# 基于DPCT的序列比对软件迁移与性能评估

2023

intel.

# Contents

01

背景介绍

02

迁移过程

03

实验结果

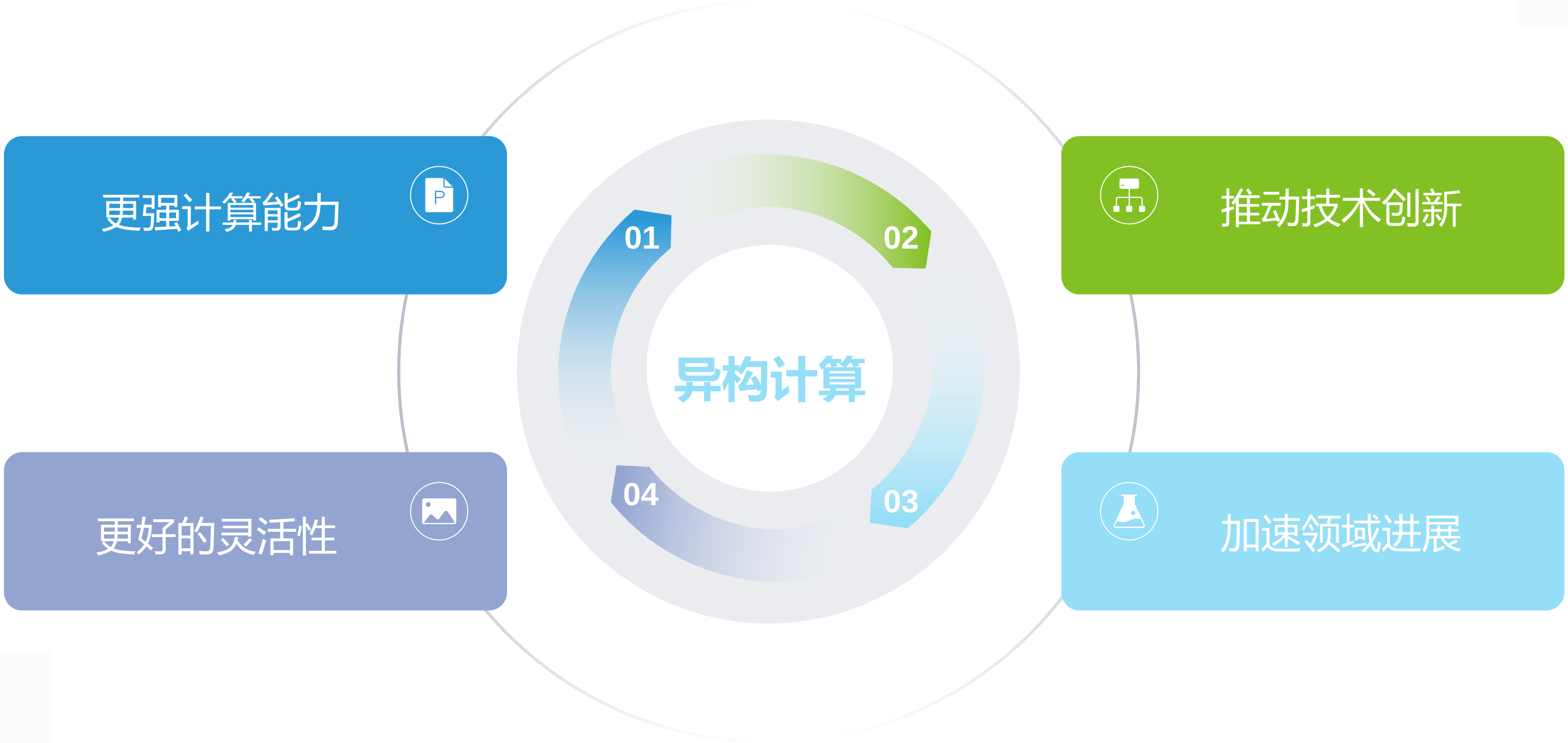
04

总结

/01

# 背景介绍

# 异构计算在HPC领域中的优势



异构计算

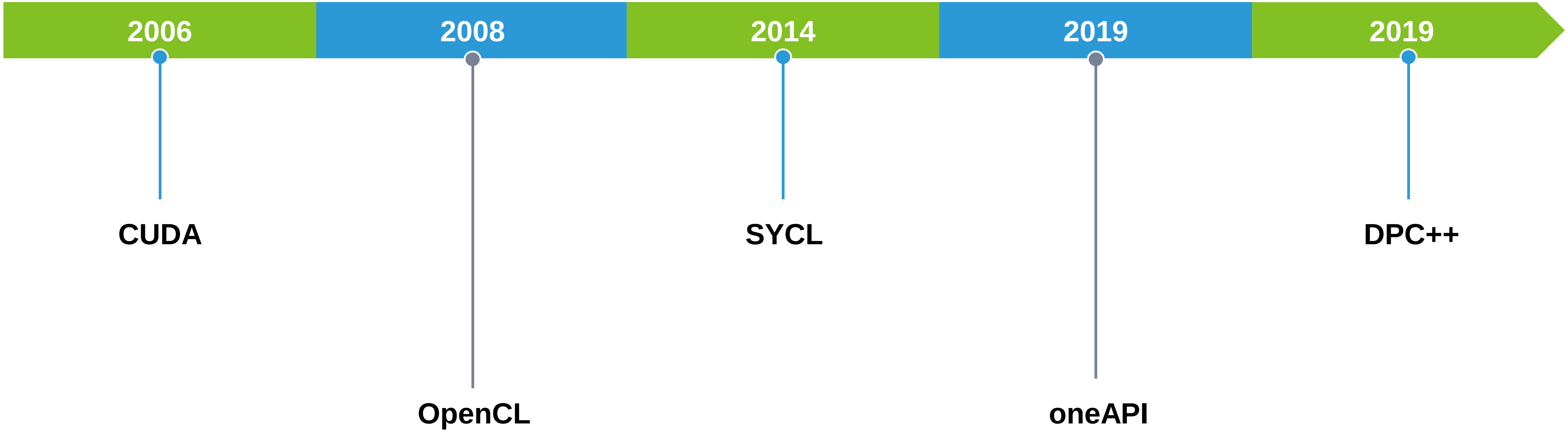
01 更强计算能力

02 推动技术创新

03 加速领域进展

04 更好的灵活性

# 异构编程架构的发展



2006

CUDA

2008

OpenCL

2014

SYCL

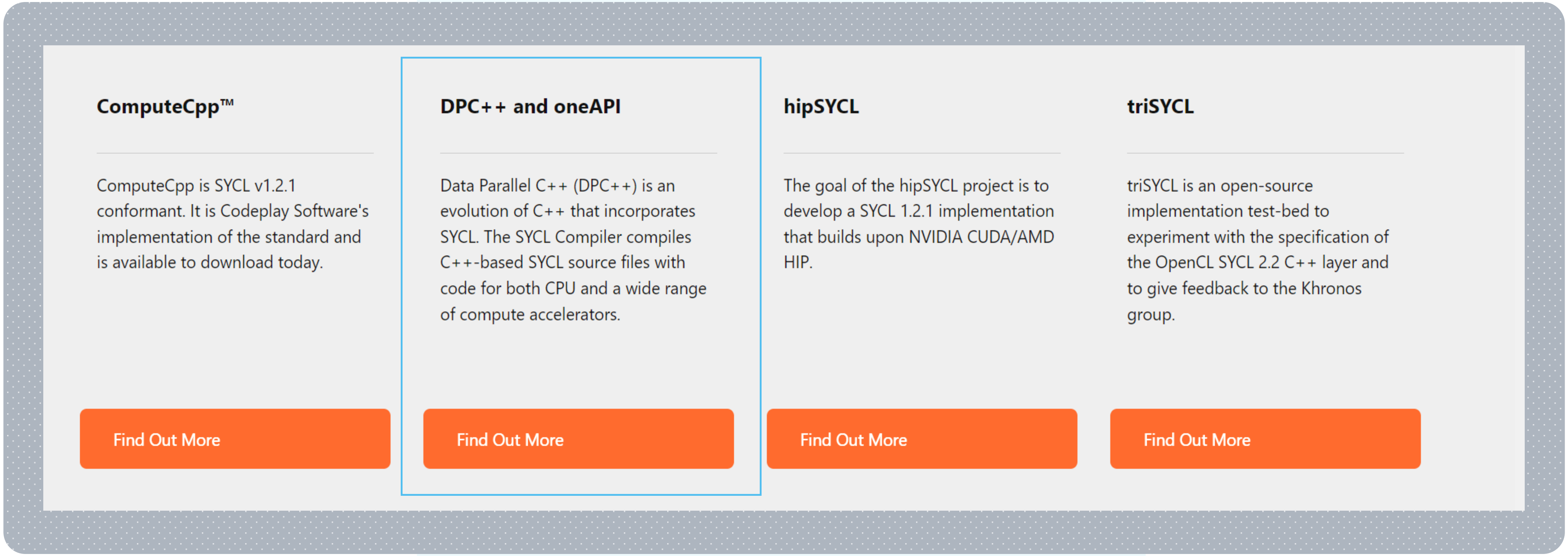
2019

oneAPI

2019

DPC++

# SYCL的主要实现



### ComputeCpp™

ComputeCpp is SYCL v1.2.1 conformant. It is Codeplay Software's implementation of the standard and is available to download today.

### DPC++ and oneAPI

Data Parallel C++ (DPC++) is an evolution of C++ that incorporates SYCL. The SYCL Compiler compiles C++-based SYCL source files with code for both CPU and a wide range of compute accelerators.

