Mnożenie macierzy przez wektor

Dane typu całkowitego (int)

Tomasz Rybiński - dokument Nikita Lysiuk – kod sekwencyjny i równoległy Kacper Górak - kod sekwencyjny i równoległy

Informacje na temat architektury testowej

Model procesora: Intel Xeon CPU E5-2670 v3

Ilość rdzeni na socket: 12 Ilość watków na rdzeń: 2

Ilość socketów: 2

sum += int_val;

return 0;

double end_time = omp_get_wtime(); __

Maksymalne taktowanie pojedyńczego rdzenia: 3.100MHz Taktowanie podstawowe pozostałych rdzeni: 2.300Mhz

Pamięć podkręczna: 30720K System operacyny: AlmaLinux 8.7

Werska kernela: 4.18.0-425.13.1.el8_7.x86_64

Wersja kompilatora: g++ (GCC) 8.5.0 20210514 (Red Hat 8.5.0-20)

PROGRAM SEKWENCYJNY

Kod programu sekwencyjnego:

Sposób kompilacji: **g++ -fopenmp ./matrix_2D_vector_single.cpp**

```
Doświadczenie zostało przeprowadzone na dwóch macierzach testowych
include <random>
  constexpr int N = 100'000;
constexpr int M = 145'000;
  std::vector<std::vector<long long>> matrix;
       matrix.reserve(N);
for (int i = 0; i < N; ++i) {</pre>
                                                                                             Ten blok kodu tworzy dwuwymiarową macierz o
                                                                                             wymiarach N × M, wektor o rozmiarze M oraz wektor
                                                                                             wynikowy o rozmiarze N, wykorzystując std::vector.
  } catch (const std::bad_alloc& e) {
   std::cerr << "Manual allocation failed: " << e.what() << std::endl;</pre>
                                                                                             Użyto bloku try-catch, aby przechwycić ewentualny
                                                                                             wyjątek std::bad_alloc, który może wystąpić przy alokacji
       return 1:
                                                                                             dużej ilości pamięci.
  std::vector<lo
  std::random_device rd;
                                                                                             Ten fragment kodu odpowiada za wypełnienie macierzy
  std::mt19937 gen(rd());
                                                                                             oraz wektora losowymi liczbami całkowitymi z zakresu od
  std::uniform_int_distribution<long long> dis(-1'000'000'000, 1'000'000'000)
//std::uniform_int_distribution<long long> dis(-10, 10);
                                                                                             -1000 000 000 do 1000 000 000. Użyto
                                                                                            std::random_device jako źródła entropii do inicjalizacji
  for (int i = 0; i < N; ++i) {
   for (int j = 0; j < M; ++j) {
     matrix[i][j] = dis(gen);
}</pre>
                                                                                             generatora liczb pseudolosowych std::mt19937.
                                                                                             std::uniform_int_distribution zapewnia równomierny
                                                                                             rozkład wartości w zadanym przedziale.
                                                                                             Każdy element macierzy i wektora jest przypisywany w
                                                                                             pętli, przy czym każda wartość jest generowana osobno.
  for (int i = 0; i < M; ++i) {
    vector[i] = dis(gen);</pre>
                                                                                            Zapisuje czas rozpoczęcia obliczeń przy użyciu funkcji
      ble start_time = omp_get_wtime(); -
                                                                                            -omp_get_wtime(), która zwraca aktualny czas ścienny
  for (int i = 0; i < N; i++) {
    long long sum = 0;
    for (int j = 0; j < M; j++) {
        long long val = matrix[i][j] * vector[j];
        double tmp = std::sin(static_cast<double>(val));
        int int_val = static_cast<int>(tmp * 1000);
                                                                                            (wall time) w sekundach.
                                                                                            Ta pętla wykonuje mnożenie macierzy przez wektor: dla
                                                                                            każdego wiersza macierzy obliczana jest suma iloczynów
                                                                                            odpowiadających elementów wiersza i wektora. Każdy
                                                                                             iloczyn jest następnie przekształcany na wartość
```

std::cout « "\nMatrix " « N « "x" « M « "\n"; Zapisuje czas zakończenia obliczeń, umożliwiając std::cout « "\nExecution Time (OpenMP): " « (end_time - start_time) « " sepóźniejsze obliczenie całkowitego czasu wykonania

zmiennoprzecinkową, przetwarzany funkcją sin() w celu dodania kosztu obliczeniowego, a następnie skalowany (× 1000) i rzutowany na typ całkowity. Wynikowa suma jest

Zapisuje czas zakończenia obliczeń, umożliwiając

zapisywana w wektorze.

fragmentu kodu.

