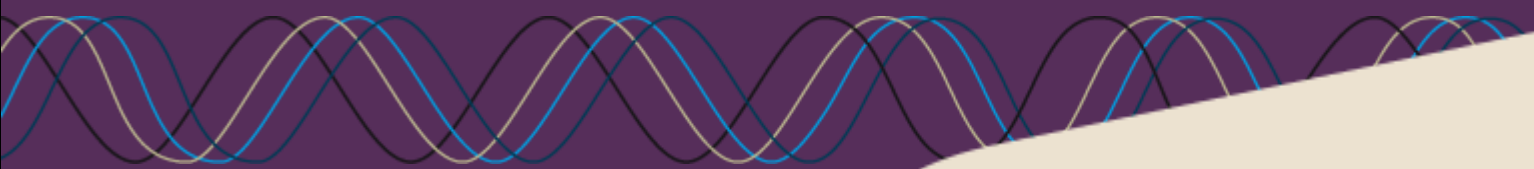
Using Thrust for high performance scientific computing

Thibault Notargiacomo,
gnthibault@gmail.com
thibault.notargiacomo@gipsa-lab.grenoble-inp.fr



gipsa-lab

Grenoble | images | parole | signal | automatique | laboratoire

17th June 2015

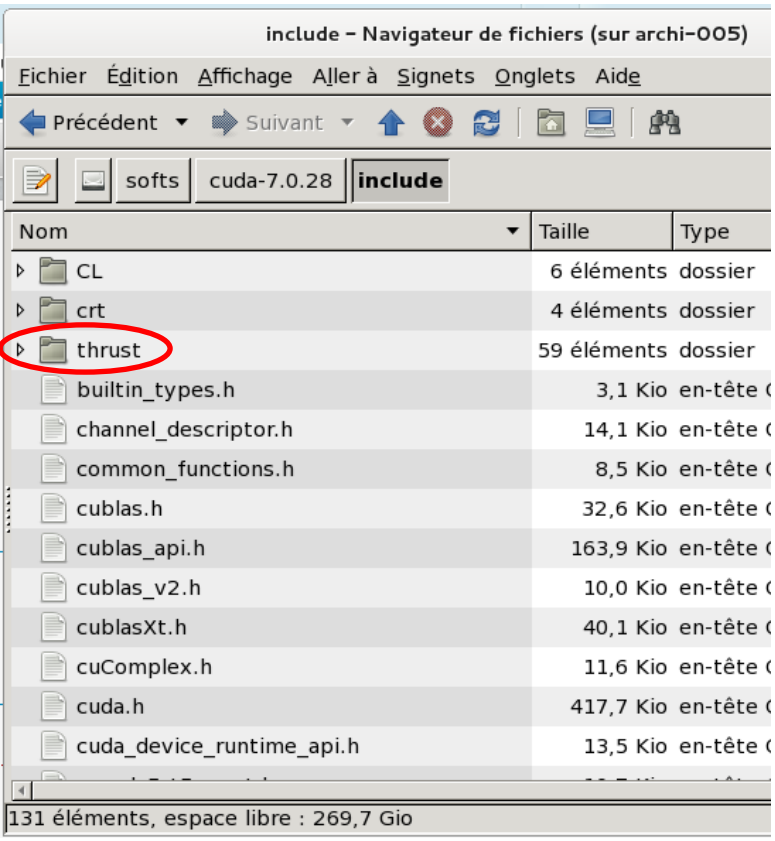
Plan

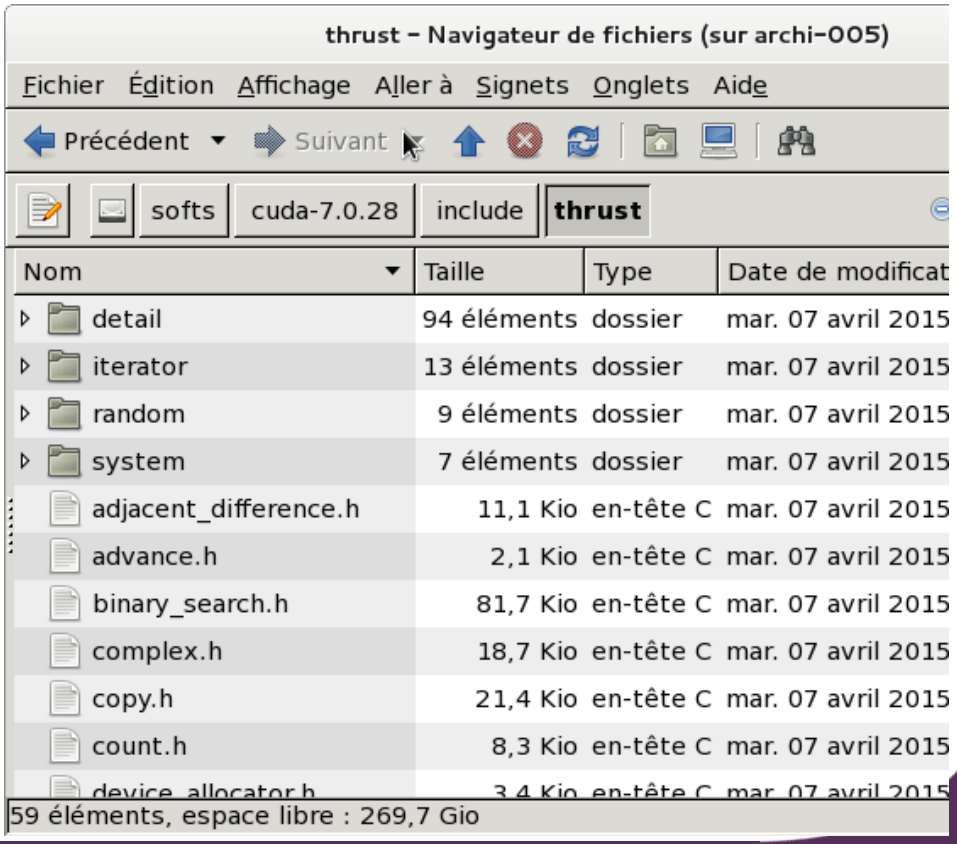
- Introduction : What is Thrust ?
- 1: Studying the device_vector class
- 2: Thrust, an asynchronous library
- 3: Thrust versatility : CPU/GPU
- 4: Thrust UVA and Multi GPU
- 5: Convex optimization using Thrust
- Interesting links
- Conclusion



What is Thrust ?

- A template library
- Not a binary
- Part of Cuda Toolkit





Compiling : Don’t be Afraid !

```
notargth@archi-005:~/Projets/Cuda_Thrust_Introduction/build$ make install
[ 20%] Built target HostDeviceVector
[ 40%] Built target DeviceBackend
[ 60%] Built target AsynchronousLaunch
[ 80%] Built target MultiGpuThrust
[100%] Building NVCC (Device) object ThrustVectorWrappingCublas/CMakeFiles/ThrustVectorWrappingCublas.dir/ThrustVectorWrappingCublas_generated_main.cu.o
/softs/cuda-7.0.28/include/thrust/detail/internal_functional.h(322): error: expression must be a modifiable lvalue
          detected during:
            instantiation of "thrust::detail::enable_if_non_const_reference_or_tuple_of_iterator_references<thrust::tuple_element<1, Tuple>::type>::type thrust::detail::unary_transform_functor<UnaryFunction>::operator()(Tuple) [with UnaryFunction=thrust::identity<float>,
Tuple=thrust::detail::tuple_of_iterator_references<float &, const float &, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>]"
/softs/cuda-7.0.28/include/thrust/detail/function.h(60): here
            instantiation of "Result thrust::detail::wrapped_function<Function, Result>::operator()(const Argument &) const [with Function=thrust::detail::unary_transform_functor<thrust::identity<float>>, Result=void,
Argument=thrust::detail::tuple_of_iterator_references<thrust::device_reference<float>, thrust::device_reference<const float>, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/for_each.inl(57): here
            instantiation of "void thrust::system::cuda::detail::for_each_n_detail::for_each_kernel::operator()(thrust::system::cuda::detail::bulk_::parallel_group<thrust::system::cuda::detail::bulk_::concurrent_group<thrust::system::cuda::detail::bulk_::agent<1UL>, 0UL>, 0UL> &, Iterator,
Function, Size) [with Iterator=thrust::zip_iterator<thrust::tuple<thrust::device_ptr<float>, thrust::device_ptr<const float>, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>>,
Function=thrust::detail::wrapped_function<thrust::detail::unary_transform_functor<thrust::identity<float>>, void>, Size=unsigned int]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/bulk/detail/apply_from_tuple.hpp(71): here
            instantiation of "void thrust::system::cuda::detail::bulk_::detail::apply_from_tuple(Function, const thrust::tuple<Arg1, Arg2, Arg3, Arg4, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type> &) [with
Function=thrust::system::cuda::detail::for_each_n_detail::for_each_kernel, Arg1=thrust::system::cuda::detail::bulk_::parallel_group<thrust::system::cuda::detail::bulk_::concurrent_group<thrust::system::cuda::detail::bulk_::agent<1UL>, 0UL>, 0UL> &,
Arg2=thrust::zip_iterator<thrust::tuple<thrust::device_ptr<float>, thrust::device_ptr<const float>, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>>,
Arg3=thrust::detail::wrapped_function<thrust::detail::unary_transform_functor<thrust::identity<float>>, void>, Arg4=unsigned int]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/bulk/detail/closure.hpp(50): here
            instantiation of "void thrust::system::cuda::detail::bulk_::detail::closure<Function, Tuple>::operator()() [with Function=thrust::system::cuda::detail::for_each_n_detail::for_each_kernel,
Tuple=thrust::tuple<thrust::system::cuda::detail::bulk_::parallel_group<thrust::system::cuda::detail::bulk_::concurrent_group<thrust::system::cuda::detail::bulk_::agent<1UL>, 0UL>, 0UL> &, thrust::zip_iterator<thrust::tuple<thrust::device_ptr<float>, thrust::device_ptr<const float>,
thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>>, thrust::detail::wrapped_function<thrust::detail::unary_transform_functor<thrust::identity<float>>, void>, unsigned int, thrust::null_type,
thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/bulk/detail/cuda_task.hpp(58): here
            [ 33 instantiation contexts not shown ]
            instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator, BinaryFunction) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>, BinaryFunction=thrust::minus<float>]"
/softs/cuda-7.0.28/include/thrust/system/detail/generic/adjacent_difference.inl(44): here
            instantiation of "OutputIterator thrust::system::detail::generic::adjacent_difference(thrust::execution_policy<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(39): here
            instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(68): here
            instantiation of "OutputIterator thrust::adjacent_difference(InputIterator, InputIterator, OutputIterator) [with InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/ThrustWrapper.cu.h(126): here
            instantiation of "void ThrustVectorWrapper<T>::FiniteForwardDifference(const ThrustVectorWrapper<T> &) [with T=float]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/Optimisation.cu.h(162): here


/softs/cuda-7.0.28/include/thrust/system/cuda/detail/assign_value.h(91): error: expression must be a modifiable lvalue
          detected during:
            instantiation of "void thrust::system::cuda::detail::assign_value(thrust::system::cuda::detail::execution_policy<DerivedPolicy> &, Pointer1, Pointer2) [with DerivedPolicy=thrust::system::cuda::detail::tag, Pointer1=thrust::device_ptr<const float>, Pointer2=thrust::device_ptr<float>]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(171): here
            instantiation of "void thrust::reference<Element, Pointer, Derived>::strip_const_assign_value(const System &, OtherPointer) [with Element=const float, Pointer=thrust::device_ptr<const float>, Derived=thrust::device_reference<const float>, System=thrust::device_system_tag,
OtherPointer=thrust::device_ptr<float>]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(139): here
            instantiation of "void thrust::reference<Element, Pointer, Derived>::assign_from(System1 *, System2 *, OtherPointer) [with Element=const float, Pointer=thrust::device_ptr<const float>, Derived=thrust::device_reference<const float>, System1=thrust::device_system_tag,
System2=thrust::device_system_tag, OtherPointer=thrust::device_ptr<float>]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(158): here
            instantiation of "void thrust::reference<Element, Pointer, Derived>::assign_from(OtherPointer) [with Element=const float, Pointer=thrust::device_ptr<const float>, Derived=thrust::device_reference<const float>, OtherPointer=thrust::device_ptr<float>]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(86): here
            instantiation of "thrust::reference<Element, Pointer, Derived>::derived_type &thrust::reference<Element, Pointer, Derived>::operator=(const thrust::reference<OtherElement, OtherPointer, OtherDerived> &) [with Element=const float, Pointer=thrust::device_ptr<const float>,
Derived=thrust::device_reference<const float>, OtherElement=float, OtherPointer=thrust::device_ptr<float>, OtherDerived=thrust::device_reference<float>]"
/softs/cuda-7.0.28/include/thrust/detail/device_reference.inl(34): here
            [ 10 instantiation contexts not shown ]
            instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator, BinaryFunction) [with DerivedPolicy=thrust::system::cuda::detail::tag,
```

