NVIDIOSO

1.0

Generated by Doxygen 1.8.7

Fri Jul 11 2014 13:31:03

Contents

Chapter 1

Main Page

NVIDIOSO NVIDIa-based cOnstraint SOlver v. 1.0

```
__CSP/COP REPRESENTATION__
```

VARIABLES

Variable has variable types.

- bool: true, false
- int: -42, 0, 69
- set of int: {}, {2, 3, 4}, 1..10

We distinguish between four different types of variables, namely:

- FD Variables: standard Finite Domain variables
- SUP Variables: SUPport variable introduced to compute the objective function. These variables have unbounded int domains.
- OBJ Variables: OBJective variables. These variables store the objective value as calculated by the objective function through standard propagation. These variables have unbounded int domains.

DOMAINS:

Domain representation may vary depending on the type of model that is instantiated. In particular, for a CPU model the domains can be represented by lists of sets of domain value. For CUDA models domains are represented as follows. There are two internal representations for an finite domain D depending on whether |D| <= max_vector or not:

- Bitmap: if |D| <= max_vector;
- List of bounds: otherwise.

By default, max_vector is equal to 256. This value can be redefined via and environment variable VECTOR_MAX.

Domains have the following structure:

| EVT | REP | LB | UB | DSZ || ... BIT ... |

where

- EVT: represents the EVenT happened on the domain;
- REP: is the REPresentation currently used; This value can be one of the following:

  - -1, -2, -3, ...: BIT represents a set of 1, 2, 3, ... bitmaps respectively. Each bitmap represents a domain subset of values {LB, UB};
  - 0 : BIT represents a Bitmap of contiguous values;
  - 1, 2, 3, ... : in BIT there are respectively 0, 1, 2, ... lists of bound. If 0 the bounds are set as {LB, UB} in the LB/UB field respectively.
- LB: Lower Bound of the current domain;
- UB: Upper Bound of the current domain;
- DSZ: Domain SiZe where DSZ <= max_vector -> REP = 0. Moreover,
  - {LB, UB}' = {LB, k} {k', UB} -> DSZ' = DSZ - (k' - k + 1);
  - LB' = LB + k -> DSZ' = DSZ - (k - LB + 1);
  - UB' = UB - k -> DSZ' = DSZ - (UB - k + 1);
- BIT: bit vector where
  - REP $<$ 0: there is a total of ($<$=) VECTOR_MAX bits representing REP pairs of bounds. The first part of BIT is used to store REP triples $<$LB, Size, Offset$>$ where Offset is a pointer to the first bit of the bitmap representing the pair {LB, LB + Size}. The second part of BIT stores the actual bitmaps.
  - REP = 0: there are UB - LB + 1 $<$= VECTOR_MAX bits of contiguous domain values;
  - REP $>$ 0: each pair of bound is identified as LB, UB (LB = UB if singlet).

OBSERVATIONS (CUDA implementation):

Shared Memory: 49152 = 48 kB per block -> keep 47 kB available.

- REP $<$ 0 there are 47 $*$ 1024 = 48128 -> (48128 - 5 $*$ 32)/32 = 1499 possible storable values. Worst case: REP = -256 -> 3 $*$ 256 triples = 3 $*$ 256 = 768 $<$ 1499 (-8=256/32).
- REP = 0 and VECTOR_MAX = 4096 the worst case is when there are 4096 sing.: ((4096 + 4096 $*$ 2 $*$ 32) / 8) / 1024 = 32.5 kB $<$ 45 kB ((tot_bits + tot_bits $*$ 2 int $*$ bit_per_int) / B) / kB.
- REP $>$ 0: 45 kB = 11520 int -> 11520 - 5 = 11515 -> 11515/2 (used two int to represent a pair of bounds) = 5757 pairs separated by at least one "hole" from each other -> 5757 $*$ 2 = 11514 such as {0, 1}, {3, 4}, ...

It means that when the domains are greater than 11514 then a check must be performed in order to apply multiple copies from global to share memory if needed.

Chapter 2

NVIDIOSO

NVIDIOSO - NVIDIa-based cOnstraint SOlver v. 1.0

Chapter 3

Todo List

Member BoolDomain::get_event () const

implement this function

Member Parser::_map_tokens

use template for key.

Member SetDomain::get_event () const

implement this function

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 5

Class Index

5.1 Class List

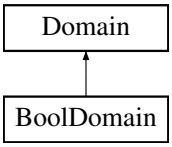
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 6

Class Documentation

6.1 BoolDomain Class Reference

Inheritance diagram for BoolDomain:



Public Member Functions

- DomainPtr clone () const

  Clone the current domain and returns a pointer to it.
- EventType get_event () const
- size_t get_size () const

  Returns the size of the domain.
- bool is_empty () const

  Returns true if the domain is empty.
- bool is_singleton () const

  Returns true if the domain has only one element.
- void print () const

  Print info about the domain.

Protected Member Functions

- DomainPtr clone_impl () const

  Clone the current domain.

Protected Attributes

- BoolValue _bool_value

  Current domain value.

