

АРХИТЕКТУРА РАЧУНАРА ПРОЈЕКТИ
ЗАДАТАК 2.2 – Оптимизација алгоритма
ТЕМА: Линеарна регресија

Аутор: Новица Тепић

Верзија: 1.0

Датум: 25.1.2023.

Садржај

Увод.....	3
Поступак и услови тестирања	3
Поређење времена извршавања	4
Објашњење логике свих програма	8
Просјечне вриједности и варијансе времена извршавања програма	9
Графички приказ резултата	11
Закључак	14

Увод

За оптимизацију алгоритма кориштен је AVX инструкцијски скуп, као и паралелизација на вишејезгреном процесору употребом OpenMP. Четири програма су реализована у С програмском језику (програм без оптимизација, AVX, OpenMP као и „обједињени“ програм који садржи AVX инструкције и OpenMP).

Поступак и услови тестирања

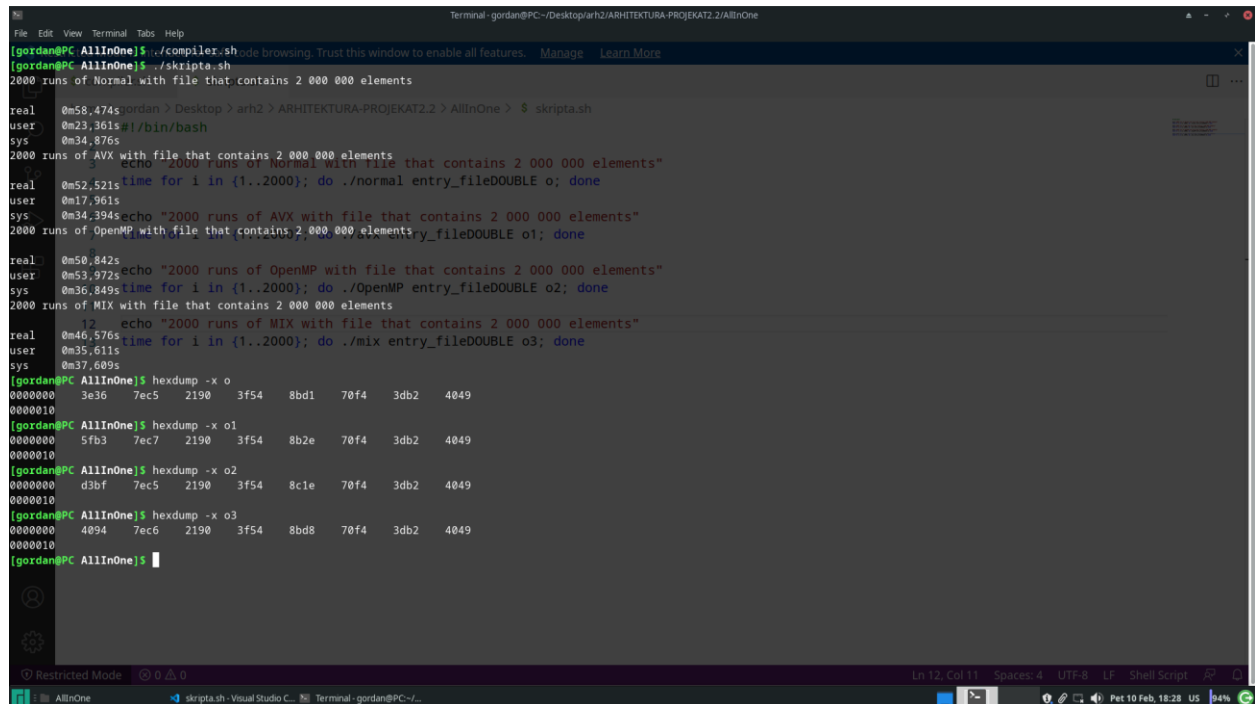
Кориштене су двије скрипте:

-прва скрипта (compiler.sh) је кориштена за компајлирање и сви програми су компајлери са –O0 оптимизацијом, с тим да је за AVX кориштен –mavx2 flag, за OpenMP је кориштен –fopenmp flag, а за MIX варијанту су кориштени и –mavx2 и –fopenmp flag.

