زمان‌بندی ماژول‌ها در محاسبات مه به روش جستجوی هم‌زیستی جانداران مبتنی بر کوله‌پشتی

دادمهر رهبری و محسن نیک‌رای

چکیده: شبکه‌های حسگر بی‌سیم دارای محدودیت‌هایی از قبیل توان پردازشی، منابع ذخیره‌سازی و تأخیر زمانی در انتقال داده‌ها به ابر می‌باشند. محاسبات مه به وسیله توسعه سرویس‌های ابری به لبه شبکه موجب کاهش ترافیک و تأخیر زمانی می‌شود و بنابراین این نوع شبکه‌ها در سیستم‌های بسیاری مانند مراقبت پزشکی، ابزارهای پوشیدنی، سیستم حمل و نقل و شهرهای هوشمند کاربرد دارد. تکنیک‌های زمان‌بندی وظایف در محاسبات مه از جمله مسایل NP-hard محسوب می‌شود. برنامه‌ها جهت اجراشدن به منابع نیاز دارند. ابزارهای لبه شبکه به حسگرها و ابر نزدیک بوده و دارای قدرت پردازشی لازم برای اجرای برنامه‌ها می‌باشند. هر ابزار لبه می‌تواند برای پیاده‌سازی سیاست‌های تخصیص منابع مورد استفاده قرار گیرد. در این مقاله، ما با ارائه یک روش مبتنی بر کوله‌پشتی بهینه‌شده با الگوریتم هم‌زیستی جانداران به تخصیص مناسب منابع به وظایف در شبکه‌های مه می‌پردازیم. روش پیشنهادی در شبیه‌ساز iFogsim به عنوان یک کتابخانه توسعه‌یافته از کلودسیم جهت پردازش مه پیاده‌سازی شده است. نتایج نشان‌دهنده بهبود در انرژی مصرفی، مصرف منابع و هزینه اجرای شبکه می‌باشد که روش پیشنهادی بهتر از روش کوله‌پشتی و الگوریتم پردازش به ترتیب ورود عمل نموده است.

کلیدواژه: **الگوریتم هم‌زیستی جانداران، زمان‌بندی، کوله‌پشتی، محاسبات مه.**

۱- مقدمه

در جهان امروز، بسیاری از وسایل ارتباطی دارای شبکه‌های حسگر بی‌سیم می‌باشند که با تعداد فراوان در بازه جغرافیایی وسیعی توزیع شده‌اند. شبکه‌های حسگر بی‌سیم، داده‌های پزشکی، حمل و نقل، شهرهای هوشمند و غیره را جمع‌آوری می‌کنند. این شبکه‌ها به عنوان زیرساختی برای اینترنت اشیا، نیازمند پردازش و تصمیم‌گیری بی‌درنگ هستند. در سیستم‌های مبتنی بر ابر، داده‌های جمع‌آوری شده با صرف زمان، هزینه و پردازش زیاد به ابر ارسال می‌شوند. ارسال داده‌های حسگر پایانی به ابر با گذر از تعدادی حسگر میانی، مسیریاب و دروازه خروجی موجب پردازش و تأخیر زمانی زیادی می‌شود [۱]. تأخیر زمانی زیاد در سیستم‌های مراقبت پزشکی می‌تواند منجر به مرگ بیمار گردد و یا در سیستم حمل و نقل منجر به بروز تصادف خواهد شد.

محاسبات مه با قرارگیری مابین ابر و گره‌های حسگر موجب جمع‌آوری، پردازش و ذخیره‌سازی داده‌ها به صورت محلی می‌شود و لذا فقط در مواقع ضروری داده‌ها به ابر ارسال می‌گردد. محاسبات مه با نزدیک‌بودن به ابر و حسگرها، موجب پردازش سریع‌تر و همچنین کاهش ترافیک شبکه می‌شود. محاسبات مه با وجود تعداد بسیار زیادی گره حسگر، انرژی مصرفی کمتری نسبت به سیستم محاسبات ابری متمرکز دارد [۲]. شبکه‌های مه دارای معماری سلسله‌مراتبی از گره‌های حسگر، ابزارهای لبه و ابر هستند. حسگرها در پایین‌ترین سطح از معماری شبکه، داده‌های جمع‌آوری شده را به سطح بالایی ارسال می‌کنند [۳] و گره‌های راه‌انداز نیز به کنترل و تغییر محیط اطراف می‌پردازند. دستورالعمل‌ها در مه شامل ارسال داده‌ها، پردازش در ابزارهای مه، تقسیم‌نمودن یک برنامه به چندین ماژول و تخصیص منابع به همه آنها جهت اجراشدن است. برنامه‌ها برای جمع‌آوری و ذخیره داده‌ها از مراکز داده کوچک استفاده می‌کنند تا این که در آینده مورد پردازش و تحلیل قرار گیرند.

در شبیه‌ساز iFogsim به عنوان یک توسعه از کلودسیم، هر ماژول از برنامه‌ها به عنوان یک گره و ارتباط بین آنها به عنوان لبه‌ای در هم‌بندی شبکه مه در نظر گرفته می‌شود. بنابراین برنامه‌های مختلف در مه دارای هم‌بندی متفاوتی نیز می‌باشند. برنامه‌های مشتری در شبکه مه می‌توانند به عنوان یک کلودلت در ماشین مجازی اجرا شوند که دارای قابلیت‌های انعطاف‌پذیری، تحرک‌پذیری، مقیاس‌پذیری و ارتجاعی می‌باشند. فعالیت کلودلت‌ها در مه شامل محاسبات، فیلتر، برداشتن پارازیت و انتصاب برچسب به داده‌ها است [۴]. ماشین‌های مجازی، منابع موجود در سخت‌افزار را در چندین وضعیت و به صورت اشتراکی فراهم می‌کنند. یکی از سیاست‌های تخصیص ماشین‌های مجازی، افزایش بازدهی منابع داخل مرکز داده است [۵].

فرایند زمان‌بندی برنامه‌ها در محیط‌های توزیع‌شده، شامل انتصاب منابع به وظایف با یک ترتیب مشخص است. استفاده از هر ماشین مجازی، هزینه‌ای را به همراه دارد و همچنین با وجود معیارهای کارایی شامل بهره‌وری، انرژی مصرفی، زمان اجرا و امنیت، پیکربندی این ماشین‌ها، یک چالش محسوب می‌شود [۶]. یکی از راه حل‌های این مسئله، روش کوله‌پشتی است. کوله‌پشتی به عنوان یک روش بهینه‌سازی به دو صورت قراردادی و ۱-۰ می‌باشد. کوله پشتی ۱-۰ در حل مسایل NP-hard کاربرد بیشتری دارد مانند تخصیص بودجه، انتخاب پروژه، تخصیص منابع، سبد سهام، تصمیم‌سازی و غیره [۷]. ما از الگوریتم هم‌زیستی جانداران برای بهینه‌سازی کوله‌پشتی استفاده می‌نماییم تا این که تأخیر زمانی را در تخصیص بهینه منابع موجود در شبکه مه کاهش دهیم.

بر طبق [۸]، الگوریتم هم‌زیستی جانداران برای یافتن مقدار کمترین در بسیاری از توابع با پارامترهای زیاد مورد بررسی قرار گرفته و دارای عملکرد بهتری نسبت به الگوریتم‌های کمترین، ژنتیک، هوش جمعی ذرات، تکامل تفاضلی، کلونی زنبورها و ترکیب هوش جمعی ذرات و زنبورها بوده است.

ما روش پیشنهای خود را با عنوان knapSOS بر روی دو مدل، مورد شبیه‌سازی و تحلیل قرار داده‌ایم. ارزیابی شبیه‌سازی بر اساس جمع‌آوری

این مقاله در تاریخ ۱۶ شهریور ماه ۱۳۹۶ دریافت و در تاریخ ۲ اردیبهشت ماه ۱۳۹۷ بازنگری شد.

دادمهر رهبری، گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه قم، قم، (email: d.rahbari@stu.qom.ac.ir).

محسن نیک‌رای، گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه قم، قم، (email: m.nickray@qom.ac.ir).

