Metody programowania równoległego

Sprawozdanie z metryk dla programów zrównoleglonych

autor: Ilona Tomkowicz

Akademia Górniczo-Hutnicza
Wydział Informatyki, Elektroniki i Telekomunikacji,
Informatyka, II stopień, I semestr

28 marca 2020

Contents

1. Opis przebiegu doświadczenia

Doświadczenie polegało na pomiarze czasu wykonania programu szacującego liczbę PI na podstawie algorytmu Monte Carlo. Wykonane zostało w trzech wersjach:

1. mały rozmiar 10^7 - na jednym wątku program wykonuje się poniżej sekundy (0,15s),
2. średni rozmiar 10^9,
3. duży rozmiar $4 \cdot 10^{10}$.

Każdy rozmiar został uruchomiony dla dwóch wersji:

1. nieskalowanej - ilość punktów jest stała dla danego typu rozmiaru problemu,
2. skalowanej - ilość punktów w danym typie problemu jest stała dla jednego procesora.

Programy zostały wykonane na architekturze VNODE z tego powodu, że Zeus nie działał. Każdy VNODE został ograniczony do jednego procesora. Po wykonaniu programów na VNODE ponawiano codziennie próby uruchomienia na Zeusie, jednak do chwili obecnej bez skutku. Załączam fragment zwróconych kodów błędu.

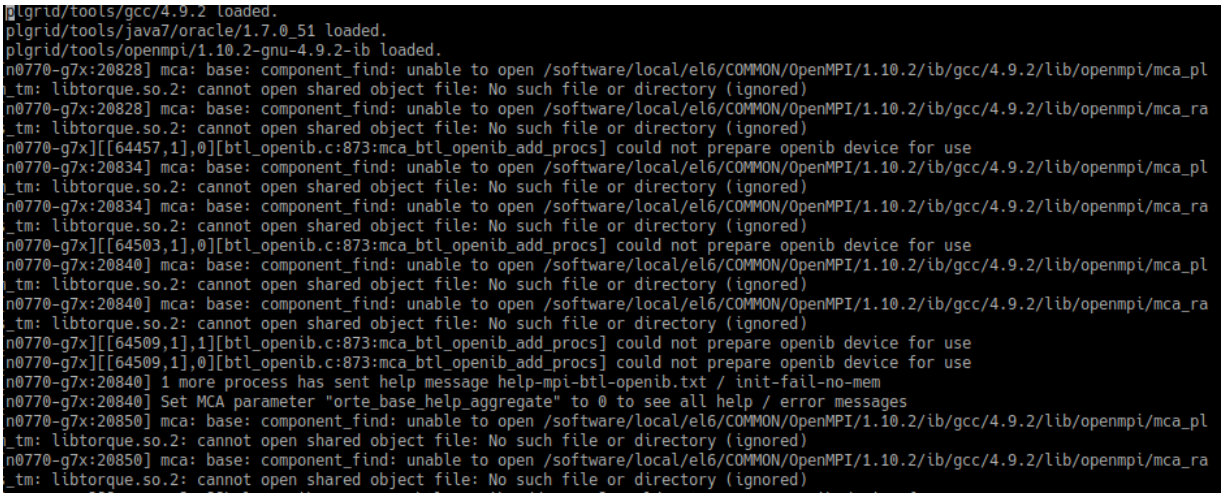

Figure 1.1: Wynik zwrócony po uruchomieniu z –constraint="intel" oraz z i bez flagi –exclusive.

Każda część eksperymentu została powtórzona trzykrotnie dla uśrednienia wyników i wyeliminowania błędów grubych, które mogłyby wynikać z chwilowego przeciążenia architektury przez innego użytkownika. Trzykrotne wykonanie jest niewielką próbką i dla uzyskania bardziej wiarygodnych wyników należałoby powtórzyć próbę kilkaset razy, ale z uwagi na dzielenie zasobów nie podjęto takich kroków. Do wyciągnięcia wniosków z zadanego ćwiczenia kilkurotne, zgrubne uśrednienie jest wystarczające.

