IICoMP Documentation

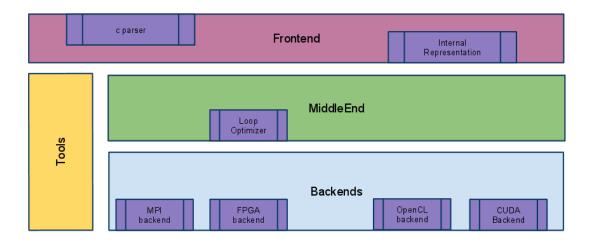
Release 1.0

Ruymán Reyes

CONTENTS

1	Front	Frontend				
	1.1	Parsing Tools	3			
	1.2	Internal Representation	3			
2	Back	ends	5			
	2.1	Common	5			
	2.2	Dot Backend	8			
	2.3	CBackend	8			
	2.4	CUDA Backend	9			
3	Tools		11			
	3.1	Tree Tools	11			
	3.2	Serialization Tools				
	3.3	Declarations Tools				
	3.4	Debug Tools	11			
4	Gloss	sary	13			
5	5 Indices and tables					
M	odule l	Index	17			
In	Index					

llCoMP is a translator framework designed for *fast prototyping*. With little effort, you can build translators from OpenMP/C to different High Performance Computing languages, libraries and frameworks. Currently we have implemented the CUDA Backend, but we have plans to implement new ones.



In the diagram (layered_design), the different layers of the framework are exposed.

The uppermost level contains the Frontend, which gives the tools required to transform the source code into the internal representation.

The MiddleEnd module encapsulates transformations from the IR to the IR, for example, loop optimizations or type data conversions.

Finally Backends module contains all the implemented backends

Tools to manipulate the internal representation (and do some other stuff), are packaged on the Tools module.

In addition, some utils and examples are presented in order to show the capabilities of the framework.

Contents:

CONTENTS 1

2 CONTENTS

CHAPTER ONE

FRONTEND

The frontend module builds the internal representation from a source file.

Two modules, representing the two phases of the code parsing, are written.

1.1 Parsing Tools

1.2 Internal Representation

BACKENDS

The Backends module packages the different backends implemented on the compiler.

The Common module contains common classes for all backends.

The DotBackend module is able to translate the *Internal Representation* (IR) to *Dot language*, which may be printed with graphviz.

The CBackend module contains writers capable of converting the IR to C or OpenMP code.

Module CudaBackend encapsulates Mutators, Visitors and Writers, capable of translating the IR to CUDA code.

2.1 Common

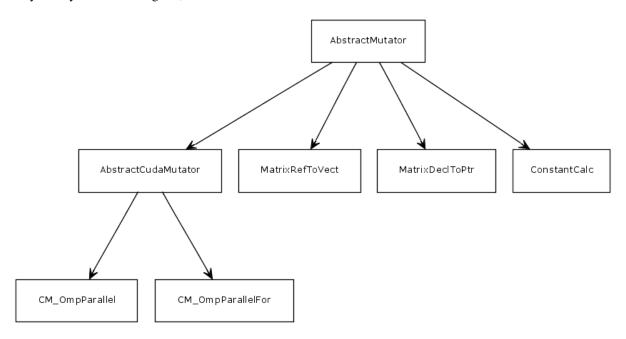
The Common module contains common operations to the different backends.

2.1.1 Mutators

Some *Mutator* are declared on the Mutators module.

AbstractMutator

As you may see on the diagram, all mutators inherits from AbstractMutator.



class AbstractMutator()

```
Abstract class representing a mutation.
     apply (ast, mutator_opt_arg=None)
          Apply a mutation
               Returns filtered node
      apply_all (ast, mutator_opt_arg=None)
          Apply mutation to all matches
               Returns pointer to last applied mutation
     fast_apply_all (ast)
          Apply mutation to all matches ignoring syntactic order
               Returns pointer to last applied mutation
     filter (ast)
          Calls to a simple filter
     filter_iterator (ast)
          Calls an iterable filter
     mutatorFunction(ast)
          Mutates the AST
               Returns Starting point of the mutation
AstSupport
Additional mutators are provided in order to easier the backend writing process.
class DeclsToParamsMutator()
     DeclsToParams
     convert (node)
          Transform a type declaration to a parameter declaration
     filter (ast)
          Filter definition Returns the first node matching with the filter
     mutatorFunction (ast)
          Mutator code
class FuncToDeviceMutator (func_call)
     Replace the definition of a FuncCall with a CUDAKernel with type __device__
      filter (ast)
          Find the declaration of the
class IDNameMutator (old, new)
     Replace and ID name with another ID name
class RemoveAttributeMutator (attr)
     Remove the child of the first apperance of an attribute inside a node
      apply (ast)
          Apply the mutation
```

2.1.2 Filter visitors

Warning: Mabye a name change is needed, SearchVisitor will be more clear.

