****

**پردازش کلان داده ها با استفاده از داده های HDFS و تکنیک های تکاملی خوشه**

**چکیده**

استفاده روز افزون از سیستم­های فعال شده اینترنت (IOT) منجر به مقادیر فراوانی از داده­ها با ساختارهای مختلف شده است. اکثر راه حل­های کلان داده­ها در بالای سیستم اکو هادوپ یا از سیستم­ فایل توزیع شده (HDFS) استفاده می­شوند. با این حال، بررسی­های ناکارآمدی در این سیستم در هنگام برخورد با داده­ها نشان داده شده­اند. برخی از تحقیقات این مسائل را برای نوع خاصی از داده­های گراف برطرف می­کنند، اما امروزه اطلاعات بیشتری از داده­ها قابل دسترس هستند. چنین مسائلی مربوط به عملکردهایی می­شوند که منجر به مسائل بزرگی از جمله فضای بزرگ­تر مورد نیاز در مراکز داده، و از بین رفتن منابع (مانند مصرف انرژی)، و مشکلات زیست محیطی (مانند انتشار بیشتر کربن) نیز می­شود. ما یک ماژول اطلاعاتی را برای سیستم اکو هادوپ ارائه می­دهیم. ما همچنین یک روش رمزگذاری توزیع شده را برای الگوریتم­های ژنتیک ارائه می­دهیم. چارچوب ما این امکان را می­دهد تا هادوپ بتواند توزیع داده­ها و قرار دادن آنها براساس تجزیه و تحلیل داده­های خوشه­ای را مدیریت کند. ما قادر هستیم طیف گسترده­ای از انواع داده­ها را مدیریت کنیم و همچنین قادر هستیم زمان استفاده از منابع پرس­وجو را بهینه کنیم. ما آزمایش­هایی را که در مجموعه داده­های متعدد انجام شده­اند را از طریق LUBM ایجاد می­کنیم.

**فهرست شرایط-** روش­های خوشه­بندی، محاسبات توزیع شده، مدیریت اطلاعات، بهینه­سازی، مقیاس­پذیری

**1. مقدمه**

ایجاد علومی از داده­ها با چالش­های بسیاری مواجه بوده است. یک مسئله اصلی وجود دارد و آن این است که امروزه کلان داده­ها، پویا و ناهمگن هستند، و چندین منابع را که اغلب ساختار استانداردی ندارند را جمع­آوری می­کنند.

اکثر تجزیه و تحلیل داده­های مدرن، ابزارهای مدیریت و سرویس را طراحی می­کنند و در سیستم فایل توزیع شده هادوپ (HDFS) به عنوان یک انبار داده­ای مورد استفاده قرار می­دهند؛ گاهی اوقات هم این ابزار تحلیلی از سرویس­هایی که توسط اکوسیستم هادوپ پردازش شده­اند استفاده می­کند. هادوپ از لحاظ هزینه و عملکرد بسیار خوب است.

قابلیت انعطاف­ هادوپ، مقیاسی را روی مسائل مدیریتی داده­ها ارائه می­دهند و در آن کاربران به صورت ناکارآمد کار انجام می­دهند [1]. هونگ و همکارانش (a) راهکارهایی که کاربران به منظور اضافه کردن ماشین­ها برای غلبه بر محاسبات انجام می­دهند نحوه تمرکز و استفاده از کدهای منابع کاربران را به ما نشان می­دهد، و (b) بسیاری از کاربران HDFS معتقدند که این کدها برای پردازش دسته­ای طراحی شده­اند. از این رو، خوب است که کدها برای مدت طولانی در پس­زمینه بکار برده شوند و حتی تصور هم نمی­کنند که منابع مورد استفاده این فرآیندها هستند.

در باجدا پاولیکوفسکی و همکارانش روی هاداپ کار کردند [2]، و نویسندگان نمونه­ای از چنین ناکارآمدی را ارائه دادند و داده­های ساخت یافته را 50 بار برآورد کردند. با این حال، انفجار اقدام مهم داده­های نیمه ساخت یافته، به صورت چندگانه و ساختارنیافته بر طبق بررسی­های مایکل واکر است که در وبلاگ به آن اشاره شده است [3]. شرکت­ داده­های بین­المللی (IDC) برآورد کرده است که حجم داده­های دیجیتال از 40 تا 50 درصد در سال رشد خواهد کرد [4]. تا سال 2020، IDC [4] پیش­بینی کرده است که این تعداد به 40 زتابایت خواهد رسید. تا سال 2020، جهان 50 بار مقادیر داده­ای و 75 بار محتوای داده­ها را تولید خواهد کرد. نیازهای سختی برای ابزارهای تجزیه و تحلیل داده فعلی به منظور اندازه­گیری کلان داده­ها وجود دارد و آنها به طور موثر از منابع پردازش استفاده می­کنند.

روهلوف و همکارانش در سال 2011 [5] نحوه ذخیره داده­های گراف در هادوپ را با استفاده از نمای سه گانه توضیح دادند. آنها همچنین نشان دادند که چطور تطبیق الگوی زیر نمودار در یک نمودار مقیاس­پذیر انجام می­شود. اگرچه تمرکز بر روی گراف­های وب معنایی بود، اما تکنیک­هایی که در این مقاله ارائه شده بود انواع نمودارها را تعمیم می­داد. سیستم SHARD نتیجه این مقاله بود. از این رو، تکنیک­های هادوپ پشتیبانی می­شوند تا ظرفیت تطبیق الگوریتم زیر نمودار ترسیم شود.

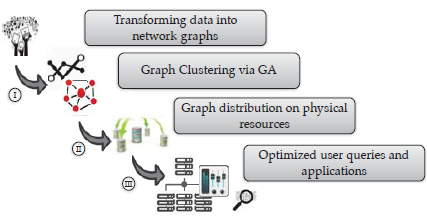
در سال 2011، هوانگ و همکارانش مقیاس پرس­وجو SPARQL و گراف­های RDF [1] را که به طور کارآمد در تکنیک­ها توسط روهلوف و همکارانش بیان شده بودند را نشان دادند [5]. هوانگ و همکارانش [1] 1340 عامل را که کمتر از تکنیک­های کارآمد است را معرفی کرده­اند [5]. و از سایر تکنیک­های جایگزین برای پردازش الگوریتم­های زیر که براساس هادوپ هستند را استفاده کردند.

در برخی موارد، راه حل­های کلان داده از HDFS به عنوان یک ابزار ذخیره­ساز استفاده نمی­کنند. با این حال، آنها از همان روش مقیاس­پذیری افقی استفاده می­کنند. ما راه حل­هایی را که بر روی HDFS کار انجام می­دهند را آزمایش و ارائه دادیم. نمونه­هایی از ابزارهای ذخیره­ساز HDFS به عنوان آپاچی اسپارک [6] و مسوس عمل می­کنند [7]. سیستم هادوپ از طریق منابع YARN و منبع داده HDFS، و HAMR پشتیبانی می­شود [8]. از این رو، بهینه­سازی HDFS با چارچوب HDFS ارائه می­شود و منجر به بهینه­سازی تعداد زیادی از راه حل­های کلان داده­ها می­شود.

اسپارک [6] و استورم [9] جدیدترین ابزار کلان داده­ها را ارائه کردند. آپاچی استورم [9] یکی دیگر از راه حل­های داده­ای است که از YARN برای تجزیه و تحلیل داده­های نامحدود استفاده می­کرده­اند. استورم هادوپ را ایجاد می­کند و به صورت دسته­ای آن را پردازش می­کند. برخی از تلاش­ها در بهینه­سازی Storm ایجاد می­شوند. یک مثال از زمانبندی بهینه­سازی در [10] یافت شده است، و با افزوده شدن Storm به آن در [11] ارائه شده است. تمرکز ما در این بررسی بیشتر روی بهینه­سازی انبار داده­ای HDFS است (در حال حاضر داده­ها در HDFS قرار دارند). با این حال، در برخی از موارد، Strom [9] جریان داده­ها را پردازش می­کند و نتایج خروجی را به منظور پردازش بیشتر در HDFS ذخیره می­کند، تا جایی که رویکرد ما امکانات تحلیلی داده را در بسته­های هادوپ انجام دهد.

رشد داده­های نیمه/چند/غیر ساختاری و توانایی آنها در پردازش داده­ها به طور موثر، یک مسئله اصلی بود، و هدف اصلی این داده­ها بهبود کارایی سیستم فایل توزیع شده هادوپ (HDFS) به منظور بررسی داده­های مدرن و بهبود منابع HW بود. با وجود اینکه تکنیک­های SHARD [5] و SPALQL [1] مقیاس­پذیر و قابل تعمیم هستند، اما محدودیت­هایی در مورد نحوه مقابله با داده­های مدرن نیز وجود دارد. علاوه بر این، محدودیت­های دیگری در مورد نحوه مدیریت جریان تغییرات پویای داده نیز وجود دارند. همانطور که هاجر و همکارانش بحث کرده­اند یکی از این محدودیت­ها ممکن است به کاربرد مقیاس­پذیری و ذخیره­سازی الگوریتم خوشه­بندی که در چنین کارهایی مورد استفاده قرار گرفته­اند اضافه شوند [12].

ما داده­ها را تغییر دادیم و آنها را در گراف براساس مقیاس گرافیک ذخیره­سازی کردیم تا ساختار آن ایجاد شود و بتواند تغییرات ضروری را انجام دهد، و سپس ساختار سه­گانه را برای نقاط داده، به منظور ایجاد خوشه سه­گانه که در بخش معماری به طور کامل شرح داده شده است اضافه می­کند. این تکنیک­ها به ما این امکان را می­دهند که (1) داده­ها را از منابع مختلف جمع­آوری و سپس آنها را با وجود ساختار داده­های مختلف به چهارگوش تبدیل می­کنیم، (2) تغییرات جریان پایگاه داده گراف به صورت پویا می­باشند، و (3) هاجر و همکارانش یک نسخه جدید از کاربرد داده­ها را جمع­آوری کرده­اند [12]. رمزگذاری جدید کروموزوم­ها به منظور بررسی مسائل خوشه­بندی داده­های مدرن به همراه تکنیک­های نوین متقاطع، جهش و ارزیابی برای ارائه نیازمندی­های تکنولوژی جدید رمزگذاری توزیع شده مورد استفاده قرار گرفته­اند. بعدها، ما زیرنمودارهای موجود در HDFS را براساس وابستگی خوشه به منظور تولید پردازش داده­های بهینه­سازی پرس­وجو توزیع کردیم.



شکل 1: مراحل چهارچوب ارائه شده محاسبات

شکل 1 مفاهیم و ماژول­ها را در یک چارچوب پیشنهادی نشان می­دهد و توضیح آن به شرح زیر است: (1) پس از جمع­آوری داده­ها یا جمع­آوری مجموعه داده­های قدیمی، این ماژول داده­ها را به نمودارهای شبکه مورد نظر تبدیل می­کند؛ (2) با یافتن الگوها در نمودار، ماژول داده­ها را به بلوک­های مناسب توزیع می­کند؛ (3) بلوک­ها در داخل ماشین سمت راست توزیع می­شوند؛ و (4) HDFS بهینه­سازی شده به عنوان یک منبع داده پرس­وجوها را اجرا می­کند و یک سیستم­عامل را برای اعمال الگوریتم­های گراف به صورت کارآمد به منظور کاهش منابع مورد استفاده ارائه می­کند.

برای خلاصه کردن چارچوب پیشنهادی، HDFS قادر است داده­های مدرن را با ایجاد ساختار اطلاع­رسانی داده­ها، شناسایی، توزیع و مدیریت داده­ها در سیستم فایل مقیاس­پذیر را بهبود ببخشد. بدین ترتیب، چارچوب نتایج در بهینه­سازی و در منابع کارآمد سیستم اکو هادوپ و دیگر ابزارها از HDFS به عنوان یک منبع ذخیره­سازی توزیع شده استفاده می­شود.