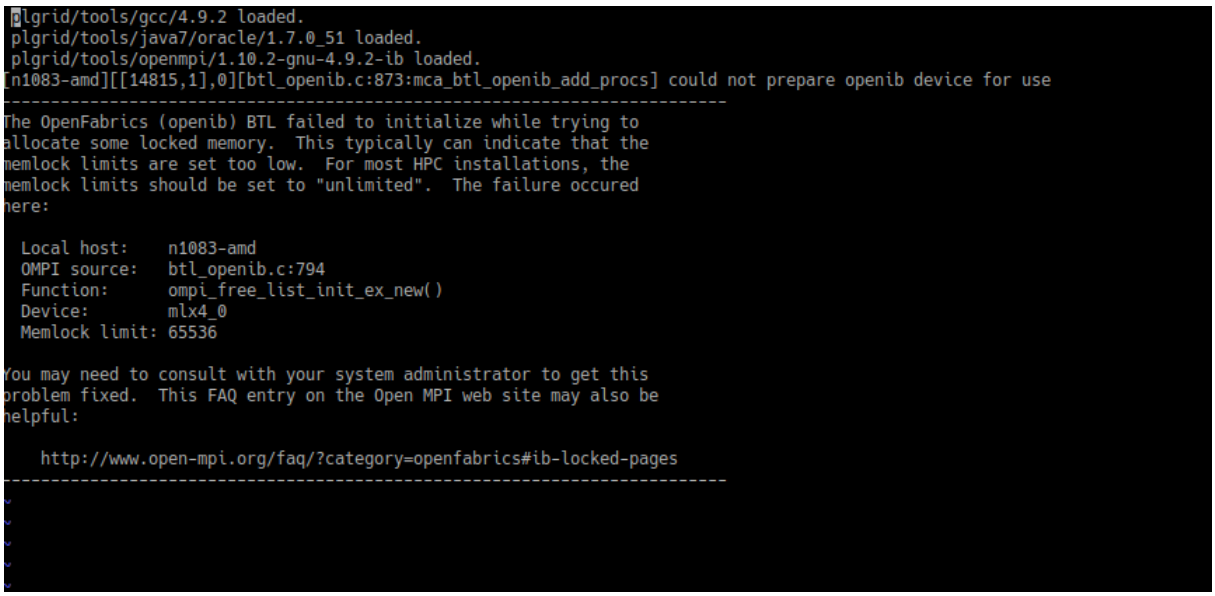

Figure 1.2: Wynik zwrócony po uruchomieniu bez –constraint="intel" i bez –exclusive.

Polecam do otwarcia Excela użyć wersji desktopowej, bo w takiej był przygotowany. Otwarcie w przeglądarce zmienia wygląd wykresów.

2. Wykresy

Do przedstawienia wyników zostały użyte wykresy liniowe, bez zazanaczenia punktów pomiaru aby nie zaciemniać wykresu. Dla zależności czasu wykonania od ilości procesorów umieszczono jednocześnie na wykresie wartości z wersji skalowane i nieskalowanej, dlatego że wartości mają podobne rzędy wielkości. Przy umieszczeniu osobno na dwóch wykresach: trzech trendów skalowanych i trzech nieskalowanych, czasy dla problemów małych i średnich wygladają jak płaskie kreski na poziomie 0 w porównaniu z problemem dużym. Nie dawało to wystarczającej informacji o kształcie wykresów, a informacja ta jest ważna, więc w przedstawieniu czasów przyjęto inną formę niż w pozostałych metrykach. Wykresy przyśpieszenia, efektywności i części sekwencyjnej są sporządzone według sugestii z instrukcji do ćwiczenia.