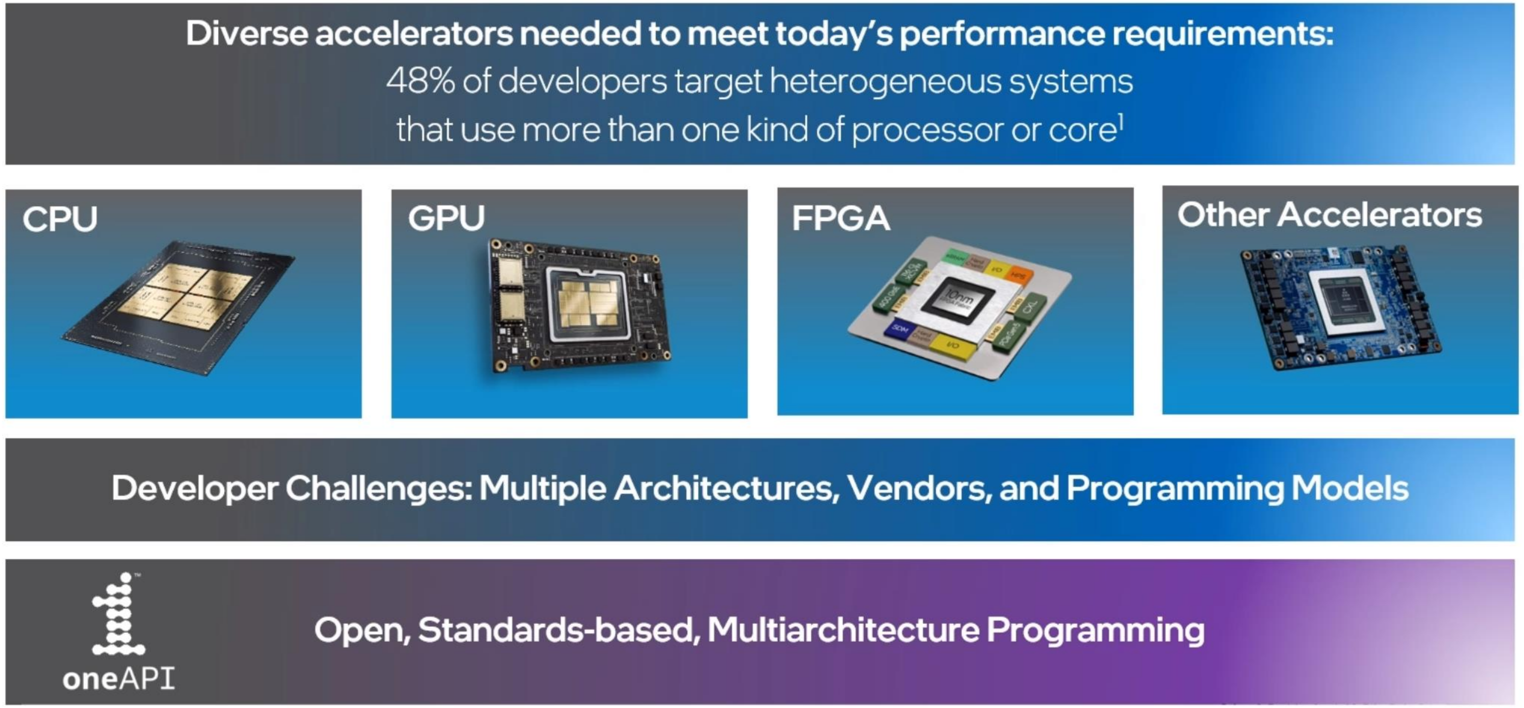
### hipSYCL

The goal of the hipSYCL project is to develop a SYCL 1.2.1 implementation that builds upon NVIDIA CUDA/AMD HIP.

### triSYCL

triSYCL is an open-source implementation test-bed to experiment with the specification of the OpenCL SYCL 2.2 C++ layer and to give feedback to the Khronos group.

# DPC++的优势



**Diverse accelerators needed to meet today's performance requirements:**
48% of developers target heterogeneous systems
that use more than one kind of processor or core[1]

CPU
GPU
FPGA
Other Accelerators

**Developer Challenges: Multiple Architectures, Vendors, and Programming Models**

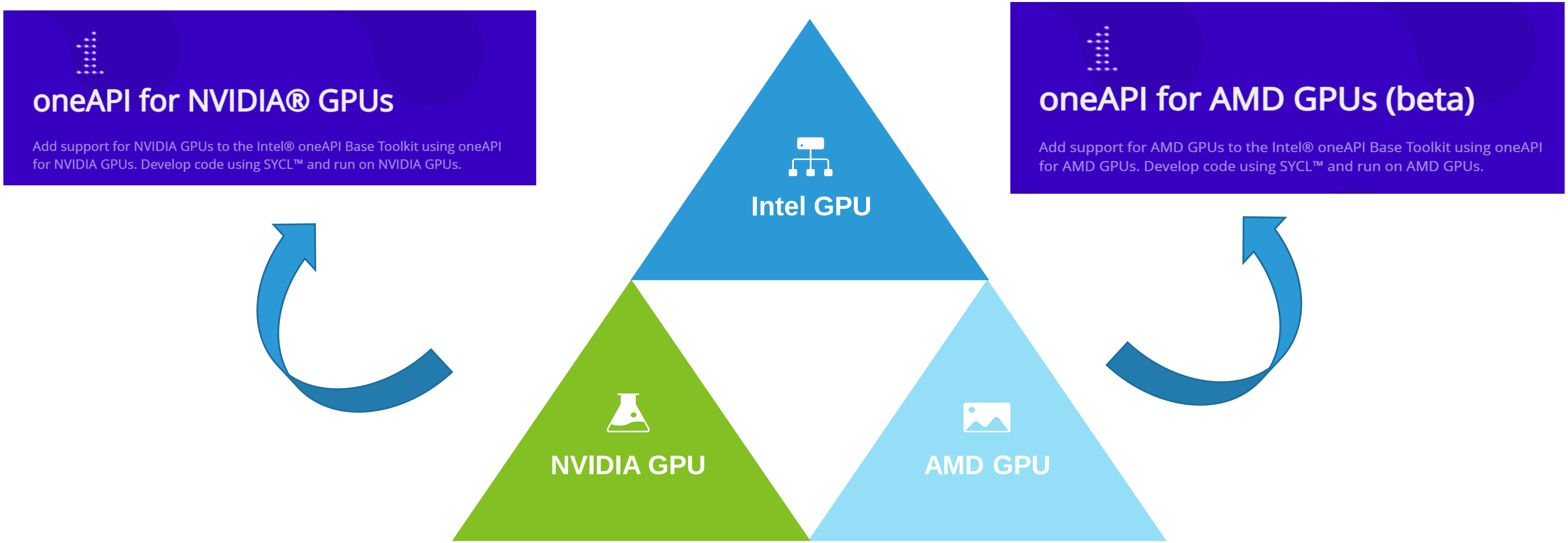
oneAPI

**Open, Standards-based, Multiarchitecture Programming**

## 跨架构性

可以在不同架构上的GPU，CPU，AI加速器上编写程序并且运行

# 跨架构性

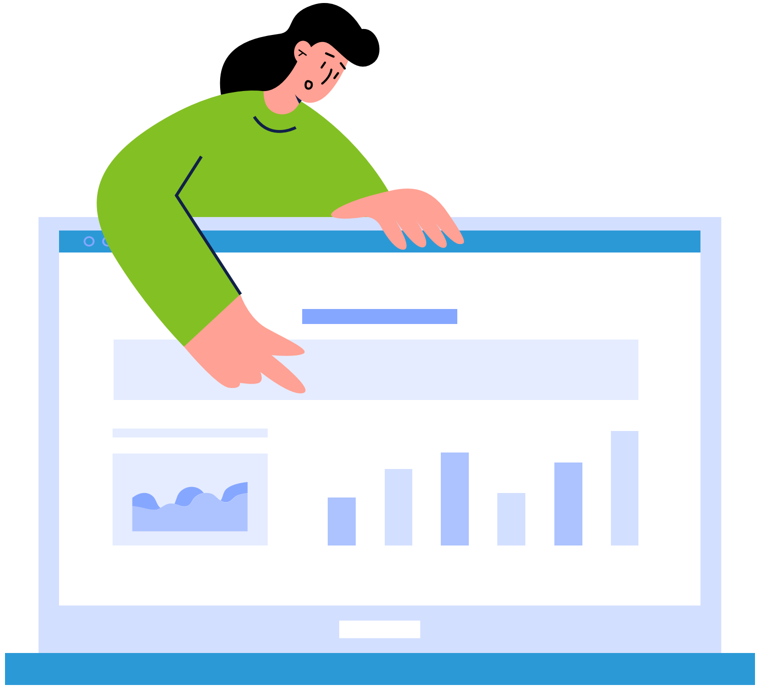


## oneAPI for NVIDIA® GPUs

Add support for NVIDIA GPUs to the Intel® oneAPI Base Toolkit using oneAPI for NVIDIA GPUs. Develop code using SYCL™ and run on NVIDIA GPUs.

## oneAPI for AMD GPUs (beta)

Add support for AMD GPUs to the Intel® oneAPI Base Toolkit using oneAPI for AMD GPUs. Develop code using SYCL™ and run on AMD GPUs.