Compiling : Don’t be Afraid !

```
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>, BinaryFunction=thrust::minus<float>]"
/softs/cuda-7.0.28/include/thrust/system/detail/generic/adjacent_difference.inl(44): here
          instantiation of "OutputIterator thrust::system::detail::generic::adjacent_difference(thrust::execution_policy<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(39): here
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(68): here
          instantiation of "OutputIterator thrust::adjacent_difference(InputIterator, InputIterator, OutputIterator) [with InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/ThrustWrapper.cu.h(126): here
          instantiation of "void ThrustVectorWrapper<T>::FiniteForwardDifference(const ThrustVectorWrapper<T> &) [with T=float]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/Optimisation.cu.h(162): here


/softs/cuda-7.0.28/include/thrust/system/cuda/detail/trivial_copy.inl(108): error: a value of type "const float *" cannot be used to initialize an entity of type "void *"
        detected during:
          instantiation of "void thrust::system::cuda::detail::trivial_copy_n(thrust::system::cuda::detail::cross_system<System1, System2> &, RandomAccessIterator1, Size, RandomAccessIterator2) [with System1=thrust::host_system_tag, System2=thrust::system::cuda::detail::tag,
RandomAccessIterator1=const float *, Size=std::ptrdiff_t, RandomAccessIterator2=thrust::device_ptr<const float>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/copy_cross_system.inl(151): here
          instantiation of "RandomAccessIterator2 thrust::system::cuda::detail::copy_cross_system(thrust::system::cuda::detail::cross_system<System1, System2>, RandomAccessIterator1, RandomAccessIterator1, RandomAccessIterator2, thrust::random_access_traversal_tag,
thrust::random_access_traversal_tag, thrust::detail::true_type) [with System1=thrust::host_system_tag, System2=thrust::system::cuda::detail::tag, RandomAccessIterator1=const float *, RandomAccessIterator2=thrust::device_ptr<const float>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/copy_cross_system.inl(245): here
          instantiation of "RandomAccessIterator2 thrust::system::cuda::detail::copy_cross_system(thrust::system::cuda::detail::cross_system<System1, System2>, RandomAccessIterator1, RandomAccessIterator1, RandomAccessIterator2, thrust::random_access_traversal_tag,
thrust::random_access_traversal_tag) [with System1=thrust::host_system_tag, System2=thrust::system::cuda::detail::tag, RandomAccessIterator1=const float *, RandomAccessIterator2=thrust::device_ptr<const float>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/copy_cross_system.inl(279): here
          instantiation of "OutputIterator thrust::system::cuda::detail::copy_cross_system(thrust::system::cuda::detail::cross_system<System1, System2>, InputIterator, InputIterator, OutputIterator) [with System1=thrust::host_system_tag, System2=thrust::system::cuda::detail::tag,
InputIterator=const float *, OutputIterator=thrust::device_ptr<const float>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/copy.inl(54): here
          instantiation of "OutputIterator thrust::system::cuda::detail::copy(thrust::system::cuda::detail::cross_system<System1, System2>, InputIterator, InputIterator, OutputIterator) [with System1=thrust::host_system_tag, System2=thrust::system::cuda::detail::tag,
InputIterator=const float *, OutputIterator=thrust::device_ptr<const float>]"
/softs/cuda-7.0.28/include/thrust/detail/copy.inl(37): here
          [ 16 instantiation contexts not shown ]
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator, BinaryFunction) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>, BinaryFunction=thrust::minus<float>]"
/softs/cuda-7.0.28/include/thrust/system/detail/generic/adjacent_difference.inl(44): here
          instantiation of "OutputIterator thrust::system::detail::generic::adjacent_difference(thrust::execution_policy<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(39): here
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(68): here
          instantiation of "OutputIterator thrust::adjacent_difference(InputIterator, InputIterator, OutputIterator) [with InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/ThrustWrapper.cu.h(126): here
          instantiation of "void ThrustVectorWrapper<T>::FiniteForwardDifference(const ThrustVectorWrapper<T> &) [with T=float]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/Optimisation.cu.h(162): here


/softs/cuda-7.0.28/include/thrust/detail/internal_functional.h(322): error: expression must be a modifiable lvalue
        detected during:
          instantiation of "thrust::detail::enable_if_non_const_reference_or_tuple_of_iterator_references<thrust::tuple_element<1, Tuple>::type>::type thrust::detail::unary_transform_functor<UnaryFunction>::operator()(Tuple) [with UnaryFunction=thrust::identity<float>,
Tuple=thrust::detail::tuple_of_iterator_references<const float &, const float &, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>]"
/softs/cuda-7.0.28/include/thrust/detail/function.h(60): here
          instantiation of "Result thrust::detail::wrapped_function<Function, Result>::operator()(const Argument &) const [with Function=thrust::detail::unary_transform_functor<thrust::identity<float>>, Result=void, Argument=thrust::detail::tuple_of_iterator_references<const float &,
thrust::device_reference<const float>, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/for_each.inl(57): here
          instantiation of "void thrust::system::cuda::detail::for_each_n_detail::for_each_kernel::operator()(thrust::system::cuda::detail::bulk_::parallel_group<thrust::system::cuda::detail::bulk_::concurrent_group<thrust::system::cuda::detail::bulk_::agent<1UL>, 0UL>, 0UL> &,
Iterator, Function, Size) [with Iterator=thrust::zip_iterator<thrust::tuple<const float *, thrust::device_ptr<const float>, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>>,
Function=thrust::detail::wrapped_function<thrust::detail::unary_transform_functor<thrust::identity<float>>, void>, Size=unsigned int]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/bulk/detail/apply_from_tuple.hpp(71): here
          instantiation of "void thrust::system::cuda::detail::bulk_::detail::apply_from_tuple(Function, const thrust::tuple<Arg1, Arg2, Arg3, Arg4, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type> &) [with
Function=thrust::system::cuda::detail::for_each_n_detail::for_each_kernel, Arg1=thrust::system::cuda::detail::bulk_::parallel_group<thrust::system::cuda::detail::bulk_::concurrent_group<thrust::system::cuda::detail::bulk_::agent<1UL>, 0UL>, 0UL> &,
Arg2=thrust::zip_iterator<thrust::tuple<const float *, thrust::device_ptr<const float>, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>>,
Arg3=thrust::detail::wrapped_function<thrust::detail::unary_transform_functor<thrust::identity<float>>, void>, Arg4=unsigned int]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/bulk/detail/closure.hpp(50): here
          instantiation of "void thrust::system::cuda::detail::bulk_::detail::closure<Function, Tuple>::operator()() [with Function=thrust::system::cuda::detail::for_each_n_detail::for_each_kernel,
Tuple=thrust::tuple<thrust::system::cuda::detail::bulk_::parallel_group<thrust::system::cuda::detail::bulk_::concurrent_group<thrust::system::cuda::detail::bulk_::agent<1UL>, 0UL>, 0UL> &, thrust::zip_iterator<thrust::tuple<const float *, thrust::device_ptr<const float>,
thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>>, thrust::detail::wrapped_function<thrust::detail::unary_transform_functor<thrust::identity<float>>, void>, unsigned int, thrust::null_type,
thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type, thrust::null_type>]"
/softs/cuda-7.0.28/include/thrust/system/cuda/detail/bulk/detail/cuda_task.hpp(58): here
```