Additional Inherited Members

6.1.1 Member Function Documentation

6.1.1.1 EventType BoolDomain::get_event ( ) const [virtual]

Get event on this domain

Todo implement this function

Implements Domain.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/bool_domain.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/bool_domain.cpp

6.2 Constraint Class Reference

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/constraint.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/constraint.cpp

6.3 ConstraintStore Class Reference

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/constraint_store.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/constraint_store.cpp

6.4 CPModel Class Reference

Public Member Functions

- virtual void add_variable (VariablePtr ptr)
- virtual void add_constraint (ConstraintPtr ptr)
- virtual void add_search_engine (SearchEnginePtr ptr)

Protected Attributes

- std::list< VariablePtr > _variables

  Variables.
- ConstraintPtr _constraint_store

  Constraint Store.
- SearchEnginePtr _search_engine

  Search engine.

6.4.1 Member Function Documentation

6.4.1.1 void CPModel::add_constraint (ConstraintPtr *ptr*) [virtual]

Add a constraint to the model. It linkes constraints to variables, actually defining the constraint graph.

Parameters

ptr	pointer to the constraint to add to the model

6.4.1.2 void CPModel::add_search_engine (SearchEnginePtr *ptr*) [virtual]

Add a search engine to the model.

Parameters

ptr	pointer to the search engine to use to explore the search space.

6.4.1.3 void CPModel::add_variable (VariablePtr *ptr*) [virtual]

Add a variable to the model. It linkes variables to constraints, actually defining the constraint graph.

Parameters

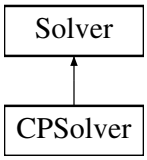
ptr	pointer to the variable to add to the model

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cp_model.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cp_model.cpp

6.5 CPSolver Class Reference

Inheritance diagram for CPSolver:



Public Member Functions

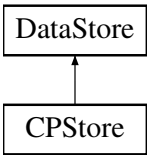
- void **run** ()

The documentation for this class was generated from the following file:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cp_solver.h

6.6 CPStore Class Reference

Inheritance diagram for CPStore:



Public Member Functions

- virtual bool load_model (std::string="")

  Load model from input file (FlatZinc model)
- virtual void init_model ()
- virtual void print_model_info ()

  Print info about the model.
- virtual void **print_model_variable_info** ()
- virtual void **print_model_domain_info** ()
- virtual void **print_model_constraint_info** ()

Static Public Member Functions

- static CPStore * get_store (std::string in_file)

  Constructor get (static) instance.

Protected Member Functions

- CPStore (std::string)

  Protected constructor for singleton pattern.

Additional Inherited Members

6.6.1 Member Function Documentation

6.6.1.1 void CPStore::init_model ( ) [virtual]

Init store with the loaded model. This method works on the internal state of the store. It uses a generator to generate the right instances of the objects (e.g. CUDA-FD variabes) and add them to the model. A generator takes tokens as input and returns the corresponding pointer to the instantiated objects.

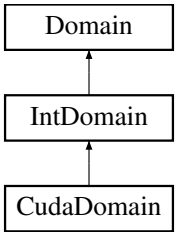
Implements DataStore.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cp_store.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cp_store.cpp

6.7 CudaDomain Class Reference

Inheritance diagram for CudaDomain:



Public Member Functions

- DomainPtr clone () const

  Clone the current domain and returns a pointer to it.
- void init_domain (int min, int max)
- size_t get_allocated_bytes () const
- EventType get_event () const

  Get event on the current domain.
- CudaDomainRepresenation get_representation () const

  Get current domain representation (i.e., bitmap or list)
- size_t get_size () const
- void set_bounds (int min, int max)
- bool set_singleton (int)

  Set domain as singleton.
- bool subtract (int)

  Subtract the element from the domain (see int_domain.h)
- void add_element (int val)
- void in_min (int)

  Increase the lower_bound (see int_domain.h)
- void in_max (int)

  Decrease the upper_bound (see int_domain.h)
- void print () const

  Print info about domain.

Static Public Member Functions

- static constexpr int EVT_IDX ()
- static constexpr int **REP_IDX** ()
- static constexpr int **LB_IDX** ()
- static constexpr int **UB_IDX** ()
- static constexpr int **DSZ_IDX** ()
- static constexpr int **BIT_IDX** ()

Additional Inherited Members

6.7.1 Member Function Documentation

6.7.1.1 void CudaDomain::add_element (int *val*) [virtual]

Add an element to the current domain (see int_domain.h).

Note

if the element is out of the initial bounds, no element will be added, i.e., the domains mantain the original size.

Implements IntDomain.

6.7.1.2 static constexpr int CudaDomain::EVT_IDX ( ) [inline], [static]

Constants used to retrieve the current domain description. Domain represented as: | EVT | REP | LB | UB | DSZ || ... BIT ... |. See system_description.h.

6.7.1.3 size_t CudaDomain::get_allocated_bytes ( ) const

Get the number of allocated bytes needed for representing the current domain w.r.t. its lower and upper bounds.

Returns

the number of allocated bytes.

6.7.1.4 size_t CudaDomain::get_size ( ) const [virtual]

Get domain size. It returns the currenst size of the domain, checking whether there are "holes" according to the current representation of the domain (i.e., bitmap or list):

Returns

the current domain's size.

Implements IntDomain.

6.7.1.5 void CudaDomain::init_domain (int *min*, int *max*) [virtual]

Initializes domain with default values:

- Event: no event;
- Representation: list or bitmap according to [min, max];
- Lower bound: min;
- Upper bound: max;
- Size: $|max - min + 1|$ or MAX_INT if max = MAN_INT()/2 and min = MIN_INT() / 2, etc..

Note

It instantiate an array of ints of at most MAX_BYTES_SIZE.

Parameters

min	lower bound of the domain
max	upper bound of the domain

Returns

it fails whenever consistency check on min/max fails (i.e., max $<$ min).

