www.ElitesJournal.ir

مجله نخبگان علوم و مهندسی

Journal of Science and Engineering Elites

ISSN 2538-581X

جلد ۲- شماره ۳- سال ۱۳۹۶

روشی ترکیبی برای رتبه‌بندی سرویس‌های محاسبات ابری

آرزو جهانی[1*]

۱- دانشکده مهندسی برق و کامپیوتر، دانشگاه تبریز

[*]a.jahani@tabrizu.ac.ir

ارسال: تیر ماه ۹۶ پذیرش: مرداد ماه ۹۶

چکیده

رایانش ابری نوعی مدل محاسباتی بر مبنای اینترنت است. این مدل محاسباتی با ارائه مدل پرداخت به ازای مصرف تأمین‌کنندگان سرویس را قادر ساخته تا سرویس‌های گوناگونی را با ویژگی‌های کیفیت مختلف ارائه نمایند. کاربران این گونه سرویس‌ها با داشتن کاربردهای متفاوت در پی یافتن سرویسی متناسب با نیاز خود هستند که امروزه به عنوان چالشی مطرح است. رتبه‌بندی، سرویس‌های متفاوت ارائه شده توسط تهیه‌کنندگانِ گوناگون را، به منظور انتخاب مناسب‌ترین سرویس، ارزیابی می‌کند. این مقاله در راستای کمک به کاربر در جهت انتخاب مناسب سرویس، روش CCSR را به عنوان روشی ترکیبی برای رتبه‌بندی سرویس‌ها ارائه می‌کند. ترکیبی بودن روش به این دلیل است که راه‌کار ارائه شده بر خلاف راه‌کارهای قبلی در راستای رتبه‌بندی سرویس‌ها از هر دو اطلاعات درونی و برونی در طی رتبه‌بندی استفاده می‌کند. همچنین با انتخاب تعدادی سرویس نامزد و رتبه‌بندی آن با، از رتبه‌بندی تمام سرویس‌ها خودداری می‌کند. راه‌کار پیشنهادی با روش‌های رتبه‌بندی مبتنی بر AHP و SVD مقایسه گردیده است و نتایج نشان می‌دهد که راه‌کار ارائه شده به دلیل انتخاب تعدادی سرویس نامزد و رتبه‌بندی آن‌با، دارای پیچیدگی کمتر، زمان پاسخ کمتر و انعطاف‌پذیری بیشتر نسبت به سایر روش‌ها است.

کلمات کلیدی: تهیه‌کنندگان سرویس، رتبه‌بندی، کاربرد، محاسبات ابری، ویژگی‌های کیفیت.

۱. مقدمه

نیازمندی‌های متفاوت کاربران همزمان با پیشرفت و افزایش تعداد فناوری‌های مبتنی بر اینترنت، اصلی‌ترین چالش کاربران در انتخاب سرویس مناسب برای سپردن کاربرد است. یکی از فناوری‌های نوین برای ارائه‌ی خدمات در سطح اینترنت، فناوری رایانش ابری است. رایانش ابری، مدل محاسباتی بر پایه‌ی اینترنت است که الگویی تازه برای عرضه، مصرف و تحویل خدمات رایانشی را (شامل زیرساخت به عنوان سرویس، نرم‌افزار به عنوان سرویس، بستر به عنوان سرویس) با به‌کارگیری شبکه‌ی اینترنت ارائه می‌کند. به گونه‌ای که این دسترسی بتواند با کمترین نیاز به مدیریت منابع و یا نیاز به دخالت مستقیم فراهم‌کننده سرویس به سرعت فراهم شده یا آزاد (رها) گردد [۱]. تکامل این مدل محاسباتی به حدی بوده است که می‌توان آن را پس از چهار مؤلفه‌ی، آب، برق، گاز و تلفن به عنوان مؤلفه‌ی بعدی دانست که همواره مورد نیاز انسان‌ها می‌باشد [۲].

عموماً مصرف‌کننده‌های رایانش ابری مالک زیرساخت فیزیکی ابر نیستند، بلکه برای اجتناب از هزینه‌های سرمایه‌ای، آن را از عرضه‌کنندگان شخص ثالث اجاره می‌کنند. آن با منابع را در قالب سرویس مصرف می‌کنند و تنها بهای منابعی را که به کار

می‌برند، می‌پردازند. بنابراین تعداد سرویس‌های ابری ارائه شده در رایانش ابری متفاوت بوده و این سرویس‌ها دارای ویژگی‌های کیفیت (QoS) متفاوتی هستند [۳], [۴]. کاربران با داشتن کاربردهای متفاوت در انتخاب مناسب‌ترین سرویس ابری بر اساس نیازمندی‌های خود دچار مشکل می‌شوند. پس، انتخاب مناسب‌ترین سرویس بر اساس نیازمندی‌های کاربردِ کاربر، به عنوان یک مشکل تحقیقاتی است و غالباً موفقیت زیرساختِ کاربرد، توسط مصرف آن سرویس تعیین می‌شود. به گونه‌ای که انتخاب نادرست سرویس منجر به استفاده نکردن از تمام قدرت تهیه‌کنندگان می‌شود. به منظور کمک به کاربر در جهت انتخاب مناسب‌ترین سرویس، سیستم‌های رتبه‌بندی به وجود آمده‌اند [۵]. این سیستم‌ها، با ارزیابی [۶] و مقایسه سرویس‌های ابری، مناسب‌ترین سرویس‌ها را یافته و به کاربر نمایش می‌دهند. مطمئناً با وجود تعداد بسیار بالای سرویس‌های ابری، انتخاب صرفاً یک سرویس امکان‌پذیر نخواهد بود. اما می‌توان با اختصاص رتبه به سرویس‌ها، کاربر را از رتبه‌ی سرویس‌دهنده‌ها در قبال کاربردش مطلع ساخت.

در طی مقایسه‌ی سرویس‌ها، اطلاعات کیفیت سرویس مورد نیاز است. این اطلاعات می‌تواند توسط خود تهیه‌کنندگان و یا توسط بخش سومی [۷] اندازه‌گیری شود. البته برای تهیه این اطلاعات، یک سری ویژگی‌های کیفیت تعریف شده است [۸]. مثل؛ زمان پاسخ، تأخیر و دسترس‌پذیری. مقدار این ویژگی‌ها، درجه‌ی کیفی سرویس‌ها را نمایش می‌دهند [۹]. از طرف دیگر ممکن است کاربر دارای دو نوع نیازمندی‌های کیفیت باشد: ضروری و غیرضروری. نیازمندی‌های ضروری، ویژگی‌هایی هستند که از بالاترین درجه اهمیت از نظر کاربر برخوردارند و لزوماً باید توسط سرویس‌ها ارائه شوند. نیازمندی‌های غیرضروری، ویژگی‌هایی هستند که اهمیت زیادی ندارند و اگر توسط سرویس‌ها، ارائه نشوند، مشکلی وجود نخواهد داشت [۱۰].

در این راستا و جهت حل چالش انتخاب سرویس، مقاله‌ی حاضر روشی CCSR (Combination Cloud Service Rank) را برای رتبه‌بندی سرویس‌های ابری بر اساس مقادیر کیفیت مورد نیاز کاربر ارائه کرده است. این روش برای رتبه‌بندی سرویس‌های ابری، ابتدا تعدادی سرویس نامزد انتخاب می‌کند تا از رتبه‌بندی تمامی سرویس‌ها خودداری کند. زیرا با رتبه‌بندی تمام سرویس‌ها، مطمئناً سرویس‌های با رتبه‌ی بیشتر نمی‌توانند اکثر نیازمندی‌های کاربر را ارضا نمایند و بهتر است در رتبه‌بندی دخالت داده نشوند. از طرف دیگر، این روش می‌تواند در طی رتبه‌بندی از هر دو نیازمندی‌های ضروری و غیرضروری کاربر بهره بگیرد و همچنین هر دو اطلاعات درونی و برونی را در رتبه‌بندی دخالت می‌دهد. اطلاعات برونی شامل مقادیر کیفیت هر سرویس می‌باشد که توسط ابزارهای دیده‌بانی [۱۱] اندازه گرفته می‌شوند. در مقابل اطلاعات درونی شامل اطلاعات به دست آمده از کاربران غیرفعال یا کاربران قبلی سیستم است که در گذشته از سیستم کمک گرفته و رتبه‌بندی اعلام شده از سیستم را مورد استفاده قرار داده‌اند و در پایان در صورت تمایل برای سرویسِ مورد استفاده امتیاز وارد نموده‌اند تا میزان رضایت خود را از سرویس بیان کنند. روش پیشنهادی با سه روش پیشین مقایسه شده و نتایج آزمایش‌ها کاهش پیچیدگی الگوریتم، کاهش زمان پاسخ سیستم و همچنین انعطاف‌پذیری بالای CCSR را در مقایسه با روش‌های قبلی نشان می‌دهد.

ادامه مقاله به این صورت سازمان‌دهی شده است؛ بخش دوم به بررسی راه‌کارهای پیشین می‌پردازد. بخش سوم مدل‌سازی‌های استفاده شده را به تفصیل بیان می‌کند. بخش چهارم شامل راه‌کار پیشنهادی است و در ادامه بخش پنجم نتایج آزمایش‌ها را بیان می‌کنند. در پایان بخش ششم نتیجه‌گیری و کارهای آتی را نشان می‌دهد.