داده‌های مرتبط با ردیابی اشیا و انسان‌ها، انتقال داده‌ها به لبه شبکه، پردازش و ارسال داده‌ها به ابر است. نتایج شبیه‌سازی با روش‌های کوله‌پشتی و زمان‌بندی به ترتیب ورود مقایسه شده است.

ادامه مقاله به این صورت می‌باشد. در بخش ۲ کارهای مرتبط قبلی با زمان‌بندی در ابر و مه به صورت خلاصه آورده شده است. شبکه مه با دو مدل از برنامه‌های ردیابی در بخش ۳ ارائه شده است. در بخش ۴ روش پیشنهادی برای زمان‌بندی به صورت دقیق، توضیح داده شده است. جزئیات نتایج و ارزیابی آنها در بخش ۵ آمده و بخش ۶ مربوط به نتیجه‌گیری از این تحقیق و کارهای آینده است.

۲- کارهای مرتبط

هدف از زمان‌بندی، تخصیص بهینه منابع به وظایف می‌باشد. اهداف زمان‌بندی شامل هزینه، زمان اتمام کارها، بیشینه‌سازی تعداد کارهای ورودی، بهره‌وری ماشین مجازی، مصرف انرژی، قابلیت اطمینان و امنیت است. روش‌های بهینه‌سازی شامل الگوریتم‌های اکتشافی، فرااکتشافی [۷] و [۹] و غیره می‌باشند که در ادامه آورده شده است. مدل‌های منبع شامل یک یا چند ماشین مجازی و تأمین‌کننده، اشتراک‌گذاری داده‌های میانی، هزینه انتقال و ذخیره‌سازی داده، ایستایی و پویایی است [۹] و [۱۰]. روش‌های زمان‌بندی دارای پارامترها و الگوریتم‌های مختلفی هستند و ما این روش‌ها را به سه دسته، سنتی، اکتشافی و فرااکتشافی تقسیم می‌نماییم که به صورت ذیل هستند:

در زمان‌بندی جریان‌های کاری علمی دارای محدودیت مهلت زمانی چندسطحی، هر تأمین‌کننده، ماشین مجازی ناهمگن و سرویس ذخیره‌سازی عمومی را پیشنهاد می‌دهد [۱۱]. یکی از روش‌های زمان‌بندی، بررسی زودترین زمان پایان است که در آن پارامترهای هدف شامل هزینه و زمان پایان کارها، بهینه می‌شود. در [۱۲] مؤلفان، زیرساخت‌های تجاری را به عنوان یک سرویس ابری تحلیل نمودند. آنها از اصل پرِتو پیشرو به عنوان ابزاری جهت پشتیبانی از تصمیم جهت مصالحه‌ای بین راه حل‌های مناسب استفاده نمودند و با افزایش زمان پایان کارها به میزان ۵٪ توانستند هزینه زمان‌بندی را به نصف برسانند [۱۲].

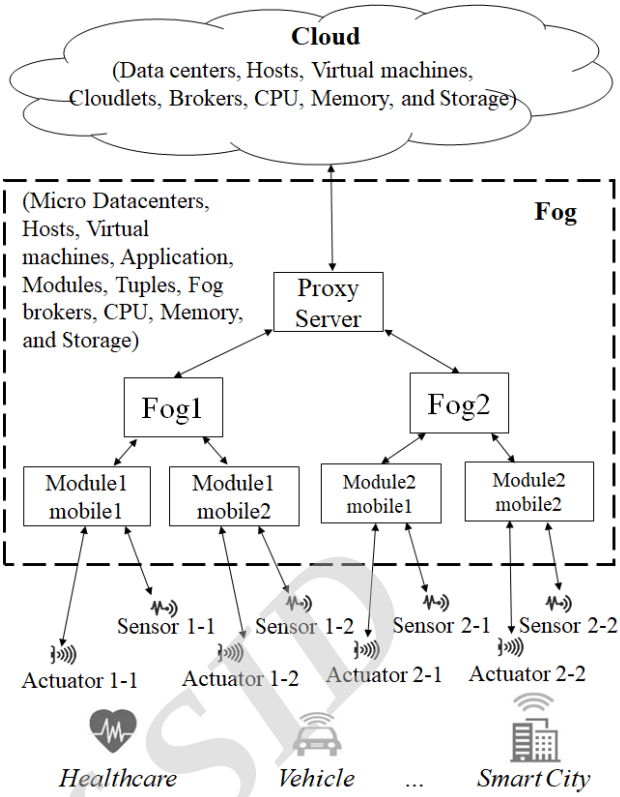
در روشی دیگر، زمان‌بندی تحمل‌پذیر خطا برای تخصیص ماشین‌های مجازی بر اساس دیرترین زمان اجرای وظایف انجام شده است. مشکل این روش، کاهش کارایی به دلیل استفاده از ماشین‌های مجازی ارزان است [۱۳]. در [۱۴] مؤلفان یک زمان‌بند جریان کاری به صورت بی‌درنگ ارائه نمودند که قابلیت تحمل خطاهای سخت‌افزاری را دارد. الگوریتم پیشنهادی آنها دارای سه مرحله از نوع تخصیص ارتجاعی منابع در ابر است. در مرحله اول، عملیات انتقال به عقب (انتقال زمان شروع وظایف بدون تغییر در وضعیت زیرمجموعه‌های آنها) برای آن دسته از وظایف دارای هم‌پوشانی در مهاجرت ماشین‌های مجازی انجام می‌شود. در مرحله دوم به منظور افزایش توانایی در ایجاد و تغییر ماشین مجازی سازگار با وظیفه جدید، عملیات تغییر اندازه منابع انجام می‌شود. در مرحله سوم اگر منبع در یک دوره زمانی مشخص، بیکار باشد آن گاه به چند منبع کوچک‌تر که دارای قابلیت پردازش وظایف باشد تقسیم می‌شود. اجرای این روش نسبت به روش‌های معمول قبلی، دارای بهره‌وری بهینه منابع است.

در [۱۵] مؤلفان به بررسی تأثیر سه روش زمان‌بندی در محیط مه پرداختند. این روش‌ها شامل زمان‌بندی به ترتیب ورود، هم‌زمانی و اولویت بر حسب تأخیر زمانی است. در روش هم‌زمانی، برنامه‌های ورودی بدون توجه به حجم مصرف منابع، اجرا می‌شوند. در روش زمان‌بندی به ترتیب ورود، برنامه‌ها به ترتیب ورود، اجرا می‌شوند و اگر توان پردازشی مرکز داده کمتر از درخواست برنامه باشد آن گاه برنامه در صف زمان‌بندی قرار می‌گیرد. در سومین روش، برنامه‌ها بر اساس کمترین تأخیر زمانی، زمان‌بندی می‌شوند. محققان از کتابخانه iFogsim جهت شبیه‌سازی استفاده نمودند که شامل دو بازی ردیابی سیگنال‌های مغزی و ردیابی سیگنال‌های نوری است. نتایج نشان‌دهنده تأخیر زمانی بیشتر در روش هم‌زمانی نسبت به روش‌های به ترتیب ورود و اولویت تأخیر زمانی است. همچنین تعداد ماژول‌ها در هر وسیله برای روش هم‌زمانی مربوط به بازی ردیابی سیگنال‌های نوری بیشتر از روش‌های دیگر است.

