МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет информационных технологий

Кафедра параллельных вычислений

ОТЧЕТ

О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ 1

студента Бородина Артёма Максимовича 3 курса, группы 19205

Новосибирск, 2022

Цель

Научиться разрабатывать простые программы численного моделирования, применять базовые средства оптимизации программ, выполнять оценку и анализ производительности программ, пользоваться средствами профилирования.

Задание

Решение волнового уравнения методом конечных объёмов. В качестве типов данных нужно использовать double.

Алгоритм моделирует распространение волны в двумерной области, инициированной импульсом из заданного узла сетки. В начальный момент времени значения искомой функции U на сетке инициализируются нулями. На каждом шаге моделирования значения искомой функции пересчитываются по заданной формуле.

Входные данные: **Nx=Ny=8000, Nt=100**.

Процессор: Intel(R) Xeon(R) Gold 6128 CPU @ 3.40GHz

Ход работы

Времена замеров

Оптимизации	Время, сек
Без оптимизации	445,7
O1	90,7
O2	86,4
O3	77,6
Оптимизация операций	52,1
Просчет индексов	48,3

	Неправильные переходы, %	Кэш промахи 1го уровня, %	Кэш промахи 3го уровня, %
Быстрый вариант	0	0,119	0
Вариант без оптимизаций	0	0,146	0,000793

Характеристика самого быстрого варианта программы

Elapsed Time: 70.186s

CPU Time:	**62.832s**	
Effective Time:	62.832s	
Spin Time:	0s	
Overhead Time:	0s	
Instructions Retired:	734,757,000,000	
Microarchitecture Usage:	**70.2%**	**of Pipeline Slots**
CPI Rate:	0.315	
Total Thread Count:	1	
Paused Time:	0s	

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions

Function	Module	CPU Time	% of CPU Time
WaveSim::Run	wavesim	56.147s	89.4%
[Outside any known module]	[Unknown]	3.132s	5.0%
WaveSim::phi	wavesim	2.090s	3.3%
std::ostream::write	libstdc++.so.6.0.28	1.213s	1.9%
WaveSim::init	wavesim	0.150s	0.2%
[Others]	N/A*	0.100s	0.2%

*N/A is applied to non-summable metrics.

Характеристика программы без оптимизаций

Elapsed Time: 481.799s

CPU Time:	**471.064s**	
Instructions Retired:	4,462,211,000,000	
Microarchitecture Usage:	**60.3%**	**of Pipeline Slots**
Total Thread Count:	1	
Paused Time:	0.084s	

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions

Function	Module	CPU Time	% of CPU Time
WaveSim::Run	wavesim	425.357s	90.3%
WaveSim::phi	wavesim	19.360s	4.1%
WaveParams::getSy	wavesim	10.069s	2.1%
[Outside any known module]	[Unknown]	7.513s	1.6%
WaveParams::getSx	wavesim	6.681s	1.4%
[Others]	N/A*	2.085s	0.4%

*N/A is applied to non-summable metrics.

Аннотированный листинг самого быстрого варианта

S ▲	Source	CPU Ti…	CPU Time…	Instru…	Instructions Retir…
51	for (n = 0; n < Nt; ++n) {				
52					
53	//time_point<high_resolution_clock> stepStart = high_resoluti				
54	for (y = 1; y < Ny - 1; ++y) {				
55	for (x = 1; x < Nx - 1; ++x) {	1.970s	1.970s	3.8%	27,846,000,000
56	int pos = y * Nx + x;	3.027s	3.027s	1.1%	8,143,000,000
57	int prevPos = (y - 1) * Nx + x;	1.874s	1.874s	3.7%	27,438,000,000
58					
59	double currentPos = UCurrent[pos];	2.872s	2.872s	5.4%	39,797,000,000
60	double pPos = P[pos];	3.373s	3.373s	6.8%	50,269,000,000
61					
62	double avgx =	0.125s	0.125s	0.2%	1,139,000,000
63	((UCurrent[pos + 1] - currentPos) *	1.544s	1.544s	2.9%	21,386,000,000
64	(P[prevPos] + pPos) +	3.478s	3.478s	6.1%	44,608,000,000
65	(UCurrent[pos - 1] - currentPos) *	3.187s	3.187s	5.8%	42,891,000,000
66	(P[prevPos - 1] + P[pos - 1])) * hxsqr;	3.393s	3.393s	6.2%	45,747,000,000
67	double avgy =	1.734s	1.734s	2.0%	14,365,000,000
68	((UCurrent[(y + 1) * Nx + x] - currentPos) *	3.799s	3.799s	6.6%	48,263,000,000
69	(P[pos - 1] + pPos) +	3.102s	3.102s	6.0%	44,098,000,000
70	(UCurrent[prevPos] - currentPos) *	3.208s	3.208s	5.5%	40,086,000,000
71	(P[prevPos - 1] + P[prevPos])) * hysqr;	0.105s	0.105s	0.2%	1,394,000,000
72	double result = 2 * currentPos - UPrev[pos] + tausqr	9.227s	9.227s	14.4%	105,876,000,000
73	UNext[pos] = result;	3.618s	3.618s	7.1%	51,850,000,000
74	if (result > UMax) {	6.510s	6.510s	11.2%	82,416,000,000
75	UMax = std::abs(result);				
76	}				
77	}				
78	}				

