به نام خدا

پروژه نهایی

برنامه‌نویسی چندهسته‌ای

هدف:

هدف از این پروژه، افزایش مهارت در برنامه‌نویسی چندنخی و یادگیری تکنیکهای افزایش کارایی برنامه‌های موازی است. به عنوان یک مثال قابل‌موازی‌سازی پرکاربرد در محاسبات علمی، برنامه محاسبه دترمینان ماتریس در نظر گرفته شده است. در این پروژه، شما ابتدا برنامه سریال این کاربرد را تحلیل کرده و به طراحی و پیاده‌سازی روشی برای موازی‌سازی آن بر روی پردازنده چندهسته‌ای، پردازنده گرافیکی یا هر دو می‌پردازید. پس از آن، برنامه موازی‌شده به دقت صحت سنجی، تحلیل و ارزیابی شده و مشکلاتی که کارایی را محدود می‌کند شناسایی و رفع می‌شود. در نهایت کد موازی نوشته شده به طور کامل به ازای اندازه‌های ورودی مختلف ارزیابی و نتیجه گزارش می‌شود.

یکی از چالش‌های برنامه‌نویسی موازی، کاهش کارایی (یا محدود بودن کارایی) پس از انجام موازی‌سازی به دلیل گلوگاه[1] شدن برخی از منابع سیستم است. این اتفاق می‌تواند دلایل مختلفی از قبیل تخصیص بار نامتوازن، عدم سازگاری با معماری بستر مورداستفاده، عدم به‌کارگیری کامل منابع موجود و ... داشته باشد. یکی از هدف اصلی این پروژه، شناسایی گلوگاههای کارایی و رفع آنها برای دستیابی به تسریع حداکثری است. به‌طورکلی برای بهبود کد باید به نکات زیر توجه کنید:

- شناخت کامل بستر سخت‌افزاری موردنظر و محدودیت‌های آن
- شناسایی گلوگاه‌های برنامه و دلیل ایجاد گلوگاه
- شناسایی امکانات موجود در بستر مورداستفاده
- طراحی الگوریتم موازی مقیاس پذیر که بتواند در تقابل با ویژگی‌های متنوع بسترهای مختلف سخت‌افزاری و گلوگاههای آنها عملکرد خوبی داشته باشد
- آشنایی با تنظیمات مختلف کامپایلر اعم از تنظیمات بهینه‌سازی و اثر آن بر اجرای کد
- انجام بهبودهای جزئی (Fine Tuning)

۱. کاربرد: دترمینان و نحوه محاسبه آن

دترمینان یکی از مشخصه‌های مهم ماتریس‌های مربعی است. این مشخصه دارای کاربردهای زیر است:

- بررسی وجود یا عدم وجود ماتریس معکوس و محاسبه آن در صورت وجود
- محاسبه مقادیر ویژه ماتریس
- دترمینان ژاکوبی برای تغییر متغیر انتگرال‌های چندبعدی
- ...

روش‌های متعددی برای محاسبه دترمینان یک ماتریس وجود دارد. برای مثال برای ماتریس‌های ۲ در ۲ به‌صورت زیر داریم:

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad Det(A) = ad - bc$$

برای ماتریس‌های بزرگ‌تر می‌توان به روش‌های زیر اشاره کرد:

- فرمول لاپلاس
- فرمول لایب‌نیتز
- طرح ساروس (فقط برای ماتریس‌های ۳ در ۳)

[1] Bottleneck

برای مثال، محاسبه دترمینان یک ماتریس ۳ در ۳ با استفاده از روش لاپلاس به‌صورت زیر است:

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a \begin{vmatrix} e & f \\ h & i \end{vmatrix} - b \begin{vmatrix} d & f \\ g & i \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

همان‌طور که مشاهده می‌شود، محاسبه ماتریس‌های مرتبه بالاتر ساختاری بازگشتی دارد. برای درک بهتر این ساختار محاسبه دترمینان یک ماتریس ۴ در ۴ را در ادامه می‌بینیم:

$$\begin{vmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{vmatrix} = a \begin{vmatrix} f & g & h \\ j & k & l \\ n & o & p \end{vmatrix} - b \begin{vmatrix} e & g & h \\ i & k & l \\ m & o & p \end{vmatrix} + c \begin{vmatrix} e & f & h \\ i & j & l \\ m & n & p \end{vmatrix} - d \begin{vmatrix} e & f & g \\ i & j & k \\ m & n & o \end{vmatrix}$$