در تحقیقات اخیر راه حل­های متقاعد کننده­ای در تحلیل بعدی و معماری لامبدا ارائه شده است. سانگ و همکارانش [13] تحقیقات اخیر را در نوع داه­ها، مدل­های ذخیره­سازی، روش­های تجزیه و تحلیل کاربرد شبکه کلان داده­ها مورد بررسی قرار دادند. آنها همچنین چالش­ها و توسعه کلان داده­ها را برای پیش­بینی روند فعلی و آینده خلاصه کرده­اند. سانگ و همکارانش [13] نحوه جریان داده­ها در زمان واقعی را با افزایش سرویس­های جریان آنلاین نشان دادند، و همچنین نحوه بررسی یک سیستم را که براساس SQL که جریان DB نام دارد به منظور تجزیه و تحلیل اطلاعات پایدار نشان دادند.

بررسی­های [15]، [16]، [17] و [18] بر تحلیل زمان واقعی تمرکز دارند، و شامل تلاش­های معماری لامبدا هستند، و نویسندگان [18] در آن معماری سیستم­های کلان داده­های بحرانی را معرفی می­کنند. آنها نحوه زیرساخت­های موجود در کلان داده­ها را که زمانبر هستند نشان دادند و فقط بر روی برنامه­های کاربردی تمرکز کردند. باسانتا و همکارانش [18] این مسئله را از دیدگاه جامعه سیستم­های زمان واقعی مورد بررسی قرار دادند. آنها معماری تحلیلگر زمانبندی (TC) را به عنوان یک گروه TC در پردازش دستی و جریان پردازش آنلاین TC را در نظر گرفتند. باسانتا و همکارانش [18] به طور کلی تبدیل پشته کلان داده­ها را در پشته کلان داده TC به همراه اعمال نیازمندی­ها و چالش­های برنامه­های پشته TC ارائه دادند.

T-Hoarder [19] چارچوبی است که توییت­ها را همراه با داده­های مربوط به زمان فضا را به دست می­آورد؛ همچنین خلاصه­ای از اطلاعات و تحلیل مربوط به فعالیت توییتر را به صورت یک رویداد در صفحه وب ارائه می­کند. تجزیه و تحلیل بررسی­ها و معماری لامبدا همراه با آپاچی کُدو [20] و مجموعه­ای از بررسی­ها را که عملکرد سریع­تری در پردازش OLAP و عملکرد قوی در زمان بارز حجم کاری را دارد نشان می­دهد. با این حال، این یک نتیجه است، تا جایگاه برخورد داده­های هوشمند، و تکنولوژی­ها بتوانند حجم کاری پردازش داده­ها را کاهش دهند، و اولین نسخه آزمایشی اینتل را در سفیر 2017 ارائه دادند [22]. اینتل نوع جدید حافظه پایدار را با یک فضای خاص نسبت به DRAM و تاخیر کمتری را نسبت به SSD ارائه می­دهد.

**2 محدوده کاری**

حجم کاری فعلی که در سیستم در حال اجرا می­باشد (در آن ناکارآمدی وجود دارد) منجر به فضای بیشتر نیازمندی در مراکز داده و برخی از پیامدهای محیطی و باعث افزایش انتشار کربن در مصرف انرژی می­شود [1]. این می­تواند به دلیل مصرف انرژی اضافی و عملکرد پایین منابع سخت­افزاری که شرکت­ها را تحت تاثیر قرار می­دهد باشد. ما به یک مقیاس کارآمد نیاز داریم.

نظریه گراف یک اصول را به خوبی مورد بررسی قرار می­دهد. دکتر روی مارستن در وبلاگش [23] خاطرنشان کرد که نظریه گراف یکی از روش­های کلیدی در فهم و درک استفاده از کلان داده­ها است. دکتر مارستن رو ی گوگل در طی فرآیند مدرن­سازی گراف با استفاده از لینک­های بین اسناد و زمینه­های وب معنایی، تمرکز دقیقی داشت. در نتیجه، “گوگل یک موتور جستجوی وب را تولید کرد و از رقبای ثابت خود پیشی گرفت” [23]. بسیاری از داده­ها را می­توان به نمودار تبدیل کرد. برعکس، بسیاری از مسائل را می­توان به مسائل گراف تبدیل کرد. با استفاده از نظریه­ها و الگوریتم­های گراف، بسیاری از این مسائل را می­توان به طور موثر حل کرد.

ما کار SHARD روهلوف و همکارانش را در نظر گرفتیم [5] و آن را همانطور که در بخش معماری نشان داده شده است به عنوان داده­های امروزی به منظور تبدیل انواع داده به نمودار چهارگانه ارائه کردیم. همانطور که در بخش­های پایگاه داده گرافیکی بحث کردیم روش­هایی برای مقیاس داده­های گراف با استفاده از سیستم­های اکو توزیع شده مانند هادوپ وجود دارد. علاوه بر این، موفقیت بزرگ هوانگ و همکارانش زمانی اتفاق می­افتد [1] که بورس تحصیلی روهلوف و همکارانش مورد پذیرش قرار گرفته باشد [5] و داده­های RDF برای غلبه بر محدودیت­های مربوط به تکنیک روهلوف و همکارانش بهینه­سازی می­شوند.

قبلا تلاش­هایی برای بهینه­سازی گراف­ها صورت گرفته است؛ به عنوان مثال SHARD [5]، عمل درهم­سازی داده­ها را انجام می­دهد. با این حال، عمل درهم­سازی منجر به جابه­جایی محدودیت­ها در گراف­های RDF و حرکت داده­های شبکه­ای نیز می­شود. مثال دیگری از هوانگ و همکارانش بدین صورت است [1] اشیاء متصل به یک موضوع پردازش می­شوند تا یک بلوک مشابه برای یک یا دو تقاطع بین موضوع و جسم ایجاد شود (یک یا دو لبه فاصله بین موضوع و اشیاء). با این حال، محدودیت­های فضا به علت افزایش اندازه داده­ها وجود دارد. همچنین، محدودیت استفاده از چنین الگوریتمی که به یک گراف متصل است نیز وجود دارد. سایر کارهایی مانند سمپالا [24]، یا Hive، PigSPARQL [25] & [26]، MapMerge [27] و MAPSIN [28] تا حدی مقیاس­پذیری را برطرف می­کنند. با این وجود، چنین کاری از ذخیره­سازی­های مختلف که سه برابر هستند و از مزایای Hive، MR و Impala استفاده می­کند و با استفاده از چارچوب ما می­توانیم پارتیشن­بندی را بهینه­سازی کنیم.

**2.1 پایگاه داده­های گراف**

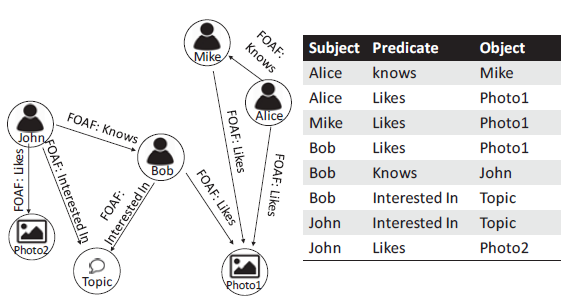
داده­های جاری و برنامه­های مدرن به محدودیت­های ذخیره و پردازش با استفاده از پایگاه داده­های سنتی، بویژه مدل ارتباطی منجر می­شوند. دقت پایگاه داده­های گراف افزایش پیدا کرده است، و این موضوع تقریبا در اوایل دهه 90 [29] مجددا مورد بررسی قرار گرفت. اهمیت پایگاه­های اطلاعاتی معمولا براساس روابط بین داده­ها است، و به طور مساوی یا حتی بیشتر از اطلاعات موجودیت­ها می­باشند [30]. پروژه­ها در زمینه­های مختلف به پایگاه­های زیر دقت ویژه­ای دارند (مانند زیست­شناسی [31]، وب معنایی [32]، وب کاوی [33] و شیمی [34]).

سیلبرشاتس و همکارانش [35] به طور کلی یک مدل داده (مدل پایگاه داده) را که مجموعه­ای از ابزارهای مفهومی است را برای مدل­سازی روابط بین شناسه­ها مورد استفاده قرار دادند. سه مولفه مدل پایگاه داده عبارتند از [35]: (1) مجموعه­ای از انواع ساختار داده­ای، (2) مجموعه­ای از قوانین یکپارچه و (3) رابط قوانین و عملگرها.

رنزو در هر مورد، “مدل­های پایگاه داده گراف را می­تواند به عنوان یک ساختار داده برای طرح و نمونه­ها را به عنوان یک نمودار تعمیم یافته، و دستکاری داده­ها را به عنوان عملیات گراف بیان کند” و آنها را مشخص می­کنند [30].

بسیاری از پایگاه داده­های گراف در زمینه­های تحقیقاتی و صنعتی توسعه پیدا کرده­اند. برخی از تفاوت­ها مانند DEX، AllegroGraph، Neo4J، Infinite Graph، HypergraphDB و Sones در [3] یافت شدند. همانطور که هوانگ و همکارانش توضیح داده بودند بعضی از آنها حتی برای فرمت داده­های سه­گانه نیز طراحی شده بودند [1].

ذخیره­سازی سه­گانه یا RDF یک پایگاه داده براساس گرافیک و بازیابی سه­گانه از طریق پرس­وجوهای معنایی می­باشد. موجودیت سه­گانه داده از شیء subject-predicate-subject تشکیل شده است. ذخیره­های سه­گانه داده­ها به صورت سه­گانه نشان داده می­شوند و از طریق زبان پرس­وجو بازیابی می­شوند؛ با این حال، برخی از تفاوت­های کلیدی پایگاه­های داده رابطه­ای نیز وجود دارند، و به طور عمده ذخیره­های سه­گانه را بهینه­سازی می­­کنند.



شکل 2: ذخیره RDF سه­گانه و نشان دادن آن به عنوان گراف

ذخیره سه­گانه در سه­گانه در شکل 2 نشان داده شده است. در برخی از تحقیقات توسعه­هایی در رابطه با سیستم پایگاه داده خوشه RDF صورت گرفته است. در حال حاضر پایگاه داده­های خوشه­ای RDF قابل دسترس هستند، و شامل YARS [36]، SHARD [5] و Jena [37]، Elephas و Virtuoso [38] هستند، و معمولا درهم­سازی سه­گانه را در گره­های محاسباتی متعدد و گره­های پرس­وجو را در زمان به صورت موازی پارتیشن­بندی می­کنند.

**2.2 تشخیص جامعه و الگوریتم تکاملی چند هدفه**

در نظریه گراف، خوشه­ها اغلب زمانی که معیارهای خاصی از تراکم و ضخامت توسط یک الگوریتم بهینه­سازی می­شوند به صورت الگوریتم تعریف می­شوند، و نتیجه حاصل از آن تقسیم شبکه به جوامع می­شود [39]. در بسیاری از موارد، بهینه­سازی اندازه­های مسائل NP-hard را پیدا می­کند. معمولا، الگوریتم به صورت تقریبی از حل مسائل NP-hard استفاده می­کند. یکی از الگوریتم­های فراابتکاری برای حل تقریبی مسائل NP-hard الگوریتم تکاملی (EA) است که در [40] توضیح داده شده است. برنامه استفاده از GA (الگوریتم ژنتیک) و تشخیص جامعه در [41]، [42]، [43]، [44]، [45] و [46] و کاربرد EA در [47] و [48] توضیح داده شده است. چانگ و همکارانش [49] برنامه کاربردی بهینه­سازی کلونی را در تشخیص جامعه شرح داده­اند.

مدولاسیون [50] یکی از محبوب­ترین معیارهایی است که برای نتیجه­گیری کیفیت خوشه­بندی غیرتداخلی گراف، بویژه در جامعه تجزیه و تحلیل شبکه مورد استفاده قرار می­گیرد [42]، [43]، [44]، [45]. مسئله کشف خوشه­بندی با حداقل مدولاسیون NP-Complete نام دارد [51]. در نتیجه، بسیاری از الگوریتم­های چندجمله­ای اکتشافی زمانه توسعه پیدا کرده­اند [52]، [53]، [54]، [55].

یکی از الگوریتم­های تشخیص جامعه، الگوریتم Girvan-newman است [56]، و دارای لبه­هایی با حداکثر مرکزیت است و به طور پیاپی از شبکه حذف می­شود تا جایی که هیچ لبه­ای باقی نماند. حالت مدولاسیون می­تواند به صورت معادله (1) تعریف شود [57]، nc تعداد کل جوامع، m تعداد لبه­ها در گراف، li تعداد کل لبه­ها در داخل جامعه i، di برابر مجموع تمام گره­های i است.

