ARYA VARASTE

۱۴۰۰/۰۹/۲۸

بازیابی هوشمند اطلاعات

تمرین سوم

a.varaste.n@gmail.com

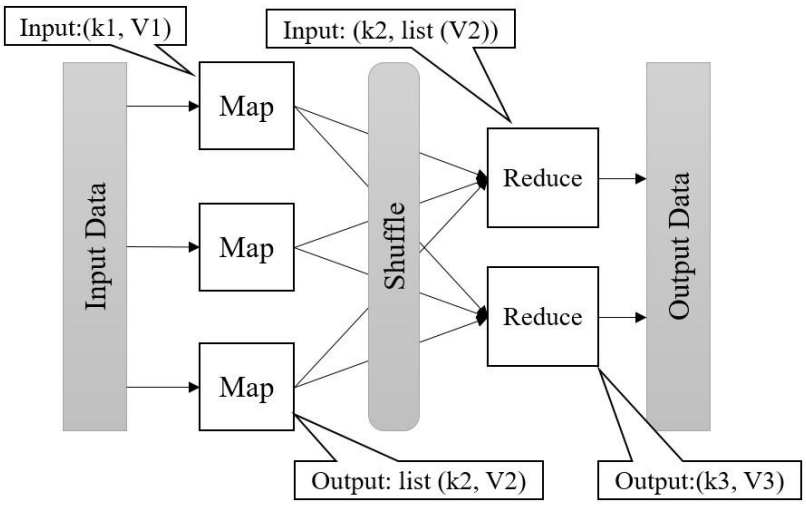
بخش اول

مدل برنامه نویسی Map Reduce جهت انجام عملیات پردازشی موازی بر روی خوشه‌ای از کامپیوترها کاربرد دارد. در واقع این مدل برنامه نویسی، خود از دو قسمت Map یا همان نگاشت و Reduce یا همان کاهش تشکیلی شده است. نگاشت-کاهش روشی است که به صورت گسترده‌ای درحل مسایل کلان داده مورد استفاده قرار می‌گیرد. اما اشکالات و نقایص فراوانی در این روش وجود دارد. عدم وجود تمام داده‌های مورد نیاز برای پردازش برروی هرگره، ایجاد گلوگاه در شبکه، عدم استفاده از تمام ظرفیت پردازشی و حافظه‌ای هر گره از جمله مهمترین اشکالات این روش می‌باشد که باعث بالا رفتن زمان اجرای الگوریتم می‌گردد.

این روش مجموعه‌ای از جفت‌های کلید/مقدار ورودی را می‌گیرد و مجموعه‌ای از جفت‌های کلید/مقدار خروجی را تولید می‌کند. کاربر محاسبات را به صورت دو تابع بیان می کند: تابع نگاشت و تابع کاهش.

تابع نگاشت، یک جفت ورودی می‌گیرد و مجموعه‌ای از جفت‌های کلید/مقدار میانی را تولید می‌کند. MapReduce تمام مقادیر میانی مرتبط با کلید میانی یکسان I را گروه بندی می‌کند و آن‌ها را به تابع کاهش می‌دهد.

تابع کاهش، یک کلید میانی I و مجموعه ای از مقادیر را برای آن کلید می‌پذیرد. این مقادیر را با هم ادغام می‌کند تا مجموعه‌ای از مقادیر احتمالاً کوچکتر را تشکیل دهد. معمولاً فقط صفر یا یک مقدار خروجی در هر احضار کاهش تولید می‌شود. مقادیر میانی از طریق یک تکرار کننده به تابع کاهش کاربر ارائه می شود. این به ما اجازه می‌دهد تا لیست‌هایی از مقادیری را مدیریت کنیم که آنقدر بزرگ هستند که در حافظه جای نمی‌گیرند.



