ارائه یک سیستم پیشنهاد دهنده مقیاس پذیر با استفاده از الگوریتم پردازش موازی MapReduce

حمیدرضا سلیمیان[۱]، هادی خسروی[۲]
دانشکده کامپیوتر موسسه آموزش عالی صفاهان، Hamid@salimian.me[۱]
دانشگاه شهرکرد، khosravi@eng.sku.ac.ir[۲]

چکیده

رشد سریع اطلاعات در اینترنت از یک سو، و از سوی دیگر گم شدن کاربران در میان ساختار پیچیده اطلاعات، روز به روز اهمیت و جایگاه مفاهیمی مانند سیستم‌های پیشنهاد دهنده را بالاتر می‌برد. این رشد سریع کاربران و آیتم‌ها مخصوصا در سایت‌های تجاری خود یک چالش اصلی برای این نوع سیستم‌ها شده است که دقت و مقیاس پذیری آنها را بشدت تحت تاثیر قرارداده است. به صورتی که مقیاس پذیر بودن یک سیستم پیشنهاددهنده، اولویت بالاتری نسبت به دیگر عوامل حتی دقت در سایت های بزرگ شده است. در این مقاله با استفاده از الگوریتم MapReduce بر بستر Hadoop، سیستم پیشنهاددهنده‌ای با دقت بالا مقیاس پذیر شده است. نتایج ارزیابی نشان‌دهنده مقیاس‌پذیر بودن روش پیشنهادی در اندازه‌های مختلف است.

کلمات کلیدی

الگوریتم MapReduce، بستر Hadoop، سیستم های پیشنهاد دهنده، مقیاس پذیری.

۱- مقدمه

امروزه شاهد رشد بسیار سریع حجم و ساختار اطلاعات در اینترنت هستیم. کاربران نیز بیش از گذشته در این محیط شلوغ و حجیم وب سایت‌ها به خاطر ساختار پیچیده و حجم زیاد اطلاعات گم می شوند. بنابراین، رعایت شخصی‌سازی و ساده سازی وب بیش از گذشته برای کاربران و صاحبان سایت‌های تجاری و فروشگاهای اینترنتی اهمیت یافته است [1]. یکی از موارد مهم شخصی‌سازی وب، سیستم‌های پیشنهاد دهنده به کاربر می‌باشد. چرا که سیستم در میان حجم عظیمی از اطلاعات و یا محصولات به کاربر پیشنهاداتی می‌دهد که او می‌پسندد و یا به آن نیاز دارد. به طور کلی سیستم-های پیشنهاد دهنده به سیستم‌ها و ابزارهایی گفته می‌شود که پیشنهادهایی برای آیتم های که کاربر از آنها استفاده می‌کند تهیه می‌کند. این پیشنهادات می تواند محصول، صفحه، خبر، کاربر مشابه و یا حتی تبلیغ باشد [2]. هر چقدر الگوریتم و مدل محاسبه گر و پردازشی سیستم پیشنهاد دهنده قوی‌تر و جامع‌تر باشد میزان رضایت‌مندی کاربر از آیتم‌های پیشنهادی و به طور کلی از سایت، بالاتر می‌رود و نهایتاً در سایت‌های تجارت الکترونیک باعث افزایش فروش محصولات می‌شود. سیستم‌های پیشنهاد دهنده نقش بسیار مهمی و حیاتی در سایت‌های پر بازدید و با کاربران و محصولات بالایی در زمینه های سرگرمی، محتوا، تجارت الکترونیک، سرویس‌دهی، تبلیغات و شبکه های اجتماعی و غیره مانند سایتهای netflix, youtube , amazon, Last.fm, imdb yahoo و ... ایفا می کند [3].

مقیاس‌پذیری الگوریتم‌های فعلی با اطلاعات و بانک‌های اطلاعاتی بزرگ امروزی به یکی از مهمترین چالش‌های سیستمهای پیشنهاد دهنده تبدیل شده است. آیتم‌های بانک‌های اطلاعاتی فعلی، مانند تعداد محصولات، کاربران و تعاملات کاربران مانند نظرها، امتیازها وغیره با سرعت بسیار زیادی در حال رشد و گسترش هستد و الگوریتم‌های فعلی در مقیاس بسیار محدودتر تست و ارزیابی شده‌اند و در مقیاسهای بزرگ کارایی خود را از دست می‌دهند [4].