(1)

یی و همکارانش [58] “در الگوریتم ژنتیک (GA)، جمعیت کروموزوم، کدگذاری راه حل­های انتخابی/مسائل بهینه­سازی فردی، و راه حل­های بهتر را تکامل داده­اند”. سپس، راه حل ژنتیک در فرمت کروموزوم و در فرمول­های کروموزومی ارائه می­دهد، و GA به صورت تصادفی/قطعی راه­اندازی می­شود. سپس، GA قصد دارد آن را از طریق برنامه­های تکراری و چندین عمل ژنتیک مانند انتخاب، تقاطع و جهش بهبود ببخشد. سرانجام، جستجوهای محلی و عملگرهای جستجوی مرزی برای اعمال نتایج به خوبی مورد استفاده قرار می­گیرند.”

لی و سانگ [46] یک الگوریتم ژنتیک فشرده را توسعه دادند. ما کارهای قبلی خود را با استفاده از GA از [12]، [59] و [60] به دلایل مختلف انتخاب کردیم: (1) فضای جستجو بزرگ، (2) NP-Completeness حداکثر مدولاسیون [51]، (3) قادر است مقیاس نمودارها را بزرگ کند، و همچنین (4) قادر است تغییرات منابع مختلف داده­ها را به صورت پویا منعکس کند. با این حال، برخی از تغییرات در GA [12] برای سازگاری چارچوب پیشنهادی HDFS لازم و ضروری بود. در بخش معماری، ما الگوریتم ژنتیک توزیع شده خود را با استفاده از منابع باز بهینه­سازی چند هدفه و الگوریتم­های ژنتیک، Jmetal [61]، و Apache Jena Elephas [37] به منظور ذخیره و مدیریت داده­های RDF توسعه دادیم. بعدها، ما مجموعه­ای از داده­های شناخته شده و به همراه آن گراف RDF بزرگ را که توسط آزمایشات دانشگاه تولید شده بود را تایید کردیم (LUBM).

**2.3 سرویس توسعه HDFS**

همانطور که قبلا ذکر شد، HDFS به عنوان یک منبع داده توزیع شده راه­حل­های داده­های مدرن بزرگ مانند HAMR [8]، Mesos [7]، آپاچی اسپارک [6] و صدها نفر دیگر را توسعه می­دهد. چنین راه حل­هایی دارای بسیاری از برنامه­های توسعه یافته HDFS یا دارای بیش از یک سرویس هستند که در HDFS در حال اجرا می­باشند (به شکل 3 نگاه کنید).



شکل 3: توسعه سرویس­ها و برنامه­های کاربردی HDFS.

**3 عملکرد و کارآیی HDFS**

از سیستم اکو هادوپ برای پردارش سازمان داده­ها و ایجاد برنامه­های کاربردی بر روی آن، که بستگی به موارد و سازمان داده­ها دارد استفاده می­شود. گروهی از IT BI (کسب و کار هوشمند) در کسب و کارها و موسسات چنین سیستم­هایی را به منظور رسیدن به اهداف خود پیکربندی می­کنند، و سپس روی داده­ها و موارد مورد استفاده در آنها تمرکز می­کنند.

اکثر سازمان داده­ها برای موارد خاص جمع­آوری و استفاده می­شوند. بعدها، این داده­ها که در انتظار ذخیره تیم BI به منظور ایجاد و استفاده از آنها هستند، نتایج موجود در داده­ها جمع­آوری می­کنند و در منابع مختلف با ساختار چندگانه مورد استفاده قرار می­گیرند.

هوانگ و همکارانش [1] و روهلوف و همکارانش [5]، اجرای هادوپ و سرویس­هایی که برای اجرای HDFS که دارای عدم بهینه­سازی در گراف بودند را طراحی کردند. برخی از علل ناکارآمدی HDFS عبارتند از [1]: (1) عمل پارتیشن درهم­سازی به طور پیش فرض توسط هادوپ ممکن است داده­های مرتبط را هدایت کند تا محاسبات آنها به صورت فیزیکی بیش از مجموعه­ها به پایان برسد، و این عمل به طور موثر موجب انتقال گسترده داده­ها بین منابع می­شود تا عملیات گراف به پایان برسد. بدین ترتیب، ترکیب داده­های مربوطه در هر [1] حاصل می­شود [1]؛ (2) هادوپ اهمیت تمام بلوک­های داده و پارتیشن­ها، و همچنین حفظ محدوده همسایگان را بین خوشه و از لحاظ فیزیکی بهبود بهره­وری را در نظر می­گیرد، و (3) HDFS داده­های گراف را بهینه­سازی نمی­کند.

هوانگ و همکارانش [1] یک مسئله کارایی با تکنیک گفته شده روهلوف و همکارانش در سیستم مبتنی بر هادوپ نشان دادند [5]. با این حال، شیوه­ای که هوانگ و همکارانش ارائه داده بودند [1] روهلوف و همکارانش [5] روی آن شیوه کار انجام دادند تا بتوانند سراسر مسئله را تعمیم ببخشند. آنها روی یک نوع فایل خاص و یک الگوریتم خوشه­بندی ساده تمرکز می­کنند، و ما معتقدیم که این روش دارای برخی مشکلات زمانی در کلان داده­های پویا ساخت یافته/ نیمه ساخت یافته/ساخت یافته چندگانه است و با این مشکلات نیز مقابله می­کند. در بررسی قبلی، هاجر و همکارانش [12] ما محدودیت­هایی را تایید کردیم. این بررسی نشان داد که چگونه الگوریتم­های ژنتیک برای خوشه­بندی چنین داده­هایی استفاده می­شوند. ما از این روش در تکنیک هاجر و همکارانش [12]، Pizzuti [43] و [42] همراه با لیستی از کارهای دیگر مانند [48]، [46]، [62] و [63] پس از ساخت یک روش تبدیل به منظور تبدیل اطلاعات مورد نظر گراف داده­ها استفاده کردیم. نتایج به دست آمده از توسعه کار در [12] به منظور تعمیم هوانگ [1] و همکارانش [5] مورد استفاده قرار گرفتند.

در هادوپ، ایده اصلی این است که محاسبات را به داده­ها برسانیم؛ به عنوان مثال، نگاشت کاهش، قسمت نگاشت می­تواند به سرعت محاسبات را در یک ظرف که بخشی از منبع داده می­باشد انجام دهد. از سوی دیگر، مرحله کاهش اطلاعات داده­ها را از خروجی­های نگاشت­ها جمع­آوری می­کند و در بیشتر موارد، بخش­هایی از داده­ها باید از طریق شبکه به ظروف مناسب که برخی از گیرنده­های خاص را اجرا می­کنند انتقال داده شوند. خروجی چنین نگاشت­هایی معمولا از چندین نگاشت جمع­آوری شده به دست می­آید. برنامه­نویسان باید این گونه مسائل را تا حدی کنترل کنند. به همین دلیل است که حتی در آموزش کلودرا توسعه­دهندگان تمایل دارند دستورالعمل­هایی را برای برنامه­نویسان به منظور استفاده از نگاشت به جای استفاده از ردیف­ها ارائه دهند. آموزش کلودرا همچنین توسعه­دهندگان را تشویق می­کند تا از زمانبندی پرس­وجو [64] استفاده کنند تا زمانی که بتوانند از چنین اتصالاتی محافظت کنند. از این رو، ما به شدت بر این باوریم که خوشه هادوپ باید یک حس آگاهی از داده­ها را داشته باشد، در حالی که داده­های خروجی نگاشت برای همان کلید باید حداقل تا آنجا که امکان دارد یک دستگاه در مجاورت آن باشد.

از آنجا که برنامه­های کاربردی و مشاغل متفاوت از یک usecase دیگر هستند، تمام موارد آینده را پوشش نمی­دهند. یک راه حل برای بهینه­سازی چنین مشاغلی، جمع­آوری داده­ها به عنوان یک چارچوب پیشنهادی است. تمام الگوریتم­ها در چنین مواردی پوشش داده نمی­شوند، اما اکثر الگوریتم­های گراف به طور خاص به داده­هایی که مرتبط هستند متصل می­شوند (به عنوان مثال پیمایش گراف).

**4 شاخص­های آگاه داده­های HDFS**

**4.1 تبدیل گراف**

ما در مقدمه منابع و مسائل مدرن داده­ها را مورد بحث قرار دادیم. و بیان کردیم که داده­های ابر از منابع مختلف S=(S1,S2,…,SZ) ناشی می­شود که SN منبع Z و Z ≥ 1 است. این منابع مختلف تولید یا حاوی داده­ها با ساختارهای مختلف می­باشد،اما گاهی اوقات برای اشخاص مشابه، و برای داده­ها و ساختارها مختلف می­باشد. D=(DS1,DS2,…,DSZ)، که در آن DSZ یک ساختار داده که از منبع Z ناشی می­شود و شامل مجموعه بی­پایان DS (ساختار) DSZ است.

داده­ها با ساختار در یک گراف G(V,E) به عنوان یک گراف غیرثباتی و با تعداد راس­های |V|=m و تعداد لبه­های |E|=n تبدیل می­شوند. این تغییر و تحول در بخش معماری به طور کلی توضیح داده شده است.

**4.2 خوشه­بندی گراف**

ما به یک گراف از رئوس V و یال­های E به عنوان G(V,E) اشاره کردیم. همچنین، تعداد رئوس |V|=m، و تعداد یال­های |E|=n و خوشه C=(C1,C2,C3,…,CJ) به عنوان پارتیشن V مجموعه­­هایی را بهم متصل می­کنند. ما C را که خوشه G است و شامل خوشه­های J نیز می­شود را فراخوانی می­کنیم. تعدادی از خوشه­های j زمانی که C شامل فقط زیرمجموعه C1=V دارای حداقل j=1 است و زمانی که هر خوشه Cj شامل فقط یک رئوس است دارای حداکثر j=m است. ما خوشه C1 را به عنوان گراف G تشخیص دادیم. گراف G[Cj]=(Cj,E(Cj))، که بدین صورت است E(Cj)={{WW}={E:WWCj}}. سپس، E(C)=UJ=1E(Cj) مجموعه داخلی یال­های خوشه و E/E(C) مجموعه خارجی یال­های خوشه است. تعدادی از یال­های خوشه داخلی دلالت بر m(C) دارد و به تعدادی از یال­های خوشه خارجی دلالت دارد.