## 灵活性

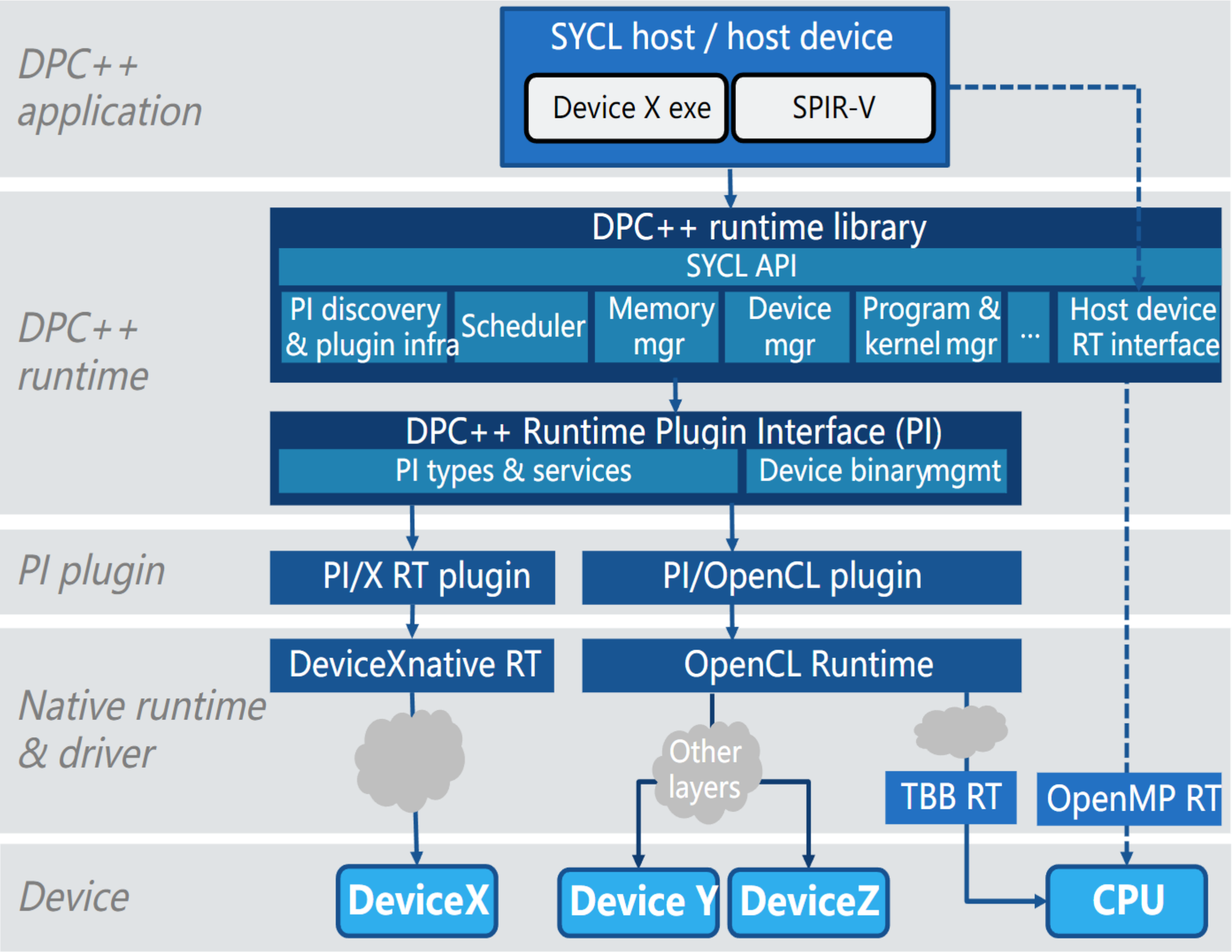
- DPC++ 设计时考虑了可移植性，使得编写的代码可以在不同类型的硬件上运行

## 易开发性

- DPC++基于SYCL标准实现，SYCL标准是C++的扩展，没有特定的针对某种特定架构，有C++开发经验可以很快上手

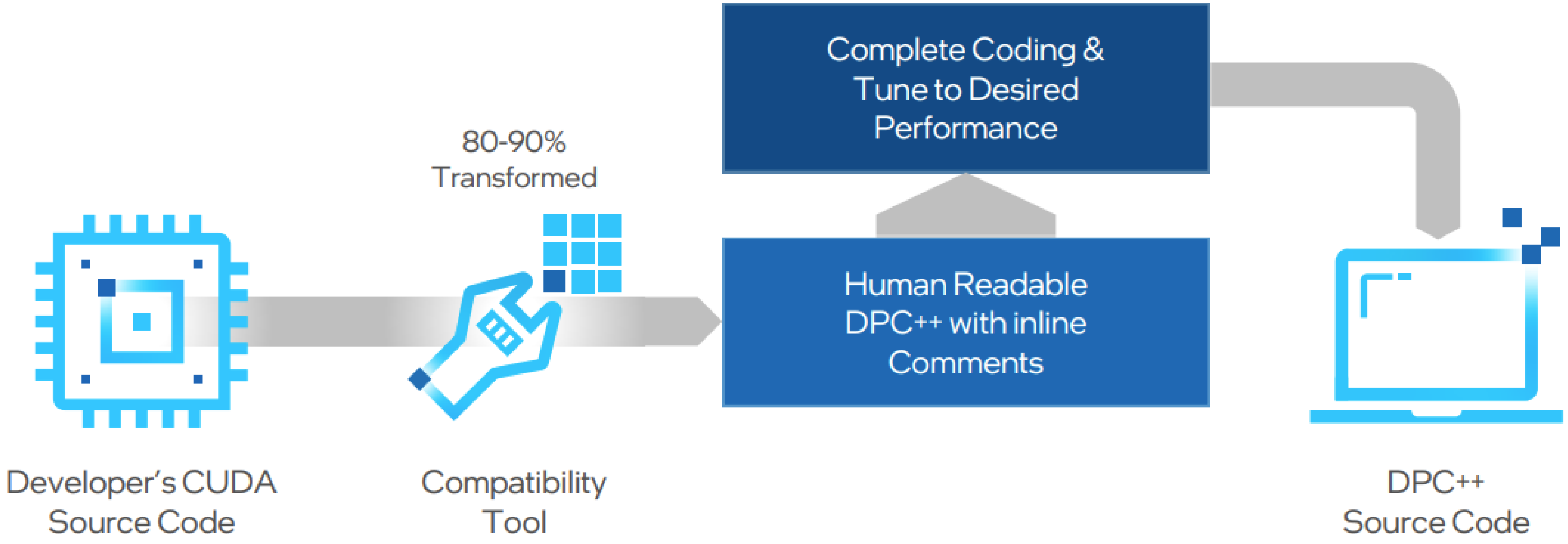
# DPC++

Complier



# DPCT

## Intel DPC ++ Compatibility Tool Usage Flow



Complete Coding & Tune to Desired Performance

80-90% Transformed

Human Readable DPC++ with inline Comments

Developer's CUDA Source Code

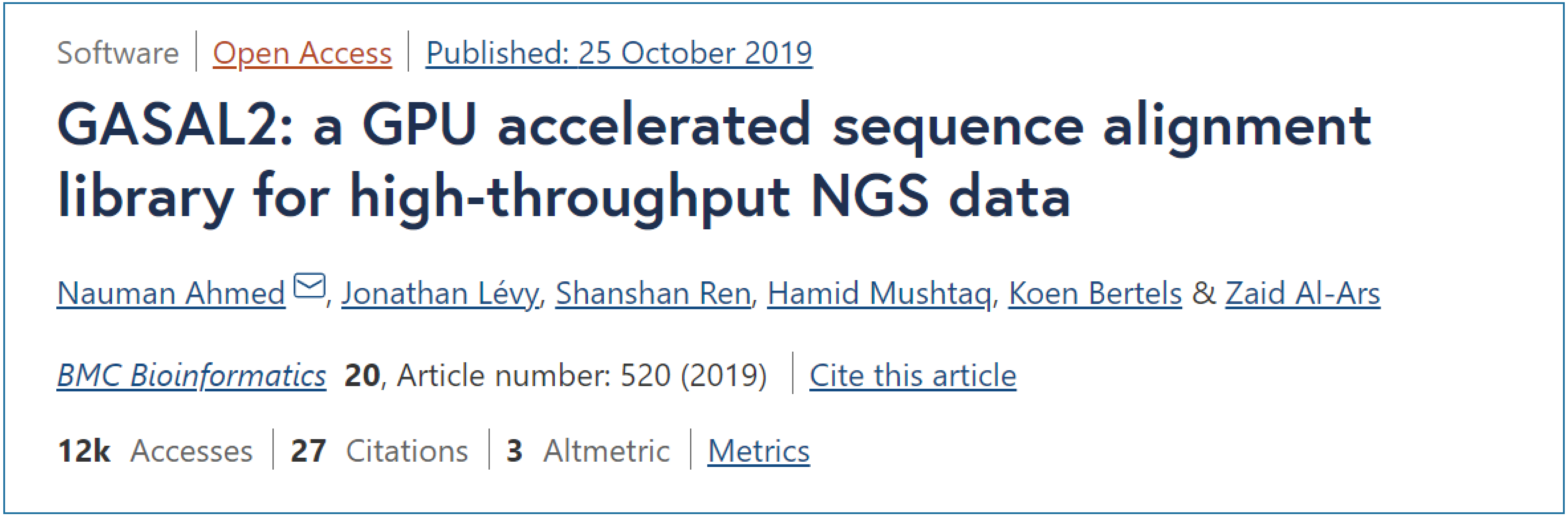
Compatibility Tool

DPC++ Source Code

# /02

# 迁移过程

# GASAL2



Software | Open Access | Published: 25 October 2019

## GASAL2: a GPU accelerated sequence alignment library for high-throughput NGS data

Nauman Ahmed ✉, Jonathan Lévy, Shanshan Ren, Hamid Mushtaq, Koen Bertels & Zaid Al-Ars