در این مقاله یکی از روشهای قبلی که داری دقت مناسب در سیستم‌های پیشنهاد دهنده هستند مقیاس‌پذیر خواهد شد. در ادامه در بخش ۲ به تعریف مفاهیم و تکنیک‌های مورد استفاده در این مقاله می‌پردازیم. دربخش ۳ نگاهی خلاصه به کارهای گذشته می‌شود. در بخش ۴ روش مقیاس‌پذیر شده شرح داده می‌شود و در نهایت در بخش ۵ به آنالیز و اثبات مقیاس پذیری روش پیشنهادی پرداخته می‌شود.

۲- ادبیات مقاله

در کلی‌ترین تعریف، به هر نرم‌افزار، ابزار و یا تکنیکی که پیشنهادهایی در قالب آیتم برای کاربر تهیه کند سیستم‌های پیشنهاد دهنده گفته می‌شود. این سیستم‌ها خود به دو دسته محتوا محور و فیلترمشترک تقسیم می‌شوند.

۲-۱- انواع سیستم‌های پیشنهاددهنده

در محتوا محور سیستم یاد می‌گیرد که آیتم‌هایی را پیشنهاد دهد که مشابه آیتمهایی باشد که کاربر قبلا دوست داشته است. معیار شباهت در این روش ویژگی های خود آیتم‌ها هستند که با هم مقایسه می‌شوند. در روش فیلترمشترک، سیستم به کاربر، آیتم‌هایی را پیشنهاد می‌دهد که کاربرهای دیگر که شباهت زیادی با کاربر فعلی دارند آن آیتم را در گذشته پسندیده اند

(و یا برای مثال خریده اند). این روش ساده تر و محبوب‌تر از روش قبلی یعنی محتوا محور است. در اینجا معیار شباهت کاربران مهم تلقی می‌شود در کلی-ترین حالت این معیار بر اساس امتیاز گذشته کاربران به آیتم ها محاسبه می‌شود[5] .

۲-۲- الگوریتم MapReduce

هدف این پژوهش مقیاس‌پذیر کردن یکی از روش‌های معروف و با دقت بالا در زمینه سیستم های پیشنهاد دهنده است. یکی از مسیرهای رایج در مقیاس پذیرکردن، استفاده از پردازش موازی است و روش معروف در این زمینه روش MapReduce است. این الگوریتم یک مدل برنامه نویسی است که توسط گوگل معرفی شد. همانطور که از نام آن پیدا است، این الگوریتم از دو بخش اصلی به نام Map و Reduce تشکیل شده است.

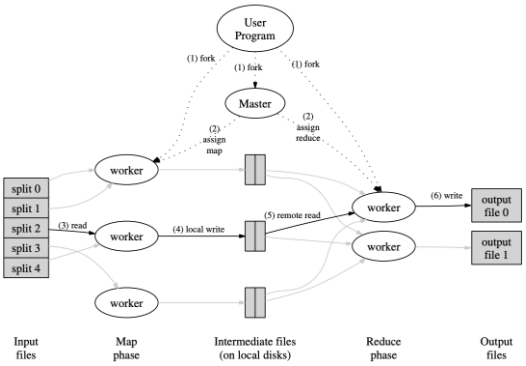
در گام اول تکنیک MapReduce، اطلاعات اولیه به بخش‌های کوچک و مساوی تقسیم می‌شود. هر کدام از این بخش ها به یک تابع Map داده می‌شود این تابع نیز مشخص می‌کند کدام جفت اطلاعات باید در کنار هم قرار گیرند. خروجی هر Map لیستی از رکوردها به صورت (k,v) است که هر رکورد از این لیست حاوی کلیدهای خاص k2 به همراه مقادیر مربوط به خودش می‌باشد.[6]

خروجی لیست‌ها تابع Map با هم ادغام شده و بر اساس کلیدها یا k2 مرتب می‌شود. حال برنامه لیست آن رکوردهایی را که کلید مشترک دارند (K2 مشترک) را به یک تابع Reduce جداگانه تحویل می‌دهد. در نهایت توابع Reduce بعد از محاسبات نهایی خروجی را در حافظه جانبی ذخیره می کنند [7]. در شکل ۱ و ۲ نحوه عملکرد این تابع آورد شده است.

Mapper: $\langle k1, v1\rangle \longrightarrow$ list $\langle k2, v2\rangle$

Reducer: $\langle k2, \text{list}\langle v2\rangle\rangle \longrightarrow$ list$\langle k3, v3\rangle$

۲شکل ۱- نمای کلی تابع MapReduce در [12]



شکل ۲ - نحوه عملکرد کلی تابعMapReduce در [6]

۲-۳- بستر Hadoop

در واقع Hadoop یک بستر و معماری برای اجرای MapReduce می باشد که به زبان جاوا و کد باز می‌باشد که توسط شرکت آپاچی ساخته شد. این بستر معروف‌ترین و محبوب‌ترین بستر برای پیاده‌سازی این الگوریتم است.