Test nr 1. 2'000 x 4'000 / Test nr 2. 100'000 x150'000

Wyniki dla programu sekwencyjnego

Czas dla 2000 x 4000	Czas dla 100000 x150000
0.839403 (s)	451.756 (s) ~ 7.5 minuty

Wyniki te posłużą nam zaraz jako punkt odniesienia do programu równoległego

PROGRAM RÓWNOLEGŁY

Sposób kompilacji: g++ -fopenmp ./matrix_2D_vector.cpp
Program równoległy był uruchamiany za pomocą skryptu batch
za pomocą którego w pętli był uruchamiany program równoległy z odpowiednią ilością wątków,
która była ustalana za pomocą zmiennej środowiskowej OMP_NUM_THREADS

```
#include "omp.h" dodaje do programu możliwość
                                                                                                                            korzystania z funkcji i dyrektyw biblioteki OpenMP.
      tmatrix.reserve(N);
for (int i = 0; i < N; ++i) {
    matrix.emplace_back(std::vector<long long>(M));
 catch (const std::bad_alloc6 e) {
   std::cerr < "Manual allocation failed: " << e.what() << std::endl;</pre>
std::random_device rd;
#pragma omp paralle
                                                                                                                             #pragma omp parallel tworzy równoległą sekcję kodu, która będzie wykonywana
      std::mt19937 gen(rd());
std::uniform_int_distribution<long long> dis(-1'000'000'000, 1'000'000'000);
//std::uniform_int_distribution<long long> dis(-10, 10);
                                                                                                                              jednocześnie przez wiele wątków. Każdy wątek tworzy własną instancję
                                                                                                                             generatora liczb losowych (std::mt19937), aby uniknąć konfliktów
                                                                                                                              współbieżności.
           (int i = 0; i < N; ++i) {
  for (int j = 0; j < M; ++j) {
    matrix[i][j] = dis(gen);
}</pre>

    #pragma omp for collapse(2) rozdziela dwie zagnieżdżone petle (for (i)... for

                                                                                                                                 (j)...) pomiędzy dostępne wątki, co przyspiesza inicjalizację całej macierzy
                                                                                                                                  matrix[i][j] losowymi wartościami.
                                                                                                                               • #pragma omp for rozdziela wypełnianie wektora vector[i] między wątki.
                                                                                                                             Dzięki zastosowaniu #pragma omp parallel i odpowiednich dyrektyw for, oba
      #pragma omp for
for (int i = 0; i < M; ++i) {
    vector[i] = dis(gen);</pre>
                                                                                                                             procesy – wypełnianie macierzy i wektora – są wykonywane współbieżnie, co
                                                                                                                             znacznie skraca czas inicjalizacji przy dużych rozmiarach danych (N × M, M).
         start_time = omp_get_wtime();
                                                                                                                             #pragma omp parallel for powoduje równoległe wykonanie pętli for (int i = 0; i < N; i++), w której każdy wątek oblicza osobny wiersz wyników. Dla każdego wiersza macierzy (matrix[i]) obliczana jest suma iloczynów elementów wiersza i odpowiadających im wartości wektora (vector[j]). Każdy iloczyn jest dodatkowo przekształcany:

• rzautowany na double,
     agma omp parallel for
(int i = 0; i < N; i++) {
  long long sum = 0;
  for (int j = 0; j < M; j++) {
    long long val = matrix[i][j] * vector[j];
    double tmp = std::sin(static_castdouble>(val));
    int int_val = static_cast<int>(tmp * 1000);
                                                                                                                                 przepuszczany przez funkcję sin(),
mnożony przez 1000,

    konwertowany na int.
    Te operacje matematyczne celowo zwiększają złożoność obliczeniową, aby lepiej

      result[i] = sum:
                                                                                                                             pokazać wpływ równoległości (OpenMP) na czas wykonania.
Wynik (sum) zapisywany jest w wektorze result[i].
     ble end_time = omp_get_wtime();
std::cout < "\nMatrix " < N < "x" < M < "\n"; std::cout < "\nExecution Time (OpenMP): " < (end_time - start_time) < " seconds\n";
                                                                                                                             Po zakończeniu obliczeń, program wyświetla rozmiar macierzy oraz czas
                                                                                                                             wykonania operacji (różnica między end_time a start_time).
Następnie:

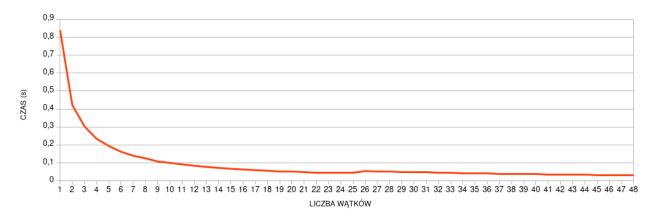
    #pragma omp parallel uruchamia blok równoległy,

    #pragma omp single zapewnia, że tylko jeden wątek (spośród wszystkich uruchomionych) wykona zawartą w nim instrukcję,
    omp_get_num_threads() zwraca liczbę wątków użytych przez OpenMP, co

            // Funkcja omp_get_num_threads() zwraca liczbę uruchomionych wątków std::cout « "Threads used: " « omp_get_num_threads() « std::endl;
                                                                                                                                  pozwala sprawdzić stopień równoległości w praktyce.
```