دسته‌ای دیگر از روش‌ها جهت حل مسایل زمان‌بندی، روش‌های مکاشفه‌ای می‌باشند. در روش‌های مکاشفه‌ای، راه حل مسئله به وسیله تعدادی قانون پیدا می‌شود. روش‌های مکاشفه‌ای کلاسیک شامل اولین، بهترین و بدترین شایستگی هستند به طوری که تأمین‌کنندگان ابر و مه می‌توانند به وسیله این روش‌ها، برنامه‌ها و وظایف را در مقیاس بزرگ حتی به صورت آفلود [۹] اجرا نمایند. در [۱۶] وظایف به وسیله الگوریتم مکاشفه‌ای زمان‌بندی شده‌اند به طوری که تابع هدف شامل زمان پایان کارها و هزینه اجرای وظایف است. نتایج نشان‌دهنده افزایش بهره‌وری و کاهش هزینه متوسط می‌باشد. در [۱۷] یک بازی تجمعی مبتنی بر خوشه‌بندی در شبکه‌های با دسترسی رادیویی مه ارائه شده است. در این مدل از معیار هزینه برای واکشی محتوای حافظه نهان استفاده گردیده به طوری که به وسیله یک زمان‌بند تعریف‌شده توسط کاربر توانسته است زمان پاسخ کلی شبکه را بهینه نماید. روشی دیگر برای زمان‌بندی ابر وسایل نقلیه، الگوریتم پویا [۱۸] است که در آن از پارامترهای طول صف و زمان پاسخ استفاده شده است. در این کار، سیستم خدمت‌دهنده به وسیله الگوریتم مارکوف به زمان‌بندی وظایف پرداخته و نرم‌افزار شبیه‌سازی OpenStack و مدل‌سازی آن با شبکه پتری صورت گرفته است. در [۱۹] وظایف بی‌درنگ موازی در شبکه‌های ناهمگن به وسیله روش مکاشفه‌ای زمان‌بندی شده‌اند. مراحل کار شامل انتخاب‌های مکرر و تخصیص بندها (اجزای موازی یک وظیفه) است که با یک برنامه‌نویسی غیر خطی به کاهش انرژی مصرفی در بندهای اجرایی پرداخته است.

در روش فرامکاشفه‌ای، الگوریتم‌های عمومی مختلفی برای حل مسایل بهینه‌سازی طراحی شده‌اند [۶]. یک از این روش‌ها، بهینه‌سازی توده ذرات است که به منظور تأمین منابع و زمان‌بندی، از روش‌های تخصیص ارتجاعی و منابع نامحدود با ماشین‌های مجازی دارای کارایی مختلف استفاده نموده است. در [۲۰] الگوریتم زنبور عسل جهت زمان‌بندی کارها در شبکه مه ارائه شده که توانسته است زمان پاسخ کلی سیستم را بهبود دهد. الگوریتم کلونی مورچگان به عنوان یک روش مکاشفه‌ای جهت محاسبات ابری متحرک با منابع معین استفاده شده است [۲۱]. این الگوریتم، وظایف آفلودشده در ابزارهای مه را با در نظر گرفتن پارامترهای تأخیر زمانی، زمان تکمیل و انرژی مصرفی اجرا می‌کند. مرتبه زمانی این الگوریتم تکاملی به تعداد دورها، وظایف و مورچگان بستگی دارد.

در [۲۲] محققان به منظور زمان‌بندی وظایف در انتقال داده‌های ویدیویی موازی در ابر مبتنی بر کمترین زمان تکمیل به ارائه روش کوله‌پشتی پرداختند. آنها با استفاده از روش بیشترین- کمترین به تخمین کامپیوترهای قدرتمند، بخش‌بندی وظایف، نگاشت تعدادی از بخش‌ها و زمان‌بندی آنها به روش کمترین زمان تکمیل پرداختند. نتایج نشان می‌دهد با در نظر گرفتن معیار تعداد بخش‌ها، روش بیشترین- کمترین بهتر از روش کمترین زمان تکمیل عمل نموده است. در [۲۳] مؤلفان به



شکل ۱: هم‌بندی محاسبات مه دارای سه لایه حسگرها و راه‌اندازها، ابزارهای مه و ابر.

وسیله الگوریتم کوله‌پشتی توانستند ترتیب بهینه‌ای از وظایف را بر اساس مهلت زمانی و کمترین هزینه بیابند. آنها برنامه‌های با ظرفیت کوچک را در نظر گرفتند و بهترین مقادیر پارامترها شامل مهلت زمانی وظایف پیش‌بینی نشده، ازدحام شبکه و اندازه وظایف غیر دقیق را به دست آوردند. در [۲۴] مسئله زمان‌بندی متفاوتی به وسیله الگوریتم کوله‌پشتی حل شده است به طوری که وظایف موجود در شبکه توری هوشمند به وسیله الگوریتم کوله‌پشتی چندگانه و معیار زمان شروع، مدیریت شده است. همچنین به منظور بهینه‌سازی کوله‌پشتی چندگانه و زمان‌بندی بار از الگوریتم کلونی مورچگان استفاده گردید. در [۲۵] مسئله زمان‌بندی وظایف در ابر به وسیله ترکیبی از الگوریتم‌های کوله‌پشتی و ژنتیک حل شده است، معیارهای ارزیابی نتایج شامل بهره‌وری پردازنده، زمان پاسخ شبکه، نرخ ورودی و خروجی داده‌های دیسک بوده است. در این روش، انرژی مصرفی ماشین‌های فیزیکی و تعداد مهاجرت ماشین‌های مجازی نسبت به روش‌های معمول به صورت قابل توجهی کاهش یافته است.

یافته‌ها نشان می‌دهد بسیاری از مطالعات سیستم‌های ابری در کتابخانه کلودسیم انجام شده است. بعضی از پروژه‌های محاسبات مه در کلودسیم و برخی هم به زبان‌های برنامه‌نویسی مختلف پیاده‌سازی شده‌اند. iFogsim به عنوان یک توسعه جدید از کلودسیم جهت شبیه‌سازی محاسبات مه [۱] بسیار مناسب است. همچنین الگوریتم زمان‌بندی پیش‌فرض در این شبیه‌ساز (زمان‌بندی به ترتیب ورود) بهینه نمی‌باشد لذا در این تحقیق با استفاده از روش کوله‌پشتی بهینه‌شده با جستجوی هم‌زیستی KnapSOS توانستیم پارامترهای انرژی مصرفی، تأخیر زمانی و مجموع مصرف منابع شبکه را بهبود دهیم.

۳- مدل سیستم

محاسبات مه با قرارگیری بین گره‌های پایانی و ابر، برنامه‌های موجود در ابزارهای مه را پردازش می‌کند. هر برنامه در محاسبات مه دارای یک هم‌بندی متفاوت است. شبیه‌ساز iFogsim دارای یک رابط گرافیکی کاربر جهت طراحی هم‌بندی‌های جدید و تغییر آنها است [۱]. عناصر قابل افزودن شامل گره‌های حسگر، راه‌انداز، مه، ابر و پیوند بین اجزا است. ما یک هم‌بندی جدید با موضوع مراقبت پزشکی ایجاد نموده و همچنین از یک مدل آماده موجود در شبیه‌سازی جهت ارزیابی نتایج استفاده می‌نماییم. این هم‌بندی‌ها می‌توانند به وسیله ماژول‌های دیگر موجود در شبیه‌ساز، بارگذاری و یا اجرا شوند.

۳-۱ مدل ۱: نظارت هوشمند به وسیله شبکه‌ای از دوربین‌های توزیع‌شده

این مورد بر اساس یک سیستم توزیع‌شده از دوربین‌های نظارتی در محیط‌های مراقبت پزشکی، حمل و نقل، امنیت و ساخت و تولید است [۱]. مدل برنامه کاربردی برای این مورد شامل ماژول‌های ذیل می‌باشد: دو تشخیص‌دهنده برای اشیا و حرکت در جریان‌های ویدئویی خام که به وسیله دوربین ارسال می‌شود. ردیابی اشیا برای محاسبه پیکربندی بهینه از دوربین PTZ (قابلیت Pan امکان حرکت ۳۶۰ درجه را به دوربین می‌دهد، قابلیت Tilt امکان حرکت عمودی تا ۹۰ درجه را به دوربین می‌دهد و قابلیت Zoom نیز امکان بررسی از فاصله نزدیک‌تر و یا دورتر را فراهم می‌کند). به عنوان تنظیم‌کننده دوربین‌ها و خدمات‌دهنده به عنوان یک راه‌انداز. رابط کاربر جهت ارسال بخشی از اشیای ردیابی‌شده به ابزارهای کاربر.

هم‌بندی مدل ۱ با عنوان DCNS در شکل ۱ آورده شده است. در این معماری، همه موارد در مستطیل با خط ممتد، ابزارهای مه هستند، M_i عبارتند از iامین ماژول و m_i نیز iامین گره موبایل است. جریان فعالیت این مدل عبارتند از: دوربین ← تشخیص حرکت ← تشخیص شیء ← ردیاب شیء و رابط کاربری ← دوربین PTZ.