-друга скрипта (skripta.sh) је кориштена за мјерење времена сва четири програма те њен садржај зависи од величине елемената (што је више елемената, мањи је број извршавања да се не би предуго чекало на крај извршавања програма, а да се јасно виде резултати).

Процесор који је кориштен је Intel i7-4510U (два физичка језгра и четири логичка језгра, а основна фреквенција рада је 2.00Ghz, док при већем оптерећењу иде и до 3.10GHz), а оперативни систем је Manjaro Linux (Arch дистрибуција).

Поређење времена извршавања



```
gordan@PC AllInOne]$ ./compiler.sh
gordan@PC AllInOne]$ ./skripta.sh
2000 runs of Normal with file that contains 2 000 000 elements
real    0m58.474s
user    0m23.361s
sys     0m34.876s
gordan@PC AllInOne]$ #!/bin/bash
echo "2000 runs of Normal with file that contains 2 000 000 elements"
time for i in {1..2000}; do ./normal entry_fileDOUBLE o; done
real    0m52.521s
user    0m17.961s
sys     0m34.394s
echo "2000 runs of AVX with file that contains 2 000 000 elements"
time for i in {1..2000}; do ./avx entry_fileDOUBLE o1; done
real    0m50.842s
user    0m53.972s
sys     0m36.849s
echo "2000 runs of OpenMP with file that contains 2 000 000 elements"
time for i in {1..2000}; do ./openmp entry_fileDOUBLE o2; done
real    0m46.576s
user    0m35.611s
sys     0m37.609s
echo "2000 runs of MIX with file that contains 2 000 000 elements"
time for i in {1..2000}; do ./mix entry_fileDOUBLE o3; done
gordan@PC AllInOne]$ hexdump -x o
00000000  3e36  7ec5  2190  3f54  8bd1  70f4  3db2  4049
00000010
gordan@PC AllInOne]$ hexdump -x o1
00000000  5fb3  7ec7  2190  3f54  8b2e  70f4  3db2  4049
00000010
gordan@PC AllInOne]$ hexdump -x o2
00000000  d3bf  7ec5  2190  3f54  8c1e  70f4  3db2  4049
00000010
gordan@PC AllInOne]$ hexdump -x o3
00000000  4094  7ec6  2190  3f54  8bd8  70f4  3db2  4049
00000010
gordan@PC AllInOne]$
```

1-2000 извршавања, фајл који има 2 000 000 елемената

Вриједност за о фајл (без оптимизација):

A = 0.001228705509

B = 50.482008094222367494177

Вриједност за о1 фајл (оптимизације помоћу AVX):

A = 0.0012287055097796983

B = 50.4820080942212

Вриједност за о2 фајл (OpenMP):