Compiling : Don’t be Afraid !

```
          [ 34 instantiation contexts not shown ]
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator, BinaryFunction) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>, BinaryFunction=thrust::minus<float>]"
/softs/cuda-7.0.28/include/thrust/system/detail/generic/adjacent_difference.inl(44): here
          instantiation of "OutputIterator thrust::system::detail::generic::adjacent_difference(thrust::execution_policy<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(39): here
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag, InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>,
OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(68): here
          instantiation of "OutputIterator thrust::adjacent_difference(InputIterator, InputIterator, OutputIterator) [with InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/ThrustWrapper.cu.h(126): here
          instantiation of "void ThrustVectorWrapper<T>::FiniteForwardDifference(const ThrustVectorWrapper<T> &) [with T=float]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/Optimisation.cu.h(162): here


/softs/cuda-7.0.28/include/thrust/system/cuda/detail/assign_value.h(91): error: expression must be a modifiable lvalue
          detected during:
          instantiation of "void thrust::system::cuda::detail::assign_value(thrust::system::cuda::detail::execution_policy<DerivedPolicy> &, Pointer1, Pointer2) [with DerivedPolicy=thrust::system::cuda::detail::tag, Pointer1=thrust::device_ptr<const float>, Pointer2=const float *]"
(179): here
          instantiation of "void thrust::system::cuda::detail::assign_value(thrust::system::cuda::detail::cross_system<System1, System2> &, Pointer1, Pointer2) [with System1=thrust::system::cuda::detail::tag, System2=thrust::host_system_tag, Pointer1=thrust::device_ptr<const float>,
Pointer2=const float *]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(171): here
          instantiation of "void thrust::reference<Element, Pointer, Derived>::strip_const_assign_value(const System &, OtherPointer) [with Element=const float, Pointer=thrust::device_ptr<const float>, Derived=thrust::device_reference<const float>,
System=thrust::system::cuda::detail::cross_system<thrust::system::cuda::detail::tag, thrust::host_system_tag>, OtherPointer=const float *]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(139): here
          instantiation of "void thrust::reference<Element, Pointer, Derived>::assign_from(System1 *, System2 *, OtherPointer) [with Element=const float, Pointer=thrust::device_ptr<const float>, Derived=thrust::device_reference<const float>, System1=thrust::device_system_tag,
System2=thrust::host_system_tag, OtherPointer=const float *]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(158): here
          instantiation of "void thrust::reference<Element, Pointer, Derived>::assign_from(OtherPointer) [with Element=const float, Pointer=thrust::device_ptr<const float>, Derived=thrust::device_reference<const float>, OtherPointer=const float *]"
/softs/cuda-7.0.28/include/thrust/detail/reference.inl(65): here
          [ 11 instantiation contexts not shown ]
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator, BinaryFunction) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>, BinaryFunction=thrust::minus<float>]"
/softs/cuda-7.0.28/include/thrust/system/detail/generic/adjacent_difference.inl(44): here
          instantiation of "OutputIterator thrust::system::detail::generic::adjacent_difference(thrust::execution_policy<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag,
InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(39): here
          instantiation of "OutputIterator thrust::adjacent_difference(const thrust::detail::execution_policy_base<DerivedPolicy> &, InputIterator, InputIterator, OutputIterator) [with DerivedPolicy=thrust::system::cuda::detail::tag, InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>,
OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/softs/cuda-7.0.28/include/thrust/detail/adjacent_difference.inl(68): here
          instantiation of "OutputIterator thrust::adjacent_difference(InputIterator, InputIterator, OutputIterator) [with InputIterator=thrust::detail::normal_iterator<thrust::device_ptr<float>>, OutputIterator=thrust::detail::normal_iterator<thrust::device_ptr<const float>>]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/ThrustWrapper.cu.h(126): here
          instantiation of "void ThrustVectorWrapper<T>::FiniteForwardDifference(const ThrustVectorWrapper<T> &) [with T=float]"
/home/notargth/Projets/Cuda_Thrust_Introduction/ThrustVectorWrappingCublas/Optimisation.cu.h(162): here


5 errors detected in the compilation of "/tmp/tmpxft_000007bd_00000000-7_main.cpp1.ii".
CMake Error at ThrustVectorWrappingCublas_generated_main.cu.o.cmake:264 (message):
  Error generating file
  /home/notargth/Projets/Cuda_Thrust_Introduction/build/ThrustVectorWrappingCublas/CMakeFiles/ThrustVectorWrappingCublas.dir//./ThrustVectorWrappingCublas_generated_main.cu.o



make[2]: *** [ThrustVectorWrappingCublas/CMakeFiles/ThrustVectorWrappingCublas.dir/ThrustVectorWrappingCublas_generated_main.cu.o] Erreur 1
make[1]: *** [ThrustVectorWrappingCublas/CMakeFiles/ThrustVectorWrappingCublas.dir/all] Erreur 2
make: *** [all] Erreur 2
```

1: device_vector class

1: device_vector class

- What it is:
  - A « container »
  - Cuda buffer Wrapper
  - Equivalent of std::vector<T>
- What it allows:
  - Equivalent of <algorithm> : fill, generate, reduce, sort, …
  - Automatic allocation/destruction
  - Handle some cuda error
  - Ease host/device copy management.
- What it cannot do:
  - Wrap cuda array, 1D,2D,3D textures nor surfaces
  - Bound checking per se

1:Classic usage

Declaration

```
//Thrust Device vectors intend to mimic std::vector class from stl, plus its algorithms
thrust::device_vector<int> deviceVector;
//Also available in host flavour
thrust::host_vector<int> hostVector;
```

Allocation

```
//Allocate vector on device
deviceVector.resize( VEC_SIZE );
//Initialize host vector as size 8 elements, each containing the value 111
hostVector.resize( VEC_SIZE, 111 );
```

Copy To device

```
//Explicit copy to device
thrust::copy( hostVector.begin(), hostVector.end(), deviceVector.begin() );
```

Compute on device

```
//Compute on device, here inclusive scan, for histogram equalization for instance
thrust::inclusive_scan( deviceVector.begin(), deviceVector.end(), deviceVector.begin() );
```

Copy To host

```
//Copy back to host
thrust::copy( deviceVector.begin(), deviceVector.end(), hostVector.begin() );
```

1:Better practical expressivity

Declaration + Allocation

```
//Declare and initialize device vector in one line
thrust::device_vector<int> deviceVector( VEC_SIZE, 111 );
```

Computation on device

```
//Compute algorithm
thrust::inclusive_scan( deviceVector.begin(), deviceVector.end(), deviceVector.begin() );
```

Read or write without explicit copy

```
//Print results
std::cout << "Version 2, vector contains: ";
for( auto it = deviceVector.begin(); it != deviceVector.end(); it++ )
{
        std::cout << " / " << *it;
        //Dereferencing iterator for reading: can also be done for writing !
}
```

1: Compatibility with user allocated memory

Handmade allocation

```
//Raw pointer to device memory
int * raw_ptr;
checkCudaErrors( cudaMalloc((void **) &raw_ptr, VEC_SIZE * sizeof(int) ) );
```

Thrust raw pointer wrapper

```
//Wrap raw pointer with a device_ptr
thrust::device_ptr<int> dev_ptr(raw_ptr);
```

Initializing using thrust utility

```
//Use device_ptr in thrust algorithms
thrust::fill(dev_ptr, dev_ptr + VEC_SIZE, (int) 111);
```

Compute on device

```
//Compute on device, here inclusive scan, for histogram equalization for instance
thrust::inclusive_scan( dev_ptr, dev_ptr + VEC_SIZE, dev_ptr );
```

Wrapper is inconvenient

```
//Print results
std::cout << "Version 3, vector contains: ";
for( int i = 0; i != VEC_SIZE; i++ )
{
        std::cout << " / " << dev_ptr[i];
        //Dereferencing pointer for reading: can also be done for writing !
}
```

1:Compatibility with user written kernels

Handwritten cuda kernel

```
__global__ void naive_sequential_scan( T* ptr )
{
        T val = 0;
        #pragma unroll
        for( auto i = 0; i < SIZE; i++ )
        {
                ptr[i] += val;
                val = ptr[i];
        }
}
```

Declaration + Allocation

```
//Declare and initialize device vector in one line
thrust::device_vector<int> deviceVector( VEC_SIZE, 111 );
```

Declare Synchronization tool

```
//Compute algorithm
cudaStream_t stream;
checkCudaErrors( cudaStreamCreate(&stream) );
```

Launch handwritten kernel

```
naive_sequential_scan<int,VEC_SIZE><<<1,1,0,stream>>>(
                thrust::raw_pointer_cast(deviceVector.data() ) );
```

Synchronize

```
checkCudaErrors( cudaStreamSynchronize( stream) );
```

1:Handle cuda error as exceptions

Declaration + Allocation

Compute on device : wrong iterator

Classic catch block

```
try
{
        //Declare and initialize device vector in one line
        thrust::device_vector<int> deviceVector( VEC_SIZE, 111 );

        //Compute algorithm
        std::cout << "Version 5, we are going to catch an exception: ";
        thrust::inclusive_scan( deviceVector.begin(), deviceVector.end()+1,
                deviceVector.begin() ); //This line purposely contains an error

}
catch( thrust::system_error &e )
{
        std::cerr << "Thrust mechanism for handling error  : " << e.what() << std::endl;
}
```

2: Thrust: An asynchronous library

2: Thrust: An asynchronous library

- Asynchronous behaviour in cuda
  - The compute / copy paradigm
  - Streams concept in cuda
  - Execution_policy in Thrust

- Asynchronous traps
  - Beware of pageable memory !
  - Data chunk size
  - Problem with default stream (--default-stream per-thread)
  - Copy engine ressource

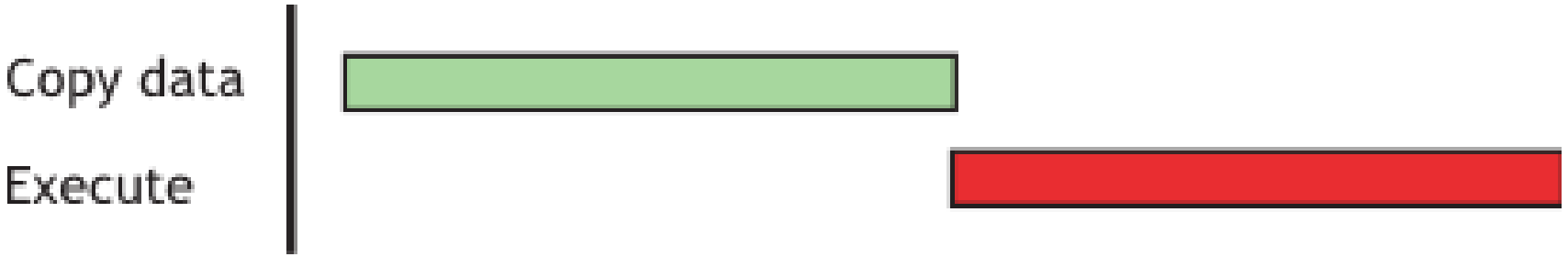
2: Thrust: An asynchronous library

- Execution_policy in Thrust could be
  - thrust::host
  - thrust::device
  - thrust::seq
  - thrust::system::omp::par
  - thrust::system::tbb::par
  - thrust::system::cuda::par(cudaStream_t)