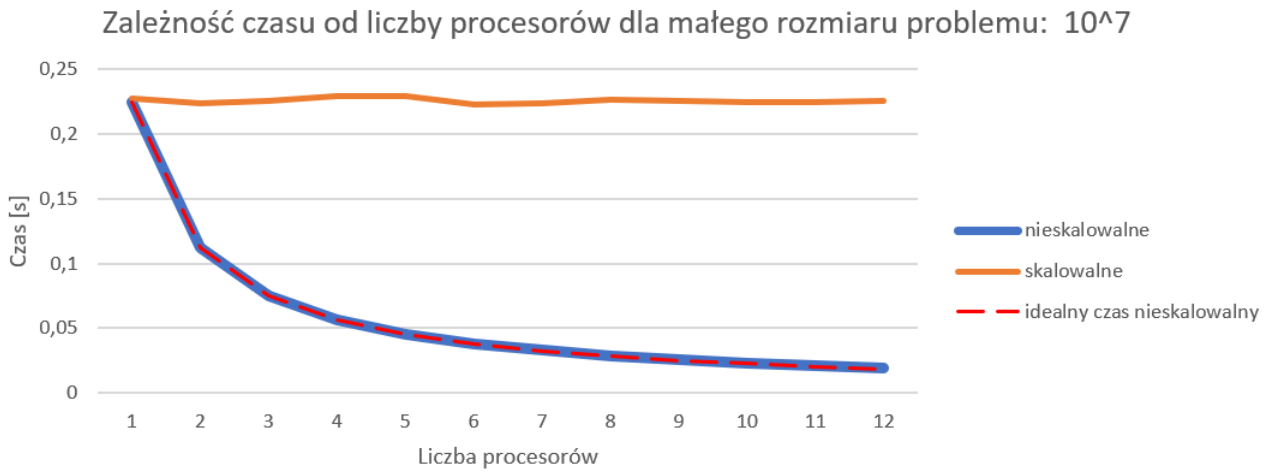

Figure 2.1: Wykres czasu dla małego problemu.

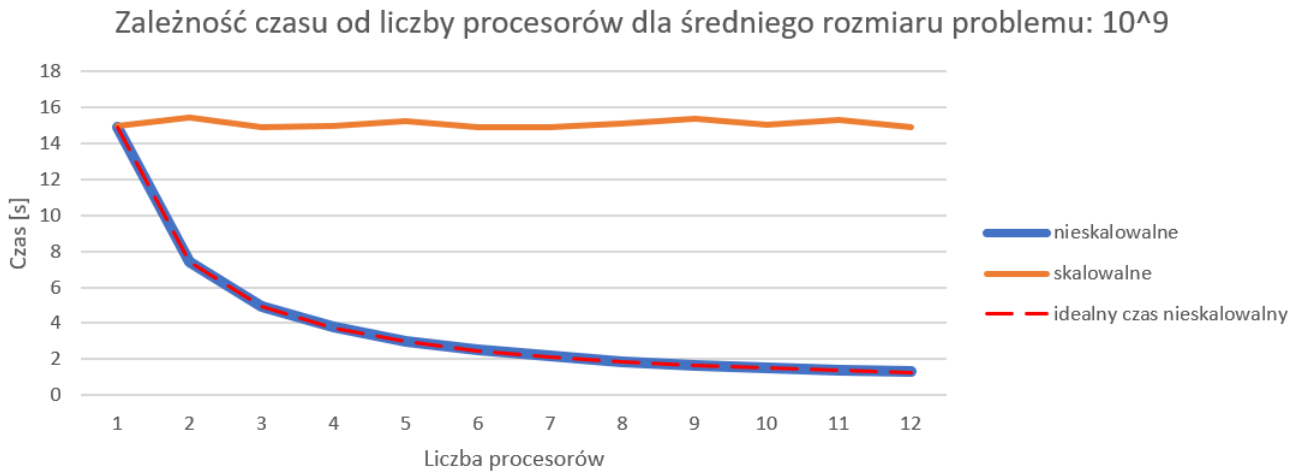

Figure 2.2: Wykres czasu dla średniego problemu.