۳-۲ مدل ۲: تشخیص فعالیت در افراد سال‌خورده

ما یک برنامه جدید برای تشخیص فعالیت در افراد سال‌خورده با عنوان EHAD ارائه نمودیم که در خانه‌های هوشمند و یا مراکز درمانی قابل استفاده است. هدف این سیستم، نظارت و پیش‌بینی فعالیت‌های روزانه و یا رخدادهای غیر معمولی است که برای افراد کم‌توان پیش می‌آید [۲۶]. برای تشخیص رویداد افتادن شخص، ما از یک حسگر ژیروسکوپ جهت یافتن سرعت زاویه‌ای و جهت حرکت و همچنین یک حسگر شتاب جهت فعالیت‌های گردشی استفاده نمودیم. فرایند تعریف‌شده برای این مدل شامل، دریافت داده‌ها از گره‌های حسگر، تشخیص فعالیت یا افتادن شخص سال‌خورده به وسیله واحد ردیاب، بررسی نتایج به وسیله واحد کنترلر و ارسال پیام به صفحه نمایش است. شبیه‌سازی این برنامه تحت معماری ارائه‌شده از مه موجود در شکل ۱ صورت گرفته است. جریان فعالیت این مدل عبارتند از جریان داده حسگر ← ردیاب فعالیت ← کنترلر ← صفحه نمایش.

تمامی کلمات اختصاری موجود در مقاله با تعریف و توضیح آنها در جدول ۱ آورده شده است.

٤- روش پیشنهادی

زمان‌بندی در دسته مسایل NP-Hard قرار دارد و زمان اجرای زیادی نیاز دارد. بنابراین ما یک روش سریع جهت تخصیص منابع به برنامه‌ها ارائه می‌دهیم. روش زمان‌بندی منابع پیش‌فرض که در شبیه‌ساز موجود است، منابع را به صورت مساوی بین ماژول‌های برنامه‌ها تقسیم می‌کند.

جدول ۱: نمادها و تعریف‌ها.

توضیح	تعریف	اختصار
بردار سود	Benefits Flowers	BF
پهنای باند	Bandwidth	BW
ساعت کلودسیم	Cloudsim Clock	CC
مصرف انرژی جاری	Current Energy Consumption	CEC
شبکه‌های مرکز داده	Data Center NetworkS	DCNS
تشخیص فعالیت افراد سال‌خورده	Elderly Human's Activity Detection	EHAD
آخرین بهره‌وری میزبان	Host's Last Utilization	HLU
آخرین بهره‌وری	Last Utilization	LU
آخرین زمان به روز رسانی بهره‌وری سیستم	Last Utilization Update Time	LUUT
میلیون دستورات اجراشده در هر ثانیه	Million Instruction Per Second	MIPS
زمان جاری	Now Time	NT
بردار همیاری	Mutual Vector	MV
هزینه اجرای قبلی	Past Execution Cost	PEC
بردار انگل	Parasite Vector	PV
جستجوی هم‌زیستی جانداران	Symbiotic Organisms Search	SOS
نرخ اجرای دستورات	Rate Per MIPS	RPM
مجموع تأخیر	Total Latency	TL
مجموع دستورات اجراشده	Total MIPS	TM
اندازه مجموع	Total Size	TS
بهره‌وری کلی پردازنده	Total Utilization Cost	TUC

در واقع با اجرای شبیه‌ساز و شروع فرایند منطبق با الگوریتم ۲، هر برنامه کاربردی، N ماژول $\{M_1, M_2, \ldots, M_n\}$ تولید می‌کند. الگوریتم پیشنهادی ما، پردازنده‌های موجود در ابزارهای مه را به این ماژول‌ها تخصیص می‌دهد. ما سیاست پیش‌فرض موجود در تابع UpdateAllocatedMips از کلاس FogDevice در شبیه‌ساز [۱] را با استفاده از الگوریتم کوله‌پشتی بهینه‌شده با جستجوی هم‌زیستی تغییر دادیم که در ادامه مقاله آن را به صورت KnapSOS آورده‌ایم.

۴-۱ جستجوی هم‌زیستی

الگوریتم هم‌زیستی بر اساس ارتباط بین دو موجود زنده است که در کل اکوسیستم پراکنده هستند. منظور از موجود زنده، در واقع نگاشتی از ماژول‌ها به پردازنده‌ها می‌باشد. در واقع هر موجود زنده به عنوان یک پاسخ برای مسئله زمان‌بندی محسوب می‌شود به طوری که در قالب یک بردار، تعیین می‌گردد هر پردازنده به کدام ماژول تخصیص یابد. بنابراین اندیس‌های این بردار برابر شماره پردازنده و مقادیر بردار نشان‌دهنده شماره ماژول می‌باشد.

الگوریتم هم‌زیستی طی سه مرحله صورت می‌گیرد. در مرحله اول با عنوان عملیات هم‌زیستی، هر دو موجود زنده در کنار یکدیگر سود می‌برند. در مرحله دوم با عنوان هم‌سفرگی، یکی از موجودات زنده سود می‌برد و دیگری نه سود می‌برد نه زیان. در مرحله سوم با عنوان انگلی، یکی از موجودات زنده سود می‌برد و دیگری زیان می‌بیند [۸].

الگوریتم ۱: KnapSOS

۱: مقداردهی اولیه M موجود زنده به صورت تصادفی.
۲: محاسبه مقادیر برازندگی به وسیله (۱).
۳: آماده‌سازی بسته‌ها (موجودات زنده) برای قراگیری در کوله‌پشتی.
۴: شروع الگوریتم SOS:
۵: مقداردهی اکوسیستم و بیشترین تعداد تکرار.
۶: شروع **حلقه ۱** (از $i=1$ تا بیشترین تعداد تکرار N)
۷: شناسایی بهترین موجود زنده.
۸: شروع **حلقه ۲** (از $j=1$ تا M بیشترین تعداد موجودات زنده موجود زنده در اکوسیستم)
۹: عملیات هم‌زیستی.
۱۰: عملیات هم‌سفرگی.
۱۱: عملیات انگلی.
۱۲: به روز رسانی بهترین موجود زنده.
۱۳: پایان **حلقه ۲**.
۱۴: پایان **حلقه ۱**.
۱۵: شروع **حلقه ۳** (از $k=1$ تا بیشترین تعداد بسته‌ها)
۱۶: بسته k (موجود زنده) را در کوله‌پشتی قرار بده به طوری که کوله‌پشتی پر نشود.
۱۷: پایان **حلقه ۳**.

بر طبق الگوریتم ۱، در خط ۱ M موجود زنده به صورت تصادفی ایجاد می‌گردند $(M=20)$. در خط ۲ مقادیر برازندگی این موجودات با استفاده از (۱) محاسبه می‌شود. در خط ۳ و ۴ موجودات زنده جهت عملیات هم‌زیستی آماده می‌شوند. در خط ۵ بیشترین تعداد تکرار مراحل الگوریتم به عنوان N برای اجراشدن در خطوط ۶ تا ۱۳ تعیین می‌گردد $(N=50)$. عملیات هم‌زیستی، هم‌سفرگی و انگلی به ترتیب در خطوط ۸، ۹ و ۱۰ اجرا می‌شوند. در خط ۱۱ بهترین موجودات زنده با کمترین مقدار برازندگی برای پرکردن کوله‌پشتی انتخاب می‌شوند. در خطوط ۱۴ تا ۱۶ بهترین بسته‌ها (موجودات زنده) برای قرارگیری در کوله‌پشتی انتخاب می‌شوند به طوری که کوله‌پشتی پر نشود.