۲. راه‌کارهای پیشین

در جهت حل چالش کمک به کاربر برای انتخاب صحیح سرویس‌دهنده، تحقیقات زیادی انجام گرفته است. برخی از محققان، این چالش را با انتخاب سرویس [۱۲]، برخی با ترکیب سرویس [۱۳] و برخی دیگر با رتبه‌بندی و مقایسه‌ی سرویس‌ها [۱۴] حل کرده‌اند. تمرکز اصلی مقاله‌ی حاضر بر روی رتبه‌بندی و مقایسه‌ی سرویس‌ها بر اساس مقادیر کیفیت سرویس‌ها می‌باشد که

آن هم شامل دو بخش می‌باشد: راه‌کارهای ارزیابی و مقایسه‌ی سرویس‌ها و راه‌کارهای رتبه‌بندی. مقاله‌ی حاضر از اطلاعات به دست آمده از بخش اول استفاده کرده و راه‌کار جدیدی را در بخش رتبه‌بندی ارائه می‌دهد. راه‌کارهای ارزیابی و مقایسه‌ی سرویس‌ها، بدون دریافت نیازمندی‌های کاربر، صرفاً به مقایسه‌ی سرویس‌ها بر اساس مقادیر کیفیت می‌پردازند. ابزارهای دیده‌بانی و ابزارهای تست، عمده راه‌کارهای موجود در این بخش هستند. این ابزارها، می‌توانند از طریق اندازه‌گیری مقادیر کیفیت سرویس‌ها در طی دیده‌بانی ویژگی‌های متفاوت سرویس‌ها، به مقایسه‌ی سرویس‌ها بپردازند. در حالت کلی، این ابزارها مقادیر کیفیت سرویس‌ها را محاسبه و اندازه‌گیری می‌نمایند و غالباً نتایج این ابزارها در قالب مجموعه داده‌های واقعی در دسترس عموم قرار می‌گیرد. سیستم رتبه‌بندی دانشگاه‌با، مثال بارزی است که با محاسبه و امتیازدهی دانشگاه‌ها بر اساس ویژگی‌های متفاوت، اقدام به رتبه‌دهی به دانشگاه‌ها می‌نماید. نتایج به دست آمده از بخش ارزیابی سرویس‌ها، در بخش رتبه‌بندی سرویس‌ها مورد استفاده قرار می‌گیرد. زیرا راه‌کارهای رتبه‌بندی با فرض داشتن مقادیر کیفیت سرویس‌ها، با دریافت بازه‌ی نیازمندی‌های کاربر به رتبه‌بندی سرویس‌ها می‌پردازند.

از جمله راه‌کارهای ارزیابی، راه‌کار Cloudstone [۱۵] است که در سال ۲۰۰۸ ارائه گردید. این راه‌کار، یک ابزار اندازه‌گیری کارایی برای سرویس‌های مبتنی بر اینترنت و وب۲ است. این ابزار توانایی اندازه‌گیری کارایی سرویس‌های ابری را بر اساس تعدادِ کاربرانی که همزمان می‌تواند پاسخگو باشد را دارد. پس از یک سال تیم Cloudharmony [۱۶] راه‌کاری باهم‌ین نام را برای ارزیابی سرویس‌های ابری ارائه کردند. این ابزار توانایی سنجش تمامی سرویس‌های موجود را بر اساس تمامی معیارها دارا است و به صورت رایگان در اختیار افراد قرار دارد. پس از آن، اَنگ و همکارانش [۱۷], [۱۸] راه‌کار CloudCmp را ارائه کردند که یک مقایسه کننده سامان‌دهی شده بود و فقط کارایی و هزینه‌ی تهیه‌کنندگان را بررسی می‌کرد. ابزار CloudCmp بر روی چهار تهیه‌کننده‌ی: آمازون، میکروسافت Azure، گوگل AppEngine و Rackspace اجراشده و نتایجِ مقایسه‌ی سرویس‌ها در [۱۷] وجود دارند. مزیت اصلی روش CloudCmp نسبت به روش‌های قبلی که تا سال ۲۰۱۰ ارائه شده بودند این است که هیچ یک از روش‌های قبل، قابل توسعه برای همه‌ی تهیه‌کنندگان نبودند (مثلاً برای تهیه‌کنندگان Paas پاسخگو نبودند).

شرکت Compuware [۱۹] در سال ۲۰۱۱ یک ابزار دیده‌بانی برای سرویس‌های ابری ارائه کرد. این ابزار، یک دید جهانی از تهیه‌کنندگان ارائه می‌کند که قابلیت اطمینان و سازگاری سرویس‌های ابری را نشان می‌دهد و برای دو نوع سرویس پلتفرم به عنوان سرویس و زیرساخت به عنوان سرویس به‌کار برده می‌شوند. این پروژه‌ی بزرگ به صورت رایگان بر روی اینترنت برای استفاده‌ی کاربران قرار داده شده است. ابزار دیده‌بانی CloudSleuth [۱۹] برای اندازه‌گیریِ کارایی، دو معیار، زمان پاسخ و دسترس‌پذیری را در نظر می‌گیرد و سرویس‌های موجود در ۵۰ کشور و ۷۵ ایالت را شامل می‌شود. کاربران می‌تواند دو معیار زمان پاسخ و دسترس‌پذیری را در هر سرویس اندازه‌گیری کرده و نتایج را مورد مقایسه قرار دهند. در سال ۲۰۱۲ [۲۰] اولین ابزار محک برای ارزیابی و مقایسه‌ی سیستم‌های محاسبات ابری مبتنی بر داده ارائه گردید. تمرکز اصلی این راه‌کار ارائه‌ی تعدادی معیار جدید و آزمونی مناسب برای ارزیابی سیستم‌های ابری است که اجراکننده‌ی کاربرد با داده‌های بزرگ هستند. به گونه‌ای که بتوان کل سیستم را به جای یک بخش یا مؤلفه مورد ارزیابی و آزمون قرار داد. در طی سال‌های ۲۰۱۲ و ۲۰۱۳، رحمان و همکارانش [۲۱] با بیان این موضوع که استفاده از ابزارهای محک به دلیل تزریق بارکاری ساختگی، بی‌فایده به نظر می‌رسد. چارچوبی را برای دیده‌بانی سرویس‌های ابری بر اساس بازخورد کاربران سیستم ارائه نمودند. چارچوب ارائه شده، از بازخورد کاربرانِ قبلی سیستم استفاده می‌نماید تا کارایی هر سرویس را پیشگویی کرده و ذخیره نماید. زیرا کاربرانِ سیستم، به صورت واقعی کاری را به سرویس می‌سپارند و سرویس به صورت واقعی این کار را انجام داده و نتایج را در اختیار کاربر قرار می‌دهد. بنابراین با قرار دادن یک مؤلفه در سمت کاربرِ استفاده‌کننده از سرویس، کارایی واقعی سیستم اندازه گرفته می‌شود.

از جمله راه‌کارهای رتبه‌بندی سرویس‌ها، راه‌کار مبتنی بر SVD [۲۲] است که توسط چان و چیو ارائه گردیده است. این راه‌کار با استفاده از روش تجزیه مقادیر منفرد (SVD) سعی می‌کند تا بهترین تهیه‌کننده‌ی سرویس را برای یک کاربردِ خاص با یک مجموعه از نیازمندی‌ها تعیین نماید. راه‌کار SRS [۲۳] راه‌کاری است که به دو صورت ایستا و پویا رتبه‌بندی سرویس‌ها را انجام می‌دهد. در حالت ایستا، رتبه‌ی تمام تهیه‌کنندگانِ موجود در فهرست سرویس‌ها، بدون توجه به نیاز کاربر مشخص می‌گردد. در حالت پویا، رتبه‌ی سرویس‌ها بر اساس نیازمندی‌های کاربر مشخص گردیده و نمایش داده می‌شود. در ادامه راه‌کار مبتنی بر مطابقت SLA ارائه گردید که بخشی از پروژه‌ی بزرگ[24] cirrocumulus است.

در سال ۲۰۱۲ [۲۵], [۲۶]، راه‌کاری ارائه گردید که برای رتبه‌بندی سرویس‌ها می‌توانست از هر دو اطلاعات بازخورد کاربران قبلی و اطلاعات کیفیت سرویس استفاده نماید. اما این راه‌کار، اطلاعات مورد نیاز کاربران را فقط به صورت اولویت دریافت می‌نمود. راه‌کار Cloudrank برای رتبه‌بندی سرویس‌ها از پیشگویی مقادیر کیفیت سرویس استفاده می‌کند به این صورت که می‌تواند کاربرانِ مشابه را یافته و رتبه‌بندیِ یکسانی را برای کاربرانِ مشابه ارائه دهد [۲۷]. راه‌کار مبتنی بر AHP [۱۴] که در سال ۲۰۱۲ ارائه گردید، از روش تجزیه سلسله‌مراتبی استفاده می‌نماید. این راه‌کار، ضمن ارائه‌ی چارچوب SMIcloud برای رتبه‌بندی سرویس‌ها، توانایی در نظر گرفتن تمام نیازمندی‌های ضروری و غیرضروری کاربر را در طی رتبه‌بندی دارا می‌باشد. راه‌کار دیگری که در سال ۲۰۱۳ [۲۸] ارائه شد می‌توانست رتبه‌بندیِ پویای سرویس‌های ابری را با استفاده از اطلاعات توافق شده بین سرویس‌دهنده و کاربر (SLA) که از تهیه‌کنندگانِ سرویس‌ها دریافت می‌کرد، انجام دهد. معایب کلی راه‌کارها، عدم استفاده از هر دو اطلاعات درونی و برونی، در نظر نگرفتن هر دو نیازمندی‌های ضروری و غیرضروری کاربر، مقیاس‌پذیری پایین، انعطاف‌پذیری پایین و همچنین محدودیت در افزودن ویژگی جدید است.

۳. فرضیات CCSR

این بخش به بررسی فرضیات و مدل‌سازی‌های ابتدایی راه‌کار پیشنهادی اختصاص یافته است. مدل‌سازی بردار کیفیت سرویس، سرویس‌های ابری، کاربران فعال و همچنین کاربران غیرفعال در ادامه به تفصیل بیان می‌گردد.

۱.۳. مدل‌سازی بردار ویژگی‌های کیفیت سرویس

ارزیابی و مقایسه‌ی سرویس‌ها صرفاً توسط مقایسه‌ی اطلاعات کیفیت آن با امکان‌پذیر است. بنابراین اندازه‌گیری ویژگی‌های کیفیت سرویس‌ها امری ضروری می‌باشد. اما ویژگی‌های کیفیت باید ابتدا شناسایی گردند. پس از بررسی‌های متعدد، کنسرسیوم CSMIC [۲۹]، توانست استانداردی را بیان کند که در آن اکثر ویژگی‌های کیفیت پوشش داده شده است. این ویژگی‌ها با نام ویژگی‌های SMI، برای ارزیابی و مقایسه‌ی سرویس‌های ابری ارائه شده است. ویژگی‌های ارائه شده بر اساس استاندارد ISO طراحی گردیده است. ویژگی‌های SMI شامل هفت ویژگی اصلی و زیر ویژگی‌هایی برای هر یک از آن با است. این مقاله برای رتبه‌بندی سرویس‌ها، از این ویژگی‌ها استفاده می‌کند. تمام ویژگی‌های اصلی در جدول (۱)، نشان داده شده است [۱۴]. جدول (۱)، ویژگی‌های کیفیت سرویس و تعریف هر ویژگی را نشان می‌دهد [۳۰].