A = 0.0012287055097577185

B = 50.482008094222905

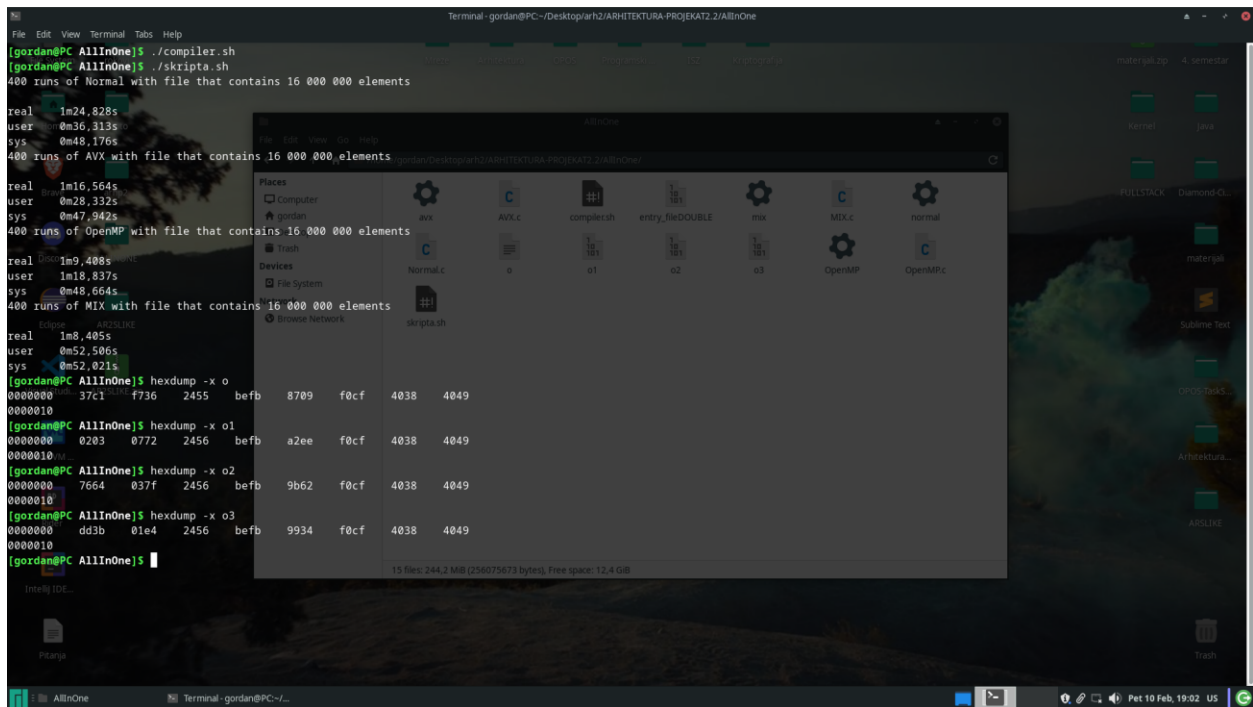
Новица Тепић 1102/20

Вриједност за о3 фајл (OpenMP+AVX):

A = 0.00122870550976376

B = 50.48200809422241

Аналогно вриједи и за остале резултате (може се видјети и на основу хексадецималних бројева).



```
gordan@PC AllInOne$ ./compiler.sh
gordan@PC AllInOne$ ./skripta.sh
400 runs of Normal with file that contains 16 000 000 elements
real 1m24.828s
user 0m36.313s
sys 0m48.176s
400 runs of AVX with file that contains 16 000 000 elements
real 1m16.564s
user 0m28.332s
sys 0m47.942s
400 runs of OpenMP with file that contains 16 000 000 elements
real 1m9.488s
user 0m18.837s
sys 0m48.664s
400 runs of MIX with file that contains 16 000 000 elements
real 1m8.405s
user 0m52.586s
sys 0m52.021s
gordan@PC AllInOne$ hexdump -x o
00000000 37c1 4736 2456 befb 8709 f0cf 4038 4049
00000010
gordan@PC AllInOne$ hexdump -x o1
00000000 0203 0772 2456 befb a2ee f0cf 4038 4049
00000010
gordan@PC AllInOne$ hexdump -x o2
00000000 7664 037f 2456 befb 9b62 f0cf 4038 4049
00000010
gordan@PC AllInOne$ hexdump -x o3
00000000 dd3b 01e4 2456 befb 9934 f0cf 4038 4049
00000010
gordan@PC AllInOne$
```