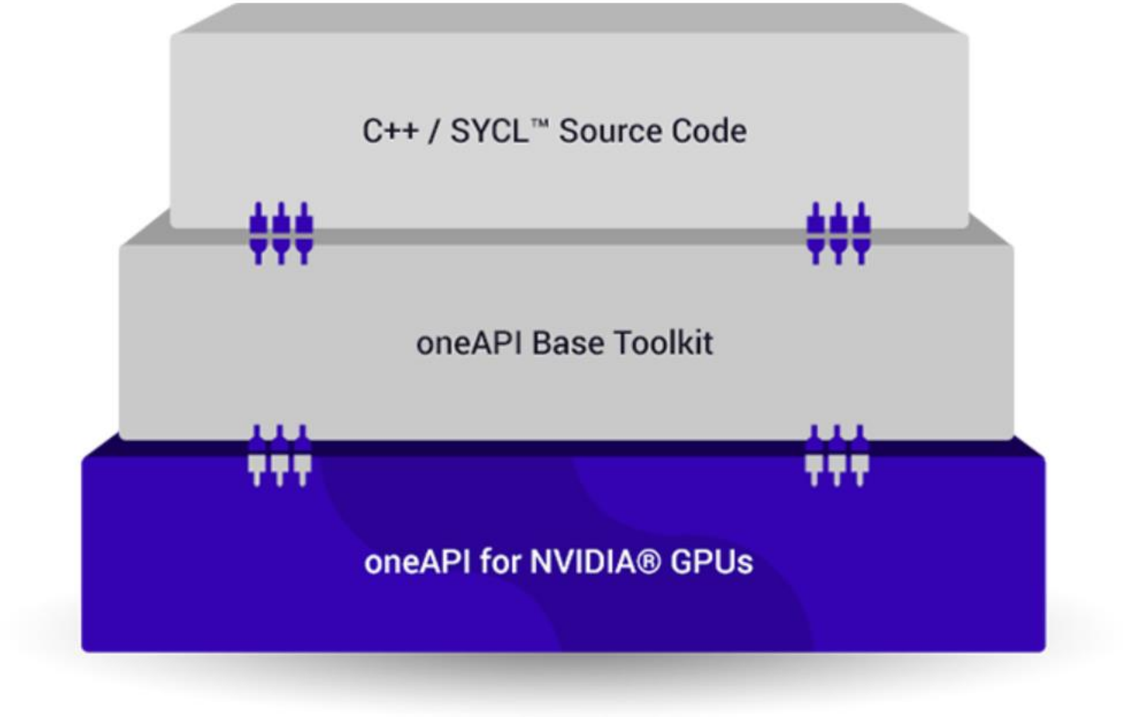
*BMC Bioinformatics* **20**, Article number: 520 (2019) | Cite this article

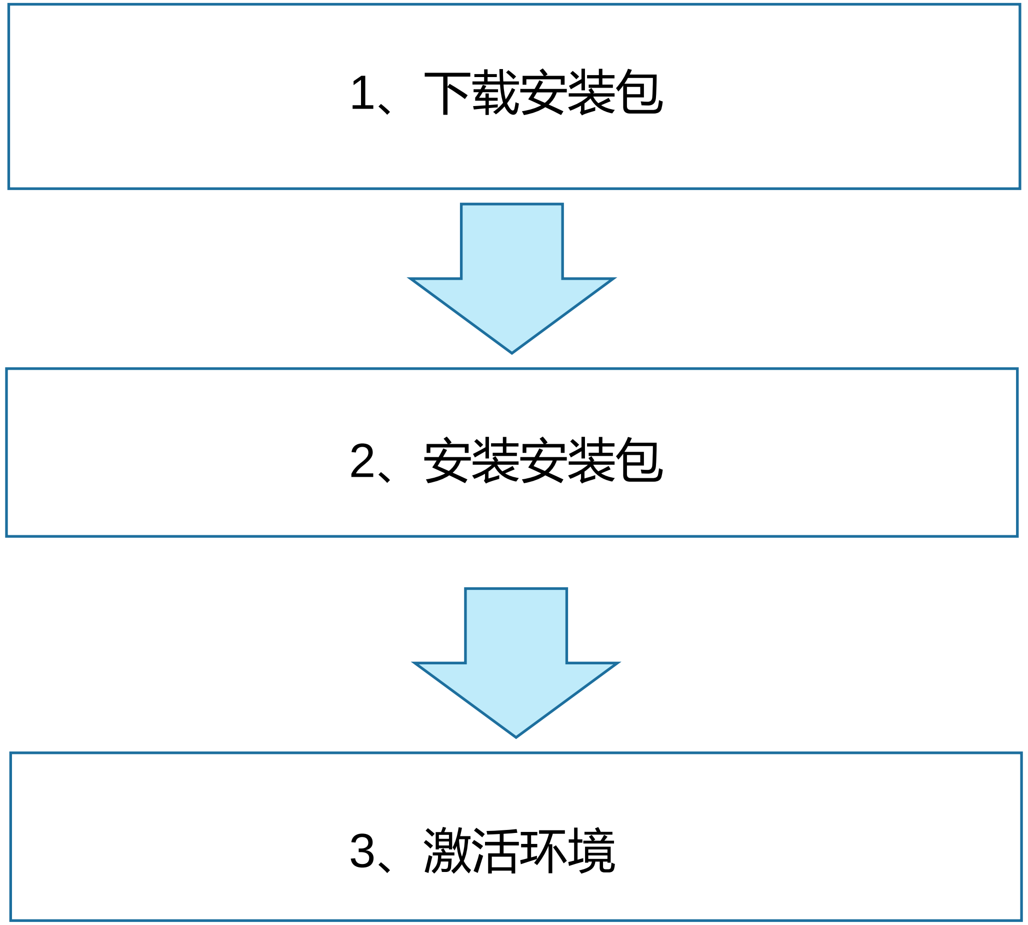
**12k** Accesses | **27** Citations | **3** Altmetric | Metrics

https://github.com/nahmedraja/GASAL2

# 实验环境

## oneAPI for NVIDIA® GPUs





1、下载安装包

2、安装安装包

3、激活环境

# 实验环境

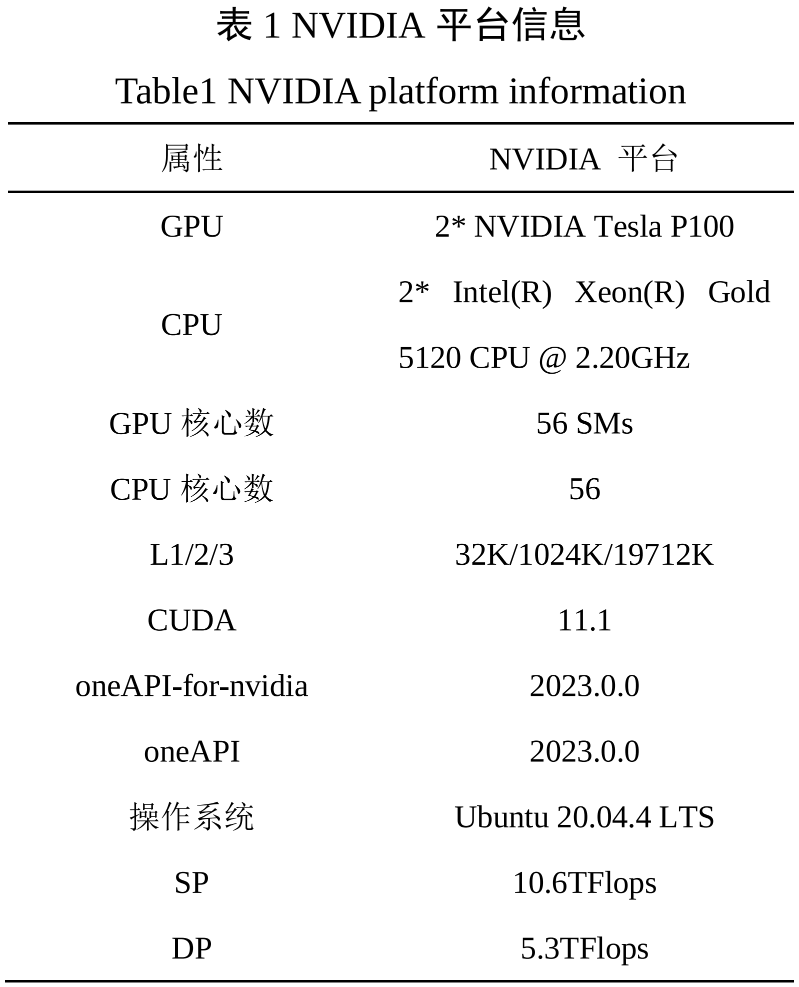


表 1 NVIDIA 平台信息

Table1 NVIDIA platform information

| 属性 | NVIDIA 平台 |
|---|---|
| GPU | 2* NVIDIA Tesla P100 |
| CPU | 2* Intel(R) Xeon(R) Gold 5120 CPU @ 2.20GHz |
| GPU 核心数 | 56 SMs |
| CPU 核心数 | 56 |
| L1/2/3 | 32K/1024K/19712K |
| CUDA | 11.1 |
| oneAPI-for-nvidia | 2023.0.0 |
| oneAPI | 2023.0.0 |
| 操作系统 | Ubuntu 20.04.4 LTS |
| SP | 10.6TFlops |
| DP | 5.3TFlops |