۳- پیشینه تحقیق

در ابتدا به بررسی اجمالی کارهای مهم انجام شده در زمینه سیستم‌های پیشنهاد دهنده پرداخته می‌شود. در [8] ابتدایی گام‌ها یک سیستم فیلتر مشترک به نام Tapestry توسط مرکز تحقیقات زیراکس توسعه داده شد. در [9] نیز سیستم فیلتر مشترکی بحت عنوان GroupLens ایجاد شد. در [10] سیستم فیلتر مشترکی بر اساس شبکه بیزین ایجاد شده است. در [11] بجای پیداکردن شباهت میان کاربران، شباهت میان آیتم‌ها بدست آورده می-شود و اثبات می‌کند که این روش آیتم محور دقت بالاتری نسبت به کاربر محور دارد. در [12] به خوشه‌بندی کاربران پرداخته می‌شود و شباهت کاربران را با سرخوشه آنها می‌سنجد تا دقت و سرعت را بالا ببرد. در [13] از روش-های داده کاوی برای تولید پیشنهادات خود استفاده می‌کند. در [14] نیز از روش داده کاوی به همراه نوع خاص از کلاس‌بندی محصولات برای افزایش دقت استفاده می‌شود. بخش دوم مقالات به بررسی الگوریتم‌های شباهت پرداخته شده است. یکی از رایج‌ترین روش های پیدا کردن شباهت روش پیرسون است. با این حال [15] یک روش بهینه شده پیرسون را ارائه کرده چرا که معتقد است بازه امتیازات داده شده در سیستم‌های پیشنهاد دهنده مطلق هستند. در [16] نیز روش پیرسون بهبود داده شده است و آن را روش پیرسون وزن دار نامیدند. در [17] محققین با استفاده از تابع سیگموئید، فرمول پیرسون را بهبود بخشیدند. بعضی کاربران در سیستم ، به آیتم‌هایی امتیاز بالایی می‌دهند در صورتی که زیاد آنها را دوست ندارند و بر حسب عادت ، امتیاز بالا می‌دهد و برعکس بعضی کاربران امتیاز بالا نمی‌دهند در حالی که آن آیتم ها را واقعا دوست دارند.برای حل و در نظر گرفتن این رفتار، [18] فرمولی تحت عنوان کسینوس تنظیم شده را ارائه دادند. در [19] فرمولی شباهتی تحت نام ژاکارد معرفی می‌کندکه بر این اساس شباهت را معیاری در رابطه تعداد مشترکات می‌داند. در [20] فرمولی تحت عنوان اختلاف میانگین مربع را معرفی می کند.این فرمول فقط مقدار مطلق امتیازات را در نظر می‌گیردو تعداد رای های مشترک را در نظر نمی‌گیرد.در نهایت در [21] تمامی معیار های مشابه را با هم مقایسه و مشکلات آنهارا بررسی می‌کند و سه معیار شباهت معرفی می‌کند که ادغام این سه نتیجه بسیار مطلوبی در شباهت دارد.در این پژوهش ما این مقاله را مقیاس‌پذیر کرده‌ایم. و در نهایت در بخش سوم، مقالاتی هستند که مانند این پژوهش یک روش از سیستم های پیشنهاددهنده را با استفاده از MapReduce مقیاس پذیر کرده اند. در[22] یک روش ساده کاربر محور را مقیاس پذیر می‌کند. در [23] روش آیتم محور را با الگوریتم، مقیاس‌پذیرمی‌کند. در [24] و [25] روش SVD از سیستم های پیشنهاد دهنده را به صورت محدود مقیاس‌پذیر کرده. اما در [26] از الگوریتم تقسیم و حل وپیاده سازی آن با MapReduce ان را مقیاس‌پذیرکرده.

۴- روش پیشنهادی

همانطور که در بخش قبلی به آن اشاره شد در [21] سه معیار شباهتی معرفی شد که با توجه به ارزیابی آن مقاله دقت بسیار خوبی دارد ولی به دلیل حجم بالای پردازش‌ها مقیاس‌پذیر نمی‌باشد. از این رو در این بخش ما طی ۳ مرحله در MapReduce که اصطلاحا به آن Jobگفته می‌شود آن را مقیاس پذیر می‌کنیم. در ابتدا فرمول‌های شباهت را به ترتیب بررسی می‌کنیم.
در ابتدا به شباهت PSS در فرمول شماره ۱ می پردازیم