بر طبق الگوریتم ۱، سه مرحله اصلی از الگوریتم جستجوی هم‌زیستی به صورت ذیل می‌باشد:

هم‌زیستی

۱: انتخاب تصادفی یک موجود زنده.
۲: محاسبه بردار همیاری MV و بردار سود BF به طوری که $MV=(X_i+X_j)/2$.
۳: $(BF1, BF2)$ مقادیر تصادفی با مقدار ۱ یا ۲ می باشند.
۴: به روز رسانی موجودات x_i و x_j بر اساس ارتباط همیاری بین آنها.
۵: $x_i^{new}=X_i+rand(0,1)\times(X_{best}-MV\times BF1)$ و $x_j^{new}=X_j+rand(0,1)\times(X_{best}-MV\times BF2)$.
۶: محاسبه مقادیر برازندگی موجودات به روز شده.
۷: اگر موجودات به روز شده بهتر از موجودات قبلی باشند آن گاه به عنوان اعضای جدید مورد پذیرش قرار می‌گیرند.

هم‌سفرگی

۱: انتخاب تصادفی موجود زنده X_j.
۲: به روز رسانی موجود زنده X_i با همکاری X_j.
۳: محاسبه مقدار برازندگی موجود زنده جدید.
۴: اگر موجود زنده جدید بهتر از قبلی باشد آن گاه مورد پذیرش قرار می‌گیرد.

انگلی

۱: انتخاب تصادفی موجود زنده X_j.

۲: ایجاد انگل PV از موجود زنده X_i.
۳: محاسبه برازندگی موجود زنده جدید.
۴: اگر برازندگی PV بهتر از برازندگی X_j باشد آن گاه X_j را با انگل PV جایگزین می‌نماییم.

الگوریتم KnapSOS از تابع برازندگی (۱) جهت یافتن بهترین موجود زنده به ماژول‌ها استفاده می‌کند

$$Fitness = \frac{1}{TUC + BW} \quad (1)$$

که در (۱) TUC برابر بهره‌وری کلی پردازنده برای پردازنده‌های تخصیص‌یافته به ماژول موجود در برنامه و BW نیز پهنای باند لینک ارتباطی به ماژول در برنامه است. هدف از اجرای الگوریتم، یافتن کمترین مقدار برازندگی می‌باشد.

٤-۲ پیچیدگی زمانی الگوریتم KnapSOS

برای محاسبه پیچیدگی زمانی الگوریتم KnapSOS ۱، تعداد موجودات زنده را برابر M و تعداد تکرار را برابر N در نظر می‌گیریم. خطوط ۱ تا ۳ برای همه موجودات زنده انجام می‌شود و بنابراین پیچیدگی آن برابر $3M$ می‌شود. خط ۴ که یک توضیح می‌باشد و پیچیدگی ندارد. پیچیدگی خط ۵ برابر عدد ثابت C است. خط ۶ شروع حلقه ۱ پیچیدگی N دارد. خط ۷ مرتب‌سازی موجودات زنده بر اساس مقادیر برازندگی با پیچیدگی $M \log M$ است. خط ۸ شروع حلقه ۲ با پیچیدگی M است. در خطهای ۹ تا ۱۱ پیچیدگی عملیات هم‌زیستی برابر $7C$، هم‌سفرگی برابر $4C$ و انگلی برابر $4C$ می‌باشد که مجموع آنها برابر $15C$ است. خط ۱۲ دارای پیچیدگی C است. خط ۱۶ نیز به ازای تمام موجودات زنده تا پرشدن کوله‌پشتی تکرار می‌شود یعنی از پیچیدگی M می‌باشد

$$O(3M + C + N \times (M + 15C + M \log M + C + M)) \quad (2)$$

چون مقدار ثابت تأثیری در محاسبه پیچیدگی زمانی ندارد لذا (۲) به صورت ذیل ساده می‌شود

$$O(3M + N \times (2M + M \log M)) \quad (3)$$

در نتیجه، پیچیدگی زمانی به صورت ذیل می‌باشد

$$O(NM \log M) \quad (4)$$

٤-۳ زمان‌بندی ماژول‌ها مبتنی بر الگوریتم KnapSOS

تابع اصلی اجرایی شبیه‌ساز به صورت شبه‌کد در الگوریتم ۲ آورده شده است.

الگوریتم ۲: زمان‌بندی مبتنی بر الگوریتم KnapSOS

۱: ایجاد برنامه و ارسال برای واسطه مه.
۲: افزودن برنامه به ابزار مه.
۳: شروع **حلقه ۱** (به ازای تمامی ناحیه‌ها)
۴: شروع **حلقه ۲** (به ازای تمامی دوربین‌ها)
۵: ایجاد ابزار مه (نام گره، MIPS، اندازه حافظه، کمترین و بیشترین پهنای باند، توان و توان در حالت بیکاری).
۶: پایان **حلقه ۲**.
۷: پایان **حلقه ۱**.
۸: افزودن ماژول‌های مدل (برنامه) به ابزارهای مه.
۹: اتصال ماژول‌ها با لبه‌ها.
۱۰: تعریف روابط ورودی/ خروجی و حلقه‌ای از ماژول‌های برنامه بر اساس جریان فعالیت مدل.
۱۱: مقداردهی اولیه نگاشت ماژول‌ها.
۱۲: ارسال برنامه.
۱۳: شروع شبیه‌ساز iFogsim.
۱۴: فراخوانی زمان‌بند KnapSOS.
۱۵: شروع **حلقه ۳** (به ازای تمامی ماژول‌ها)
۱۶: اگر ماژول برنامه در پردازنده جاری در حال اجرا باشد آن گاه برو به پردازنده بعدی.
۱۷: وگرنه پردازنده را به ماژول برنامه تخصیص بده.
۱۸: پایان **حلقه ۳**.
۱۹: به روز رسانی انرژی مصرفی، مصرف منابع شبکه و هزینه اجرا.
۲۰: متوقف‌کردن شبیه‌ساز iFogsim.
۲۱: استخراج خروجی و ارزیابی نتایج.

در الگوریتم ۲ در خط ۱، برنامه ایجاد و به واسطه مه ارسال می‌شود. در خط ۲، برنامه به ابزار مه اضافه می‌گردد. در خطهای ۳ تا ۷ به ازای هر دوربین و ناحیه یک ابزار مه در نظر گرفته می‌شود. برنامه بر اساس ویژگی‌های موجود در خط ۵ به ابزار مه اضافه می‌شود. در خط ۸ ماژول‌های برنامه در ابزارهای مه ایجاد می‌گردند و بر اساس هم‌بندی تعریف‌شده در مدل DCNS و یا EHAD (که در بخش‌های ۳-۱ و ۳-۲ معرفی شدند) در خطهای ۹ و ۱۰ با هم مرتبط می‌شوند.

مرحله بعدی، نگاشت ماژول‌ها در خط ۱۱، ارسال برنامه تعریف‌شده در خط ۱۲ و شروع شبیه‌ساز iFogsim در خط ۱۳ است. عملیات زمان‌بندی برای تخصیص پردازنده‌های موجود در ابزارهای مه به ماژول‌ها با فراخوانی الگوریتم ۱ یعنی KnapSOS در خط ۱۴ انجام می‌شود. در خط ۱۵ تا ۱۸ همه ماژول‌ها جهت اجرا مورد بررسی قرار می‌گیرند، بر طبق خط ۱۶ اگر هر ماژول در پردازنده جاری در حال اجرا باشد آن گاه کنترل به پردازنده بعدی می‌رود تا این که ماژول دیگری را بررسی کند وگرنه بر طبق خط ۱۷، پردازنده جاری به این ماژول تخصیص می‌یابد.

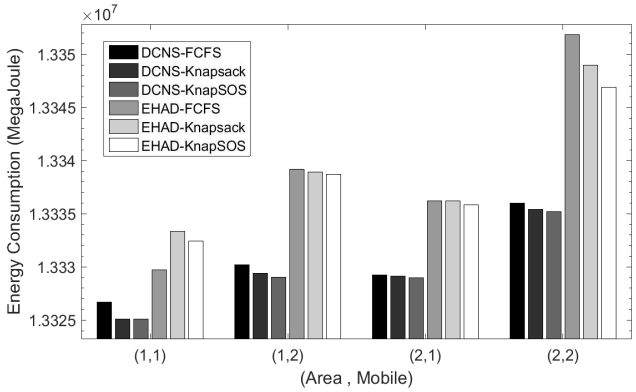
بعد از انجام عملیات زمان‌بندی، پارامترهای هزینه، مصرف منابع شبکه و مصرف انرژی به ترتیب با استفاده از (۵)، (۶) و (۷) در خط ۱۹ به روز رسانی می‌شوند. در نهایت شبیه‌ساز iFogsim در خط ۲۰ متوقف شده و خروجی‌های شبیه‌ساز در خط ۲۱ استخراج و ارزیابی می‌گردد.

