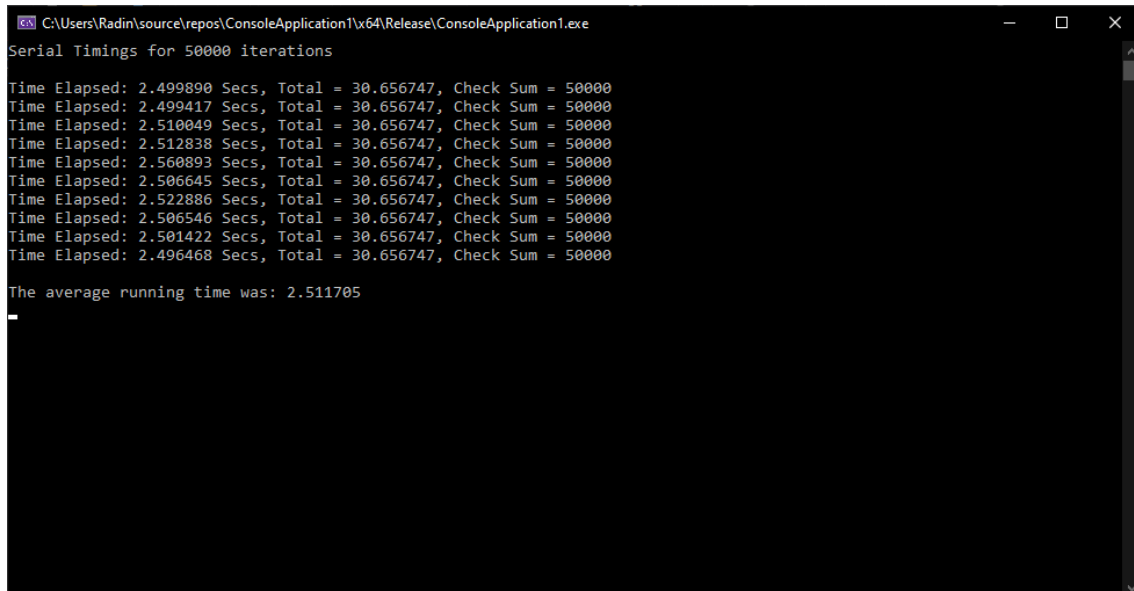


۱ مرحله اول

ابتدا برنامه سریال داده شده را روی حالت Release اجرا می‌کنیم. نتیجه را در شکل ۱ می‌بینیم. زمان اجرای برنامه در حالت سریال به طور میانگین تقریباً ۲/۵۱ ثانیه است.



```
C:\Users\Radin\source\repos\ConsoleApplication1\Release\ConsoleApplication1.exe
Serial Timings for 50000 iterations
Time Elapsed: 2.499890 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.499417 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.510049 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.512838 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.560893 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.506645 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.522886 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.506546 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.501422 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 2.496468 Secs, Total = 30.656747, Check Sum = 50000
The average running time was: 2.511705
```

شکل ۱: زمان اجرای برنامه در حالت سریال

سوال ۱-

دو حلقه با شمارنده k که به انجام محاسبات ریاضی می‌پردازند، بیشترین زمان اجرا را در این کد دارند. برای تسریع زمان اجرا، می‌توان اجراهای مختلف آن‌ها (حلقه j) را بر روی هسته‌های مختلف موازی‌سازی کرد.

سوال ۲-

به دلیل اینکه عوامل زیادی (مانند وضعیت و بار سیستم در هنگام اجرای برنامه) روی زمان اجرا تاثیر می‌گذارند، بهتر است برای اندازه‌گیری زمان اجرا به جای یک بار، چند بار برنامه را اجرا کنیم و در نهایت زمان‌های صرف شده را میانگین بگیریم.

در اینجا (در حالت سریال)، تنها زمان اجرا از اجرایی به اجرای دیگر (به دلیل گفته شده) متفاوت است. همچنین باید توجه داشت در بسیاری از موارد، ممکن است برنامه به علت داشتن ایراداتی (مانند شرایط

مسابقه در برنامه‌های موازی) نتایج آن به غلط از اجرایی به اجرای دیگر تغییر کند. چنین اتفاقی نیز معمولاً با چندین بار اجرا قابل کشف است.