Аннотированный листинг программы без оптимизаций

S ▲	Source	C…	CPU …	Instr…	Instructions Retire…
45	for (n = 0; n < Nt; ++n) {				
46					
47	//time_point<high_resolution_clock> stepStart = high_resolution_clock::now();				
48	for (y = 1; y < Ny - 1; ++y) {	0.020s	0.020s	0.0%	34,000,000
49	for (x = 1; x < Nx - 1; ++x) {	11.031s	11.031s	1.5%	67,660,000,000
50	double avgx =	38.159s	38.159s	10.5%	469,013,000,000
51	((UCurrent[y * Nx + x + 1] - UCurrent[y * Nx + x]) * (P[(y - 1) *	82.584s	82.584s	14.5%	649,094,000,000
52	(UCurrent[y * Nx + x - 1] - UCurrent[y * Nx + x]) *	37.358s	37.358s	8.4%	372,657,000,000
53	(P[(y - 1) * Nx + x - 1] + P[y * Nx + x - 1])) / (2 * hx * hx);	37.874s	37.874s	7.2%	321,640,000,000
54	double avgy =	21.515s	21.515s	6.9%	306,272,000,000
55	((UCurrent[(y + 1) * Nx + x] - UCurrent[y * Nx + x]) * (P[y * Nx	53.210s	53.210s	2.8%	125,647,000,000
56	(UCurrent[(y - 1) * Nx + x] - UCurrent[y * Nx + x]) *	23.234s	23.234s	4.9%	216,920,000,000
57	(P[(y - 1) * Nx + x - 1] + P[(y - 1) * Nx + x])) / (2 * hy * hy)	32.862s	32.862s	10.9%	487,050,000,000
58	double result = 2 * UCurrent[y * Nx + x] - UPrev[y * Nx + x] + tau * tau	64.286s	64.286s	19.7%	878,951,000,000
59	UNext[y * Nx + x] = result;	13.196s	13.196s	3.7%	164,458,000,000
60	if (result > UMax) {	10.029s	10.029s	2.2%	96,781,000,000
61	UMax = std::abs(result);				
62	}				
63	}				
64	}				

Граф вызовов самого быстрого варианта программы

Function Stack	C…	CPU …	Instru…	Micr…	Instructions Retire…
Total	62.832s	0s	100.…	70.2%	0
_start	62.101s	0s	99.7%	70.5%	0
__libc_start_main	62.101s	0s	99.7%	70.5%	0
main	62.101s	0s	99.7%	70.5%	0
WaveSim::Run	60.757s	56.147s	97.2%	70.4%	697,629,000,000
WaveSim::phi	2.110s	2.090s	1.4%	69.4%	9,911,000,000
[Outside any known modul	1.524s	0.246s	0.7%	24.5%	34,000,000
WaveSim::init	0.942s	0.150s	0.2%	11.4%	289,000,000
std::abs	0.035s	0s	0.0%	0.0%	0
WaveSim::saveBinary	1.313s	0.085s	2.5%	76.8%	1,598,000,000
[Outside any known module]	0.030s	0.030s	0.0%		34,000,000
[Skipped stack frame(s)]	0.682s	0s	0.3%	27.8%	0
[Unknown stack frame(s)]	0.050s	0s	0.0%	100.0%	0