جدول ۱- مجموعه ویژگی‌های SMI [۳۰]

ردیف	ویژگی کیفیت سرویس	تعریف
۱	قابلیت جوابگویی	بازه‌ی دستیابی به داده‌های کاربر در زمان تقاضای کاربر
۲	چابکی	امکان تغییر و یا گسترش سرویس‌های ابری بدون صرف هیچ هزینه‌ای
۳	قابلیت اطمینان	احتمال کارایی سرویس‌های ابری بر اساس توافق بین کاربر و تهیه‌کننده
۴	هزینه	هزینه‌ی مورد نیاز برای اجرای سرویس (هزینه منابع)
۵	کارایی	تعداد درخواست سرویس ابری پاسخ داده شده در یک بازه زمانی مورد نظر
۶	امنیت	حفاظت از داده
۷	قابلیت استفاده	سازش سریع و یادگیری آسان سرویس‌های ابری برای استفاده کنندگان سرویس

فرض می‌کنیم تعداد کل ویژگی‌های کیفیت (شامل ویژگی‌ها و زیرویژگی‌ها) Q باشد. در بین تمام ویژگی‌های کیفیت، تعدادی از ویژگی‌ها دارای مقادیر مثبت و تعدادی دارای مقادیر منفی هستند. ویژگی‌های دارای مقادیر بهینه مثبت، ویژگی‌هایی مانند مقیاس‌پذیری و یا قابلیت اطمینان هستند که هر چه مقدار آن‌ها بیشتر باشد، ارجحیت بیشتری دارند. در مقابل ویژگی‌های با مقادیر بهینه منفی، مانند زمان پاسخ و خطا هستند که هر چه مقدار آن‌ها کمتر باشد، ارجحیت بیشتری دارند. بنابراین لازم است نوع ویژگی‌ها مشخص گردد. برای انجام این کار، بردار ویژگی‌های کیفیت را با بردار Q نمایش می‌دهیم که مقادیر آن می‌توانند ۱+ و یا ۱- باشند. مقادیر ۱+، نشانگر بهینه مثبت بودن آن ویژگی و در برابر مقدار ۱-، نشانگر بهینه منفی بودن ویژگی است. بنابراین مقادیر این بردار ثابت و از قبل تعیین شده است. این بردار برای هفت ویژگی اصلی در رابطه (۱) نشان داده شده است. (به دلیل امکان مقایسه‌ی راه‌کار پیشنهادی با سایر راه‌کارها، اندازه بردار Q در تمامی آزمایش‌های انجام شده در بخش پنجم برابر با هفت که همان تعداد ویژگی‌های اصلی است، در نظر گرفته شده است.)

$$Q = \{1,1,1,0,1,1,1\} \quad (۱)$$

۲.۳. مدل‌سازی سرویس‌های ابری

همه‌ی سرویس‌های ابریِ موجود در فهرست سیستم، تشکیل مجموعه‌ی سرویس‌ها را می‌دهند. مجموعه‌ی سرویس‌ها در رابطه‌ی (۲) نشان داده شده است. هر یک از سرویس‌های موجود در این رابطه، با مجموعه ویژگی‌های کیفیت آن‌ها نمایش داده شده است که در رابطه (۳) نشان داده شده است.

$$S = \{s_n | 1 \le n \le N\} \quad (۲)$$

$$s_n = \{q_{nj} | 1 \le j \le Q\} \quad (۳)$$

در رابطه (۲)، S نشانگر مجموعه سرویس‌های ابری است که شامل سرویس‌هایی همانند S_n است که n می‌تواند مقداری بین ۱ تا N (تعداد کل سرویس‌ها) باشد. در رابطه (۳)، هر سرویس s_n، با ویژگی‌های کیفیت آن همانند q_{nj} نمایش یافته است که اگر تعداد کل ویژگی‌های کیفیت را (شامل ویژگی‌ها و زیرویژگی‌ها) Q در نظر بگیریم، مقدار j می‌تواند عددی بین ۱ تا Q باشد.

۳.۳. مدل‌سازی کاربران فعال سیستم

به تمامی کاربرانی که در یک زمان واحد، تقاضای رتبه‌بندی را از سیستم داشته باشند، کاربران فعال گفته می‌شود. هر کاربر فعال با مجموعه‌ی نیازمندی‌با، ضرایب یا میزان اهمیت هر نیاز و لیست نیازمندی‌های ضروری و غیرضروری مشخص می‌گردد. هر کاربر فعال با رابطه (۴) نمایش داده شده است. نیازمندی‌های کاربر در رابطه‌ی (۵)، ضرایب ویژگی‌ها در رابطه (۶) و نیازمندی‌های ضروری و غیرضروری به ترتیب با رابطه (۷) و (۸) نمایش داده شده است.

$$U = \{R, C, HR, SR\} \quad (۴)$$

$$R = \{R_j | 1 \le j \le Q\} \quad (۵)$$

$$C = \{C_j | 1 \le j \le Q \ , \ 0 \le C_j \le 1\} \quad (۶)$$

$$HR = \{HR_j | 1 \le j \le Q\} \quad HR_j \in \{0,1\} \quad (۷)$$

$$SR = \overline{HR} = \{SR_j | 1 \le j \le Q\} \ SR_j \in \{0,1\} \quad (۸)$$

در رابطه (۴)، U نشانگر یک کاربر فعال است که با مجموعه نیازمندی‌ها (R)، ضرایب یا میزان اهمیت هر ویژگی (C)، مجموعه نیازمندی‌های ضروری (HR) و مجموعه نیازمندی‌های غیرضروری (SR) نمایش داده شده است.

در رابطه (۵)، نیازمندی‌های کاربر به ازای هر ویژگی کیفیت نشان داده شده است. R_j، بیانگر نیازمندی‌های مربوط به ویژگی کیفیت j است. در واقع این نیازمندی‌با، مقادیری است که کاربر به ازای هر ویژگی وارد می‌کند تا بیان کند به چه میزان از آن

ویژگی نیاز دارد. این مقادیر می‌توانند بسته به نوع ویژگی، به سه صورت: بازه‌ای، دودویی و یا عددی وارد شوند. در نوع بازه‌ای، کاربر نیاز خود را به صورت یک بازه تعریف می‌کند. مثلاً ممکن است کاربر نیاز زمان پاسخ خود را به صورت [۱۰،۲۰] وارد نماید. بدین معنی که هر سرویس دارای ویژگی زمان پاسخ در این بازه، مورد قبول کاربر است.در نوع دودویی، کاربر نیاز خود را به صورت صفر یا یک وارد می‌کند که آیا آن ویژگی را خواستار است (یک) یا خیر (صفر). در نوع عددی، کاربر صرفاً یک عدد را وارد می‌کند که خواستار آن ویژگی فقط با آن مقدار خاص است. این توضیحات به مدل‌سازی جداگانه‌ای نیاز ندارد. زیرا سیستم رتبه‌بندی پیشنهادی پس از دریافت نیاز کاربر، تمام آن‌ها را به صورت بازه در نظر می‌گیرد. مثلاً نوع باینری را به صورت [۰،۰] برای مقدار باینری صفر یا [۱،۰] برای مقدار باینری ۱، در نظر می‌گیرد و نوع عددی را به صورت یک بازه با مقدار [مقدار مورد نظر،۰] در نظر می‌گیرد و تمام این ورودی‌ها، صرفاً به دلیل هدف مقاله یعنی کمک به کاربر و سهولت دریافت اطلاعات است.

در رابطه (۶)، هر C_j نشانگر ضریب وارد شده توسط کاربر برای ویژگی j است. این ضرایب در واقع میزان اهمیت هر ویژگی را از نظر کاربر نشان می‌دهند و دارای مقادیری ما بین صفر و ۱ هستند. اما نیازی نیست مجموع ضرایب برابر با یک باشد.

در رابطه (۷)، هر HR_j می‌تواند دارای مقداری برابر با یک یا صفر باشد. مقدار یک نشان می‌دهد که ویژگی j به عنوان یک ویژگی ضروری تعریف شده است و باید توسط سرویس‌های رتبه‌بندی شده ارضا گردد. در مقابل، مقدار صفر نشان می‌دهد که ویژگی j به عنوان یک ویژگی غیرضروری تعریف شده است.

در رابطه (۸)، هر SR_j، نشانگر ضروری یا غیرضروری بودن ویژگی j است. در واقع این رابطه نقطه مقابل رابطه‌ی (۷) است و با داشتن یکی از این روابط می‌توان دیگری را به دست آورد.

۴.۳. مدل‌سازی کاربران غیرفعال

کاربران غیرفعال (قبلی) سیستم، کاربرانی هستند که قبلاً از سیستم استفاده کرده و برای سرویس‌ها رتبه دریافت نموده‌اند و در پایان، در صورت تمایل میزان رضایت‌مندی خود را از سرویس‌ها و به ازای هر ویژگی کیفی، به صورت اعدادی در بازه‌ی صفر و یک وارد نموده‌اند. مجموعه‌ی کاربران غیرفعال در رابطه (۹) نشان داده شده است. هر کاربر غیرفعال نیز با امتیازاتی به ازای هر ویژگی کیفیت در رابطه (۱۰) نشان داده شده است. نیازمندی‌های وارد شده توسط کاربران غیرفعال نیز در رابطه (۱۱) نشان داده شده است. این نیازمندی‌ها در فاز چهارم بخش ۴، مورد استفاده قرار خواهد گرفت.

$$IU = \{\mathrm{iu}_i \mid 1 \le \mathrm{i} \le \mathrm{I}\} \qquad (۹)$$

$$\mathrm{iu}_\mathrm{i} = \{\mathrm{C}_{ij} \mid 1 \le \mathrm{j} \le \mathrm{Q},\ 0 \le \mathrm{C}_{ij} \le 1\} \qquad (۱۰)$$

$$R_\mathrm{i} = \{\mathrm{R}_{ij} \mid 1 \le \mathrm{j} \le \mathrm{Q}\} \qquad (۱۱)$$

در رابطه (۹)، IU نشانگر لیست کاربران غیرفعال است که شامل iu_i هایی است که بیانگر i امین کاربر غیرفعال است (اگر در زمان پاسخ به کاربر فعال، I کاربر غیرفعال وجود داشته باشد.). زیرا تعداد کاربران غیرفعال با گذر زمان و با افزایش تعداد کاربران فعال سیستم، که میزان رضایت خود را وارد می‌کنند، بیشتر می‌شود.