در الگوریتم خوشه­بندی، ما از یک ماژولار به عنوان یک اندازه شایستگی در روش هاجر و همکارانش استفاده کردیم [12]. سپس ماژول Q به صورت کسری از لبه­هایی که در گروه 1 یا 2 قرار می­گیرند، تعریف می­شود و منهای تعداد لبه­های مورد انتظار در گروه­های 1 و 2 برای یک گراف تصادفی با همان توزیع درجه گره به صورت شبکه می­شود. از این رو، تعداد واقعی یال­ها بین v و w منهای تعداد مورد انتظار بین آنها برابر Avw-(KvKw)/2m است. ماژولار می­تواند به صورت معادله زیر بدست آید (2) [57]:

(2)

به این نکته توجه داشته باشید که معادله (2) فقط دو گروه شبکه را پارتیشن­بندی می­کند. برای شناسایی جوامع متعدد در یک نمودار؛ فرمول باید به صورت زیر تعریف شود (3) [57]:

(3)

در این فرمول eij کسری از لبه­ها با آخرین راس­های جامعه i و دیگری در جامعه j است (4) [57]:

(4)

و ai کسری از یال­های پایانی است که رئوس را در جامعه iام به عنوان یک معادله به هم پیوند می­دهد (5) [57]:

(5)

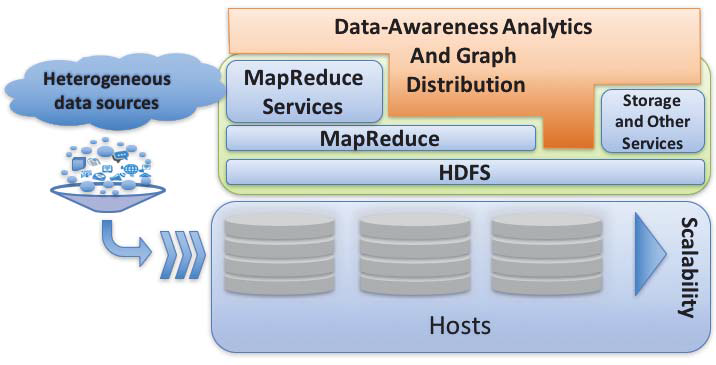
**4.3 فرضیات و توزیع گراف**

همانطور که قبلا توضیح داده شد، سه چالش عمده با بهینه­سازی HDFS مواجه شدند؛ و دو نفر از آنها در مورد نحوه هشدار هادوپ و توزیع داده­ها توضیح دادند. فرضیات و آزمایشاتی را که انجام دادیم عبارتند از (1) ذخیره داده­های درون خوشه­ای در یک ماشین واحد و (2) ذخیره داده­های میان خوشه­ای که نزدیک ماشین­ها بودند و گامی بزرگ در جهت بهینه­سازی HDFS برداشتند. M=(M1,M2,…,MI) یک مجموعه از ماشین­هایی است که منابع محاسباتی هادوپ را نشان می­دهد، و I متعلق به تعداد مجموعه شایستگی نیز است. و Mj، Mj+1، Mi+n ماشین­هایی هستند که به صورت فیزیکی فاصله شبکه آن بین Mi و Mi+1 نزدیکتر از Mi­ و Mi+n است. خوشه Cj با تمام رئوس­ها و یال­ها باید در همان گراف پارتیشن یا حداقل باید در همان ماشین Mi جایگزین شود. زمانی که در سمت چپ Mi وجود ندارد باید کمترین مقدار Mi+1 و غیره جایگزین شود، و نزدیک­ترین ماشین، بهترین نتیجه را ارائه می­دهد. Cj و Cj+1 بیشترین یال­های خوشه میانی نسبت به Cj و Cj+n را دارد سپس، Cj و Cj+1 باید در کوچکترین مقدار احتمالی m جایگزین شود، و بازه M، و Mi+m (نزدیک­ترین ماشین فیزیکی) بین 0≤m≤1 است.

**5 کل معماری**

چارچوب خوشه­بندی ما (DEGA-Gen) بخشی از داده آگاهی ارائه شده است که در ماژول روی بالاترین داده ذخیره­سازی توزیع شده در حال اجرا می­باشد و در شکل 4 نشان داده شده است.

این چارچوب با HDFS و سرویس­هایی که قابل دسترس هستند ارتباط برقرار می­کند تا بتواند خوشه­های به­روز شده را به عنوان جریانی از داده­ها در HDFS ارائه کند. هدف ما دستیابی به بهینه­سازی به همراه قرار دادن داده­های مرتبط و کاهش سربار در حرکت داده­ها بین میزبان است. انتقال داده­ها عمدتا در فرآیندهای تجمعی یا پیوند رخ می­دهند.



شکل 4. ماژول آگاهی داده­ها و الگوریتم خوشه­بندی تکاملی توزیع شده که بخشی از هادوپ است.

**5.1 ساختار توزیع شده گراف­های RDF**

گام اول چارچوب داده­ای HDFS، به یک گراف RDF توزیع شده تبدیل می­شود. این روند با استفاده از منابع باز Apache Jena و Apache Jena Elephas انجام می­شود. داده­های ارائه شده HDFS مجموعه­ای از داده­ها را چهار برابر افزایش می­دهد؛ و دلایل این فرآیند در خوشه­بندی براساس ژنتیک توضیح داده می­شوند. برخلاف استفاده گسترده از عمل چهارگانه، ما از فیلد اضافی به صورت چهارگوش استفاده کردیم و وابستگی خوشه سه­گانه را در گراف چهارگانه نشان دادیم؛ و آن را فیلد نامیدیم (کروموزوم ID). این فرآیند استفاده از ذخیره­های چهارگانه را برای بهبود فرآیند فرعی (رمزگذاری و نمایش خوشه­بندی توزیع شده) به کار می­رود.

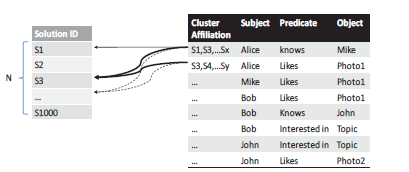
تمام انواع داده­ها را نمی­توان به گراف تبدیل کرد. با این حال، کلان داده­ها در مورد use-caseها و برخی داده­های use-case هستند. این یک تغییر مستقیم (SHARD)، و اسناد پایگاه داده ارتباطی است (به عنوان مثال) و می­توان آن را به عنوان یک گراف RDF نشان داد که هر ویژگی از آن نشان دهنده رکورد هر رابطه بین دو گره است، و یک گره نشان دهنده مقدار فیلد کلیدی و گره دیگر مقدار نشان دهنده فیلد ویژگی است. در موارد دیگر، (مانند داده­های متنی)، روابط بین نقاط داده می­توانند براساس use case تعریف شوند. به عنوان مثال، پردازش زبان طبیعی را می­توان برای هر رابطه (کلمه پیش از آن) و (کلمه پس از آن) برای هر کلمه به منظور ایجاد مدل پیش­بینی تعریف کرد.

**5.2 خوشه­بندی مبتنی بر ژنتیک**

**5.2.1 بیان رمز­گذاری**

ما از کدگذاری [65] برای غلبه بر مسائل کدگذاری بزرگ در مطالعات قبلی که در [65] ذکر شده بود استفاده کردیم. چنین رمزگذاری از تعریف خوشه­ها حاصل می­شود.

با این حال، حتی با چنین کدگذاری در [65]، راه حل­ها هنوز هم می­توانند به صورت طولانی­تر به عنوان یک مقیاس داده­ای بیان شوند. در نتیجه، کلاینت GA در خارج از حافظه به صورت دستی راه حل­ها را به عنوان یک مقیاس داده اجرا می­کند. از سوی دیگر، ما از تکنیک ذخیره­سازی به عنوان اطلاعات اضافی به همراه گراف سه­گانه HDFS، و به همراه پوشش نقاط داده­ای <Node><Predicate><Node>triples، روهلوف و همکارانش [5]، into<chromosome\_part><Node><predicate><Node>quadruples استفاده می­کنیم. ما به <chromosome\_part> به عنوان فهرست solution\_Ids که متعلق به جمعیت گره خاص می­باشد اشاره کردیم. این کدگذاری به جمعیت فهرستی از اعداد صحیح با اندازه ثابت در کلاینت GA که solution\_IDs نام دارد منجر می­شود. این تکنیک این امکان را فراهم می­کند که کلاینت بتواند خوشه GA را روی مجموعه داده­ها با اندازه بزرگ مقیاس­بندی کند و بتواند آن را در HDFS حفظ کند.



شکل 5. انتقال RDF سه گانه به RDF چهارگانه (راه حل کدگذاری)

این ایده ایده­ای بود که راه حل­ها را به عنوان داده­ها و به عنوان ارث­بری تمام ویژگی­های مقیاس­پذیری که در نمودار قابل اعمال می­باشد مورد ارزیابی قرار می­داد. بنابراین، اندازه جمعیت X در کلاینت دارای یک اندازه ثابت است و از اندازه داده (X) صرف نظر نیز می­کند. ما به این تکنیک به عنوان کروموزوم­های توزیع شده، و به عنوان یک مفهوم در مورد توزیع ژن­ها که با راه حل­هایی همراه است اشاره کردیم. شکل 5 توضیح می­دهد که چگونه داده­های گراف در فرمت RDF ذخیره می­شوند و توضیح می­دهد که چگونه کدگذاری را حل­ها در داده­های RDF انجام می­شود.

ما از Apache Jena و Jena Alephas استفاده کردیم و منابع باز را برای مطابقت نیازمان تغییر دادیم. کلاسConvert­\_to\_quads\_Chromo برای تبدیل گراف RDF به Quads در شکل 5 توسعه داده شده است. این کلاس حاوی ترکیب کننده، کاهنده، نگاهنده و قابلیت نوشتن مناسب و همچنین کلاس­های ورودی و خروجی فرمت شده برای پردازش داده­های RDF نیز است. هر بلوک سه­گانه داده تبدیل به یک چهارم ژن تصادفی (بخشی از راه حل­ها) که متعلق به آن است می­شود و سپس آن را در HDFS ذخیره می­کند.

**5.2.2 توابع هدف**

در الگوریتم خوشه­بندی، ما مدولاسیون را به عنوان یک هدف در تکنیک هاجر و همکارانش به حداکثر رساندیم [65]. در [57]، مدولاسیون Q تعریف می­شود و سپس به عنوان کسری از لبه­ها در گروه 1 یا 2 قرار می­گیرد، و منهای تعداد لبه­های مورد انتظار در گروه­های 1 و 2 در یک گراف تصادفی با همان توزیع درجه گره به عنوان شبکه می­شود. از این رو، تعداد واقعی لبه­ها بین v و w و بدون حد انتظار از لبه­های بین آنها برابر Avw – (kv k­w)/2m است. لذا به معادلات 5-1 در بخش داده­های HDFS مراجعه کنید. به این نکته توجه داشته باشید که حداکثرسازی ماژلار تنها هدف نیست. هدف دیگر این است که طول راه حل را به حداقل برسانیم. با توجه به لبه­های درون خوشه­ای به عنوان لبه­های بین خوشه­ای، برخی از راه حل­ها بدون تغییر در مدولاسیون به دست می­آیند. از این رو، این راه حل­ها نیاز به داشتن شایستگی کمتر دارند اما به طور کامل نادیده گرفته می­شوند (ترکیبی از راه حل­های دیگر که ممکن است به خوشه­بندی بهتری منجر شود).

از آنجایی که ارزیابی ما در مورد مجموعه داده­ها، مفاهیم و روابط بین آنها در هر دو جهت در نظر گرفته می­شود (یک گراف بی­جهت)، ما از معادله (3) استفاده کردیم. Use caseها که در روابط تعریف شده به یک گراف بی­جهت منجر می­شوند، و دارای یک مدولاسیون گسترده­ای برای یک گراف جهت­دار ارائه می­کند و می­تواند از آن استفاده کند. از سوی دیگر، هدف خوشه­بندی یافتن جابجایی بهتر برای داده­های گراف است. از این رو، گراف­های جهت­دار به عنوان گراف­های بدون جهت در نظر گرفته می­شوند در حالی که خوشه­بندی لزوما کاربر را وادار نمی­کند که از همان فرضیه در هنگام پرس­وجو داده­ها استفاده کند.

**5.2.3 جزئیات مراحل خوشه­بندی ژنتیک**

**جمعیت اولیه**

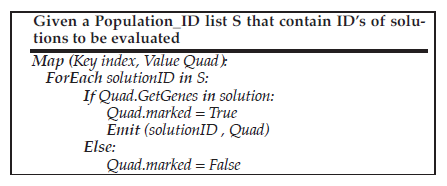
جمعیت اولیه فرآیند ایجاد مجموعه­ای از راه حل­های متنوع است. همانطور که در بخش کدگذاری و زیربخش توضیح داده شده است، ما داده­های RDF سه­گانه را به چهارگانه تبدیل کردیم و قادر بودیم یک ژن (بخشی از راه حل) را برای هر راس ذخیره کنیم، تا جایی که ژن به فهرستی از IDهای راه حل به صورت تصادفی متعلق به یک راس خاص باشد. به عنوان مثال، اگر یک D چهارگانه دارای S1 و S5 به عنوان ژن باشد، آن را به عنوان راه حل S1 و S5 ترجمه می­کند؛ و هر دو لبه D را به عنوان یک لایه بین خوشه­ای که دو خوشه جداگانه را متصل می­کنند در نظر می­گیرند.