1 0,839403 1,98875318367589 3 0,0422075 1,98875318367589 3 0,032588 2,77407894562904 4 0,193572 4,33638646085178 5 0,161534 5,19644780655371 6 0,139382 6,02231995523095 7 0,125406 6,8934355637965 8 0,107521 7,80687493605900 9 0,099239 8,40042272169121 0,091446 9,1792205235876 11 0,0338943 10,0054839903110 12 0,0717771 11,6945794689393 11 0,054839903110 12 0,0773314 10,85462050344360 13 0,0717771 11,69457946893930 14 0,0054839103110 15 0,0631954 13,28265981884216 13,28265981884216 13,28265981884216 13,28265981884216 14,16102072353550 17 0,056112 14,95542044482460 19 0,05321684 15,787623942417320 11 0,0459162 18,28119487239800 11 0,0458391 17,3577051682103 0,0441495 19,01274080114160 22 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 16,6637577051682103 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 15,66605274358390 16,66347527272730 15,66605274358390 16,66347527272730 16,0459162 19,0	Liczba wątków	Czas (s)	Przyśpieszenie
2 0.422075 1.98875318367589 3 0.302588 2,77407894562904 3 0.233407 3,59630602338405 4 0.193572 4,33638646085178 5 0.161534 5,1964780665371 6 0.7139382 6,02231995523095 7 0.125406 6,69348356537985 8 0.107521 7,80687439605900 9 0.0999239 8,40042272169121 10 0.091446 9,17922052358769 11 0.0838943 10,00548039003110 12 0.0773314 10,85462050344360 13 0.0777771 11,69457946893930 14 0.0670795 12,51355481182780 15 0.0631954 13,28265981384720 16 0.0592756 14,1610207233555769 17 0,056112 14,95942044482460 18 0.0531684 15,78762949421080 19 0.0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0549239 16,491477272739 26 0,0549239 17,49860849661350 27 0,058992 17,04894292432370 28 0,0441495 19,01274080114160 29 0,0492349 17,49860849661350 30 0,0479697 17,9597153489661350 31 0,044596 19,04274080114160 25 0,045936 19,04274080114160 25 0,0549239 17,49860849661350 31 0,0479697 17,9597159490866 32 0,044596 21,8835757314430 33 0,0444986 19,59887823 34 0,0428466 20,1920814890739 35 0,045739 17,49860849661350 36 0,0479697 17,95971594908650 37 0,045967 17,95971594908650 38 0,045739 17,49860849661350 39 0,045562 18,83355222505880 30 0,0479697 17,95971594908650 31 0,0459892 17,0489429243270 32 0,0444986 19,5998893878230 34 0,0428466 20,19208148007380 35 0,0447999 17,95971594908650 36 0,045739 17,49860849661350 37 0,045866 22,3775758449676 38 0,038566 22,3775758449676 39 0,038566 22,3775758449676 30 0,038566 24,6260314881740 40 0,038566 24,6260314881740 41 0,0356441 24,08160864800350 44 0,034566 24,6260314881740 44 0,034566 24,6260314881740 45 0,034566 24,6260314881740 46 0,034566 24,6260314881740 47 0,0345866 24,6260314881740 48 0,0345866 24,6260314881740 49 0,0345866 24,6260314881740 40 0,0345799 26,57444372143560 47 0,0311911 26,91161901952800 46 0,0311911 26,91161901952800	1	0.839403	1
3 0,302888 2,77407894562904 4 0,233407 3,59630602338405 4 0,193572 4,3363864055178 5 0,161534 5,19644780665371 6 0,139382 6,02231995523095 7 0,125406 6,6934835653796 8 0,107521 7,80687493605909 9 0,0999239 8,40042272169121 10 0,091446 9,17922052358769 11 0,0838943 10,00548309003110 12 0,07773114 10,85462053044360 13 0,0717771 11,69457946893930 14 0,0570795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487293800 21 0,0441495 19,01274080114160 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0563992 17,04894298263270 28 0,059992 16,49177272730 30 0,047697 17,95971594908650 31 0,0467381 18,43375160861030 20 0,047697 17,95971594908650 31 0,0467381 18,43375160861030 20 0,047697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222550580 33 0,044986 19,5908935878230 34 0,0428466 20,19208148007330 35 0,047697 17,95971594908650 36 0,045544 21,21306238801520 37 0,0385701 21,79927803459200 38 0,0375109 22,849805517740 40 0,036737 23,6158180963925 41 0,0345666 24,62603414891740 42 0,034666 20,192081488007330 44 0,0428466 20,19208148007330 44 0,045544 21,21306238801520 44 0,044596 19,5908935878230 45 0,0455362 18,86358222550580 46 0,0519308 16,16387577314430 47 0,0365441 24,08160864800350 48 0,0365544 21,21306238801520 49 0,0485666 24,62603414891740 40 0,0367579 20,5472138723796 44 0,0345666 24,62603414891740 45 0,0345666 24,62603414891740 46 0,0345245 26,91161901952800 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800 47 0,0311911 26,91161901952800	2	•	1.98875318367589