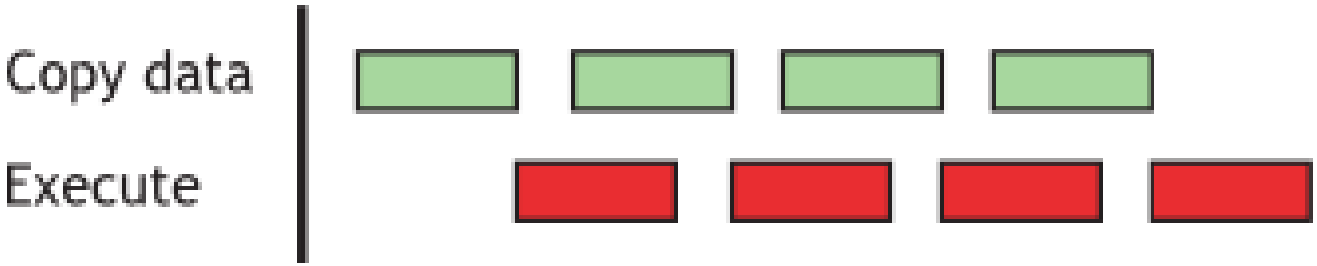
2: Thrust: Multiple stream approach

Achieving Copy / Compute overlapping

Avoid large datasets



Prefere small data chunks



2: Thrust: Multiple stream approach V1

Stream vector

```
//Declare  and initialize cuda stream
std::vector<cudaStream_t> vStream(nbOfStrip);
for( auto it = vStream.begin(); it != vStream.end(); it++ )
{
        cudaStreamCreate( &(*it) );
}
```

Only one loop

```
//Now, we would like to perform an alternate scheme copy/compute in a loop using the copyToDevice/Compute/CopyToHost for each stream scheme:
for( int j=0; j!=nbOfStrip; j++)
{
```

Synchronize

```
    size_t offset = stripSize*j;
    size_t nextOffset = stripSize*(j+1);
    cudaStreamSynchronize(vStream.at(j));
```

Copy to device

```
    cudaMemcpyAsync(thrust::raw_pointer_cast(deviceVector.data())+offset, hostVector+offset, stripSize*sizeof(float), cudaMemcpyHostToDevice, vStream.at(j));
```

Compute

```
    thrust::transform( thrust::cuda::par.on(vStream.at(j)), deviceVector.begin()+offset, deviceVector.begin()+nextOffset, deviceVector.begin()+offset, computeFunctor<float>() );
```

Copy to host

```
    cudaMemcpyAsync(hostVector+offset, thrust::raw_pointer_cast(deviceVector.data())+offset, stripSize*sizeof(float), cudaMemcpyDeviceToHost, vStream.at(j));
}
```

2: Thrust: Multiple stream approach V2

Synchronize loop

```
for( int j=0; j!=nbOfStrip; j++)
{
        cudaStreamSynchronize(vStream.at(j));
}
```

Copy to device loop

```
for( int j=0; j!=nbOfStrip; j++)
{
        size_t offset = stripSize*j;
        cudaMemcpyAsync(thrust::raw_pointer_cast(deviceVector.data())+offset,
        hostVector+offset, stripSize*sizeof(float), cudaMemcpyHostToDevice, vStream
}
```

Compute loop

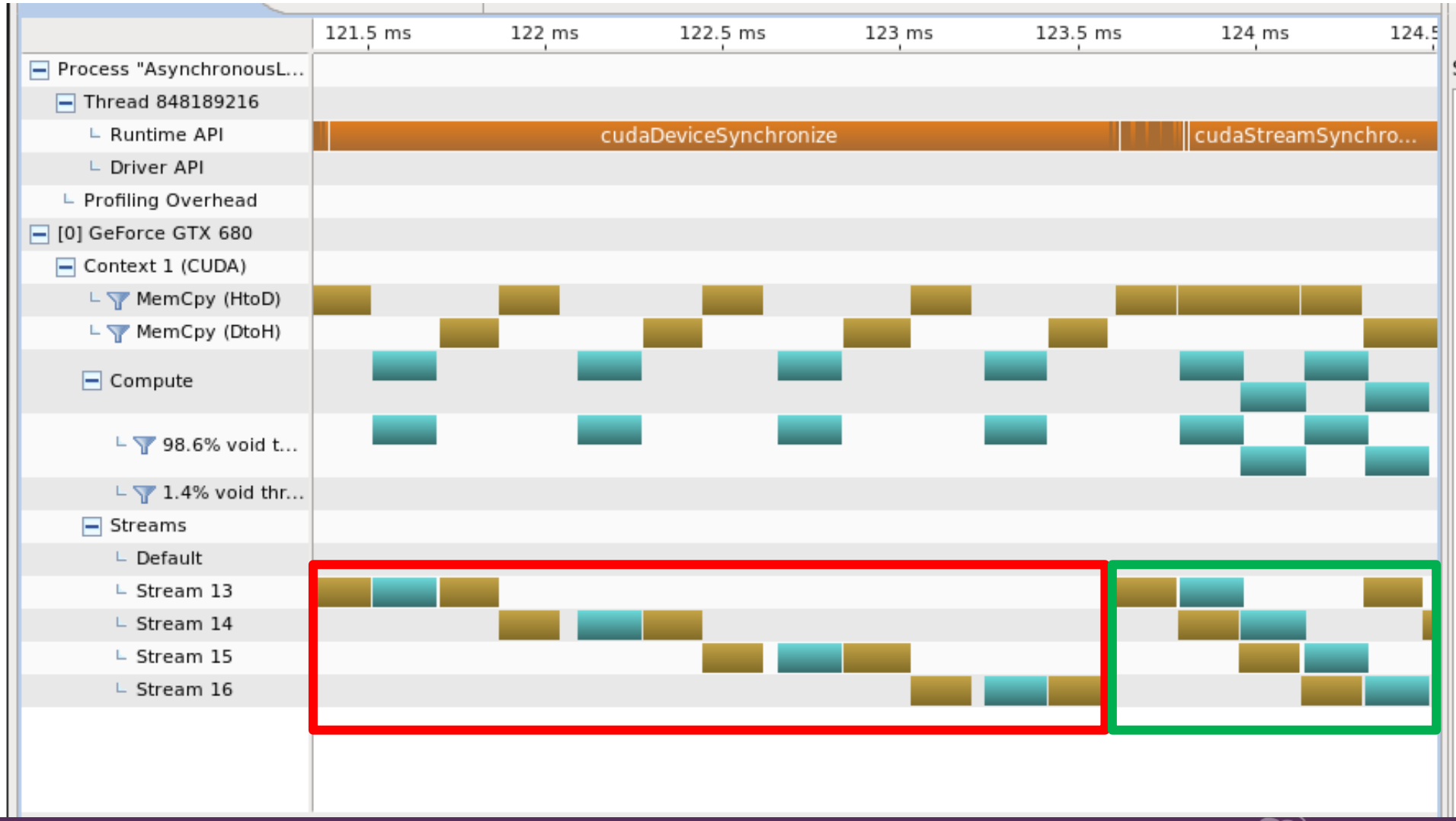
```
for( int j=0; j!=nbOfStrip; j++)
{
        size_t offset = stripSize*j;
        size_t nextOffset = stripSize*(j+1);
        thrust::transform( thrust::cuda::par.on(vStream.at(j)), deviceVector.begin()+offs
        deviceVector.begin()+nextOffset, deviceVector.begin()+offset,
        computeFunctor<float>() );
}
```

Copy to host loop

```
for( int j=0; j!=nbOfStrip; j++)
{
        size_t offset = stripSize*j;
        cudaMemcpyAsync(hostVector+offset,
        thrust::raw_pointer_cast(deviceVector.data())+offset, stripSize*sizeof(float),
        cudaMemcpyDeviceToHost, vStream.at(j));
}
```

2: Thrust: An asynchronous library

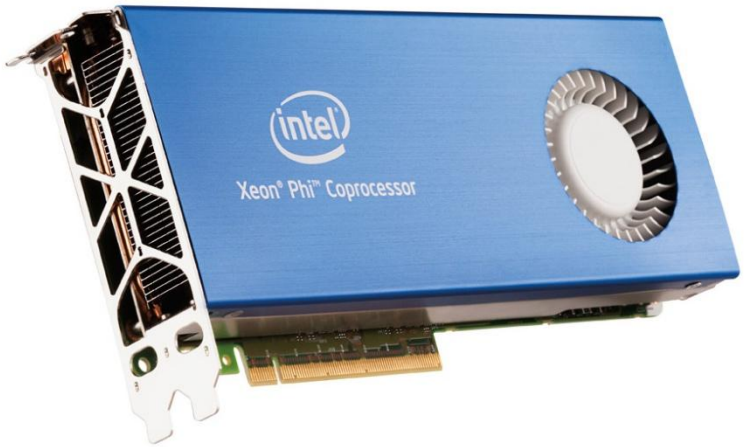
Who ’s who ?



3: Thrust versatility : CPU/GPU

3: Thrust versatility : CPU/GPU

- Versatility
  - Code once, get all implementations
  - Ease GPU speedup calculation
  - Intel vs Nvidia : grab Popcorn and sit



3: Thrust device system

- High level concept

- Multiple possible backends :
  - THRUST_DEVICE_SYSTEM_CUDA
  - THRUST_DEVICE_SYSTEM_OMP
  - THRUST_DEVICE_SYSTEM_TBB

- Compile time decision
  - Using option -DTHRUST_DEVICE_SYSTEM



3: Benchmarking backends on sort

CmakeLists.txt

```
###############################################
#         Miscellaneous parallel computing lib       #
###############################################

#Change device execution for fun !
set(THRUST_DEVICE_SYSTEM THRUST_DEVICE_SYSTEM_CUDA)
#set(THRUST_DEVICE_SYSTEM "THRUST_DEVICE_SYSTEM_OMP -Xcompiler -fopenmp" )
#set(THRUST_DEVICE_SYSTEM THRUST_DEVICE_SYSTEM_TBB)

list( APPEND CUDA_NVCC_FLAGS -DTHRUST_DEVICE_SYSTEM=${THRUST_DEVICE_SYSTEM}
```

3: Benchmarking backends on sort

Core code

Start timer

```
//Now measure how many time it take to perform sorting operation
auto begin = std::chrono::high_resolution_clock::now();
```

Compute

```
thrust::sort( deviceVector.begin(), deviceVector.end() );
```

Conditional synchronization point

```
#if THRUST_DEVICE_SYSTEM == THRUST_DEVICE_SYSTEM_CUDA
//Synchronize because of aynchronous behaviour in cuda mode
cudaDeviceSynchronize();
#endif // THRUST_DEVICE_SYSTEM == THRUST_DEVICE_SYSTEM_CUDA
```