Implements IntDomain.

6.7.1.6 void CudaDomain::set_bounds (int *min*, int *max*) [virtual]

It specializes the parent method in order to set up the array of (int) values. It istantiates a domain [min, max]. This actually updates the bounds and it performs consistency checking and updating of the domain size.

Parameters

min	lower bound
max	upper bound

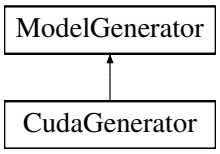
Implements IntDomain.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cuda_domain.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cuda_domain.cpp

6.8 CudaGenerator Class Reference

Inheritance diagram for CudaGenerator:



Public Member Functions

- VariablePtr get_variable (TokenPtr)

  See "model_generator.h".
- ConstraintPtr get_constraint (TokenPtr)

  See "model_generator.h".
- SearchEnginePtr get_search_engine (TokenPtr)

  See "model_generator.h".

Protected Attributes

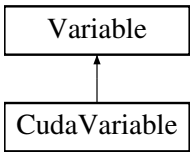
- std::string **_dbg**

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cuda_model_generator.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cuda_model_generator.cpp

6.9 CudaVariable Class Reference

Inheritance diagram for CudaVariable:



Public Member Functions

- CudaVariable ()
- CudaVariable (int idv)
- void set_domain ()
- void set_domain (int lw, int ub)
- void set_domain (std::vector< int > elems)
- void print () const

  print info about the current domain

Additional Inherited Members

6.9.1 Constructor & Destructor Documentation

6.9.1.1 CudaVariable::CudaVariable ( )

Base constructor: create a variable with new id. The id is given by a global id generator.

6.9.1.2 CudaVariable::CudaVariable (int *idv*)

One parameter constructor: create a variable with a given id.

Parameters

idv	identifier to give to the variable

6.9.2 Member Function Documentation

6.9.2.1 void CudaVariable::set_domain ( )

Set domain's bounds. If no bounds are provided, an unbounded domain (int) is istantiated. If an array of elements A is provided, the function instantiates a domain D = [min A, max A], deleting all the elements d in D s.t. d does not belong to A.

6.9.2.2 void CudaVariable::set_domain (int *lw,* int *ub*)

Set domain's bounds. A new domain [lw, ub] is generated.

Parameters

lw	lower bound
ub	upper bound

6.9.2.3 void CudaVariable::set_domain (std::vector< int > *elems*)

Set domain's elements. A domain {d_1, ..., d_n} is generated.

Parameters

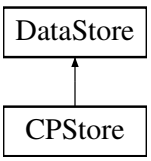
elems	vector of domain's elements

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cuda_variable.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/cuda_variable.cpp

6.10 DataStore Class Reference

Inheritance diagram for DataStore:



Public Member Functions

- virtual bool load_model (std::string="")=0
- virtual void init_model ()=0

  Init model using the information read from files.
- virtual void print_model_info ()=0

  Print info about the model.
- virtual CPModel * get_model ()

  Get the instantiated model.
- virtual void **print_model_variable_info** ()
- virtual void **print_model_domain_info** ()
- virtual void **print_model_constraint_info** ()

Protected Member Functions

- DataStore (std::string in_file)

Protected Attributes

- bool **_timer**
- bool **_verbose**
- std::string **_dbg**
- std::string **_in_file** = ""
- CPModel * _cp_model

  CP Model.

6.10.1 Constructor & Destructor Documentation

6.10.1.1 DataStore::DataStore (std::string *in_file*) [protected]

Constructor.

Parameters

in_file	file path of the model to parse.

6.10.2 Member Function Documentation

6.10.2.1 virtual bool DataStore::load_model (std::string = " ") [pure virtual]

Load model from input file (FlatZinc model).

Note

: the model described as a set of tokens is stored in the Tokenization class used by the parser. The parser has access to the set of tokens and it manages them in order to retrieve the correct set of tokens to initialize variables, and constraints. See Parser interface.

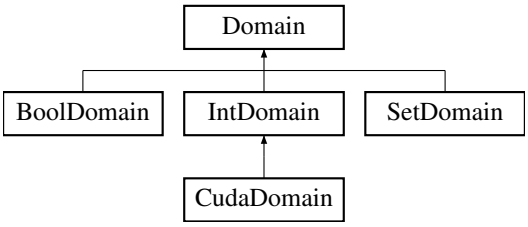
Implemented in CPStore.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/data_store.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/data_store.cpp

6.11 Domain Class Reference

Inheritance diagram for Domain:



Public Member Functions

- void set_type (DomainType dt)
- DomainType **get_type** () const
- virtual DomainPtr clone () const =0

  Clone the current domain and returns a pointer to it.
- virtual EventType get_event () const =0

  Get the current event on the domain.
- virtual size_t get_size () const =0

  Returns the size of the domain.
- virtual bool is_empty () const =0

  Returns true if the domain is empty.
- virtual bool is_singleton () const =0

  Returns true if the domain has only one element.
- virtual void print () const =0

  Print info about the current domain.

Static Public Member Functions

- static constexpr int MIN_DOMAIN ()

  Constants for int min/max domain bounds.
- static constexpr int MAX_DOMAIN ()

  Constants for int min/max domain bounds.

Protected Attributes

- std::string **_dbg**
- DomainType **_dom_type**

6.11.1 Member Function Documentation

6.11.1.1 void Domain::set_type (DomainType *dt*)

Set domain's type (use get_type to get the type).

