Опыт ИВМиМГ СО РАН в применении вычислительных технологий Huawei

Марченко Михаил Александрович,
Городничев Максим Александрович
ИВМиМГ СО РАН

Совместный семинар ЦКП ССКЦ и НГУ
"Высокопроизводительные вычисления"

24 ноября 2022 г.

План

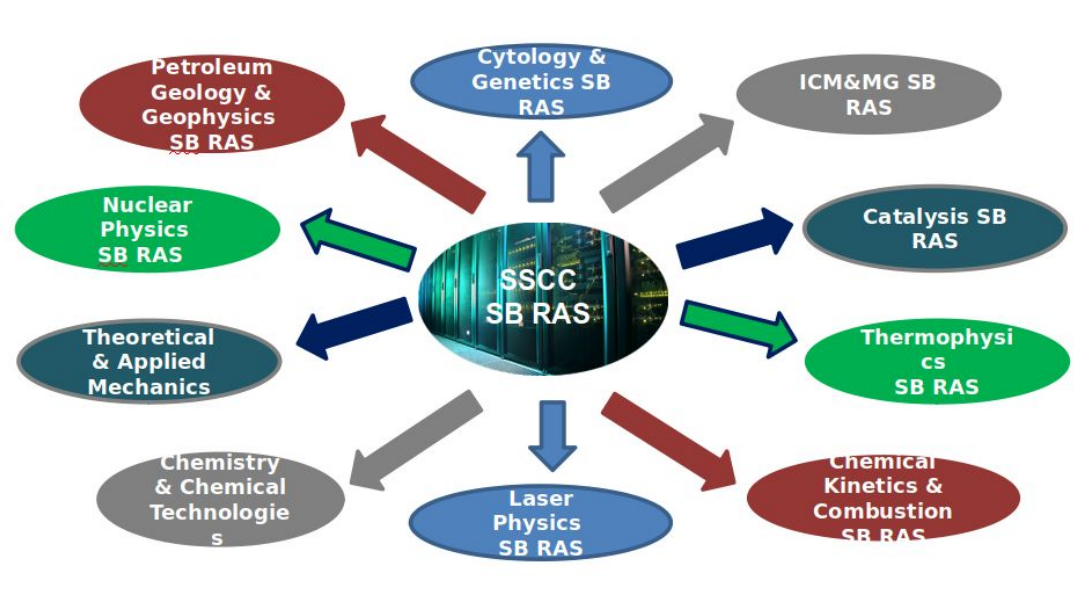
- Обзор совместных проектов ИВМиМГ СО РАН - Huawei
- Процессор Kunpeng
- Портирование PARMONC на кластер на Kunpeng
- Нейропроцессоры Ascend
- Обзор ML фреймворка MindSpore
- Physics informed neural networks: пример приложения

ИВМиМГ

Обзор совместных проектов ИВМиМГ СО РАН - Huawei

ИВМиМГ

Центр коллективного пользования
Сибирский суперкомпьютерный центр СО РАН



- Процессорная база Intel и Хуавей
- Более 200 пользователей из 24 организации, больше 200 пользователей
- С использованием оборудования ССКЦ в год выполняется более 100 НИР на общую сумму более 700 млн. рублей
- Текущая производительность
- 0,2 Петафлопс на процессорах Intel
- и 2.5 Петафлопс в FP16 на сервере Huawei ATLAS 9000

ИВМиМГ

Разработка оптимизированного под вычислительные решения Хуавей программного обеспечения

Портирование суперкомпьютерных приложений ИВМиМГ и других пользователей ССКЦ на ARM процессоры Kunpeng:

- Библиотека PARMONC для распараллеливания программ статистического моделирования
- Решатели для систем дифференциальных и интегро-дифференциальных уравнений

Портирование приложений ИИ и МО на процессоры Ascend:

- Решение задач сейсмики (с ИНГГ СО РАН)
- Определение структур на детекторах С-Тау фабрики

(с ИЯФ СО РАН)

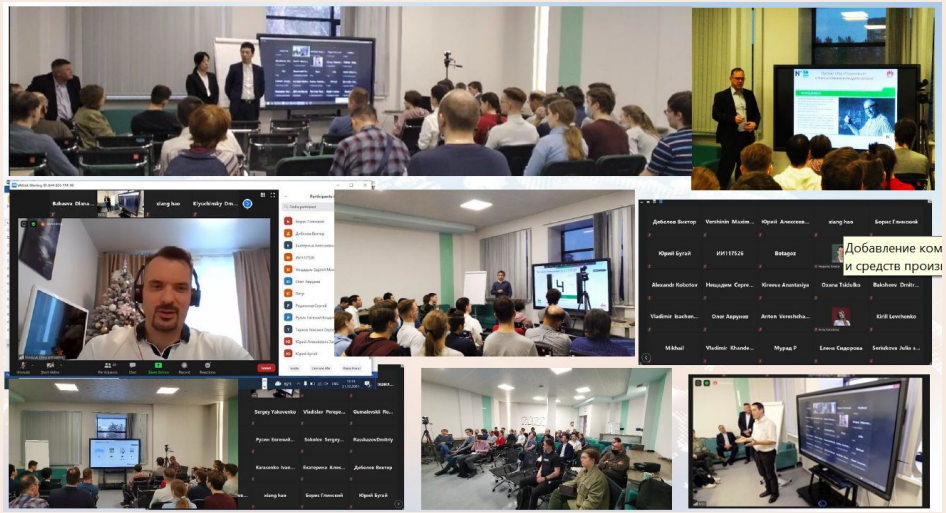
- Задачи ряда научно-образовательных организаций и бизнес-заказчиков

Инфраструктурные и образовательные проекты:

- Сотрудничество в создание экосистемы Kunpeng
- Курсы в НГУ по решениям Хуавей для методов ИИ и МО (обучение и повышение квалификации)
- Научно-практический семинар СО РАН - Хуавей

ИВМиМГ

Как мы в ИВМиМГ работаем со студентами



Преподаем студентам:

- Теория численных методов для решения прикладных задач в области физики, химии, биологии, геологии, общественных и гуманитарных наук
- Параллельное программирование
- Архитектура вычислительных систем и сетей
- Методы искусственного интеллекта и теория нейросетей
- Методы оптимизации сетей
- Практическая работа на суперкомпьютерах различных производителей (Intel, Nvidia, Huawei)

Вместе работаем:

- Выполнение грантов и проектов Минобрнауки и научных фондов
- Анализ и решение задач заказчиков с использованием суперкомпьютеров
- Формирование студенческих команд под управлением квалифицированных специалистов для решения задач заказчиков

ИВМиМГ


Joint Thesis Program 2022
scholarship program
for Master and PhD students

ИВМиМГ

Joint Thesis Program Overview

Goal

To help **grow** PhD students with actual research experience and motivation to graduate

Format: a scholarship program with topics provision and consultant support

Target audience: Master and PhD entrants, 1st–2nd year PhD students, who already have programming projects experience

How to get a scholarship:

1. Win application contest on Huawei topics
2. Work on relevant topic with the supervisor and be nominated by institute

ИВМиМГ

Joint Thesis Program Responsibilities

ICMMG

- Help to formulate research topics based on Huawei tasks
- Provide supervisors and reviewers
- Manage Master and PhD students to ensure a successful graduation
- Organizes seminars on the progress of research at least once a month

Huawei

- Provide challenging research tasks and consultants for them
- Provide financial support for students and supervisors
- Evaluate the results of the research projects
- Provide experts to take part in seminars on the progress of research

ИВМиМГ

Joint Thesis Program Tracks

Master students		PhD students				
1	2	1	2	3	4	Grad
Diploma project		PhD dissertation				
Getting Experience		**Diving Into Research**		**Diving Into Research**		

Two-year programs:

1. **Getting Experience**

   For Master students:

   - More practical topics
   - ~20 hours a week expected
   - Oriented on getting R&D experience

2. **Diving Into Research**

   For PhD students:

   - Challenging research topics
   - ~30 hours a week expected
   - Oriented on getting a PhD degree in 2-3 years
   - Recommended to teach and/or supervise younger students

ИВМиМГ

Joint Thesis Program Mentorship

Role	Responsibilities	Interaction
Supervisor **ICMMG**	• To track whether a student makes progress and to advise on a research methodology • To help a student to meet the formal criteria: publications, dissertation text, etc.	Regular meetings with student expected (once in a week or two)
Consultant **Huawei**	• To advise a student and a supervisor on a research content • To evaluate research results	Consultations on request, participation in seminars once a month expected

ИВМиМГ

Joint Thesis Program Benefits

Building an ecosystem where PhD students will grow and work on actual topics:

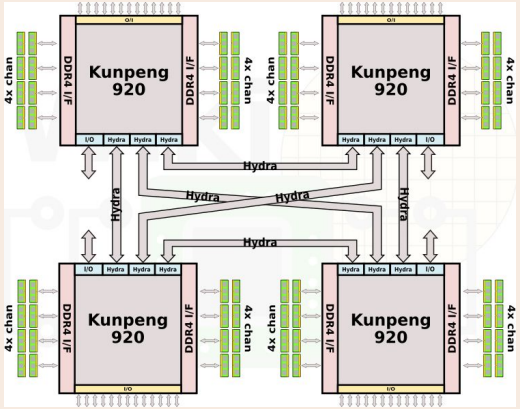
- Attract students interested in research on a serious basis
- Exchange of experience between academic and industrial mentors
- Increase the number of young specialists with confirmed higher qualification
- Increase the number of young teachers and academic supervisors
- Make a basis for scientific cooperation projects

ИВМиМГ

Процессор Kunpeng

ИВМиМГ

Процессор Kunpeng 920-6426



https://fuse.wikichip.org/kunpeng-920-4smp/

Designer: HiSilicon, ARM Holdings
Manufacturer: TSMC
Market Server
Introduction: September, 2018 (announced), January 7, 2019 (launched)
Frequency 2,600 MHz
ISA ARMv8.2 (ARM)
Microarchitecture TaiShan v110
Word Size 64 bit
Cores 64
Threads 64
Max Memory 2 TiB

Тебибайт

Memory Bandwidth: Single 23.84 GiB/s, Double 47.68 GiB/s, Quad 95.37 GiB/s, Octa 190.7 GiB/s
Supported ARM Extensions & Processor Features: NEON Advanced SIMD extension, CRC-32 checksum Extension, Crypto Cryptographic Extension, FP16 ARMv8.2-A half-precision floating-point extension

ИВМиМГ

Core Peak performance (x48 or x64 cores)

Data types		Operations				
		add	mul	div	sqrt	fma
Scalar data types	16-bit float	5.2*	5.2	0.43	0.24	10.4
	32-bit float	5.2	5.2	0.43	0.29	-
	64-bit float	5.2	5.2	0.43	0.29	-
	128-bit float	0.08	0.11	0.14	0.11	-
128-bit vector data types	8 × 16-bit float	41.6	41.6	0.5	0.25	83.2
	4 × 32-bit float	20.8	20.8	0.87	0.58	41.6
	2 × 64-bit float	5.2	5.2	0.43	0.29	10.4

* Billions of floating point operation per second (GFLOPS)

fma - fused multiply-add operations

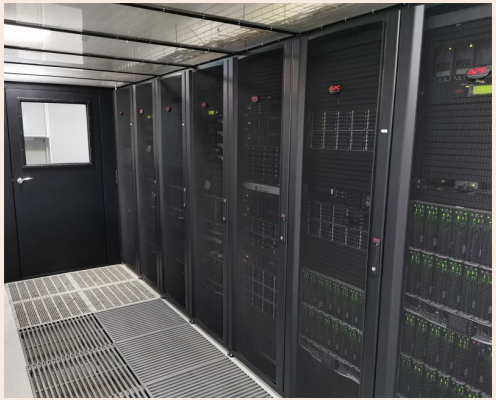
Broadwell
CPU (2 x) Intel Xeon E5-2697A v4 2.6 Ггц
~42 GFLOPS @ DP FMA

ИВМиМГ

Installation of a Kunpeng-based cluster at Siberian Supercomputer Center (ICM&MG SB RAS)

Hardware:

- head node: TaiShan 200 2280 server: 2*48 cores 256 gb ram
- 3 worker nodes: TaiShan 200 2280 servers: each has 2*64 cores 1024 gb ram
- Mellanox Infiniband Switch
- 24GE+4x10GE HUAWEI Switch
- MikroTik router



Installed Software:

- OpenEuler operating system
- Development software including:
  - compilers: GNU Compilers,
  - MPI: OpenMPI,
  - version control: git,
  - math libraries: HML
- Cluster management and monitoring software: SLURM, OpenLDAP, Munge

ИВМиМГ

Оборудование: серверы TaiShan

- Модель 1280: сервер высокой плотности, 1U, 2 x CPU Kunpeng 920 (64 ядра) 2,6 ГГц, 10 дисков 2,5 дюйма SAS/SATA/SSD или 8 дисков 2,5 дюйма NVMe SSD.
- 2180: 2U, 1 x Kunpeng 920, 14 дисков SAS/SATA/SSD.
- 2280E: 5G, 2U, 2 x CPU Kunpeng 920: для периферийных вычислений, 5g.
- 2480: 2U, 4 x Kunpeng 920, 24 SSD-дисков NVMe, 32 x DDR4, до 4 оптических портов 10GE/25GE или 2 оптических портов 100GE, PCIe: высокопроизводительные вычисления, работа с базами данных и облачные вычисления
- 2280 V2: 2U, 2 x Kunpeng 920, 28 NVMe.
- 5290: 4U, 2 x Kunpeng 920, дисковое хранилище до 1 PB.
- 5280 V2: 4U, 2 x Kunpeng 920, дисковое хранилище до 560 ТБ.

ИВМиМГ

Кстати: rand() и многопоточность?

rand() - not thread safe

rand_r() - thread safe but questionable quality

Porting PARMONC Library and ELSHOW Application to Kunpeng-920-based system

Mikhail Marchenko
Igor Marinin
Vadim Chesnokov,
Maxim Gorodnichev
Sergey Kireev
Alexander Russkov

Objectives

1. Installation of a Kunpeng-based cluster at Siberian Supercomputer Center
2. Porting of
   a. the PARMONC Library: a library for development of programs that implement massively parallel stochastic simulation on distributed memory systems
   b. the ELSHOW application: simulation of electron avalanches in gases

ИВМиМГ

PARMONC: a framework for parallel implementation of large scale Monte Carlo simulations

Publications on PARMONC framework and its applications

1. Marchenko M. (2011) PARMONC - A Software Library for Massively Parallel Stochastic Simulation. In: Malyshkin V. (eds) Parallel Computing Technologies. PaCT 2011. Lecture Notes in Computer Science, vol 6873. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-23178-0_27
2. Rogasinsky S. V., Marchenko M. A. Stochastic simulation of electron avalanches on supercomputers //AIP Conference Proceedings. – American Institute of Physics, 2014. – Т. 1628. – №. 1. – С. 1116-1123.
3. M. A. Marchenko “Efficient computational approaches for parallel stochastic simulation on supercomputers”. 2014 Nova Science Publishers, Inc.
4. M. A. Marchenko “Efficient use of manycore accelerators for stochastic simulation of electron avalanches on supercomputers”, DOI: http://dx.doi.org/10.14529/cmse130406 (published in Russian: M. A. Марченко, “Эффективное использование многоядерных сопроцессоров при суперкомпьютерном статистическом моделировании электронных лавин”, *Вестн. ЮУрГУ. Сер. Выч. матем. информ.*, **2**:4 (2013), 80–93)
5. A. V. Edelev, M. A. Marchenko, and O. Yu. Basharina. Identification of the energy system critical elements using the PARMONC library // 2021 J. Phys.: Conf. Ser. 1715 012064, doi:10.1088/1742-6596/1715/1/012064
6. V. I. Belousov. Parallel implementation of electron transfers with Monte Carlo method using the BRAND software package and the PARMONC library, https://www.elibrary.ru/item.asp?id=18899748 (published in Russian: Белоусов В. И. Распараллеливание расчёта переноса электронов методом Монте-Карло при использовании комплекса BRAND и библиотеки программ PARMONC // Вопросы атомной науки и техники. Серия: Физика ядерных реакторов. 2013. №1. С. 11–17.)

…..

26. Lotova G. Z. Supercomputer simulation of electron avalanches in gases with calculation of diffusive characteristics //9th International Workshop on Simulation. – 2018. – Т. 66. – С. 67.
27. Phap, V. M., Van Binh, D., Nam, N. H., Edelev, A. V., & Marchenko, M. A. (2020). Analysis of Economic-Technical Potential of Renewable Power Sources for the Establishment of National Renewable Energy Center in Ninh Thuan Province, Vietnam. In E3S Web of Conferences (Vol. 209, p. 06022). EDP Sciences.
28. Ivanov A. Analysis of the Influence of Random Noise upon the Properties of Stochastic Oscillators by the Monte Carlo Method Using Supercomputer //Параллельные вычислительные технологии (ПаВТ'2018). – 2018. – С. 112-125.
29. Pertsev, N.V., Loginov, K.K. & Topchii, V.A. Analysis of a Stage-Dependent Epidemic Model Based on a Non-Markov Random Process. J. Appl. Ind. Math. 14, 566–580 (2020). https://doi.org/10.1134/S1990478920030151
30. Nonlinear effects in the dynamics of HIV-1 infection predicted by mathematical model with multiple delays,Discrete & Continuous Dynamical Systems - S,13,9,2365,2384,2019-11-22,Nikolay Pertsev,Konstantin Loginov,Gennady Bocharov,1937-1632_2020_9_2365,Mathematical model,integro-differential equations,HIV infection,stability and dynamics, doi: 10.3934/dcdss.2020141

ИВМиМГ

Current PARMONC users

- Institute of Computational Mathematics and Mathematical Geophysics SB RAS

- Budker Institute of Nuclear Physics SB RAS

- Institute of Mathematics SB RAS

- Obninsk Institute for Nuclear Power Engineering

- The Institute for System Dynamics and Control Theory SB RAS

Applications

- Development of electron avalanches (ELSHOW)
- Vulnerability analysis of gas transmission systems
- Infection development simulation
- Epidemy development simulation
- Weather monitoring: simulation of cloud remote sensing

etc.