برای محاسبه دترمینان یک ماتریس ۳×۳ به محاسبه دترمینان ۳ ماتریس ۲×۲ و برای محاسبه دترمینان یک ماتریس ۴×۴ به محاسبه دترمینان ۱۲ ماتریس ۲×۲ نیاز است. به همین صورت برای محاسبه دترمینان یک ماتریس $n \times n$ به محاسبه دترمینان $\frac{N!}{2!}$ ماتریس ۲×۲ نیاز است. این عبارت نشان‌دهنده رشد بسیار زیاد تعداد دترمینانهای ۲×۲ موردنیاز برای محاسبه‌ی دترمینان ماتریس‌های بزرگ است. برای مثال برای محاسبه‌ی دترمینان یک ماتریس ۱۴×۱۴ به دترمینان ۴۳۵۸۹۱۴۵۶۰۰ ماتریس ۲×۲ نیاز است. برای ذخیره‌سازی اطلاعات موردنیاز این ماتریس‌ها با فرض ۴ بایتی بودن درایه‌ها، حدوداً به ۶۹۷۴۲۶ مگابایت حافظه نیاز دارد که خود این نیز چالشهایی ایجاد می‌کند. روشهای دیگری نیز برای محاسبه دترمینان وجود دارد. برای مثال، می‌توان یک ماتریس مربعی را با استفاده از روش تجزیه LU Decomposition به صورت حاصلضرب یک ماتریس بالا مثلثی و یک ماتریس پایین مثلثی نوشت. طبق قوانین محاسبه دترمینان، دترمینان حاصل‌ضرب دو ماتریس مربعی با اندازه یکسان برابر است با ضرب دترمینان آن دو ماتریس. همچنین در ماتریس‌های بالا و پایین مثلثی، دترمینان برابر با حاصل‌ضرب درایه‌های قطر اصلی است.

۲. شرح پروژه

با توجه به نکات بالا روشهای متنوعی برای محاسبه دترمینان وجود دارد که هر کدام به نوعی مصالحه‌ای بین سرعت محاسبات، میزان موازی سازی و حجم حافظه مورد نیاز برای ذخیره نتایج میانی ارائه می‌دهند. در این پروژه، شما باید با استفاده از موازی‌سازی روی پردازنده چندهسته‌ای، پردازنده‌ی گرافیکی و یا هر دو، محاسبه دترمینان یک ماتریس مربعی را تسریع دهید. در روش مورداستفاده‌ی شما به‌کارگیری تمامی روش‌هایی که در درس آشنا شده‌اید و تمامی کتابخانه‌های موجود در CUDA Toolkit مجاز است.

۱. برنامه شما به‌عنوان ورودی فایلهایی حاوی ماتریس‌های مربعی دریافت می‌کند. در هر فایل تعدادی ماتریس با اندازه‌های مختلف وجود دارد. اندازه ماتریس‌ها می‌تواند یکی از مقادیر ۴، ۸، ۱۱، ۱۳، ۳۲ و ۶۴ باشد.
۲. اعداد درایه‌های ماتریس اعدادی بین ۰ تا ۹ است و درایه‌ها با فاصله (space) از هم جداشده‌اند (ماتریس به صورت ردیفی ذخیره شده است).
۳. ماتریس‌ها با یک خط خالی (\n) از یکدیگر جدا شده‌اند.
۴. فایلهای ورودی در کنار برنامه اجرایی شما در پوشه‌ای به نام data_in قرار می‌گیرند و ساختار txt دارند.
۵. برنامه شما باید قابلیت اجرا در کنسول یا ترمینال را داشته باشد و توانایی خواندن خودکار همه فایلهای موجود در پوشه data_in را داشته باشد.
۶. خروجی برنامه‌ی شما به ازای هر فایل ورودی یک فایل خروجی است که در هر خط آن، دترمینان ماتریس موردنظر قرار داده شده است. همه فایلهای خروجی باید در پوشه‌ای به نام data_out ذخیره شوند.
۷. استفاده از روش‌های تقریبی با حفظ دقت به‌اندازه مناسب (خطا کمتر از ۱ درصد) بلامانع است.
۸. برای موازی‌سازی یکی از روش‌های زیر را انتخاب کنید:
   i. استفاده از پردازنده چندهسته‌ای (حداکثر ۷۰ درصد نمره پروژه)
   ii. استفاده از پردازنده گرافیکی (حداکثر ۷۰ درصد نمره پروژه)
   iii. استفاده همزمان از هر دو بستر (۱۰۰ درصد نمره پروژه)

۳. موارد قابل تحویل

موارد زیر را در پوشه تحویلی خود قرار دهید:

۹. کد برنامه در قالب پروژه‌ی Visual Studio 2017 و قابلیت کامپایل مجدد.
۱۰. فایل اجرایی با فرمت exe.
۱۱. گزارشی کامل شامل موارد زیر:
   i. مدل پردازنده مورداستفاده.
   ii. میزان حافظه موجود.
   iii. نتایج پروفایلینگ کد سریال و شناسایی و تحلیل گلوگاه‌ها و توجیه الگوریتم موازی استفاده شده.
   iv. زمان اجرای برنامه (میانگین حداقل ۳ بار اجرا) برای فایلهای داده شده.
   v. تحلیل زمان اجرا، میزان بهره‌وری هسته‌ها، و به طور کلی نتایج ارزیابی کیفیت موازی سازی توسط پروفایلر
   vi. توضیح کامل تکنیکهایی که برای بهبود کد و Fine Tuning استفاده شده است.
۱۲. ارائه روشی برای حل مسئله در اندازه‌های بسیار بزرگ (این بخش امتیازی است).
۱۳. نام پوشه‌ی تحویلی را به‌صورت زیر تنظیم کرده و آن را آپلود کنید.
   i. FirstName_LastName_StudentID.zip