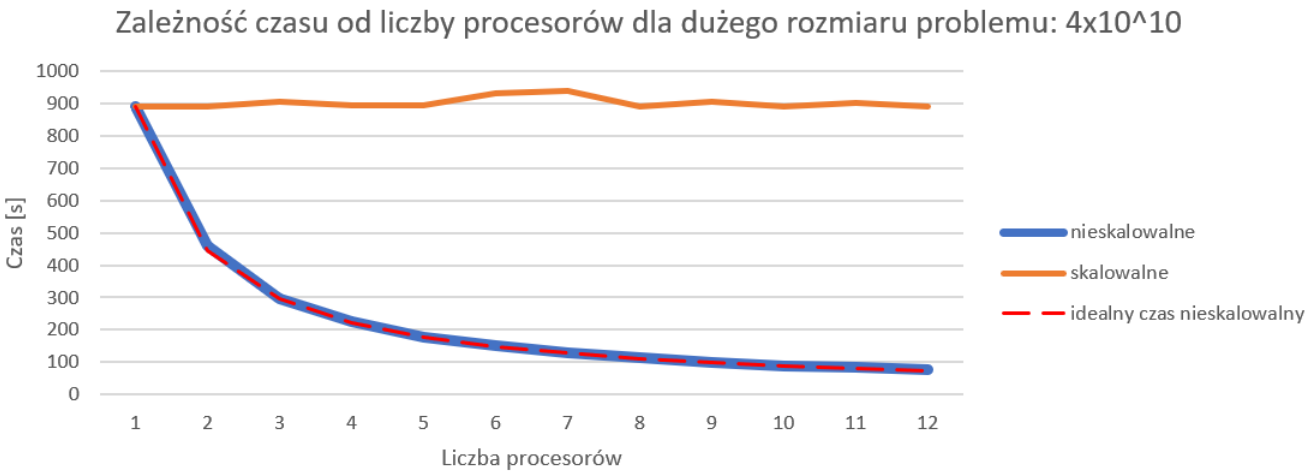

Figure 2.3: Wykres czasu dla dużego problemu.

Jak można zauważyć wszystkie trzy wykresy są bardzo do siebie podobne. Główną różnicą jest skala na osi pionowej, wahająca się od części setnych sekundy po tysiąc sekund.

Dla warsji ze skalowaniem idealną charakterystyką jest linia prosta. Biorąc pod uwagę małą skalę wykresu dla małych danych czas dla ćwiczenia skalowanego jest za każdym razem taki sam. Jest to tak mały problem, że "nie ma czasu" na zarejestrowanie różnic. Dla skalowania średniego wahania - zważając na skalę - są już większe. Widać fluktuacje wielkości, które jednak są na tyle losowe, że mieszczą się w granicach błędu statystycznego. Dla dużego problemu widać uskok na poziomie 6-7 procesorów. Jest to różnica rzędu 20-30 sekund, co jest znaczącym skokiem w porównaniu z poprzednio opisywanymi trendami. Według szczegółowych danych z Excela widać, że skok nie jest spowodowany podniesieniem wartości jednej próbki, lecz wszystkich trzech na raz. Uważam więc, że jest to cecha architektury, na której program został wykonany. Najwyraźniej któryś z procesorów , który został włączony do użycia na poziomie 6-7 jest inny niż wcześniej używane procesory

- program wykonuje się na nim nieznacznie dłużej i przez to reszta musiała czekać, co wydłużyło sumaryczny czas pracy.

W wersji bez skalowania idealną charakterystykę uzyskano dzieląc czas referencyjny (wykonania programu na 1 procesorze) przez liczbę procesorów. Jak widać trend "idealny" pokrywa się dla każdego rozmiaru problemu z trendem nieskalowanym. Jedynie dla dużego problemu lekko podnosi się na poziomie dwóch procesorów - tylko jeden pomiar na 3 jest zgodny z idealnym wynikiem. Mogłoby to oznaczać tak jak poprzednio użycie innego procesora, jednak w tym przypadku może to być również błąd losowy, który zniknąłby po wykonaniu większej liczby pomiarów.