3 0,233407 3,59630602338405 4 0,193572 4,33638646085178 5 0,161534 5,19644780665371 6 0,139382 6,02231995523095 7 0,125406 6,69348356537965 8 0,107521 7,8068749360590 9 0,0999239 8,40042272169121 10 0,091446 9,17922052358769 11 0,0838943 10,00548399003110 12 0,0773314 10,85462050344360 13 0,0777371 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0554518 10 0,0554518 10 0,0570795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,161020723535676 17 0,0566112 14,95942044482460 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 22 0,0441495 19,01274080114160 22 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 16,6387577314430 1655 0,0519308 16,6387577314430 1655 0,0519308 16,6387577314430 1655 0,0519308 16,6387577314430 1655 0,0519308 16,6387577314430 1655 0,0519308 16,6387577314430 17,94894292463270 17,04	3	,	•
4 0,193572 4,3363864085178 5 0,161534 5,19644780665371 6 0,139382 6,02231995523095 7 0,125406 6,69348356537965 8 0,107521 7,80687493605900 9 0,0999239 8,40042272169121 10 0,091446 9,17922052358769 11 0,0838943 10,0054830903110 12 0,0773314 10,8546250344360 13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,8265981384720 16 0,0592756 14,1610207235358769 17 0,056112 14,959420444842460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,357705182103 21 0,0459162 18,28119487239800 21 0,0441495 19,01274080114160 22 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,058992 16,4914772727370 28 0,0569992 16,4914772727370 29 0,049349 17,498648661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0443866 29,1920814800730 33 0,0444986 19,59088938878230 34 0,0428466 20,1920814800730 35 0,0385701 21,799278034801740 36 0,0395701 21,799278034801740 37 0,0395701 21,799278034801740 38 0,034566 22,37757558469670 39 0,0375701 21,799278034801740 44 0,0332245 20,13161901952800 45 0,0325799 26,37243628412050 46 0,0319111 26,91161901952800 47 0,0319111 26,91161901952800		•	•
5 0,161534 5,19644780665371 6 0,139382 6,02231995523095 7 0,125406 6,69348956537965 8 0,107521 7,80687493605900 9 0,0999239 8,4004227216911 10 0,091446 9,1792052358769 11 0,0838943 10,00548309003110 12 0,0773314 10,85462050344360 13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384728 16 0,0592756 14,1610272353550 17 0,056112 14,95942044482480 18 0,0531684 15,78762949421080 19 0,0503708 16,6644764417320 20 0,0483591 17,35770516821030 21 0,0459162 18,2811948723980 22 0,041495 19,01274080114160 23 0,0441495 19,01274080114160 25 0,052739 15,4660527435839 26 0,051930		,	,
6 0,139382 6,02231995523095 7 0,125406 6,6934835637965 8 0,107521 7,80687493605900 9 0,0999239 8,40042272169121 10 0,091446 9,179220523587695 11 0,00838943 10,00548309003110 12 0,0717314 10,85462050344360 13 0,0717771 11,69457946893931 14 0,0670795 12,251355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353541 18 0,0531684 15,78762949421080 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,056892 17,04894292463270 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,044986 19,5080899 17,4986084966133 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,044586 20,19208148007380 33 0,0446986 19,50883878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,2130623801520 37 0,0395701 21,7992775083459200 40 0,036737 23,3615818096392 41 0,0355441 24,08160864800350 44 0,034866 24,262634148817740 45 0,0332245 26,91161901952800 46 0,0319218 25,7644437243560 47 0,0311911 26,91161901952800		•	·
