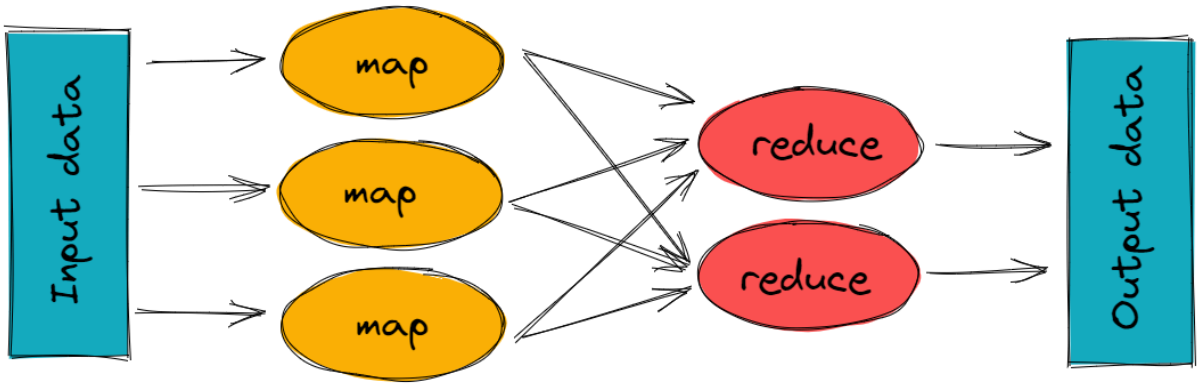
مقدمه

در این پروژه می خواهیم با نحوه مدیریت کردن پردازه[1] ها و روش های ارتباطی آنها آشنا شویم. در این تمرین با بهره گیری از عملیات هایی در سطح پردازه‌ها محاسباتی را روی داده‌هایی از کالاهای در حال گردش در انبارها انجام خواهید داد.

مدل نگاشت کاهش[2]



در دنیای امروز، به دلیل گسترش اینترنت و دستگاه های هوشمند، روزانه حجم زیادی از داده تولید می شود. در گذشته، داده های تولیدی قابلیت ذخیره و اجرا بر روی یک دستگاه سخت افزاری را داشتند اما امروزه برای بسیاری از موارد این امر غیرممکن است. نگاشت کاهش یک چارچوب و مدل برنامه نویسی است که اجازه اجرای پردازش موازی و توزیع شده بر روی مجموعه بزرگی از داده ها در یک محیط توزیع یافته را می دهد و بسیاری از مفاهیم آن از زبان های تابع‌گرا[3] مانند Lisp گرفته شده است. در ادامه به توضیح این مدل خواهیم پرداخت:

Map-Reduce از دو بخش نگاشت (Map) و کاهش (Reduce) تشکیل می‌شود.

- در قسمت نگاشت، تعدادی پردازه برای عمل نگاشت وجود دارد که کاملا مستقل از یکدیگر عمل می‌کنند و هیچ ارتباطی با یکدیگر ندارند. خروجی این مرحله تعدادی key-value

[1] Process
[2] Map-Reduce
[3] Functional

خواهد بود که برای استفاده به قسمت کاهش ارسال می‌شود. تعداد پردازه های قسمت نگاشت محدودیت خاصی ندارد و می‌تواند بر اساس منابع در دسترس و نوع داده‌ها انتخاب شود. هر کدام از پردازه های قسمت نگاشت به صورت موازی اجرا می‌شوند.

- در قسمت کاهش، خروجی‌های مرحله قبل به عنوان ورودی دریافت می‌شود و سپس بر اساس کلید، داده‌ها تقسیم می‌شوند. داده‌هایی که کلید یکسان دارند، حتما باید به یک پردازه داده شوند. هر پردازه کاهش، بر روی مجموعه ی داده‌های با کلید یکسان، عملیات مورد نظر را انجام می‌دهد و خروجی را ایجاد می‌کند.

شرح پروژه

در این پروژه قرار است به روش نگاشت-کاهش، موجودی کالاهای موجود در انبارهای یک شرکت پخش مواد غذایی و همچنین سود حاصل از فروش آن‌ها در این دوره انبارداری را محاسبه کنید.

نحوه پیاده‌سازی

برنامه شما باید این امکان را برای یک شرکت پخش مواد غذایی فراهم کند که بتواند موجودی کالاهای خود را که در انبار شهرهای مختلف وجود دارد بدست آورد و همچنین سود خود را از فروش این کالاها محاسبه کند. برای این منظور یک پوشه به نام stores به شما داده می شود که حاوی تعدادی فایل csv است. هر فایل نشان دهنده گزارش‌های انبار یک شهر است. در هر کدام از این فایل‌ها اطلاعات ورود و خروج کالاها به انبار نوشته شده است. این اطلاعات حاوی نام کالا، قیمت یک واحد، مقدار و نوع گردش (ورود/خروج) می‌باشد.

برای مثال Tehran.csv می‌تواند به شکل زیر باشد :

berenj,100,30000,input

roghan,50,50000,output

shekar,80,20000,input

makaroni,60,15000,input

...

فرض می‌شود واحد سنجش کالاها اهمیتی ندارند.

همچنین یک فایل csv دیگر با نام parts.csv در پوشه stores قرار دارد که یک خط است و نام تمامی کالاهای موجود در انبارها در آن نوشته شده است.