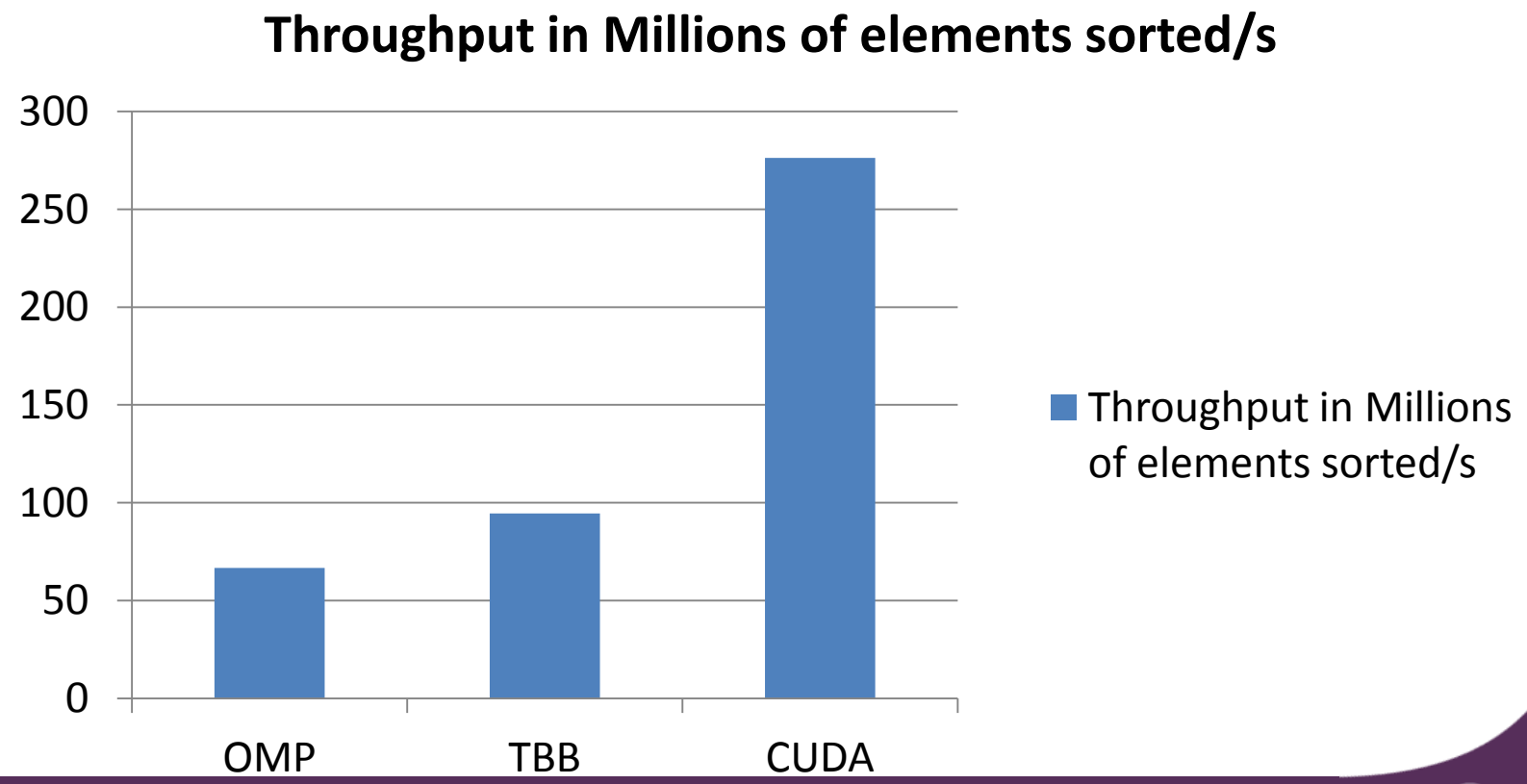
Stop timer

```
auto end = std::chrono::high_resolution_clock::now();
```

3: Benchmarking backends on sort

- Results

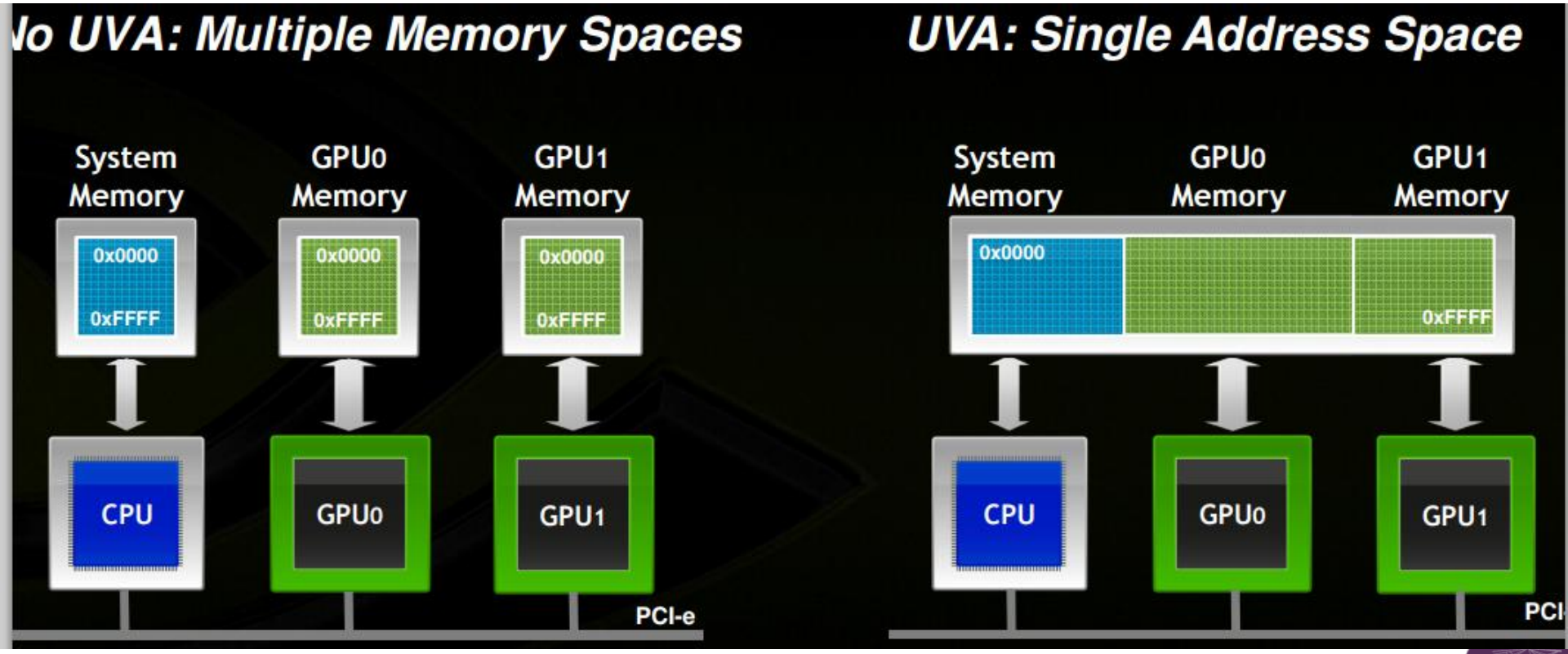
```
//OpenMP backend sorted 134’217’728 elements in 2.01271 seconds (66.685 Millions of elements/s )
//TBB backend sorted 134’217’728 elements in 1.42055 seconds (94.4827 Millions of elements/s )
//Cuda backend sorted 134’217’728 elements in 0.485675 seconds (276.353 Millions of elements/s )
```



4: Thrust UVA and Multi GPU

4: Thrust UVA and Multi GPU

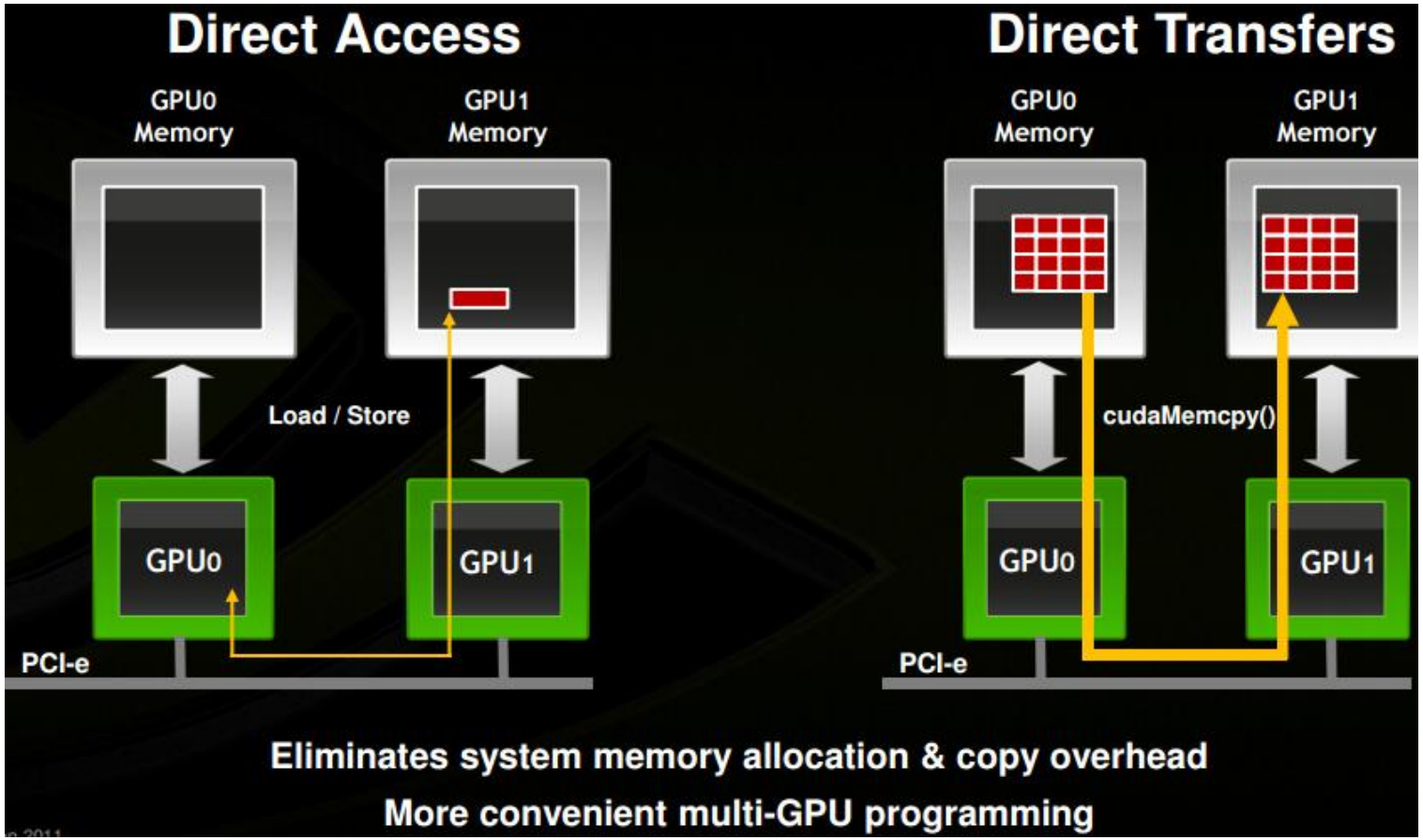
- What is Unified Virtual Addressing ?