Warning: Some cleaning needs to be done in this module.

In order to search nodes on the AST, the programmer needs to implements a *Filter*. GenericFilterVisitor is the parent of all filters, defining the common methods of all filters.

This module has other members, which are concrete implementations of the GenericFilterVisitor.

```
class GenericFilterVisitor (condition_func, prev_brother=None)
```

Returns the first node validating a condition function

```
apply (ast, ignore=, [])
```

Searchs the node matching the condition_func in the AST

Parameters

- ast Node to start search
- ignore List of nodes to ignore

Returns First matching node not in ignore list

```
dfs_iter(root, visited=None)
```

Given a starting node, root, do a depth-first search. .. warning:

This method does not garantee to transverse the tree on the gramatically correct order

Returns Last match or raise StopIteration

```
generic_visit (node, offset=0, ignore=, [])
```

Called if no explicit visitor function exists for a node. Implements preorder (syntactically correct) node visit.

param node Node being visited

param prev Syntactically previous node

param offset Not used

param ignore List of nodes to ignore

return Result of the node visit

iterate(ast)

Iterate through matching nodes

Parameter ast – Node to start the search

parentOfMatch()

Parent of the node matched

Note: This is not needed because we have a .parent attribute now

visit (node, prev, offset=1, ignore=, [])

Finds a way to visit a node

Parameters

- node Node to visit
- prev Syntactically previous node
- offset Not used
- ignore List of nodes to ignore

Returns result of the node visit

2.1. Common 7

2.2 Dot Backend

The DotBackend module contains a DotWriter, capable of produce Dot language code from the IR.

This backend is used in the Tools. Debug tool.

The DotWriter implements a visitor pattern.

```
class DotWriter (filename=None, highlight=None)
```

Generates Dot code from the IR

```
generic visit (node, offset=0)
```

Called if no explicit visitor function exists for a node. Implements preorder visiting of the node.

get_name (node)

Get the name of a node

Note: This method requires further explanation

Parameter *node* – Node to get the name

visit (node, offset=0)

Visit a node

write_highlights()

Write a label to highlight nodes

The nodes to highlight come from self.highlight

write label(node, name)

Write a label for a node

Parameters

- node Label of node
- name Name of the line

write_line (begin, name, end)

Write a connection between two nodes

Parameters

- begin Start node
- end End node
- name Name of the line

2.3 CBackend

The CBackend module contains two writers, CWriter and OmpWriter, which produces ${\bf C}$ and ${\bf OpenMP}$ code

Note: This backend is widely used on llCoMP.

class CWriter (filename=None, stream=None)

A visitor which translates the IR to plain C code

generic_visit_nodeList (nodeList, separator, offset)

Visit a list of nodes, writing values within a separator

class OmpWriter (filename=None, stream=None)

Visitor which translates the IR to C/OpenMP.

2.4 CUDA Backend

The CudaBackend module contains a set of Mutators, Filters and Templates which creates CUDA code from the IR.

2.4.1 Filters

```
class OmpForFilter (prev_brother=None)
```

Returns a OmpFor node, the parallel container and the function container

By defining specific visitor methods for *FuncDef* and *OmpParallel*, we can save the last node visited of this types. Giving the fact that the visit is done in syntax order, the last visited node will be the previous (parent) node of the wanted node.

```
iterate(ast)
```

Iterate through matching nodes

class OmpParallelFilter (condition_func=None, prev_brother=None, device=None)

Returns a OmpParallel node, the parallel container and the function container

By defining specific visitor methods for *FuncDef* and *OmpParallel*, we can save the last node visited of this types. Giving the fact that the visit is done in syntax order, the last visited node will be the previous (parent) node of the wanted node.

```
parallel_condition(node)
```

OmpParallel filter

```
visit_OmpTargetDevice (node, prev, offset=1, ignore=, [])
```

Save target device node

class OmpParallelForFilter (prev_brother=None, device=None)

Returns a omp parallel for construct

class OmpThreadPrivateFilter (prev_brother=None)

Returns the ThreadPrivate constructs

2.4.2 Templates

Currently, templates are held inside Mutators code.

