My Project

Generated by Doxygen 1.8.6

Mon Sep 28 2015 13:20:08

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

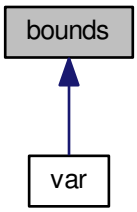
Chapter 4

Class Documentation

4.1 bounds Struct Reference

`#include <BP_Data.hpp>`

Inheritance diagram for bounds:



Public Member Functions

- ~bounds ()

Public Attributes

- int type
- double lb
- double ub

4.1.1 Constructor & Destructor Documentation

4.1.1.1 bounds::~bounds ( ) [inline]

4.1.2 Member Data Documentation

4.1.2.1 double bounds::lb

4.1.2.2 int bounds::type

4.1.2.3 double bounds::ub

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

- BP_Data.hpp

4.2 BP_Input Class Reference

`#include <BP_Data.hpp>`

Public Member Functions

- BP_Input (string file_name)
- bounds getBterms (const int i) const
- const vector< elem > & getMatcoeff (const int i) const
- const vector< elem > & getMatcoeff2 (const int j) const
- unsigned getNcons () const
- unsigned getNvars () const
- var getVar (const int j) const
- vector< var > getVars () const
- int getDirection () const
- unsigned getScale () const
- void setDigits (double number)
- ~BP_Input ()

Protected Attributes

- int nvars
- int ncons
- int nbinvars
- int nintvars
- int direction
- unsigned scale
- vector< vector< elem > > matcoeff
- vector< vector< elem > > matcoeff2
- vector< bounds > bterms
- vector< var > vars

Friends

- ostream & operator<< (ostream &os, const BP_Input &bs)

4.2.1 Constructor & Destructor Documentation

4.2.1.1 BP_Input::BP_Input (string *file_name*)

4.2.1.2 BP_Input::~BP_Input ( ) [inline]

4.2.2 Member Function Documentation

4.2.2.1 bounds BP_Input::getBterms (const int *i*) const [inline]

4.2.2.2 int BP_Input::getDirection ( ) const [inline]

4.2.2.3 const vector<elem>& BP_Input::getMatcoeff (const int *i*) const [inline]

4.2.2.4 const vector<elem>& BP_Input::getMatcoeff2 (const int *j*) const [inline]

4.2.2.5 unsigned BP_Input::getNcons ( ) const [inline]

4.2.2.6 unsigned BP_Input::getNvars ( ) const [inline]

4.2.2.7 unsigned BP_Input::getScale ( ) const [inline]

4.2.2.8 var BP_Input::getVar (const int *j*) const [inline]

4.2.2.9 vector<var> BP_Input::getVars ( ) const [inline]

4.2.2.10 void BP_Input::setDigits (double *number*)

4.2.3 Friends And Related Function Documentation

4.2.3.1 ostream& operator<< (ostream & *os,* const BP_Input & *bs*) [friend]

4.2.4 Member Data Documentation

4.2.4.1 vector<bounds> BP_Input::bterms [protected]

4.2.4.2 int BP_Input::direction [protected]

4.2.4.3 vector<vector<elem> > BP_Input::matcoeff [protected]

4.2.4.4 vector<vector<elem> > BP_Input::matcoeff2 [protected]

4.2.4.5 int BP_Input::nbinvars [protected]

4.2.4.6 int BP_Input::ncons [protected]

4.2.4.7 int BP_Input::nintvars [protected]

4.2.4.8 int BP_Input::nvars [protected]

4.2.4.9 unsigned BP_Input::scale [protected]

4.2.4.10 vector<var> BP_Input::vars [protected]

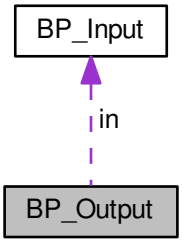
The documentation for this class was generated from the following files:

- BP_Data.hpp
- BP_Data.cpp

4.3 BP_Output Class Reference

```
#include <BP_Data.hpp>
```

Collaboration diagram for BP_Output:



Public Member Functions

- BP_Output (const BP_Input &i)
- BP_Output & operator= (const BP_Output &out)
- int assignment (int var) const
- void assign (int var, bool b)

Protected Attributes

- const BP_Input & in
- vector< bool > varAssignment

Friends

- ostream & operator<< (ostream &os, const BP_Output &out)
- istream & operator>> (istream &is, BP_Output &out)