Source: http://on-demand.gputechconf.com/gtc-express/2011/presentations/cuda_webinars_GPUDirect_uva.pdf

4: Thrust UVA and Multi GPU

- Peer to peer memory access

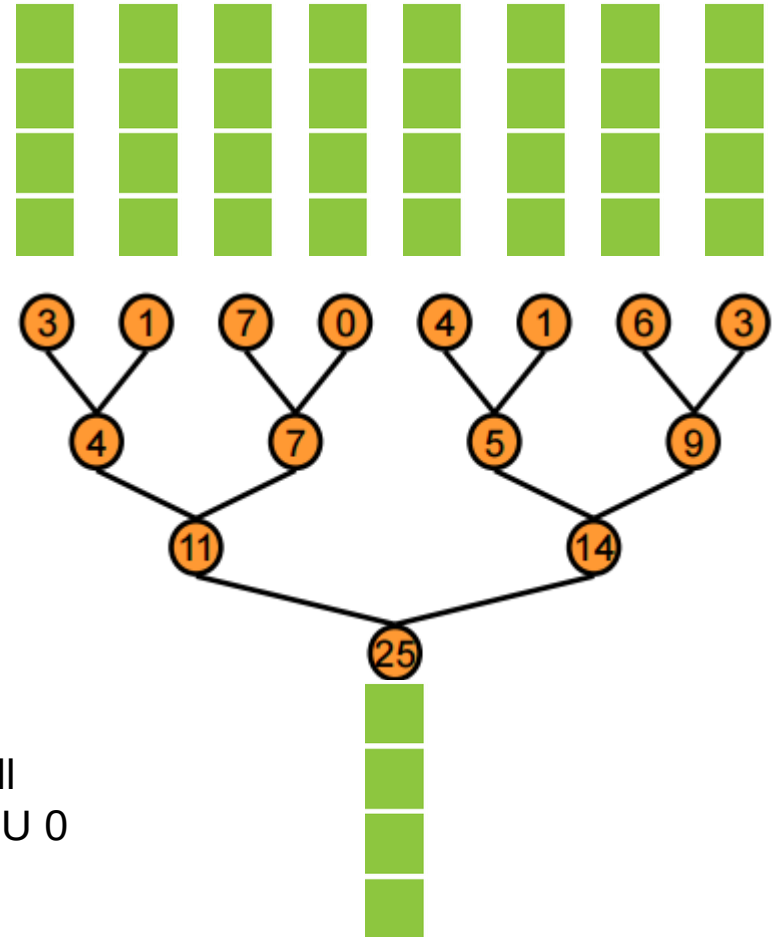


Source: http://on-demand.gputechconf.com/gtc-express/2011/presentations/cuda_webinars_GPUDirect_uva.pdf

4: Thrust UVA and Multi GPU

- Peer to peer memory reduction through thrust

Input: 8 Gpu, each containing a vector



Output: addition of all vectors to one on GPU 0

4: Thrust UVA and Multi GPU

- Peer to peer memory reduction through thrust

Set current device

Memory is allocated on right device

Grant access to all device having superior IDs

```
for( int i = 0; i != nb_device; i++ )
{
    //Set device as the current device
    checkCudaErrors( cudaSetDevice( i ) );

    //Initialize memory
    vpDeviceVector.emplace_back(
            std::make_shared<thrust::device_vector<int> >( sizeVector, 111 ) );

    //Enable Peer to Peer access, ie, current device can acces to memory of all superior device IDs
    for( int j = i+1; j < nb_device; j++ )
    {
            checkCudaErrors( cudaDeviceEnablePeerAccess(j, 0) );
    }
}
```

4: Thrust UVA and Multi GPU

- Peer to peer memory reduction through thrust

Get upper power of 2

Perform a associative binary operation

Reduction is log2(n) in number of steps

```
// This is where reduction take place
int maxTid = giveReductionSize(nb_device);
while( maxTid != 0 )
{

        //Reduce from high IDs to low ones
        for(int i = 0; i < maxTid; ++i)
        {
                reduceVector( vpDeviceVector, i, maxTid );
        }

        //Half the work is remaining
        maxTid /= 2;
}
```

4: Thrust UVA and Multi GPU

- Peer to peer memory reduction through thrust

```
void reduceVector( std::vector<std::shared_ptr<thrust::device_vector<int> > >& v, int tid, int maxTid
{
        if( tid + maxTid <  v.size() )
        {
                //Set current device
                cudaSetDevice( tid );

                // We add vector tid and vector tid+maxTid and put the result into vector tid
                thrust::transform( v.at(tid)->begin(), v.at(tid)->end(), v.at(tid+maxTid)->begin(),
                        v.at(tid)->begin(), thrust::plus<int>() );
        }
}
```

Check bound

Set current active GPU

Transparent thrust transformation

5: Convex optimization using Thrust and Cublas

5: Convex optimization using Thrust and Cublas

- Why convex optimization on GPU ?
  - Unnecessary on small well posed systems
  - Ill-posed problems needs iterative methods
  - Iterative methods are expensive for large systems
  - Large problems needs parallelism

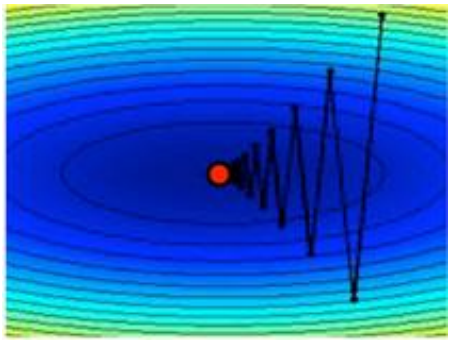
5: Convex optimization using Thrust : Steepest descent

- Simple algorithm for convex linear systems
  - Quadratic objectif function: easily differentiable
  - Aka Least square solution

$$\min_{x \in \mathbb{R}^d} f(x) = \frac{1}{2} \|AX - B\|^2$$

  - Solved by step each time going in the opposite sense of the gradient:

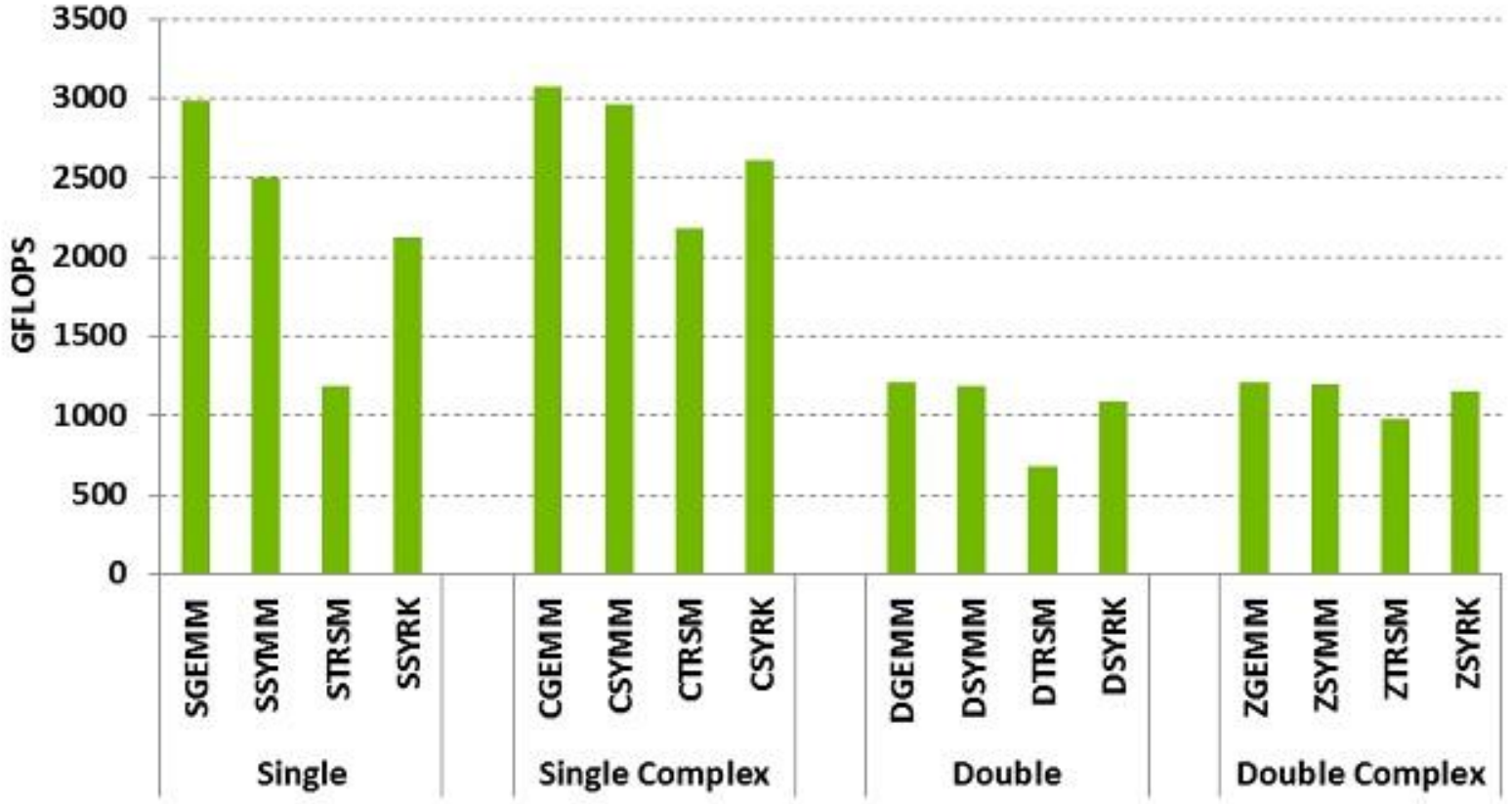
$$\nabla f(x) = A^t AX - A^t B$$



Source: Gabriel Peyré

5: Convex optimization using Thrust : What is Cublas ?