نحوه پیاده سازی به این صورت است که در ابتدا پردازه اولیه وجود دارد که آدرس پوشه stores در آرگومان ورودی به آن داده می شود و به ازای **هر انبار یک پردازه جدید** ایجاد می کند. حال پردازه هر انبار، باید با خواندن فایل csv متناظر و پردازش داده‌ها، موجودی هر کالا در آن انبار را محاسبه کند (منظور از موجودی باقیمانده آن کالا در انبار است) و به پردازه متناظر کالا که قرار است موجودی آن کالا در تمام انبارها را حساب کند انتقال دهد. موجودی هر کالا شامل موجودی تعدادی و موجودی ریالی می باشد.همچنین هر پردازه انبار باید سود کلی حاصل از این دوره انبارداری را محاسبه کند و به پردازه اصلی انتقال دهد. دقت کنید ترتیب خروج کالاها از انبار به ترتیب ورود آنها می باشد. منظور این است که اگر دو مرتبه محصول برنج وارد انبار شده است و اکنون سفارش برنج برای انبار آمده است **ابتدا از ورودی اولیه باید سفارش تامین شود در صورت اتمام نوبت به ورودی دوم می رسد.** این نکته در محاسبات سود نهایی مهم است.

همانطور که در بالا اشاره شد **به ازای هر نوع کالا، یک پردازه جداگانه** خواهیم داشت که **توسط پردازه اولیه ایجاد می‌شود** و موجودی تعدادی و ریالی کل آن کالا در انبارهای شرکت را با اطلاعاتی که از پردازه های انبارهای مختلف می گیرد محاسبه می کند. توجه داشتید باشید نام و تعداد کل کالاها در ابتدا نامعلوم است و باید با خواندن فایل parts.csv و در پردازه اولیه، این اطلاعات را بدست آورید. همچنین پردازه هر انبار گزارشی مربوط به سود کالاهای خواسته شده توسط کاربر در انبار خودش را به پردازه پدرش (یعنی پردازه اصلی) گزارش می‌کند.

در انتها پردازه‌های کالاها و پردازه‌های انبارها می بایست مقادیر نهایی خود را برای گزارش نهایی **تحویل پردازه پدر** خود یعنی پردازه اولیه دهند.

توجه داشته باشید که برنامه شما به صورت تعاملی[4] است؛ به صورتی که اینکه اطلاعات خواسته شده برای چه کالاهایی گزارش شوند، **توسط کاربر مشخص می‌شود**. نحوه گرفتن ورودی از کاربر و نمایش اطلاعات محاسبه شده در ترمینال بر عهده خودتان است.

[4] interactive

پس از اتمام پردازش پردازه‌ها و چاپ گزارش، برنامه پایان می یابد. دقت کنید که اتمام پردازه‌های فرزند حتما قبل اتمام پردازه اولیه و برنامه اصلی انجام شود. همچنین تضمین می‌شود کاربر ورودی معتبر و با فرمت درست وارد خواهد کرد.

دقت کنید که برای انتقال اطلاعات لازم بین هر دو پردازه **پدر و فرزند** باید از **unnamed pipe** استفاده شود. برای انتقال اطلاعات مورد نیاز بین دو پردازه که ارتباط **پدر-فرزندی ندارند**، از **named pipe** استفاده می‌شود.

نکات تکمیلی

- برای ساخت پردازه‌ها توسط پردازه اصلی، حتما از فراخوانی های سیستمی fork و exec برای ساخت و اجرای آنها استفاده کنید.
- دقت کنید به ازای هر نوع پردازه‌ای که در برنامه ایجاد می‌شود، باید حداقل یک فایل cpp مربوط به آن پردازه در فایلهای پروژه شما وجود داشته باشد منطق آن پردازه در آن پیاده سازی شده است.
- فرمت انتقال داده ها میان پردازه‌ها بر عهده خودتان است.
- بعد از استفاده از pipe ها، آنها را ببندید.[5]
- به دلیل استفاده از نوع pipe ها در هر مرحله فکر کنید. در زمان تحویل سوالاتی در این باره پرسیده خواهد شد.
- دقت شود **تنها راه ارتباطی میان پردازه ها استفاده از pipe است** و هیچ راه دیگری قابل قبول نیست. **اطلاعاتی که از طریق آرگومان به پردازه‌ها منتقل می‌شود باید بسیار ناچیز باشد.**
- هیچ نوع دیگری از پیاده سازی بجز مدلی که در بالا توضیح داده شد قابل قبول نیست.
- برای هر گونه ارتباط و پیام ارسالی بین پردازه‌ها باید log مناسب در مکان مناسبی چاپ شود تا روند اجرای برنامه قابل بررسی و صحت‌سنجی باشد. برای مثال ترتیب بسته شدن پردازه‌ها صرفا از طریق log قابل بررسی خواهد بود. همچنین در صورت عدم ذخیره log مناسب رفع مشکلات احتمالی برای خود شما نیز سخت خواهد شد.
- رابط کاربری مناسب و قابل درک برنامه شما (به گونه ای که کسی که کد شما را ندیده هم بتواند آن را اجرا کند و نتیجه بگیرد) بخشی از نمره پروژه را تشکیل می‌دهد و حتما به آن دقت شود.

[5] close()

نکات پایانی و نحوهٔ تحویل

- برنامه حتما باید با استفاده از makefile و کامپایلر g ++ کامپایل و اجرا شود.
- برنامه باید در سیستم عامل لینوکس و در زمان معقول اجرا شود.
- تمامی نتایج را در یک فایل فشرده با اسم OS_CA2_<#SID>.zip در محل بارگذاری درس آپلود کنید.
- انجام این پروژه به صورت انفرادی است.
- نکاتی که در جلسه توجیهی یا گروه اسکایپ درس مطرح می شوند بخشی از صورت پروژه هستند لذا توصیه می شود که شرکت کنید.
- در صورت داشتن هرگونه سوال با مسئولان این پروژه در ارتباط باشید.

➢ soheilhm1381@gmail.com

➢ alireza.hosseini.81@gmail.com

موفق باشید!