برتری نگاشت‌کاهش، در این است که اجازه می‌دهد تا پردازش عملیات نگاشت و کاهش توزیع‌شود. فراهم آوردن این امر که هر کدام از این نگاشت‌ها مستقل از دیگران است، که خود متضمن اجرای موازی این نگاشت‌هاست. اگرچه این گفته در عمل به این صورت خواهد بود که محدود به منابع داده یا تعداد پردازنده‌های نزدیک به آن داده‌است. به صورت مشابه، مجموعه‌ای از کاهنده‌ها می‌توانند فاز کاهش را به انجام رسانند. لازمه این امر آن است که خروجی

عملیات نگاشت کلیدی یکسان را در یک زمان به همه کاهنده‌ها ارسال نماید. این روش برای الگوریتم‌هایی که به صورت دنباله‌ای از دستورهای غیرقابل موازی سازی هستند، ناکارآمد است. نگاشت‌کاهش بر روی مجموعه‌های عظیم داده‌ای بهتر جواب می‌دهد تا سرورهای تجاری. مجموعه‌های عظیم داده‌ای را می‌توان به مزارع سرور تعمیم داد. مزارعی که حجمی به بزرگی چندین پتابایت داده را در کسری از ساعت، پردازش می‌نماید. همچنین موازی‌سازی امکان بازسازی بعد از بروز خطایِ جزئی در سرورها را در طول عملیات فراهم می‌آورد: اگر یکی از نگاشت‌کنندگان یا کاهندگان دچار خطا شود، کار دوباره زمان‌بندی خواهدشد- با فرض اینکه داده‌همچنان در دسترس باشد.

یکی از روش‌ها برای بررسی میزان ارتباط کلمات شمارش رویداد همزمان[1] است. اما این روش اشکالاتی دارد. روش بهتر استفاده از فرکانس نسبی[2] کلمات است.

$$f(B \mid A) = \frac{\text{count}\ (A, B)}{\text{count}\ (A)}$$

سوال یک

در روش Stripes، جفت‌های کلمه همزمان توسط دو حلقه تودرتو ایجاد می‌شوند. به جای انتشار جفت‌های کلید-مقدار میانی برای هر جفت کلمه همزمان، اطلاعات همزمان در ابتدا در یک نوار H ذخیره می‌شود. نگاشت کننده جفت‌های کلید-مقدار را با کلمات به عنوان کلید و آرایه های انجمنی متناظر به عنوان مقادیر، که در آن هر آرایه انجمنی تعداد هم رخدادهای همسایگان یک کلمه خاص را انکد می‌کند. چارچوب اجرای MapReduce تضمین می‌کند که تمام آرایه های انجمنی با کلید یکسان در مرحله کاهش پردازش گرد هم می‌آیند. در انکدر یک مجموع عنصری از همه آرایه‌های انجمنی را با کلید یکسان انجام می‌دهد و تعدادهایی را جمع‌آوری می‌کند که مربوط به همان سلول در ماتریس هم‌رویداد هستند. آرایه انجمنی نهایی با همان کلمه کلید منتشر می شود.

سوال دو

در روش Pairs شناسه سند و محتویات سند جفت‌های کلید-مقدار ورودی را تشکیل می‌دهند. تابع نگاشت هر سند ورودی را پردازش می‌کند و جفت‌های کلید-مقدار میانی را با هر جفت کلمه همزمان به عنوان کلید و عدد صحیح به عنوان مقدار منتشر می‌کند. این به طور مستقیم توسط دو حلقه تو در تو انجام می شود:

حلقه بیرونی روی همه کلمات (عنصر سمت چپ در جفت) و حلقه داخلی روی همه همسایگان کلمه اول (عنصر سمت راست در جفت) تکرار می شود. همسایه‌های یک کلمه را می‌توان بر حسب یک پنجره یا برخی واحدهای متنی دیگر مانند جمله تعریف کرد. چارچوب اجرای MapReduce تضمین می‌کند که تمام مقادیر مرتبط با یک کلید در یک کاهنده گرد هم آمده‌اند. بنابراین، در این مورد کاهنده به سادگی تمام مقادیر مرتبط

[1] Co-Occurrence Counts
[2] Relative Frequencies

با یک جفت کلمه همزمان را جمع‌بندی می‌کند تا به تعداد مطلق رویداد مشترک در پیکره برسد، که سپس به عنوان جفت کلید-مقدار نهایی منتشر می‌شود.