در رابطه (۱۰) هر iu_i با مجموعه‌ی C_{ij} هایی تشکیل می‌گردد که هر C_{ij} بیانگر ضریب وارد شده توسط کاربر i به ازای ویژگی کیفیت j است. در رابطه (۱۱) R_i نشانگر مجموعه نیازمندی‌های کاربر غیرفعال i است که شامل R_{ij} هایی است که، نیازمندی کاربر غیرفعال i را بر اساس ویژگی کیفیت j نشان می‌دهد. این مقادیر در فاز چهارم الگوریتم به منظور یافتن کاربران مشابه، مورد استفاده قرار خواهند گرفت.

۴. راه‌کار CCSR

راه‌کار CCSR روشی برای رتبه‌بندی سرویس‌های ابری بر اساس مقادیر کیفیت سرویس است. این راه‌کار، دارای پنج فاز مجزا می‌باشد که در زیر آورده شده است:

۱) انتخاب سرویس‌های نامزد
۲) محاسبه رتبه‌ی برونی OA سرویس‌های نامزد
۳) محاسبه رتبه درونی SA سرویس‌های نامزد
۴) محاسبه رتبه‌ی کاربران مشابه به ازای سرویس‌های نامزد
۵) ترکیب نتایج و محاسبه رتبه‌ی عمومی سرویس‌های نامزد

در فاز اول رتبه‌بندی، تعدادی سرویس نامزد انتخاب می‌گردد. سرویس‌های نامزد، سرویس‌هایی هستند که می‌توانند تمام نیازهای ضروری و اکثر نیازمندی‌های غیرضروری کاربران را ارضا نمایند. پس از انتخاب سرویس‌های نامزد، این سرویس‌ها در طی سه فاز بعدی رتبه‌هایی را دریافت می‌کنند. در فاز دوم، سرویس‌ها بر اساس مقادیر ویژگی‌های کیفیت رتبه‌بندی می‌شوند. در فاز سوم، سرویس‌ها بر اساس میزان امتیازات وارد شده توسط کاربران غیرفعال رتبه‌بندی می‌شوند. در فاز چهارم، ابتدا کاربران مشابه یافته می‌شود. سپس سرویس‌های نامزد بر اساس میزان حضورشان در نتایج ارائه شده برای کاربران مشابه رتبه دریافت می‌کنند. در فاز نهایی یا فاز پنجم، تمامی رتبه‌های به دست آمده باهم ترکیب می‌شوند تا رتبه‌ی نهایی سرویس‌های نامزد به دست آید. روش CCSR در شکل (۱)، آورده شده است. در ادامه نحوه محاسبه سرویس‌های نامزد، رتبه‌ها و ترکیب رتبه‌ها با جزئیات بیان خواهد گردید.

الگوریتم CCSR (Combining Cloud Service Rank)

ورودی: مجموعه سرویس‌ها (S)، مجموعه کاربران فعال (U)، مجموعه کاربران غیرفعال (IU) و بردار ویژگی کیفیت
خروجی: رتبه‌بندی ارائه شده برای کاربر فعال R(G)

۱ محاسبه سرویس‌های نامزد
۲ محاسبه رتبه‌ی ارزیابی‌های برونی R(OA) به ازای تمام سرویس‌های نامزد // با استفاده از F(OA)
۳ محاسبه رتبه‌ی ارزیابی‌های درونی R(SA) به ازای تمام سرویس‌های نامزد // با استفاده از F(SA)
۴ محاسبه رتبه‌ی کاربران مشابه R(SU) به ازای تمام سرویس‌های نامزد // با استفاده از F(SU)
۵ ترکیب نتایج مراحل قبل و محاسبه رتبه‌ی عمومی R(G)

شکل ۱- الگوریتم CCSR

۱.۴. انتخاب سرویس‌های نامزد (فاز اول)

سرویس‌ها نامزد، سرویس‌هایی هستند که تمام نیازمندی‌های ضروری و اکثر نیازمندی‌های غیرضروری کاربر را ارضا می‌نمایند. این سرویس‌ها در طی فاز اول روش CCSR انتخاب می‌گردند. بنابراین این فاز به بررسی و ارزیابی کل سرویس‌ها بر اساس نیاز کاربر می‌پردازد تا بتواند به تعداد K سرویس نامزد را انتخاب کند. متغیر K تعداد سرویس‌های نامزد را نشان می‌دهد و توسط طراح سیستم رتبه‌بندی انتخاب می‌گردد. نحوه محاسبه سرویس‌های نامزد در شکل (۲)، نشان داده شده است. مطابق با شکل (۲)، ابتدا متغیر K که توسط طراح سیستم به عنوان تعداد سرویس‌های نامزد انتخاب شده است، با تعداد کل سرویس‌ها مقایسه می‌شود. اگر K بیشتر از تعداد سرویس‌های موجود باشد، تمامی سرویس‌ها به عنوان سرویس نامزد انتخاب می‌گردند و الگوریتم خاتمه می‌یابد.

در غیر این صورت، الگوریتم سعی می‌کند از بین سرویس‌های موجود، سرویس‌هایی که مقدار QoS آن‌ها مطابق با نیازمندی‌های کاربر است را انتخاب کند. این سرویس‌ها به مجموعه سرویس‌های نامزد اضافه می‌گردند. سپس تا زمانی که

تعداد سرویس‌های نامزد کوچک‌تر از متغیر K باشد (سطر۴) و نیازمندی غیرضروری وجود داشته باشد (سطر ۵)، یکی از نیازمندی‌های غیرضروری کاربر با کمترین ضریب یافته شده (سطر ۶) و از میان ویژگی‌های غیرضروری حذف می‌گردد (سطر ۷-۹) و مجدداً تعدادی سرویس نامزد مطابق با نیازمندی کاربر انتخاب می‌گردد.

اگر تعداد سرویس‌های نامزد کوچک‌تر از متغیر K نبود (سطر۱۲)، ابتدا تعداد سرویس‌های نامزد یافته شده بررسی می‌گردد. زیرا ممکن است به دلیل نبود هیچ نیازمندی غیرضروری، هیچ سرویس نامزدی یافته نشده باشد. در این صورت الگوریتم پیامی مبنی بر عدم توانایی در یافتن سرویس‌های متناسب با نیازمندی کاربر نمایش داده و خاتمه می‌یابد. در غیر این صورت، مجموعه سرویس‌های نامزد یافته شده برگردانده شده و الگوریتم وارد فاز بعدی می‌گردد.

محاسبه سرویس‌های نامزد (CS)

ورودی: مجموعه سرویس‌ها (S)، مجموعه کاربران فعال (U)، تعداد سرویس‌های نامزد (K)
خروجی: مجموعه سرویس‌های نامزد (CS)

۱ **اگر** (|S|=<k>) **پس** // اگر تعداد سرویس‌های نامزد یا K بیشتر یا مساوی با تعداد کل سرویس‌ها باشد.
۲ CS=S
در غیر این صورت
۳ سرویس‌هایی را مثل s_n را از مجموعه S انتخاب کن به طوری که $(\min R_j =< q_{nj} =< \max R_j)$ و آن‌ها را در مجموعه سرویس‌های نامزد (CS) قرار بده.
۴ **اگر** (|CS|< K) **پس**
۵ **اگر** (sum(SR > 0) **پس** // اگر هنوز تعدادی نیازمندی غیرضروری وجود دارد.
۶ ایندکس $\min\{C \leq SR^{transpose} \mid C_j \leq SR_j != 0 , 1 \leq j \leq Q\}$ را یافته و درون متغیر Index قرار بده.
۷ C(Index)=∞
۸ SR(Index)=0
۹ R(Index)=[-∞,+∞]
۱۰ برو به مرحله ۳
۱۱ **در غیر این صورت**
۱۲ **اگر** (|CS|==0) **پس //** اگر الگوریتم نتوانست هیچ سرویس نامزدی بیابد.
۱۳ برگردان "هیچ سرویس متناسب با نیاز کاربر یافت نشد."
۱۴ الگوریتم را خاتمه بده.
۱۵ **پایان اگر**
۱۶ **پایان اگر**
۱۷ **پایان اگر**
۱۸ **پایان اگر**
۱۹ مجموعه سرویس‌های نامزد CS را برگردان.

شکل ۲- محاسبه سرویس‌های نامزد

۲.۴. محاسبه رتبه برونی OA (فاز دوم)

این فاز به رتبه‌بندی تمام سرویس‌های نامزد می‌پردازد. این کار را صرفاً بر اساس ویژگی‌های کیفیت سرویس‌ها انجام می‌دهد. در واقع در این فاز تمامی سرویس‌های نامزد بر اساس میزان توانایی آن‌ها در برآورده کردن نیازمندی‌های کاربر رتبه دریافت می‌کنند. محاسبه رتبه به این صورت است که ابتدا تمام سرویس‌های نامزد، داخل یک تابع به نام $F(OA)$ قرار گرفته و مقدار دریافت می‌کنند. سپس این مقادیر به ترتیب نزولی مرتب شده و رتبه دریافت می‌کنند. یعنی هر سرویس با مقدار $F(OA)$ بیشتر دارای رتبه‌ی کمتری خواهد بود. مطابق با فرضیات گفته شده، هر کاربر دارای دو نوع نیازمندی‌های ضروری و غیرضروری است. بنابراین لازم است نوع نیازمندی در تابع $F(OA)$ مد نظر قرار گیرد. این تابع به صورت رابطه (۱۲) خواهد بود. اگر تعداد ویژگی‌های کیفیت سرویس‌ها را Q در نظر بگیریم. رتبه‌ی برونی مطابق با رابطه (۱۲) محاسبه می‌گردد.

مطابق با رابطه (۱۲) به ازای هر ویژگی کیفیت:

اگر ویژگی جزء نیازمندی‌های ضروری کاربر باشد (یعنی HR آن برابر با ۱ بوده و SR آن صفر باشد) مقدار ویژگی سرویس ضرب در مقدار ضریب وارد شده توسط کاربر می‌شود. البته چون تمامی مقادیر کیفیت سرویس‌ها و نیازمندی‌های کاربر به صورت بازه در نظر گرفته شده است (در زیر بخش ۳-۳ بیان گردید)، منظور از مقدار ویژگی سرویس، طول بازه‌ی مربوط به آن ویژگی است.