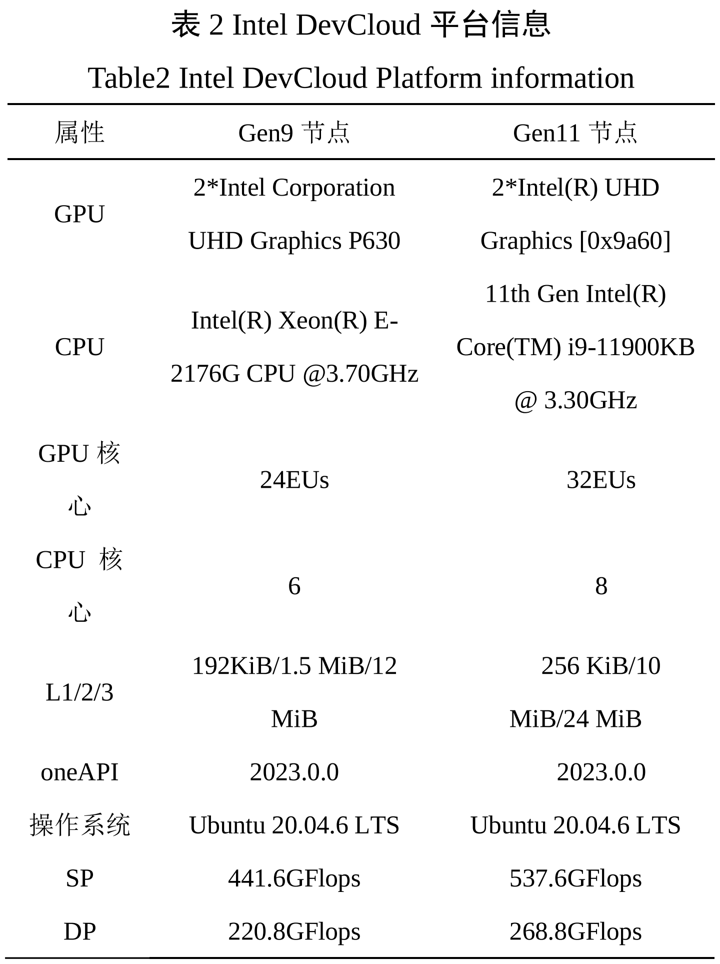


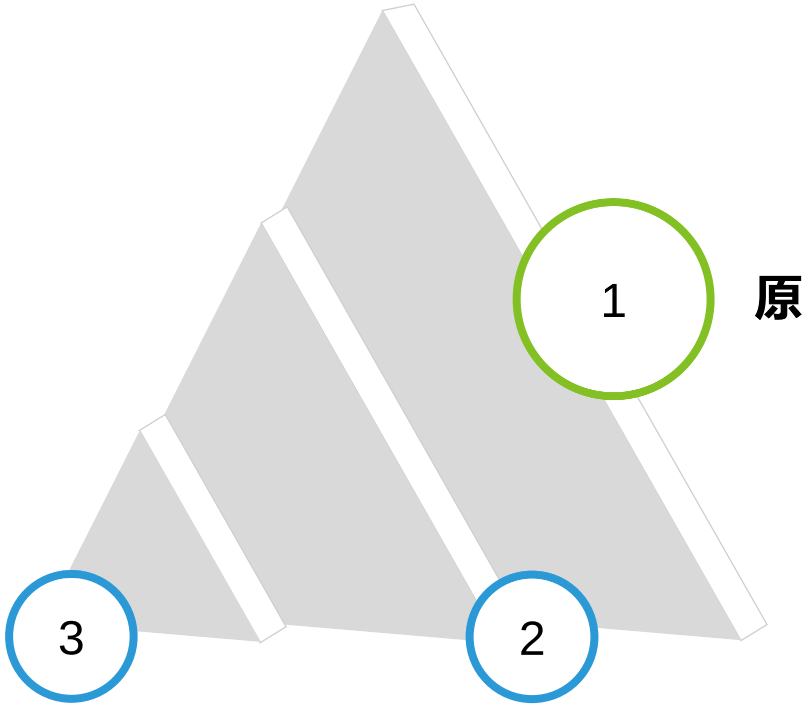
表 2 Intel DevCloud 平台信息

Table2 Intel DevCloud Platform information

| 属性 | Gen9 节点 | Gen11 节点 |
|---|---|---|
| GPU | 2*Intel Corporation UHD Graphics P630 | 2*Intel(R) UHD Graphics [0x9a60] |
| CPU | Intel(R) Xeon(R) E-2176G CPU @3.70GHz | 11th Gen Intel(R) Core(TM) i9-11900KB @ 3.30GHz |
| GPU 核心 | 24EUs | 32EUs |
| CPU 核心 | 6 | 8 |
| L1/2/3 | 192KiB/1.5 MiB/12 MiB | 256 KiB/10 MiB/24 MiB |
| oneAPI | 2023.0.0 | 2023.0.0 |
| 操作系统 | Ubuntu 20.04.6 LTS | Ubuntu 20.04.6 LTS |
| SP | 441.6GFlops | 537.6GFlops |
| DP | 220.8GFlops | 268.8GFlops |

# DPCT迁移流程

**DPCT迁移流程**



**原程序进行编译**

**拦截获取相关信息，生成相关文件**

**进行转换API**

# DPCT迁移流程

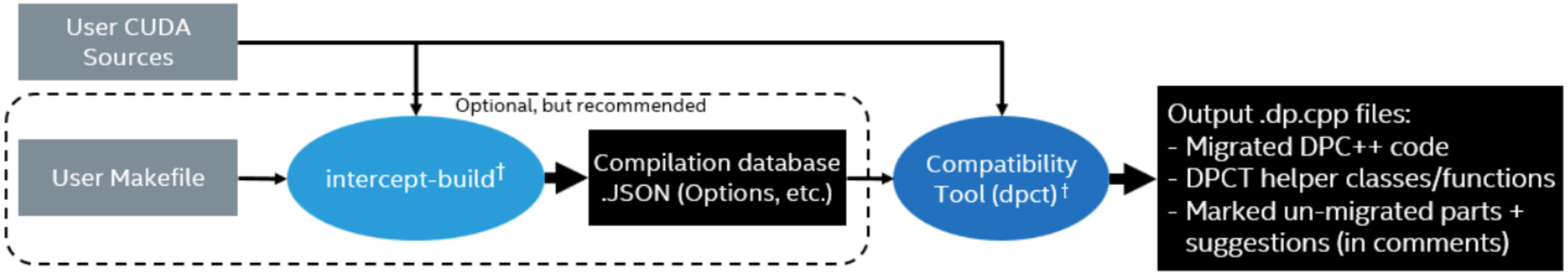
一、运行intercept-build脚本，它能自动跟踪编译时相关需要修改的信息

```
intercept-build make
```

```
compile_commands.json
```

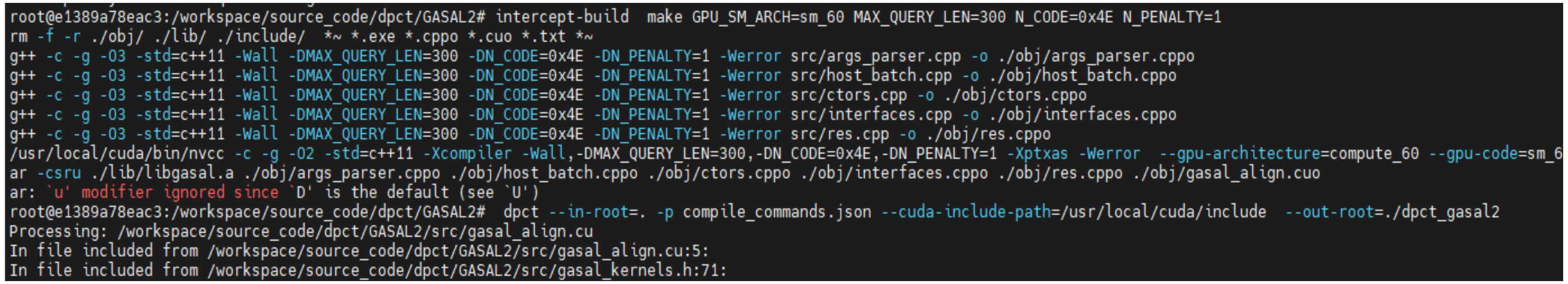
二、运行DPCT

```
dpct -p ./ --in-root=./ --out-root=output *.cu
```


User CUDA Sources
Optional, but recommended
User Makefile
intercept-build†
Compilation database .JSON (Options, etc.)
Compatibility Tool (dpct)†
Output .dp.cpp files:
- Migrated DPC++ code
- DPCT helper classes/functions
- Marked un-migrated parts + suggestions (in comments)

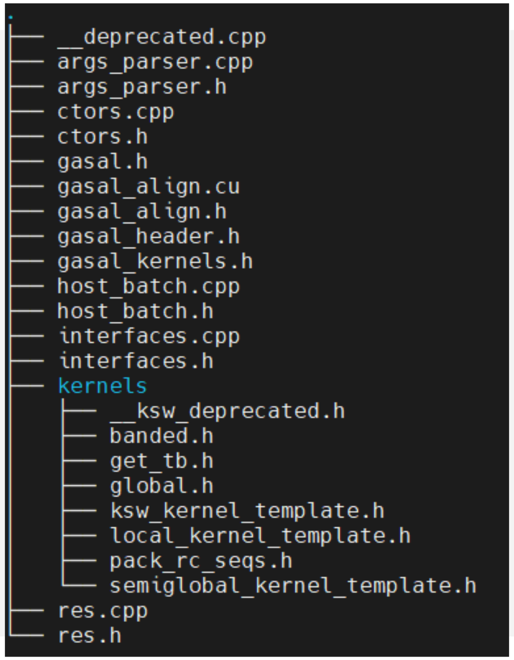
† Certain CUDA language header files may need to be accessible to the Intel® DPC++ Compatibility Tool