Parameters

dt	domain type of type DomainType

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/domain.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/domain.cpp

6.12 FactoryModelGenerator Class Reference

Static Public Member Functions

- static ModelGenerator * get_generator (GeneratorType gt)

  Get the right instance of a generator based on the input.

The documentation for this class was generated from the following file:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/factory_generator.h

6.13 FactoryParser Class Reference

Static Public Member Functions

- static Parser * get_parser (ParserType pt)

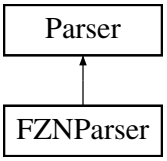
  Get the right parser based on the input.

The documentation for this class was generated from the following file:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/factory_parser.h

6.14 FZNParser Class Reference

Inheritance diagram for FZNParser:



Public Member Functions

- **FZNParser** (std::string ifile)
- bool more_variables () const

Ask whether there are more variables to get.

- bool more_constraints () const

  Ask whether there are more constraits to get.

- bool more_search_engines () const

  Ask whether there are more search engines to get.

- TokenPtr get_variable ()
- TokenPtr get_constraint ()
- TokenPtr get_search_engine ()
- TokenPtr get_next_content ()

  Get next (pointer to) token (i.e., FlatZinc element)

- void print () const

  Print info about the parser.

Additional Inherited Members

6.14.1 Member Function Documentation

6.14.1.1 TokenPtr FZNParser::get_constraint ( ) [virtual]

Get a "constraint" token.

Returns

token pointer to a "constraint" token.

Implements Parser.

6.14.1.2 TokenPtr FZNParser::get_next_content ( ) [virtual]

Get next (pointer to) token (i.e., FlatZinc element)

Set position on file to the most recent position

Implements Parser.

6.14.1.3 TokenPtr FZNParser::get_search_engine ( ) [virtual]

Get a "search_engine" token.

Returns

token pointer to a "search_engine" token.

Implements Parser.

6.14.1.4 TokenPtr FZNParser::get_variable ( ) [virtual]

Get a "variable" token.

Returns

token pointer to a "variable" token.

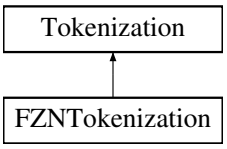
Implements Parser.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/fzn_parser.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/fzn_parser.cpp

6.15 FZNTokenization Class Reference

Inheritance diagram for FZNTokenization:



Public Member Functions

- TokenPtr get_token ()

Additional Inherited Members

6.15.1 Member Function Documentation

6.15.1.1 TokenPtr FZNTokenization::get_token ( ) [virtual]

Specialized method: It actually gets the right token according to the FlatZinc format. Analysis is perfomed on "_c_token".

Implements Tokenization.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/fzn_tokenization.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/fzn_tokenization.cpp

6.16 IdGenerator Class Reference

Public Member Functions

- void reset_int_id ()

  Reset id generator.
- void reset_str_id ()

  Reset id generator.
- void set_base_offset (int)

  Set (base) ids (if not already set)
- void set_base_prefix (std::string)

  Set (base) ids (if not already set)
- int **get_int_id** ()
- std::string **get_str_id** ()
- int **new_int_id** ()
- std::string **new_str_id** ()
- int **curr_int_id** ()
- std::string **curr_str_id** ()
- void **print_int_id** ()
- void **print_str_id** ()

Static Public Member Functions

- static IdGenerator * get_instance ()
  Constructor get (static) instance.

Protected Member Functions

- IdGenerator ()
- std::string n_to_str (int)
  Convert numbers to string.

6.16.1 Constructor & Destructor Documentation

6.16.1.1 IdGenerator::IdGenerator ( ) [protected]

Protected constructor: a client cannot instantiate Singleton directly.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/id_generator.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/id_generator.cpp

6.17 InputData Class Reference

Public Member Functions

- bool **verbose** () const
- bool **timer** () const
- int **max_n_sol** () const
- std::string get_in_file () const
  Get input file (path to)
- std::string get_out_file () const
  Get output file (path to)

Static Public Member Functions

- static InputData * get_instance (int argc, char *argv[])
  Constructor get (static) instance.

Protected Member Functions

- InputData (int argc, char *argv[])

6.17.1 Constructor & Destructor Documentation

6.17.1.1 InputData::InputData (int *argc*, char * *argv[]*) [protected]

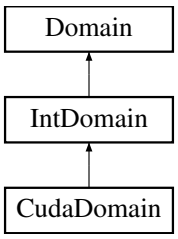
Protected constructor: a client cannot instantiate Singleton directly. Exit if the user did not set an input file!

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/input_data.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/input_data.cc

6.18 IntDomain Class Reference

Inheritance diagram for IntDomain:



Public Member Functions

- bool is_singleton () const

  Returns true if the domain has only one element.
- bool is_empty () const

  Returns true if the domain is empty.
- virtual int get_lower_bound () const

  Get the domain's lower bound.
- virtual int get_upper_bound () const

  Get the domain's upper bound.
- virtual void init_domain (int min, int max)=0
- virtual void set_bounds (int min, int max)=0
- virtual size_t get_size () const =0
- virtual EventType get_event () const =0

  Get the current event on the domain.
- virtual bool set_singleton (int val)=0
- virtual bool subtract (int val)=0
- virtual void add_element (int val)=0
- virtual void in_min (int min)=0
- virtual void in_max (int max)=0
- void print () const =0

  Print info about the domain.