۴. ارزیابی

- جهت سنجش عادلانه زمان‌های اجرای برنامه‌ی شما و صحت محاسبات، برنامه‌های شما روی بستر زیر اجرا خواهد شد:
  - Intel Haswell-E 5820k
  - 64GB DDR4-3000 RAM
  - NVIDIA RTX2080TI
  - CUDA 10.1
  - Windows 10 with May 2020 update
  - Visual Studio 2017
- برنامه‌های اجرایی خود را در حالت release و با توجه به نسخه‌ی CUDA گفته‌شده کامپایل کنید تا از مشکلات احتمالی جلوگیری شود.
- جهت بررسی همخوانی کد ارائه‌شده و برنامه اجرایی، پروژه‌های شما مجدداً کامپایل خواهند شد.
- موارد زیر در ارزیابی کیفیت پروژه نقش دارند:
  - استفاده از اسم‌های مناسب و استاندارد برای متغیرها.
  - کامنت گذاری کد.
  - ماجولار بودن کد.
  - استفاده از امکانات پردازنده گرافیکی و کتابخانه‌های موجود در CUDA.
  - کیفیت گزارش همراه کد.
  - انجام پروفایلینگ قبل و بعد از موازی‌سازی و Fine Tune کردن کد
  - میزان تسریع حاصل شده
  - قابلیت مقیاس پذیری (عملکرد خوب در مواجهه با اندازه های بزرگتر ماتریس و فایلهای بیشتر و بزرگتر، قابلیت استفاده از همه منابع سیستم در صورت وجود: مثلا شناسایی همه کارتهای گرافیکی متصل به سیستم، شناسایی هسته‌های موجود در سیستم، شناسایی حجم حافظه سیستم و انطباق با شرایط)
  - تحویل تلفنی (یا اینترنتی) پروژه برای توضیح در مورد کد و پاسخ به سوالات

برخی از امکانات قابل‌استفاده در این پروژه عبارت‌اند از:

۱. Shared memory
۲. Streams
۳. همپوشان کردن محاسبات CPU و GPU و انتقال داده بر روی PCIe
۴. استفاده از تکنیک خط لوله

۵. نکات آخر

۱. انجام پروژه به صورت انفرادی است. با توجه به حساسیتهای بوجود آمده در ترم جاری به دلیل ارزیابی غیرحضوری، کدها و گزارشات تحویلی توسط یک ابزار خودکار و همچنین به صورت تصادفی توسط استاد درس مشابهت سنجی خواهد شد. وجود هر گونه تشابه بین کدها یا گزارشات به منزله نمره صفر برای هر دو طرف درگیر است. همچنین پروژه پس از تحویل (آپلود در سایت درس)، به صورت تلفنی/اینترنتی نیز برای پرسش و پاسخ تحویل گرفته خواهد شد. خواهشمندم به صورت جدی این موضوع را مورد توجه قرار دهید.

۲. برای آشنایی با نحوه بهینه کردن کد OpenMP یا CUDA لازم است حتما به لینکهای قسمت مراجع مراجعه کنید. مواردی که در این لینکها توصیه شده است مورد سوال خواهد بود.

۳. مهلت تحویل و بارگذاری پروژه یک هفته پس از آخرین امتحان است. با توجه به اینکه نمره پروژه بخش قابل توجهی از نمره پایانی خواهد بود، انجام آن را به روزهای پایانی موکول نکنید. همچنین با توجه به آخرین اطلاعیه معاونت آموزشی مبنی بر شروع امتحانات از ۱۴ تیرماه، توصیه اکید می‌شود پروژه را قبل از شروع امتحانات تحویل دهید.

موفق باشید.

۶. مراجع

- نحوه استفاده از Intel VTune برای بهینه کردن کد OpenMP: شامل روشهای پروفایل کردن کد و تکنیکهای Tune کردن

https://software.intel.com/content/www/us/en/develop/documentation/vtune-cookbook/top.html
https://www.dideo.ir/search?q=vtune+openmp

- نحوه استفاده از Nvidia Nsight برای بهینه کردن کد CUDA

http://web.stanford.edu/class/cme213/files/lectures/Lecture_15_nvvp.pdf
http://on-demand.gputechconf.com/gtc/2015/presentation/S5174-Christoph-Angerer.pdf
https://developer.nvidia.com/nsight-compute-videos
https://developer.nvidia.com/gtc/2020/video/s21771
https://www.dideo.ir/search?q=nsight+cuda

- اسلاید OpenMP Performance Tips موجود در پوشه **اسناد دیگر** در سایت درس
- اسلاید CUDA C Best Practices Guide موجود در پوشه **اسناد دیگر** در سایت درس