بر طبق شبیه‌ساز iFogsim، واسطه‌ها، برنامه‌ها، ابزارهای مه و روابط مورد نیاز بین ماژول‌ها ایجاد شده و سپس عملیات زمان‌بندی جهت تخصیص منابع فراخوانی می‌شود. در الگوریتم ۲ تمامی مراحل تعریف‌شده برای هر دو مدل از سیستم پیشنهادی اجرا می‌گردد.

به منظور قراردادن هر یک از اجزا در هم‌بندی مه، یک ابزار مه با استفاده از الگوریتم ۴ ایجاد می‌گردد. منظور از تاپل‌ها، واحدهای ارتباطی بین اجزا در iFogsim است که در کلاسی با عنوان Tuple تعریف شده و این کلاس از کلاس کلودلت در کلودسیم [۲۷] ارث‌بری داشته است [۱]. هر ابزار مه درون خود یک مرکز داده کوچک دارد که دارای پردازنده و میزبان‌هایی با قدرت پایین‌تر نسبت به ابر می‌باشند. ابزار مه به حسگرهای پایانی و ابر نزدیک هستند و بنابراین می‌توانند عملیات پردازش را به سرعت انجام دهند و داده‌های ذخیره‌شده را به صورت محلی ذخیره نمایند. برنامه‌ها در ابزارهای مه دارای تعدادی ماژول می‌باشند که مانند کلودلت‌ها در ابر هستند.

الگوریتم ۳: ایجاد برنامه

۱: افزودن همه ماژول‌ها به برنامه (نام ماژول، ظرفیت حافظه).



شکل ۲: انرژی مصرفی برای دو مدل از سیستم با روش‌های زمان‌بندی به ترتیب ورود FCFS، کوله‌پشتی Knapsack و KnapSOS.

۲: افزودن همه لبه‌ها بین ماژول‌ها برای برنامه (مبدأ، مقصد، نام ماژول مقصد، اندازه پردازنده تاپل و جهت).
۳: نگاشت تاپل‌ها (نام ماژول، نوع تاپل ورودی و تاپل خروجی).
۴: افزودن حلقه‌ها به ماژول‌ها.

در الگوریتم ۲ به منظور ایجاد برنامه، ماژول‌ها و نگاشت برنامه‌ها به ماژول‌ها در گراف هم‌بندی از کلاس FogDevice استفاده می‌شود. روابط ورودی و خروجی به وسیله تاپل‌ها تعریف می‌شوند. در مرحله نهایی، تعدادی حلقه جهت نظارت بر تأخیر زمانی ردیاب‌های اشیا تعریف شده است. مراحل مذکور در الگوریتم ۳ تعریف می‌گردد و در الگوریتم ۲ مورد فراخوانی قرار می‌گیرد.

الگوریتم ٤: ایجاد ابزار مه

۱: ایجاد لیست پردازنده‌ها.
۲: ایجاد میزبان‌ها (ورودی، سیستم عامل، ماشین‌های مجازی، هزینه کلی و هزینه در هر محل ذخیره‌سازی).
۳: ایجاد محل‌های ذخیره‌سازی.
۴: تنظیم تأخیر زمانی، کمترین و بیشترین پهنای باند.
۵: نگاشت ماژول‌ها به برنامه.

٤-٤ پارامترهای ارزیابی

پارامتر هزینه اجرای شبیه‌سازی بر اساس (۵) می‌باشد

$$Cost = PEC + (CC - LUUT) \times RPM \times LU \times TM \qquad (۵)$$

در (۵)، PEC برابر هزینه اجرای قبلی در ابزار مه، CC برابر ساعت کلودسیم، $LUUT$ برابر آخرین زمان به روز رسانی بهره‌وری سیستم، RPM برابر نرخ $MIPS$، LU برابر آخرین میزان بهره‌وری و TM برابر میزان $MIPS$ کلی برای میزبان است. مصرف کلی شبکه بر اساس (۶) است

$$NetworkUsage = \frac{\sum_{i=1}^{N} (TL_i \times TS_i)}{MST} \qquad (۶)$$

روابط بین ماژول‌ها به وسیله تاپل‌ها مشخص می‌شود و مصرف منابع شبکه به اندازه تاپل‌های منتقل‌شده در زمان معین بستگی دارد. در (۶)، TL_i و TS_i برابر مجموع تأخیرها و مجموع اندازه تاپل‌های مربوط به ماژول‌ها، N برابر تعداد کل تاپل‌ها و MST برابر بیشترین زمان شبیه‌سازی است. همچنین انرژی مصرفی شبیه‌سازی برای هم‌بندی کامل شبکه به وسیله (۷) محاسبه می‌گردد

$$Energy = CEC + (NT - LUUT) \times HLU \qquad (۷)$$

در (۷)، CEC برابر انرژی مصرفی جاری، NT برابر زمان فعلی، $LUUT$ برابر زمان آخرین به روز رسانی بهره‌وری سیستم و HLU برابر میزان آخرین بهره‌وری میزبان است.

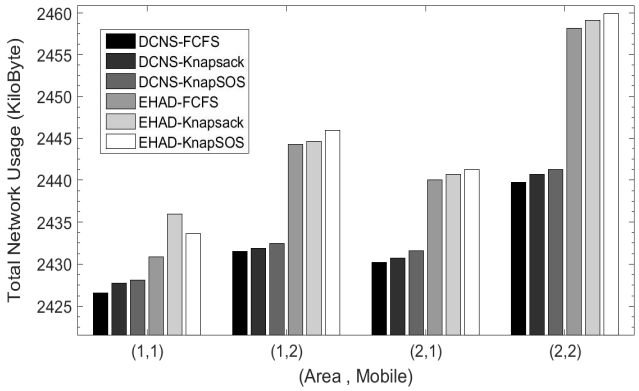
پارامترهای مذکور در کلاس FogDevice از شبیه‌ساز iFogsim محاسبه می‌گردد. ما نتایج مربوط به شبیه‌سازی را بر اساس پارامترهای معرفی‌شده با سه روش کوله‌پشتی، زمان‌بندی به ترتیب ورود و کوله‌پشتی بهینه با جستجوی هم‌زیستی مقایسه می‌نماییم. بعد از تغییر تابع updateAllocatedMips در کلاس FogDevice، ما دو مدل سیستم را در شبیه‌ساز اجرا نموده و به تحلیل نتایج می‌پردازیم. الگوریتم KnapSOS به ما کمک می‌کند تا این که انرژی مصرفی و تأخیر زمانی اجرای ماژول‌ها را کاهش دهیم.

