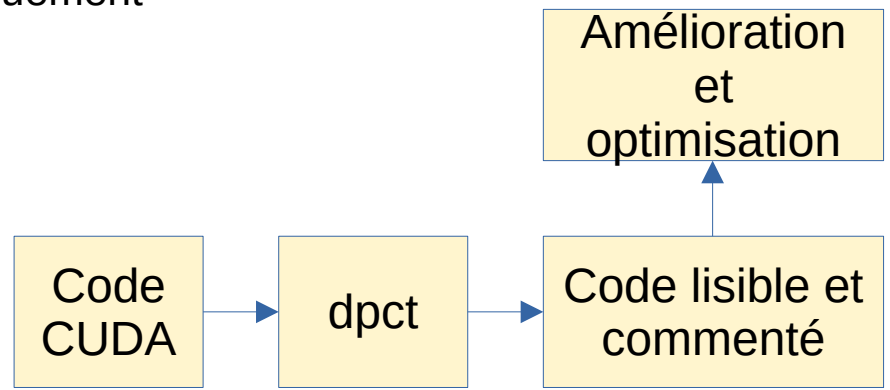
Intel DPC++ Compatibility tools

Migration de code CUDA vers oneAPI

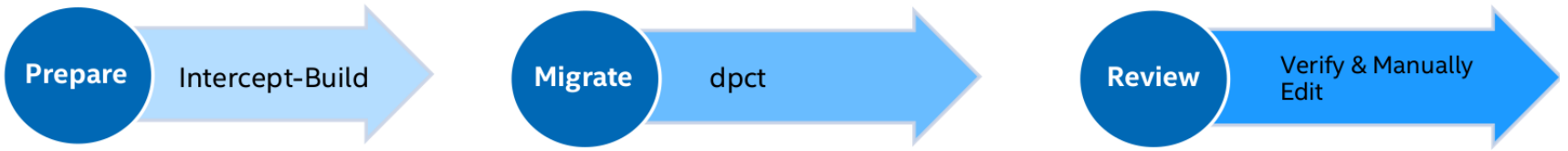
Qu’est-ce c’est ?

- Outil de migration de code CUDA vers l’écosystème oneAPI.
- Fonctionne sous Windows et sous Linux
- Ajouts de commentaires pertinents afin d’aider les développeurs lors de la migration.
- Supporte les versions CUDA 8.0, 9.x, et 11.0 à 11.6
- Seule la partie CUDA est impactée par la migration
- Le code migré sert de base pour les futures développement et optimisations
- 80 % à 90 % du code migré automatirquement



Workflow de migration

Migration d’une base de code large



Création de la base de données de compilation

Intercept-build make

Migration du code source

Dpct -p compile_commands.json ...

Edition du code

Migration code simple

Migration de l’exemple vecAdd

dpcpt –cuda-include-path=/usr/local/cuda-11.7/include -in-root=src src/vector_add.cu

```
NOTE: Could not auto-detect compilation database for file 'vector_add.cu' in '/home/mordicus/dev/oneapi/oneAPI-samples/Tools/Migration/vector-add-dpct/src' or any parent directory
.
The directory "dpct_output" is used as "out-root"
Processing: /home/mordicus/dev/oneapi/oneAPI-samples/Tools/Migration/vector-add-dpct/src/vector_add.cu
/home/mordicus/dev/oneapi/oneAPI-samples/Tools/Migration/vector-add-dpct/src/vector_add.cu:32:14: warning: DPCT1003:0: Migrated API does not return error code. (*, 0) is inserted.
 You may need to rewrite this code.
    status = cudaMemcpy(Result, d_C, VECTOR_SIZE*sizeof(float), cudaMemcpyDeviceToHost);
             ^
Processed 1 file(s) in -in-root folder "/home/mordicus/dev/oneapi/oneAPI-samples/Tools/Migration/vector-add-dpct/src"

See Diagnostics Reference to resolve warnings and complete the migration:
https://software.intel.com/content/www/us/en/develop/documentation/intel-dpcpp-compatibility-tool-user-guide/top/diagnostics-reference.html
```

Le fichier de sortie « vector_add.dp.cpp » est se trouve dans dpct_output_directory