7 0,125406 6,69348356537965 8 0,107521 7,80687493605909 9 0,0999239 8,40042272169121 10 0,091446 9,17922052358769 11 0,0838943 10,0054830900311 12 0,0773314 10,85462050344360 13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,1610272353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,053708 16,6644747320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,498608150850 30 0,0479697 17,9597164980850 31 0,0467381 18,43375160861330 32 0,0455362 18,86358222505880 33 0,0444986 19,95088935878230 34 0,044986 19,95088935878230 35 0,044596 22,3775755849670 36 0,043854 21,21306238801520 37 0,0395701 21,79927803459200 40 0,036737 23,615818096392 41 0,0355441 24,08160864800350 44 0,0335245 21,21306238801520 45 0,0335245 26,91161901952800 46 0,0319318 1 26,91161901952800 47 0,031911 26,91161901952800		•	,
8 0,107521 7,80687493605900 9 0,0999239 8,40042272169121 10 0,091446 9,17922052358769 11 0,0838943 10,00548309003110 12 0,07773314 10,85462050344360 13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274588390 26 0,0519308 16,16387577314430 27 0,0508992 17,04894292463270 29 0,0492349 17,9597759480849661350 30 0,0479697 17,95971594908633 31 0,0467381 18,43375160861030 32 0,045362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,044506 22,37757558469670 36 0,0385701 21,79927803459200 37 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 30 0,038506 22,37757558469670 30 0,038506	7	,	•
9 0,0999239 8,40042272169121 10 0,091446 9,1792205238769 11 0,0838943 10,00548309003110 12 0,0773314 10,85462050344360 13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 19 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0483591 17,35770516821030 21 0,0483591 17,35770516821030 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0445966 29,19208148007380 33 0,0444986 19,5908935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0308506 22,37757558406970 40 0,036737 23,61581809639290 41 0,036541 24,081806393990 41 0,036561 24,37757558406970 42 0,036561 24,6603414891740 40 0,036737 23,61581809639290 41 0,038566 22,37757558406970 42 0,0348566 24,62603414891740 43 0,0332245 26,91161901952800 46 0,031288 25,76444372143560 47 0,0311911 26,91161901952800	8	•	,
10 0,091446 9,17922052358769 11 0,0838943 10,00548309003110 12 0,0773314 10,85462050344360 13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,9594204482460 18 0,053768 16,66447622417820 19 0,0503708 16,66447624417322 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,4660527435839 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,0489429463270 30 0,0479697 17,95971594908650 31		•	·
11 0,0838943 10,00548309003110 12 0,0777314 10,85462050344360 14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,05503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,4914772727273 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0475067 17,95971594908650 31 0,0467381 18,43375160861030 32 </td <td></td> <td>•</td> <td>,</td>		•	,
12		•	•
13 0,0717771 11,69457946893930 14 0,0670795 12,51355481182780 16 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,1638757314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,047697 17,9597159490650 31 0,0467381 18,43375160861030 32 0,045362 18,86358222505880 33 0,0444986 19,5908893587230 34		•	,
14 0,0670795 12,51355481182780 15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,8635822505880 32 0,045362 18,86358222505880 33 0,0444986 19,59088935878230 34 <td></td> <td>•</td> <td>,</td>		•	,
15 0,0631954 13,28265981384720 16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,59971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,044986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,04		•	,
16 0,0592756 14,16102072353550 17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,491477272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0473697 17,95971594908650 31 0,0467381 18,86358222505880 32 0,045362 18,8635822505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36		,	•
17 0,056112 14,95942044482460 18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,4880489661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,045362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,04408524 21,21306238801520 37 <td></td> <td>•</td> <td></td>		•	