2.4.3 Mutators

A separate Mutator have been written for each OpenMP construct. Their parent is Backends.CudaBackend.Mutators.Common

class AbstractCudaMutator (clauses=None, kernel_name='loopKernel', kernel_prefix=")

Common methods to work with CUDA

buildDeclarations (numThreads, reduction_node_list, shared_node_list, ast)

Builds the declaration section @param numThreads number of threads @return Declarations subtree

buildHostReduction (reduction_vars, ast)

Instanciate the reduction pattern

@return Compound with the reduction code

buildKernel (shared_list, private_list, reduction_list, loop, ast)

Build CUDA Kernel code

getThreadNum (node)

Gets the maximum number of threads needed

2.4. CUDA Backend 9

```
get_names (elem, ast)
```

Return a list of names for a type

The following constructs have been implemented:

OpenMP Parallel OpenMP Parallel For

```
{\bf class} \ {\bf CM\_OmpParallelFor} \ ({\it clauses=None, kernel\_name='loopKernel', kernel\_prefix=''})
```

This mutator locates a omp parallel for reduction, and then translate the original source to an equivalent cuda implementation

```
apply_all (ast)
```

Apply mutation to all matches

@return last node changed

filter(ast)

Filter definition

@return first node matching with the filter

mutatorFunction (ast, ompFor_node)

Main mutator for OpenMP Parallel For construct

Writes the optimized code of an OpenMP Parallel For construct, building a kernel overwriting the for loop.

OpenMP for

automodule:: Backends.CudaBackend.Mutators.CM_OmpFor members:

CHAPTER THREE

TOOLS

A set of tools have been developed in order to easier the work with the internal representation.

3.1 Tree Tools

Tree manipulation tools

3.2 Serialization Tools

3.3 Declarations Tools

Provide functionality to look for declarations on the AST

3.4 Debug Tools

12 Chapter 3. Tools

CHAPTER FOUR

GLOSSARY

Mutator A mutator is...

Filter A filter is...

Rest Rest is...

CHAPTER FIVE

INDICES AND TABLES

- Index
- Module Index
- Search Page

MODULE INDEX

```
В
Backends, 5
Backends.CBackend.Writers.CWriter, 8
Backends.CBackend.Writers.OmpWriter, 8
Backends.Common.Mutators.AbstractMutator,
Backends.Common.Mutators.AstSupport,6
Backends.Common.Visitors.GenericVisitors,
Backends.CudaBackend.Mutators.CM_OmpParallel,
Backends.CudaBackend.Mutators.CM_OmpParallelFor,
Backends.CudaBackend.Mutators.Common,
Backends.CudaBackend.Visitors.CM_Visitors,
Backends.DotBackend.Writers.DotWriter,
CBackend, 8
Common, 5
CudaBackend, 9
DotBackend, 8
Frontend, 3
Т
Tools, 11
```

18 Module Index

INDEX

A	C
AbstractCudaMutator (class in Back-	CBackend (module), 8
ends.CudaBackend.Mutators.Common),	CM_OmpParallelFor (class in Back-
9	ends.CudaBackend.Mutators.CM_OmpParallelFor),
AbstractMutator (class in Back-	10
ends.Common.Mutators.AbstractMutator),	Common (module), 5
5	convert() (Backends. Common. Mutators. Ast Support. Decls To Params Mutators) (Backends) (Common. Mutators) (Common. Mutators
apply () (Backends. Common. Mutators. Abstract Mutator. Abstra	
method), 6	CudaBackend (module), 9
apply() (Backends.Common.Mutators.AstSupport.Remov	
method), 6	ends.CBackend.Writers.CWriter), 8
apply() (Backends.Common.Visitors.GenericVisitors.Gen	nericFilterVisitor
method), 7	DAIL TO DAY () A C () D. 1
apply_all() (Backends.Common.Mutators.AbstractMutators.Ab	
method), 6	ends.Common.Mutators.AstSupport), 6
method), 10	Rhfsa <u>lled F() (BMk Թուսեր Parallad Frol</u> visitors. Generic Visitors. Generic Filter Visit method), 7
method), 10	DotBackend (module), 8
В	DotWriter (class in Back-
Backends (module), 5	ends.DotBackend.Writers.DotWriter),
Backends.CBackend.Writers.CWriter (module), 8	8
Backends.CBackend.Writers.OmpWriter (module), 8	
Backends.Common.Mutators.AbstractMutator (mod-	F
ule), 5	fast_apply_all() (Back-
Backends.Common.Mutators.AstSupport (module), 6	ends. Common. Mutators. Abstract Mutator. Abstract Mutator
Backends.Common.Visitors.GenericVisitors (module),	method), 6
7	Filter, 13
Backends.CudaBackend.Mutators.CM_OmpParallel	filter() (Backends.Common.Mutators.AbstractMutator.AbstractMutator
(module), 10	method), 6
$Backends. Cuda Backend. Mutators. CM_Omp Parallel For$	filter() (Backends.Common.Mutators.AstSupport.DeclsToParamsMutator
(module), 10	method), 6
Backends.CudaBackend.Mutators.Common (module),	filter() (Backends.Common.Mutators.AstSupport.FuncToDeviceMutator
9	method), 6 filter() (Backends.CudaBackend.Mutators.CM_OmpParallelFor.CM_Omp
Backends.CudaBackend.Visitors.CM_Visitors (mod-	method), 10
ule), 9	filter_iterator() (Back-
Backends.DotBackend.Writers.DotWriter (module), 8	ends.Common.Mutators.AbstractMutator.AbstractMutator
buildDeclarations() (Backends.CudaBackend.Mutators.Common.Abstrac	
method), 9	Frontend (module), 3
buildHostReduction() (Back-	FuncToDeviceMutator (class in Back-
ends CudaBackend Mutators Common Abstrac	etCudaMutands.Common.Mutators.AstSupport), 6
mathad) 0	
buildKernel() (Backends.CudaBackend.Mutators.Commo	on. AbstractCudaMutator
method), 9	generic_visit() (Back-
	ends.Common.Visitors.GenericVisitors.GenericFilterVisitor