سوال سه

بدیهی است که الگوریتم Pairs در مقایسه با روش Stripes، تعداد زیادتری جفت کلید-مقدار می‌سازد. متد Stripes بسیار فشرده‌تر است، زیرا جفت عنصر سمت چپ برای هر کلمه همزمان تکرار می‌شود. متد Stripes همچنین کلیدهای میانی کمتر و کوتاه‌تری می‌ساز، بنابراین چارچوب مرتب‌سازی کمتری انجام می‌دهد. اما، مقادیر در رویکرد Stripes پیچیده‌تر هستند و با سربار بیشتری نسبت به رویکرد Pairs همراه هستند.

در مقایسه این روش‌ها باید به مقیاس پذیری اهمیت ویژه‌ای داد. متد Stripes این فرض را در نظر دارد که، در هر نقطه از زمان، هر نوار به اندازه کافی کوچک است که در حافظه قرار گیرد، در غیر این صورت، صفحه‌بندی حافظه به طور قابل توجهی بر عملکرد تأثیر می‌گذارد. اندازه آرایه انجمنی با اندازه واژگان محدود می‌شود. بنابراین، اما رویکرد Pairs از این محدودیت رنج نمی‌برد، زیرا نیازی به نگهداری داده‌های میانی در حافظه ندارد.

برای داده‌های در اندازه مجموعه‌هایی که در این تمرین در اختیار ما قرار گرفت روش Stripes هم از لحاظ حافظه و نیز از نظر سرعت اجرا بهتر عمل می‌کند. در زیر مقایسه‌ای از این دو متد ارائه شده است:

Pairs	Stripes
آسان برای درک و پیاده سازی	درک و پیاده‌سازی دشوارتر
تعداد زیادی جفت کلید-مقدار ایجاد می‌کند	جفت های کلید-مقدار کمتری ایجاد می‌کند
حافظه زیادی جفت کلید-مقدار مصرف می‌کند	اندازه حافظه آرایه‌های انجمنی در نگاشت‌ها می‌تواند زیاد باشد
زمان اجرای بیشتر	زمان اجرای کوتاه‌تر
	از ترکیب کننده‌ها بهتر استفاده می‌کند

بخش دوم

سوال یک

" یک نود را بن بست می‌گوییم اگر یا هیچ یال خروجی نداشته باشد و یا همه یال‌های خروجی آن به یک نود بن‌بست متصل باشند." وجود بن بست میتواند باعث گیرکردن خزنده شده و عملکرد آن را مختل سازد. جهت کشف بن بست‌ها مجموعه گره‌ها را به دو دسته تقسیم می‌کنیم:

۱) آن‌هایی که هیچ لینک خارج شونده‌ای ندارند
۲) سایر نودها بجز گروه اول

پس از این برای هر نود از مجموعه نودهای گروه دوم بررسی می‌کنیم که آیا نودی وجود دارد که همه لینک‌های خارج شده از آن به گره‌های دسته اول رفته باشند؟ اگر نودی این‌چنین یافت شد، آن را به مجموعه نودهای گروه اول اضافه می‌کنیم و عملیات را مجدداً با مجموعه‌های جدید تکرار می‌کنیم.

با استفاده از کتابخانه NetworkX از فایل متنی داده شده گراف را شکل داده (inputgraph) و گراف حاصل شده را به تابع deadlock می‌دهیم.

سوال دو

PageRank تابعی است که به هر صفحه در وب یک عدد اختصاص می‌دهد، با این هدف که به مهم‌ترین صفحات، بالاترین امتیازات داده شود.

با استفاده از کتابخانه NetworkX از فایل متنی داده شده گراف را شکل داده و گراف حاصل شده را به تابع prank می‌دهیم. حاصل را در result ریخته و در فایل‌های pr_10k.csv و pr_800k.csv ذخیره می‌کنیم.

سوال سه

چون وجود بن بست می‌تواند باعث گیرکردن خزنده شده و عملکرد آن را مختل سازد، با حذف کردن نودهای بن بست، عملکرد الگوریتم پیج‌رنک بهتر می‌شود.