18 0,0531684 15,78762949421080 19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 17,04894292463270 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0453362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38<		,	,
19 0,0503708 16,66447624417320 20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 17,04894292463270 28 0,0508992 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0340866 24,62603414891740 42 0,0		•	,
20 0,0483591 17,35770516821030 21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,3757558469670 39 0,0375109 22,28498059177400 40 0,036737 23,61581809639290 41 0,034		•	•
21 0,0459162 18,28119487239800 22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306233801520 37 0,0395701 21,7992780345920 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0348566 24,62603414891740 43 0,034		•	·
22 0,0441495 19,01274080114160 23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,033		•	
23 0,0441495 19,01274080114160 24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,035441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332		,	,
24 0,0441495 19,01274080114160 25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,031		•	•
25 0,0542739 15,46605274358390 26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0445709 20,54721387237960 38 0,0408524 21,21306238801520 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 47 0,031		•	,
26 0,0519308 16,16387577314430 27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,77927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 46 0,0318288 25,76444372143560 47 0,031		,	•
27 0,0508992 16,49147727272730 28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800		•	
28 0,0508992 17,04894292463270 29 0,0492349 17,49860849661350 30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 46 0,0311911 26,91161901952800		•	•
30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 46 0,0311911 26,91161901952800	28		
30 0,0479697 17,95971594908650 31 0,0467381 18,43375160861030 32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 46 0,0311911 26,91161901952800	29	0,0492349	17,49860849661350
32 0,0455362 18,86358222505880 33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0311911 26,91161901952800		•	
33 0,0444986 19,59088935878230 34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76414372143560 47 0,0311911 26,91161901952800	31	0,0467381	18,43375160861030
34 0,0428466 20,19208148007380 35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	32	0,0455362	18,86358222505880
35 0,0415709 20,54721387237960 36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	33	0,0444986	19,59088935878230
36 0,0408524 21,21306238801520 37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	34	0,0428466	20,19208148007380
37 0,0395701 21,79927803459200 38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	35	0,0415709	20,54721387237960
38 0,038506 22,37757558469670 39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	36	0,0408524	21,21306238801520
39 0,0375109 22,84898059177400 40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	37	0,0395701	21,79927803459200
40 0,036737 23,61581809639290 41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	38	0,038506	22,37757558469670
41 0,0355441 24,08160864800350 42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	39	0,0375109	22,84898059177400
42 0,0348566 24,62603414891740 43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	40	0,036737	23,61581809639290
43 0,034086 25,26457884994510 44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	41	0,0355441	24,08160864800350
44 0,0332245 26,91161901952800 45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	42	0,0348566	24,62603414891740
45 0,0325799 26,37243628412000 46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	43	0,034086	25,26457884994510
46 0,0318288 25,76444372143560 47 0,0311911 26,91161901952800	44	0,0332245	26,91161901952800
47 0,0311911 26,91161901952800	45	0,0325799	26,37243628412000
, ,	46	0,0318288	25,76444372143560
48 0,0307166 27,32734091663790	47	0,0311911	26,91161901952800
	48	0,0307166	27,32734091663790