Protected Attributes

- int **_lower_bound**
- int **_upper_bound**

Additional Inherited Members

6.18.1 Member Function Documentation

6.18.1.1 virtual void IntDomain::add_element (int *val*) [pure virtual]

It computes the union of the current domain with the domain represented by the singleton element given in input to the method. If the element is out of [lower_bound, upper_bound] it enlarges the domain.

Parameters

val	element to add to the current domain.

Implemented in CudaDomain.

6.18.1.2 virtual size_t IntDomain::get_size ( ) const [pure virtual]

Returns the size of the domain. This function should be implemented by derived classes according to their internal domain representation.

Note

upper_bound - lower_bound + 1 could not be the actual size of the domain.

Returns

the current domain's size.

Implements Domain.

Implemented in CudaDomain.

6.18.1.3 virtual void IntDomain::in_max (int *max*) [pure virtual]

It updates the domain according to the maximum value.

Parameters

max	domain value.

Implemented in CudaDomain.

6.18.1.4 virtual void IntDomain::in_min (int *min*) [pure virtual]

It updates the domain according to the minimum value.

Parameters

min	domain value.

Implemented in CudaDomain.

6.18.1.5 virtual void IntDomain::init_domain (int *min*, int *max*) [pure virtual]

Initialize domain: this function is used to set up the domain as soon it is created. Classes that derive IntDomain specilize this method according to their internal representation of domain.

Implemented in CudaDomain.

6.18.1.6 virtual void IntDomain::set_bounds (int *min*, int *max*) [pure virtual]

Set domain's bounds. It updates the domain to have values only within the interval min..max.

Note

it does not update _lower_bound and _upper_bound here for efficiency reasons.

Parameters

lower	lower bound value
upper	upper bound value

Implemented in CudaDomain.

6.18.1.7 virtual bool IntDomain::set_singleton (int *val*) [pure virtual]

Set domain to the singleton element given in input.

Parameters

val	the value to set as singleton

Returns

true if the domain has been set to singleton, false otherwise.

Implemented in CudaDomain.

6.18.1.8 virtual bool IntDomain::subtract (int *val*) [pure virtual]

It intersects with the domain which is a complement of the value given as input, i.e., subtract a value from the current domain.

Parameters

val	the value to subtract from the current domain

Returns

true if succeed, false otherwise.

Implemented in CudaDomain.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/int_domain.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/int_domain.cpp

6.19 Logger Class Reference

Public Member Functions

- void **set_out_file** (std::string)
- void **set_verbose** (bool)
- void message (std::string)

  Print message on stdout or file (print_message force printing)
- void **print_message** (std::string)
- void log (std::string)

  Print log on stdout or file.
- void **oflog** (std::string)
- void error (std::string)

  *Print error message on cerr (optional: **FILE** and **LINE**)*
- void **error** (std::string, const char *)
- void **error** (std::string, const char *, const int)

Static Public Member Functions

- static Logger * get_instance (std::string log_file="")

  Constructor get (static) instance.

Protected Member Functions

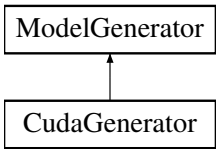
- **Logger** (std::string="")

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/logger.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/logger.cpp

6.20 ModelGenerator Class Reference

Inheritance diagram for ModelGenerator:



Public Member Functions

- virtual VariablePtr get_variable (TokenPtr)=0
- virtual ConstraintPtr get_constraint (TokenPtr)=0
- virtual SearchEnginePtr get_search_engine (TokenPtr)=0

6.20.1 Member Function Documentation

6.20.1.1 virtual ConstraintPtr ModelGenerator::get_constraint (TokenPtr) [pure virtual]

These methods create the instances of the objects and return the correspondent (shared) pointers to them.

Parameters

TokenPtr	pointer to the token describing a constraint. If the token does not correspond to the object to instantiate, it returns nullptr.

Implemented in CudaGenerator.

6.20.1.2 virtual SearchEnginePtr ModelGenerator::get_search_engine (TokenPtr) [pure virtual]

These methods create the instances of the objects and return the correspondent (shared) pointers to them.

Parameters

TokenPtr	pointer to the token describing a search engine. If the token does not correspond to the object to instantiate, it returns nullptr.

Implemented in CudaGenerator.

6.20.1.3 virtual VariablePtr ModelGenerator::get_variable (TokenPtr) [pure virtual]

These methods create the instances of the objects and return the correspondent (shared) pointers to them.

Parameters

TokenPtr	pointer to the token describing a variable. If the token does not correspond to the object to instantiate, it returns nullptr.

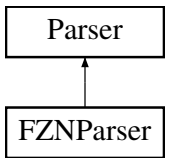
Implemented in CudaGenerator.

The documentation for this class was generated from the following file:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/model_generator.h

6.21 Parser Class Reference

Inheritance diagram for Parser:



Public Member Functions

- void set_input (std::string)

  Set input.
- void add_delimiter (std::string)

  Add delimiter to tokenizer.
- int get_current_line ()

  Get current (parsed) line.
- bool is_failed () const

  Check whether the parser has failed.
- virtual bool more_tokens ()
- virtual void open ()
- virtual void close ()
- virtual std::string get_next_token ()
- virtual bool more_variables () const =0
- virtual bool **more_constraints** () const =0
- virtual bool **more_search_engines** () const =0
- virtual TokenPtr get_variable ()=0
- virtual TokenPtr **get_constraint** ()=0
- virtual TokenPtr **get_search_engine** ()=0
- virtual TokenPtr get_next_content ()=0
- virtual void print () const =0

  Print info.