Identification of the energy system critical elements using the PARMONC library

A V Edelev[1], M A Marchenko[2,3] and O Yu Basharina[4]
[1]Melentiev Energy Systems Institute SB RAS, Lermontov St., 130, Irkutsk, Russia, 664033
[2]Institute of Computational Mathematics and Mathematical Geophysics SB RAS, 6, Ac. Lavrentieva ave., Novosibirsk, Russia, 630090
[3]Novosibirsk State University, Novosibirsk-90, 2, Russia, 630090
[4]Matrosov Institute for System Dynamics and Control Theory SB RAS, Lermontov St., 134, Irkutsk, Russia, 664033

E-mail: flower@isem.irk.ru

Abstract. In this paper, we presented a Monte Carlo-based approach for vulnerability analyse of energy systems. For high-performance Monte Carlo simulation the PARMONC software

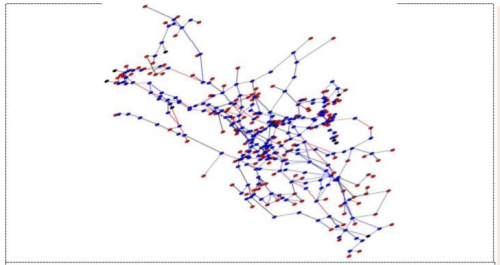


Figure 2. The topology of the Unified Gas Supply System of Russia.

NONLINEAR EFFECTS IN THE DYNAMICS OF HIV-1 INFECTION PREDICTED BY MATHEMATICAL MODEL WITH MULTIPLE DELAYS

Nikolay Pertsev*
Sobolev Institute of Mathematics
Siberian Branch of the Russian Academy of Sciences, Omsk Branch
Omsk, 644043, Russian Federation
Marchuk Institute of Numerical Mathematics
Russian Academy of Sciences
Moscow, 119333, Russian Federation
Konstantin Loginov
Sobolev Institute of Mathematics
Siberian Branch of the Russian Academy of Sciences, Omsk Branch
Omsk, 644043, Russian Federation
Gennady Bocharov
Marchuk Institute of Numerical Mathematics
Russian Academy of Sciences
Moscow, 119333, Russian Federation

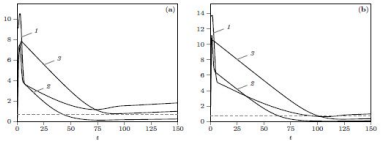


Figure 6. Computational Experiment 5. Dynamics of $y_1(t)$, $y_2(t)$, $y_3(t)$ for $R_0 = 13.2135$. (a) $V^0 = 10^2$, (b) $V^0 = 10^6$.

spreading of HIV infection, as a consequence, the infection dynamics is different compared to the previous experiment 5.

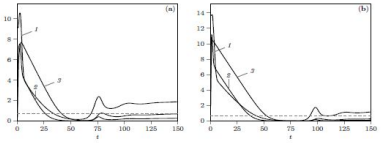


Figure 7. Computational Experiment 6. Dynamics of $y_1(t)$, $y_2(t)$, $y_3(t)$ for $R_0 = 3.3789$. (a) $V^0 = 10^4$, (b) $V^0 = 10^6$.

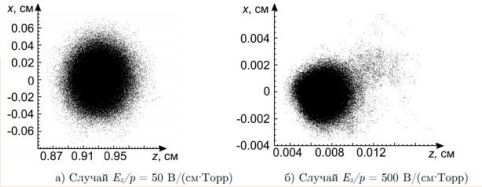
The results presented in Table 1 and Figure 7 show that an increase of α_L can lead to the dynamics reminiscent of infection eradication in a relatively short time after initial infection in contrast to the behavior observed in Experiment 5.

УДК 519.245, 537.563.22

ЭФФЕКТИВНОЕ ИСПОЛЬЗОВАНИЕ МНОГОЯДЕРНЫХ СОПРОЦЕССОРОВ ПРИ СУПЕРКОМПЬЮТЕРНОМ СТАТИСТИЧЕСКОМ МОДЕЛИРОВАНИИ ЭЛЕКТРОННЫХ ЛАВИН[1]

М.А. Марченко

Для моделирования развития электронных лавин в газе разработаны трехмерный параллельный алгоритм метода Монте-Карло и программа ELSHOW, реализованная с использованием комбинирования принципов крупно- и мелкозернистого параллелизма. Для реализации параллельных вычислений на высокопроизводительных гибридных вычислительных системах с сопроцессорами Intel Xeon Phi используется хорошо зарекомендовавшая себя библиотека PARMONC. Применение разработанной технологии распараллеливания существенно уменьшает вычислительную трудоемкость оценки таких интегральных характеристик, как число частиц в лавине, коэффициент ударной ионизации, скорость дрейфа и др.



а) Случай $E_z/p = 50$ В/(см·Торр) б) Случай $E_z/p = 500$ В/(см·Торр)

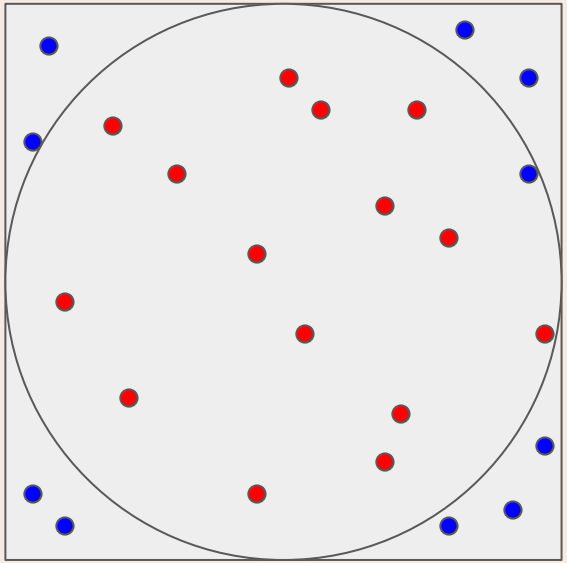
Future Applications

- Modelling Baikal ecology
- Virus epidemiology modelling
- Modeling risks for power transmission systems
- Modeling the formation of atmospheric aerosols

ИВМиМГ

Monte Carlo Methods

example : $S_{circle} = S_{square}$ * <number of red points> / <total number of points>



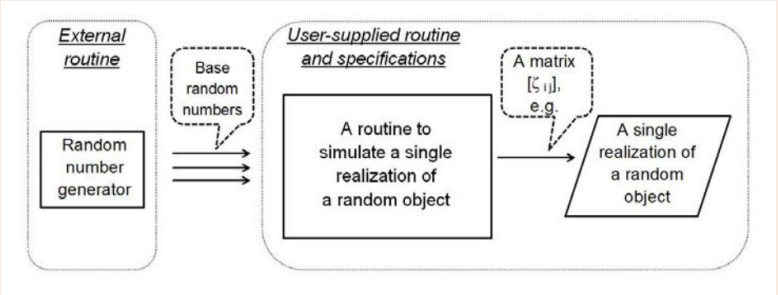
General scheme of the methods:

- generate realizations of a random variable (a realization can be a very complex object -- such as an electron cloud in the ELSHOW)
- compute some values based on the set of realizations

ИВМиМГ

PARMONC: principles

- An application developer provides a **sequential subroutine (C/C++/Fortran) to simulate a single realization of a random object** of interest (the subroutine can use a random number generator from PARMONC or other generators)
- PARMONC **engine calls the subroutine** multiple times in parallel to compute large number of realizations **on a parallel computer**
- In the course of simulation, the PARMONC periodically calculates and **saves in files the subtotal results of simulation**
- The PARMONC provides **check-pointing and restore** of computations.



ИВМиМГ

ELSHOW: electron avalanche simulation: comparison of results on Intel (original code) and Kunpeng (ported code) based computers

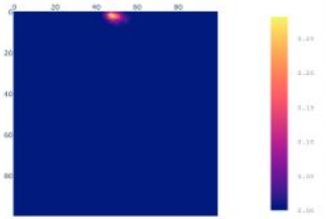
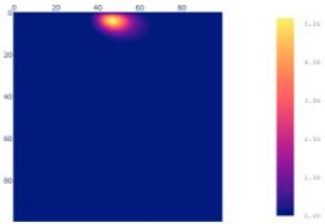
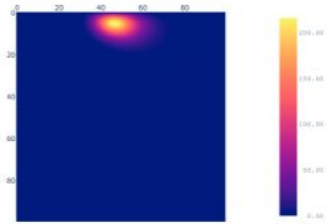
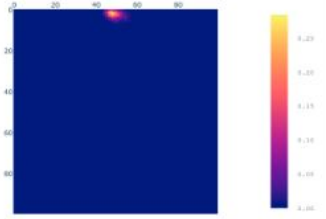
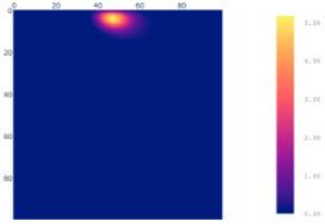
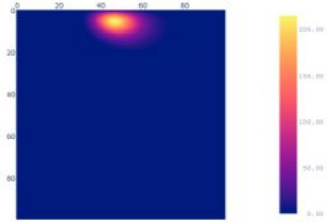
Random object: an electron cloud in its development

PARMONC runs in parallel multiple simulations of cloud development and averages the result:

we see the behavior of the averaged cloud dynamic

***

subtle differences can be explained by the random nature of cloud dynamic and different implementation of math functions

Time step (approx)	80000	240000	400000
Original code, Intel Core i9-9900 K, Intel tools, 8 MPI Process			
Ported, Kunpeng 920, GNU Tools + Open MPI, 8 MPI Processes			

Visualization of the dynamic of an averaged electron cloud: original code (Intel Compilers, x86-64 CPU) vs ported version (GNU Compilers, Kunpeng).

ИВМиМГ

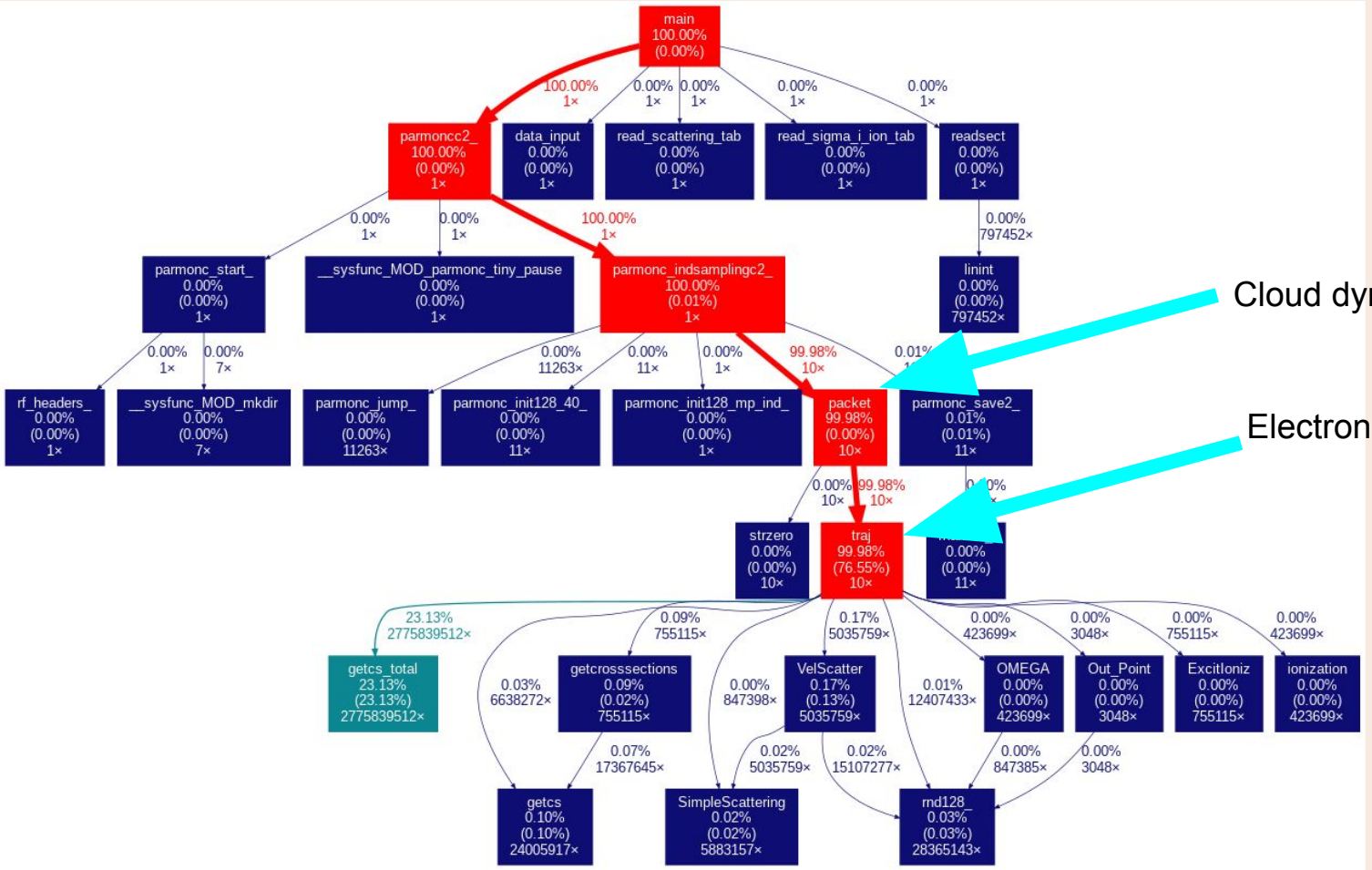
PARMONC and ELSHOW porting

The software was developed for Intel-based cluster. Specific constructs supported by Intel Compilers were used in the program .

These parts of the software were rewritten to comply with current compilers from GCC.

HML library was used for math functions: sin, cos, exp, log, log10, pow

ИВМиМГ


main
100.00%
(0.00%)
parmoncc2_
100.00%
(0.00%)
1×
data_input
0.00%
(0.00%)
1×
read_scattering_tab
0.00%
(0.00%)
1×
read_sigma_i_ion_tab
0.00%
(0.00%)
1×
readsect
0.00%
(0.00%)
1×
parmonc_start_
0.00%
(0.00%)
1×
__sysfunc_MOD_parmonc_tiny_pause
0.00%
(0.00%)
1×
parmonc_indsamplingc2_
100.00%
(0.01%)
1×
linint
0.00%
(0.00%)
797452×
rf_headers_
0.00%
(0.00%)
1×
__sysfunc_MOD_mkdir
0.00%
(0.00%)
7×
parmonc_jump_
0.00%
(0.00%)
11263×
parmonc_init128_40_
0.00%
(0.00%)
11×
parmonc_init128_mp_ind_
0.00%
(0.00%)
1×
packet
99.98%
(0.00%)
10×
parmonc_save2_
0.01%
(0.01%)
11×
strzero
0.00%
(0.00%)
10×
traj
99.98%
(76.55%)
10×
getcs_total
23.13%
(23.13%)
2775839512×
getcrosssections
0.09%
(0.02%)
755115×
VelScatter
0.17%
(0.13%)
5035759×
OMEGA
0.00%
(0.00%)
423699×
Out_Point
0.00%
(0.00%)
3048×
ExcitIoniz
0.00%
(0.00%)
755115×
ionization
0.00%
(0.00%)
423699×
getcs
0.10%
(0.10%)
24005917×
SimpleScattering
0.02%
(0.02%)
5883157×
rnd128_
0.03%
(0.03%)
28365143×
Cloud dynamic
Electron Random Path

ИВМиМГ

Slight difference in numerical results: averaged cloud energy

Step number (approx)	Original code, Intel Core i9-9900K, Intel tools, 1 MPI Process	Ported, Kunpeng 920, GNU Tools (GCC version 9.3.1) + Open MPI, 1 MPI Process (GNU stdlib math library)	Ported, Kunpeng 920, GNU Tools (GCC version 9.3.1) + Open MPI, 1 MPI Process (HML math library)
0	0	0	0
40000	48.8385536637895	48.8385536637896	48.8385536637895
80000	156.996305180383	156.996305180383	156.996305180383
120000	436.968068994815	436.968068994815	436.968068994816
160000	1296.97817736653	1296.97817736653	1296.97817736653
200000	3840.51391941098	3840.51391941098	3840.51391941096
240000	11442.3547202889	11442.3547202889	11442.3547202888
280000	33846.4911477149	33846.4911477149	33846.4911477147
320000	100308.739999047	100308.739999047	100308.739999046
360000	296206.364554502	296206.364554504	296206.364554503
400000	874819.331140992	874819.331141071	874819.331140995

ИВМиМГ

Performance: HML vs standard library

Mean time to compute cloud realization:

HML: 0.3000643033298999E+02
Standard lib: 0.3000020763865998E+02

In fact, library math functions do not constitute a substantial part of computations

Ported, Kunpeng 920, GNU Tools (GCC version 9.3.1) + Open MPI, 1 MPI Process
(HML math library for sin, cos, exp, log, log10, pow; daxpy was not used)

```
/home/maxim/hu/work/charts/hml_tests/O3_with_hml/original/parmonc_d
   0.1000000000000000E+03      total sample volume
   0.3000643033298999E+02      mean computational time per sample
   0.3532431598439873E+21      absolute error upper bound
   0.1000000000000000E+10      relative error upper bound in %
   0.1386452555295164E+43      variance upper bound
```

Ported, Kunpeng 920, GNU Tools (GCC version 9.3.1) + Open MPI, 1 MPI Process (GNU GLIBC math library)

```
/home/maxim/hu/work/charts/hml_tests/O3_stdlib/o
   0.1000000000000000E+03      total sample vol
   0.3000020763865998E+02      mean computation
   0.3532431598439885E+21      absolute error u
   0.1000000000000000E+10      relative error u
   0.1386452555295174E+43      variance upper b
```

ИВМиМГ

HML vs Standard lib: Synthetic test

```
[maxim@hpc-lab examples]$ ./hml_libm_example1_c 10
Sin arg 10.0000000000000000 (0x1.4p+3) = -0.0383444461752397 (-0x1.3a1e21c9c9d3p-5), time = 0.018627
Cos arg 10.0000000000000000 (0x1.4p+3) = 0.3689383389207389 (0x1.79caf8cf9edc6p-2), time = 0.020986
Exp arg 10.0000000000000000 (0x1.4p+3) = 1005016.8038934221258387 (0x1.eabb19b97eb2bp+19), time = 0.010903
Log arg 10.0000000000000000 (0x1.4p+3) = -5605050.8214816786348820 (-0x1.561aeb49327e4p+22), time = 0.014681
Log10 arg 10.0000000000000000 (0x1.4p+3) = -2434242.6425567297264934 (-0x1.29261523f4c86p+21), time = 0.018082
Pow arg 10.0000000000000000 (0x1.4p+3) = 1.0000001100000144 (0x1.000001d8724b8p+0), time = 0.024535
[maxim@hpc-lab examples]$ ./libm_example1_c 10
Sin arg 10.0000000000000000 (0x1.4p+3) = -0.0383444461752426 (-0x1.3a1e21c9c9edp-5), time = 0.026740
Cos arg 10.0000000000000000 (0x1.4p+3) = 0.3689383389206684 (0x1.79caf8cf9e8dp-2), time = 0.027876
Exp arg 10.0000000000000000 (0x1.4p+3) = 1005016.8038934221258387 (0x1.eabb19b97eb2bp+19), time = 0.010650
Log arg 10.0000000000000000 (0x1.4p+3) = -5605050.8214816786348820 (-0x1.561aeb49327e4p+22), time = 0.014806
Log10 arg 10.0000000000000000 (0x1.4p+3) = -2434242.6425567297264934 (-0x1.29261523f4c86p+21), time = 0.021731
Pow arg 10.0000000000000000 (0x1.4p+3) = 1.0000001100000144 (0x1.000001d8724b8p+0), time = 0.023551
```

HML

Standard lib

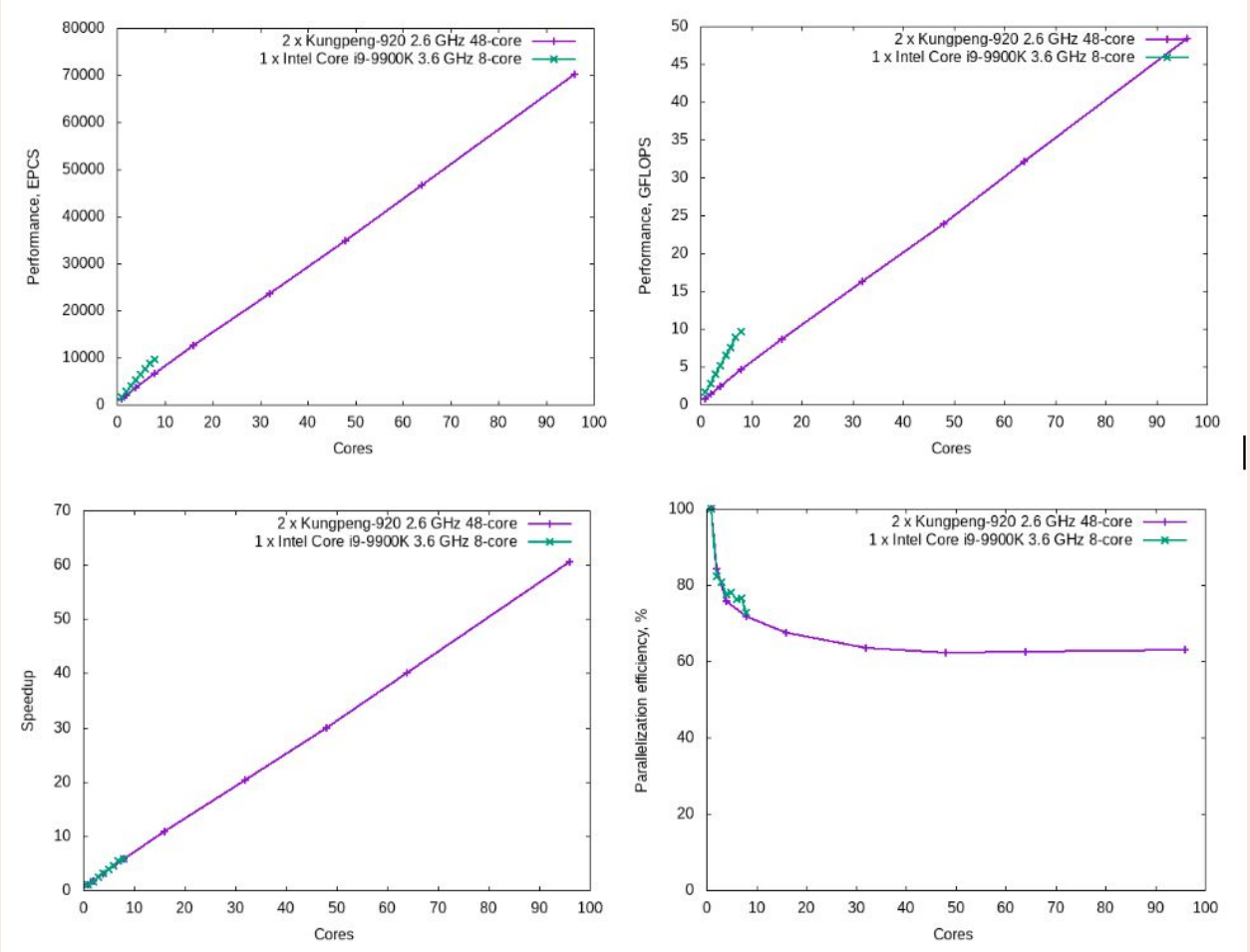
ИВМиМГ

ELSHOW: Parallel efficiency

Intel Core i9-9900K CPU

vs

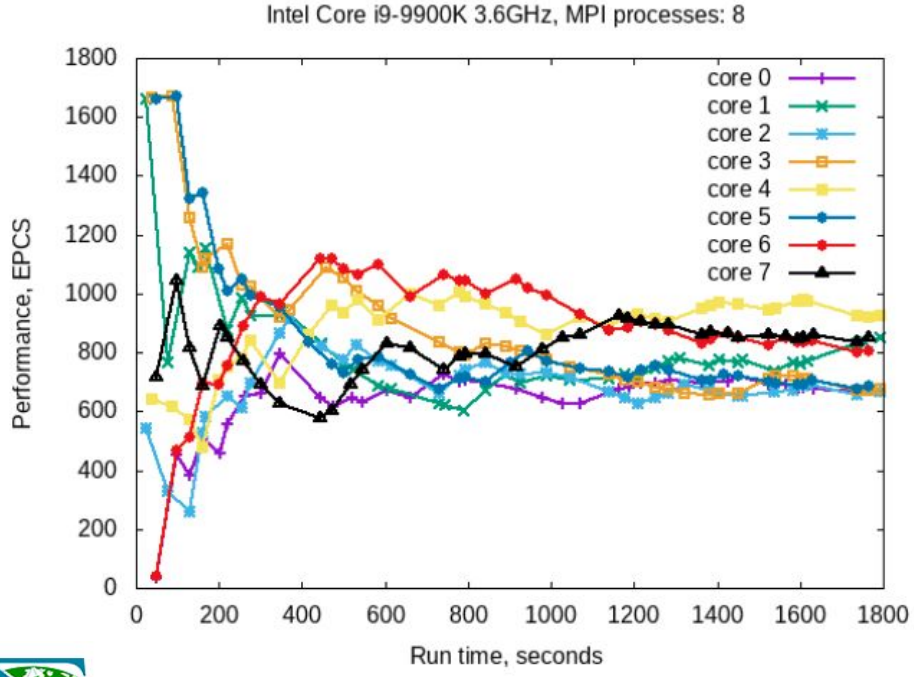
2 x Kunpeng 920-4826 CPU

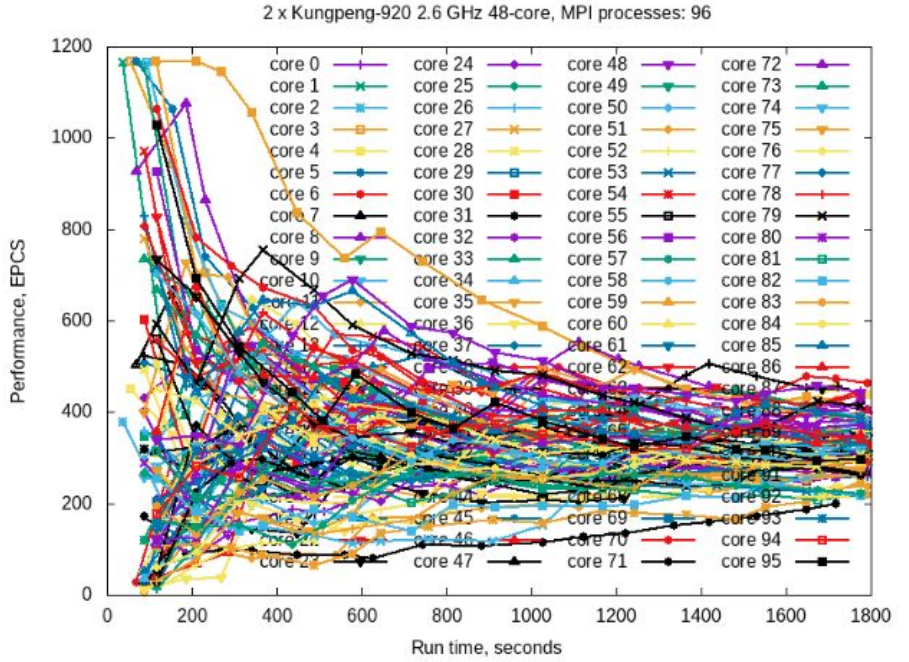


EPCS - Electron Path Calculations per Second

ИВМиМГ

Dynamics of the EPCS value for various MPI processes for two different multi-core computing systems





ИВМиМГ

EPCS - Electron Path Calculations per Second

Basic Compiler Optimization

- **Computer: Kunpeng-920 2.6 GHz**
- **Compiler: gcc + gfortran 9.3.1**
- Differences in time between launches of the same executable: within 5s
- Average time of several experiments was taken
- Adding -march=native and -mtune=native keys does not change the executable code and performance results.

ELSHOW optimization keys	**PARMONC optimization keys**					
	-O0	-O1	-O2	-O3	-O3 -march=native	-O3 -mtune=native
-O0	1057,08	1056,11	1056,05	1055,45	1055,56	1055,49
-O1	462,74	461,7	461,67	461,91	461,84	461,93
-O2	376,72	375,21	375,1	375,21	375,2	375,12

ИВМиМГ

Energy Efficiency

	2 x Kunpeng-920 2.6 GHz 48-core	1 x Intel Core i9-9900K 3.6 GHz 8-core
PARMONC+ELSHOW performance (perpass = 8)	70 255 EPCS	9664 EPCS
TDP	2 x 158 W	95 W
Performance / TPD	222 EPCS/W	102 EPCS/W
PARMONC+ELSHOW power usage	n/a	56.5 W
Performance / power usage	n/a	170.7 EPCS/W

ИВМиМГ

Results

- Kunpeng-based cluster was installed at Siberian Supercomputer Center
- The software was ported:
  - the PARMONC library: a library for development of programs that implement massively parallel stochastic simulation on distributed memory systems
  - the ELSHOW application: simulation of electron avalanches in gases

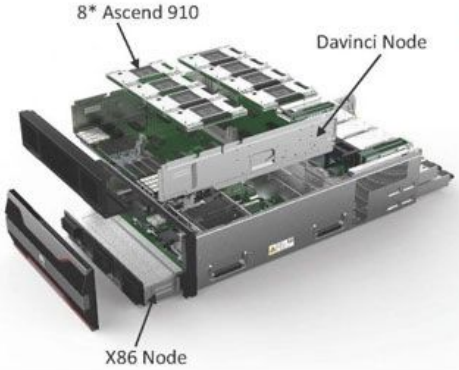
ИВМиМГ

Нейропроцессоры Ascend

ИВМиМГ

Оборудование AI

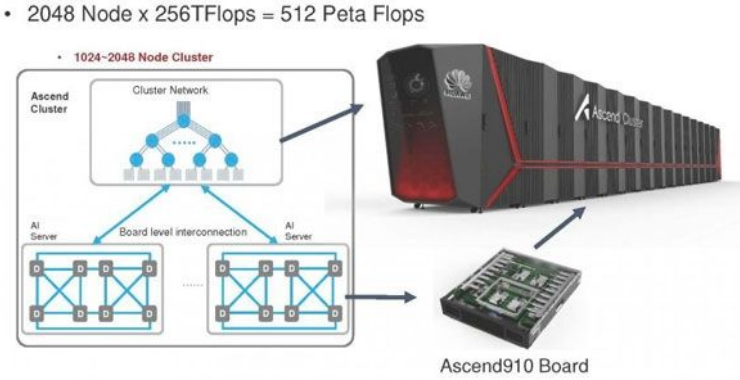
Ascend 910 AI Server



Features	AI Server SPEC.
Specification	8 * Davinci 2 * Xeon CPU + 24 DIMM
Performance	2PFops/Chassis ,256T/AI Module
Memory	24DIMM, Up to 1.5TB
Storage	6 * 2.5inch, NVME; 24TB 2 * 2.5inch, SAS/SATA, Raid1
Interface	8*100G Fiber 4 * PCIe IO
Power	6000W
Ambient Temperature	5~35℃

Ascend 910 Cluster

- 2048 Node x 256TFlops = 512 Peta Flops



256–1024 петафлопс для операций с плавающей запятой (FP16), Ascend 910

https://www.servethehome.com/huawei-ascend-910-provides-a-nvidia-ai-training-alternative/

Heng Liao, Jiajin Tu,
Jing Xia, Xiping Zhou
DaVinci: A Scalable Architecture for Neural Network Computing
2019

Модуль ИИ-ускорителя Atlas 200 (модель 3000)

Ascend 310

Комплект ИИ-инструментов разработчика Atlas 200 DK

Ascend 310

Карта вывода Atlas 300I (Модель 3000/3010)

Ascend 310

Плата машинного обучения Atlas 300T (Модель: 9000)

Ascend 910

Периферийная ИИ-станция Atlas 500

1 Atlas 200

ИИ-сервер граничных вычислений Atlas 500 Pro (Model: 3000)

4 x Atlas 300I

Сервер формирования логических выводов Atlas 800 (Модель: 3000)

8 x Atlas 300I

Сервер формирования логических выводов Atlas 800 (Модель: 3010)

7 x Atlas 300I

Сервер машинного обучения Atlas 800 (модель: 9000)

8 x Atlas 300T, 4 Kunpeng 920
2,24 петафлопс для операций с плавающей запятой (FP16)

https://forum.huawei.com/enterprise/en/huawei-atlas-ai-computing-platform/thread/748583-895

Процессоры Ascend 310 и Ascend 910

Ascend 310

SoC, предназначена для применения ИИ моделей (inference)

16 TOPS@INT8 or 8 TOPS@FP16:

- 2x Da Vinci Max AI cores
- 8x ARM Cortex-A55 CPU cores
- 8 MB on-chip buffer
- 16 channel video decode – H.264/H.265
- 1 channel video encode – H.264/H.265
- TSMC 12 nm FFC process
- 8 W

Ascend 910

SoC, предназначена для обучения моделей

256 TFLOPS@FP16 или 512 TOPS@INT8

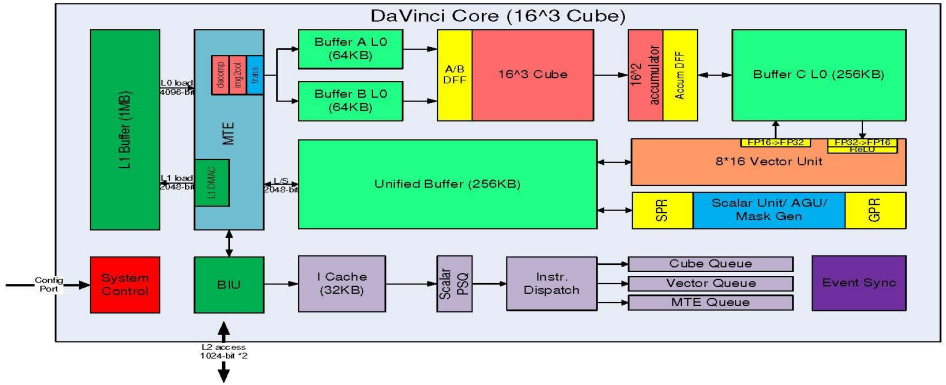
- 32x Da Vinci Max AI cores arranged in 4 clusters
- 1024-bit NoC Mesh @ 2 GHz, with 128 GB/s bandwidth Read/Write per core
- 3x 240Gbit/s HCCS ports for Numa connections
- 2x 100Gbit/s RoCE interfaces for networking
- 4x HBM2E, 1.2 TB/s bandwidth
- 3D-SRAM stacked below AI SoC die
- 32 MB on-chip buffer
- 128 channel video decode – H.264/H.265
- TSMC 7+ nm EUV (N7+) process
- 350 W

https://en.wikipedia.org/wiki/HiSilicon

Ядро DaVinci в максимальной конфигурации (для Ascend 910) содержит 4096 блоков Cube для вычислений с половинной точностью (FP16). Также в ядро входят специализированные блоки для обработки скалярных (INT8) и векторных величин.