method), 7

```
(Back- parentOfMatch()
                                                                                                    (Back-
generic_visit()
         ends.DotBackend.Writers.DotWriter.DotWriter
                                                               ends.Common.Visitors.GenericVisitors.GenericFilterVisitor
         method), 8
                                                               method), 7
generic_visit_nodeList()
                                             (Back-
                                                      R
         ends.CBackend.Writers.CWriter.CWriter
         method), 8
                                                      RemoveAttributeMutator
                                                                                   (class
                                                                                                    Back-
                                                                                             in
GenericFilterVisitor
                                              Back-
                         (class
                                     in
                                                               ends.Common.Mutators.AstSupport), 6
         ends.Common.Visitors.GenericVisitors),
                                                      Rest, 13
get_name() (Backends.DotBackend.Writers.DotWriter.DotWriter
         method), 8
                                                      Tools (module), 11
get_names() (Backends.CudaBackend.Mutators.Common.AbstractCudaMutator
         method), 9
getThreadNum()
                                             (Back-
         dNum() (Back-
ends.CudaBackend.Mutators.Common.AbstractCudaMutator,
method), 7
                                                      visit() (Backends.DotBackend.Writers.DotWriter.DotWriter
                                                               method), 8
I
                                                      visit_OmpTargetDevice()
IDNameMutator
                       (class
                                              Back-
                                    in
                                                               ends. Cuda Backend. Visitors. CM\_Visitors. Omp Parallel Filter
         ends.Common.Mutators.AstSupport), 6
                                                               method), 9
iterate() (Backends.Common.Visitors.GenericVisitors.GenericFilterVisitor
         method), 7
iterate() (Backends.CudaBackend.Visitors.CM_Visitors.OmpForFilter_write_nightights()
                                                                                                    (Back-
         method), 9
                                                               ends.DotBackend.Writers.DotWriter.DotWriter
                                                               method), 8
M
                                                      write_label() (Backends.DotBackend.Writers.DotWriter.DotWriter
Mutator, 13
                                                               method), 8
mutatorFunction()
                                             (Back-
                                                      write_line() (Backends.DotBackend.Writers.DotWriter.DotWriter
         ends.Common.Mutators.AbstractMutator.AbstractMutatormethod), 8
         method), 6
mutatorFunction()
                                             (Back-
         ends. Common. Mutators. Ast Support. Decls To Params Mutator\\
         method), 6
mutatorFunction()
                                             (Back-
         ends.CudaBackend.Mutators.CM OmpParallelFor.CM OmpParallelFor
         method), 10
0
OmpForFilter
                     (class
                                   in
                                              Back-
         ends.CudaBackend.Visitors.CM_Visitors),
OmpParallelFilter
                        (class
                                     in
                                              Back-
         ends.CudaBackend.Visitors.CM Visitors),
OmpParallelForFilter
                          (class
                                              Back-
                                      in
         ends.CudaBackend.Visitors.CM Visitors),
OmpThreadPrivateFilter\\
                            (class
         ends.CudaBackend.Visitors.CM Visitors),
OmpWriter
                    (class
                                   in
                                              Back-
         ends.CBackend.Writers.OmpWriter), 8
Р
parallel_condition()
                                             (Back-
         ends. Cuda Backend. Visitors. CM\_Visitors. Omp Parallel Filter
         method), 9
```

20 Index