2.1 Metryki skalowane

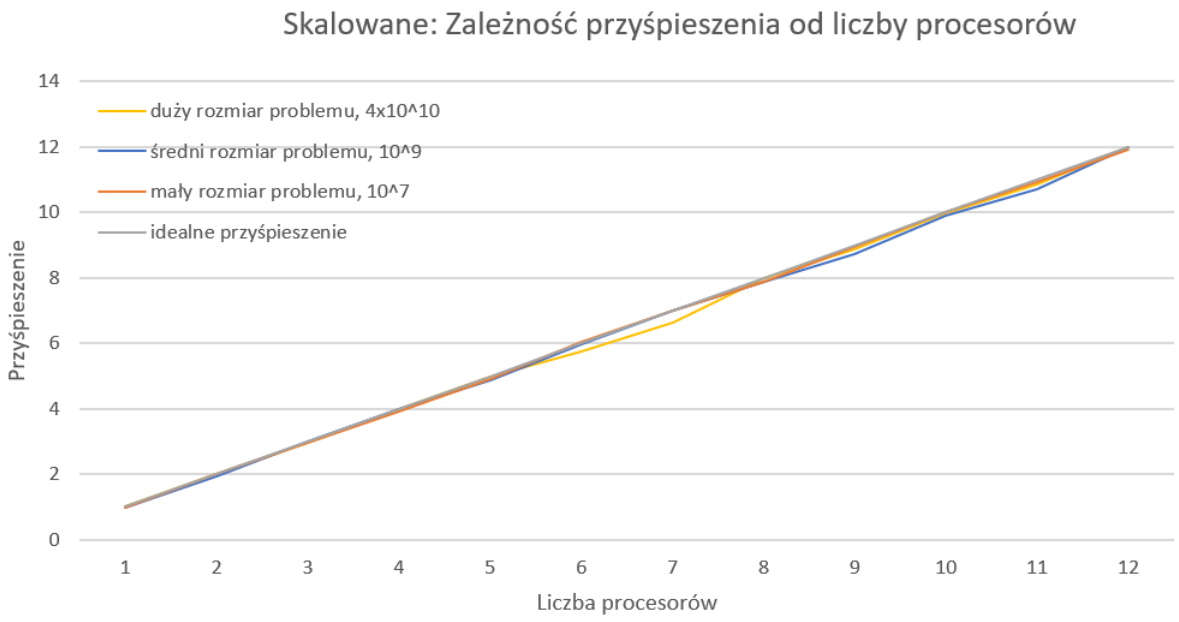

Figure 2.4: Wykres przyśpieszenia dla małego problemu.

Linie prawie idealnie się pokrywają poza odchyleniem dla dużego problemu na poziomie 6-7 procesora. Jest to zrozumiałe, że wydłużony czas wykonania programu zmniejsza przyśpieszenie.

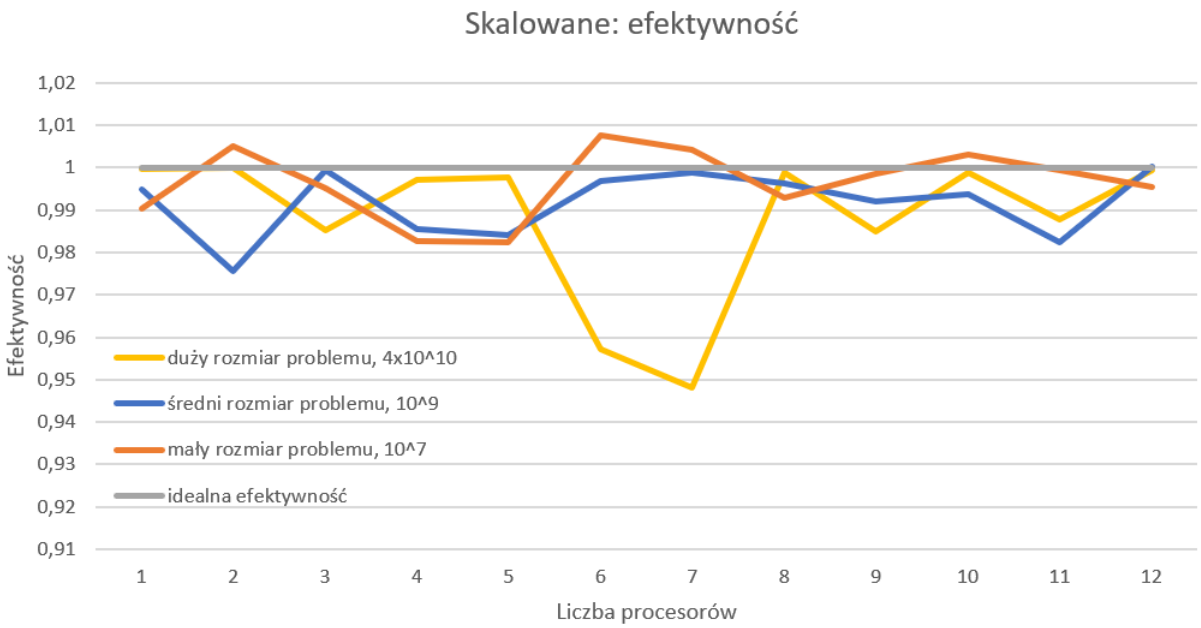

Figure 2.5: Wykres efektywności dla średniego problemu.