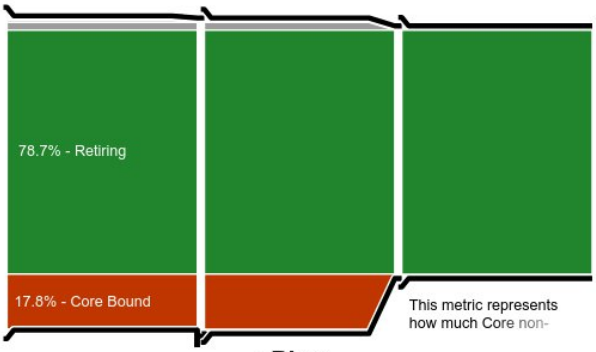
Граф вызовов программы без оптимиизаций

Function Stack	CPU…	CPU …	Instru…	Micr…	Instructions Retire…
Total	471.064s	0s	100.…	60.3%	0
_start	470.047s	0s	99.9%	60.3%	0
__libc_start_main	470.047s	0s	99.9%	60.3%	0
main	470.047s	0s	99.9%	60.3%	0
WaveSim::Run	469.696s	425.357s	99.9%	60.4%	4,156,177,000,…
WaveSim::phi	36.405s	19.360s	5.8%	50.9%	168,640,000,000
WaveParams::getSy	10.159s	10.069s	1.7%	27.2%	74,783,000,000
WaveParams::getSx	6.721s	6.681s	0.3%	70.5%	15,232,000,000
[Outside any known module]	0.115s	0s	0.0%	19.1%	0
func@0x2fca	0.050s	0s	0.0%		0
[Outside any known module]	5.533s	0.251s	0.4%	20.1%	85,000,000
WaveSim::init	1.318s	0.611s	0.2%	35.9%	6,494,000,000
WaveSim::saveBinary	1.083s	0.251s	0.4%	82.0%	2,805,000,000
[Outside any known module]	0.351s	0.351s	0.0%	6.6%	153,000,000
[Skipped stack frame(s)]	0.591s	0s	0.0%	26.7%	0
__libc_start_main	0.366s	0s	0.1%	100.0%	0
[Unknown stack frame(s)]	0.050s	0s	0.0%	88.2%	0
[Outside any known module]	0.010s	0.010s	0.0%	23.5%	0

Ограничения самого быстрого варианта программы

Elapsed Time: 70.301s

Clockticks:	230,554,000,000	
Instructions Retired:	734,026,000,000	
CPI Rate:	0.314	
Retiring:	78.7%	of Pipeline Slots
Light Operations:	78.2%	of Pipeline Slots
FP Arithmetic:	36.7%	of uOps
FP x87:	0.0%	of uOps
FP Scalar:	36.7%	of uOps
FP Vector:	0.0%	of uOps
Memory Operations:	24.9%	of Pipeline Slots
Fused Instructions:	2.8%	of Pipeline Slots
Non Fused Branches:	1.6%	of Pipeline Slots
Nop Instructions:	0.1%	of Pipeline Slots
Other:	12.0%	of Pipeline Slots
Heavy Operations:	0.5%	of Pipeline Slots
Front-End Bound:	0.7%	of Pipeline Slots
Bad Speculation:	0.2%	of Pipeline Slots
Back-End Bound:	20.4%	of Pipeline Slots
Memory Bound:	2.5%	of Pipeline Slots
Core Bound:	17.8%	of Pipeline Slots
Divider:	0.0%	of Clockticks
Port Utilization:	25.6%	of Clockticks
Cycles of 0 Ports Utilized:	2.6%	of Clockticks
Cycles of 1 Port Utilized:	4.1%	of Clockticks
Cycles of 2 Ports Utilized:	10.8%	of Clockticks
Cycles of 3+ Ports Utilized:	33.2%	of Clockticks
Vector Capacity Usage (FPU):	12.5%	
Average CPU Frequency:	3.7 GHz	
Total Thread Count:	1	
Paused Time:	0s	



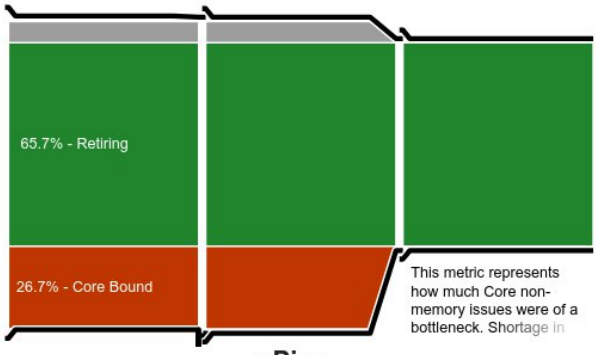
µPipe

This diagram represents inefficiencies in CPU usage. Treat it as a pipe with an output flow equal to the "pipe efficiency" ratio: (Actual Instructions Retired)/(Maximum Possible Instruction Retired). If there are pipeline stalls decreasing the pipe efficiency, the pipe shape gets more narrow.