در مجموعه­ای از راه حل­های وجود دارد، که در آن T، از یک t سه گانه گراف G ایجاد می­شود و S مجموعه­ای از راه حل­های S در جمعیت است. این بسیار مهم است که حداکثر اندازه این فهرست، را که اندازه عدد صحیح جمعیت است در نظر داشته باشید. فرآیند جمعیت اولیه در شکل 5 نشان داده شده است. کلاینت GA فقط یک آرایه دو بعدی از IDهای راه حل و شایستگی ماژول (اعداد صحیح و اعداد شناور) را ذخیره می­کند، و به کلاینت این امکان را می­دهد که فرآیند انتخاب و عملگرهای GA توزیع شده را آغاز کنند. کار بر روی یک آرایه دو بعدی ثابت کوچک، که در آن ژن­های واقعی در بلوک­های داده­ای در یک روش توزیع شده با استفاده از HDFS ذخیره می­شوند مقیاس­پذیری بیشتری را ارائه می­کند.

**ارزیابی راه حل­ها**

ارزیابی با استفاده از توابع هدف که در بخش هدف توصیف شده بودند انجام می­شود. هر راه حل توسط محاسبات مدولاسیون نظیر گراف ارزیابی می­شود، و یال­های موجود در گراف به عنوان یال­های خوشه میانی مشخص می­شوند. ما خوشه­هایی را که با عمل حذف یال­ها را مشخص می­کند و مولفه­های گراف غیرمتصل را در نظر می­گیرند تشخیص دادیم. سپس، ما مدولاسیون را با در نظر گرفتن تشخیص دوباره یال­ها به عنوان یال­های خوشه میانی محاسبه کردیم.

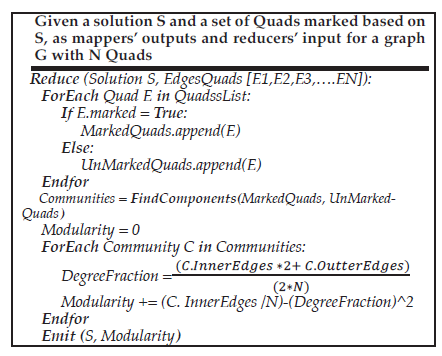
فرآیند محاسبه مدولاسیون در یک گراف بزرگ هم در منابع و هم در زمان وقت­گیر است، بنابراین ما تصمیم گرفتیم آن را با استفاده از وظایف توزیع شده به صورت چهارگانه به همراه داده­های اضافی بهبود ببخشیم. با استفاده از HDFS و مجموعه داده­های چندگانه در ماشین، ما قادر هستیم هر بار مجموعه­ای از راه حل­ها (تولید) را به صورت همزمان پردازش کنیم.



شکل 6. ارزیابی راه حل­های توزیع شده (وظیفه نگاشت).

کلاینت الگوریتم­های راه حل جاری را به صورت چهارگانه در HDFS ذخیره می­کند، و فهرستی از شناسه­های راه حل (فهرست اعداد صحیح) را برای ارزیابی می­فرستد. شکل 6 کارهای ارزیابی نگاشت را نشان می­دهد.

تابع نگاشت برای هر یک از گراف­ها به صورت چهارگانه بخشی از گراف چانک را در نظر می­گیرد. Jena Elephas با کلاس ورودی و خروجی اصلاح شده چهارچوب کروموزوم به جای چهارچوب پیش­فرض مورد استفاده قرار می­گیرد. هر ظرفی که در خوشه HDFS کار انجام می­دهد عملیات نگاشت را بر روی تکه­های گراف تخصیص می­دهد. پس از ترسیم تمام چانک­ها، در جفت <Keys,Values> شناسه­های راه حل و چارچوب­هایی که بخشی از راه حل­های مربوطه هستند، کار ترکیب کردن را بدین صورت انجام می­دهند. تمام مقادیر یک کلید همانند <Key,List of Values> که هر یک دارای راه حل­ها و فهرستی از چارچوب­های مشخص شده و علامت­گذاری شده نیز هستند (گراف­هایی که حاشیه­های بینشان به صورت خوشه­ای علامت­گذار شده­اند) نشان داده می­شوند. مرحله نهایی شامل کاهش وظایف است که در شکل 7 توضیح داده شده است.



شکل 7. ارزیابی راه حل­های توزیع شده (وظیفه کاهش).

تابع FindComponents با استفاده از یک الگوریتم خطی اصلاح شده تعداد لبه­های دورن خوشه­ای و لبه­های بین خوشه­ای را برای هر جامعه ذخیره و سپس اجرا می­کند. زمانی که وظایف کاهش پایان می­یابند، نتایج کاهنده­ها در HDFS نوشته می­شوند و نتایج شامل راه حل به همراه ماژولار می­شود. نتایج حاصل از یک آرایه دو بعدی ثابت از شناسه­های راه حل اعداد صحیح به صورت مناسب برای هر راه حل ایجاد می­شود. الگوریتم تکاملی این فایل را می­خواند و همچنان ارزیابی فرآیندهای انتخاب، تقاطع و جهش ادامه دارد. در نسل گذشته، یک قطعه اضافی از اطلاعات کنترل شده توسط یک متغیر بولی پیکربندی و سپس در HDFS نوشته می­شود؛ و این قطعه شامل وابستگی خوشه­ای در هر گره است. دلیل اینکه آنها فقط در نسل گذشته نوشته می­شدند، این است که هزینه نوشتن در HDFS قبل از اینکه در هر زمان مورد نیاز باشد تعیین و سپس کاهش پیدا می­کرد.

**انتخاب، جهش و Crossover**

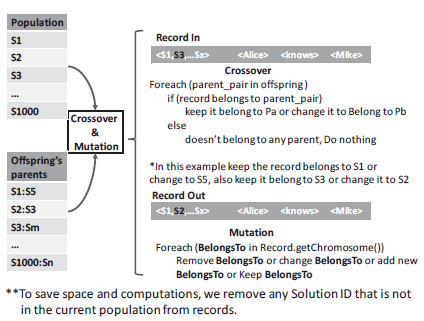
ما کروموزوم­ها را به صورت توزیع شده ذخیره کردیم، ما نیاز داشتیم که عملگرهای GA مورد استفاده را در منبع باز Jmetal تغییر دهیم تا بتوانیم آنها را بر روی چارچوب مربوط به گراف اجرا کنیم. این روش بوسیله ماژول­های تقاطع توزیع شده و ماژول­های crossover انجام می­شود، و در عوض وظایف تقاطع و جهش در جمعیت مربوطه ایجاد می­شوند.

بعد از ارزیابی جمعیت، فرآیند انتخاب که مبتنی بر هر راه حل ID و شایستگی است شروع می­شود. انتخاب تورنمنتن یک انتخاب مورد استفاده است، و دلیل آن این است از همگرایی راه حل­های بهینه محلی، که مبتنی بر روش کدگذاری است جلوگیری می­کند. با درجه­بندی جمعیت و انتخاب راه حل هر کلاس، مجموعه­ای از والدین همراه با IDهای فرزند ایجاد می­شوند. شکل 8 یک نمودار سطح بالا است که مراحل ایجاد عملگر الگوریتم GA را نشان می­دهد.



شکل 8. الگوریتم ژنتیک توزیع شده برای خوشه­بندی RDF

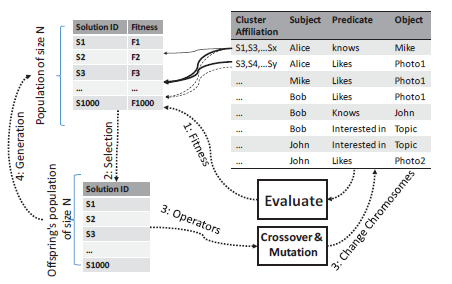
راه حل­های ذخیره­سازی و کدگذاری در روش توزیع شده مزایای جمعیت­های کوچک و ثابت را در یک طرف کلاینت به ارمغان می­آورد. با این حال، عملگرهای GA در منبع باز Jmetal نیاز به اصلاح و همچنین NSGALL دارند، تا ما بتوانیم از آنها استفاده کنیم. NSGALL راه حل­های اصلی جمعیت و فرزندان را ایجاد می­کند و در یک زمان ممکن این راه حل­ها را ارزیابی می­کند؛ چنین وضعیتی سربار کارهای HDFS است. در عوض، ما NSGALL را به DNSGALL (توزیع NSGALL) با ایجاد مجموعه­ای از راه حل­ها و سپس ارزیابی عملگرهای GA را در MR2 تغییر دادیم. شکل 9 وظیفه جهش توزیع شده تقاطع و توزیع را نشان می­دهد.



شکل 9 عملگر جهش و تقاطع در جستجوی ژنتیکی

وظیفه جهش و تقاطع توزیع شده گرفتن جمعیت به عنوان ورودی در طول نتایج انتخاب است، برای هر یک از فواصل داده­ای، کروموزوم­ها به ترتیب تغییر می­کنند. وظیفه هر راه حل ID (ژن) حذف جمعیت فعلی که هیچ تعلقی ندارند تا فضای محاسباتی را بتواند ذخیره کند. سپس، همانطور که در شکل 8 نشان داده شده است، جمعیت جدید فرزندان به ارزیابی ارسال می­شوند. در اینجا ما باید به این نکته توجه داشته باشیم که راه حل­هایی که متعلق به نسل قبلی می­باشند، ارزیابی نخواهند شد به این دلیل که آنها از قبل آمادگی داشتند. این یک تکنیک کپی شایستگی است که حجم زیادی از محاسبات را زمانی که کلان داده­ها مجموعه­ای طولانی از نسل­ها را به کار می­برند ذخیره می­کند.

فرآیندهای بازنمایی، جمعیت اولیه، ارزیابی، انتخاب و ارزیابی جمعیت فرزندان در شکل 10 نشان داده شده است. اعداد نشان دهنده دستورات فرآیندها و وظایف هستند. ما داده­های پویا را به عنوان یکی از 5 محدودیت­های کلان داده­ها شناختیم (سرعت، تنوع، وراثت، مقدار و حجم)، و الگوریتم را زمانی که یک راه حل برای سلسله­ای از نسل­ها همگرا می­شود، به حالت تعلیق درمی­آید و سپس کار را ادامه می­دهد تا زمانی که داده­های جدید به منظور رسیدن به آخرین نسل به دست آیند.



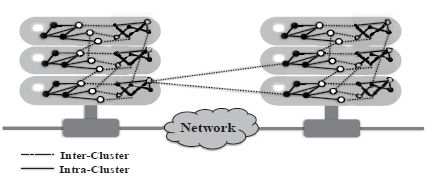
شکل 10. جریان فرآیند خوشه­بندی الگوریتم ژنتیک توزیع شده.

تعلیق الگوریتم تضمین می­کند که خوشه­بندی به داده­های جدید اعمال می­شود در حالی که داده­های قدیمی دارای بهترین وابستگی نیز هستند، بنابراین نیازی نیست که فرآیند خوشه­بندی از ابتدا آغاز شود.

**5.3 پارتیشن­بندی و جاگذاری کردن**

پس از خوشه­بندی داده­های RDF، آخرین مرحله این بود که داده­ها را مجددا تقسیم کنیم و گراف را به ترتیب چهارگانه کنیم. در این مرحله هدف این بود که به جای عمل چهارگانه که متعلق به یک خوشه بود با درجه بالایی از اتصال برای اطمینان خوشه چهارگانه را پارتشین­بندی می­کردند. هدف دیگر این بود که خوشه­های میانی را به صورت فیزیکی به خوشه پارتیشن­بندی شده متصل کند، تا فاصله نگاشت خوشه میانی نشان داده شود. شکل 11 تخصیص مورد نظر را به صورت چهارگانه نشان می­دهد، و فرض بر این است که فاصله افقی در شکل نشان دهنده فاصله فیزیکی بین گره­های محاسباتی است (فاصله مسیریابی شبکه).

ما با توجه به تعداد مسیر­های بین آنها (شبکه، مسیریاب­ها، سوئیچ­ها و ...) فاصله گره­های HDFS را تعیین کردیم. ما HDFS در بیشتر دستگاه­ها بهم متصل کردیم سپس با استفاده از شبکه­های متعدد به منظور ایجاد فاصله در مسیریابی راه اندازی کردیم. همانطور که در شکل 11 نشان داده شده است اسکریپت جایگزین شبکه چهارگانه نیز شده است.



شکل 11. قرار دادن شبکه­های چهارگانه در گره­های HDFS مختلف.