Protected Member Functions

- Parser ()

  Constructor.
- **Parser** (std::string)

Protected Attributes

- Tokenization * _tokenizer

  Tokenizer: it tokenizes lines read from the input file.
- std::ifstream * _if_stream

  Input stream (from file)
- std::string **_input_path**
- std::string **_dbg**
- bool **_open_file**
- bool **_open_first_time**
- bool **_more_tokens**
- bool **_new_line**
- bool **_failure**
- int _current_line

  Number of lines read so far.
- std::string _delimiters

  Delimiter to use to tokenize words.
- std::streampos _curr_pos

  Other variables needed to move into the file.
- std::map< size_t, TokenPtr > _map_tokens

6.21.1 Member Function Documentation

6.21.1.1 void Parser::close ( ) [virtual]

Close the file.

Note

: alternating open() and close() the client can decided how much text has to be parsed.

6.21.1.2 virtual TokenPtr Parser::get_next_content ( ) [pure virtual]

Give next Token. A Token is built from a (string) token and represents a semantic object read from the FlatZinc model given in input. It holds other useful info related to the (string) token itself, e.g., line where the token has been found. If this function is call and no other Token is available it returns nullprt.

Implemented in FZNParser.

6.21.1.3 std::string Parser::get_next_token ( ) [virtual]

Get next token. This function returns a string corresponding to the token parsed according to the internal state of the object (i.e., pointer in the text file).

6.21.1.4 virtual TokenPtr Parser::get_variable ( ) [pure virtual]

Get methods: get variables, constraints, and the search engine. They increment the counter of available tokens. The tokens are returned in order w.r.t. their variables.

Implemented in FZNParser.

6.21.1.5 bool Parser::more_tokens ( ) [virtual]

Check if the internal status has more tokens to give back to the client.

6.21.1.6 virtual bool Parser::more_variables ( ) const [pure virtual]

Get methods: more tokens of the same related type (i.e., variables, constraints, and search engine). These methods should be used together with the "get" methods.

Implemented in FZNParser.

6.21.1.7 void Parser::open ( ) [virtual]

Open the file. The file is open (if not already open) and the pointer is placed on the last position read. If the file is open for the first time, the pointer is placed on the first position.

6.21.2 Member Data Documentation

6.21.2.1 std::map< size_t, TokenPtr > Parser::_map_tokens [protected]

Pointers to all tokens parsed so far.

Todo use template for key.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/parser.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/parser.cpp

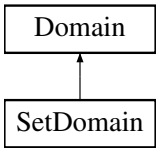
6.22 SearchEngine Class Reference

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/search_engine.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/search_engine.cpp

6.23 SetDomain Class Reference

Inheritance diagram for SetDomain:



Public Member Functions

- virtual void set_values (std::vector< int > elems)
- virtual std::vector< int > get_values () const
- DomainPtr clone () const

  Clone the current domain and returns a pointer to it.
- EventType get_event () const
- size_t get_size () const

  Returns the size of the domain.
- bool is_empty () const

  Returns true if the domain is empty.
- bool is_singleton () const

  Returns true if the domain has only one element.
- void print () const

  Print info about the domain.

Protected Member Functions

- DomainPtr **clone_impl** () const

Protected Attributes

- std::vector< int > **_d_elements**

Additional Inherited Members

6.23.1 Member Function Documentation

6.23.1.1 EventType SetDomain::get_event ( ) const [virtual]

Get event on this domain

Todo implement this function

Implements Domain.

6.23.1.2 std::vector< int > SetDomain::get_values ( ) const [virtual]

Get a vector containing the current values contained in the domain.

Returns

the current elements in the domain

6.23.1.3 void SetDomain::set_values (std::vector< int > *elems*) [virtual]

Set bounds and perform some consistency checking. It throws "no solutions" if consistency checking fails.

Parameters

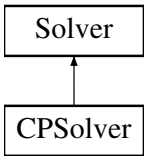
elems	vector of domain's elements

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/set_domain.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/set_domain.cpp

6.24 Solver Class Reference

Inheritance diagram for Solver:



Public Member Functions

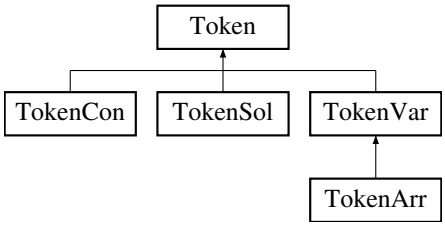
- virtual void **run** ()=0

The documentation for this class was generated from the following file:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/solver.h

6.25 Token Class Reference

Inheritance diagram for Token:



Public Member Functions

- **Token** (TokenType)
- int **get_id** () const
- void **set_type** (TokenType)
- TokenType **get_type** () const
- virtual void print () const

  Print info about the token.

Protected Attributes

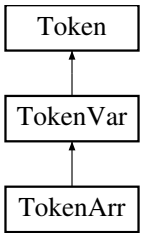
- std::string **_dbg**
- TokenType **_tkn_type**

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token.cpp

6.26 TokenArr Class Reference

Inheritance diagram for TokenArr:



Public Member Functions

- void **set_size_arr** (int)
- int **get_size_arr** () const
- void set_array_bounds (int lw, int up)
- int **get_lw_bound** () const
- int **get_up_bound** () const
- int get_lower_var () const
- int get_upper_var () const
- bool is_var_in (int var) const
- bool **is_var_in** (std::string) const
- void set_output_arr ()

  Identifies the current variable array as a support variable array.
- bool **is_output_arr** () const
- void print () const

  Print info methods.