Wykres dla programu równoległego 2000 x 4000

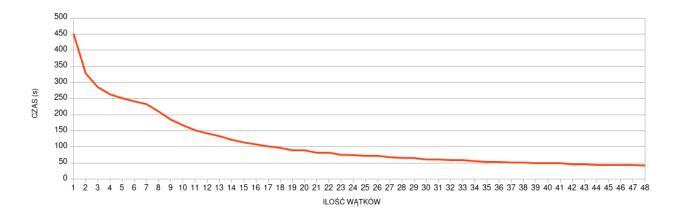


Wnioski dla małego zestawu danych

Jak widać na załaczonych danych problem bardzo dobrze zeskalował się dla niewielkiego zestawu danych. Benefity z wykorzystania wielu wątków do obliczeń okazały się bardzo zadowalające, co pozwoliło przyśpieszyć program 27-krotnie dzięku wykorzystaniu 48 wątków. Zmniejszając czas pracy z 0,839403 (s) do 0,0307166 (s)

Liczba wątków	Czas (s)	Przyśpieszenie
1	451,756	1
2	328,338	1,37588704322984
2 3	285,369	1,58305912695492
4	263,234	1,71617648176147
5	250,98	1,7999681249502
6	241,43	1,87116762622706
7	232,586	1,94231811029039
8	209,932	2,15191585846846
9	184,41	2,44973699907814
10	167,117	2,70323186749403
11	151,538	2,98114004408135
12	141,844	3,18487916302417
13	133,284	3,38942408691216
14	122,159	3,69809837998019
15	113,815	3,96921319685454
16	107,657	4,19625291434835
17	101,483	4,45154360828907
18	96,8686	4,66359584013808
19	89,166	5,06646030998363
20	88,913	5,08087681216470
21	82,1607	5,49844390322867
22	81,8736	5,51772488323465
23	75,333	5,99678759640529
24	75,333	6,08897125720255
25	74,1923	6,27878403771529
26 26	71,9490	6,25076620161861
27	67,7571	6,66728652790630
28	65,77	6,86872434240535
29	65,2121	6,92748738347638
30	60,9096	7,41682756084427
31	60,0615	7,52155707066923
32	59,0778	7,64679795117625
33	58,3506	7,74209691074299
34	55,3681	8,15913856534719
35	53,7035	8,41204018360070
36	53,6953	8,41332481613847
37	51,8469	8,71326926007148
38	50,4188	8,96007044991154
39	49,6051	9,10704746084576
40	48,5854	9,29818422818378
41	48,7111	9,27419007166744
42	46,2665	9,76421384803259
43	45,9828	9,82445610097689
44	43,4644	10,39370151204200
45	44,4877	10,15462700926320
46	44,1038	10,24301760846000
47	42,7668	10,56324064461220
48	41,9951	10,75735026229250
10	T1,0001	10,75755020223250

Wykres dla programu równoległego 100000 x150000



Wnioski dla dużego zestawu danych

Program na dużym zestawie danych również został wyraźnie przyśpieszony dzięki wykorzystaniu wielowątkowości. Zestaw ten nie skaluje się natomiast tak dobrze jak na małym zbiiorze danych, przyśpieszenie nie jest tak znaczące jak w przypadku poprzedniego testu, natomiast i tak znacząco przyśpieszyło to nasze obliczenia

Podsumowanie doświadczenia

Problem mnożenia macierzy przez wektor jest dobrym przykładem benefitów wykorzystania architektóry równoległej.

Zwiększenie ilości wątków wykorzystanej do obliczeń dobrze skaluję się z czasem potrzebnym do wykonania potrzebnych działań.

Zarówno dla małego jak i dużego zestawu danych czepiemy sporo korzyści, przyśpieszając naszą pracę. Warto natomiast podkreślić że współczynnik przyśpieszenia spada wraz z ilością przydzielonych wątków, spowodowane jest to chociażby koniecznością przydzielania operacji do wątku co jest wymagającym zadaniem czasowo dla sprzętu. Największe przyśpieszenie obserwujemy w zakresie od 1 do 8 wątków, oznacza to że później nie dostajemy już tak znaczącego przyśpieszenia, natomiast warto jest wykorzystać nadal większą ilość wątków dla zwiększenia efektywności obliczeń