Migration vecAdd

dpct reporte un warning pour nous indiquer qu'il n'a sans doute pas correctement géré une instruction.

```
vector_add.cu:32:14: warning: DPCT1003:0: Migrated API does not return error code. (*, 0) is inserted.
```

Nous retrouvons ce warning dans le fichier de sortie

```
39      /*
40      DPCT1003:0: Migrated API does not return error code. (*, 0) is inserted. You
41      may need to rewrite this code.
42      */
43      status = (q_ct1.memcpy(Result, d_C, VECTOR_SIZE * sizeof(float)).wait(), 0);
```

A comparer avec la version CUDA

```
32      status = cudaMemcpy(Result, d_C, VECTOR_SIZE*sizeof(float), cudaMemcpyDeviceToHost);
33      if (status != cudaSuccess) {
34          printf("Could not copy result to host\n");
35          exit(EXIT_FAILURE);
36      }
```

Migration vecAdd

```
59  catch (sycl::exception const &exc) {
60    std::cerr << exc.what() << "Exception caught at file:" << __FILE__
61              << ", line:" << __LINE__ << std::endl;
62    std::exit(1);
63  }
```

Cependant dpct ajoute une gestion minimale des erreurs grâce aux exceptions.

Migration vecAdd

Par défaut, le code produit par dpct utilise l'USM (unified shared memory). Ce n'est pas toujours la méthode la plus éfficace selon le contexte. Il est possible d'indiquer à dpct d'utiliser les buffers à la place.

```
27      d_A = sycl::malloc_device<float>(VECTOR_SIZE, q_ct1);
28      d_B = sycl::malloc_device<float>(VECTOR_SIZE, q_ct1);
29      d_C = sycl::malloc_device<float>(VECTOR_SIZE, q_ct1);
```

```
dpct --usm-level=none --cuda-include-path=/usr/local/cuda-11.7/include -in-root=src  src/vector_add.cu
```

Dpct va alors utiliser les buffer à la place de la mémoire partagée
Deux valeurs sont possibles pour –usm-level

- restricted
- none

Migration vecAdd

```
dpct --usm-level=restricted --cuda-include-path=/usr/local/cuda-11.7/include -in-root=src  src/vector_add.cu
```

```
27      d_A = sycl::malloc_device<float>(VECTOR_SIZE, q_ct1);
28      d_B = sycl::malloc_device<float>(VECTOR_SIZE, q_ct1);
29      d_C = sycl::malloc_device<float>(VECTOR_SIZE, q_ct1);
```

```
dpct --usm-level=none --cuda-include-path=/usr/local/cuda-11.7/include -in-root=src  src/vector_add.cu
```

```
26      d_A = (float *)dpct::dpct_malloc(VECTOR_SIZE * sizeof(float));
27      d_B = (float *)dpct::dpct_malloc(VECTOR_SIZE * sizeof(float));
28      d_C = (float *)dpct::dpct_malloc(VECTOR_SIZE * sizeof(float));
29
30      dpct::get_default_queue().submit([&](sycl::handler &cgh) {
31          auto d_A_acc_ct0 = dpct::get_access(d_A, cgh);
32          auto d_B_acc_ct1 = dpct::get_access(d_B, cgh);
33          auto d_C_acc_ct2 = dpct::get_access(d_C, cgh);
```

Migration d’un projet plus complex

Rodinia-NW
Recherce de similarités dans les séquences d’acides aminés

Migration d’un projet plus complex

Initialisation du projet

- cd rodinia-nw-dpct
- make clean
- intercept-build make
- cat compile_command.json


```
[
    {
        "command": "nvcc -c  -o needleman_wunsch_cu -D__CUDACC__=1 src/needle.cu",
        "directory": "/home/mordicus/dev/cours/rodinia-nw-dpct",
        "file": "/home/mordicus/dev/cours/rodinia-nw-dpct/src/needle.cu"
    }
]
```