2-400 извршавања, фајл који има 16 милиона елемената

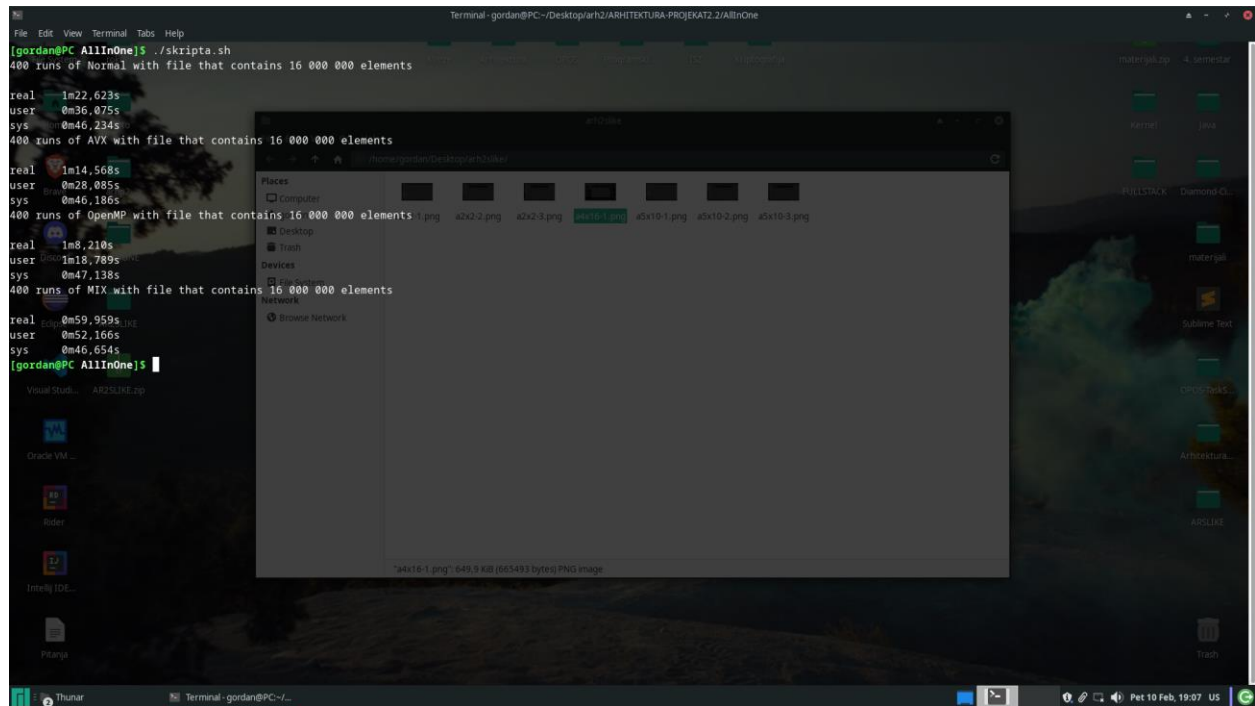
```

Terminal - gordan@PC~/Desktop/arh2/ARHITEKTURA-PROJEKAT2.2/AllInOne
File Edit View Terminal Tabs Help
[gordan@PC AllInOne]$ ./skripta.sh code browsing. Trust this window to enable all features. Manage Learn More
500 runs of Normal with file that contains 10 000 000 elements
$ compiler.sh $ skripta.sh X C main.c
real 1m3,756s
user 0m28,283s
sys 0m35,226s
500 runs of AVX with file that contains 10 000 000 elements
$ echo "500 runs of Normal with file that contains 10 000 000 elements"
time for i in {1..500}; do ./normal entry_fileDOUBLE o; done
real 0m57,727s
user 0m22,138s
sys 0m35,361s
500 runs of OpenMP with file that contains 10 000 000 elements
time for i in {1..500}; do ./avx entry_fileDOUBLE o1; done
real 0m51,847s
user 1m2,010s
sys 0m35,578s
500 runs of MIX with file that contains 10 000 000 elements
time for i in {1..500}; do ./mix entry_fileDOUBLE o2; done
real 0m45,633s
user 0m40,373s
sys 0m35,414s
[gordan@PC AllInOne]$ hexdump -x o
00000000 90b9 f054 1c05 3f1d c406 b79f 3f51 4049
00000010
[gordan@PC AllInOne]$ hexdump -x o1
00000000 4c5c eecd 1c05 3f1d cec8 b79f 3f51 4049
00000010
[gordan@PC AllInOne]$ hexdump -x o2
00000000 9580 ef58 1c05 3f1d cab6 b79f 3f51 4049
00000010
[gordan@PC AllInOne]$ hexdump -x o3
00000000 1812 eece 1c05 3f1d ce86 b79f 3f51 4049
00000010
[gordan@PC AllInOne]$

```

3-500 izvršavanja, fajl koji ima 10 000 000 elemenata

Треба да напоменем да на другом приложеном screenshot-у вријеме није било одговарајуће за MIX варијанту, па сам извршио програм још пар пута и добио боља времена, што ће бити приложено касније у табели али и помоћу одговарајућег screenshot-a:



Приказани су и резултати извршавања при чему о фајл има резултате основног програма без оптимизација, о1 резултате гдје су кориштене AVX инструкције, о2 резултате са OpenMP варијантом, а о3 резултате са комбинованим приступом, тј. OpenMP и AVX варијантом.