$$Proximity\ (r_{u,p}, r_{v,p}) = 1 - \frac{1}{1+\exp\left(-|r_{u,p} - r_{v,p}|\right)}$$

$$Significance\ (r_{u,p}, r_{v,p}) = \frac{1}{1+\exp\left(-|r_{u,p} - r_{med}| \cdot |r_{v,p} - r_{med}|\right)}$$

$$Singularity\ (r_{u,p}, r_{v,p}) = 1 - \frac{1}{1+\exp\left(-\left|\frac{r_{u,p}+r_{v,p}}{2} - \mu_p\right|\right)}$$

$$PSS\ (r_{u,p}, r_{v,p}) = Proximity\ (r_{u,p}, r_{v,p}) \times Significance\ (r_{u,p}, r_{v,p}) \times Singularity\ (r_{u,p}, r_{v,p})$$

$$sim(u,v)^{PSS} = \sum_{p \in I} PSS(r_{u,p}, r_{v,p})$$

فرمول ۱ - فرمول‌های pss در [21]

در ادامه معیار JPSS و URP و در نهایت NHSM در فرمول ۳،۲ و ۴ آورده شده است.

$$sim(u,v)^{JPSS} = sim(u,v)^{PSS} \cdot sim(u,v)^{Jaccard}$$

فرمول ۲ - فرمول شباهت بر اساس PSS در [21]

$$sim(u,v)^{URP} = 1 - \frac{1}{1+\exp\left(-|\mu_u - \mu_v| \cdot |\sigma_u - \sigma_v|\right)}$$

فرمول ۳ - فرمول URP در [21]

$$sim(u,v)^{NHSM} = sim(u,v)^{JPSS} \cdot sim(u,v)^{URP}$$

فرمول ۴ - فرمول NHSM در [21]

برای دستیابی به NHSM باید پارامترهایی نظیر واریانس هر کاربر، میانگین امتیازات کاربر و هر محصول و تعداد امتیازات کاربر را برای حل معادلات بالا داشت از این رو job اول و دوم طبق شکل ۳ وظیفه محاسبه این پارامتر ها را دارد.

شکل ۳ - طرح کلی سه job این پژوهش

۴-۱- Job شماره صفر

در فاز Map، ابتدا ورودی که در این جا دیتاست آماده شرکت MovieLens است را به قطعات مختلف توسط Hadoop شکسته شده و هر کدام به گره‌های پردازشی مختلف داده می‌شود. هر قطعه شامل خطاهایی که هر خط شامل شماره کاربر، شماره فیلم و امتیاز کاربر است. در فاز Map این مرحله کلید میانی و مشترک برای مرحله Shuffle را شماره کاربر و مقدار آن را شماره فیلم و امتیاز کاربر مشخص کند. در مرحله Shuffle بسته های هر Map را گرفته و آن را بر اساس کلید آن که شماره کاربر است گروه بندی و مرتب سازی می کند و هر گروه با شماره کاربر منحصر به فرد را به هر Reduce می‌فرستد.
در فاز Reduce نیز جمع امتیازات، تعداد امتیازات، واریانس و لیست شماره هر فیلم کاربر را محاسبه می‌کند و خروجی نهایی مانند جدول ۱ است.

Userid	sum	count	variance	movie list
1	8.5	3	2	31,33,106
3	6	2	1	31,55
6	7	2	1	47,75

جدول ۱ - خروجی در فاز Reduce در Job 0

۴-۲- Job شماره یک

فاز Map این مرحله دو ورودی دارد یکی ورودی اولیه برنامه و دومی ورودی job صفرکه نمونه آن جدول ۱ است. هر ورودی به تابع Map جداگانه خود داده می‌شود.در Map نوع اول ورودی اولیه برنامه را گرفته و این بار بر خلاف Job صفر، شماره فیلم را به عنوان کلید میانی و مشترک برای مرحله Shuffle در نظر می‌گیرد و و مقدار ان را شماره کاربر، امتیاز آن و یک رشته type0 که در فاز Reduce برای تمییز این دو Map استفاده می‌شود. جدول ۲ نمونه این خروجی است.

key	value
movieid	["type0",userId,rate]
31	["type0",1 ,2.5]
31	["type0", 3,1.5]
88	["type0",1 ,3.0]
88	["type0", 6,4.5]
106	["type0",1 ,3.0]

جدول ۲ - خروجی در فاز نوع صفر Map در Job 1

در نوع بعدی از Map ورودی را از خروجی مرحله قبل می‌گیرد. کلید مشترک باز هم شماره فیلم ولی این بار با مقدار ان را با شماره کاربر، جمع امتیازات، تعداد امتیازات و واریانس پر می‌کند خروجی مانند جدول ۳ می‌شود.