# 迁移过程

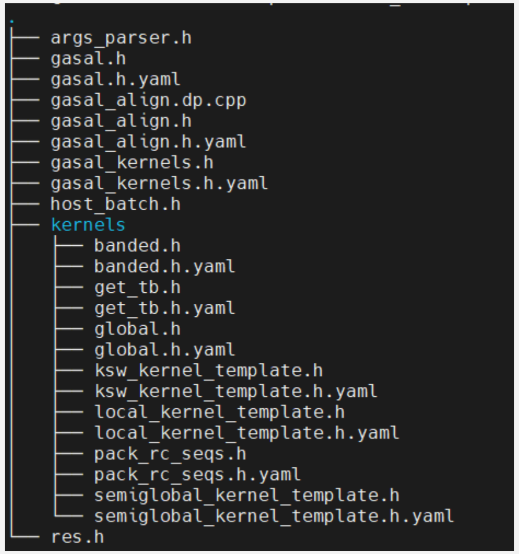


```
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2# intercept-build  make GPU_SM_ARCH=sm_60 MAX_QUERY_LEN=300 N_CODE=0x4E N_PENALTY=1
rm -f -r ./obj/ ./lib/ ./include/  *~ *.exe *.cppo *.cuo *.txt *~
g++ -c -g -O3 -std=c++11 -Wall -DMAX_QUERY_LEN=300 -DN_CODE=0x4E -DN_PENALTY=1 -Werror src/args_parser.cpp -o ./obj/args_parser.cppo
g++ -c -g -O3 -std=c++11 -Wall -DMAX_QUERY_LEN=300 -DN_CODE=0x4E -DN_PENALTY=1 -Werror src/host_batch.cpp -o ./obj/host_batch.cppo
g++ -c -g -O3 -std=c++11 -Wall -DMAX_QUERY_LEN=300 -DN_CODE=0x4E -DN_PENALTY=1 -Werror src/ctors.cpp -o ./obj/ctors.cppo
g++ -c -g -O3 -std=c++11 -Wall -DMAX_QUERY_LEN=300 -DN_CODE=0x4E -DN_PENALTY=1 -Werror src/interfaces.cpp -o ./obj/interfaces.cppo
g++ -c -g -O3 -std=c++11 -Wall -DMAX_QUERY_LEN=300 -DN_CODE=0x4E -DN_PENALTY=1 -Werror src/res.cpp -o ./obj/res.cppo
/usr/local/cuda/bin/nvcc -c -g -O2 -std=c++11 -Xcompiler -Wall,-DMAX_QUERY_LEN=300,-DN_CODE=0x4E,-DN_PENALTY=1 -Xptxas -Werror  --gpu-architecture=compute_60 --gpu-code=sm_6
ar -csru ./lib/libgasal.a ./obj/args_parser.cppo ./obj/host_batch.cppo ./obj/ctors.cppo ./obj/interfaces.cppo ./obj/res.cppo ./obj/gasal_align.cuo
ar: `u' modifier ignored since `D' is the default (see `U')
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2#  dpct --in-root=. -p compile_commands.json --cuda-include-path=/usr/local/cuda/include  --out-root=./dpct_gasal2
Processing: /workspace/source_code/dpct/GASAL2/src/gasal_align.cu
In file included from /workspace/source_code/dpct/GASAL2/src/gasal_align.cu:5:
In file included from /workspace/source_code/dpct/GASAL2/src/gasal_kernels.h:71:
```

- **操作简单**
- **自动转换**



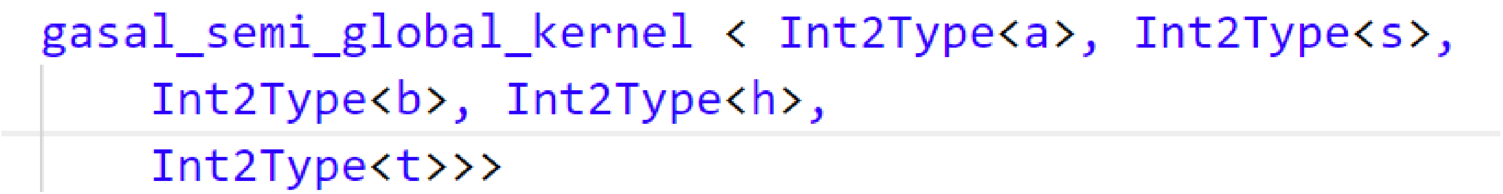
```
.
├── __deprecated.cpp
├── args_parser.cpp
├── args_parser.h
├── ctors.cpp
├── ctors.h
├── gasal.h
├── gasal_align.cu
├── gasal_align.h
├── gasal_header.h
├── gasal_kernels.h
├── host_batch.cpp
├── host_batch.h
├── interfaces.cpp
├── interfaces.h
├── kernels
│   ├── __ksw_deprecated.h
│   ├── banded.h
│   ├── get_tb.h
│   ├── global.h
│   ├── ksw_kernel_template.h
│   ├── local_kernel_template.h
│   ├── pack_rc_seqs.h
│   └── semiglobal_kernel_template.h
├── res.cpp
└── res.h
```



```
.
├── args_parser.h
├── gasal.h
├── gasal.h.yaml
├── gasal_align.dp.cpp
├── gasal_align.h
├── gasal_align.h.yaml
├── gasal_kernels.h
├── gasal_kernels.h.yaml
├── host_batch.h
├── kernels
│   ├── banded.h
│   ├── banded.h.yaml
│   ├── get_tb.h
│   ├── get_tb.h.yaml
│   ├── global.h
│   ├── global.h.yaml
│   ├── ksw_kernel_template.h
│   ├── ksw_kernel_template.h.yaml
│   ├── local_kernel_template.h
│   ├── local_kernel_template.h.yaml
│   ├── pack_rc_seqs.h
│   ├── pack_rc_seqs.h.yaml
│   ├── semiglobal_kernel_template.h
│   └── semiglobal_kernel_template.h.yaml
└── res.h
```

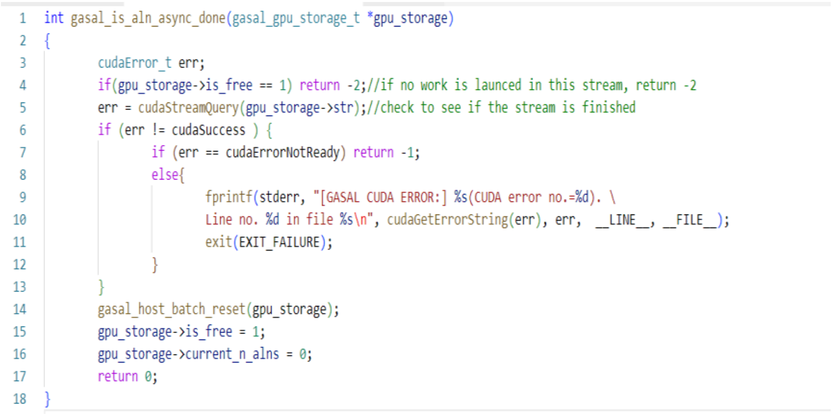
# 迁移过程

- 在Int2Type这种语法下尖括号匹配转换时会多一个尖括号

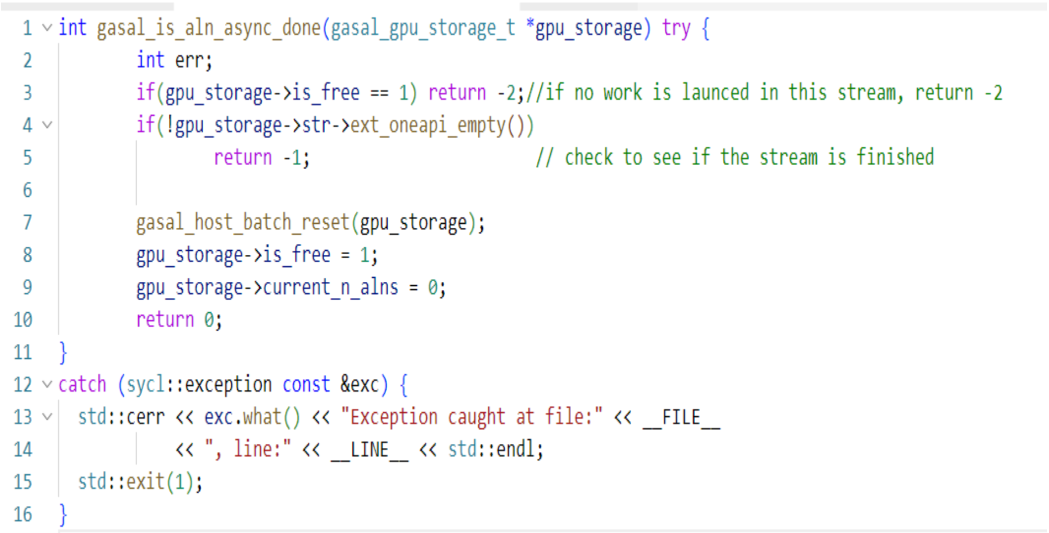


```
gasal_semi_global_kernel < Int2Type<a>, Int2Type<s>,
    Int2Type<b>, Int2Type<h>,
    Int2Type<t>>>
```