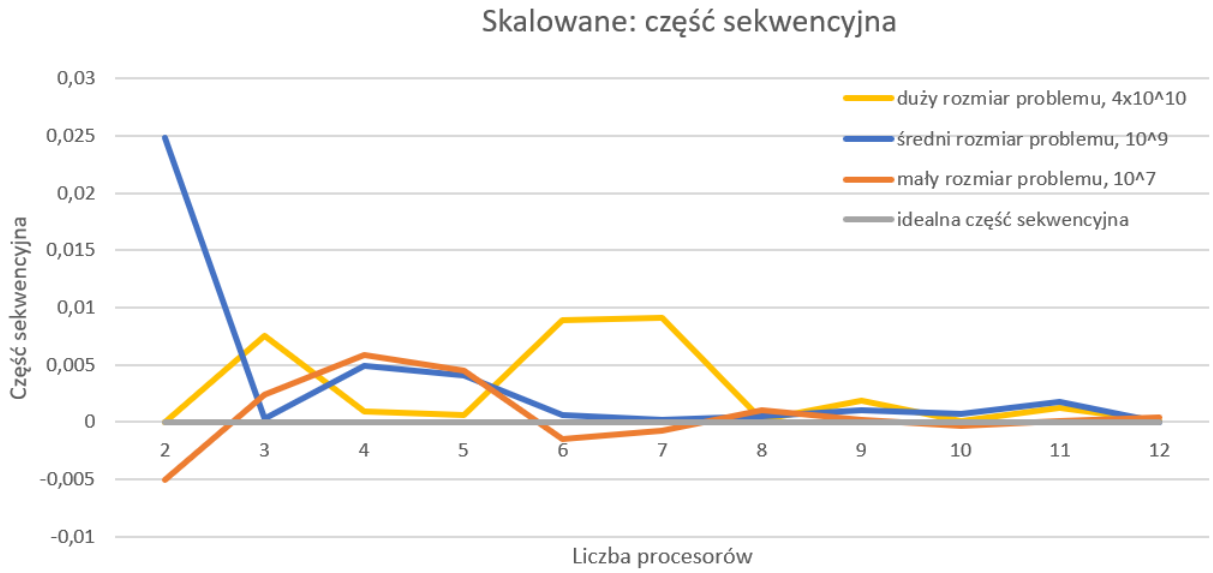

Figure 2.6: Wykres udziału części sekwencyjnej dla dużego problemu.

Skok w czasie dla dużego problemu skalowalnego na poziomie 6-7 procesorów jest szczególnie widoczny tutaj - w metrykach efektywności i części sekwencyjnej. Efektywność widocznie spada, a udział części sekwencyjnej

rośnie w tych obszarach. Poza tym dość dużym odchyleniem od normy charakteryzuje się trend średni dla części sekwencyjnej przy 2 procesorach. Nie ma to odzwierciedlenia w innych wykresach - pomiary dla p=2 i średniego rozmiaru nie odbiegają znacznie od reszty dla czasu, przyśpieszenia czy efektywności. Uważam, że jest to cecha tego rodzaju metryki - dla nawet małych błedów w liczniku wzoru przy małej liczbie procesorów błąd ten jest mnożony razy 2. Przy 3 procesorach błąd mnożymy już tylko razy 1,5 itd. Łatwo więc o duży skok spowodowany małym błędem pomiarowym właśnie szczególnie gdy procesorów jest mało. Tę teorię potwierdza też sam wykres - fluktuacje wygaszają się wraz ze wzrostem liczby procesorów.

Reszta wahań w tych wykresach jest marginalna i jest skutkiem małych odchyleń losowych, o których była mowa w omówieniu wykresu zależności czasu od liczby procesorów. Efektywność powyżej 100% dla małego problemu nie jest objawem super liniowości (nie ma podstaw do jej uzyskania, typu eliminacja przepełnień cache), lecz błędów pomiaru dla małej próbki i małego rozmiaru problemu. Tego typu zjawiska nie obserwujemy już dla średniego i dużego problemu, bo wyniki dla dużych prób są wiarygodniejsze i mniej podatne na tego typu błędy.