key	value
movieid	["type1",userId,sum,count,variance]
31	["type1",1 ,8.8 ,3 ,2]
31	["type1", 3 ,6 ,2 ,1]
47	["type1", 3 ,6 ,2 ,1]
47	["type1", 6 ,7 ,2 ,1]

جدول ۳ - خروجی در فاز نوع یک Map در Job 1

در فاز Shuffle این دو خروجی جمع شده و بر اساس شماره فیلم گروه بندی شده و به فاز Reduceتحویل داده می‌شود. حال این فاز با در دست داشتن اطلاعات کلی کاربر و امتیازشان به هر فیلم، شروع به کنار هم گذاشتن هر کاربر در کنار هم و تشکیل جفت کاربری که در Job بعد به آن احتیاج داریم مانند جدول ۴ می‌دهد.

userid1	userid2	rate1	rate2	Count1	Count2	Variance1	Variance2	Sum1	sum2
1	3	2.5	1.5	3	2	2	1	8.8	6

جدول ۴ - نمونه یک خروجی از یک Reduce در Job 1

در جدول ۴ این رکورد مربوط به کاربر ۱ و ۳ درباره فیلم شماره ۳۱ بود. ممکن است این دو کاربر در فیلم های دیگر نیز امتیاز داده باشند که رکورد آنها با دیگر کاربران برای همین فیلم و یا فیلم های دیگر، توسط Reduce های دیگر تولید می‌شود دو نکته بدیهی است. یک اینکه ترتیب در اینجا مهم نیست مثلا شباهت ۱ و ۳ و شباهت ۳ و۱، که در این طرح جلوی آن گرفته شده است.و اینکه حالت هایی حساب می شود که هر دو کاربر به آن فیلم خاص امتیاز داده باشند نه همه حالات دو دویی کاربر. در این فاز هر Reducer دارای تعدادی جفت اطلاعات کاربر، که یک نمونه آن جدول ۴ است، می‌باشد. در اینجا دو استراتژی برا ذخیره این جفت اطلاعات میتوان داشت

۱. ذخیره هر جفت اطلاعات ، در یک خط جداگانه
۲. ذخیره تمامی جفت اطلاعات یک Reducer ، در یک خط

در حالت اول ، باید در نظر داشت که تعداد جفت اطلاعات بسیار زیاد است و ممکن است به دها هزار صد ها هزار و حتی میلیون ها رکورد شود و از آنجایی که ما در Job بعدی می‌خواهیم هر خط تحویل یک Map شود، در حالت اول تعداد بسیار زیادی Map ایجاد می‌شود.در حالت دوم نیز برای مثال اگر کاربران زیادی به فیلم امتیاز داده باشند، ممکن است که تعداد جفت اطلاعات در هر خط به چند صد و حتی چند هزار برسد. در اینجا ما می‌توانیم روش دوم را با کمی تغییر بهینه کنیم به این صورت که یک حداکثر برای تعداد جفت های هر خط تعیین کرده و اگر بیش از آن عدد شده جفت ها به خط های بعد بروند. نمونه آن در شکل ۴ آورده شده. بهترین عدد بر اساس تجربه این پژوهش عددی در نزدیکی، میانگین تعداد دفعات امتیاز دهی به هر آیتم است.

شکل ۴ - روش ذخیره سازی در job یک

۳-۴- Job شماره دو

حال در این مرحله چون برای هر جفت اطلاعات کامل را داریم، به محاسبه معیار های اصلی می‌پردازیم.در ابتدا برای هر جفت در فاز Map مقدار PSS آن را طبق فرمول گفته شده محاسبه می‌کنیم.سپس ما برای بدست آوردن معیار شباهت PSS به مجموع تمامی جفت هایی که دو کاربر دارند نیاز داریم.برای تحقق این امر ما نیاز داریم تمامی این جفت ها در مرحله Reduce یکجا جمع شده باشند به همین دلیل ما کلید مشترک برای مرحل shuffle را ترکیب شماره دو کاربر در نظر می‌گیریم. جدول ۵ را در نظر بگیرید.

key	value
user1,user2	[pair user1 , user2 :Mn] + [PSS(u1,u2)]
1,3	[pair 1 , 3 :M1] + [0.2]
1,3	[pair 1 , 2 :M2] + [0.4]

جدول ۵ - گروه بندی بر اساس کلید ترکیبی دو کاربر

در مرحله shuffle تمامی جفت کاربر هایی که برای مثال برای کاربر های ۱ و ۳ هستند به یک تابع Reduce داده می‌شود.حال در فاز Reduce با داشتن PSS هر جفت، معیار شباهت با جمع PSS های هر جفت بدست می‌آید همچنین معیار های jaccard نیز با محاسبه مشترکات،سپس معیار JPSS بدست می‌آید. معیار URP با داشتن واریانس و در مرحله آخر از ضرب دو معیار معیار نهایی NHSM بدست می آید. این نتایج مجددا در خروجی ذخیره می‌شود.