سوال ۳-

دو حالت Debug و Release تنها دو پیکربندی (Configuration) مختلف برای کامپایل و اجرای برنامه‌هاست و به خودی خود مفهوم متفاوتی از هم ندارند. اما به صورت پیش‌فرض در ویژوال استودیو تفاوت‌هایی بین ویژگی‌های (Properties) این دو پیکربندی وجود دارد. اصلی‌ترین عاملی که موجب سریع‌تر بودن حالت Release می‌شود، فعال بودن بهینه‌سازی‌های (Optimizations) کامپایلر، برخلاف حالت Debug، است.

به صورت پیش‌فرض در حالت Release، تنظیمات بهینه‌سازی کامپایلر روی حالت O2 (بیشترین بهینه‌سازی برای سرعت اجرا) است. در حالی که در حالت Debug تنظیمات آن روی Od (غیرفعال بودن بهینه‌سازی‌ها) است.

سوال ۴-

از آنجا که اجراهای حلقه مستقل از هم هستند، می‌توان آن را موازی کرد. برای این کار از روش تجزیه Geometric Decomposition و الگوی Loop Parallelism استفاده می‌کنیم.

۲ مرحله دوم

با استفاده از خط زیر، اجراهای حلقه را روی هسته‌های مختلف موازی‌سازی می‌کنیم.

```
#pragma omp parallel for
```

مطابق شکل ۲، زمان اجرای برنامه در این حالت برخلاف انتظار بسیار بیشتر شده و به طور میانگین ۴۴/۷۲ ثانیه است. علت این امر آن است که در این کد متغیرهای اشتراکی به شدت استفاده می‌شوند و این باعث کندی اجرا می‌شود. همچنین به علت وجود شرایط مسابقه روی این متغیرهای اشتراکی (از جمله sum و total) وجود دارد که باعث متفاوت بودن خروجی چاپ شده در اجراهای مختلف است.

```
C:\Users\Radin\source\repos\ConsoleApplication1\x64\Release\ConsoleApplication1.exe
Parallel Timings for 50000 iterations
Time Elapsed: 44.024956 Secs, Total = 841.260033, Check Sum = 44023
Time Elapsed: 46.480888 Secs, Total = 834.574349, Check Sum = 43499
Time Elapsed: 45.036823 Secs, Total = 834.661668, Check Sum = 44026
Time Elapsed: 43.885063 Secs, Total = 822.980088, Check Sum = 44431
Time Elapsed: 43.809934 Secs, Total = 793.651043, Check Sum = 44577
Time Elapsed: 44.416701 Secs, Total = 795.495718, Check Sum = 44312
Time Elapsed: 42.950877 Secs, Total = 823.061937, Check Sum = 44417
Time Elapsed: 47.773969 Secs, Total = 855.398802, Check Sum = 43753
Time Elapsed: 45.732635 Secs, Total = 828.678362, Check Sum = 43965
Time Elapsed: 43.183733 Secs, Total = 824.458403, Check Sum = 44573
The average running time was: 44.729558
```

شکل ۲: طولانی‌تر شدن زمان اجرا و تفاوت خروجی برنامه در اجراهای مختلف به علت استفاده مکرر از متغیرهای اشتراکی و وجود شرایط مسابقه

با محلی کردن برخی متغیرها به کمک عبارت زیر، مطابق شکل ۳ می‌بینیم که سرعت اجرای برنامه به علت موازی‌سازی روی هسته‌های مختلف بیشتر شده است و به مقدار میانگین 0.64 ثانیه رسیده است. اما همچنان خروجی برنامه به علت وجود شرایط مسابقه در اجراهای مختلف متفاوت است.

```
private(k, sumx, sumy)
```

مطابق شکل ۴، با اضافه کردن کد زیر در بخش‌هایی که روی متغیرهای sum و total می‌نویسند، مشکل شرایط مسابقه‌ای و متفاوت بودن نتایج اجراهای مختلف برطرف شده است. زمان اجرای برنامه در این حالت نیز به طور میانگین 0.64 ثانیه است.

```
#pragma omp critical
```

با استفاده از reduction به جای critical، برنامه باز هم به طور میانگین در زمان 0.64 ثانیه (شکل ۵) اجرا می‌شود.