Пиковая производительность: 320 терафлопс для операций с плавающей запятой (FP16) и 640 триллионов операций в секунду для операций с целыми числами (INT8) при максимальном потреблении электроэнергии 310 Вт.

DaVinci Core



- **Cube**: 4096(16^3) FP16 MACs + 8192 INT8 MACs
- **Vector**: 2048bit INT8/FP16/FP32 vector with special functions (activation functions, NMS- Non Minimum Suppression, ROI, SORT)
- Explicit memory hierarchy design, managed by MTE

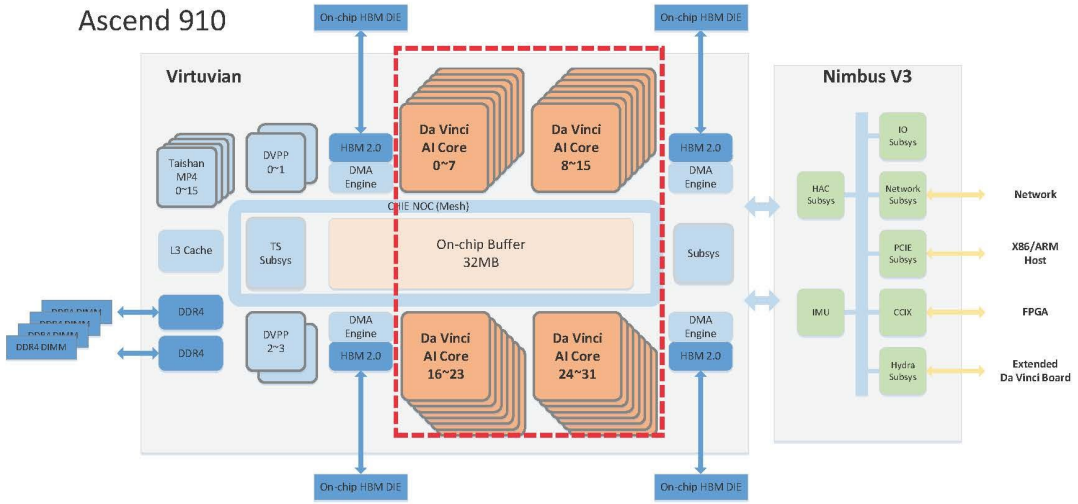
https://www.servethehome.com/huawei-ascend-910-provides-a-nvidia-ai-training-alternative/

core: 3D Cube Tensor Computing Engine (4096 FP16 MACs + 8192 INT8 MACs), Vector unit (2048bit INT8/FP16/FP32) and scalar unit

см. также
https://forum.huawei.com/enterprise/en/huawei-davinci-ai-chip-architecture/thread/616780-895
https://forum.huawei.com/enterprise/en/how-the-da-vinci-architecture-ai-core-helps-to-accelerate-ai-computing/thread/600088-100504

Heng Liao, Jiajin Tu,
Jing Xia, Xiping Zhou
DaVinci: A Scalable Architecture for Neural Network Computing
2019

AI Training SoC



Heng Liao, Jiajin Tu,
Jing Xia, Xiping Zhou
DaVinci: A Scalable Architecture for Neural Network Computing
2019

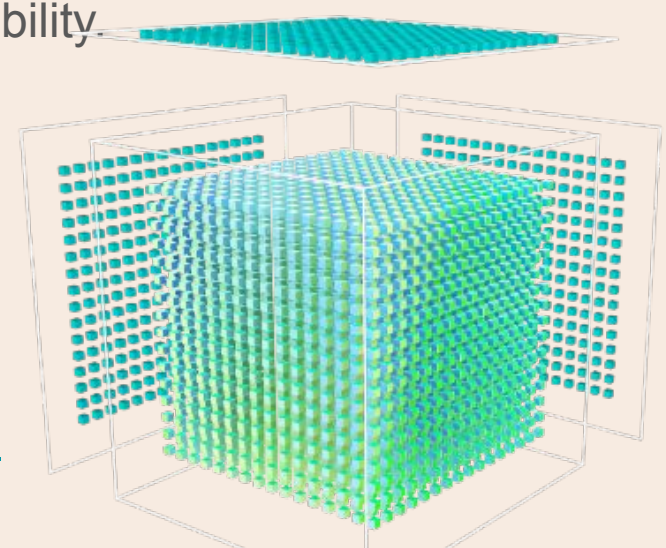
https://www.servethehome.com/huawei-ascend-910-provides-a-nvidia-ai-training-alternative/

Da Vinchi Architecture

3D elastic Cube (16 x 16 x 16)

Powerful computing: Completes 4096 FP16 MAC operations within a clock cycle.

High efficiency: Supports IPs of dozens of milliwatts to processors of hundreds of watts, allowing for smooth device-edge-cloud scalability.

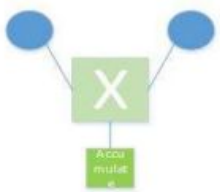


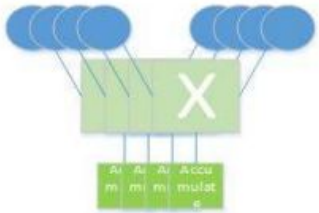
https://www.firstxw.com/view/237148.html

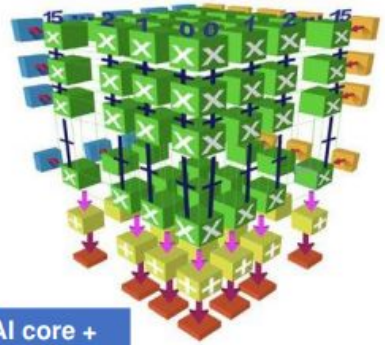
Building Blocks and their Computation Intensity

1D Scalar Unit + **2D Vector Unit** + **3D Matrix Unit**

Full flexibility | Rich & efficient operations | High intensity







N	N^2	N^3
1	1	1
2	4	8
4	16	128
8	64	512
16	**256**	**4096**
32	1024	32768
64	4096	262144

	GPU + Tensor core	AI core + SRAM
Area (normalized to 12 nm)	5.2mm^2	13.2mm^2
Compute power	1.7Tops fp16	8Tops fp16

Heng Liao, Jiajin Tu,
Jing Xia, Xiping Zhou
DaVinci: A Scalable
Architecture for Neural Network
Computing
2019

Overview of Atlas AI Computing Platform

https://www.prnewswire.co.uk/news-releases/huawei-launches-ai-ecosystem-program-in-europe-with-100m-euros-investment-in-5-years-843754587.html

Доступ к оборудованию

- ЦКП “Сибирский суперкомпьютерный центр” ИВМиМГ СО РАН – сейчас
- Облако Huawei – в перспективе

Что надо уметь (для начала)?

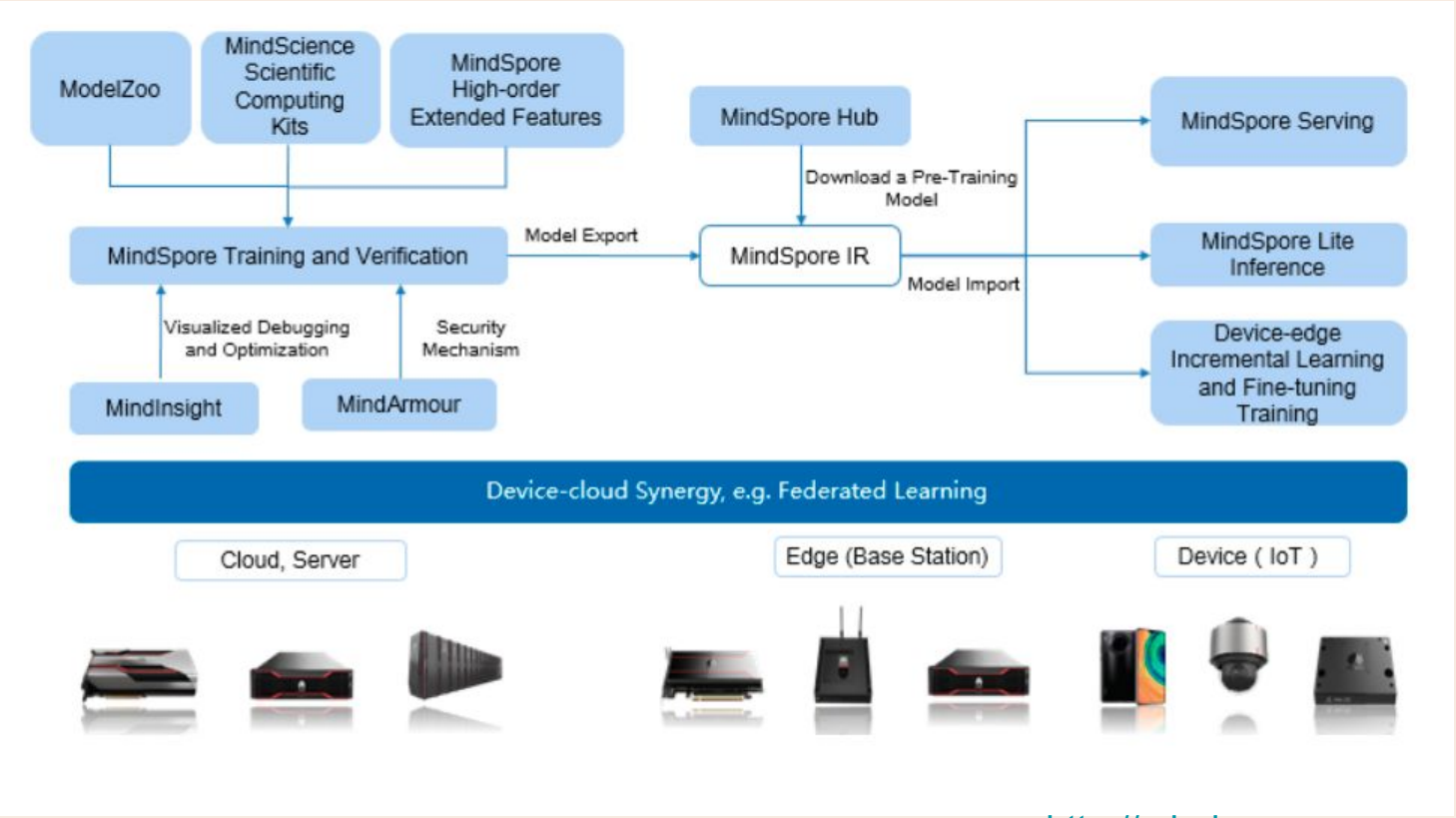
- Linux
- SSH
- Python

MindSpore

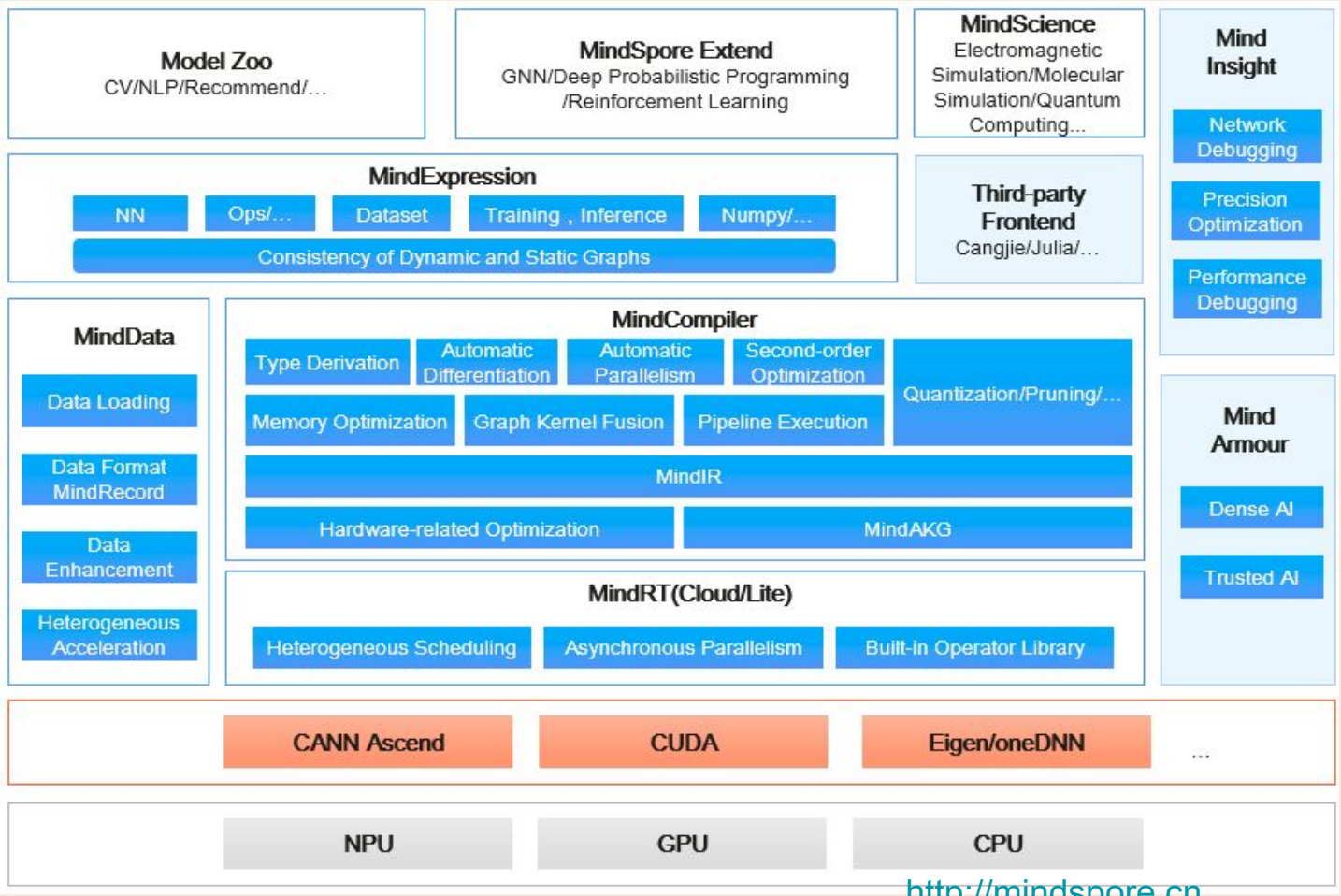
Обзор ML фреймворка MindSpore

ИВМиМГ

Обзор фреймворка Mindspore

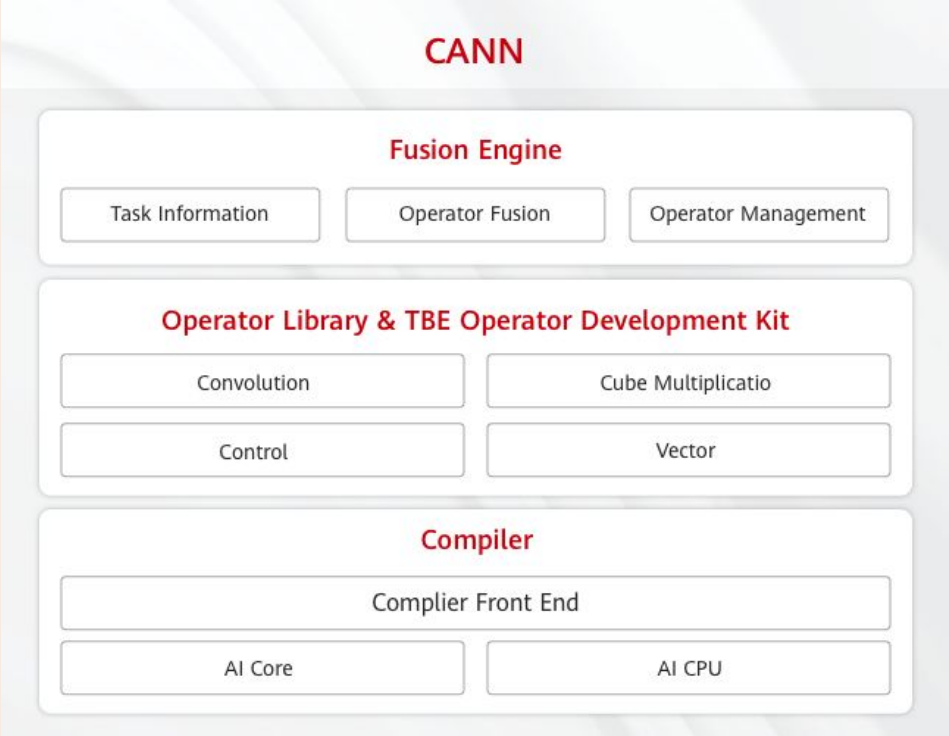


http://mindspore.cn



ИВМиМГ

http://mindspore.cn



An operator library & TBE operator development kit, improving development efficiency to better match the Ascend chip enablement

- Fusion Engine: Operator fusion, reducing operator memory porting

- TBE operator development tool: Various pre-configured APIs for custom operator development and automatic tuning, shortening the project duration and saving manpower

- Operator library: The high-performance operator library based on the Ascend processor, featuring in-depth collaboration and tuning

- Compiler: A compiler and binary toolkit of heterogeneous hybrid programming languages based on C/C++ extension, providing ultimate performance and efficient programming to support Ascend chips in all scenarios

CANN 5.0.4 TBE Custom Operator Developer Guide 01
https://support.huawei.com/enterprise/en/doc/EDOC1100234132

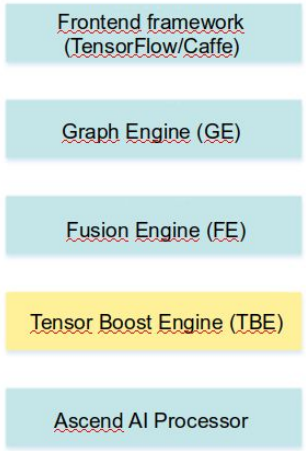
CANN: Compute Architecture for Neural Network

CANN Layer: High-Performance Operators Library and Custo m Operator Development Tools

Tensor Boost Engine (TBE):

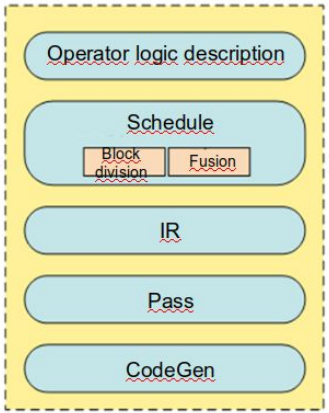
TBE enables custom operator development based on TVM. The corresponding neural network operators can be developed through APIs and the custom operator programming GUI.