- 有些关于设备状态或者队列信息的API转换需要自己去查阅相关资料去手动实现



```
int gasal_is_aln_async_done(gasal_gpu_storage_t *gpu_storage)
{
        cudaError_t err;
        if(gpu_storage->is_free == 1) return -2;//if no work is launced in this stream, return -2
        err = cudaStreamQuery(gpu_storage->str);//check to see if the stream is finished
        if (err != cudaSuccess ) {
                if (err == cudaErrorNotReady) return -1;
                else{
                        fprintf(stderr, "[GASAL CUDA ERROR:] %s(CUDA error no.=%d). \
                        Line no. %d in file %s\n", cudaGetErrorString(err), err,  __LINE__, __FILE__);
                        exit(EXIT_FAILURE);
                }
        }
        gasal_host_batch_reset(gpu_storage);
        gpu_storage->is_free = 1;
        gpu_storage->current_n_alns = 0;
        return 0;
}
```



```
int gasal_is_aln_async_done(gasal_gpu_storage_t *gpu_storage) try {
        int err;
        if(gpu_storage->is_free == 1) return -2;//if no work is launced in this stream, return -2
        if(!gpu_storage->str->ext_oneapi_empty())
                return -1;                      // check to see if the stream is finished

        gasal_host_batch_reset(gpu_storage);
        gpu_storage->is_free = 1;
        gpu_storage->current_n_alns = 0;
        return 0;
}
catch (sycl::exception const &exc) {
  std::cerr << exc.what() << "Exception caught at file:" << __FILE__
            << ", line:" << __LINE__ << std::endl;
  std::exit(1);
}
```

# 编译文件

DPC++采用C++17标准,采用跨架构GPU编译运行的时候只需修改编译器就可

Intel平台：icpx -fsycl -fsycl-targets=spir64_gen

Nvidia平台：clang++ -fsycl -fsycl-targets=nvptx64-nvidia-cuda

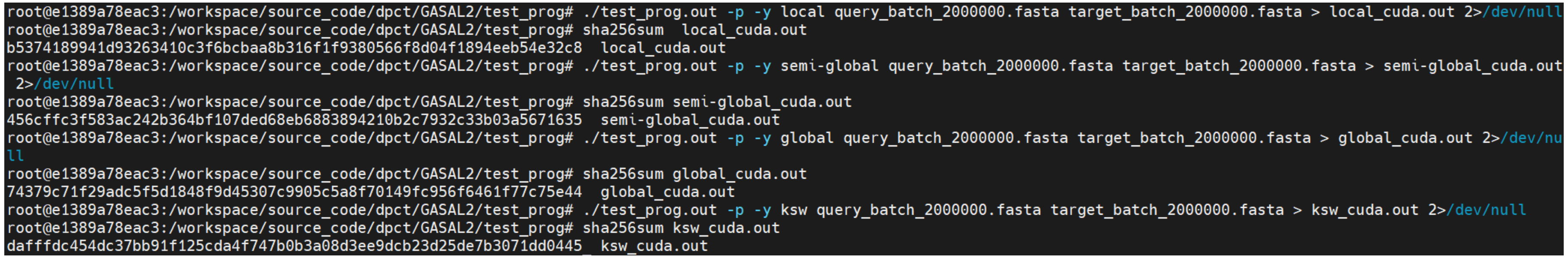
-arch=compute_60;-code=sm_60

-Xsycl-target-backend=nvptx64-nvidia-cuda --cuda-gpu-arch=sm_60

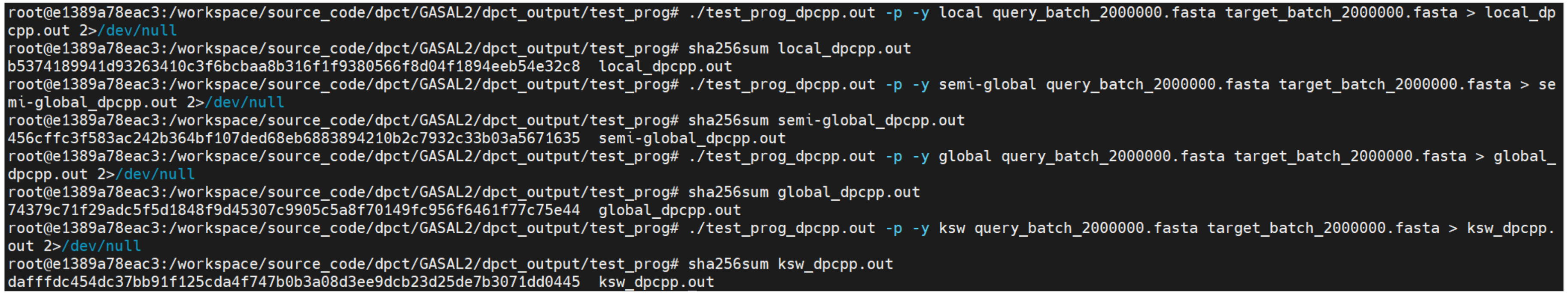
# /03

# 实验结果

# 实验结果

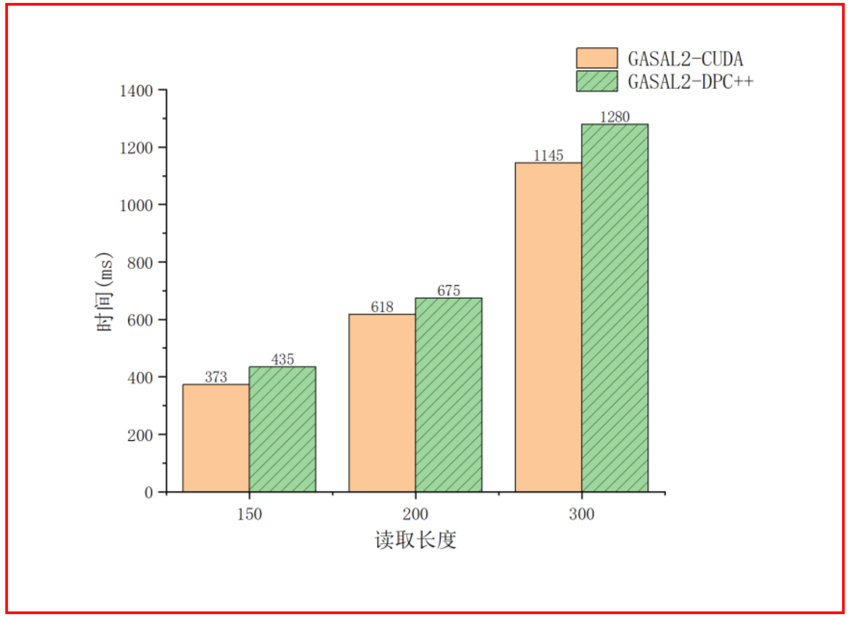


```
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# ./test_prog.out -p -y local query_batch_2000000.fasta target_batch_2000000.fasta > local_cuda.out 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# sha256sum  local_cuda.out
b5374189941d93263410c3f6bcbaa8b316f1f9380566f8d04f1894eeb54e32c8  local_cuda.out
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# ./test_prog.out -p -y semi-global query_batch_2000000.fasta target_batch_2000000.fasta > semi-global_cuda.out
 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# sha256sum semi-global_cuda.out
456cffc3f583ac242b364bf107ded68eb6883894210b2c7932c33b03a5671635  semi-global_cuda.out
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# ./test_prog.out -p -y global query_batch_2000000.fasta target_batch_2000000.fasta > global_cuda.out 2>/dev/nu
ll
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# sha256sum global_cuda.out
74379c71f29adc5f5d1848f9d45307c9905c5a8f70149fc956f6461f77c75e44  global_cuda.out
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# ./test_prog.out -p -y ksw query_batch_2000000.fasta target_batch_2000000.fasta > ksw_cuda.out 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/test_prog# sha256sum ksw_cuda.out
dafffdc454dc37bb91f125cda4f747b0b3a08d3ee9dcb23d25de7b3071dd0445  ksw_cuda.out
```

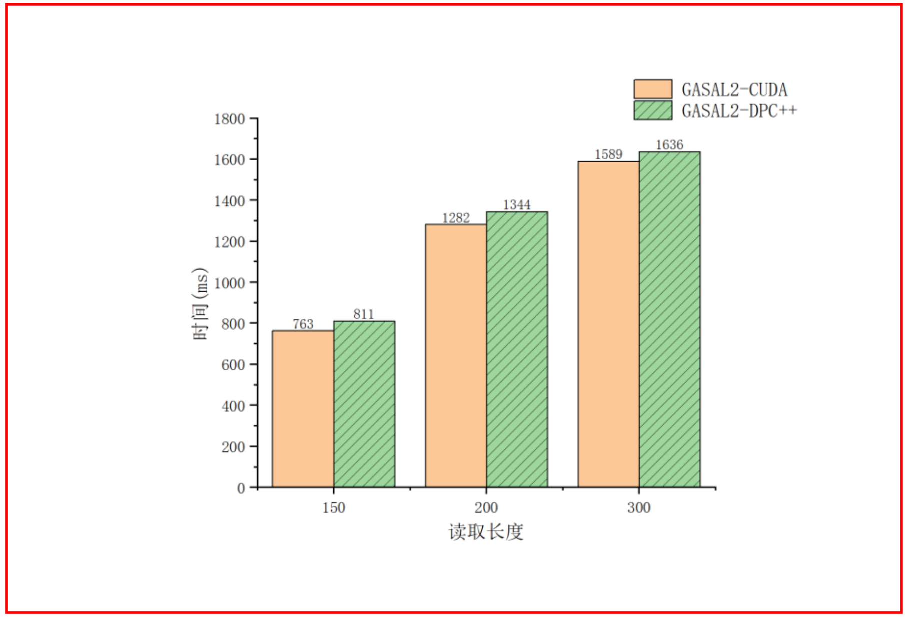


```
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# ./test_prog_dpcpp.out -p -y local query_batch_2000000.fasta target_batch_2000000.fasta > local_dp
cpp.out 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# sha256sum local_dpcpp.out
b5374189941d93263410c3f6bcbaa8b316f1f9380566f8d04f1894eeb54e32c8  local_dpcpp.out
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# ./test_prog_dpcpp.out -p -y semi-global query_batch_2000000.fasta target_batch_2000000.fasta > se
mi-global_dpcpp.out 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# sha256sum semi-global_dpcpp.out
456cffc3f583ac242b364bf107ded68eb6883894210b2c7932c33b03a5671635  semi-global_dpcpp.out
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# ./test_prog_dpcpp.out -p -y global query_batch_2000000.fasta target_batch_2000000.fasta > global_
dpcpp.out 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# sha256sum global_dpcpp.out
74379c71f29adc5f5d1848f9d45307c9905c5a8f70149fc956f6461f77c75e44  global_dpcpp.out
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# ./test_prog_dpcpp.out -p -y ksw query_batch_2000000.fasta target_batch_2000000.fasta > ksw_dpcpp.
out 2>/dev/null
root@e1389a78eac3:/workspace/source_code/dpct/GASAL2/dpct_output/test_prog# sha256sum ksw_dpcpp.out
dafffdc454dc37bb91f125cda4f747b0b3a08d3ee9dcb23d25de7b3071dd0445  ksw_dpcpp.out
```