اگر ویژگی جزء نیازمندی‌های غیرضروری کاربر باشد (یعنی HR آن برابر با صفر بوده و SR آن ۱ باشد) اختلاف مقدار ویژگی از مقدار مورد نیاز کاربر، ضرب در مقدار ضریب وارد شده توسط کاربر می‌شود. هدف از انجام این کار، کاهش انحراف از مقدار مورد نیاز کاربر است و سرویس‌هایی که دارای انحراف کمتری از مقدار مورد نیاز کاربر باشند، مقدار بیشتری رو به خود اختصاص خواهند داد. به دلیل اینکه مقادیر کیفیت و نیازمندی‌ها به صورت بازه در نظر گرفته شده‌اند، محاسبه‌ی اختلاف مقدار ویژگی از مقدار مورد نیاز کاربر، با اختلاف اشتراک بازه‌ها از اجتماع بازه‌های مربوطه محاسبه می‌گردد (به فرض سرویسی دارای مقدار ویژگی [۰٫۸] است و مقدار مورد نیاز کاربر [۰٫۶] است. پس اختلاف آن‌ها برابر با اختلاف طول اشتراک از طول اجتماع این دو بازه خواهد بود. یعنی $|(0{,}8)| - |(0{,}6) \cap (0{,}8)| = |(0{,}8)| - |(0{,}6)| = 8 - 6 = 2$).

مقادیر به دست آمده برای هر ویژگی کیفیت در مقدار موجود در بردار ویژگی Q ضرب می‌شود. بنابراین چون تمام ویژگی‌های دارای مقدار بهینه مثبت، دارای مقدار ۱+ در بردار ویژگی Q هستند، مقدار آن‌ها تفاوتی نخواهد کرد. اما در مقابل ویژگی‌های دارای مقدار بهینه منفی، دارای مقدار ۱- در بردار Q هستند، بنابراین مقدار منفی آن‌ها در معادله دخالت داده می‌شود. بنابراین این امر که مقدار منفی آن‌ها بیشتر مورد قبول است محقق می‌گردد.

$$F(OA) = \sum_{j=1}^{Q} Q_j \left(C_j \times |q_{nj}| \times HR_j + C_j \times \left(|q_{nj} \cup R_j| - |q_{nj} \cap R_j|\right) \times SR_j\right) \quad (۱۲)$$

پس از محاسبه تمامی مقادیر $F(OA)$، آن‌ها را مرتب‌سازی نزولی کرده و اندیس یا مکان قرارگیری آن‌ها در دنباله مرتب شده را درون $R(OA)$ قرار می‌دهیم. برای انجام دادن این کار از الگوریتم مرتب‌سازی انتخابی استفاده می‌کنیم که به ازای هر تعداد اعدادی که باید مرتب شوند (در اینجا این تعداد برابر تعداد سرویس‌های نامزد K است)، $K\text{-}1$ بار آرایه پیمایش شده و در هر پیمایش کوچک‌ترین عنصر پیدا می‌گردد و رتبه‌ی آن در $R(OA)$ ذخیره می‌گردد و مقدار $F(OA)$ آن به بی‌نهایت تغییر داده می‌شود تا پیمایش بعدی انجام گردد.

۳.۴. محاسبه رتبه درونی SA (فاز سوم)

این فاز به رتبه‌بندی تمام سرویس‌های نامزد می‌پردازد. این کار صرفاً بر اساس امتیاز وارد شده توسط کاربران غیرفعال انجام می‌شود. در واقع در این فاز تمامی سرویس‌های نامزد بر اساس میزان توانایی آن‌ها در برآورده کردن نیازمندی‌های کاربران غیرفعال رتبه دریافت می‌کنند. محاسبه رتبه به این صورت است که ابتدا تمام سرویس‌های نامزد، داخل یک تابع به نام F(SA) قرار گرفته و مقدار دریافت می‌کنند. سپس این مقادیر به ترتیب نزولی مرتب شده و رتبه دریافت می‌کنند. یعنی هر سرویس با مقدار F(SA) بیشتر دارای رتبه‌ی کمتری خواهد بود. مطابق با فرضیات گفته شده، هر کاربر دارای دو نوع نیازمندی‌های ضروری و غیرضروری است. بنابراین لازم است نوع نیازمندی در تابع F(SA) مد نظر قرار گیرد. این تابع به صورت رابطه

(۱۳) خواهد بود. اگر تعداد ویژگی‌های کیفیت سرویس‌ها را Q در نظر بگیریم. رتبه‌ی درونی مطابق با رابطه (۱۳) محاسبه می‌گردد.

بنابراین مطابق با رابطه (۱۳) به ازای هر ویژگی کیفیت:

- به ازای تمام ویژگی‌ها، اگر ویژگی جزء نیازمندی‌های ضروری کاربر باشد (یعنی HR آن برابر با ۱ بوده و SR آن صفر باشد) و سپس به ازای تمامی کاربران غیرفعالی که امتیازی را برای سرویس مورد نظر وارد کرده‌اند، مقدار ویژگی سرویس ضرب در مقدار امتیاز وارد شده توسط کاربران غیرفعال گردیده و سپس نتیجه به دست آمده به تعداد کاربران غیرفعال تقسیم می‌گردد.
- به ازای تمام ویژگی‌ها، اگر ویژگی جزء نیازمندی‌های غیرضروری کاربر باشد (یعنی HR آن برابر با صفر بوده و SR آن ۱ باشد) و سپس به ازای تمامی کاربران غیرفعالی که امتیازی را برای سرویس مورد نظر وارد کرده‌اند، اختلاف مقدار ویژگی از مقدار مورد نیاز کاربر، ضرب در مقدار امتیاز وارد شده توسط کاربران غیرفعال گردیده و سپس نتیجه به دست آمده به تعداد کاربران غیرفعال تقسیم می‌گردد. هدف از انجام این کار، کاهش انحراف از مقدار مورد نیاز کاربر است و سرویس‌هایی که دارای انحراف کمتری از مقدار مورد نیاز کاربر باشند، مقدار بیشتری رو به خود اختصاص خواهند داد.

مقادیر به دست آمده برای هر ویژگی کیفیت در مقدار موجود در بردار ویژگی Q ضرب می‌شود. اما چون ویژگی‌های دارای مقدار بهینه منفی، دارای مقدار ۱- در بردار Q هستند، بنابراین مقدار منفی آنها در معادله دخالت داده می‌شود.

$$F(SA) = \sum_{j=1}^{Q}(Q_j \sum_{j=1}^{T}(C_{ij} \times |q_{nj}| \times HR_j + C_{ij} \times (|q_{nj} \cup R_j| - |q_{nj} \cap R_j|) \times SR_j)/T) \quad (۱۳)$$

پس از محاسبه تمامی مقادیر $F(SA)$، آن‌ها را مرتب‌سازی نزولی کرده و اندیس یا مکان قرارگیری آن‌ها در دنباله مرتب شده را درون $R(SA)$ قرار می‌دهیم. همانند حالت قبل، برای انجام دادن این کار از الگوریتم مرتب‌سازی انتخابی استفاده می‌کنیم.

۴.۴. محاسبه رتبه کاربران مشابه (SU) (فاز چهارم)

در این فاز، به ازای هر کاربر فعال موجود در سیستم، چند کاربر مشابه از بین کاربران غیرفعال انتخاب می‌گردد. با توجه به اینکه هر کاربر فعال یا غیرفعال در هنگام ورود به سیستم، نیازمندی‌های کیفیت کاربرد خود را به عنوان ورودی وارد می‌کند. پس می‌توان کاربران مشابه را با استفاده از این نیازمندی‌های وارد شده تعیین کرد. به این ترتیب که نیازمندی‌های هر کاربر فعال، با نیازمندی‌های کاربران غیرفعال مقایسه می‌شود و اگر تمامی نیازمندی‌ها مشابه هم بودند، کاربر غیرفعال به عنوان کاربر مشابه انتخاب می‌شود. این فرآیند تا زمان انتخاب تعداد مشخصی کاربر مشابه (که توسط طراح سیستم تعیین می‌گردد و تعداد آن برابر با D است.) ادامه می‌یابد و در هر تکرار، یکی از نیازمندی‌های غیرضروری با کمترین میزان اهمیت نادیده گرفته می‌شود.

این فرآیند دقیقاً همان فرآیند انجام شده در فاز اول برای تعیین سرویس‌های نامزد می‌باشد. بنابراین می‌توان همان شکل (۲) را با اندکی تغییرات در نام متغیرها به کار برد. الگوریتم ارائه شده در شکل (۳) نمایش داده شده است.

شکل (۳)، کاربران مشابه با هر کاربر فعال را تعیین می‌کند. سپس باید، رتبه‌ی هر سرویس نامزد بر اساس تعداد ظاهر شدن آن در نتایج ارائه شده برای کاربران مشابه تعیین گردد (البته فرض بر این است که مقادیر وارد شده توسط کاربران غیرفعال فیلتر شده هستند و هیچ کاربری نتوانسته امتیاز نامعقولی را به سرویسی اختصاص دهد.). فرض کنید به ازای یک کاربر فعال، سه کاربر مشابه همانند جدول (۲)، تعیین گردیده است و سرویس‌های نامزد آن کاربر همانند جدول (۳) باشد. رتبه‌ی سرویس‌های

نامزد بر اساس کاربران مشابه، همانند شکل (۴)، محاسبه خواهد گردید. به این صورت که برای هر سرویس نـامزد، تـک تـک کاربران مشابه بررسی می‌گردند. اگر آن سرویس نامزد در بین رتبه‌های داده شده برای کاربر مشابه وجود داشته باشـد، آنگـاه، مقداری برابر با (۱+ چندمین رتبه – تعداد کل کاربران مشابه) به رتبه‌ی اعطا شده به سرویس نامزد اضافه می‌گردد. در غیر ایـن صورت، اگر آن سرویس نامزد در بین رتبه‌های داده شده برای کاربر مشابه وجود نداشته باشد، مقداری برابر با صفر بـه رتبـه‌ی اعطا شده به سرویس نامزد اضافه می‌گردد. این کار تا جایی که تمامی سرویس‌های نامزد رتبه دریافت نمایند ادامه می‌یابد. این فرآیند در شکل (۵)، نمایش داده شده است.