Logical architecture of TBE in the Ascend AI Software Stack



- **Front-end frameworks**: third-party open-source frameworks including TensorFlow and Caffe.
- **GE**: a unified IR API running on the Ascend AI Software Stack that connects to different machine learning frameworks, such as TensorFlow or Caffe. GE supports graph preparation, splitting, optimization, building, loading, execution, and management operations.
- **FE**: interconnects with GE and TBE operators. It supports operator information library and fusion pattern management, source graph fusion, and subgraph optimization. GE transfers subgraphs to FE for optimization; FE performs prebuild actions, such as modifies data type, inserts transformation operators, on the subgraphs; and then the subgraphs are returned to GE for subgraph fusion and optimization.
- **TBE**: infers necessary operator information based on IR-defined GE graphs, and provides subgraph optimization and TBE operator call information to the FE based on the operator information library and fusion patterns. Binary files generated by TBE are used by Ascend AI Processors to generate tasks to be executed on Ascend AI Processors.

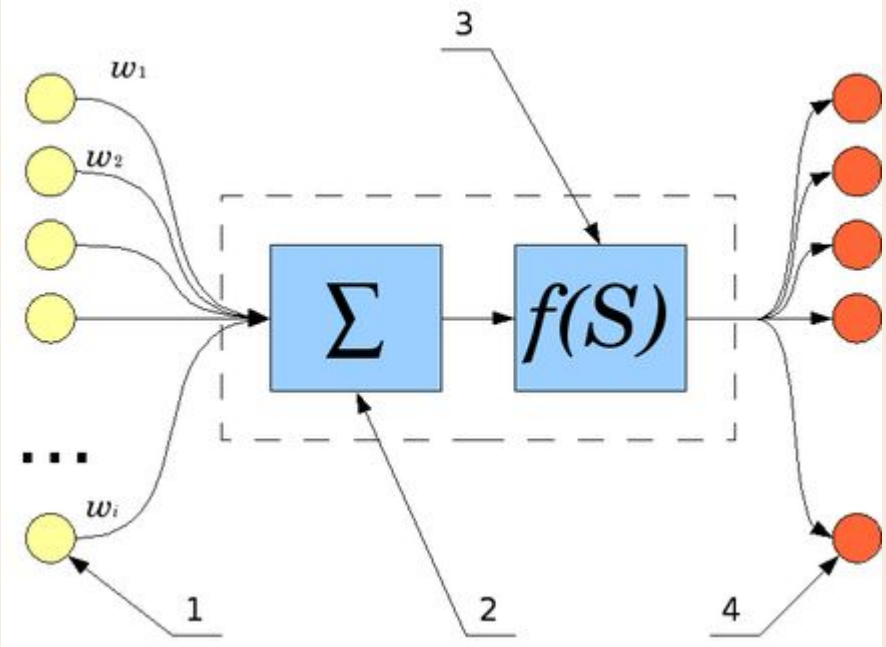
TBE functional framework

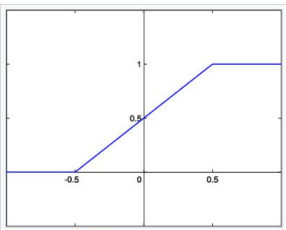


- **DSL**: provides developers with compute APIs for operator logic programming.
- **Schedule**: determines shape-oriented tiling policies using scheduling primitives for Ascend AI Processor operations. Different tiling policies are adopted for Cube, Vector, and other operators.
- **IR**: facilitates functions such as IR transformation and abstract syntax tree (AST) maintenance. It is represented by the community IR.
- **Pass**: introduces a range of build optimizations on the generated IR, including double buffering, pipeline synchronization, memory allocation management, instruction mapping, tiling for adapting to the Cube Unit.
- **CodeGen**: generates a temporary C-style code file, which can be used by the compiler to generate the operator implementation file or directly loaded and executed by a network model.

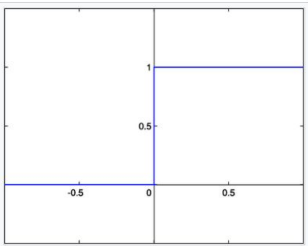
https://support.huawei.com/enterprise/en/ascend-computing/cann-pid-251168373

Искусственный нейрон



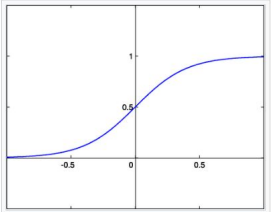


Линейная функция активации с насыщением



Пороговая функция активации

1. Нейроны, выходные сигналы которых поступают на вход данному
2. Сумматор входных сигналов
3. Вычислитель передаточной функции
4. Нейроны, на входы которых подаётся выходной сигнал данного
5. w_{i} — *веса* входных сигналов

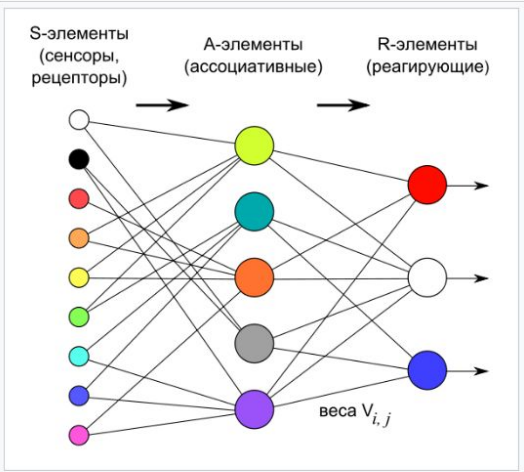


Сигмоидальная функция активации

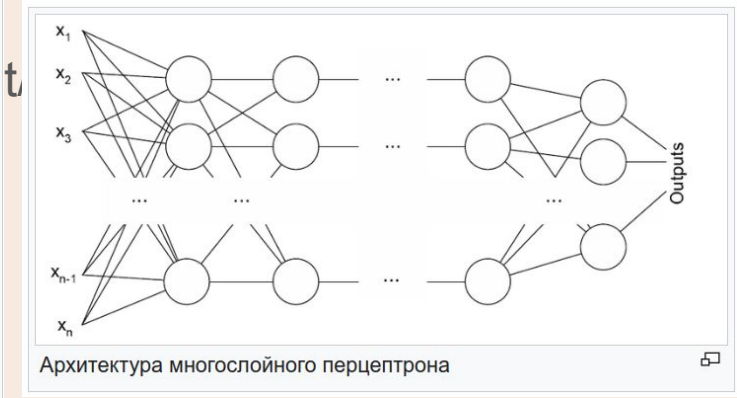
$$y = f(u), \text{ где } u = \sum_{i=1}^{n} w_i x_i + w_0 x_0$$

wikipedia

Искусственные нейронные сети (ANN)



Логическая схема перцептрона с тремя выходами



Архитектура многослойного перцептрона

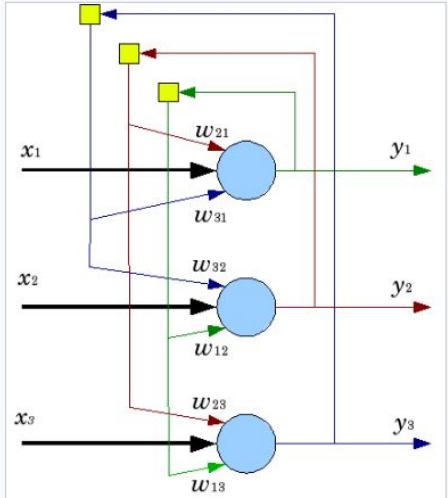


Схема сети Хопфилда с тремя нейронами

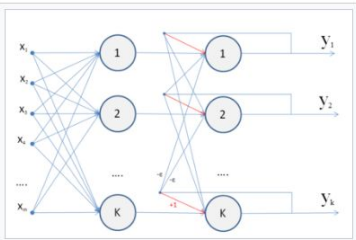
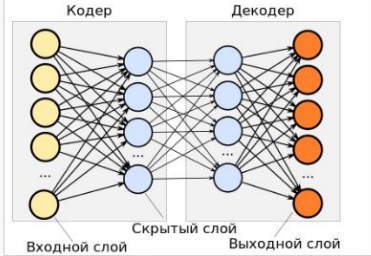


Схема сети Хемминга

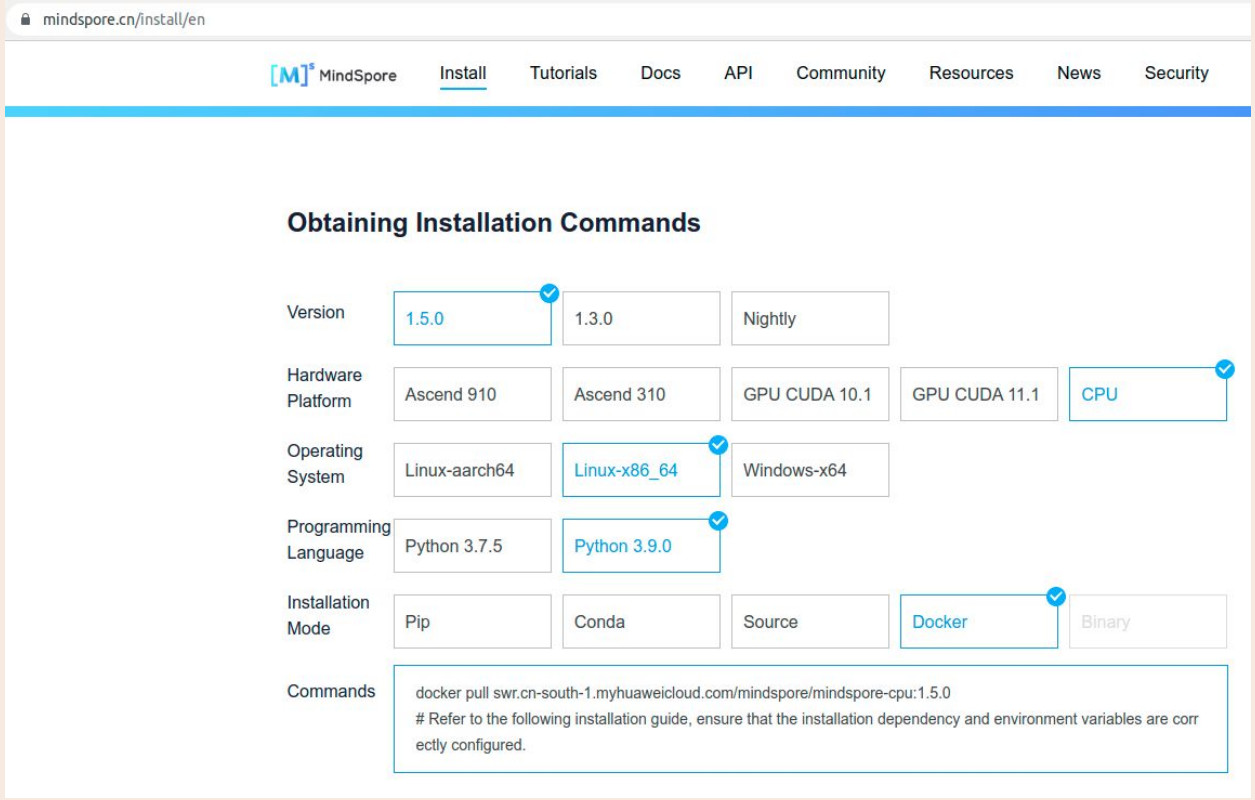


«Бутылочное горлышко» — одна из возможных топологий нейросетей, используемых для сжатия

wikipedia

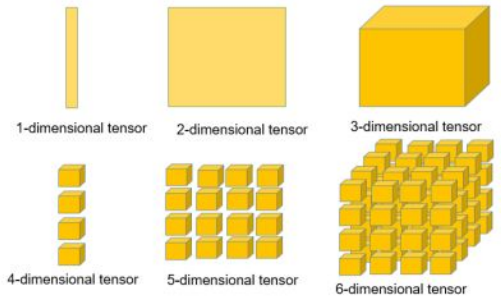
Установка MindSpore

https://www.mindspore.cn/install/en



ИВМиМГ

Концепции: тензор (mindspore.Tensor)



1. Tensor declaration and usage:

```
t1 = Tensor(np.zeros([1, 2, 3]),
    ms.float32)
assert isinstance(t1, Tensor)
assert t1.shape == (1, 2, 3)
assert t1.dtype == ms.float32
```

2. Tensor is the data carrier for parameters.

Parameter attributes:

- **default_input**: Tensor
- **name**: str
- **requires_grad**: bool
- **layerwise_parallel**: bool

3. Common tensor operations:

```
// Converts to NumPy arrays.
asnumpy()
// Obtains a tensor size.
size()
// Obtains the number of
dimensions of a tensor.
dim()
// Obtains a data type of a
tensor.
dtype
// Sets a data type of a tensor.
set_dtype()
// Obtains a shape of a tensor.
shape
```

https://bbs.huaweicloud.com/blogs/314096

Концепции: оператор (mindspore.Ops и др)

Common operators:

- array: Array-related operators
  - ExpandDims - Squeeze
  - Concat - OnesLike
  - Select - StridedSlice
  - ScatterNd
  - ...
- math: Math-related operators
  - AddN - Cos
  - Sub - Sin
  - Mul - LogicalAnd
  - MatMul - LogicalNot
  - RealDiv - Less
  - ReduceMean - Greater ...
- nn: Network operators
  - Conv2d - MaxPool
  - Flatten - AvgPool
  - Softmax - TopK
  - ReLU - SoftmaxCrossEntropy
  - Sigmoid - SmoothL1Loss
  - Pooling - SGD
  - BatchNorm - SigmoidCrossEntropy ...
- control: control operators
  - ControlDepend
- other

```
>>> data1 = Tensor(np.array([[0, 1], [2,
1]]).astype(np.int32))
>>> data2 = Tensor(np.array([[0, 1], [2,
1]]).astype(np.int32))
>>> op = P.Concat()
>>> output = op((data1, data2))
```

```
>>> cos = P.Cos()
>>> input_x = Tensor(np.array([0.24, 0.83, 0.31, 0.09]),
mindspore.float32)
>>> output = cos(input_x)
```

```
>>> input_x = Tensor(np.array([-1, 2, -3, 2, -1]),
mindspore.float16)
>>> relu = nn.ReLU()
>>> relu(input_x)
[0.  2.  0.  2.  0.]
```

Основные модули

mindspore.dataset	Предоставляет API для загрузки и обработки общих наборов данных, таких как MNIST, CIFAR-10, VOC, COCO, ImageFolder и CelebA. transforms предоставляет API-интерфейсы обработки и дополнения данных на основе OpenCV, PIL и собственной реализации. text предоставляет API для обработки текста.
mindspore.common	Предоставляет API для тензоров, параметров (таких как вес и смещение), типов данных (dtypes) и инициализаторов (для инициализации параметров).
mindspore.context	Задает контекст, например режимы Graph и PyNative, тип устройства, IR-граф или хранилище данных, профилирование, память устройства и автоматический параллелизм.
mindspore.nn	Общие операторы нейронной сети, такие как Layer, Loss, Optimizer, Metrics и базовые классы Cell и Wrapper.
mindspore.ops	Примитивные операторы (требуется инициализация), составные операторы и функциональные операторы (которые являются инициализированными примитивными операторами).
mindspore.train	Модель (API для обучения, проверки и вывода), обратный вызов (колбэки) (мониторинг потерь, сохранение контрольных точек и сводная запись), сериализация (загрузка контрольных точек и экспорт модели) и квантизация (квантование).