Ограничения программы без оптимизаций

Elapsed Time: 483.895s

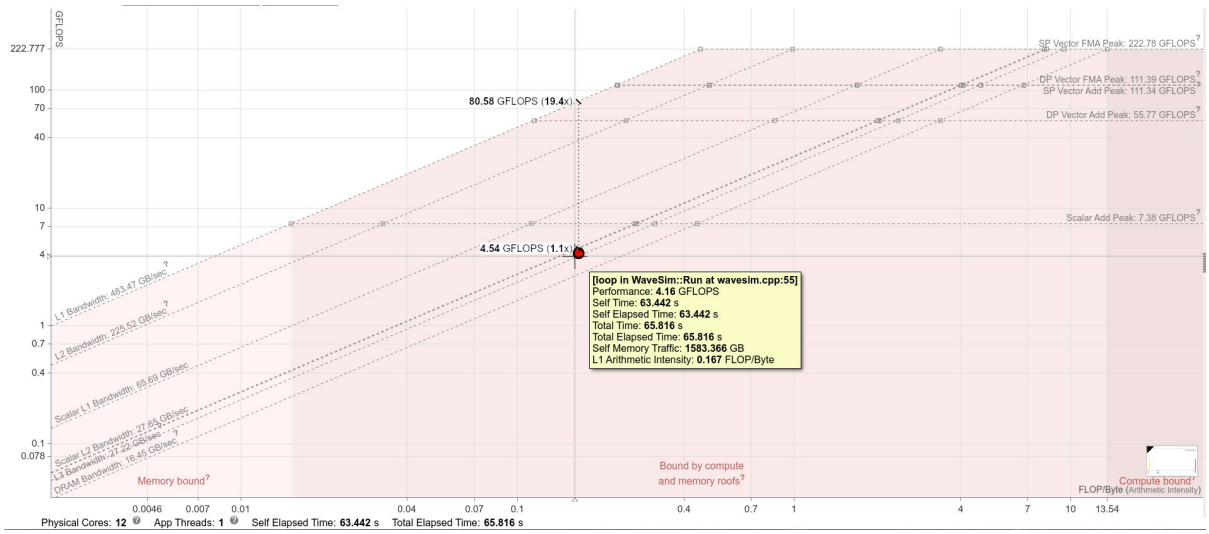
Clockticks:	1,747,583,000,000	
Instructions Retired:	4,474,587,000,000	
CPI Rate:	0.391	
Retiring:	65.7%	of Pipeline Slots
Front-End Bound:	0.4%	of Pipeline Slots
Bad Speculation:	0.2%	of Pipeline Slots
Back-End Bound:	33.6%	of Pipeline Slots
Memory Bound:	6.9%	of Pipeline Slots
L1 Bound:	4.8%	of Clockticks
L2 Bound:	0.8%	of Clockticks
L3 Bound:	0.3%	of Clockticks
DRAM Bound:	1.0%	of Clockticks
Store Bound:	0.1%	of Clockticks
Core Bound:	26.7%	of Pipeline Slots
Divider:	9.6%	of Clockticks
Port Utilization:	26.6%	of Clockticks
Cycles of 0 Ports Utilized:	3.6%	of Clockticks
Cycles of 1 Port Utilized:	5.3%	of Clockticks
Cycles of 2 Ports Utilized:	12.0%	of Clockticks
Cycles of 3+ Ports Utilized:	29.1%	of Clockticks
Vector Capacity Usage (FPU):	12.5%	
Average CPU Frequency:	3.7 GHz	
Total Thread Count:	1	
Paused Time:	0s	



µPipe

This diagram represents inefficiencies in CPU usage. Treat it as a pipe with an output flow equal to the "pipe efficiency" ratio: (Actual Instructions Retired)/(Maximum Possible Instruction Retired). If there are pipeline stalls decreasing the pipe efficiency, the pipe shape gets more narrow.

Roofline самого быстрого варианта программы



Roofline программы без оптимизаций