۵- ارزیابی

برای این پژوهش از ۵ سرور با ۸ گیگابایت رم، پردازنده ۴ هسته‌ای،۲۰ گیگابایت فضای دیسک سخت به عنوان گره پردازشی در Hadoop استفاده شده‌است. یکی از معیار های مهم در اثبات مقیاس پذیری معیار سرعت دادن می‌باشدکه به صورت فرمول ۵ است.

$$S_p = \frac{T_1}{T_p}$$

فرمول ۵ - معیار سرعت دادن

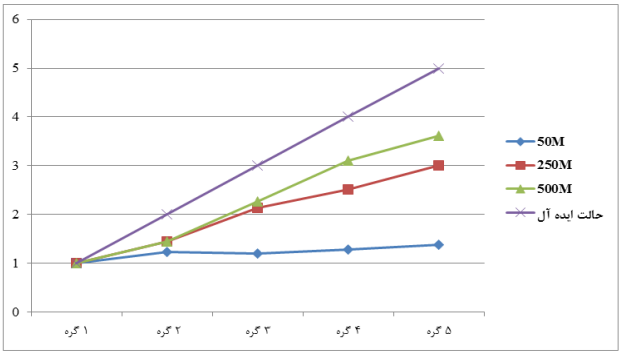
بر این اساس درصورتی فرایند ما مقیاس پذیر است که افزایش پردازنده یک رابطه خطی با معیار داشته باشد.در این پژوهش ما سه آزمایش با سه سایز دیتاست مختلف انجام دادیم.در هر بار آزمایش به ترتیب پردازش ها را با ۱، ۲، ۳ ، ۴ و ۵ نود کارگر طبق جدول ۶ انجام دادیم.

دیتاست	۱ نود	۲ نود	۳ نود	۴ نود	۵ نود
۵۰ مگابایت	۳۶~۳۴	۳۲~۲۸	۳۲~۲۷	۲۸	۲۶
۲۵۰ مگابایت	۱۴۶~۱۴۲	۱۰۲~۹۸	۶۹~۶۵	۵۸	۴۸
۵۰۰ مگابایت	۲۸۳~۲۸۰	۱۹۵~۱۹۳	۱۲۶~۱۲۲	۹۱~۸۸	۷۸

جدول ۶ - نتایج معیار سرعت دادن

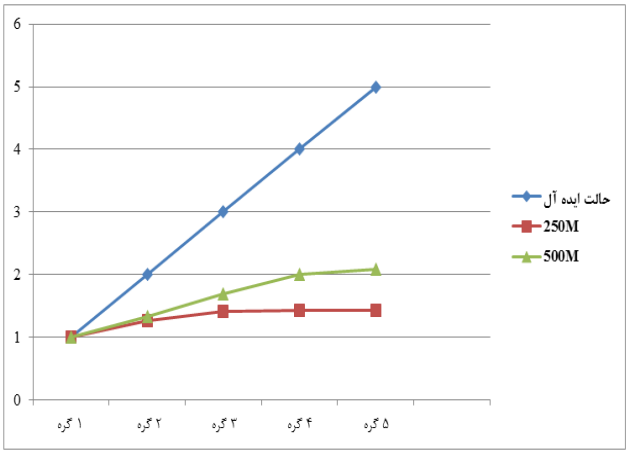
نتایج را بر اساس فرمول در جدول و نمودار ۱ قراردادیم. مقادیر همگی بر اساس ثانیه هستند و بعضی به صورت بازه آورده شده .چرا که

عوامل خطا مانند سرعت انتقال بین نود ها در متغییر بودن زمان پردازش موثر هستند. براساس فرمول هرچه نمودار به نیمساز شبیه‌ترباشد مقیاس‌پذیرتر است.