MindSpore Features

1. Auto Differ. Python APIs for higher

efficiency. • IR-based compilation

optimization for better

performance

2. Auto Parallelism(data and network model)

3. Chip-oriented deep graph optimization

4. Adaptive graph segmentation

5. Unified model IR across Huawei product lien, CPU, GPU

6. Different debug mode : graph and pynative

MindSpore Design: Auto Differ

Graph: TensorFlow

- Non-Python programming based on graphs
- Complex representation of control flows and higher-order derivatives

Operator overloading: PyTorch

- Runtime overhead
- Backward process performance is difficult to optimize.

Source code transfer: MindSpore

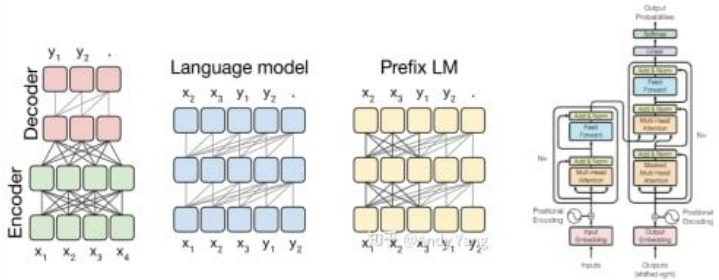
- Python APIs for higher efficiency
- IR-based compilation optimization for better performance

https://forum.huawei.com/enterprise/en/mindspore-architecture-and-mindspore-design/thread/746627-895

Auto Parallelism

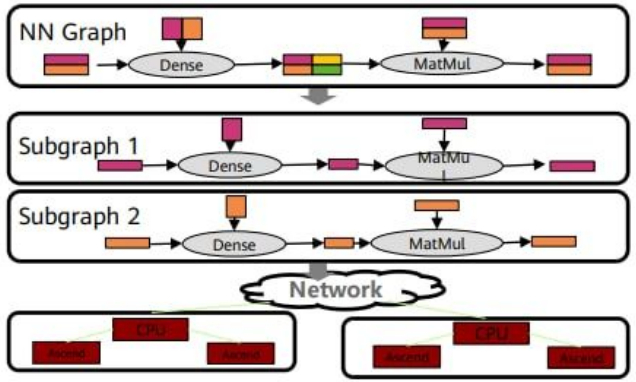
Challenges

Ultra-large models realize efficient distributed training: As NLP-domain models swell, the memory overhead for training ultra-large models such as Bert (340M)/GPT-2(1542M) has exceeded the capacity of a single card. Therefore, the models need to be split into multiple cards before execution. Manual model parallelism is **used currently**. Model segmentation needs to be designed and the cluster topology needs to be understood. The development is extremely challenging. The performance is lackluster and can be hardly optimized.



Key Technologies

Automatic graph segmentation: It can segment the entire graph based on the input and output data dimensions of the operator, and integrate the data and model parallelism. Cluster topology awareness scheduling: It can perceive the cluster topology, schedule subgraphs automatically, and minimize the communication overhead.



Effect: Realize model parallelism based on the existing single-node code logic, improving the development efficiency tenfold compared with manual parallelism.

https://forum.huawei.com/enterprise/en/mindspore-architecture-and-mindspore-design/thread/746627-895

Parallel

- Data parallelism: splits data into many batches and then allocates the batches to each device for model computation
- Model parallelism: splits the model across multiple devices, which includes op-level model parallelism, pipeline model parallelism and optimizer model parallelism
- Hybrid parallelism: contains data parallelism and model parallelism.

https://www.mindspore.cn/docs/en/r1.7/design/distributed_training_design.html?highlight=distributed

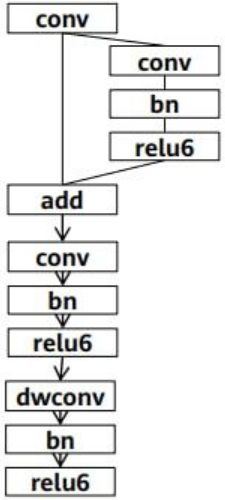
https://www.mindspore.cn/docs/programming_guide/en/r1.6/distributed_training.html

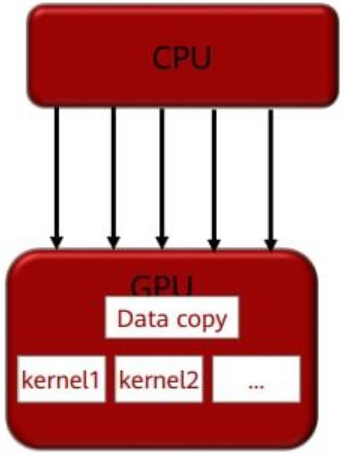
On-Device Execution (1)

Challenges

Challenges for model execution with supreme chip computing power:

Memory wall, high interaction overhead, and data supply difficulty. Partial operations are performed on the host, while the others are performed on the device. The interaction overhead is much greater than the execution overhead, resulting in the low accelerator usage.

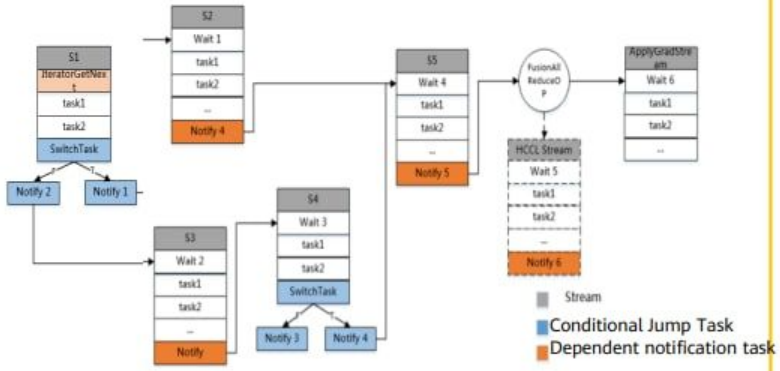




Large data interaction overhead and difficult data supply

Key Technologies

Chip-oriented deep graph optimization reduces the synchronization waiting time and maximizes the parallelism of data, computing, and communication. Data pre-processing and computation are integrated into the Ascend chip:



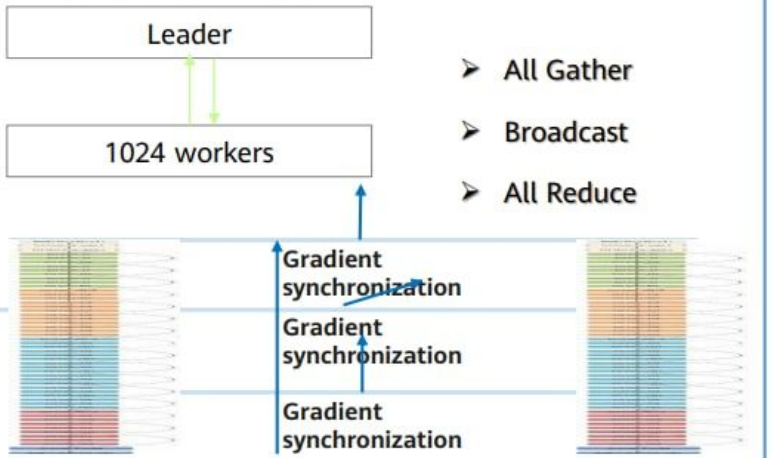
Effect: Elevate the training performance tenfold compared with the on-host graph scheduling.

https://forum.huawei.com/enterprise/en/mindspore-architecture-and-mindspore-design/thread/746627-895

On-Device Execution (2)

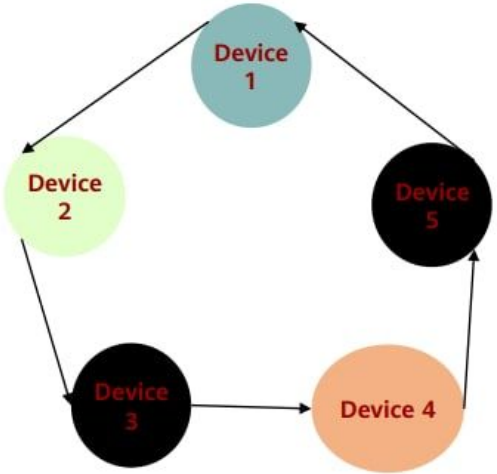
Challenges

Challenges for distributed gradient aggregation with supreme chip computing power:
the synchronization overhead of central control and the communication overhead of frequent synchronization of ResNet50 under the single iteration of 20 ms; the traditional method can only complete All Reduce after three times of synchronization, while the data-driven method can autonomously perform All Reduce without causing control overhead.



Key Technologies

The optimization of the **adaptive graph segmentation driven by gradient data** can realize decentralized All Reduce and synchronize gradient aggregation, boosting computing and communication efficiency.



Effect: a smearing overhead of less than 2 ms

https://forum.huawei.com/enterprise/en/mindspore-architecture-and-mindspore-design/thread/746627-895

Distributed Device-Edge-Cloud Synergy Architecture

Challenges

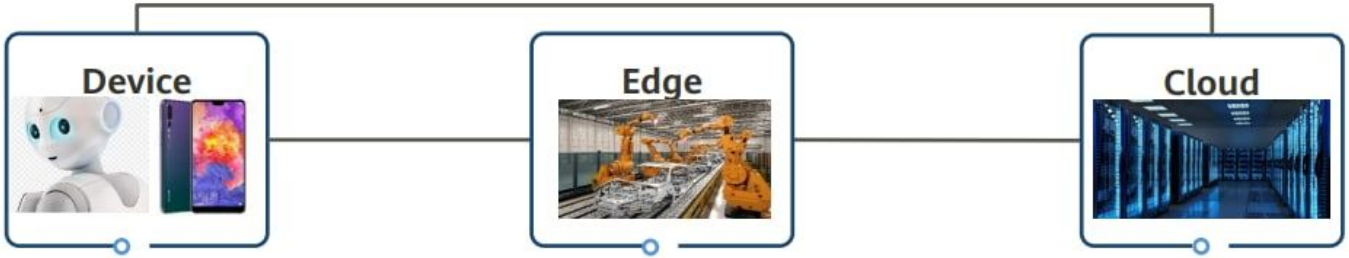
The diversity of hardware architectures leads to full-scenario deployment differences and performance uncertainties. The separation of training and inference leads to isolation of models.

Key Technologies

- **Unified model IR** delivers a consistent deployment experience.
- **The graph optimization technology featuring software and hardware collaboration** bridges different scenarios.
- Device-cloud Synergy Federal Meta Learning breaks the device-cloud boundary and updates the multi-device collaboration model in real time.

Effect: consistent model deployment performance across all scenarios thanks to the unified architecture, and improved precision of personalized models

On-demand collaboration in all scenarios and consistent development experience



https://forum.huawei.com/enterprise/en/mindspore-architecture-and-mindspore-design/thread/746627-895

AI in seismics, optimization on Ascend

Marchenko M.[1,3]
Duchkov A.[2,3]
Gorodnichev M.[1,3]
Nikitin A., [2,3]
Korostelev D., [2,3]
Vershinin M., [2,3]
Russkov A., [1,3]
Marinin I. [1,3]

[1] *Institute of Computational Mathematics and Mathematical Geophysics SB RAS*
[2] *Institute of Petroleum Geology and Geophysics SB RAS*
[3] *Novosibirsk State University*

Outline

- Intro to passive-seismic data processing: event location by coherent summation
- Seismic-wave traveltimes by Neural-Network Eikonal solver – **Ascend**
- passive-seismic data processing – **Kunpeng + Ascend**

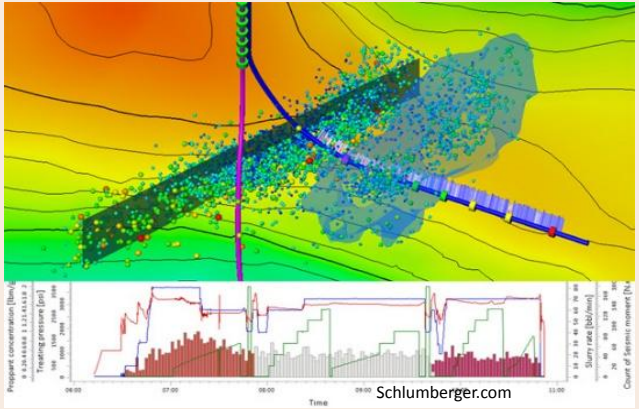
Passive Seismic Monitoring

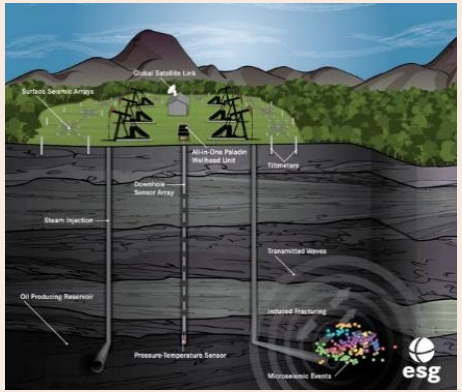
Aim is to study seismic activisations:

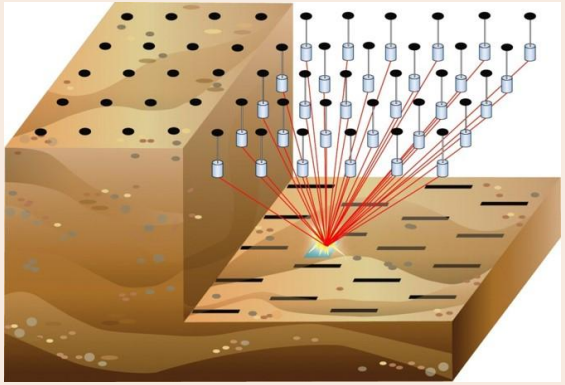
1. Hydraulic fracturing
2. Mining
3. Hydrocarbon field development

Geophysical method:

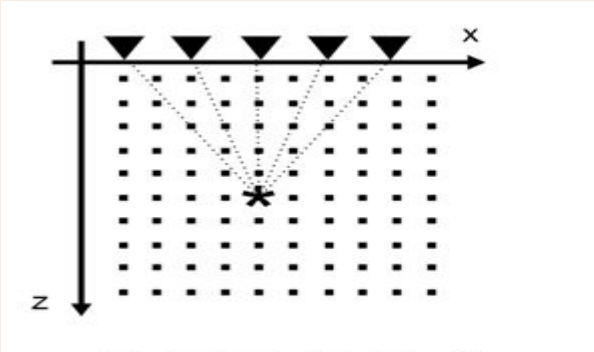
Monitor seismicity by dense surface array. We get several Tb of data.