محاسبه کاربران مشابه (SU)

ورودی: مجموعه کاربران فعال (u)، مجموعه کاربران غیرفعال (IU) و تعداد کاربران مشابه (D)

خروجی: مجموعه کاربران مشابه (SU)

۱ کاربرانی مثل iu_i را از مجموعه IU انتخاب کن به طوری که $(\min R_j =< \min (R_{ij}) =< \max R_j)$ و آن با را در مجموعه کاربران مشابه (SU) قرار بده.
۲ **اگر** $(|SU|< D)$ **پس**
۳ **اگر** $(sum(SR > 0)$ **پس** // اگر هنوز تعدادی نیازمندی غیرضروری وجود دارد.
۴ ایندکس $\{1\leq j\leq Q \mid C_j \leq SR_j \; != 0\}$ را یافته و درون $\min\{C \leq SR^{transpose}$ متغیر Index قرار بده.
۵ $C(Index)=\infty$
۶ $SR(Index)=0$
۷ $R(Index)=[-\infty,+\infty]$
۸ برو به مرحله۱
۹ **در غیر این صورت**
۱۰ **اگر** $(|SU|==0)$ **پس//** اگر الگوریتم نتوانست هیچ کاربر مشابه‌ای بیابد.
۱۱ برگردان "هیچ کاربر مشابه‌ای وجود ندارد."
۱۲ الگوریتم را خاتمه بده.
۱۳ **پایان اگر**
۱۴ **پایان اگر**
۱۵ **پایان اگر**
۱۶ اگر $(|SU| < D)$، مکان‌های خالی آرایه را با صفر پر کن تا $(|SU|==D)$.
۱۷ مجموعه کاربران مشابه SU را برگردان.

شکل ۳- محاسبه کاربران مشابه

جدول ۲- یک نمونه از کاربران مشابه SU یافته شده پس از اجرای الگوریتم(۳)

	رتبه‌بندی ارائه شده به کاربران مشابه		
کاربران مشابه	Rank(1)	Rank(2)	Rank(3)
SU_1	S_{40}	$S1_{20}$	S_{60}
SU_2	S_3	S_{40}	S_{90}
SU_3	S_3	S_{40}	$S1_{20}$

جدول ۳- یک نمونه از سرویس‌های نامزد یافته شده پس از اجرای الگوریتم(۲)

سرویس‌های نامزد		
CS(1)	CS(2)	CS(3)
S_{40}	S_3	S_{62}

F(CS(1)) = (۳-۱+۱)+(۳-۲+۱)+(۳-۲+۱) = ۷			
F(CS(2))= (۰)+(۳-۱+۱)+(۳-۱+۱) = ۶			
F(CS(3)) = (۰)+(۰)+(۰) = ۰			
F(SU)	۷	۶	۰
R(SU)	۱	۲	۳

شکل ۴- نحوه محاسبه سرویس‌های نامزد بر اساس کاربران مشابه

نحوه محاسبه مقادیر R(SU)

ورودی: مجموعه سرویس‌های نامزد یافته شده از الگوریتم (۲)، کاربران مشابه یافته شده از الگوریتم(۳)

خروجی: مقادیر F(SU) به ازای تمام سرویس‌های نامزد

۱ **برای i=1 تا K**
۲ F(i)=0
۳ **برای SU=1 تا D**
۴ **اگر** CS(i) عضو سرویس‌های رتبه‌بندی شده‌ی اعطا شده به کاربر مشابه (R(SU)) بود **پس**
۵ ایندکس R(SU) را بیاب و در Index قرار بده.
۶ F(i)=F(i)+K-index+1
۷ **در غیر این صورت**
۸ F(i)=F(i)+0
۹ **پایان اگر**
۱۰ **پایان**
۱۱ **پایان**
۱۲ اندیس حاصل از مرتب‌سازی نزولی آرایه F(SU) را درون آرایه R(SU) قرار بده.

شکل ۵- نحوه محاسبه R(SU)

مطابق با شکل (۵)، تمام سرویس‌های نامزد، در سطر یازدهم دارای مقدار *F(i)* هستند. همانند فازهای قبل، تمامی *F(i)* ها را مرتب‌سازی نزولی کرده و اندیس یا مکان قرارگیری آن‌ها در دنباله مرتب شده را درون *R(SU)* قرار می‌دهیم. همانند حالت قبل، برای انجام دادن این کار از الگوریتم مرتب‌سازی انتخابی استفاده می‌کنیم.

۵.۴. ترکیب نتایج (فاز پنجم)

پس از اتمام چهار فاز اول الگوریتم CCSR، تمام سرویس‌های نامزد تولید شده در فاز اول، سه نوع رتبه دریافت نموده‌اند. فاز پنجم، وظیفه‌ی ترکیب نتایج به دست آمده از فازهای دوم، سوم و چهارم را بر عهده دارد. فرض کنید سه سرویس نامزد در فاز اول یافته شده باشد. اگر رتبه‌های محاسبه شده در فازهای دوم تا چهارم، مطابق جدول (۴) باشد. نحوه‌ی محاسبه‌ی رتبه‌ی عمومی در فاز پنجم، در شکل (۶) نشان داده شده است.

جدول ۴- یک نمونه رتبه‌های محاسبه شده برای سه سرویس نامزد

	سرویس‌های نامزد		
	CS(1)	CS(2)	CS(3)
فاز دوم R(OA)	۱	۲	۳
فاز سوم R(SA)	۱	۳	۲
فاز چهارم R(SU)	۲	۱	۳

F(CS(1)) = (۳-۱+۱)+(۳-۱+۱)+(۳-۲+۱) = ۸			
F(CS(2))= (۳-۲+۱)+(۳-۳+۱)+(۳-۱+۱) = ۶			
F(CS(3)) = (۳-۳+۱)+(۳-۲+۱)+(۳-۳+۱) = ۴			
F(G)	۸	۶	۴
R(G)	۱	۲	۳

شکل ۶- یک نمونه محاسبه رتبه نهایی برای سه سرویس نامزد

مطابق شکل (۶)، نحوه‌ی محاسبه رتبه نهایی به این صورت است که به ازای هر سرویس نامزد، تمام رتبه‌های به دست آمده در فازهای قبلی باهم ترکیب می‌شوند. مثلاً به ازای سرویس نامزد (۱)، رتبه‌ی نهایی به این صورت محاسبه شده است: سرویس نامزد (۱) در فاز دوم، رتبه‌ی اول را کسب کرده است. بنابراین امتیازی برابر با (۱+۱-۳) که توسط رابطه (۱+ چندمین رتبه – تعداد کل سرویس‌های نامزد) محاسبه می‌گردد را دریافت می‌کند. همین سرویس نامزد، در فاز سوم نیز رتبه‌ی اول را کسب کرده است. بنابراین مجدداً امتیازی را برابر با (۱+۱-۳) دریافت می‌کند. در فاز چهارم، رتبه‌ی دوم را کسب کرده است و امتیازی را برابر با (۱+۲-۳) دریافت می‌کند. مجموع امتیازات سرویس نامزد (۱)، برابر خواهد بود با (۸=۲+۳+۳). امتیازات سایر سرویس‌ها نیز محاسبه می‌گردد. سپس این امتیازات مرتب نزولی شده و رتبه دریافت می‌کنند. شکل (۷) این فرآیند را نشان می‌دهد.

نحوه محاسبه رتبه نهایی سرویس‌های نامزد R(G)

ورودی: مجموعه سرویس‌های نامزد یافته شده از فاز اول (الگوریتم (۲))، رتبه‌های یافته شده از فاز ۱تا۴
خروجی: مقادیر R(G) به ازای تمام سرویس‌های نامزد

```
۱  برای i=1 تا K
۲      F(i)=0
۳      برای j=1 تا 3   // به تعداد فازهایی که برای سرویس‌های نامزد رتبه تعیین کرده‌اند.
۴          Index= R(j,i)   // رتبه‌ی اعطا شده به سرویس نامزد i توسط فاز j را نشان می‌دهد.
۵          F(i)=F(i)+K-index+1
۶      پایان
۷  پایان
۸  رتبه‌ی حاصل از مرتب‌سازی نزولی آرایه F را درون R(G) قرار بده.
```

شکل ۷- نحوه محاسبه رتبه نهایی سرویس‌های نامزد

۶.۴. محاسبه پیچیدگی

این بخش به محاسبه‌ی پیچیدگی الگوریتم ارائه شده تخصیص یافته است. الگوریتم CCSR دارای پنج فاز مجزا می‌باشد. فاز اول مربوط به یافتن سرویس‌های نامزد بوده و زمان‌برترین فاز الگوریتم است. این فاز اهمیت و ضرورت زیادی دارد. زیرا اگر سرویس‌های نامزد به درستی یافته نشوند، فازهای دیگر نمی‌توانند جبرانی برای آن باشند. در این فاز K سرویس نامزد یافته می‌شود که این کار با پیمایش مجموعه کل سرویس‌ها انجام می‌شود. اما مسئله اینجاست که با چند بار پیمایش مجموعه‌ی سرویس‌ها می‌توان K سرویس نامزد را یافت. این مورد کاملاً به تعداد سرویس‌های نامزد و همچنین تعداد ویژگی‌های غیرضروری کاربر بستگی دارد. زیرا تا وقتی که تعداد سرویس‌های یافته شده به K نرسیده، با حذف یک ویژگی غیرضروری، مجموعه سرویس‌ها مجدداً پیمایش می‌گردد. بنابراین پیچیدگی در بدترین حالت عبارت است از O(N|SR|) (N تعداد کل سرویس‌ها و $|SR|$ تعداد نیازمندی‌های غیرضروری کاربر است). اما داشتن تعداد سرویس نامزد بالا غیرمنطقی به نظر می‌رسد. زیرا نمایش هزاران سرویس رتبه‌بندی شده به کاربر با نمایش مجموعه‌ی کل سرویس‌های موجود تفاوتی ندارد. بنابراین مقاله حاضر، طی آزمایشی که در بخش بعدی نتایج آن آورده شده است، تعداد سرویس‌های نامزد را برابر با $K=۱۰۰$، در نظر گرفته است. با این وجود پیچیدگی در بدترین حالت از مجموعه O(N2) خواهد بود.