Migration d’un projet plus complex

```
needle.cu:52:16: warning: DPCT1003:10: Migrated API does not return error code. (*, 0) is inserted. You may need to rewrite this code.
    err_code = cudaDriverGetVersion(&version);
               ^
needle.cu:52:16: warning: DPCT1043:11: The version-related API is different in SYCL. An initial code was generated, but you need to
adjust it.
needle.cu:54:63: warning: DPCT1009:12: SYCL uses exceptions to report errors and does not use the error codes. The original code
was commented out and a warning string was inserted. You need to rewrite this code.
        printf("Error \"%s\" checking driver version: %s.\n", cudaGetErrorName(err_code), cudaGetErrorString(err_code));
                                                             ^
needle.cu:54:91: warning: DPCT1009:13: SYCL uses exceptions to report errors and does not use the error codes. The original code
was commented out and a warning string was inserted. You need to rewrite this code.
        printf("Error \"%s\" checking driver version: %s.\n", cudaGetErrorName(err_code), cudaGetErrorString(err_code));
                                                                                         ^
needle.cu:154:9: warning: DPCT1049:14: The work-group size passed to the SYCL kernel may exceed the limit. To get the device limit,
query info::device::max_work_group_size. Adjust the work-group size if needed.
        needle_cuda_shared_1<<<dimGrid, dimBlock>>>(referrence_cuda, matrix_cuda
        ^
needle.cu:162:9: warning: DPCT1049:15: The work-group size passed to the SYCL kernel may exceed the limit. To get the device limit,
query info::device::max_work_group_size. Adjust the work-group size if needed.
...
```

Migration d’un projet plus complex

```
1   CXX = nvcc
2   TARGET = needleman_wunsch_cu
3   SRCS = src/needle.cu
4   DEPS = src/needle_kernel.cu src/needle.h
5
6   # Use predefined implicit rules and add one for *.cu files.
7   %.o: %.cu
8       $(CXX) -c $(CXXFLAGS) $(CPPFLAGS) $< -o $@
9
10  all: $(TARGET)
11
12  $(TARGET): $(SRCS) $(DEPS)
13      $(CXX) $(SRCS) -o $@
14
15  run: $(TARGET)
16      ./$(TARGET) 4096 16
17
18  .PHONY: clean
19  clean:
20      rm -f $(TARGET) *.o result.txt
```

Nouveau Makefile

Migration d’un projet plus complex

La compilation échoue :

```
dpcpp src/needle.dp.cpp -o n
src/needle.dp.cpp:61:14: error: assigning to 'int' from incompatible type 'typename info::param_traits<info::device, (device)4143U>::return_type' (aka 'basic_string<char>')
            dpct::get_current_device().get_info<sycl::info::device::version>(),
            ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
1 error generated.
make: *** [Makefile:13: n] Error 1
```

Mais nous il faut s’occuper des warnings en premier

Migration d’un projet plus complex

warning: DPCT1003:0: Migrated API does not return error code. (*, 0) is inserted. You may need to rewrite this code.
warning: DPCT1043:1: The version-related API is different in SYCL. An initial code was generated, but you need to adjust it.
warning: DPCT1009:2: SYCL uses exceptions to report errors and does not use the error codes. The original code was commented out and a warning string was inserted. You need to rewrite this code.

- Supprimer la gestion inutile des erreurs
- Mettre à jour le code avec les API Sycl

Migration d’un projet plus complex

```
49      int version = 0;
50      int err_code = 999;
51      /*
52  DPCT1003:10: Migrated API does not return error code. (*, 0) is inserted. You
53  may need to rewrite this code.
54  */
55      /*
56  DPCT1043:11: The version-related API is different in SYCL. An initial code was
57  generated, but you need to adjust it.
58  */
59      err_code =
60          (version =
61              dpct::get_current_device().get_info<sycl::info::device::version>(),
62           0);
63      if (err_code != 0)
64          /*
65  DPCT1009:12: SYCL uses exceptions to report errors and does not use the error
66  codes. The original code was commented out and a warning string was inserted.
67  You need to rewrite this code.
68  */
69          /*
70  DPCT1009:13: SYCL uses exceptions to report errors and does not use the error
71  codes. The original code was commented out and a warning string was inserted.
72  You need to rewrite this code.
73  */
74          printf(
75              "Error \"%s\" checking driver version: %s.\n",
```

Migration d’un projet plus complex

```
make run
./needleman_wunsch_dpcpp 4096 16
Start Needleman-Wunsch
Processing top-left matrix
Processing bottom-right matrix
```

Conclusion

Migrez les projets de manière incrémental
Si dpct génère plusieurs erreurs lors de la migration, migrez les fichiers un par un.