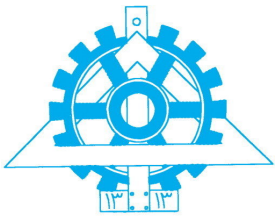
تمرین کامپیوتری سوم



دانشکده مهندسی برق و کامپیوتر | **گزارش کار** | سیستم‌های عامل - بهار 1400

استاد:
دکتر مهدی کارگهی

تاریخ:
۱۴۰۰/۰۳/۰۵

نام و نام خانوادگی:
دانشور امراللهی

مقدمه

در این تمرین شما به تحلیل داده‌هایی که از مشخصات و قیمت فروش خانه ها جمع‌آوری شده‌است پرداخته شده است. در ابتدا برنامه اقدام به خواندن و تجزیه مجموعه داده[1]ی ارائه شده کرده و آنها را در حافظه خود ذخیره می‌کند. پس از استخراج داده‌ها و ویژگی‌های آنها، برنامه اقدام به برچسب گذاری ستون قیمت داده‌ها، بدست آوردن میانگین و انحراف معیار داده ها و در نهایت اقدام به تعیین رده قیمتی خانه ها می‌کند. این تمرین به دو روش این مسئله پیاده‌سازی شده است که در ادامه گزارش، نتایج حاصل آمده است.

[1] Dataset

پیاده‌سازی سری

سوال اوّل

چرا برای پیاده‌سازی یک برنامه بصورت چندریسه‌ای، بهتر است ابتدا این برنامه بصورت سری پیاده‌سازی شود؟

پیاده‌سازی به صورت سری، به ما کمک می‌کند بخش‌هایی از برنامه که می‌توانند به صورت موازی اجرا شوند را راحت‌تر پیدا کنیم. همچنین از آنجا که زمان اجرای برنامه در حالت موازی برای ما مهم است، بهتر است معیاری برای مقایسه داشته باشیم تا میزان تسریع را متوجه شویم که آن معیار می‌تواند پیاده‌سازی سری باشد.

سوال دوم

با بررسی زمان اجرای بخش‌های مختلف برنامه، Hotspot[2] های برنامه را مشخص کنید.

با استفاده از کتابخانه time.h و فراخوانی متد ()clock در بخش‌های کوچک برنامه، متوجه می‌شویم که زمان‌برترین بخش برنامه ورودی خواندن از فایل‌های csv است. محاسبات برنامه عملیات بسیار ساده‌ای دارند (جمع و ضرب عددهای گویا) که پیچیدگی خاصی ندارد. دیسک multithreaded نمی‌باشد.

جدول اوّل

زمان‌های اجرای ۶ اجرای متوالی از برنامه و میانگین آن‌ها را به ازای ورودی نمونه‌ای که در شرح تمرین آمده است، در جدول زیر بیاورید (فایل csvداده‌شده گسترش‌داده شده تا ۱۰۰۰۰۰۰ سطر)

اجرای اوّل	اجرای دوم	اجرای سوم	اجرای چهارم	اجرای پنجم	اجرای ششم	میانگین
0m1.911s	0m1.856s	0m1.971s	0m1.904s	0m1.870s	0m1.963s	0m1.912s

[2] توابعی که در برنامه‌تان بیشترین زمان اجراها را به خود اختصاص می‌دهند.

پیاده‌سازی چندریسه‌ای

سوال سوم

اگر هنگام موازی‌سازی برنامه به زمان اجرای بیشتری نسبت به حالت سری برخورد کنید، چه رویکردهایی را برای کاهش زمان اجرا و استفاده حداکثری از موازی‌سازی پیش می‌گیرید؟

تا جای ممکن سعی می‌کنیم mutex lock ها را کاهش دهیم. هنگام ساخت یک thread، جایی که می‌تواند (نیازی به پاسخ جواب thread های دیگر که والدش ساخته نداشته باشد که بخواهد join شوند همه) پیش برود. تعداد thread ها را تغییر می‌دهیم (هر دو حالت افزایش یا کاهش) و زمان برنامه را اندازه می‌گیریم.