پارتیشن­ها براساس تعداد ماشین­ها ایجاد می­شوند، هر ماشین دارای پارتیشن مخصوص به خود است. وظیفه نگاشت کاهش اسکن کردن کوآدراپل و مکان کوآدراپل که مربوط به خوشه تصادفی در یک پارتیشن بود است، سپس کوآدراپل جایگزین شده را ، که منجر به تمام اتصال­های ID خوشه شده بود ارسال کرده بود و در مرحله بعدی ذخیره کرده بود. دومین اسکن به صورت چهارگانه نزدیک به خوشه­های درونی است و در همان پارتیشن قرار می­گیرد و در مجموعه داده­های اصلی منتشر می­شود. علاوه بر این، دومی در نزدیکترین خوشه­های میانی قرار می­گیرد. زمانی که خوشه­های متصل به هم وجود نداشته باشد، یک خوشه به صورت تصادفی از مجموعه داده­ها انتخاب می­شود تا زمانی که اطلاعات بیشتری در دسترس نباشد. به عنوان مثال، شکل 11، نشان می­دهد که خوشه­ها چگونه با سه اتصال بین خوشه­ای در همان گره HDFS قرار می­گیرند و چگونه با دو اتصال بین خوشه­ای در گره بعدی HDFS قرار می­گیرند، و چگونه خوشه­های بیشتری در گره­های بعدی قرار می­گیرند.

**6 آزمایشات و نتایج**

ما این آزمایش را به دو بخش تقسیم کردیم: بخش اول طراحی و آزمایش الگوریتم خوشه­بندی در ذخیره­سازی گراف به منظور تست نتایج خوشه­بندی و بخش دوم در مورد تست­ها و مقایسه تاثیر چارچوب بهینه­سازی در HDFS است. تمام نمودارها و مدل­های فرآیند با استفاده از جدول پردازش می­شوند [66].

**6.1 تبدیل گراف و خوشه­بندی**

ما صحت الگوریتم خوشه­بندی را تایید کردیم و اطمینان حاصل کردیم که نتایج قابل اعتماد و قابل مقایسه نیز هستند. ما برخی از مجموعه داده­های شناخته شده را با دقت انتخاب کردیم و اطمینان دادیم که آنها مجموعه داده­هایی هستند که در بررسی­های قبلی برای مقایسه مورد استفاده قرار گرفتند. این مجموعه­ها عبارتند از:

انجمن کاراته زاخاری: گراف شامل 34 رئوس و 78 لبه است. گره­ها نشان دهنده اعضای انجمن دانشگاه کاراته می­باشند و ارتباطات بین آنها نشان دهنده الگوهای ارتباطی هستند. این انجمن در سال 1997 جمع­آوری شد [67].

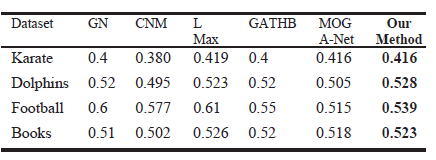
دلفین پوزه بطری معمولی: شبکه از 62 دلفین به همراه تعاملات بین آنها ایجاد شده و در طول 7 سال، یعنی سال 1994 جمع­آوری شده است.

کتاب­های سیاسی ایالات متحده: شبکه از 105 گره و 441 لبه ایجاد شده است. شبکه نشان می­دهد که کتاب­های مربوط به سیاست ایالات متحده، اغلب باهم خریداری می­شدند.

فوتبال دانشکده آمریکایی: شبکه از 115 گره که نشان دهنده تیم هستند، و 613 لبه اتصال ایجاد شده است.

جدول 1 نتایج اعتبارسنجی الگوریتم را نشان می­دهد و آن را با بعضی از الگوریتم­های محبوب مقایسه می­کند. الگوریتم ما در بعضی موارد مدولاسیون را به حداکثر رساندند و حالت مدولاسیون را در بقیه بدست آورده­اند. بعضی از الگوریتم­ها به دلیل مدولاسیون بالا حذف شدند؛ همانگونه که دانیل آلوئیز، سونیا کافییری و همکارانش نیز وجود دارند چنین نتایجی برای خوشه سخت غیر ممکن است [68]. آنها هر یک از این مجموعه داده­های مدولاسیون را به صورت بهینه پیدا و ثابت کرده­اند.

جدول 1 مقایسه به حداکثر رساندن مدولاسیون



نتایج در جدول 1 و [68] ثابت می­کند که رویکرد ما نتایجی را که به راه حل­های بهینه نزدیک می­باشند را ارائه می­دهد (در مقایسه با سایر الگوریتم­های محبوب) و کیفیت این نتایج در بیشتر موارد بهتر است. بعضی از موارد با مدولاسیون نسبتا پایینی نشان داده شده­اند.

ما دریافتیم کهانتخاب در نقش راه حل­ها بسیار اهمیت دارد. علاوه بر این، برای مجموعه داده­های خاص، انتخاب باینری به مدولاسیون بالا را در نسل­های کوچکتر پوشش می­دهد، در حالی که، انتخاب تصادفی می­تواند نرخ بالاتری از جهش­ها را از شایستگی بهینه محلی فراهم کند.



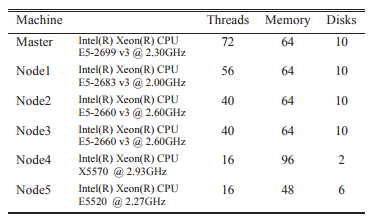
شکل 12. شایستگی جمعیت در مقابل ترسیم توزیعی نسل (شبکه انجمن کاراته).

ارزیابی­ها پس از هر جمعیت گزارش می­شود و همگرایی راه حل­ها را برای نسل­ها تجزیه و تحلیل می­کند. ما گراف­ها و مدل­های فرآیند محاسبه شده را با از بین بردن آرایه­های جمعیت و با استفاده از ترسیم توزیعی به منظور نشان دادن نتایج بصری هر نسل را تولید کردیم. ما بین توزیع مدولاسیون و تعداد نسل­ها به منظور استخراج مدولاسیون­ها ارتباط برقرار کردیم. شکل 12 توزیع مدولاسیون را در مقابل نسل نشان می­دهد.

برای مقیاس رویکرد کلان داده، ما از LUBM برای تولید داده­های گراف RDF استفاده کردیم و همانطور که در جدول 2 نشان داده شده است از خوشه­ای با ویژگی زیر، استفاده کردیم. هر کانتینری دارای 48 دیسک قابل دسترس، و دو هسته پردازنده و 4 گیگابایت حافظه می­باشد.

ما از 6 گره محاسباتی در مقایسه با 10 و 20 گره در بررسی­های مشابه استفاده کردیم. ما فقط تعداد گره­ها را به منظور تایید اثر ارتباطات شبکه و فاصله در شبکه مشاهده کردیم. با این حال، YARN مفهومی منطقی است که منابع را نسبت به تعداد گره­ها مقایسه می­کند. خوشه محاسباتی و پیکربندی ما به 86 کانتینر، که هر کدام 4 گیگابایت حافظه و 2 بند (یک هسته) CPU (حداکثر 2.93 گیگاهرتز و حداقل 2.00 گیگاهرتز)، 344 گیگابایت حافظه و 172 بند CPU (86 هسته)، و در مقایسه با کل 80 گیگابایت حافظه و 40 هسته CPU و 40 دیسک مساوی بین 20 گره محاسباتی هانگ و همکارانش تقسیم می­شود [1].

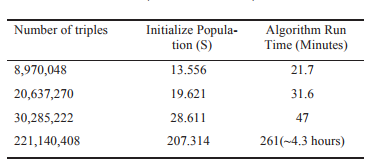
جدول 2 پیکربندی خوشه هادوپ



دانشگاه لهایف بنچمارک (LUBM) یک دانشگاهی است که در حوزه آنتولوژی داده­های مصنوعی OWL و RDF با مقیاس­پذیر بالا و به چهارده پرس­وجو که نشان دهنده ویژگی آن دانشگاه است. LUBM یک معیار گسترده است که از جامعه وب معنایی استفاده می­کند.

ما مجموعه­ای از چندین داده با اندازه­های مختلف به منظور مقایسه رفتار الگوریتم تولید کردیم. ما قبل از خوشه­بندی به طور پیش­فرض نوع تایپ را حذف کردیم و یک روش برای افزایش کیفیت خوشه­ها و پیچیدگی گراف ارائه دادیم [1]. زمان اجرای راه حل­های جمعیت اولیه را تجزیه و تحلیل کردیم؛ و حجم جمعیت موجود در آن 1000 راه حل در هر نسل است. جدول 3 زمان اجرارا به منظور تبدیل گراف به یک گراف چهارگانه و جمعیت اولیه را نشان می­دهد. خطای مقادیر زیادی از داده­ها، به همراه مقدار پرس­وجوی آنها روی داده­ها پردازش می­شوند و قادر هستند تکنولوژی­های جدید را بهینه­سازی کنند [22]. به این نکته توجه کنید که چارچوب ما، زمانی که داده­ها به شدت پردازش می­شوند یا به طور مداوم مورد سوال قرار می­گیرند به بهترین نحو مورد استفاده قرار می­گیرد و باعث پاسخ سریع و کاهش هزینه­های سیستم سخت­افزاری نیز می­شود.

جدول 3 زمان اجرای الگوریتم & جمعیت اولیه (مجموعه داده­های LUBM)

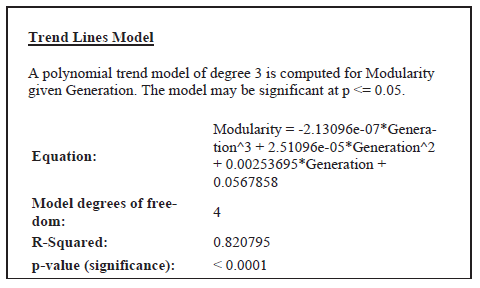


ما داده­های LUBM را بیشتر تجزیه و تحلیل کردیم. مقدار زیادی مقادیر سه­گانه وجود دارد، و در آن جمعیت اولیه با یال­های تقاطع خوشه­ای به صورت تصادفی در طول زمان اجرا نمی­شوند. شکل 13 همگرایی جمعیت را با به حداکثر رساندن مدولاسیون یک مجموعه داده که 30M چهارچوب دارد در طول زمان نشان می­دهد.



شکل 13. به حداکثر رساندن مدولاسیون همگرایی (30 میلیون LIBM سه­گانه)

در شکل 14 فرآیند مدل 30 میلیون مجموعه داده سه­گانه LUBM نشان داده شده است.



شکل 14. توصیف مدل فرآیند (30 میلیون مدل سه­گانه)

تکنیک کدگذاری که ما معرفی کردیم فضای راه حل به همراه راه حل­های بهینه محلی را ایجاد می­کند. از این رو، با استفاده از نرخ جهش 100 درصدی و تقاطع چندین نقطه از سقوط راه حل­های محلی بهینه جلوگیری می­کنیم. با وجود اینکه نرخ جهش روی 100 درصد تنظیم شده بود، یال­های درونی خوشه به عنوان یال­های بین خوشه­ای هیچگونه تفاوتی را در نوع مدولاسیون راه حل مشاهده نکرده بودند. از سوی دیگر، کدگذاری راه حل­هایی را تحت تاثیر جهش کمتر از 100 درصد بودند را ارائه داد (~72 درصد از راه حل­ها تحت تاثیر داده­های استفاده شده هستند). کدگذاری ارائه شده نسبت به کدگذاری سنتی کمتر تحت تاثیر جهش است. از این رو، نرخ جهش بالا برای تاثیر راه حل­ها لازم و ضروری است.

شکل 15 مدولاسیون و تعداد آنها را در تمام نسل­ها توصیف می­کند (هر بین برابر محدوده­ای از مقدار بین x تا مقدار بین x بعدی است). از آنجایی که بیشتر یال­های درون خوشه تحت تاثیر تعدادی از جوامع نیستند، با این حال با داشتن چنین یال­هایی راه حل­ها تحت تاثیر شایستگی و تعدادی از توضیحات مدولاسیون بالا با غیر حداکثر ماژول نیز نیستند. بنابراین، راه حل­های چهارگانه بر توانایی الگوریتم هیچ تاثیری نمی­گذارند.



شکل 15. شمارش مدولاسیون در تمام نسل­ها به عنوان مدولاسیون بین (30 میلیون LUBM سه­گانه).