- A powerful library (Basic Linear Algebra Subprogram)



5: Convex optimization using Thrust and Cublas

- Our strategy: Wrap everything inside a higher level interface

Cublas official interface

```
cublasSgemv(handle, transA, m, n, alpha, A, lda, B, ldb, beta, C, ldc)
```

Our wrapper interface

```
void Prod(const ThrustVectorWrapper<T>& Input, ThrustVectorWrapper<T>& Output)
```

Thrust interface

```
thrust::transform( m_deviceVector.begin(), m_deviceVector.end(), in.begin(),
m_deviceVector.begin(), thrust::plus<T>() );
```

Our wrapper interface

```
void Add( const ThrustVectorWrapper<T>& Input )
```

5: Convex optimization using Thrust and Cublas

- Resulting algorithm:

```
while( (niter < nbIteration) && (L2Error > convergenceTol) )
{
        A.Prod( X, Ax );                                  // Ax = A * x
        Ax.Substract( B );                                // Ax = Ax - b
        A.transProd( Ax, grad );                          // grad = A^t(Ax - B)
        A.Prod( grad, Ag );                               // Ag = A * gradient
        gradstep = grad.GetNorm22()/Ag.GetNorm22();       // Compute gradient step
        X.Saxpy( grad, -gradstep, false );                // Update solution

        L2Error = Ax.GetNorm22();                         // Compute functional at current step

        niter++;                                          // Ready for next iteration
}
```

Output:

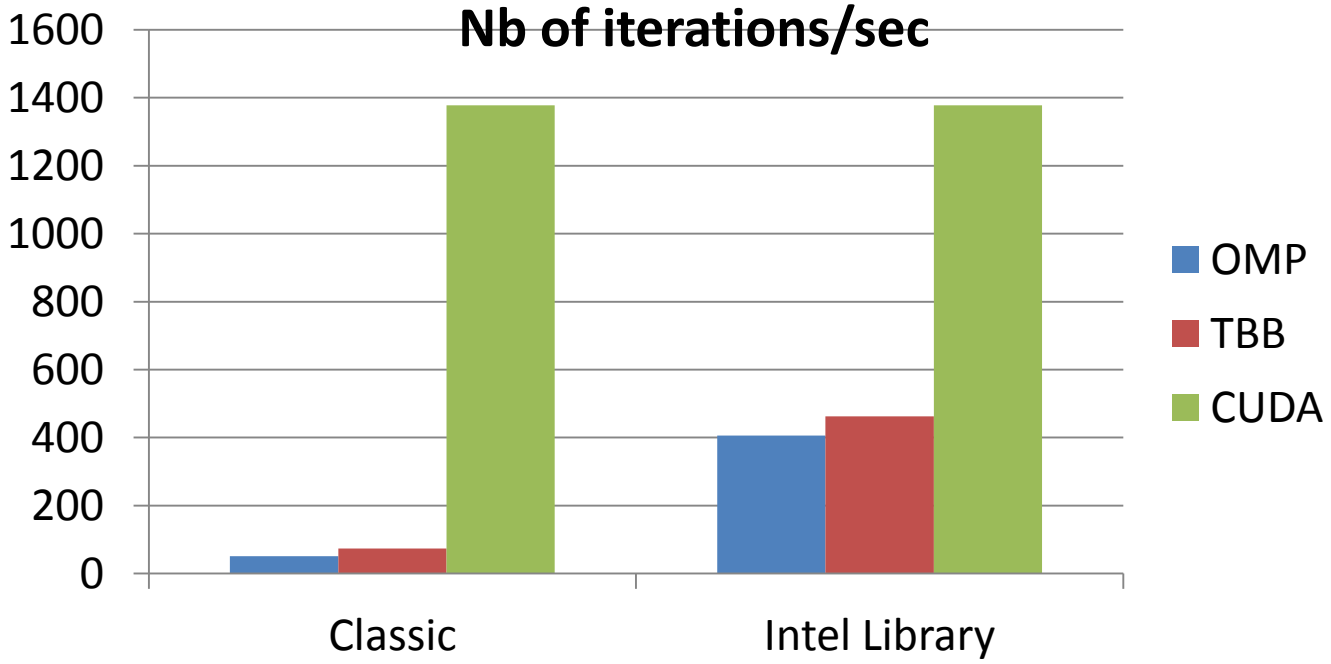
```
./ThrustVectorWrappingCublas
Iteration : 0 over 1000 , L2 error = 653.522
Iteration : 1 over 1000 , L2 error = 164.205
Iteration : 2 over 1000 , L2 error = 82.2171
Iteration : 3 over 1000 , L2 error = 68.4766
Iteration : 4 over 1000 , L2 error = 59.1165
Iteration : 5 over 1000 , L2 error = 52.7413
```

5: Convex optimization using Thrust and Cublas : Benchmark

//CPU code linked with default gsl_cblas lib and default gcc gomp threading library
//OpenMP backend performed 1000 iterations of gradient descent elements in 19.6776 seconds (50.8192 iterations per seconds)
//TBB backend performed 1000 iterations of gradient descent elements in 13.6715 seconds (73.145 iterations per seconds)

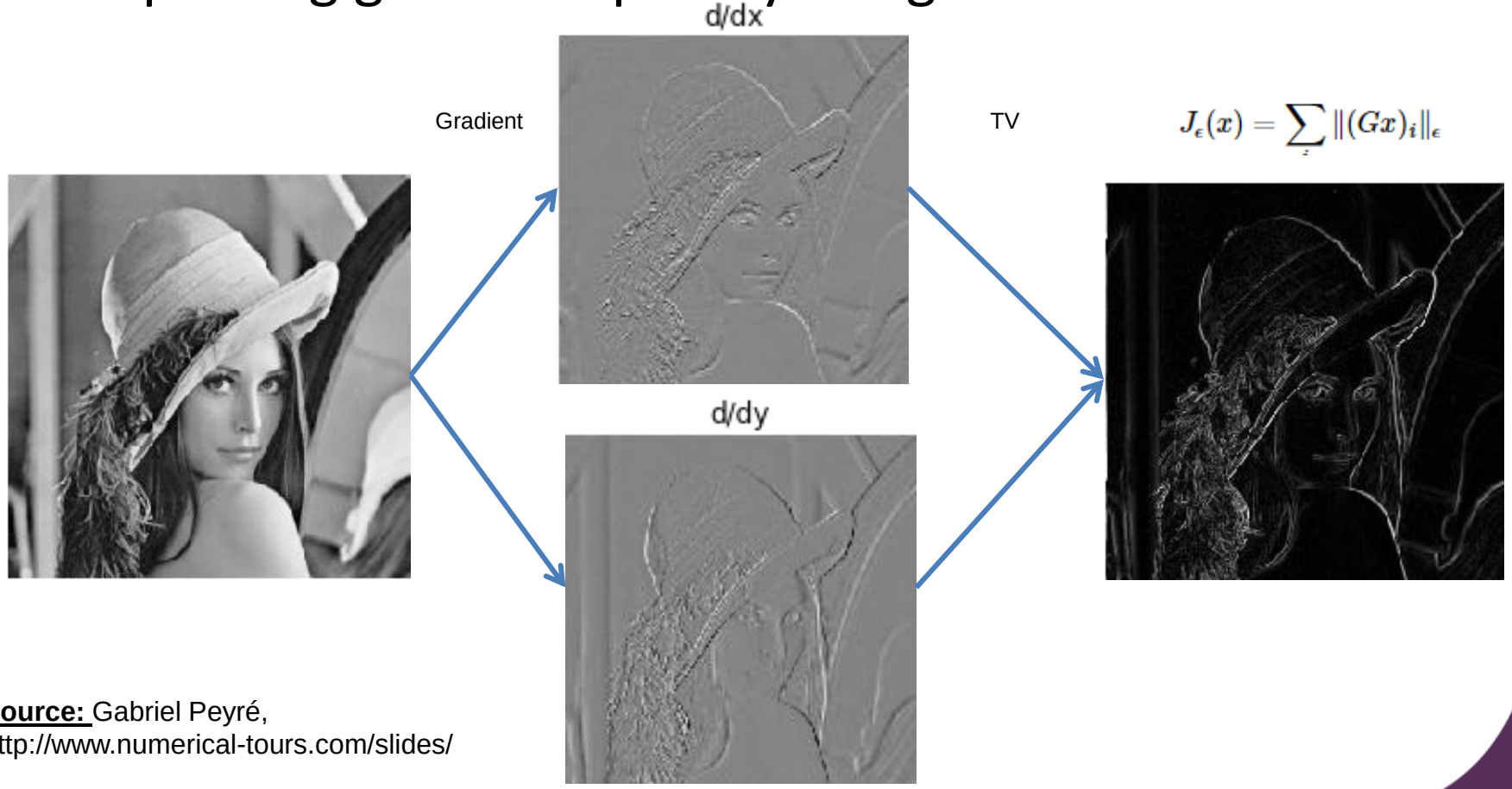
//CPU code Linked with MKL from Intel, and openMP runtime from intel (iomp5 instead of gomp
//OpenMP backend performed 1000 iterations of gradient descent elements in 2.46626 seconds (405.473 iterations per seconds)
//TBB backend performed 1000 iterations of gradient descent elements in 2.163 seconds (462.32 iterations per seconds)

//Cuda Backend
//Cuda backend performed 1000 iterations of gradient descent elements in 0.725926 seconds (1377.55 iterations per seconds



5: Gradient descent for signal processing

- Exploiting gradient sparsity in signals:



Source: Gabriel Peyré, http://www.numerical-tours.com/slides/

5: Gradient descent for signal processing

- Denoising as an optimization problem:

y

x

$$J_\epsilon(x) = \sum_i \|(Gx)_i\|_\epsilon$$

- Helps crafting our objective function

$$\min_{x \in \mathbb{R}^d} \; f(x) = \frac{1}{2}\|y - x\|^2 + \lambda J_\epsilon(x)$$