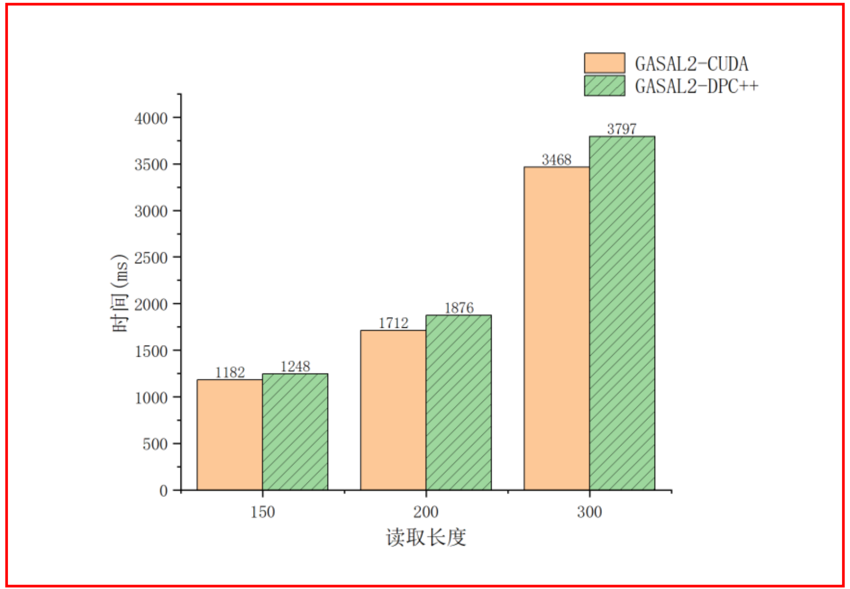
# 性能对比



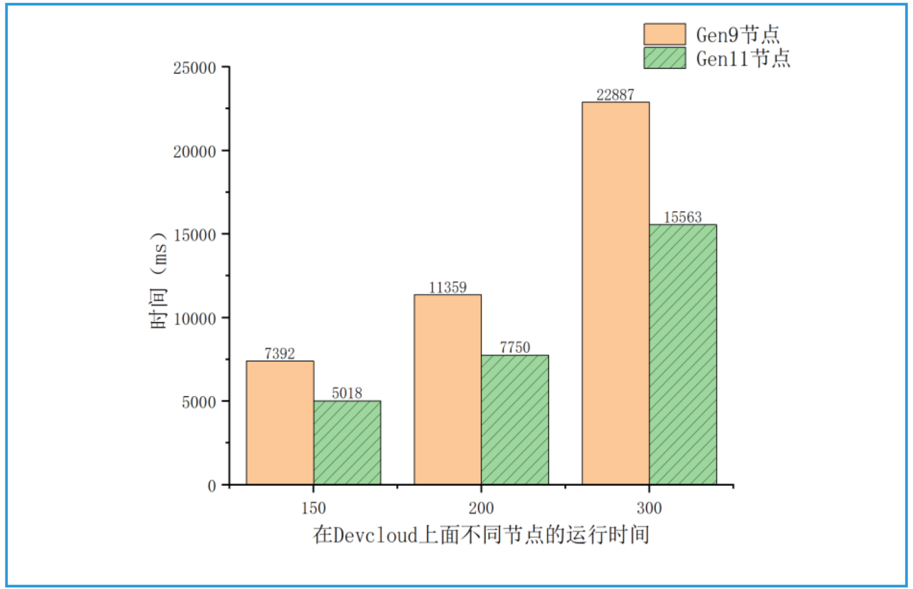
GASAL2-CUDA
GASAL2-DPC++
时间(ms)
1400
1200
1000
800
600
400
200
0
373
435
618
675
1145
1280
150
200
300
读取长度



GASAL2-CUDA
GASAL2-DPC++
时间(ms)
1800
1600
1400
1200
1000
800
600
400
200
0
763
811
1282
1344
1589
1636
150
200
300
读取长度

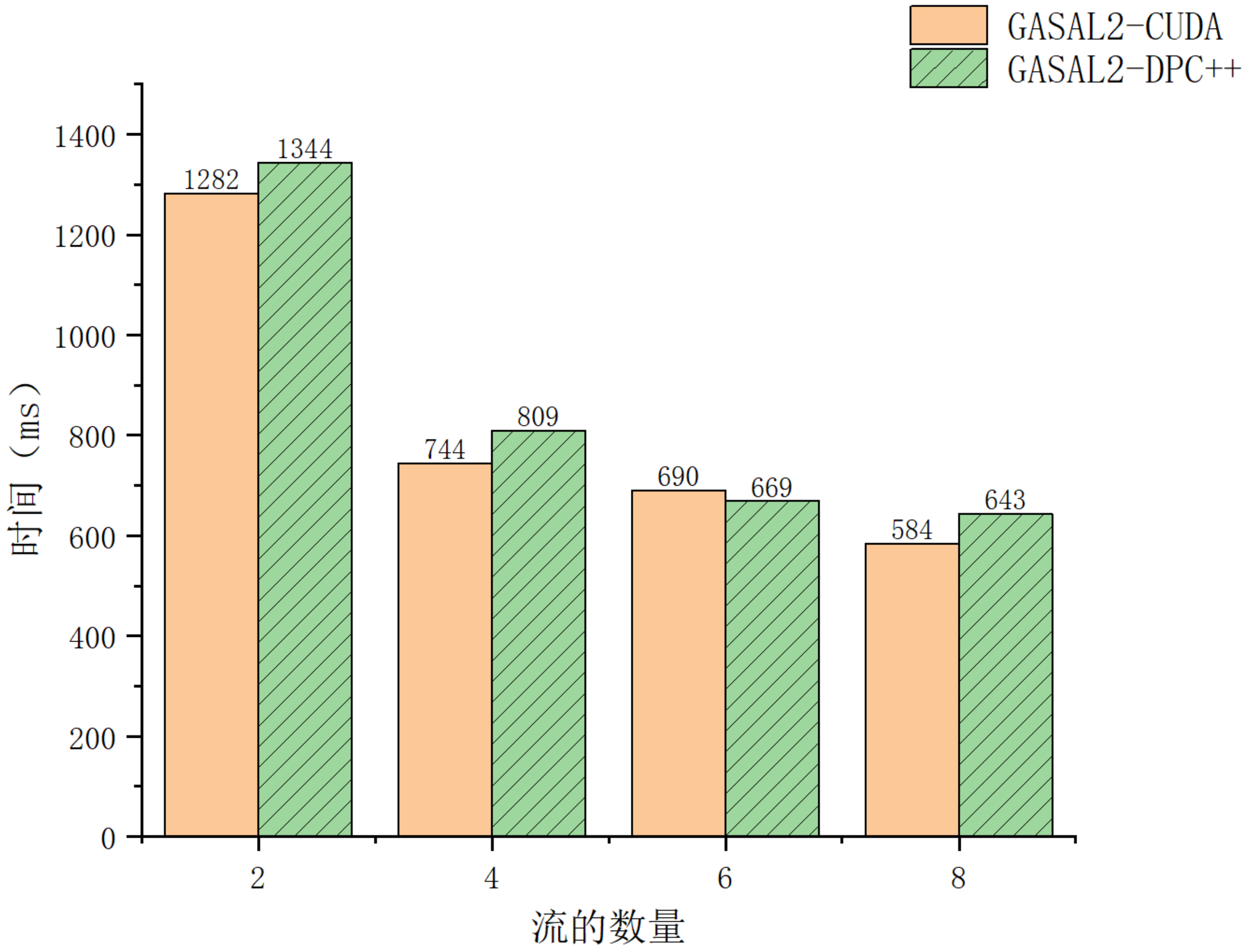


GASAL2-CUDA
GASAL2-DPC++
时间(ms)
4000
3500
3000
2500
2000
1500
1000
500
0
1182
1248
1712
1876
3468
3797
150
200
300
读取长度



Gen9节点
Gen11节点
时间（ms）
25000
20000
15000
10000
5000
0
7392
5018
11359
7750
22887
15563
150
200
300
在Devcloud上面不同节点的运行时间

# 实验结果


GASAL2-CUDA
GASAL2-DPC++
时间（ms）
0
200
400
600
800
1000
1200
1400
1282
1344
744
809
690
669
584
643
2
4
6
8
流的数量

# /04

# 总结

# 总结

可以快速转化

强大的跨架构性

需要细粒度的优化

需要进行适度修改

intel®