فاز دوم، فاز یافتن رتبه‌ی برونی (OA) است که محاسبه $F(OA)$ دارای زمان ناچیز و محاسبه $R(OA)$ به دلیل استفاده از

مرتب‌سازی انتخابی دارای زمان $O(K2)$ است. فاز سوم نیز همانند فاز دوم دارای پیچیدگی $O(K2)$ است. فاز چهارم، برای یافتن کاربران مشابه دارای پیچیدگی $O(|IU|)$ است ($|IU|$ تعداد کاربران غیرفعال است). فاز پنجم، به دلیل استفاده از دو حلقه‌ی تودرتو دارای پیچیدگی $O(K2)$ است ($K \leq N$). بنابراین پیچیدگی کلی الگوریتم از مرتبه‌ی $O(N2+3K2+K2)=O(N2)$ است که ($K<N$). این در حالی است که پیچیدگی زمانی راه‌کار مبتنی بر AHP که برای رتبه‌بندی صرفاً از اطلاعات برونی استفاده می‌کرد، در بدترین حالت از مرتبه O(N3) است. بنابراین پیچیدگی زمانی راه‌کار ارائه شده در مقایسه با راه‌کار قبلی کاهش یافته است.

۵. نتایج آزمایش‌ها

آزمایش‌ها و ارزیابی‌های راه‌کار پیشنهادی در سه گروه: آزمایش‌های انعطاف‌پذیری، زمان پاسخ و تعداد سرویس نامزد K انجام شده است. کلیه‌ی آزمایش‌ها بر روی مجموعه داده‌ی واقعی QWS [۳۱] انجام شده است که یک مجموعه داده واقعی، جمع‌آوری شده از بیش از ۲۵۰۰ سرویس وب است. تمامی آزمایش‌ها و همچنین پیاده‌سازی راه‌کارها در محیط نرم‌افزار MATLAB-R 2012a در سیستمی با پردازنده Intel core5Duo 2.53 GHz انجام شده است. نتایج به تفصیل در ادامه آورده شده است.

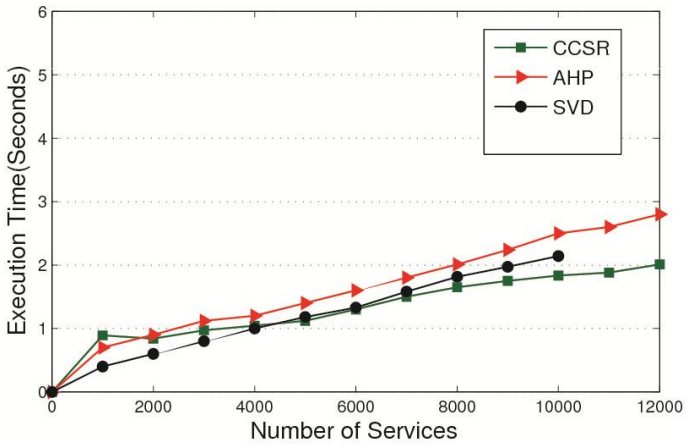
۱.۵. آزمایش انعطاف‌پذیری

این آزمایش مربوط به میزان کار یا پیچیدگی محاسباتی اضافه شده به الگوریتم، به ازای افزوده شدن هر ویژگی کیفیت جدید است. با افزوده شدن هر ویژگی جدید، راه‌کار پیشنهادی، دارای تغییری در فاز دوم و سوم در روابط نوشته شده خواهد بود که هیچ زمان و پیچیدگی محاسباتی بیش‌تری را به سیستم القا نمی‌کند. در واقع راه‌کار پیشنهادی دارای انعطاف‌پذیری افزوده شدن هر تعداد ویژگی جدید است. اما راه‌کار مبتنی بر AHP برای مقایسه‌ی دو سرویس، تمام ویژگی‌ها را در هر سرویس، دو به دو باهم مقایسه می‌کند. یعنی در واقع به ازای هر ویژگی، از ماتریسی با تعداد سطرها و ستون‌های برابر با تعداد سرویس‌ها استفاده می‌کند. بنابراین با افزوده شده هر ویژگی جدید، یک ماتریس جدید نیز به سیستم افزوده می‌گردد که زمان مورد نیاز و پیچیدگی محاسباتی الگوریتم را افزایش می‌دهد. راه‌کار مبتنی بر SVD نیز برای مقایسه‌ی سرویس‌ها از ماتریسی با تعداد سطر برابر با تعداد سرویس‌ها و تعداد ستون برابر با تعداد ویژگی‌ها استفاده می‌کند. بنابراین با افزوده شدن هر ویژگی جدید، یک سطر به ماتریس اضافه می‌گردد که آن هم نیازمند زمان و افزایش پیچیدگی محاسباتی است. بنابراین راه‌کار پیشنهادی دارای انعطاف‌پذیری بیشتری است.

۲.۵. آزمایش زمان پاسخ

این آزمایش بر روی راه‌کار ارائه شده و دو راه‌کار پیشین اجرا شده است. در طی این آزمایش تعداد کاربران یا تعداد درخواست‌ها یک در نظر گرفته شده است. همچنین تعداد سرویس‌ها از ۱ تا ۱۲،۰۰۰ سرویس با گام ۱۰۰۰ تغییر کرده است و تعداد سرویس‌های نامزد برابر با ۱۰۰۰ در نظر گرفته شده است. برای هر مورد از آزمایش، هفت ویژگی اصلی کیفیت به صورت تصادفی تولید شدند. میانگین نتایج ۳۰ بار اجرا برای همه‌ی آزمایش‌ها در شکل (۸) گزارش شده است.

مطابق با شکل (۸)، زمان پاسخ راه‌کار پیشنهادی CCSR در مقایسه با راه‌کارهای پیشین کاهش یافته است. راه‌کار مبتنی بر AHP به دلیل نحوه‌ی مقایسه‌ی سرویس‌ها که در آزمایش انعطاف‌پذیری بیان گردید، زمان بیشتری را برای رتبه‌بندی نیاز دارد. همچنین این راه‌کار بدون داشتن فاز تعیین سرویس‌های نامزد، به رتبه‌بندی تمامی سرویس‌ها می‌پردازد.

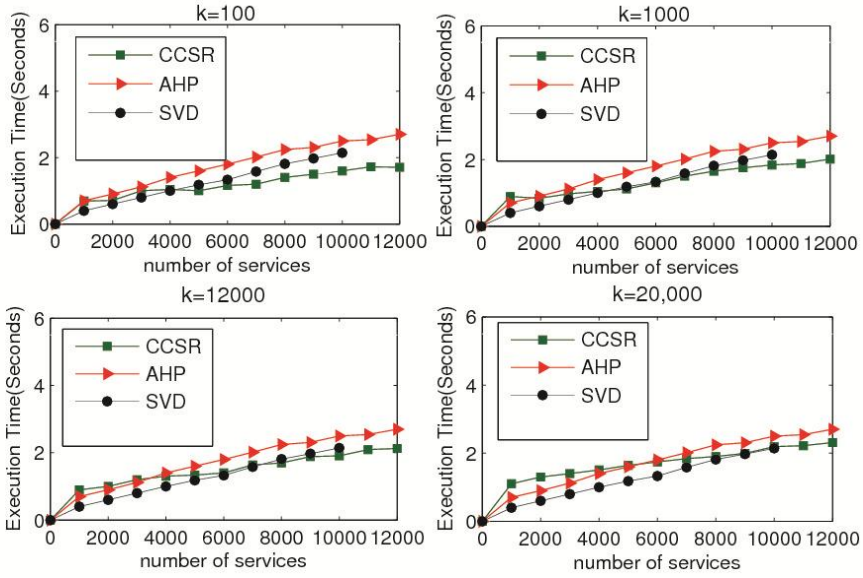


شکل ۸- نتایج آزمایش زمان پاسخ با افزایش تعداد سرویس‌ها

راه‌کار SVD نیز به دلیل استفاده از یک روش ریاضی به نام SVD [۳۲] توانایی پاسخ به بیشتر از ۱۰،۰۰۰ را ندارد. زیرا اعمال روش SVD بر روی ماتریس‌های بزرگ زمان‌بر و حتی غیرممکن است. اگر این آزمایش با تعداد سرویس‌های نامزد کمتری انجام شود، تمامی نتایج کمتر از نتایج کنونی خواهد بود. در ادامه این کار انجام شده است تا تأثیر تعداد سرویس‌های نامزد بررسی گردد.

۳.۵. آزمایش تعداد سرویس‌های نامزد K

در این آزمایش تأثیر تعداد سرویس‌های نامزد با افزایش تعداد سرویس‌ها مورد بررسی قرار گرفته است. در طی آزمایش تعداد سرویس‌ها از ۱ تا ۱۲،۰۰۰ با گام ۱۰۰۰ تغییر یافته است. تعداد کاربران یا کاربردها برابر با یک در نظر گرفته شده بود. تمامی آزمایش‌ها چهار بار در شرایط یکسان و برای چهار مقدار مختلف K برابر با ۱۰۰، ۱۰۰۰، ۱۲،۰۰۰ و ۲۰،۰۰۰ انجام شد. برای هر مورد از آزمایش، هفت ویژگی اصلی کیفیت به صورت تصادفی تولید شده‌اند. میانگین نتایج ۳۰ بار اجرا برای همه‌ی آزمایش‌ها در شکل (۹) گزارش شده است.



شکل ۹- نتایج آزمایش تعداد سرویس‌های نامزد (k) با افزایش تعداد سرویس‌ها

مطابق با نتایج ارائه شده در شکل (۹)، زمان پاسخ در راه‌کار پیشنهادی، با افزایش تعداد سرویس‌ها افزایش می‌یابد. این مورد از قبل هم پیش‌بینی می‌شد. زیرا پیچیدگی زمانی الگوریتم نیز به تعداد سرویس‌های نامزد بستگی داشت. اما این زمان در دو

راه‌کار مبتنی بر AHP و SVD به تعداد سرویس‌های نامزد بستگی ندارد. زیرا این راه‌کارها تمامی سرویس‌ها را رتبه‌بندی می‌کنند. راه‌کار SVD به دلیل استفاده از یک روش ریاضی SVD توانایی پاسخگویی به بیشتر از ۱۰،۰۰۰ سرویس را ندارد. با این وجود زمان پاسخ در راه‌کار پیشنهادی کمتر از سایر راه‌کارها است و این کاهش زمان با تعداد سرویس‌های نامزد K=۱۰۰ بیشتر به چشم می‌خورد. بنابراین این راه‌کار در حضور تعداد سرویس نامزد کمتر بهترین کارایی را دارد و اگر تعداد سرویس‌های نامزد بیشتری مدنظر کاربر یا طراح سیستم قرار دارد، بهتر است در فاز اول تمامی سرویس‌ها را به عنوان سرویس نامزد در نظر بگیرد تا از افزایش زمان پاسخ به دلیل پیمایش مجموعه‌ی سرویس‌ها پیشگیری شود.