در مقیاس بالا، ما 221M فاصله اولیه را توسط LUBM سه­گانه پوشش دادیم. شکل 16 توزیع و همبستگی بین اندازه شایستگی و فرکانس را در تمام نسل­ها نشان می­دهد.



شکل 16. همگرایی به منظور به حداکثر رساندن مدولاسیون بالای نسل­ها (221 میلیون LUBM سه­گانه).

شکل 17. حداقل، حداکثر، میانگین و میانی را برای هر مدولاسیون سازگاری نشان می­دهد. نرخ جهش بالا فضایی را برای برخی از نسل­های مختلف نگهداری می­کند.



شکل 17. همگرایی به منظور به حداکثر رساندن مدولاسیون بالا نسل­ها را نشان می­دهد (221 میلیون LUBM سه­گانه).

**6.2 آزمایشات عملکرد سیستم**

در این بخش، ما عملکرد سیستم را برخلاف ذخیره­های RDF چندگانه که شامل SHARD بعد از خوشه­بندی است اندازه­گیری می­کنیم. ما کلودرا ایمپالا را برای ایجاد جدول روی داده استفاده کردیم. ما عمدتا روی استفاده از زمان و منابع تمرکز می­کنیم. همانطور که در جدول 2 دیده می­شود تنظیمات سیستم مورد استفاده قرار گرفته­اند. ما به چارچوب به عنوان پارتیشن­بندی خوشه اشاره می­کنیم.

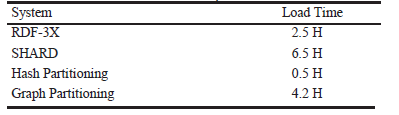
در آزمایشات، ما یک مجموعه داده را با استفاده از LUBM تولید کردیم. اندازه مجموعه داده­ها از 37 تا 142 گیگابایت در قالب Ntriples و از 200 تا 600 میلیون به صورت سه­گانه است (جدول 5 را مشاهده کنید). جدول 6 هر پیچیدگی پرس­وجو را به عنوان تعدادی از پیوست­ها در هر پرس­وجو را نشان می­دهد.

6.2.1 زمان بارگذاری و خوشه­بندی

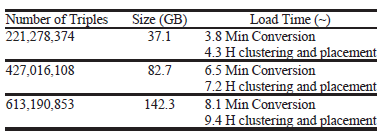
هوانگ و همکارانش نتایج بارگذاری 270 میلیون RDF سه­گانه در 20 ماشین خوشه HDFS را در جدول 4 نشان دادند. الکساندر و همکارانش [24]، 10 ماشین خوشه را در 40 دقیقه بارگذاری کردند. نتایج ما در جدول 5 نشان داده شده است.

به دلیل تفاوت در منابع، معیارهای پرس­وجو در بخش مقایسه عملکرد پرس­وجو، ما نتایج را نرمالسازی کردیم تا بتوانیم زمان پاسخ پرس­وجو را مقایسه کنیم. به طور قابل توجهی یک کار در مورد پارتیشن RDF براساس تجزیه و تحلیل خوشه­های داده­ای وجود دارد. با این حال، تاثیر آن در بخش مقایسه عملکرد پرس­وجو توضیح داده شده است، و در زمان پرس­وجو در بخش بهینه­سازی ذخیره شده است. این نکته بسیار مهم است زمانی که مقادیر نسبت به حجم ذخیره­سازی GB سه برابر بزرگتر می­شود؛ عمل فشرده­سازی می­تواند اندازه مسئله ذخیره­سازی را حل کند، اما نمی­تواند میزان داده­های مورد نیاز را پردازش کند. همچنین این نکته هم بسیار مهم است زمانی که نتایج خوشه­بندی توسط معماری ارائه شده ذخیره می­شود، داده­ها و تغییرات جدید قابل دسترس هستند و نیازی به شروع مجدد ندارند. به عبارت دیگر، اضافه کردن تغییرات به داده­ها، تا حدودی به زمان واقعی نزدیک است و با اضافه کردن داده­های سه­گانه جدید به سرور فیزیکی خوشه نیز به آن متصل می­شود.

جدول 4 زمان بارگذاری 270 میلیون سیستم توسط هوانگ و همکارانش [1]



جدول 5 زمان اجرای پارتیشن خوشه

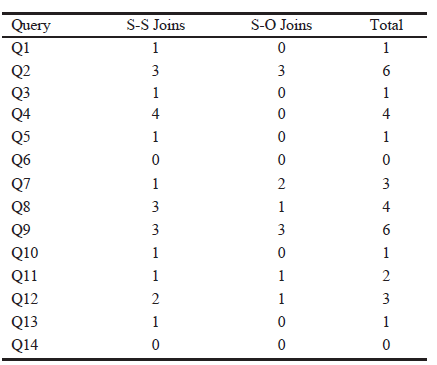


**6.2.2 مقایسه عملکرد پرس­وجو**

برای مقایسه بقیه بررسی­ها، ما ابتدا به درک پیچیدگی هر پرس­وجو از 14 پرس­وجو نیاز داریم. همانطور که در جدول 6 نشان داده شده است [1]، هر پرس­وجو دارای تعدادی پیوست­های subject-to-subject (S-S) و subject-to-object (S-O) است.

پارتیشن­بندی عمل نگاشت درهم­سازی یک کلید (Subject) را برای بلوک مشابه نشان می­دهد. بنابراین، مسائل بهینه­سازی در subject-to-subject کمتر از subject-to-subject رخ می­دهد. پیوست­های subject-to-object باعث می­شود که بسیاری از مسائل بهینه­سازی در پراکندگی داده­های درهم­سازی ایجاد شود، به این دلیل که مقادیر کلید مشابه (نقاط مربوط به داده) در همان بلوک­های داده­ای تایید نمی­شوند.

جدول 6 پیچیدگی پرس­وجو LUBM [1]

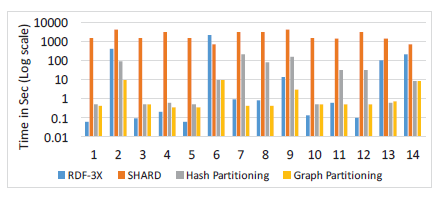


هوانگ و همکارانش [1] پرسشنامه­ای را به دو گروه تقسیم کردند. پرس­وجوهای 1، 3، 4، 5، 7، 8، 10، 11 و 12 به سرعت در RDF به صورت تک ماشین ذخیره می­شدند و در RDF-3X اجرا می­شدند؛ و پرس­وجوهای 2، 6، 9، 13 و 14 در ذخیره­های RDF کندتر می­شدند. برای پرس­وجوهای سریع، اندازه داده­ها زیاد اهمیتی ندارند به این دلیل قبل از اینکه اسکن شوند، کاهش پیدا می­کردند [1]. از این رو، از کار انداختن چنین ذخیره­هایی در چندین ماشین مزایای بسیاری ندارد و تنها تاخیر شبکه­ای را برای شروع نشان می­دهد [1]. از سوی دیگر، پرس­وجوهای آهسته در ذخیره­های RDF تک ماشین فاقد مقیاس­پذیری هستند.

برخلاف چارچوب داده­های آگاه HDFS، برخی از تلاش­های قبلی برای پرس­وجوهای آهسته به دست آمده است، به عنوان مثال پارتیشن­بندی داده­ها در SHARD. با این حال، عمل درهم­سازی منجر به محدودیت­های subject-to-subject به دلیل نیاز به حرکت داده­های متوسط در شبکه می­شود. مثال دیگری از هوانگ و همکارانش ارائه شده است و بدین صورت است اشیاء متصل به یک موضوع مورد پردازش قرار می­گیرند تا به یک بلوک مشابه به منظور تقسیم بین subject و object تبدیل شوند. با ای حال، محدودیت­های فضا به علت افزایش اندازه داده­ها وجود خواهد داشت؛ همچنین محدودیت استفاده از چنین الگوریتمی در یک گراف متصل نیز وجود دارد [1]. ما از خوشه­بندی در چارچوبمان استفاده کردیم و از جهش استفاده نکردیم. بنابراین، داده­های سه­گانه نیازی به تکرار در بسیاری از موارد مختلف پراتشین­ها ندارند، و هیچ افزایشی در اندازه داده­ها رخ نخواهد داد (به غیر از تکرار قابلیت اطمینان HDFS از چیزی استفاده نخواهند کرد). سایر کارهایی مانند سمپالا [24]، یا با استفاده از HIVE، PigSPARQL [25] & [26]، ادغام نگاشت [27] و MAPSIN [28] مقیاس­پذیری را برطرف می­کنند و از ذخیره­های مختلف سه­گانه نیز استفاده می­کنند.

همانطور که در شکل 18 قابل مشاهده است هوانگ و همکارانش [1] چارچوب مختلف 270 میلیون LUBM سه­گانه را پوشش دادند.

همانطور که هوانگ و همکارانش مشاهده کردند [1] نتایج تغییر داده­ها در هر پارتیشن زیرگروهی شامل راس­هایی است که دو جهش هر subject را در subject سه­گانه به طور موثر در زمان اجرا افزایش می­دهد. سپس، به طور قابل توجهی افزایش پیدا می­کند و همچنین یک گام از پردازش تکراری را پس از پایان پرس­وجو اضافه می­کند. هوانگ و همکارانش [1] آن را به عنوان یک توافق میان اندازه و زمان پاسخ سریع توضیح دادند.

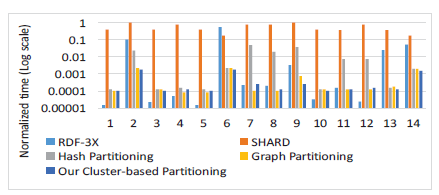


شکل 18. زمان پرس­وجو هوانگ (270 میلیون LUBM سه­گانه)

دو جهش تضمین می­کنند که تکنیک بهینه­سازی پرس­وجوهای 2، 7، 8، 9، 11 و 12 فعال شده هستند. از سوی دیگر، بقیه پرس­وجوها در مقایسه با داده­های پارتشین­بندی شده درهم به آهستگی متوقف می­شوند، به این دلیل که یک گام مورد نیاز برای حذف تکرارهایی که توسط الگوریتم تکراری ایجاد شده­اند نیز وجود دارد. تضمین جهش­ها هیچ پرس­وجویی را بهینه نمی­کند. پرس­وجوهای 1، 3، 4، 5، 6، 10، 13 و 14 در نمودار گرافیکی درهم­سازی تقسیم­بندی می­شوند یا فقط پیوست­های تکراری subject-to-subject را حذف می­کنند.

مقایسه پارتیشن­بندی براساس خوشه­بندی و SHARD به شما این امکان را می­دهند که به طور مستقیم از مزایای تکنیک پارتیشن­بندی پیشنهادی در سه مکان سه­بعدی ساده HDFS استفاده کنید. برای مقایسه نتایج ما ابتدا تفاوت منابع HDFS را مورد بررسی قرار می­دهیم. دو اکوسیستم مختلف هادوپ بسیار سخت هستند به این دلیل که ما ایده­ای دقیق برای تنظیمات و یا آنچه که در ماشین در حال اجرا باشد را مورد پرس­وجو قرار دادیم. با این حال، یک راه حل این است که SHARD را در اکوسیستم هادوپ به صورت سه­گانه انجام دهیم، سپس نسبت تفاوت را در آن پیدا کنیم. به عبارت دیگر، اگر پرس­وجو Q1 در HDFS زمان X را در خوشه (A) و 2X را در خوشه (B) با همان HDFS به صورت بومی داشته باشد، خوشه (A) برای Q1 سریعتر عمل می­کند. بنابراین، بهینه­سازی Q1 با سرعت Y با استفاده از یک الگوریتم خاص Q1 در زمان X/Y در خوشه A و 2X/Y در خوشه B در الگوریتم خاص شروع می­شود. به این نکته توجه داشته باشید که بهینه­سازی یک الگوریتم خاص از لحاظ Y دارای عملکرد برابری است و نسبت به HDFS بومی از خوشه­ای که صرف نظر شده است استفاده می­کند. شکل 19 نتایج مقایسه­ها (میانگین) را برای چارچوب HDFS آگاه (پارتیشن­بندی مبتنی بر خوشه) پس از اجرای 20 پرس­وجو نشان می­دهد و میانگین واریانس آن نیز 0.06 می­باشد.