4.3.1 Constructor & Destructor Documentation

4.3.1.1 BP_Output::BP_Output (const BP_Input & *i*)

4.3.2 Member Function Documentation

4.3.2.1 void BP_Output::assign (int *var,* bool *b*)

4.3.2.2 int BP_Output::assignment (int *var*) const [inline]

4.3.2.3 BP_Output & BP_Output::operator= (const BP_Output & *out*)

4.3.3 Friends And Related Function Documentation

4.3.3.1 ostream& operator<< (ostream & *os,* const BP_Output & *out*) [friend]

4.3.3.2 istream& operator>> (istream & *is,* BP_Output & *out*) [friend]

4.3.4 Member Data Documentation

4.3.4.1 const BP_Input& BP_Output::in [protected]

4.3.4.2 vector<bool> BP_Output::varAssignment [protected]

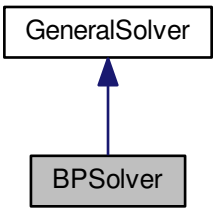
The documentation for this class was generated from the following files:

- BP_Data.hpp
- BP_Data.cpp

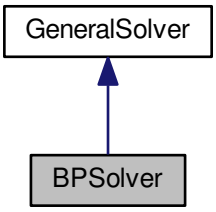
4.4 BPSolver Class Reference

`#include <BPSolver.hpp>`

Inheritance diagram for BPSolver:



Collaboration diagram for BPSolver:



Public Member Functions

- BPSolver (BP_Input *in)
- ~BPSolver ()
- void printCurrent ()

  Constructor for cloning s.

4.4.1 Constructor & Destructor Documentation

4.4.1.1 BPSolver::BPSolver (BP_Input * *in*) [inline]

4.4.1.2 BPSolver::~BPSolver ( ) [inline]

4.4.2 Member Function Documentation

4.4.2.1 void BPSolver::printCurrent ( ) [inline]

Constructor for cloning s.

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

- BPSolver.hpp

4.5 Clock Class Reference

`#include <Clock.hpp>`

Static Public Attributes

- static double globalClock

4.5.1 Member Data Documentation

4.5.1.1 double Clock::globalClock [static]

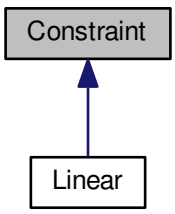
The documentation for this class was generated from the following files:

- Clock.hpp
- Clock.cpp

4.6 Constraint Class Reference

`#include <Constraint.hpp>`

Inheritance diagram for Constraint:



Classes

- struct SortGreater

Public Member Functions

- Constraint ()
- ~Constraint ()
- void setNumberOfIntegerVariables (int number)
- int & getNumberOfIntegerVariables ()
- int getType ()
- int getDeltaViolation ()
- int getDeltaViolationDegree ()
- int getViolation ()
- bool isOneway ()
- void isOneway (bool set)
- std::unordered_map< int, coefType > & getCoefficients ()
- std::vector< IntegerVariable * > & getVariables ()
- int getViolationDegree ()
- int getArgument (int i)
- void setInvariant (std::shared_ptr< Invariant > invar)
- std::shared_ptr< Invariant > & getInvariant ()
- unsigned getScopeSize ()
- bool operator< (Constraint &cons) const
- bool operator> (Constraint &cons) const
- virtual int setDeltaViolation ()
- virtual int setDeltaViolationDegree ()
- virtual int updateViolation ()
- virtual int updateViolationDegree ()
- virtual bool testCons ()
- virtual bool testObj ()

Protected Attributes

- int Violation = 0
- int ViolationDegree = 0
- int DeltaViolation = 0
- int DeltaViolationDegree = 0
- int priority
- int type
- bool oneway = false
- int numberOfIntegerVariables = 0
- unsigned scopeSize
- std::vector< int > arguments
- std::vector< IntegerVariable * > variables
- std::unordered_map< int, coefType > coefficients
- std::shared_ptr< Invariant > invariant

4.6.1 Constructor & Destructor Documentation

4.6.1.1 Constraint::Constraint ( ) [inline]

4.6.1.2 Constraint::~Constraint ( ) [inline]

4.6.2 Member Function Documentation

4.6.2.1 int Constraint::getArgument (int *i*) [inline]