2.2 Metryki nieskalowane

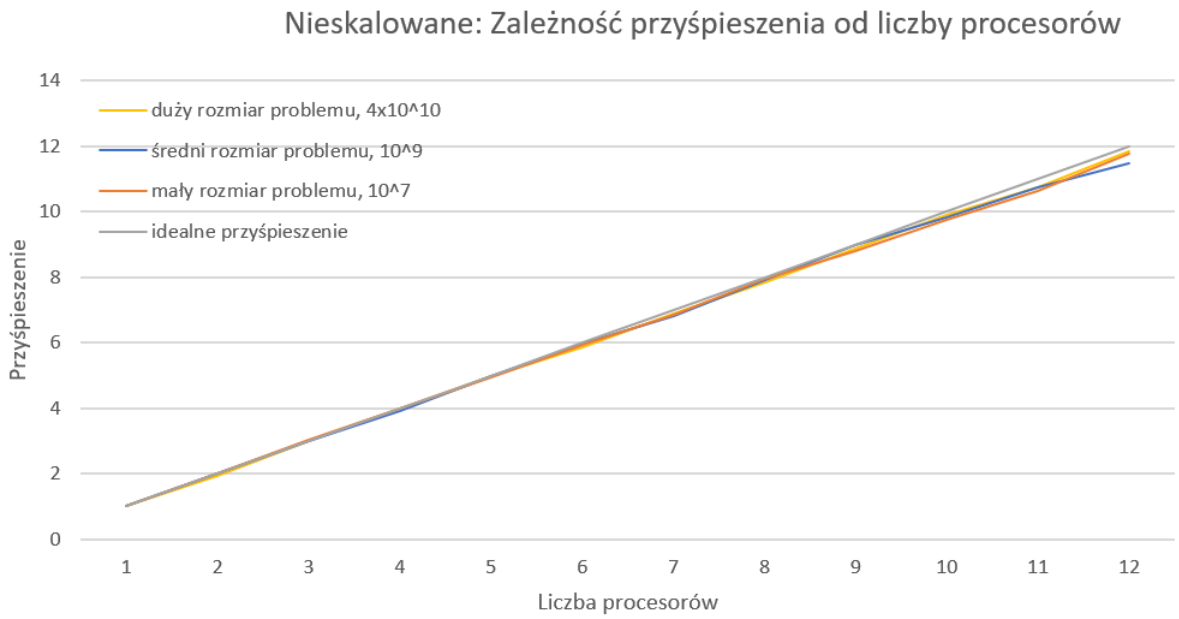

Figure 2.7: Wykres przyśpieszenia dla małego problemu.

