# Sprawozdanie
## Teoria Współbieżności
## Niezabezpieczona modyfikacja wartości

Akademia Górniczo-Hutnicza w Krakowie
05.10.2024 r.
Bartosz Sajecki

## 1. Dane techniczne

Obliczenia przeprowadzane były na laptopie z systemem Manjaro Linux
Kernel 6.1.77-2-MANJARO (64-bit)
Procesor 16 x AMD Ryzen 7 4800H
Pamięć operacyjna 16 GB

## 2. Cel ćwiczenia

Uruchamiając jednocześnie wiele wątków modyfikujących pamięć współdzieloną zbadać co szybciej doprowadzi do pojawienia się błędów, zwiększanie liczby wątków czy zwiększanie zapotrzebowania poszczególnych wątków na zasoby komputera.

## 3. Wstęp teoretyczny

Nawet najprostsza modyfikacja wartości zmiennej w języku Java, wykonywana na przykład przy pomocy instrukcji `i += 1;` wymaga tego, aby wartość zmiennej została najpierw wczytana do rejestru procesora po to, aby następne do wartości tej dodać pewną liczbę. Po operacji dodawania konieczne jest zapisanie otrzymanej wartości do miejsca w pamięci programu, z którego została ona pierwotnie pobrana.

Przykładowo, poniższy fragment kodu w Javie

```
int i = 0;
for (int j = 0; j < 10; j++) {
    i += j;
}
```

jest tłumaczony na następujące sekwencje bytecode, wykonywane następnie przez maszynę wirtualną Javy

```
0: iconst_0
1: istore_1    // wczytaj i=0
2: iconst_0
3: istore_2    // wczytaj j=0 (licznik pętli)
4: iload_2     // załaduj wartość j
5: bipush      10   // dodaj wartość 10 na stos (warunek końca)
7: if_icmpge   20   // idź do instrukcji 20 jeśli j≥10
10: iload_1    // wczytaj i
11: iload_2    // wczytaj j
12: iadd       // dodaj i+j
13: istore_1   // zapisz wynik dodawania w zmiennej i
14: iinc 2, 1  // zwiększ wartość j o 1
17: goto 4     // rozpocznij pętlę od początku
```

Instrukcje wczytywania, inkrementacji oraz zapisywania wartości zmiennej znajdują się w linijkach oznaczonych numerami 10-13.

## 3.1. Dostęp wielu wątków do tego samego adresu w pamięci komputera

Wszystko działa bez zarzutu gdy program uruchamiamy w jednym wątku i instrukcje maszyny wirtualnej uruchamiane są po kolei. Jednak w momencie, gdy mamy do czynienia na przykład z dwoma wątkami współdzielącymi tę samą przestrzeń adresową w pamięci komputera, zdarzyć się może, że jeden wątek wczyta wartość zmiennej, ale zanim zdąży ją zwiększyć i zapisać nową wartość, to inny wątek zdąży odczytać jeszcze nienadpisaną wartość, zmodyfikować ją po swojemu, a następnie zapisać. Wtedy pierwszy wątek modyfikuje zmienną i zapisuje ją do pamięci komputera, nadpisując efekt pracy drugiego wątku.
Sytuacja taka może być mało prawdopodobna, natomiast jeśli komputer jest obciążony innymi procesami lub gdy kalkulacja trwa długo, to szansa na powstanie błędu realnie się zwiększa.

## 4. Przebieg ćwiczenia

Przebadano następujące wartości liczby wątków oraz długości pętli inkrementującej:
Ilość wątków: $[1,2,3,5,10,15,30,50,100,200,500,1000,2000,3000,4000,5000,10000]$
Długość pętli: $[1,2,3,5,10,15,30,50,100,200,500,1000,2000,3000,4000,5000,10000]$

Dla każdej liczby wątków z powyższej listy tworzono jeden wątek inkrementujący zmienną i jeden ją dekrementujący, które uruchamiają pętle modyfikujące zmienną znajdującą się pod tym samym adresem w pamięci o długości pracy pętli z zakresu wartości podanych w liście powyżej. Ponieważ wątki tworzono parami, końcowa wartość modyfikowanej zmiennej powinna wynosić 0. Na koniec pracy programu wypisano ostateczną wartość zmiennej na której program zakończył swoją pracę.

## 5. Wyniki obliczeń

W poniższej Tabeli 1. liczby w pierwszym wierszu to liczba wątków modyfikujących zmienną uruchomiona jednocześnie, a liczba w pierwszej kolumnie to długość pętli modyfikującej zmienną. Wartość zapisana w komórce to wartość końcowa modyfikowanej zmiennej.

| wątki / iteracje | 1 | 2 | 3 | 5 | 10 | 15 | 30 | 50 | 100 | 200 | 500 | 1000 | 2000 | 3000 | 4000 | 5000 | 10000 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -3 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 30 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 50 | 0 | 0 | 0 |
| 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 0 | 0 |
| 200 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 500 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -369 | 0 | 0 | 0 | 415 | 0 |
| 1000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1000 | 353 | 3327 | -1000 |
| 2000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 536 | 0 | 517 | -3028 | 0 | -448 |
| 3000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -4635 | 0 | 0 | 2396 | 0 | -200 | -985 | -4212 |
| 4000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 721 | 5278 | 6011 | 4731 | 12732 |
| 5000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1083 | -4009 | 5284 | 5188 |
| 10000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4183 | 0 | 2649 | -2335 | 0 | -13190 | 24748 | 36939 | 29659 |

Wynik działania programu jest różny od zera na przykład dla dziesięciu tysięcy wątków przy pętli modyfikującej składającej się z trzech iteracji, i jest to dosyć nietypowo występujący błąd biorąc pod uwagę pozostałe wyniki.

## 6. Analiza wyników

Dla pętli o długości dziesięciu tysięcy iteracji pierwszy błąd pojawił się przy stu wątkach, natomiast przy dziesięciu tysiącach wątków pierwszy błąd pojawił się już przy pętli o długości trzech iteracji, lub, gdyby pominąć ten wynik, to przy pętli długości tysiąca iteracji.
Jednak dla innych, mniejszych liczb wątków błędy pojawiały się jeszcze wcześniej. Przykładowo dla trzech tysięcy wątków pierwszy błąd wystąpił przy pętli długości pięćdziesięciu iteracji.

## 7. Wnioski

Aplikacje wielowątkowe podatne są na zakłamanie wyników obliczeń spowodowane przez dostęp do pamięci współdzielonej który nie jest synchronizowany pomiędzy wątkami.
W przypadku wykonanego eksperymentu większe znaczenie ma liczba utworzonych wątków i to jej wzrost szybciej powoduje wystąpienie błędów.
Zwiększanie liczby iteracji w metodzie modyfikującej współdzieloną wartość, poza prowokowaniem błędu będącego tematem badania, skutkuje również większym odchyłem wyniku od wartości spodziewanej wynoszącej zero.
Pomimo, że wzrost ilości wątków ma większy wpływ na wystąpienie błędów, to różnica ta nie jest znacząca w praktyce. Najważniejsze jest utrzymanie zarówno niskiej liczby wątków jak i liczby iteracji aby otrzymywać poprawne wyniki przy niezabezpieczonej pamięci programu.