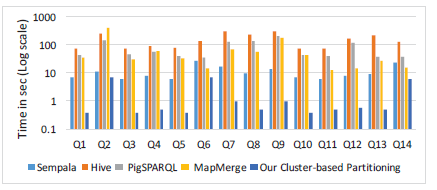
نتایج افزایش عملکرد زمان اجرا برای پرس­وجوهای 2، 3، 6، 9، 13 و 14 گزارش شده­اند. عملکرد بسیار نزدیکی در پرس­وجو 1 که تاثیر آن کمتر است در پرس­وجو 4، 5، 7، 8، 11 و 12 رخ داده است. دلیل بهبود subject-to-subject گام اضافی است که تکرار در پراکندگی گراف را از بین می­برد و توسط هوانگ و همکارانش مورد استفاده قرار گرفته است [1]. از سوی دیگر، افزایش عملکرد مربوط به اتصال subject می­شود که باعث کمتر شدن مقادیر داده­های اسکن شده می­شود و توسط عمل پارتیشن­بندی مبتنی بر خوشه منتقل می­شود، به این دلیل که پارتیشن­بندی مبتنی بر خوشه حضور اکثر اشیاء مرتبط با افراد را در یک پارتیشن­بندی بدون تکرار تضمین می­کند (اسکن کمتر). چنین نتایجی فضای ذخیره­سازی را نسبت به بخش گرافیکی کمتر گزارش می­کنند [1].



شکل 19. زمان اجرای پرس­وجو، داده HDFS آگاه (270 میلیون LUBM سه­گانه).

ما بیشتر ترافیک شبکه را در هر گره محاسباتی HDFS با استفاده از ابزارهای مانیتورینگ عملکرد لینوکس را تحلیل کردیم. دستورات SAR برای نظارت بر آداپتورهای شبکه بر روی هر ماشین در طی پرس­وجو استفاده می­شوند. نتایج حاصل از گزارش هر گره HDFS به صورت یک فایل جمع­آوری می­شوند. تاثیر چارچوب ما در بهینه­سازی ترافیک شبکه به منظور نشان دادن LUBM در 221 میلیون گراف سه­گانه RDF به شرح زیر است: یک عامل بهینه­سازی بزرگ در اغلب موارد به S-O متصل می­شود، و باعث کاهش یک عامل 200X به منظور تقسیم­بندی SHARD درهم Q7 می­شود و سپس در یک خوشه سه­گانه قرار می­گیرد. علاوه بر این، ~160X برای Q8 به منظور بهینه­سازی Q9 در ترافیک شبکه گزارش شده است.

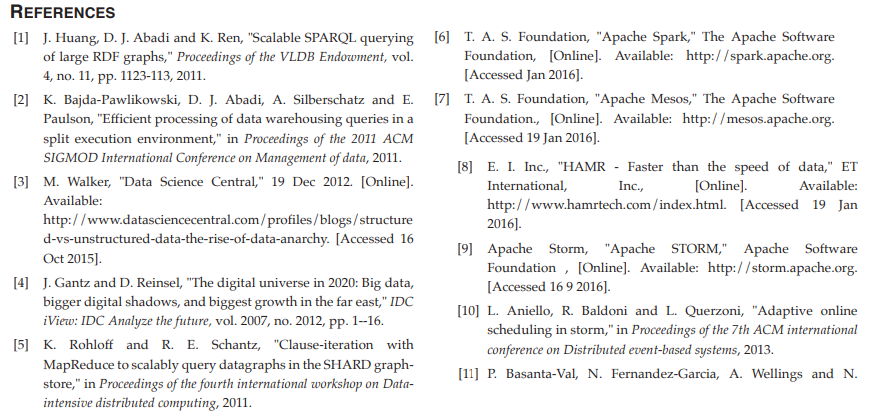
یکی از اهداف ما این است که با داده­های پویا برخورد کنیم، و براساس چارچوب­هایی مانند سمپالا، ادغام نگاشت، PigSPARQL و سمپالا HIVE، بدون در نظر گرفتن محدودیت­های آنها در برخورد با به روز­رسانی­های پویا، بتوانیم جستجو کنیم. شکل 20 نتایج را برای 221 میلیون رکورد RDF نشان می­دهد.

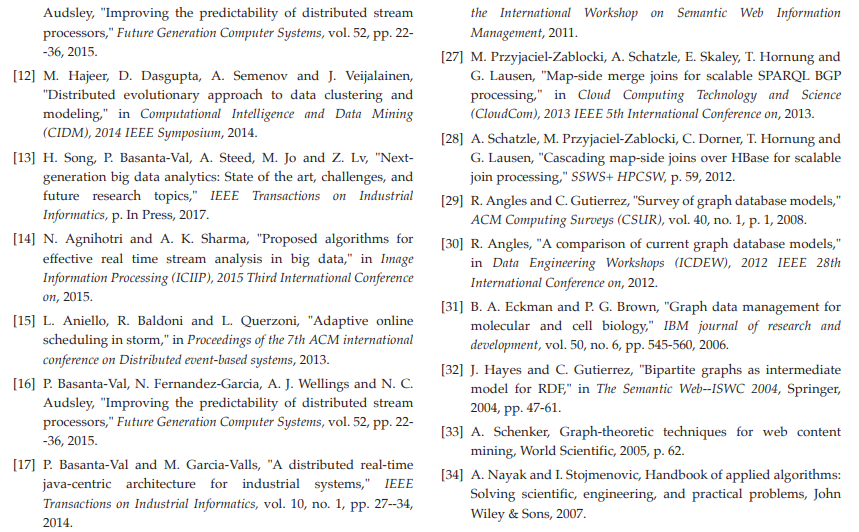


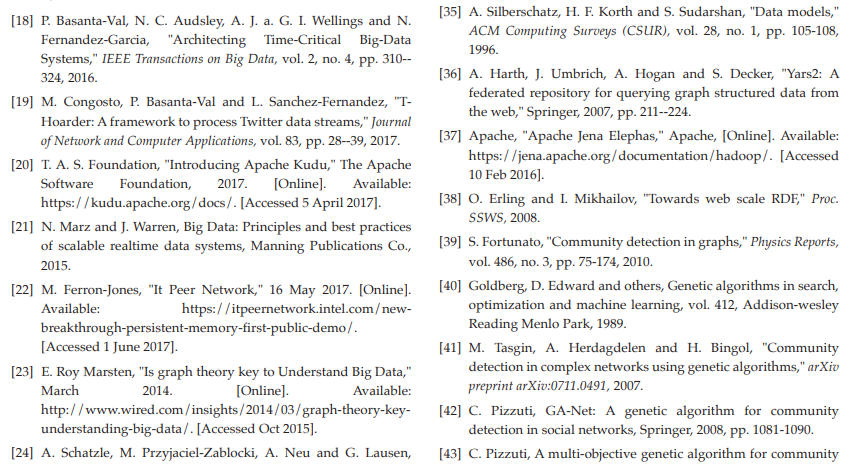
شکل 20. زمان اجرای پرس­وجو، داده HDFS آگاه (221 میلیون LUBM سه­گانه).

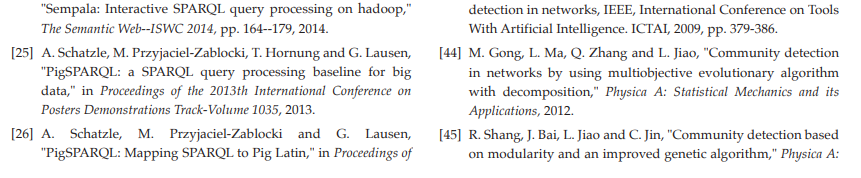
**7 نتیجه­گیری**

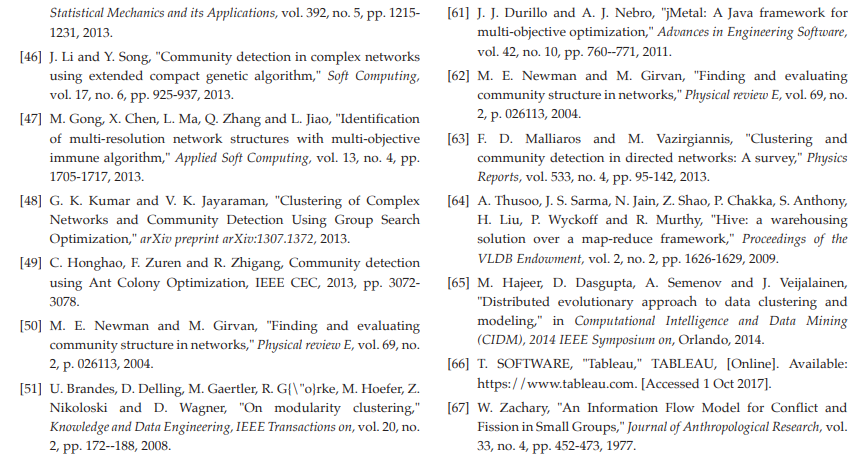
در این مقاله، ما داده­های آگاه HDFS و سرویس­های بالای HDFS را که به صورت بهینه­سازی در state-of-the-srt در حال اجرا می­باشد را ارائه کردیم. ما یک پارتیشن­بندی داده­ها را براساس خوشه به منظور جابجایی مکان فیزیکی داده­ها و منطبق شدن با منطقه گراف و در فرآیندهای HDFS ارائه کردیم. این امکان را می­دهد که پردازش موازیِ پرس­­وجوهای داده HDFS را برای منابع کمتر استفاده کنیم. چارچوب ما قادر بود برخی از تلاش­ها را سریع­تر انجام دهد و قادر بود به آرامی ذخیره­سازی داده­های RDF مقیاس­پذیر را انجام دهد. با این حال، از منابع کمتر استفاده می­کردند. بررسی­هایی که در تجزیه و تحلیل نسل بعدی و معماری لامبدا [15]، [16]، [17] و [18] همراه با آپاچی کودو و مجموعه­ای از بررسی­ها در [21] ثابت شدند عملکرد سریع­تری در پردازش حجم کاری OLAP و عملکرد قوی در حجم کاری زمان دارند و بسیار هم مهم می­باشند. با این حال، تلاش می­کند که داده­های هوشمند را در چنین روش­هایی تحت تاثیر قرار دهد. برای کارهای آتی، ما قصد داریم برای بهبود بیشتر از کدگذاری توزیع و عملگرهای ژنتیک به منظور کاهش هزینه­های محاسباتی پشتیبانی کنیم. همچنین ما قصد داریم آزمایشات را به صورت پویا به منظور سرعت بخشیدن بیشتر به جریان داده­ها به روز رسانی کنیم و به منظور استفاده از ابزارها و چارچوب معماری لامبدا و تجزیه و تحلیل نسل بعدی که در بررسی­های اخیر ارائه شده آزمایش کنیم.

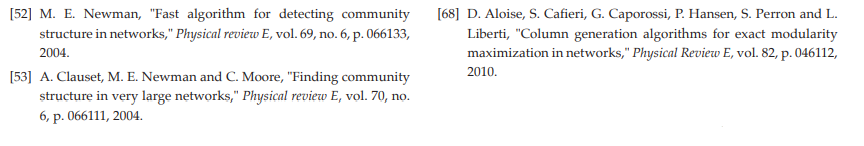


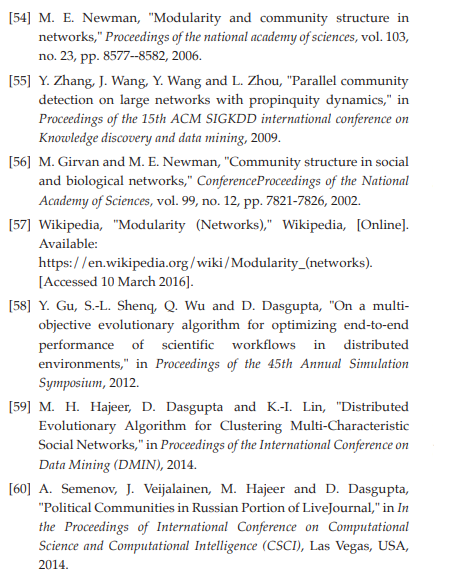












­