۵- شبیه‌سازی و نتایج

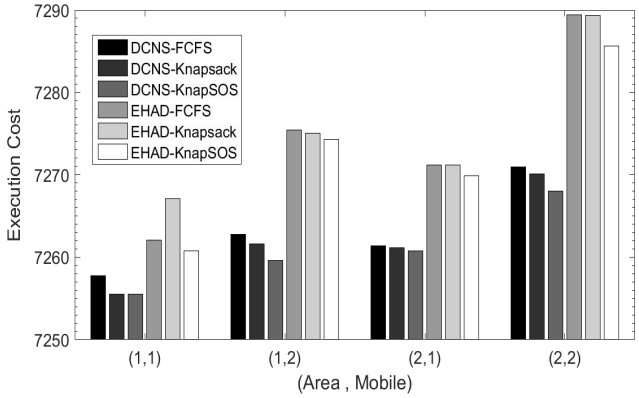
کتابخانه iFogsim مبتنی بر زبان برنامه‌نویسی جاوا بوده و دارای ماژول‌ها و کلاس‌هایی جهت شبیه‌سازی محاسبات مه است. این کتابخانه توسعه‌یافته شبیه‌ساز کلودسیم است و از بسیاری کلاس‌های کلودسیم بهره می‌برد. سیستم رایانه‌ای که شبیه‌سازی این تحقیق در آن اجرا شده است دارای پردازنده اینتل از نوع Core i۵ با سرعت ۲٫۶۷ گیگاهرتز، حافظه RAM به اندازه ۳ گیگابایت و تحت سیستم عامل مایکروسافت ویندوز ۱۰ از نوع ۳۲بیتی می‌باشد. به منظور اجرای روش زمان‌بندی پیشنهادی، ما از دو مدل هم‌بندی معرفی‌شده استفاده می‌نماییم. در الگوریتم هم‌زیستی، بیشترین تعداد تکرار برابر ۱۰۰ و تعداد موجودات زنده برابر ۳۰ در نظر گرفته شده است. روش پیش‌فرض شبیه‌ساز، زمان‌بندی به ترتیب ورود است. همچنین ما شبیه‌سازی را با سه روش زمان‌بندی به ترتیب ورود FCFS، کوله‌پشتی Knapsack و روش پیشنهادی KnapSOS برای دو مدل سیستم شامل DCNS و EHAD اجرا می‌نماییم. ما ۴ حالت مختلف از تعداد نواحی و دوربین‌ها به صورت $(Area, Mobile) = \{(۱,۱),(۱,۲),(۲,۱),(۲,۲)\}$ را در نظر گرفته و نتایج شبیه‌سازی را برای مدل‌های سیستم ارائه می‌نماییم.

در شکل ۲ انرژی مصرفی برای دو مدل سیستم بر حسب مگاژول نشان داده شده است. بر طبق نتایج در مدل DCNS، انرژی مصرفی روش پیشنهادی KnapSOS نسبت به روش‌های FCFS به اندازه ۰٫۷۳٪ و Knapsack به اندازه ۰٫۱۴٪ کاهش داشته است. در مدل EHAD، انرژی مصرفی روش پیشنهادی KnapSOS نسبت به روش‌های FCFS به اندازه ۰٫۵۷٪ و Knapsack به اندازه ۰٫۶۶٪ کاهش داشته است. بنابراین روش‌های کوله‌پشتی و KnapSOS جهت تخصیص منابع و زمان‌بندی ماژول‌ها در مه نسبت به روش زمان‌بندی به ترتیب ورود، روش مناسب‌تر و بهینه‌تری است.

شکل ۳ مجموع مصرف شبکه را بر حسب کیلوبایت نشان می‌دهد به طوری که در مدل DCNS، متوسط مصرف شبکه به وسیله الگوریتم زمان‌بندی پیشنهادی KnapSOS نسبت به روش‌های FCFS به اندازه ۵٫۴۹٪ و Knapsack به اندازه ۲٫۴۳٪ افزایش داشته است. همچنین این پارامتر برای مدل EHAD به وسیله روش KnapSOS نسبت به روش‌های FCFS به اندازه ۷٫۶۵٪ و Knapsack به اندازه ۰٫۴۱٪ افزایش داشته است. لذا پارامتر مجموع مصرف شبکه با استفاده از روش پیشنهادی این تحقیق نسبت به روش‌های زمان‌بندی به ترتیب ورود و کوله‌پشتی دارای کارایی بیشتری است.



شکل ۳: مجموع مصرف شبکه برای دو مدل از سیستم با روش‌های زمان‌بندی به ترتیب ورود FCFS، کوله‌پشتی Knapsack و KnapSOS.



شکل ۴: مقایسه هزینه اجرا در ابر برای دو مدل از سیستم با روش‌های زمان‌بندی به ترتیب ورود FCFS، کوله‌پشتی Knapsack و KnapSOS.

جدول ۲: مقایسه مدل‌های سیستم بر اساس روش‌های زمان‌بندی به ترتیب ورود FCFS، کوله‌پشتی KNAPSACK و KNAPSOS.

مدل	الگوریتم	انرژی	مصرف شبکه	هزینه
DCNS	FCFS	متوسط	متوسط	متوسط
	Knapsack	کم	متوسط	کم
	KnapSOS	کم	کم	کم
EHAD	FCFS	زیاد	زیاد	زیاد
	Knapsack	متوسط	زیاد	متوسط
	KnapSOS	کم	متوسط	کم

بر طبق شکل ۴ هزینه اجرای شبیه‌سازی در مدل DCNS به وسیله الگوریتم زمان‌بندی پیشنهادی KnapSOS نسبت به روش‌های FCFS به اندازه ۴/۰۹٪ و Knapsack به اندازه ۲/۹۲٪ کاهش داشته است. همچنین این پارامتر برای مدل EHAD به وسیله روش KnapSOS نسبت به روش‌های FCFS به اندازه ۵/۱۳٪ و Knapsack به اندازه ۵/۰۶٪ کاهش داشته است. مقایسه روش پیشنهادی این تحقیق نسبت به روش‌های زمان‌بندی به ترتیب ورود و کوله‌پشتی برای دو مدل سیستم نشان‌دهنده انرژی مصرفی و هزینه اجرای کمتری در ابر می‌باشد و همچنین روش‌های پیشنهادی نسبت به روش پیش‌فرض شبیه‌ساز iFogsim، میزان مصرف شبکه را افزایش می‌دهد که این امر موجب بیشینه‌سازی استفاده از منابع شبکه می‌گردد و به این دلیل که توانسته است زمان بیکاری منابع شبکه را کاهش دهد و منابع را مشغول‌تر نگه دارد لذا یک مزیت محسوب می‌شود.

الگوریتم‌های تکاملی و روش‌های مکاشفه‌ای دارای زمان اجرای طولانی‌تر از روش کوله‌پشتی می‌باشند اما برای رسیدن به بهترین جواب‌ها، ما الگوریتم کوله‌پشتی را با روش جستجوی هم‌زیستی بهینه نمودیم و به نتایج بهتری از روش پیش‌فرض شبیه‌ساز در زمان‌بندی دست یافتیم. مقایسه دو مدل سیستم در جدول ۲ آورده شده است. در این جدول کارایی روش‌های زمان‌بندی به ترتیب ورود، کوله‌پشتی و KnapSOS در سه سطح کم، متوسط و زیاد ارائه شده است. برای نمونه در مدل DCNS، انرژی مصرفی و هزینه اجرا نسبت به روش زمان‌بندی به ترتیب ورود از حالت متوسط به کم، کاهش یافته و همچنین در مدل EHAD نیز به همین ترتیب بوده است.

٦- نتیجه‌گیری

از آنجایی که ابر در موقعیتی دور از حسگرها قرار دارد لذا انتقال داده دارای سربار زمانی زیادی است، بنابراین محاسبات مه برای برنامه‌های مبتنی بر حسگر دارای معماری قابل پذیرش است. در این تحقیق، ما روش زمان‌بندی جدید را در شبیه‌ساز iFogsim ارائه دادیم. روش‌های پیشنهادی ما بر اساس روش کوله‌پشتی و الگوریتم هم‌زیستی جهت بهینه‌سازی آن است که در دو مدل ردیابی سیگنال‌های ارسالی از حسگرها با عنوان DCNS و EHAD مبتنی بر دوربین‌ها و راه‌اندازها شبیه‌سازی شده است. نتایج نشان می‌دهد که روش پیشنهادی KnapSOS در دو مدل (EHAD, DCNS):

- انرژی مصرفی را به اندازه (۰/۷۳٪، ۰/۵۷٪) نسبت به روش به ترتیب ورود و (۰/۱۴٪، ۰/۶۶٪) نسبت به روش کوله‌پشتی کاهش داده است.
- مجموع مصرف شبکه را به اندازه (۵/۴۹٪، ۷/۶۵٪) نسبت به روش به ترتیب ورود و (۲/۴۳٪، ۰/۴۱٪) نسبت به روش کوله‌پشتی افزایش داده است.
- هزینه اجرا را به اندازه (۴/۰۹٪، ۵/۱۳٪) نسبت به روش به ترتیب ورود و (۲/۹۲٪، ۵/۰۶٪) نسبت به روش کوله‌پشتی کاهش داده است.