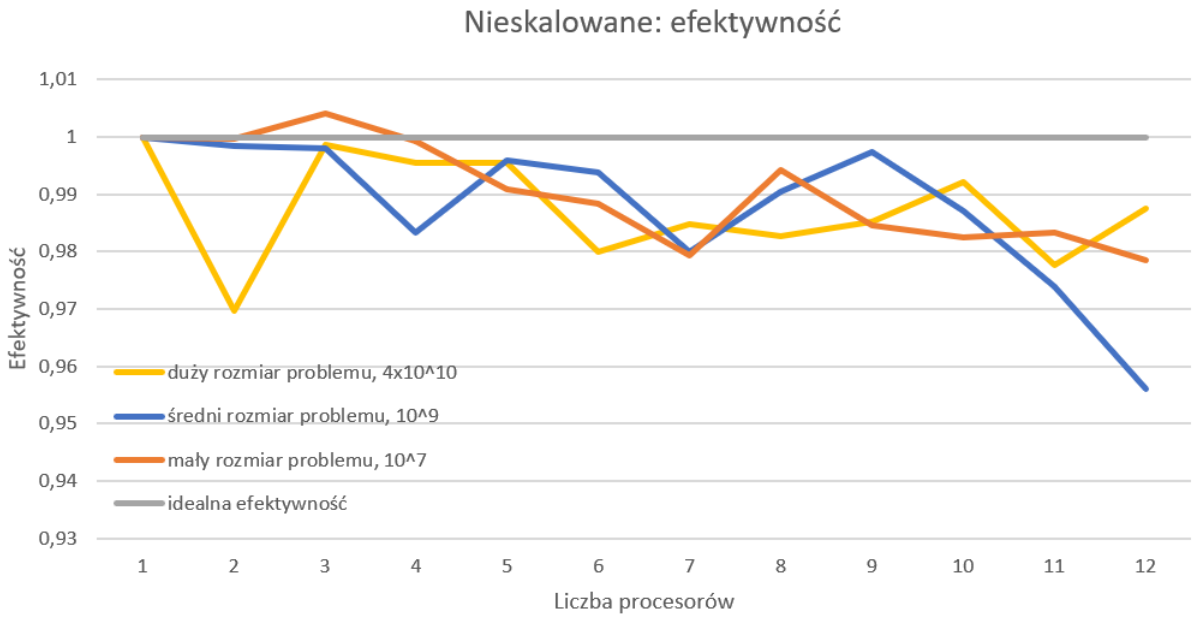

Figure 2.8: Wykres efektywności dla średniego problemu.

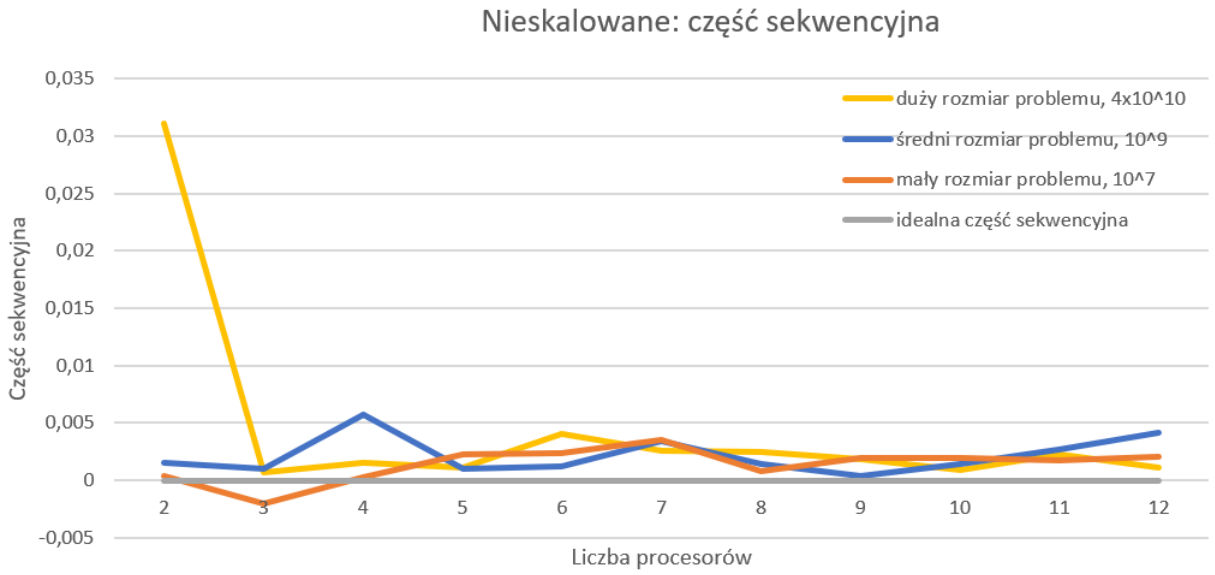

Figure 2.9: Wykres udziału części sekwencyjnej dla dużego problemu.

Tak samo jak w przypadku skalowalnym można zauważyć efektywność ponad 100% dla małego problemu. Wytłumaczenie jest takie samo. Podobnie jak dla wykresu skalowanego: dla części sekwencyjnej występuje pik

przy małej liczbie procesorów.

3. Użyte wzory

Przyśpieszenie nieskalowane. T(1) - czas dla 1 procesora, T(p) - czas dla p procesorów.

$$s(p) = \frac{T(1)}{T(p)} \tag{3.1}$$

Przyśpieszenie skalowane. T(1,1) - czas dla 1 procesora i skali k=1, T(p, k) - czas dla p procesorów i zadania przeskalowanego o skalę k.

$$s(p,k) = k\frac{T(1,1)}{T(p,k)} \tag{3.2}$$

Efektywność (dla skalowanej bierzemy s(p,k) zamiast s(p) do wzoru):

$$E(p) = \frac{s(p)}{p} \tag{3.3}$$

Część sekwencyjna (dla skalowanej bierzemy s(p,k) zamiast s(p) do wzoru):

$$f(p) = \frac{\frac{1}{s(p)} - \frac{1}{p}}{1 - \frac{1}{p}} \tag{3.4}$$