Као што се види, постоје мала одступања у резултатима али су она незнатна (односе се на мјеста са десне стране зареза, као и у асемблерском програму на првом пројектном задатку).

Очигледно је да су убрзања постигнута на основу измјерених времена.

Објашњење логике свих програма

Што се тиче програма без оптимизација, мислим да је логика и више него јасна, израчунају се суме у петљи и користе се формуле за рачунање параметара a и b , те се ти параметри смјештају у фајл.

Што се тиче AVX инструкцијског скупа, омогућио сам читавање 4 вриједности одједном (из оба низа), те су и неопходна множења и сабирање рађена помоћу одговарајућих AVX инструкција, чиме се постигло убрзање.

OpenMP је кориштен на једноставан начин, јер ми је reduction омогућио да искористим више threadova на поуздан начин јер су операције thread safe и нисам имао страха да ће доћи до нарушавања резултата.

Комбинација OpenMP и AVX инструкцијског скупа је опет користила reduction, али сада је постојао проблем са threadovima, те сам због тога онемогућио да један thread узме податке из низа (низова) другог threada тако што је сваки thread имао свој опсег који је покривао у петљи, те су остварена значајна убрзања у односу на програм без оптимизација, али и остала два програма.

Просјечне вриједности и варијансе времена извршавања програма

Треба рећи да у првој табели имамо 2000 покретања фајла који има 4 милиона елемената:

Редни број покретања	Без оптимизација	AVX	OpenMP	MIX варијанта
1.	58,474s	52,521s	50,842s	46,576s
2.	57,564s	52,566s	51,008s	46,551s
3.	57,503s	52,338s	50,783s	46,498s
ПРОСЈЕК	57.847s	52.475s	50.877s	46.541s
ВАРИЈАНСА	0.295777	0.014583	0.013610333	0.0015863333

Table 1

У другој табели посматрамо 500 покретања фајла који има 20 милиона елемената:

Редни број покретања	Без оптимизација	AVX	OpenMP	MIX варијанта
1.	1m 3.756s	57,727s	51,847s	45,633s
2.	1m 3,485s	57,534s	51,527s	45,795s
3.	1m 4,127s	57,478s	51,584s	45,882s
ПРОСЈЕК	63.789s	57.579s	51.652s	45.77s
ВАРИЈАНСА	0.10387433	0.017064333	0.029136333	0.015969

Table 2

У трећој табели посматрамо 400 покретања фајла који има 32 милиона елемената:

Редни број покретања	Без оптимизација	AVX	OpenMP	MIX варијанта
1.	1m 24,828s	1m 16,564s	1m 9,408s	1m 8,405s
2.	1m 22,623s	1m 14,568s	1m 8,210s	59,959s
3.	1m 23,080s	1m 15,329s	1m 8,088s	1m 0,665s
ПРОСЈЕК	83.510s	75.487s	68.568s	63.009s
ВАРИЈАНСА	1.3543963	1.014727	0.53208133	21.956825

Table 3

Графички приказ резултата

Вриједности на графику на у-оси су у секундама.