بنابراین روش KnapSOS نسبت به روش‌های زمان‌بندی به ترتیب ورود و حتی کوله‌پشتی دارای کارایی بیشتری است. در کارهای آینده، ما به دنبال تحقیق بر روی روش‌های مکاشفه‌ای جهت زمان‌بندی در شبکه‌های مه با در نظر گرفتن پارامترهای کیفیت سرویس و امنیت در مدل‌های سیستمی بیشتری می‌باشیم.

مراجع

[1] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "IFogsim: a toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275-1296, Jun. 2017.

[2] M. Aazam, M. St-Hilaire, C. H. Lung, and I. Lambadaris, "Pre-fog: Iot trace based probabilistic resource estimation at fog," in *Proc. 13th IEEE Annual. Consumer Communications and Networking Conf., CCNC'16,* pp. 12-17, Jan. 2016.

[3] A. M. Rahmani, T. N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, and P. Liljeberg, "Exploiting smart e-health gateways at the edge of healthcare internet-of-things: a fog computing approach," *Future Generation Computer Systems*, vol. 78, no. 2, pp. 641-658, Feb. 2017.

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14-23, Oct. 2009.

[5] P. Gupta and S. P. Ghrera, "Trust and deadline aware scheduling algorithm for cloud infrastructure using ant colony optimization," in *Proc. IEEE Int. Conf. on Innovation and Challenges in Cyber Security, ICICCS-INBUSH'16*, pp. 187-191, Noida, India, 3-5 Feb. 2016.

[6] M. A. Rodriguez and R. Buyya, "Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds," *IEEE Trans. on Cloud Computing*, vol. 2, no. 2, pp. 222-235, Apr. 2014.
[7] J. Lv, X. Wang, M. Huang, H. Cheng, and F. Li, "Solving 0-1 knapsack problem by greedy degree and expectation efficiency," *Applied Soft Computing*, vol. 41, pp. 94-103, Apr. 2016.
[8] M. Y. Cheng and D. Prayogo, "Symbiotic organisms search: a new metaheuristic optimization algorithm," *Computers & Structures*, vol. 139, vol. 139, pp. 98-112, Jul. 2014.
[9] M. A. Rodriguez and R. Buyya, "A taxonomy and survey on scheduling algorithms for scientific workflows in iaas cloud computing environments," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 8, pp. 1-23, Dec. 2017.
[10] M. E. Frincu, S. Genaud, and J. Gossa, "Comparing provisioning and scheduling strategies for workflows on clouds," in *Proc. IEEE 27th Int. Parallel and Distributed Processing Symp. Workshops and PhD Forum, IPDPSW'13,* pp. 2101-2110, Cambridge, MA, USA, 20-24 May. 2013.
[11] M. Malawski, K. Figiela, M. Bubak, E. Deelman, and J. Nabrzyski, "Scheduling multilevel deadline-constrained scientific workflows on clouds based on cost optimization," *Scientific Programming*, vol. 2015, pp. 1-13, Jan. 2015.
[12] J. J. Durillo and R. Prodan, "Multi-objective workflow scheduling in amazon ec2," *Cluster Computing*, vol. 17, no. 2, pp. 169-189, Jun. 2014.
[13] D. Poola, K. Ramamohanarao, and R. Buyya, "Fault-tolerant workflow scheduling using spot instances on clouds," *Procedia Computer Science*, vol. 29, pp. 523-533, Jun. 2014.
[14] X. Zhu, J. Wang, H. Guo, D. Zhu, L. T. Yang, and L. Liu, "Fault-tolerant scheduling for real-time scientific workflows with elastic resource provisioning in virtualized clouds," *IEEE Trans. on Parallel and Distributed Systems*, vol. 27, no. 12, pp. 3501-3517, Mar. 2016.
[15] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, "Mobility-aware application scheduling in fog computing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 26-35, Apr. 2017.
[16] X. Q. Pham and E. N. Huh, "Towards task scheduling in a cloud-fog computing system," in *Proc. 18th Asia-Pacific IEEE Network Operations and Management Symp., APNOMS'*, 4 pp., Kanazawa, Japan, 5-7 Oct. 2016.
[17] Y. Sun, T. Dang, and J. Zhou, "User scheduling and cluster formation in fog computing based radio access networks," in *Proc. IEEE Int. Conf. on Ubiquitous Wireless Broadband ICUWB*, 4 pp., Nanjing, China, 16-19 Oct. 2016.
[18] X. Chen and L. Wang, "Exploring fog computing-based adaptive vehicular data scheduling policies through a compositional formal methodpepa," *IEEE Communications Letters*, vol. 21, no. 4, pp. 745-748, Jan. 2017.
[19] H. E. Zahaf, A. E. H. Benyamina, R. Olejnik, and G. Lipari, "Energy efficient scheduling for moldable real-time tasks on heterogeneous computing platforms," *J. of Systems Architecture*, vol. 74, pp. 46-60, Mar. 2017.
[20] S. Bitam, S. Zeadally, and A. Mellouk, "Fog computing job scheduling optimization based on bees swarm," *Enterprise Information Systems*, vol. vol. 12 no. 4, pp. 373-397, Apr. 2017.
[21] T. Wang, X. Wei, C. Tang, and J. Fan, "Efficient multi-tasks scheduling algorithm in mobile cloud computing with time constraints," *Peer-to-Peer Networking and Applications*, 11, no. 4, pp. 793-807, Jul. 2017.
[22] F. Lao, X. Zhang, and Z. Guo, "Parallelizing video transcoding using map-reduce-based cloud computing," in *Proc. IEEE Int.Symp.on Circuits and Systems, ISCAS'12,* pp. 2905-2908, Seoul, South Korea, 20-23 May 2012.
[23] M. A. Rodriguez and R. Buyya, "A responsive knapsack-based algorithm for resource provisioning and scheduling of scientific workflows in clouds," in *Proc. 44th IEEE Int. Conf. on Parallel Processing, ICPP'15,* pp. 839-848, Beijing, China, 1-4 Sept. 2015.
[24] S. Rahim, *et al*., "Towards multiple knapsack problem approach for home energy management in smart grid," in *Proc. 18th IEEE Int. Conf. on Network-Based Information Systems, NBiS'15,* pp. 48-52, Taipei, Taiwan, 2-4 Sept. 2015.
[25] S. Chen, J. Wu, and Z. Lu, "A cloud computing resource scheduling policy based on genetic algorithm with multiple fitness," in *Proc. 12th IEEE Int. Conf. on Computer and Information Technology, CIT'12,* pp. 177-184, Chengdu, China, 27-29 Oct. 2012.
[26] Q. Ni, A. B. Garcia Hernando, and I. P. de la Cruz, "The elderlys independent living in smart homes: a characterization of activities and sensing infrastructure survey to facilitate services development," *Sensors*, vol. 15, no. 5, pp. 11312-11362, May 2015.
[27] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and 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, no. 1, pp. 23-50, Aug. 2011.

دادمهر رهبری در سال ۱۳۸۵ مدرک کارشناسی مهندسی کامپیوتر خود را از دانشگاه علم و صنعت ایران و در سال ۱۳۸۸ مدرک کارشناسی ارشد مهندسی کامپیوتر خود را از دانشگاه آزاد اسلامی واحد مشهد دریافت نمود. او در سال ۱۳۹۵ به دوره دکتری مهندسی فناوری اطلاعات در دانشگاه قم وارد گردید و هم‌اکنون به صورت تمام وقت مشغول تحصیل می باشد. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: زمانبندی و مدیریت منابع در محیط ابر و مه، الگوریتم های یادگیری ماشین و امنیت اطلاعات.

محسن نیک‌رای در سال ۱۳۸۱ مدرک کارشناسی مهندسی کامپیوتر خود را از دانشگاه علم و صنعت ایران دریافت کرد. ایشان مدرک کارشناسی ارشد و دکتری مهندسی کامپیوتر خود را در سال‌های ۱۳۸۵ و ۱۳۹۲ از دانشگاه تهران دریافت نمود. دکتر نیک‌رای از سال ۱۳۹۵ در گروه مهندسی کامپیوتر و فناوری اطلاعات دانشگاه قم به‌عنوان عضو هیأت علمی مشغول فعالیت می‌باشد. زمینه‌های علمی مورد علاقه نام‌برده شامل زمانبندی و مدیریت منابع در محیط ابر و مه می باشد.