Additional Inherited Members

6.26.1 Member Function Documentation

6.26.1.1 int TokenArr::get_lower_var ( ) const

Variables (idx) within the array. The index is given w.r.t. the global index of parsed tokens so far.

Returns

the lower idx of variable within the array

6.26.1.2 int TokenArr::get_upper_var ( ) const

Variables (idx) within the array. The index is given w.r.t. the global index of parsed tokens so far.

Returns

the higher idx of variable within the array

6.26.1.3 bool TokenArr::is_var_in (int *var*) const

Check whether a given variable (idx) is indexed by the array (i.e., is whithin the array.

Note

: check is performed w.r.t. both the variable string identifier (e.g., a[i]) and its global id.

Parameters

var	the variable to check membership

Returns

true if var is in the current array, false otherwise

6.26.1.4 void TokenArr::set_array_bounds (int *lw,* int *up*)

Array set and info. For example, array [1..30] of ... get_lw_bound -> 1 get_lw_bound -> 30 It sets the bounds of the array.

Parameters

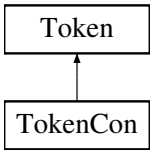
lw	lower bound
up	upper bound

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_arr.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_arr.cpp

6.27 TokenCon Class Reference

Inheritance diagram for TokenCon:



Public Member Functions

- void set_con_id (std::string)

  Get/set methods.
- std::string **get_con_id** () const

- void add_expr (std::string str)
- int get_num_expr () const

  Get the number of parameters needed by the constraint.
- std::string get_expr (int) const
- const std::vector< std::string > get_expr_array ()
- virtual void print () const

  Print info methods.

Protected Attributes

- std::string _con_id

  Info about the constraint.
- std::vector< std::string > _exprs

  Parameters involved in the constraint.

6.27.1 Member Function Documentation

6.27.1.1 void TokenCon::add_expr (std::string *str*)

Add expression (parameters) to the token that identifies the parsed constraint. For example, constraint int_↩ ne(magic[1], magic[2]) expression = "magic[1]" and "magic[2]"

Parameters

str	string representing the expression.

6.27.1.2 std::string TokenCon::get_expr (int *idx*) const

Get the string represeting the ith expression that defines the constraint.

Parameters

idx	index of the expression to return

Returns

return the idx^th expression

6.27.1.3 const std::vector< std::string > TokenCon::get_expr_array ( )

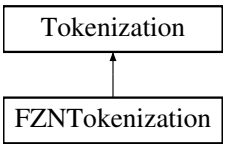
Return an array containing all the (string) expressions that define the current constraint.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_con.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_con.cpp

6.28 Tokenization Class Reference

Inheritance diagram for Tokenization:



Public Member Functions

- void **add_delimiter** (std::string)
- void **set_delimiter** (std::string)
- void **add_white_spaces** (std::string)
- void **set_white_spaces** (std::string)
- void set_new_tokenizer (std::string line)
- bool find_new_line ()

  Informs whether a new line has been found.
- bool is_failed () const

  Check whether the tokenizer has failed.
- bool need_line ()

  Asks whether the tokenizer has finished all the tokens.
- void add_comment_symb (char)

  Set preferences.
- void **add_comment_symb** (std::string)
- virtual TokenPtr get_token ()=0

  Get the string correspondent to the (filtered) token.

Protected Member Functions

- virtual bool avoid_char (char)

  It states whether the current char has to be skipped or not.
- virtual bool skip_line ()

  It states whether_c_token or the a line have to be skipped or not.
- virtual bool **skip_line** (std::string)
- virtual bool set_new_line ()
- virtual void clear_line ()
- virtual TokenPtr analyze_token ()=0

Protected Attributes

- std::string **_dbg**
- std::string **DELIMITERS** = "\t\r\n "
- std::string **WHITESPACE** = " \t"
- std::string **_comment_lines**
- bool **_new_line**
- bool **_need_line**
- bool **_failed**
- char * _c_token

  Token returned by strtok.
- char * _parsed_line

  Parsed line.

6.28.1 Member Function Documentation

6.28.1.1 virtual TokenPtr Tokenization::analyze_token ( ) [protected], [pure virtual]

Analyze token: this function acts like a filter. It analyzes _c_token and returns a string corresponding to the token cleaned from useless chars.

6.28.1.2 void Tokenization::clear_line ( ) [protected], [virtual]

It "clears" the text line by removing possible initial white spaces from line. Different heuristics may be used here.

6.28.1.3 bool Tokenization::set_new_line ( ) [protected], [virtual]

It states whether a new line has been found. Different heuristics may be used here.

6.28.1.4 void Tokenization::set_new_tokenizer (std::string *line*)

Prepare a new tokenizer (i.e., string for strtok).

