Optymalizacja Kodu Na Różne Architektury
Zadanie 2

1 Ogólne zasady zaliczenia

- Zadanie oceniane jest w skali od 0-50 punktów.
- Warunkiem koniecznym jest indywidualne oddanie zadania.
- Ocenie podlega m.in. stopień zrozumienia mechanizmów odpowiedzialnych za obserowanane zmiany w wydajności.
- Weryfikacja posiadanego modelu procesora i **dopasowanie do jego specyfikacji zooptymalizowanego kodu** (cache i jednostki wektorowe, w tym ich typ i generacja, oraz inne) jest konieczne do uzyskania pozytywnej oceny.
- Zadania umieszczone w systemie Moodle/UPEL po terminie będą bezwględnie oceniane na maksymalnie 25 punktów.

2 Zadanie

Proszę o wykonanie optymalizacji algorytmu eliminacji Gaussa wzorując się na laboratoriach oraz poniższym instruktażu.
https://github.com/flame/how-to-optimize-gemm
Proszę o umieszczenie w systemie UPEL/Moodle sprawozdania w formacie pdf. Mile widzine będą zwięzłe sprawozdania.

Proszę napisać prosty algorytm weryfikujący, czy dla zadanego rozmiaru macierzy nie zmienia się zwracane przez optymalizowaną funkcję rozwiązanie. Proszę uwzględnić rzeczywistą złożoność obliczeniową przy

szacowaniu GFLOPS-ów. Mile widziane będą pomiary liczników procesora (PAPI) dające rzeczywisty odczyt mocy obliczeniowej uzyskiwanej przez procesor.

Bardzo proszę o podkreślanie wniosków w sprawozdaniu, oraz dopasowanie rozwiązania do możliwości posiadanego modelu procesora.

Bardzo istotne są:

- mierzenie wysycenia FLOPS-ów procesora
- poprawna wektoryzacja obliczeń
- lokalność danych

2.1 Cel zadania

Celem zadania jest implementacja i optymalizacja algorytmu eliminacji Gaussa, z uwzględnieniem mikroarchitektury posiadanego procesora oraz praktycznego wykorzystania technik niskopoziomowej optymalizacji kodu.

2.2 Zakres prac

1. Zaimplementuj algorytm eliminacji Gaussa (z lub bez pivotingu).
2. Zoptymalizuj implementację pod kątem architektury CPU (cache, SIMD, itp.).
3. Zweryfikuj poprawność: wynik optymalizowanej wersji musi odpowiadać wersji referencyjnej.
4. Oszacuj liczby operacji (FLOP) i przelicz GFLOPS.
5. Przeprowadź pomiary wydajności (opcjonalnie z użyciem PAPI lub równoważnego narzędzia).

2.3 Wymagania techniczne

- Wektoryzacja obliczeń z użyciem instrukcji SIMD (np. AVX, SSE).
- Dbałość o lokalność danych (wykorzystanie pamięci cache).
- Analiza architektury CPU i dostosowanie kodu do jednostek wykonawczych.
- Pomiary czasu wykonania oraz obliczonych GFLOPS-ów.
- Wykonanie testów poprawności dla różnych rozmiarów macierzy.

3 Tabela ocen

Kryterium oceny	Opis szczegółowy	Punktacja
Sprawozdanie	Czytelność, struktura	5 pkt
Weryfikacja poprawności	Poprawny wynik	10 pkt
Optymalizacja	SIMD, pamięć, algorytmy	15 pkt
Analiza CPU	Architektura, wektoryzacja, cache	10 pkt
Pomiary wydajności	GFLOPS, liczniki sprzętowe	10 pkt
Łącznie		**50 pkt**

Uwaga: Prace oddane po terminie będą oceniane na maksymalnie 25 punktów.