-objective ant colony system algorithm for virtual machine placement in cloud computing”, 2013, <http://dx.doi.org/10.1016/j.jcss.2013.02.004,p.1-13>

[50] Akhil Goyal, Navdeep S. Chahal,” A Proposed Approach for Efficient Energy Utilization in Cloud Data Center”, International Journal of Computer Applications (0975 – 8887) Volume 115 – No. 11, April 2015,p.24- 27

[51] Elina Pacini, Cristian Mateos, Carlos Garc´ıa Garino,” Dynamic Scheduling based on Particle Swarm Optimization for Cloud-based Scientific Experiments”, CLEI ELECTRONIC JOURNAL, VOLUME 14, NUMBER 1, PAPER 2, APRIL 2014.

[52] Gao, Y., Guan, H., Qi, Z., Hou, Y., & Liu, L. (2013). A multi-objective ant colony system algorithm for 85 virtual machine placement in cloud computing. Journal of Computer and System Sciences, 79, 1230–1242. doi:10.1016/j.jcss.2013.02.004

**Abstract**

he upward growth of knowledge and communication has led to the emergence of a new computing style called cloud computing. One of the biggest advantages of cloud infrastructure suppliers is the financial discussion with minimizing costs, guaranteeing the level of the service contract and maximum profitability. In this regard, energy management in cloud data centers is an important and significant issue to achieve this goal. One way to reduce energy consumption is to free unemployed hosts and the other way is to reduce the migration of virtual machines. To this end, one of the challenges is choosing the method of placing the migrated virtual machines on the appropriate host. In this booklet, by introducing the migration reduction algorithm and the placement algorithm based on the appropriate threshold, the energy consumption in cloud data centers is reduced. This algorithm inherits from the algorithm of the first selection and descending order of hosts and virtual machines based on their rank, which is obtained based on workload, performs the placement operation. The solution provided in CloudSim software is simulated. The simulation results show a decrease in the number of virtual machine migrations, an increase in migration productivity, and a reduction in energy consumption.

**Keywords**

Green cloud computing, reducing data center power consumption, host aggregation, reducing migration

****

Payam Noor University

Faculty of Engineering

**Seminar Report**

**Department of Computer Engineering and Information Technology**

**Title**

Reduce energy consumption (green cloud) in cloud data centers with the help of meta-heuristic algorithms and adjust the reduction of virtual machine migration

**Seyed Ali Mohtarami**

**Supervisor:**

**Dr. Seyed Ali Razavi Ebrahimi**

February 2021

1. Cloud computing [↑](#footnote-ref-1)
2. National Institute of Standarads and Technology(NIST) [↑](#footnote-ref-2)
3. Scalability [↑](#footnote-ref-3)
4. Grid Computing [↑](#footnote-ref-4)
5. Cloud sim [↑](#footnote-ref-5)
6. Cluster Computing [↑](#footnote-ref-6)
7. Grid Computing [↑](#footnote-ref-7)
8. Grid Computing [↑](#footnote-ref-8)
9. Virtualization [↑](#footnote-ref-9)
10. Virtual Machines [↑](#footnote-ref-10)
11. Hypervisor [↑](#footnote-ref-11)
12. Grid Computing [↑](#footnote-ref-12)
13. Public Cloud [↑](#footnote-ref-13)
14. Hybrid Cloud [↑](#footnote-ref-14)
15. Infrastructure as a Service(Iaas) [↑](#footnote-ref-15)
16. Platform as a Service(Paas) [↑](#footnote-ref-16)
17. Software as a Service(Saas) [↑](#footnote-ref-17)
18. Cloud Data Centers (CDC) [↑](#footnote-ref-18)
19. Green cloud computing [↑](#footnote-ref-19)
20. Dynamic Power Management(DPM) [↑](#footnote-ref-20)
21. Static Power Management (SPM) [↑](#footnote-ref-21)
22. Dynamic Voltage and Frequency Scaling (DVFS) [↑](#footnote-ref-22)
23. Dynamic Component Deactivation (DCD) [↑](#footnote-ref-23)
24. Dynamic Performance Scaling (DPS) [↑](#footnote-ref-24)
25. Dynamic Frequency Scaling (DFS) [↑](#footnote-ref-25)
26. Dynamic Voltage and Frequency Scaling (DVFS) [↑](#footnote-ref-26)
27. Guest [↑](#footnote-ref-27)
28. Limited Lookahead Control (LLC) [↑](#footnote-ref-28)
29. Bin Packing Algorithm [↑](#footnote-ref-29)
30. Broker [↑](#footnote-ref-30)
31. Deadlin [↑](#footnote-ref-31)
32. Highest Potential Growth (HPG) [↑](#footnote-ref-32)