نمودار ۱ - معیار سرعت دادن

همانطور که در نمودار ۱ مشاهده می‌کنیم مقیاس پذیری با افزایش حجم بهتر می‌شود. برنامه Hadoop ورودی را به قسمت های مختلف تقسیم کرده و آن قسمت ها را بر اساس معیاری خاص به نود های کارگر بری پردازش ارسال می‌کند. نکته مهمی که باید در این قسمت در نظر گرف اینست که مقدار تقسیم بندی در بستر Hadoop قابل تنظیم است به صورتی که کمترین و بیشترین مقدار این تقسیم بندی را باید در فایل تنظیمات مشخص کنیم. در آزمایش قبلی کمترین مقدار ۱ مگابایت و بیشترین مقدار ۵ مگابایت بوده است. حال آزمایش دیگر با مقادیر جدید انجام داده به صورتی که حدهاقل مقدار تقسیم بندی ۵ مگابایت و حداکثر مقدار ۱۲ مگابایت بوده است. نتایج به صورت نمودار ۲ آمده است.



نمودار ۲ - تقسیم قطعات ۵ تا ۱۲ مگابایتی

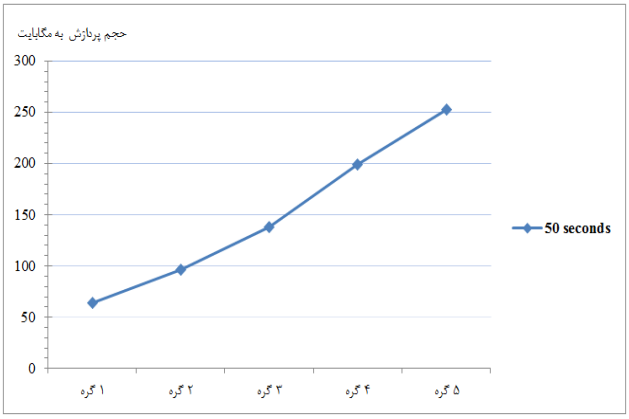
نکته مهم در نمودار ۲ ، اینست که در هر دو آزمایش چون تعداد قطعات ورودی کاهش می‌یابد(برای ۲۵۰ مگابایت ۲۱ قطعه و برای ۵۰۰ مگابایت ۴۳ قطعه)، در حالت های ۳ نود به بالا ، این قطعات به نود های ۴ و ۵ منتقل نمی‌شود و پردازش بین نود های ۱ ، ۲ و ۳ تقسیم شده بخاطر همین دلیل شیب نمودار و مقیاس پذیری آن برای بیش از ۳ نود ثابت مانده است.از خوبی های این تقسیم بندی افزایش سرعت پردازش می‌باشد.

از دیگر معیار های مهم این ارزیابی معیار isoefficiency است که این معیار بیانگر اینست چه مقدار کار باید افزایش یابد که بهره وری ثابت بماند فرمول این معیار به صورت فرمول ۶ آمده است.

$$P = \frac{l}{k} m .$$

فرمول ۶ - معیار isoefficiency

فرمول ۶ بیانگر اینست که سایز دیتاست m یک رابطه خطی با P تعداد نود ها دارد بنابراین شرط مقیاس پذیری اینست که با در نظر گرفتن زمان ثابت، سایز دیتاست و تعداد نود ها یک رابطه خطی داشته باشند.به همین منظور ما در این پژوهش زمان آجرای ثابت ۵۰ ثانیه ای را در نظر گرفتیم و محصاسبه کردیم با تعداد نود های مختلف چه حجمی توسط کل سیستم محاسبه می‌شود.



نمودار ۳ - معیار isoefficiency در زمان ۵۰ ثانیه

بر اساس نمودار ۳ و طبق اصل isoefficiency ، این برنامه و این طرح بر اساس MapReduce ، مقیاس‌پذیر است چرا که تعداد نود ها و حجم دیتاست پردازشی با یک نسبت تقریبا برابر افزایش پیدا می‌کند.

۶- نتیجه

در ابتدا مقدمه‌ای از سیستم های پیشنهادهنده، انواع آن و یکی از چالش اصلی آن یعنی مقیاس‌پذیری را شرح دادیم. سپس نگاهی به کارهای گذشته انداختیم. در بخش بعد طرح مقیاس‌پذیر بر اساس الگوریتم MapReduce بر بستر Hadoop برای یکی از معیار های مطرح شده با دقت بالا ارائه کردیم و در بخش ارزیابی با دو معیار اثبات کردیم که این طرح مقیاس پذیر است.

مراجع

[1] Ricci, F., Rokach, L., Shapira, B., Kantor, P.B., Recommender Systems Handbook, Springer US, 2011.

[2] Shah Khusro, ZafarAli, Irfan Ullah, "Recommender Systems: Issues, Challenges, and Research Opportunities," *Springer, Singapore*, vol. 376, no. Lecture Notes in Electrical Engineering, 2016.

[3] Paritosh Nagarnaik, Prof. A.Thomas, "Survey on Recommendation System Methods," in *2nd International Conference on Electronics and Communication Systems (ICECS)*, 2015.