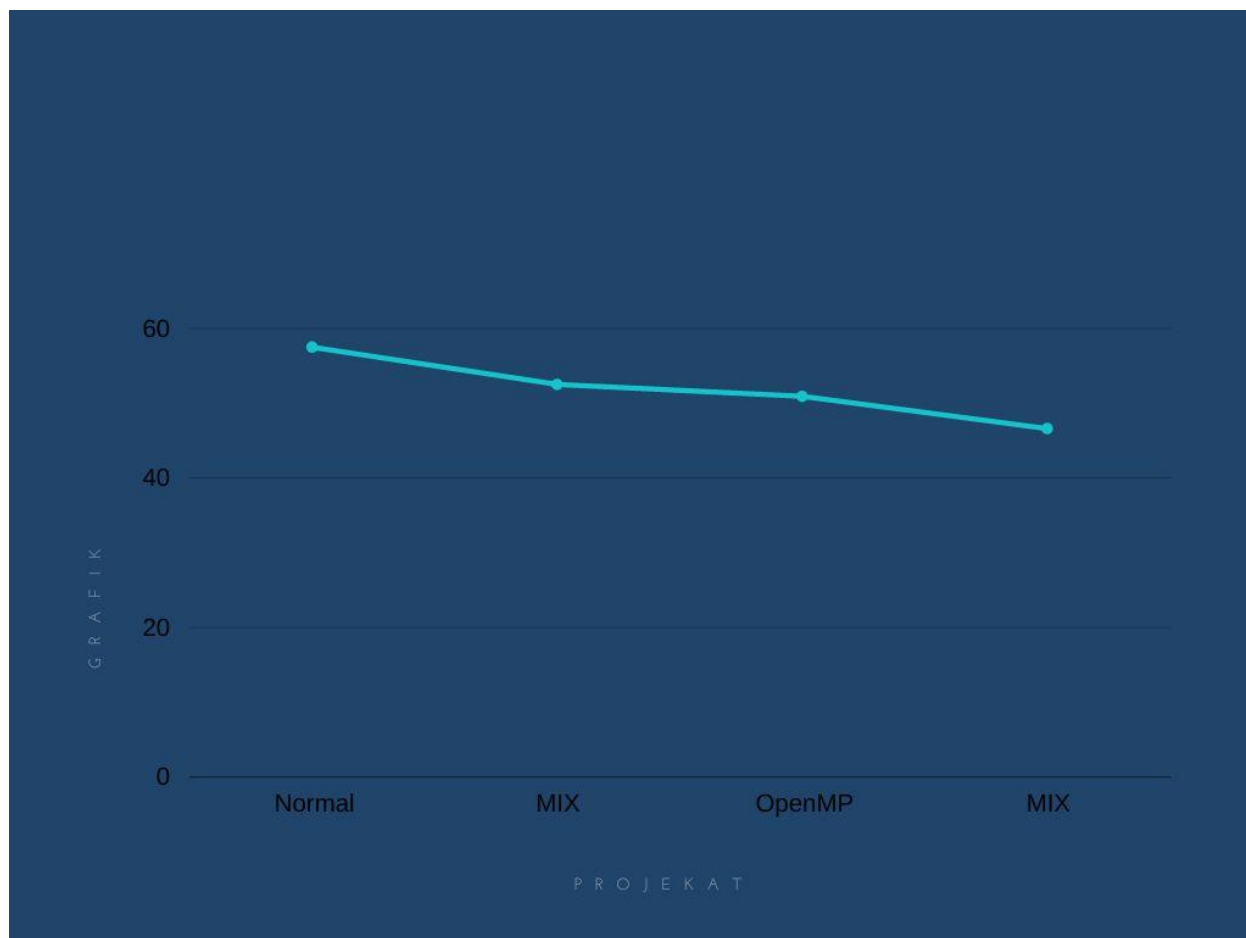


График 1 - Просјечне вриједности из table 1

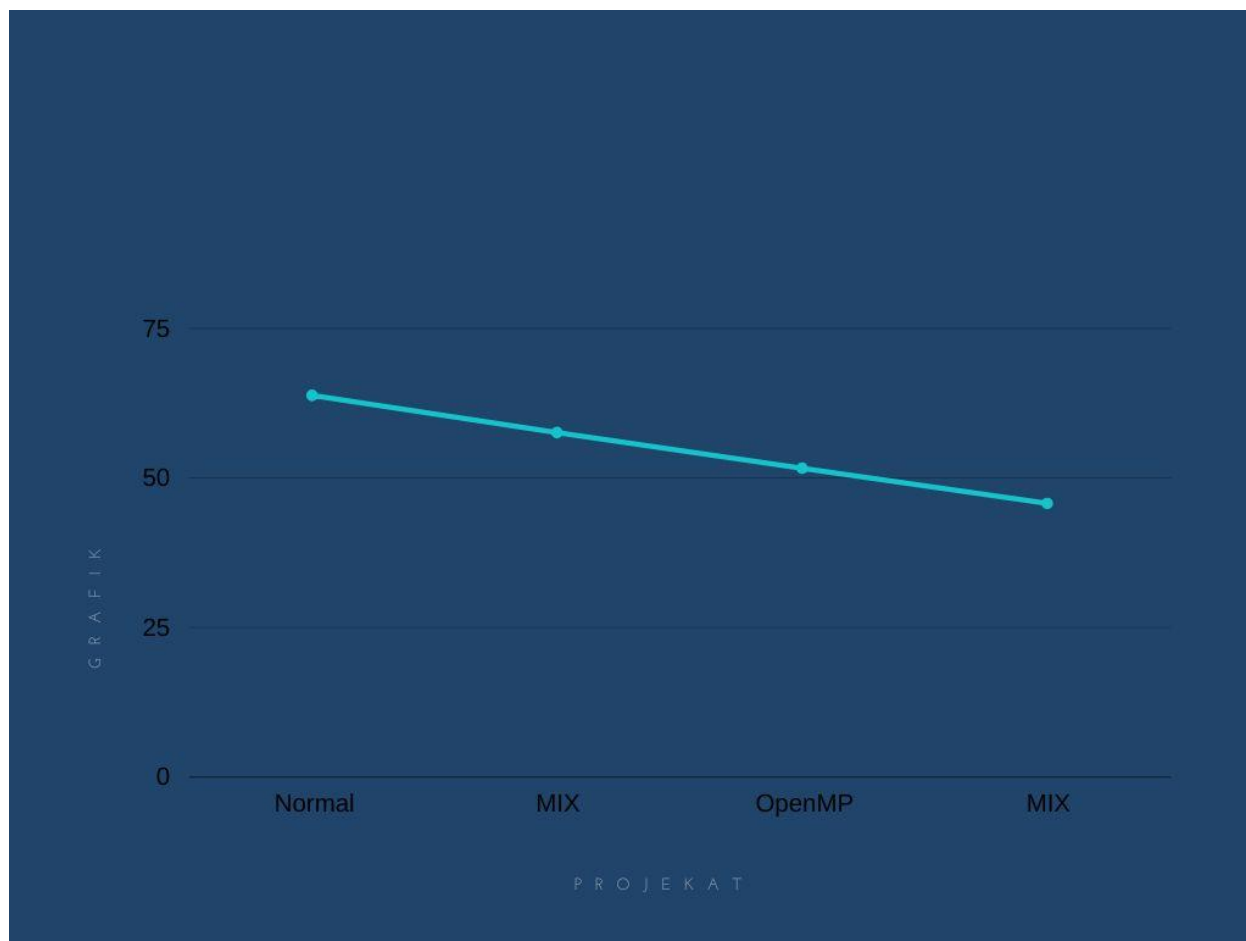


График 2 - Просјечне вриједности из table 2

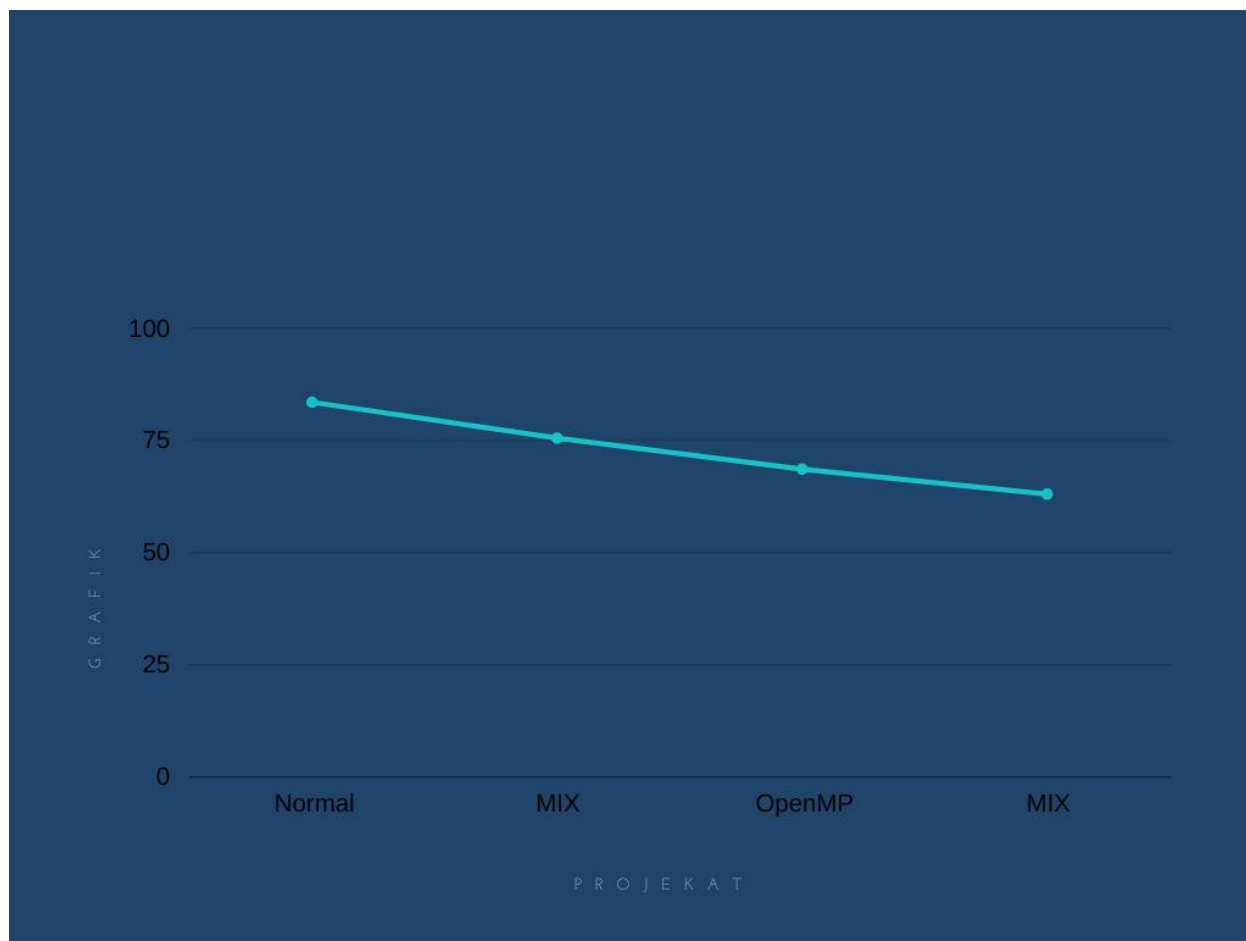


График 3 - Просјечне вриједности из table 3

Закључак

С обзиром да се ради о линеарној регресији, која нема превише комплексних операција осим рачунања сума, резултати су и више него задовољавајући.

Видимо и конзистентност брзина извршавања што показују табеле, али и резултати су конзистентни уз минимална одступања.

При извршавању програма са ОЗ оптимизацијом сам примијетио да су перформансе обрнуте, односно да програм без оптимизација има најбрже вријеме извршавања (бар на лаптопу који сам ја користио и на примјерима које сам тестирао, не мора значити да важи за сваки примјер).

Што је већи број елемената, више се показује и снага оптимизације што је логичан закључак.

При неким тестирањима (нпр. са 1 000 000 елемената), OpenMP ми је показивао јако лоше резултате за шта нисам могао наћи одговор, али на тестирањима са 2, 4 и 16 милиона елемената (у једном низу) OpenMP се показао и више него добар.