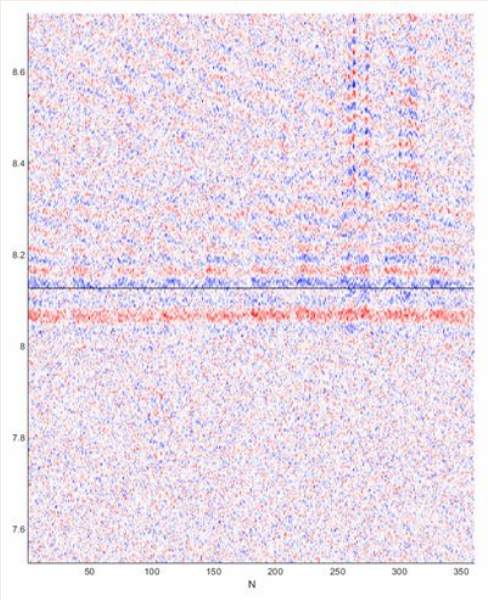


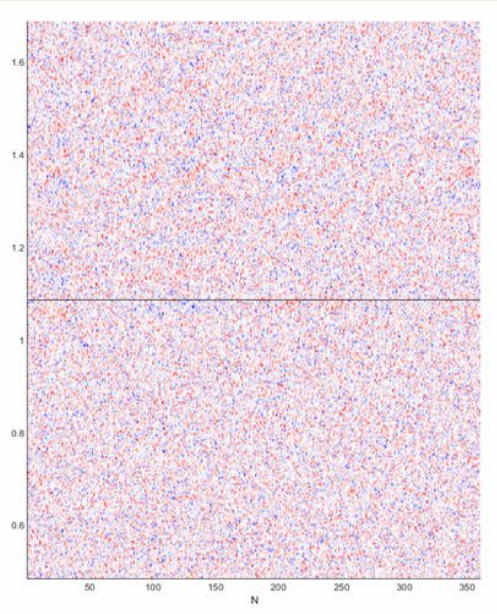




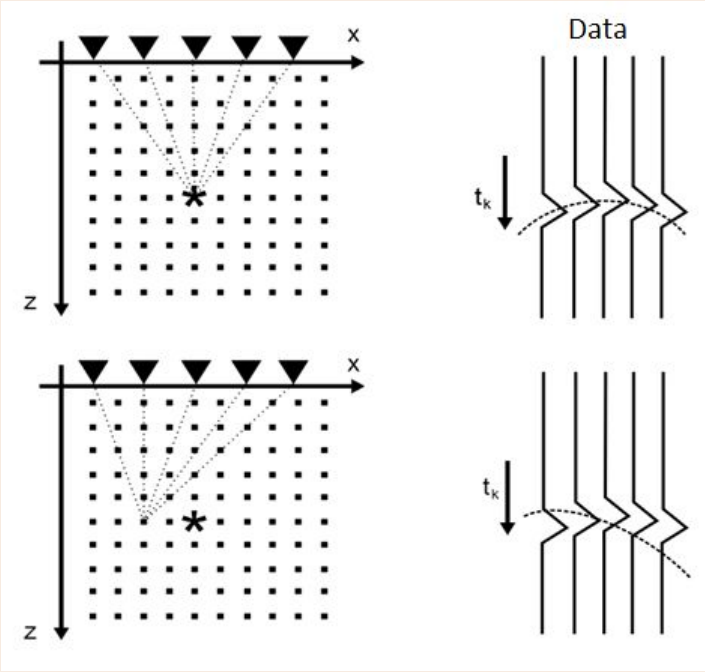
Microseismic data







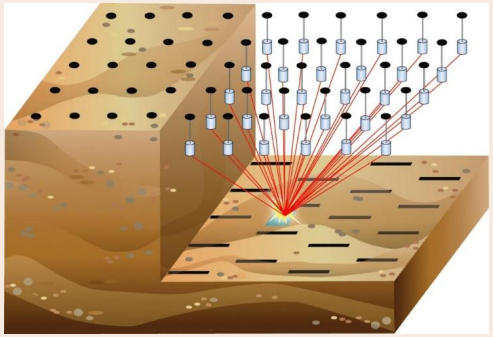
Processing: coherent summation



$$S(\mathbf{x}, t_0) = \sum_r d(\mathbf{x}_r, t_0 + t(\mathbf{x}; \mathbf{x}_r))$$

Processing: coherent summation

Computational algorithm:



```
for  k=1: K   (data samples in time)

   for  j1=1:N1
      for  j2=1:N2          } J  (subsurface grid)
         for  j3=1:N3
```

Subsurface grid:
(N1,N2,N3)=(300,300,300) -> J
Data:
(R, K)=(2500, 5*10^7)

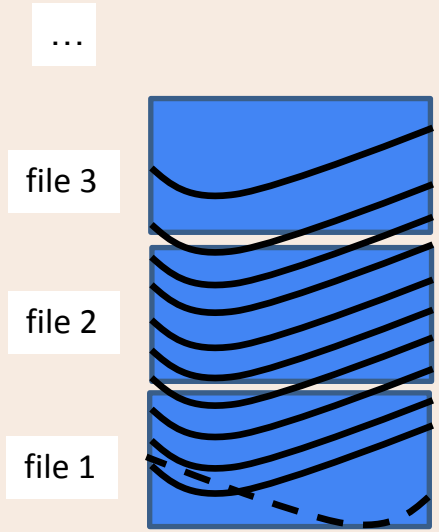
```
for  r=1:R  (receiver array)
   sum parts of traces
```

Processing: coherent summation

Algorithm modification:

Initial algorithm

```
for 1..J do    (subsurface grid)
  for 1..R do
    Traveltime curve
  end for
  for 1..K do     (time steps)
    for 1..R do   (receivers)
      Summation along traveltime curve
    end for
  end for
end for
```



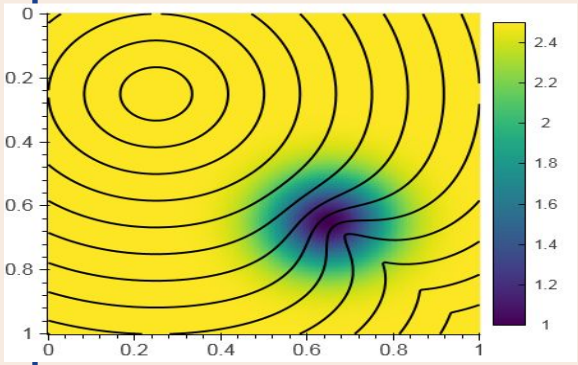
Traveltimes
(propagation of seismic waves)

Eikonal equation

Describes traveltimes of seismic rays:

$$|\nabla \tau(\mathbf{x})|^2 = v(\mathbf{x})^{-2}, \qquad \tau(\mathbf{x}_s) = 0$$

source location



Example of **FMM** solution for the low-velocity anomaly

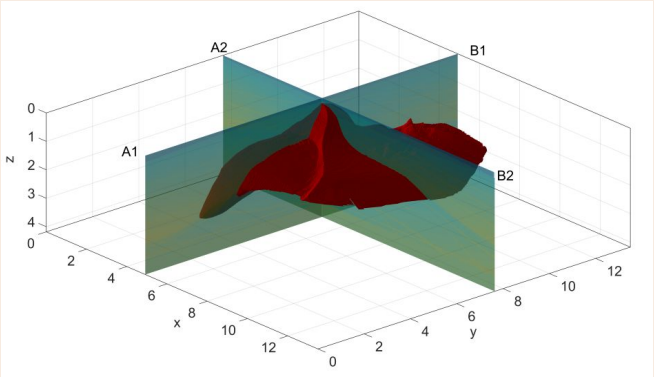
Contour lines are fronts (isochrones)

Numerical algorithms:

- *Fast Marching Method* (**FMM**)
- *Fast Sweeping Method* (**FSM**)

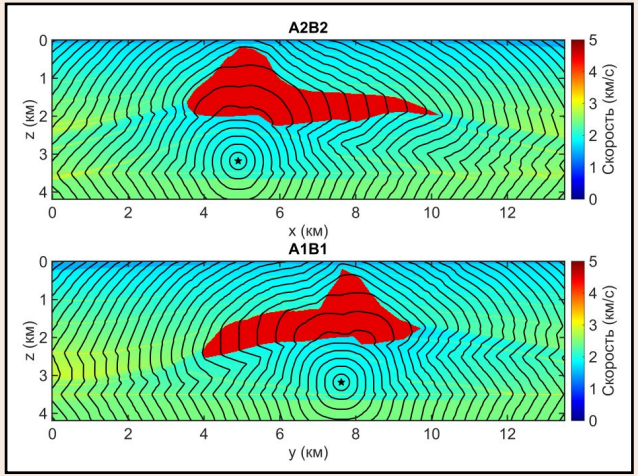
[Sethian, 1996; Zhao, 2005; Fomel, 2009; Treister, 2016]

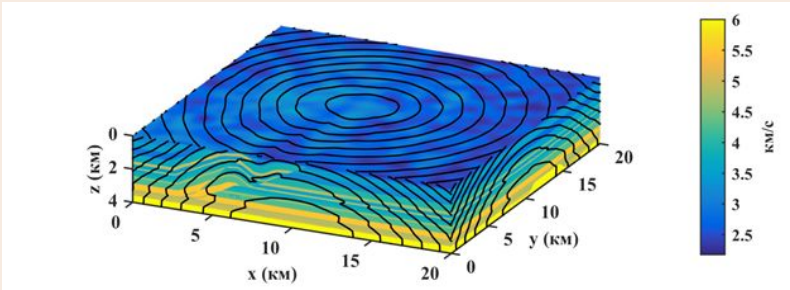
Eikonal solver - FSM



SEG/EAGE Salt (676x676x210)

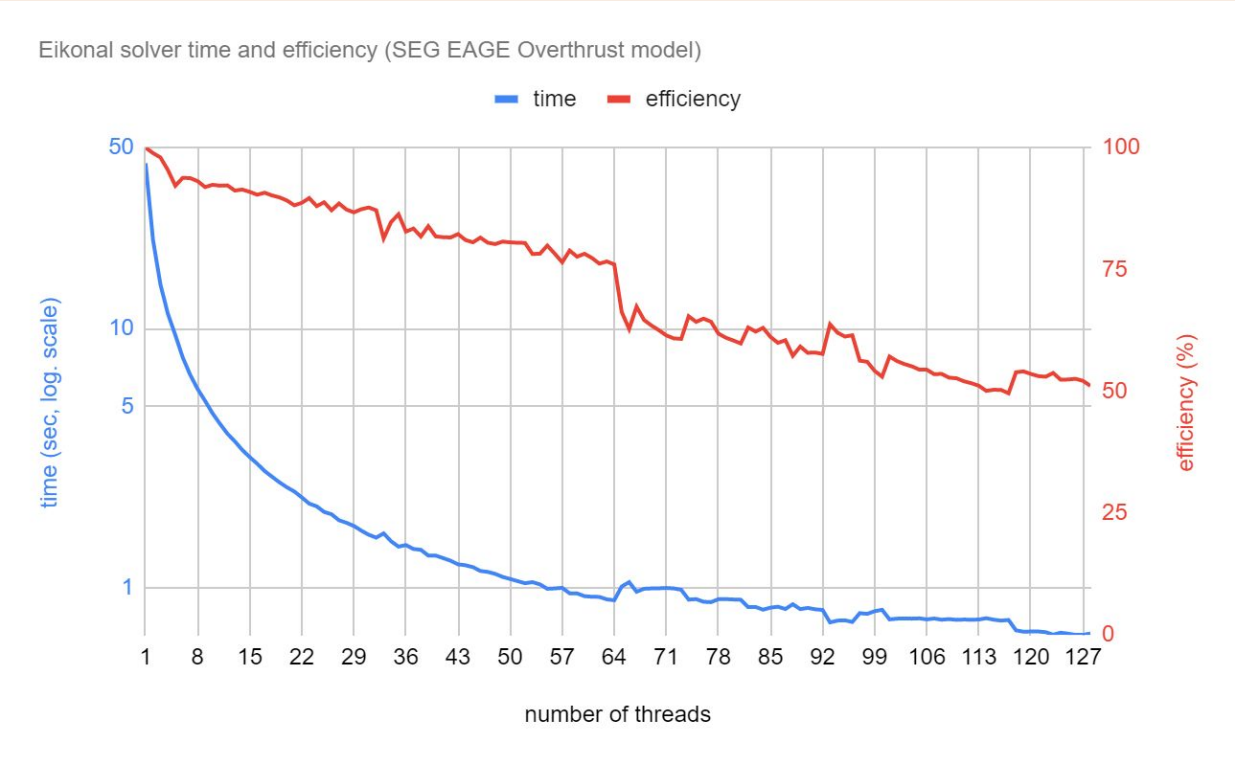
SEG/EAGE Overthrust (801x801x161)





[Nikitin, Serdyukov, Duchkov, 2018]

Eikonal solver (FSM) – testing on Kunpeng



Neural-Network Eikonal Solver on Acsend

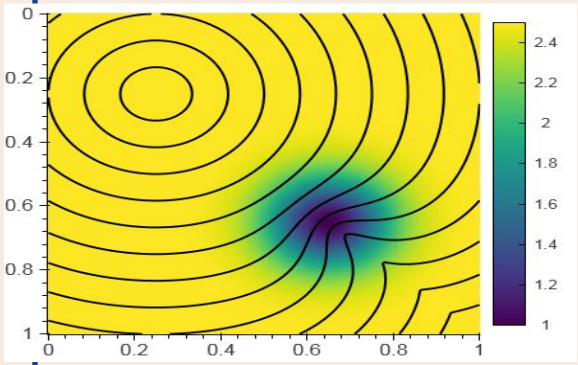
Two-point eikonal equation

$$\tau = \tau(\mathbf{x_s}, \mathbf{x_r}) = \tau(x_s, y_s, z_s, x_r, y_r, z_r) - traveltimes\ for\ any\ source\ and\ receiver\ locations$$

$$|\nabla_r \tau(\mathbf{x_s}, \mathbf{x_r})|^2 = v(\mathbf{x_r})^{-2}, \qquad \tau(\mathbf{x_s}, \mathbf{x_s}) = 0$$

$$\nabla_r = \left(\frac{\partial}{\partial x_r}, \frac{\partial}{\partial y_r}, \frac{\partial}{\partial z_r}\right) - gradient\ operator\ w.r.t.\ \ \mathbf{x_r}$$

$$\mathbf{x_s} = (x_s, y_s, z_s) - source\ location$$
$$\mathbf{x_r} = (x_r, y_r, z_r) - receiver\ location$$



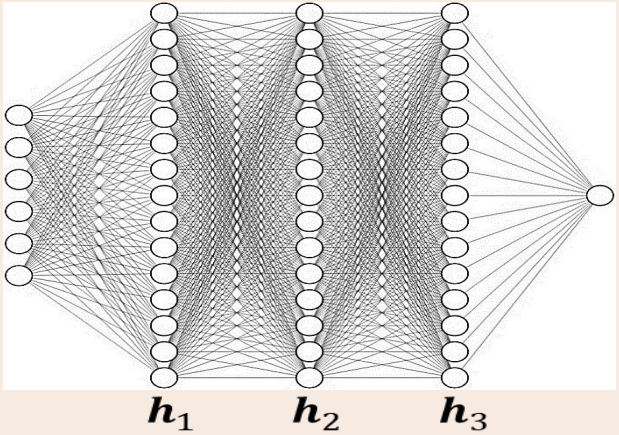
* Two-point formulation also assumes reciprocity principle, so there is one more equation w.r.t. $\mathbf{x_s}$:

$$|\nabla_s \tau(\mathbf{x_s}, \mathbf{x_r})|^2 = v(\mathbf{x_s})^{-2},$$

In this work, we omit and do not consider this additional equation

Artificial neural networks

Fully-connected neural network



Loss function

$$\boldsymbol{L}(\boldsymbol{\Theta}) = MSE = \frac{1}{N}\sum_{i=0}^{N}(f_{\boldsymbol{\Theta}}(\mathbf{x}_i) - y_i)^2 \rightarrow \min_{\boldsymbol{\Theta}}$$

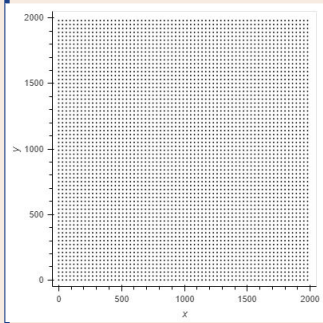
MSE – mean-squared error
N – number of training samples

Physics-informed neural networks

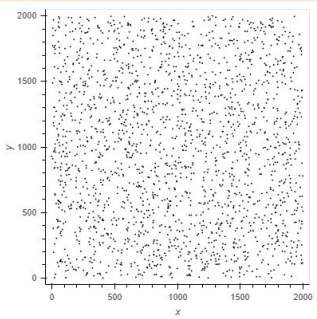
$$|\nabla \tau(\boldsymbol{x_r})|^2 = \frac{1}{V(\boldsymbol{x_r})^2}, \qquad \tau(\boldsymbol{x_s}) = 0$$

$$L = \frac{1}{N_r}\sum_{i=0}^{N_r}\left(\left[V(\boldsymbol{x_r^i})^2|\nabla\tilde{\tau}(\boldsymbol{x_r^i})|^2 - 1\right]^2 + |\tilde{\tau}(\boldsymbol{x_s})|^2\right)$$

$\tilde{\tau} = \tilde{\tau}_{\boldsymbol{\Theta}}(\boldsymbol{x_r})$ — Neural Network, N_r - number of receivers



Training set



Testing set

[Raissi, 2019]

PINN for solving two-point eikonal

$$|\nabla_{\boldsymbol{r}}\tilde{\tau}(\mathbf{x}_{\boldsymbol{s}},\mathbf{x}_{\boldsymbol{r}})|^2 = v(\mathbf{x}_{\boldsymbol{r}})^{-2}, \qquad \tau(\mathbf{x}_{\boldsymbol{s}},\mathbf{x}_{\boldsymbol{s}}) = 0$$

Factorization:

$\tilde{\tau}(\mathbf{x}_{\boldsymbol{s}},\mathbf{x}_{\boldsymbol{r}}) = R \cdot |f_{\boldsymbol{\Theta}}(\mathbf{x}_{\boldsymbol{s}},\mathbf{x}_{\boldsymbol{r}})|$ – PINN approximation of solution τ

$R = \|\mathbf{x}_{\boldsymbol{r}} - \mathbf{x}_{\boldsymbol{s}}\|$ – Euclidean distance

$f_{\boldsymbol{\Theta}}(\mathbf{x}_{\boldsymbol{s}},\mathbf{x}_{\boldsymbol{r}})$ – neural-network output