[4] Bamshad Mobasher,Honghua Dai,Tao Luo,Miki Nakagawa, "Improving the Effectiveness of Collaborative Filtering," in *School of Computer Science, Telecommunication, and Information Systems*, DePaul University, Chicago, Illinois, USA, 2001.

[5] Schafer, J.B., Konstan, J.A. & Riedl, "E-Commerce Recommendation Applications," in *J. Data Mining and Knowledge Discovery,Kluwer Academic Publishers*, 2001.

[6] Jeffrey Dean,Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Communications of the ACM - 50th anniversary issue: 1958*, vol. 51, no. 1, January 2008 , pp. 107-113, 2008.

[7] J Dean, S Ghemawat, "MapReduce: a flexible data processing tool," *Communications of the ACM - Amir Pnueli: Ahead of His Time*, vol. 53, pp. 72-77, 2010.

[8] David Goldberg, David Nichols, Brian M. Oki and Douglas Terry, "Using collaborative filtering to weave an information Tapestry," *Communications of the ACM*, vol. 35, pp. 61-70, 1992.

[9] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, "GroupLens: an open architecture for collaborative filtering of netnews," in *CSCW '94 Proceedings of the 1994 ACM conference on Computer*, Chapel Hill, North Carolina, USA, 1994.

[10] John S. Breese David Heckerman Carl Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *UAI'98 Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, Madison, Wisconsin, 1998.

[11] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, "Item-Based Collaborative Filtering Recommendation," in *Proceedings of the 10th international conference on World Wide Web*, Hong Kong, 2001.

[12] Feng-Hsu Wanga, Hsiu-Mei Shaob, "Effective personalized recommendation based on time-framed navigation clustering and association mining," *Expert Systems with Applications*, vol. 27, pp. 365-377, 2004.

[13] Yoon Ho Choa,Jae Kyeong Kimb, Soung Hie Kima, "A personalized recommender system based on web usage mining and decision tree induction," *Expert Systems with Applications*, vol. 23, pp. 329-342, 2002.

[14] Hadi Khosravi Farsani, and Mohammadali Nematbakhsh, "A Semantic Recommendation Procedure for Electronic Product Catalog," *World Academy of Science, Engineering and Technology 22*, 2006.

[15] Upendra Shardanand,Pattie Maes, "Social Information Filtering: Algorithms for Automating "Word of Mouth"," in *CHI '95 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Denver, Colorado, USA, 1995.

[16] JL Herlocker, JA Konstan, A Borchers,John Riedl, "An algorithmic framework for performing collaborative filtering," in *SIGIR '99 Proceedings of the 22nd annual international ACM SIGIR conference*, Berkeley, California, USA, 1999.

[17] Mohsen Jamali,Martin Ester, "TrustWalker: A Random Walk Model for Combining," in *the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, Paris, France, 2009.

[18] HJ Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Information Sciences*, vol. 178, pp. 37-51, 2008.

[19] G Koutrika, B Bercovitz, H Garcia-Molina, "FlexRecs: expressing and combining flexible recommendations," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management*, New York, 2009.

[20] F Cacheda, V Carneiro, D Fernández, "Comparison of Collaborative Filtering Algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems," *ACM Transactions on the Web (TWEB)*, vol. 5, 2011.

[21] H Liu, Z Hu, A Mian, H Tian, X Zhu, " A new user similarity model to improve the accuracy of collaborative filtering," *Knowledge-Based Systems*, vol. 56, pp. 156-166, 2014.

[22] Zhi-Dan Zhao ; Ming-sheng Shang, "User-based Collaborative-Filtering Recommendation Algorithms on Hadoop," in *Third International Conference on Knowledge Discovery and Data Mining*, Phuket, Thailand, 2010.

[23] Jing Jiang, Jie Lu, Guangquan Zhang, Guodong Long, "Scaling-up Item-based Collaborative Filtering Recommendation Algorithm," in *IEEE World Congress on Services*, 2011.

[24] B Bayramli, "SVD Factorization for Tall-and-Fat Matrices on Map/Reduce Architectures," in *eprint arXiv:1310.4664*, 2013.

[25] DF Gleich, PG Constantine, "Tall and Skinny QR factorizations in MapReduce architectures," in *the second international workshop on MapReduce and its applications*, New York, NY, USA, 2011.

[26] Shuoyi Zhao, Ruixuan Li, Wenlong Tian, Weijun Xiao, Xinhua Dong, "Divide-and-conquer approach for solving singular value decomposition based on MapReduce," in *Concurrency and Computation Practice and Experience*, 2014.