سوال چهارم

در هنگام پیاده‌سازی این بخش، به چه چالش‌هایی برخورد کردید و بیان کنید که به چه صورت آن‌ها را رفع کردید.

مهم‌ترین چالشی که به آن برخوردم، بیشتر بودن زمان برنامه سری از موازی بود. ابتدا با اضافه کردن متغیرهای گلوبال، تمام mutex lock ها را حذف کردم. سپس تعداد دفعاتی که thread جدید می‌ساختم و منتظر join می‌شدم را کاهش دادم. اما همچنان به جواب مورد نظر نرسیده بودم که پس از مدت زیادی وقت گذاشتن و سرچ از وبسایت stackoverflow، متوجه شدم که bottleneck برنامه من در بخش ورودی خواندن از فایل‌هاست نه از بخش محاسبات. برای مشاهده اثر موازی‌سازی، فایل‌های csv را با کپی گرفتن از خودشان و افزودن به خودشان، به ۱۰۰۰۰۰۰ رساندم و با استفاده از یک برنامه CSVParser که نوشتم، این فایل را ۴ فایل شکاندم تا برای یک پردازه ۴ ریسه‌ای استفاده شوند. حالا با مقایسه زمان‌ها متوجه شدم که برنامه سری من تقریبا ۳.۲۸ برابر برنامه موازی، زمان اجرا دارد که عددی نزدیک به ۴ (تعداد ریسه‌ها) است و از صحت برنامه خود مطمئن شدم.

همچنین برای افزایش دقت سنجش اجرای زمان برنامه، پیش از هر دور اجرا، cache را پاک می‌کردم که تسریعی در خواندن فایل رخ ندهد.

سوال پنجم

با توجه به تجربه‌ای که در پیاده‌سازی این تمرین بدست آوردید، به نظر شما در چه مواقعی استفاده از مکانیزم های همگام سازی مانند قفل[3] در یک طراحی چندریسه‌ای ضروری است؟ در این پروژه از چه مکانیزم هایی استفاده کرده اید؟ آیا استفاده از این مکانیزم ها می تواند اثر مخربی داشته باشد؟

مثلا هنگامی که یک متغیر توسط چند ریسه ممکن است مقدارش تغییر کند، باید از lock استفاده کنیم. برای مثال هنگام ++ کردن یک متغیر، اگر lock انجام ندهیم، با بررسی این عمل +۱ کردن در سطح اسمبلی متوجه می‌شویم که ممکن است CPU از ریسه فعلی گرفته شود و این عملیات ناقص بماند. بنابراین عمل ++ انجام نشده باقی می‌ماند در صورتی که برنامه به این فرض نوشته شده که این عمل به درستی انجام می‌شود.

استفاده زیاد از این قفل‌ها زمان اجرای برنامه را زیاد می‌کند.

جدول دوم

زمان‌های اجرای ۶ اجرای متوالی از برنامه و میانگین آن‌ها را به ازای ورودی نمونه‌ای که در شرح تمرین آمده است، در جدول زیر بیاورید.

اجرای اوّل	اجرای دوم	اجرای سوم	اجرای چهارم	اجرای پنجم	اجرای ششم	میانگین
0m0.528s	0m0.528s	0m0.618s	0m0.676s	0m0.641s	0m0.528s	0m0.586

میزان تسریع ($\frac{Serial\ Time}{Parallel\ Time}$) برنامه نسبت به حالت سری را در زیر بیاورید.

میانگین زمان اجرای سری	میانگین زمان اجرای موازی	میزان تسریع
0m1.912s	0m0.586	3.26

همان‌طور که مشاهده می‌شود ۳.۲۶ عددی نزدیک به ۴ می‌باشد که همان تعداد ریسه‌هاست.

[3] Lock