۶. نتیجه‌گیری

در این مقاله روشی ساده و کاربردی برای رتبه‌بندی سرویس‌های محاسبات ابری ارائه شد که دارای پیچیدگی زمانی کمتر و زمان پاسخ کمتر و انعطاف‌پذیری بیشتری بود. استفاده از هر دو اطلاعات درونی و برونی در طی رتبه‌بندی یکی از مزایای روش پیشنهادی می‌باشد. همچنین استخراج تعدادی سرویس نامزد به جای رتبه‌بندی تمامی سرویس‌ها نیز جزء مزایای روش پیشنهادی است. به عنوان کارهای آتی می‌توان مسئله‌ی رتبه‌بندی را با در نظر گرفتن جزئیات سرویس‌ها بهبود بخشید. زیرا با مطرح شدن مسئله‌ی ترکیب سرویس‌ها، رتبه‌بندی باید دقیق‌تر و بر اساس جزئیات سرویس‌های ترکیبی صورت گیرد. همچنین در ادامه‌ی این مقاله، ما قصد داریم مسئله‌ی رتبه‌بندی را توسط الگوریتم‌های تکاملی بسط دهیم تا انعطاف‌پذیری روش افزایش یابد. همچنین می‌توان از یادگیری ماشین برای حل این مسئله بهره گرفت.

۷. ضمائم

فهرست علائم		
مخفف	**علائم**	**تعریف**
Service	S	مجموعه سرویس‌های ابری
-	N	تعداد کل سرویس‌ها
Service_n	S_n	سرویس ابری n
quality_{nj}	q_{nj}	ویژگی کیفیت j از سرویس n
User	u	کاربر فعال
Requirement	R	مجموعه نیازمندی‌های کاربر فعال
Cofficient	C	مجموعه ضرایب کاربر فعال
Hard Requirement	HR	مجموعه نیازمندی‌های ضروری کاربر فعال
Soft Requirement	SR	مجموعه نیازمندی‌های ضروری کاربر فعال
Inactive User	IU	مجموعه کاربران غیرفعال
Inactive user_i	iu_i	کاربر غیرفعال i
Cofficient_{ij}	C_{ij}	ضرایب کاربر غیرفعال i برای ویژگی j
Requirement_i	R_i	مجموعه نیازمندی‌های کاربر غیرفعال i
-	K	تعداد سرویس نامزد
-	D	تعداد کاربر مشابه

۸. مراجع

1. R. Buyya, C. Vecchiola and S. T. Selvi, *Cloud Computing Architecture*, Chapter 4, pp. 111-140, 2013.
2. D. Skoutas, D. Sacharidis, A. Simitsis and T. Sellis, *Ranking and Clustering Web Services Using Multicriteria Dominance Relationships*, IEEE Transaction on Service Computing, vol. 3, no. 3, pp. 163-177, 2010.
3. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg and I. Brandic, *Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility*, Future Generation Computer Systems, vol. 25, no. 6, pp. 599-616, 2009.

4. M. J. Katchabaw, H. L. Lutfiyya and M. A. Bauer, *Usage based service differentiation for end-to-end quality of service management*, Computer Communications, vol. 28, no. 18, pp. 2146-2159, 2005.
5. A. Segev and E. Toch, *Context Based Matching and Ranking of Web Services for Composition*, IEEE Transactions on Services Computing, vol. 2, no. 3, pp. 210-222, 2009.
6. C.T. Chen and K. H. Lin, *a Decision Making Method Based on Interval Valued Fuzzy Sets for Cloud Service Evaluation*, 4th International Conference on New Trends in Information Science and Service Science (NISS), pp. 559-564, 2010.
7. M. D. Stojanovic, S. V. Bostjancic Rakas and V. S. Acimovic Raspopovic, E*nd-to-end quality of service specification and mapping: The third party approach*, Computer Communications, vol. 33, no. 11, pp. 1354-1368, 2010.
8. S. Ding, S. Yang, Y. Zhang, C. Liang and C. Xia, *Combining QoS prediction and customer satisfaction estimation to solve cloud service trustworthiness evaluation problems*, Knowledge-Based Systems, vol. 56, pp. 216-225, 2014.
9. V. Raisanen, *Service quality support-an overview*, Computer Communications, vol. 27, no. 15, pp. 1539-1546, 2004.
10. Y. Hao, Y. Zhang and J. Cao, *Web services discovery and rank: An information retrieval approach*, Future Generation Computer Systems, vol. 26, no. 8, pp. 1053-1062, 2010.
11. G. Katsaros, J. Subirats, J. O. Fitó, J. Guitart, P. Gilet and D. Espling, *A service framework for energy aware monitoring and VM management in Clouds*, Future Generation Computer Systems, vol. 29, no. 8, pp. 2077-2091, 2013.
12. Q. He, J. Han, Y. Yang, J. Grundy and H. Jin, *QoS-Driven Service Selection for Multi-tenant SaaS*, IEEE Fifth International Conference on Cloud Computingpp, Honolulu, HI, pp. 566-573, 2012.
13. S. D. Alrifai M., T. Risse, *Selecting Skyline Services for QoS-based Web Service Composition*, In Proceedings of the 19th international conference on World wide web, New York, USA, pp. 11-20, 2010.
14. W. Sobel, Sh. Subramanyam, A. Sucharitakul, J. Nguyen, H. Wong, A. Klepchukov, Sh. Patil, O Fox and D. Patterson, *Cloudstone: Multi Platform, Multi-Language Benchmark and Measurement Tools for Web 2.0*.
15. P. Miller, *The importance of benchmarking clouds: CloudHarmony*, http://www.cloudharmony.com, 2009.
16. A. Li, X. Yang, S. Kandula and M. Zhang, *CloudCmp: Comparing Public Cloud Providers*, IMC '10 Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, pp. 1-14, 2010.
17. A. Li, X. Yang, S. Kandula and M. Zhang, *CloudCmp: Shopping for a CloudMade Easy*, HotCloud'10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, USENIX Association Berkeley, CA, USA, pp. 1-7, 2011.
18. T. Abubakr, *Tools For Benchmarking the Cloud: Cloud Sleuth*, https://www.cloudsleuth.net, 2011.
19. C. Luo, J. Zhan, Z. Jia, L. Wang, G. Lu, L. Zhang, C. Xu and N. Sun, *CloudRank--D: Benchmarking and Ranking Cloud Computing Systems for Data Processing Applications*, Frontiers of Computer Science, vol. 6, no. 4, pp. 347-362 2012
20. u. Rehman, O. K. Hussain, S. Parvin and F. K. Hussain, *A Framework for User Feedback Based Cloud Service Monitoring*, Sixth International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 257-262, 2012.
21. H. Chan and T. Chieu *Ranking and Mapping of Applications to Cloud Computing Services by SVD*, IEEE/IFIP Network Operations and Management Symposium Workshops (NOMS Wksps), Osaka, pp. 362-369, 19-23 April 2010.
22. P. Choudhury, M. Sharma, K. Vikas, T. Pranshu and V. Satyanarayana, *Service Ranking Systems for Cloud Vendors*, Advanced Materials Research, vol. 433-440, pp. 3949-3953, 2012.

23. Wright State University, *Cirrocumulus: A Semantic Framework for Application and core Services Portability Across Heterogeneous Clouds*, project at Kno-e-sis Center at Wright State University, 2010.

24. S. S. Yau and Y. Yin, *QoS-Based Service Ranking and Selection for Service Based Systems*, IEEE International Conference on Services Computing, pp. 56-63, 2011.

25. Z. Zheng, X. Wu,Y. Zhang, M. R. Lyu and J. Wang, *QoS Ranking Prediction for Cloud Services*, IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 6, pp. 1213-1222 2013.

26. M. D. Dikaiakos and D. Zeinalipour Yazti, *A distributed middleware infrastructure for personalized services*, Computer Communications, vol. 27, no. 15, pp. 1464-1480, 2004.

27. S. K. Garg, S. Versteeg and R. Buyya, *a Framework for Ranking of Cloud Computing Services*, Future Generation Computer Systems, vol. 29, no. 4, pp. 1012-1023, 2013.

28. L. Qu, Y. Wang and M. A. Orgun, *Cloud Service Selection Based on the Aggregation of User Feedback and Quantitative Performance Assessment*, IEEE 10th International Conference on Services Computing, pp. 152-159, 2013.

29. CSMIC, *CSMIC SMI Overview Diagram TwoPointOne*, Carnegie Mellon University Silicon Valley, Moffett Field, CA USA, 2011.

30. CSMIC, *Service Measurement Index Version 1.0*, Carnegie Mellon University Silicon Valley, Moffett Field, CA USA, pp. 1-8, September 2011.

31. E. Almasri and Q. H. Mahmoud, *Investigating Web Services on the World Wide Web*, Refereed Track: Web Engineering-Web Service Deployment, pp. 795-804, 2008.

32. W. Ford, *Chapter 15 – The Singular Value Decomposition*, Numerical Linear Algebra with Applications Using MATLAB, 2015.

33. doi:10.1016/B978-0-12-394435-1.0001

۹. واژه نامه

فارسی	انگلیسی
زیرساخت به عنوان سرویس	Infrastructure as a Service (IaaS)
نرم افزار به عنوان سرویس	Software as a Service (SaaS)
پلتفرم به عنوان سرویس	Platform as a Service (PaaS)
کیفیت سرویس	Quality of Service (QoS)
ضروری	Essential
غیر ضروری	Non-essential
سرویس های کاندید	Candidate service
درونی	Subjective
برونی	Objective
دیده‌بانی	Monitoring
ابزار تست	benchmark
مجموعه داده	Data set
تجزیه مقادیر منفرد	Singular Value Decomposition (SVD)
توافق سطح سرویس	Service Level Agreement (SLA)
کنسرسیوم اندازه‌گیری سطح سرویس	Cloud Service Measurement Index Consortium (CSMI)
سطح اندازه‌گیری سرویس	Service Measurement Index (SMI)
سازمان بین المللی	International System Organization (ISO)
قابلیت جوابگویی	Accountability
چابکی	Agility
قابلیت اطمینان	Assurance
هزینه	cost
کارایی	performance
امنیت	Security and privacy
قابلیت استفاده	usability