راه‌حل دیگر برای جلوگیری از مشکلات ناحیه بحرانی، استفاده از قفل است. با استفاده از خطوط زیر، دو قفل برای دو متغیر اشتراکی می‌سازیم و ناحیه‌های بحرانی آن‌ها را با این قفل‌ها محافظت می‌کنیم.

```

C:\Users\Radin\source\repos\ConsoleApplication1\Release\ConsoleApplication1.exe
Parallel Timings for 50000 iterations
Time Elapsed: 0.690189 Secs, Total = 30.608400, Check Sum = 49902
Time Elapsed: 0.629373 Secs, Total = 30.577273, Check Sum = 49880
Time Elapsed: 0.627444 Secs, Total = 30.599808, Check Sum = 49891
Time Elapsed: 0.617354 Secs, Total = 30.593818, Check Sum = 49889
Time Elapsed: 0.654336 Secs, Total = 30.603646, Check Sum = 49894
Time Elapsed: 0.631191 Secs, Total = 30.593225, Check Sum = 49879
Time Elapsed: 0.675470 Secs, Total = 30.580456, Check Sum = 49893
Time Elapsed: 0.676901 Secs, Total = 30.606869, Check Sum = 49874
Time Elapsed: 0.625583 Secs, Total = 30.577165, Check Sum = 49880
Time Elapsed: 0.612118 Secs, Total = 30.617280, Check Sum = 49898
The average running time was: 0.643996

```

شکل ۳: رفع مشکل زمان اجرای طولانی برنامه با محلی کردن برخی متغیرها

```

C:\Users\Radin\source\repos\ConsoleApplication1\Release\ConsoleApplication1.exe
Parallel Timings for 50000 iterations
Time Elapsed: 0.673734 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.668638 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.649709 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.617736 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.618044 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.622349 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.639804 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.655701 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.653540 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.625738 Secs, Total = 30.656747, Check Sum = 50000
The average running time was: 0.642499

```

شکل ۴: رفع مشکل شرایط مسابقه با استفاده از راهنمای critical

```

omp_lock_t sum_lock, total_lock;
omp_init_lock(&sum_lock);
omp_init_lock(&total_lock);

```

مطابق شکل ۶، مشکل ناحیه بحرانی با استفاده از همگام‌سازی سطح پایین نیز قابل حل است.

```
C:\Users\Radin\source\repos\ConsoleApplication1\Release\ConsoleApplication1.exe
Parallel Timings for 50000 iterations
Time Elapsed: 0.653058 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.669665 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.646334 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.621171 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.651226 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.627746 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.637080 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.652863 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.626897 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.639784 Secs, Total = 30.656747, Check Sum = 50000
The average running time was: 0.642582
```

شکل ۵: رفع مشکل شرایط مسابقه با استفاده از راهنمای reduction

```
C:\Users\Radin\source\repos\ConsoleApplication1\Release\ConsoleApplication1.exe
Parallel Timings for 50000 iterations
Time Elapsed: 0.638082 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.662985 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.637586 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.648050 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.631927 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.639385 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.646443 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.637495 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.640827 Secs, Total = 30.656747, Check Sum = 50000
Time Elapsed: 0.659383 Secs, Total = 30.656747, Check Sum = 50000
The average running time was: 0.644216
```

شکل ۶: استفاده از همگام‌سازی سطح پایین (Lock) برای محافظت از ناحیه بحرانی

سوال ۱-

تعداد نخ‌های پیش‌فرض OpenMP معمولاً برابر تعداد هسته‌های موجود است. در این گزارش از یک پردازنده ۸ هسته‌ای استفاده شده است و در نتیجه از ۸ نخ برای موازی‌سازی استفاده شده است.

سوال ۲-

بله؛ به جای محافظت از ناحیه بحرانی با critical، به علت اینکه تنها اعمال ناحیه بحرانی اعمال حسابی (arithmetic) هستند، می‌توان با استفاده از atomic این اعمال را انجام داد. باید توجه داشت که برای استفاده از atomic، لازم است عبارت جمع را به جای

```
total = total + 1.0 / sqrt(sumx);
```

به شکل

```
total += 1.0 / sqrt(sumx);
```

نوشت.

سوال ۳-

با چند بار اجرا به ازای مقدار $VERYBIG = 100000$ و ۵۱۲ نخ، می‌بینیم که زمان اجرا با استفاده از reduction حدود ۱/۲۹ ثانیه و با استفاده از critical حدود ۱/۳۱ ثانیه است. در نتیجه می‌بینیم که با زیاد شدن تعداد نخ‌ها استفاده از reduction، در کاربردهایی که استفاده از آن ممکن باشد، بهتر است. افزایش تعداد iterationهای برنامه نیز به دیده شدن کندی استفاده از ناحیه بحرانی کمک می‌کند.