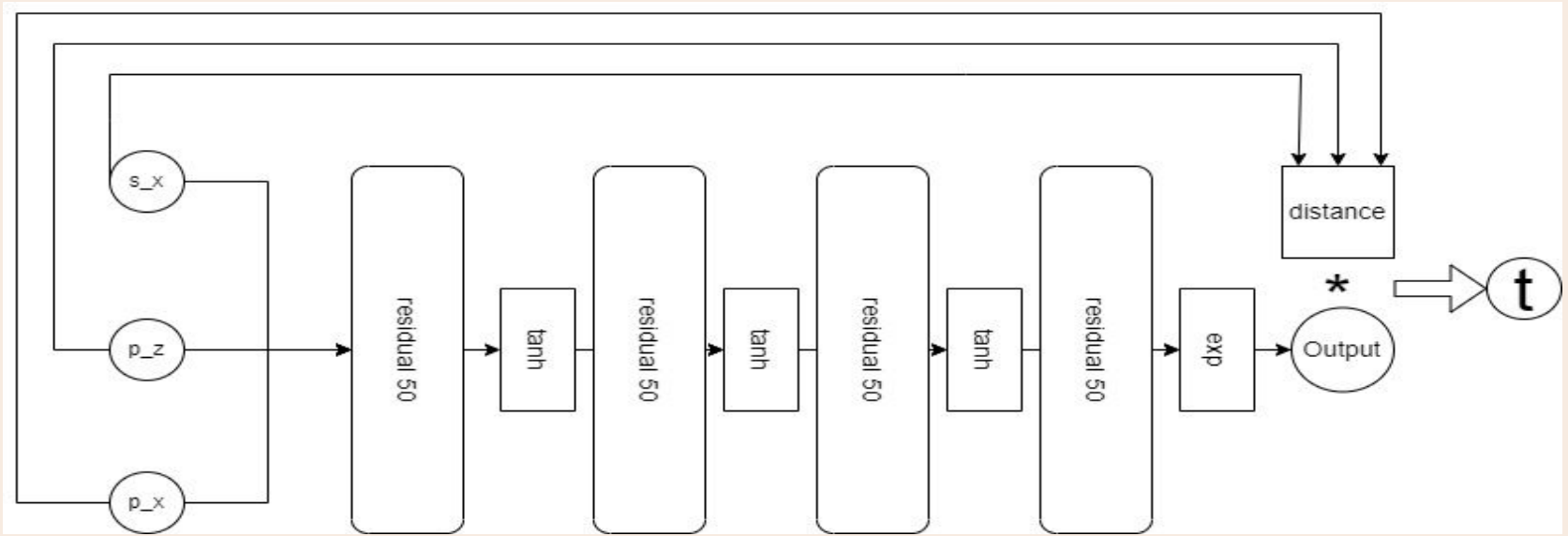
Advantages of factorization:

- introduces singularity
- satisfy boundary condition
- monotonous initial solution

$$\boldsymbol{L}(\boldsymbol{\Theta}) = \sum_{\mathbf{x}_{\boldsymbol{s}}}\sum_{\mathbf{x}_{\boldsymbol{r}}} (v(\mathbf{x}_{\boldsymbol{r}})^2 |\nabla_{\boldsymbol{r}}\tilde{\tau}(\mathbf{x}_{\boldsymbol{s}},\mathbf{x}_{\boldsymbol{r}})|^2 - 1)^2 \to \min_{\boldsymbol{\Theta}}$$

[Fomel 2009; Treister 2016; Smith 2020; Waheed 2021]

PINN Architecture



PINN Training

Distributed Training (1 epoch):

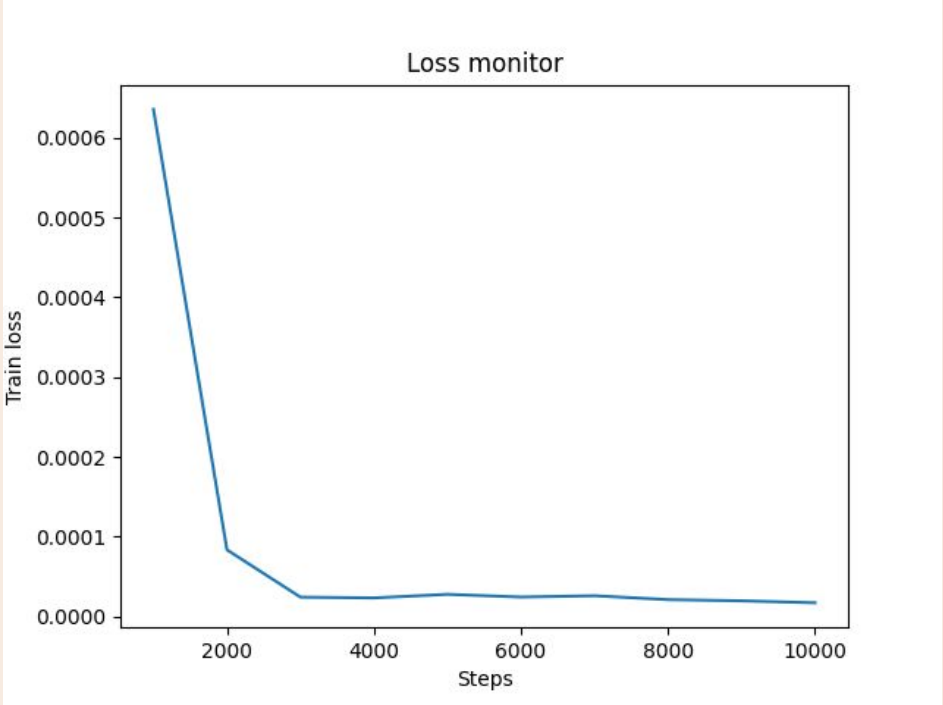
Training Devices	Time, seconds
1	531
8	110

Training optimization (3 epochs):

Amp Level	Time, seconds
-	249
O2	157
O3	145

- O2: Cast network to float16, keep batchnorm run in float32, using dynamic loss scale.
- O3: Cast network to float16, the batchnorm is also cast to float16, loss scale will not be used.
- auto: Set level to recommended level in different devices. Set level to O2 on GPU, set level to O3 on Ascend.

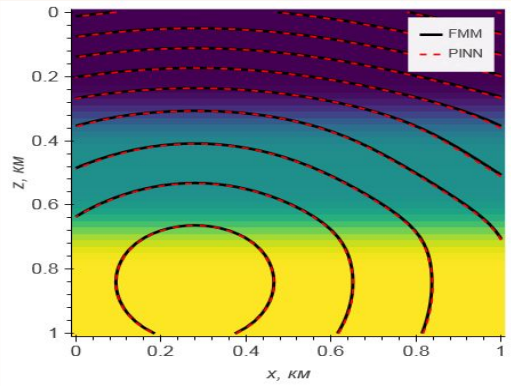
PINN Training

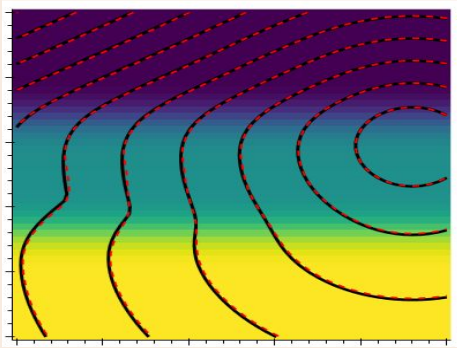


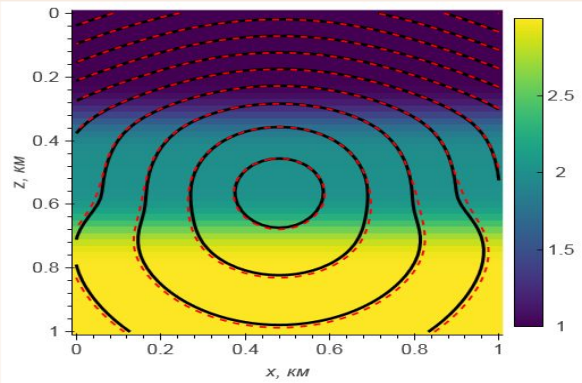
Testing: MAE: 0.03429582, MSE: 0.00135745

Similar to Tensorflow solution

Two-point traveltimes

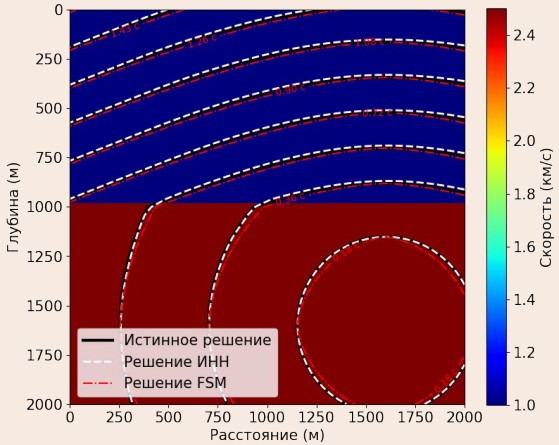






MAE < 0.8 мс

Result is similar to Tensorflow



Training on Ascend

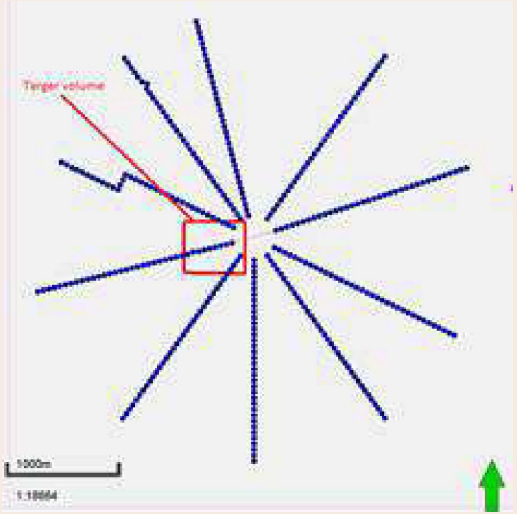
Porting NN Eikonal Solver from Tensorflow to Mindspore:

1) Split graph network into modules using nn.Cell from mindspore.
2) Create Eikonal solver from sub modules.
3) Calculate input and output dimensions for network layers like Dense etc.
4) Replace tensorflow operations (ops) to mindspore analogies.
5) Create training pipeline with mindspore.
    a) Set up dataset
    b) Define losses and metrics
    c) Set up optimizer
    d) Create mindspore Model instance

Microseismic Data Processing

Processing: coherent summation + mechanism

$$S(\mathbf{x}, t_0) = \sum_r A_r(\mathbf{M}) d(\mathbf{x}_r, t_0 + t(\mathbf{x}; \mathbf{x}_r))$$

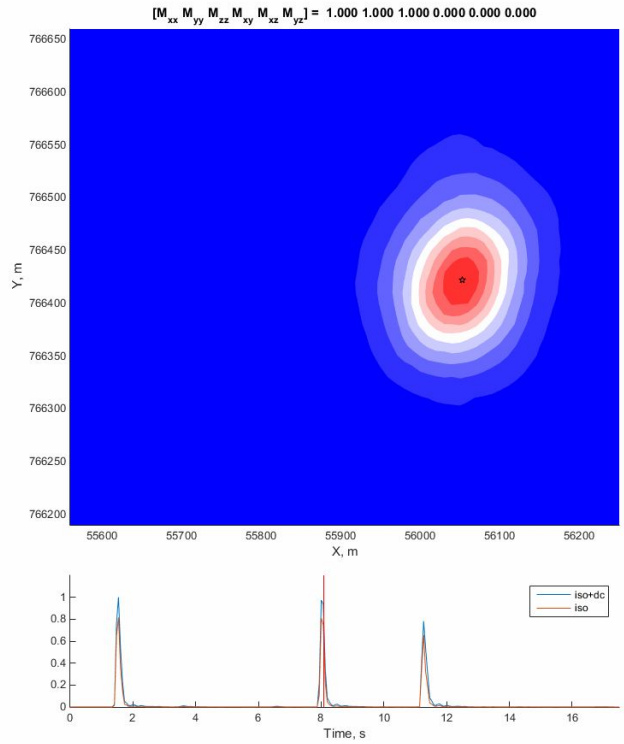


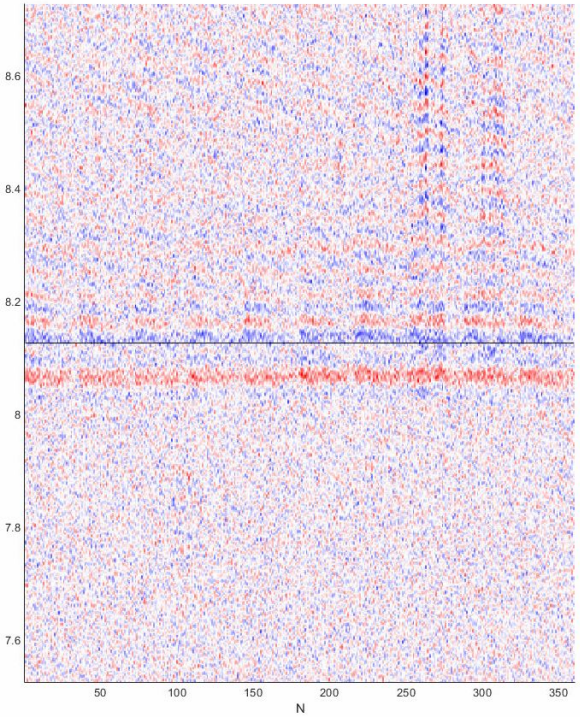
$$A_r(\mathbf{M}) = M_{ij} G_{3i,j}(\mathbf{x}_r, t_r)$$

Main computational steps:

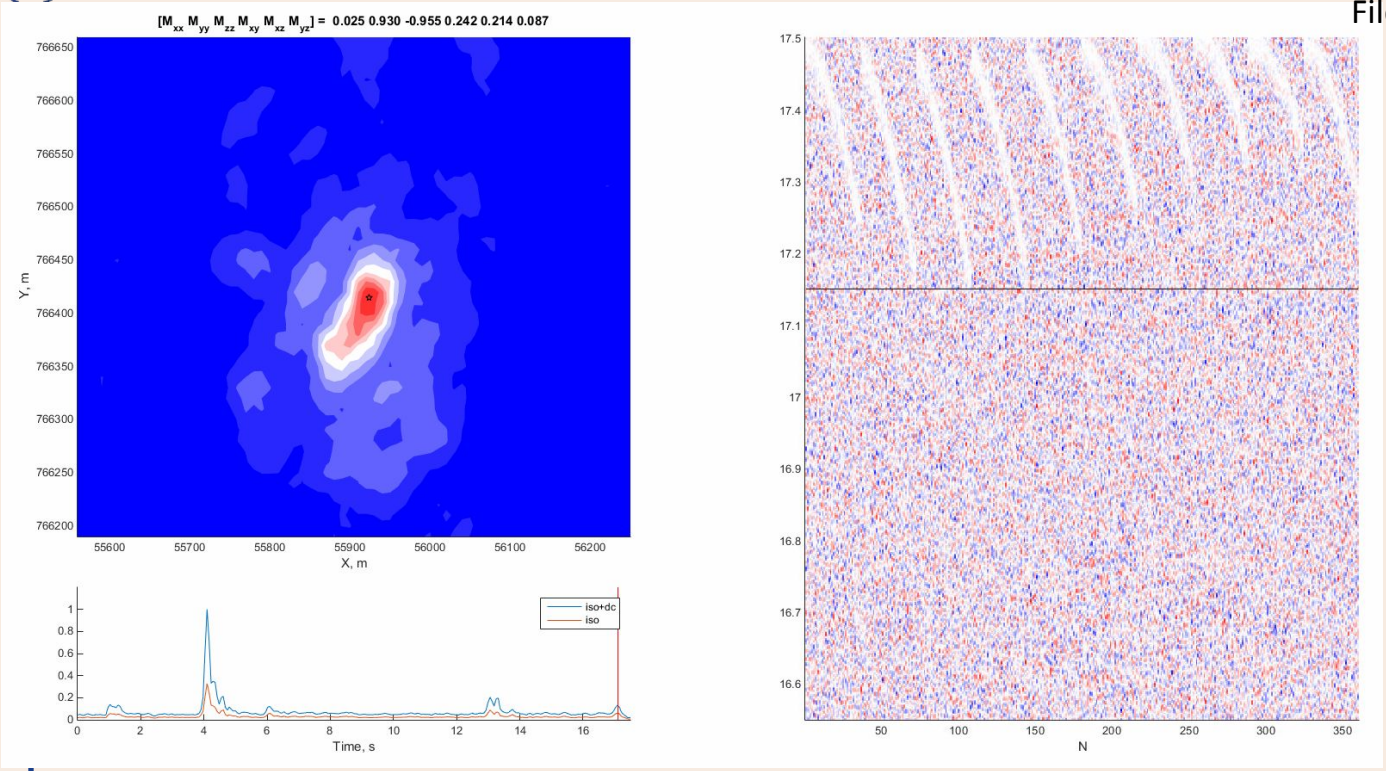
1. Traveltimes
2. Summation

File 2 17,5-35 c





File 1 0-17,5 c


[Mxx Myy Mzz Mxy Mxz Myz] = 0.025 0.930 -0.955 0.242 0.214 0.087
Y, m
X, m
iso+dc
iso
Time, s
N

Testing: Summation on Kunpeng + traveltimes on Acsend

Problems in using pre-trained NN (MINDR format) for computing traveltimes:

1. pre-trained NN can process only batches of fixed size (had to split the receiver array);
2. managed to run pre-trained NN on only one Ascend.

Conclusions on the project "AI in seismics, optimization on Ascend"

1. Seismic-wave traveltimes:
   Standard Eikonal solver – ported to **Kunpeng**

2. Seismic-wave traveltimes:
   Neural-Network Eikonal solver – ported to **Ascend**

3. Passive-seismic data processing:
   summation on **Kunpeng +** NN traveltimes on **Ascend**

Заключение: структура взаимодействия ИВМиМГ СО РАН - Хуавей

Инфраструктурные и образовательные проекты:

- Сотрудничество в создании вычислительной инфраструктуры на базе оборудования и ПО Хуавей в ИВМиМГ СО РАН
- Учебные курсы и семинары, грантовая поддержка студентов и аспирантов

Портирование суперкомпьютерных приложений ИВМиМГ СО РАН и других пользователей ССКЦ на ARM процессоры Kunpeng, разработка новых приложений

Портирование приложений ИИ и МО на процессоры Ascend, разработка новых приложений