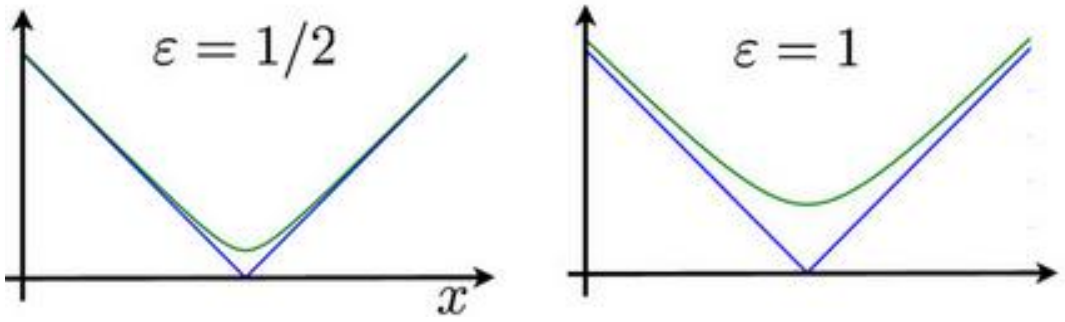
5: Gradient descent for signal processing

- Gradient of objective function gives:

$$\nabla f(x) = x - y + \lambda \nabla J_\epsilon(x)$$

- Deriving the Total Variation ?

$$\nabla J_\epsilon(x)_i = -div(u) \quad \text{where} \quad u_i = \frac{(Gx)_i}{\|(Gx)_i\|_\epsilon}$$



$\sqrt{x^2 + \varepsilon^2}$

$|x|$

- Ready for the gradient descent ☺

5: Gradient descent for signal processing

- Algorithm is:

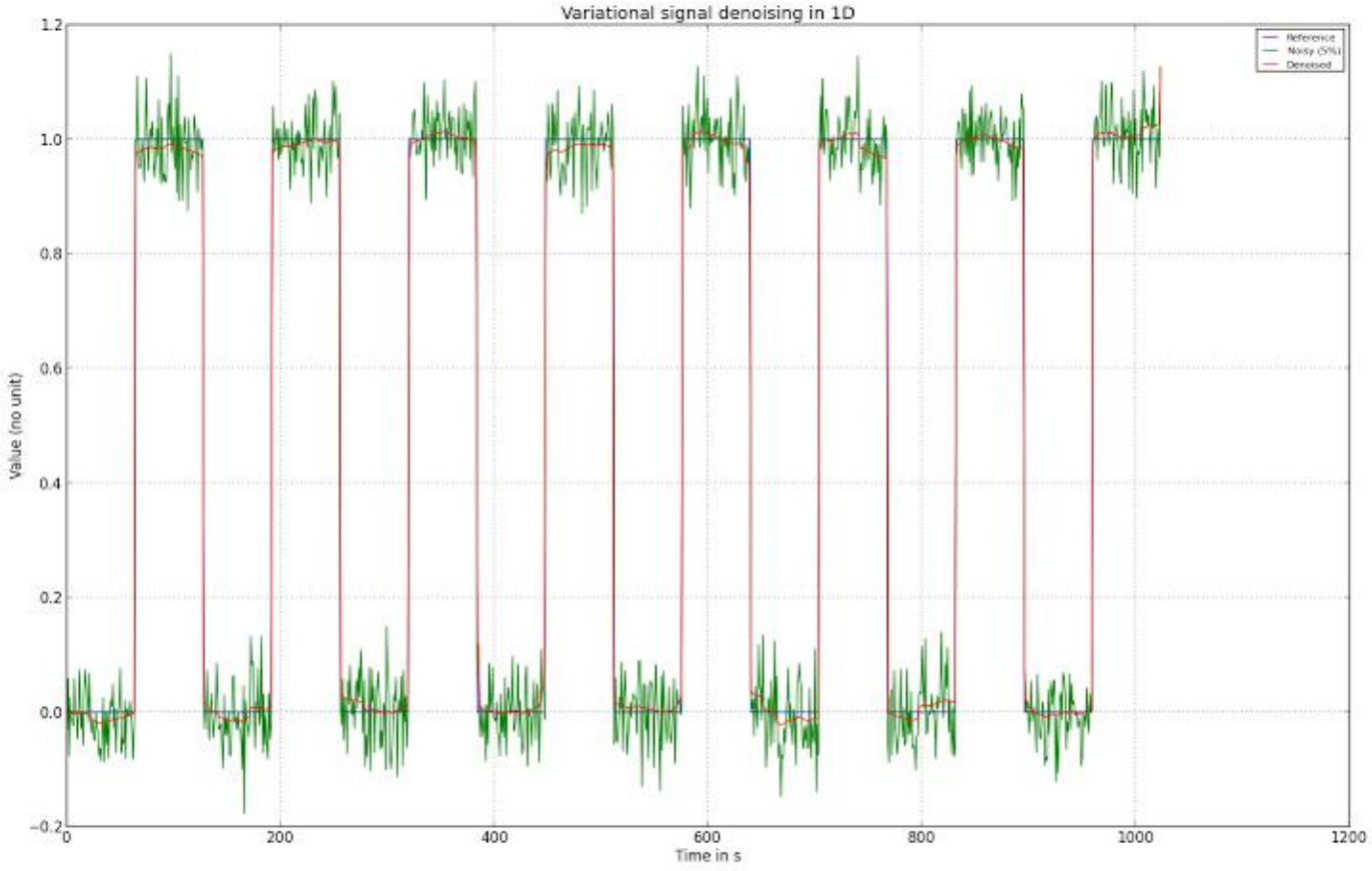
```
while( niter < nbIteration )
{
    grad.Assign( X );                                   // grad = X
    grad.Substract( Y );                                // grad = X – Y
    TvGradientTmp.FiniteForwardDifference( X );         // TvGradient = G(X)
    TvGradientTmp.ApplySmoothedTVGradient(epsilonNorm); // TvGradient = TvGradient / ||TvGradient||e
    TvGradient.FiniteBackwarDifference(TvGradientTmp);  // TvGradient = div( TvGradient / ||TvGradient||e )
    grad.Saxpy( TvGradient, -lambda, false );           // grad = X - Y + GradientTV
    X.Saxpy( grad, -stepSize, false );                  // Update solution

    niter++;                                            // Ready for next iteration
}
```

- Helpers from Thrust:

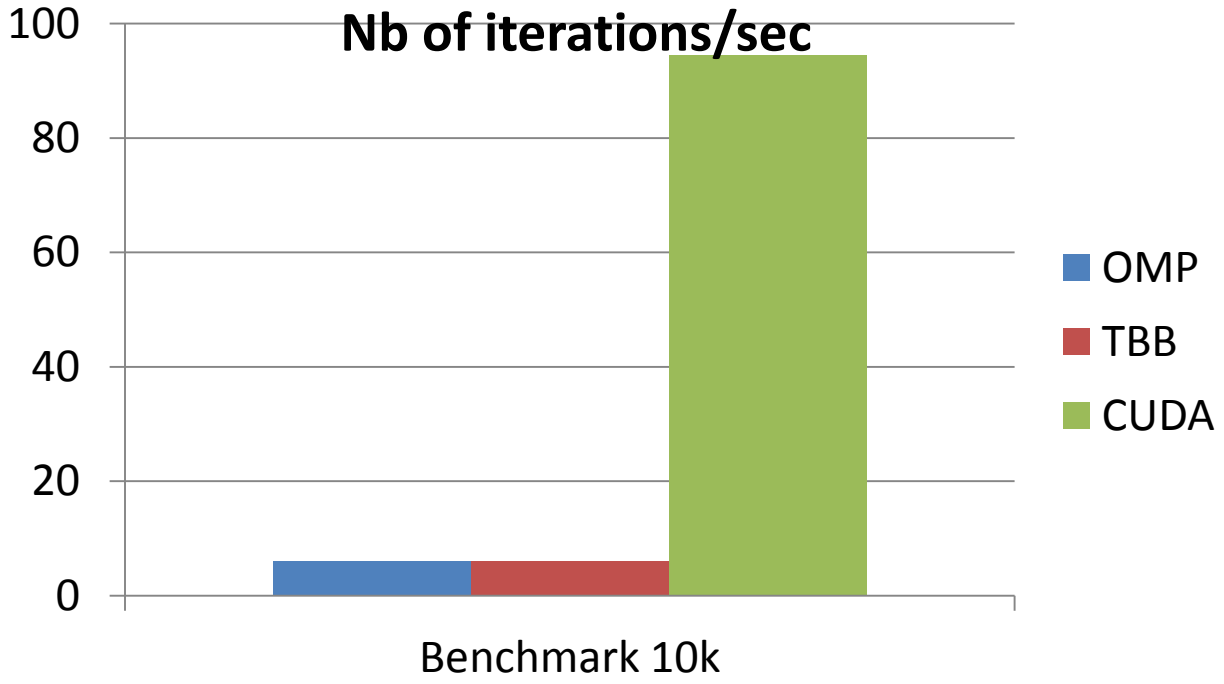
```
thrust::adjacent_difference( in.begin(), in.end(), m_deviceVector.begin());
```

5: Gradient descent for signal processing : Results in 1D



5: Gradient descent for signal processing : Benchmark

//CPU code linked with default gcc gomp threading library
//OpenMP backend performed 10000 iterations of gradient descent over 33'554'432 elements in 1672.89 seconds (5.97768 iterations per seconds)
//TBB backend performed 10000 iterations of gradient descent over 33'554'432 elements in 1648.48 seconds (6.0662 iterations per seconds)
//Cuda Backend
//Cuda backend performed 10000 iterations of gradient descent over 33'554'432 elements in 105.78 seconds (94.5358 iterations per seconds)



Cuda Community and Useful links

- Cuda Official Documentation
  - http://docs.nvidia.com/cuda/cuda-c-programming-guide/
  - http://docs.nvidia.com/cuda/cuda-runtime-api/index.html
- Thrust Official documentation
  - http://thrust.github.io/doc/modules.html
  - https://github.com/thrust/thrust/tree/master/examples
- Nvidia Cuda official forum
  - https://devtalk.nvidia.com/default/board/57/
- Stack Overflow
  - http://stackoverflow.com/search?q=cuda
- Udacity (Best MOOC for Cuda)
  - https://www.udacity.com/wiki/cs344
- Mark Harris (Chief Technologist, GPU Computing at NVIDIA)
  - https://twitter.com/harrism
  - https://twitter.com/GPUComputing
  - https://github.com/harrism
- This tutorial
  - https://github.com/gnthibault/Cuda_Thrust_Introduction
  - https://twitter.com/gnthibault

Conclusion

- Thrust allows:
  - Saving coding time
  - Clearer code
  - Intensive parameter exploration
  - Portability : CPU/GPU

- Take Home message
  - Think parallel
  - Don’t reinvent the wheel : use libraries
  - Use wrappers