Parameters

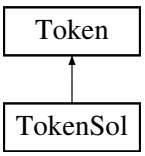
line	the string to tokenize.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/tokenization.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/tokenization.cpp

6.29 TokenSol Class Reference

Inheritance diagram for TokenSol:



Public Member Functions

- void **set_var_goal** (std::string)
- void **set_solve_goal** (std::string)
- void **set_solve_params** (std::string)
- void **set_label_choice** (std::string)
- void **set_search_choice** (std::string)
- void **set_variable_choice** (std::string)
- void **set_assignment_choice** (std::string)
- void **set_strategy_choice** (std::string)
- void set_var_to_label (std::string)

  Set the (string) identifier of a variable to label.
- std::string **get_var_goal** () const
- std::string **get_solve_goal** () const

- std::string **get_search_choice** () const
- std::string **get_label_choice** () const
- std::string **get_variable_choice** () const
- std::string **get_assignment_choice** () const
- std::string **get_strategy_choice** () const
- int num_var_to_label () const
- const std::vector< std::string > get_var_to_label () const
- std::string get_var_to_label (int idx) const
- virtual void print () const

  Print info methods.

Protected Attributes

- std::string **_var_goal**
- std::string **_solve_goal**
- std::string **_search_choice**
- std::string **_label_choice**
- std::string **_variable_choice**
- std::string **_assignment_choice**
- std::string **_strategy_choice**
- std::vector< std::string > _var_to_label

6.29.1 Member Function Documentation

6.29.1.1 const vector< std::string > TokenSol::get_var_to_label ( ) const

Identifiers of the variables to label.

Returns

a vector of string identifiers of the variable to label during the search phase.

6.29.1.2 string TokenSol::get_var_to_label (int *idx*) const

Get the string corresponding to the ith variable to label.

Parameters

idx	the index of the variable to label.

Returns

the string identifier of the idx^th variable to label.

6.29.1.3 int TokenSol::num_var_to_label ( ) const

Number of variables to label if specified by the model.

Returns

the number of variables to label.

6.29.2 Member Data Documentation

6.29.2.1 std::vector< std::string > TokenSol::_var_to_label [protected]

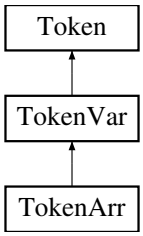
Vector of strings corresponding to the variables to label during the search phase.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_sol.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_sol.cpp

6.30 TokenVar Class Reference

Inheritance diagram for TokenVar:



Public Member Functions

- void set_var_id (std::string str)
- std::string **get_var_id** () const
- void set_objective_var ()

  Identifies the current variable as an objective variable.
- bool **is_objective_var** () const
- void set_support_var ()

  Identifies the current variable as a support variable.
- bool **is_support_var** () const
- void set_var_dom_type (VarDomainType vdt)
- VarDomainType **get_var_dom_type** () const
- void set_boolean_domain ()

  Specifies a boolean domain for the variable.
- void set_float_domain ()

  Specifies a float domain for the variable.
- void set_int_domain ()

  Specifies an integer domain for the variable.
- void set_range_domain (int lw, int ub)
- int **get_lw_bound_domain** () const
- int **get_up_bound_domain** () const
- void set_set_domain (const std::vector< int > &elems)
- const std::vector< int > **get_set_domain** ()
- virtual void print () const

  Print info methods.

Protected Attributes

- std::string **_var_id**
- bool **_objective_var**
- bool **_support_var**
- VarDomainType **_var_dom_type**
- int **_lw_bound**
- int **_up_bound**
- std::vector< int > **_set_domain**

6.30.1 Member Function Documentation

6.30.1.1 void TokenVar::set_range_domain (int *lw,* int *ub*)

Specifies a range domain for the variable with a given lower and upper bound.

Parameters

lw	lower bound
ub	upper bound

6.30.1.2 void TokenVar::set_set_domain (const std::vector< int > & *elems*)

Specifies a set domain for the variable with the given vector of elements.

Parameters

elems	vector of elements

6.30.1.3 void TokenVar::set_var_dom_type (VarDomainType *vdt*)

Set the type of the current (token) variable.

Parameters

vdt	the variable domain type of type VarDomainType.

6.30.1.4 void TokenVar::set_var_id (std::string *str*)

Set the (string) identifier of the variable represented as a token. The id is retrieved using the get_var_id() method.

Parameters

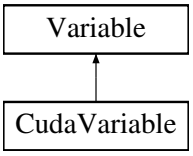
str	the string identifier of the variable.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_var.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/token_var.cpp

6.31 Variable Class Reference

Inheritance diagram for Variable:



Public Member Functions

- **Variable** (int)
- int **get_id** () const
- void set_str_id (std::string str)
- std::string **get_str_id** () const
- void **set_type** (VariableType vt)
- VariableType **get_type** () const
- virtual void set_domain (DomainType dt)
- virtual void print () const =0

  Print info about the variable.

Protected Attributes

- std::string **_dbg**
- int **_id**
- std::string **_str_id**
- VariableType **_var_type**
- DomainPtr _domain_ptr

6.31.1 Member Function Documentation

6.31.1.1 void Variable::set_domain (DomainType *dt*) [virtual]

Set domain according to the specific variable implementation.

Note

: different types of variable

Parameters

dt	domain type of type DomainType to set to the current variable

6.31.1.2 void Variable::set_str_id (std::string *str*)

Set the (string) id of the variable.

Parameters

str	the string to set as variable's identifier

6.31.2 Member Data Documentation

6.31.2.1 DomainPtr Variable::_domain_ptr [protected]

Pointer to the domain of the variable.

Note

: each variable is associated with a Finite Domain.

The documentation for this class was generated from the following files:

- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/variable.h
- /Users/fedecampe/Desktop/NVIDIOSO-PRJ/NVIDIOSO/NVIDIOSO/variable.cpp

Index