4.6.2.2 std::unordered_map<int, coefType>& Constraint::getCoefficients ( ) [inline]

4.6.2.3 int Constraint::getDeltaViolation ( ) [inline]

4.6.2.4 int Constraint::getDeltaViolationDegree ( ) [inline]

4.6.2.5 std::shared_ptr<Invariant>& Constraint::getInvariant ( ) [inline]

4.6.2.6 int& Constraint::getNumberOfIntegerVariables ( ) [inline]

4.6.2.7 unsigned Constraint::getScopeSize ( ) [inline]

4.6.2.8 int Constraint::getType ( ) [inline]

4.6.2.9 std::vector<IntegerVariable*>& Constraint::getVariables ( ) [inline]

4.6.2.10 int Constraint::getViolation ( ) [inline]

4.6.2.11 int Constraint::getViolationDegree ( ) [inline]

4.6.2.12 bool Constraint::isOneway ( ) [inline]

4.6.2.13 void Constraint::isOneway (bool *set*) [inline]

4.6.2.14 bool Constraint::operator< (Constraint & *cons*) const [inline]

4.6.2.15 bool Constraint::operator> (Constraint & *cons*) const [inline]

4.6.2.16 virtual int Constraint::setDeltaViolation ( ) [inline], [virtual]

Reimplemented in Linear.

4.6.2.17 virtual int Constraint::setDeltaViolationDegree ( ) [inline], [virtual]

Reimplemented in Linear.

4.6.2.18 void Constraint::setInvariant (std::shared_ptr< Invariant > *invar*) [inline]

4.6.2.19 void Constraint::setNumberOfIntegerVariables (int *number*) [inline]

4.6.2.20 virtual bool Constraint::testCons ( ) [inline], [virtual]

Reimplemented in Linear.

4.6.2.21 virtual bool Constraint::testObj ( ) [inline], [virtual]

Reimplemented in Linear.

4.6.2.22 virtual int Constraint::updateViolation ( ) [inline], [virtual]

Reimplemented in Linear.

4.6.2.23 virtual int Constraint::updateViolationDegree ( ) [inline], [virtual]

Reimplemented in Linear.

4.6.3 Member Data Documentation

4.6.3.1 std::vector<int> Constraint::arguments [protected]

4.6.3.2 std::unordered_map<int, coefType> Constraint::coefficients [protected]

4.6.3.3 int Constraint::DeltaViolation = 0 [protected]

4.6.3.4 int Constraint::DeltaViolationDegree = 0 [protected]

4.6.3.5 std::shared_ptr<Invariant> Constraint::invariant [protected]

4.6.3.6 int Constraint::numberOfIntegerVariables = 0 [protected]

4.6.3.7 bool Constraint::oneway = false [protected]

4.6.3.8 int Constraint::priority [protected]

4.6.3.9 unsigned Constraint::scopeSize [protected]

4.6.3.10 int Constraint::type [protected]

4.6.3.11 std::vector<IntegerVariable*> Constraint::variables [protected]

4.6.3.12 int Constraint::Violation = 0 [protected]

4.6.3.13 int Constraint::ViolationDegree = 0 [protected]

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

- Constraint.hpp

4.7 ConstraintSorter Class Reference

`#include <Constraint.hpp>`

Public Member Functions

- ConstraintSorter ()
- bool operator() (std::shared_ptr< Constraint > &cons1, std::shared_ptr< Constraint > &cons2)

4.7.1 Constructor & Destructor Documentation

4.7.1.1 ConstraintSorter::ConstraintSorter ( ) [inline]

4.7.2 Member Function Documentation

4.7.2.1 bool ConstraintSorter::operator() (std::shared_ptr< Constraint > & *cons1*, std::shared_ptr< Constraint > & *cons2*) [inline]

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

- Constraint.hpp

4.8 elem Struct Reference

`#include <BP_Data.hpp>`

Public Member Functions

- ~elem ()

Public Attributes

- int index
- double coeff

4.8.1 Constructor & Destructor Documentation

4.8.1.1 elem::~elem ( ) [inline]

4.8.2 Member Data Documentation

4.8.2.1 double elem::coeff

4.8.2.2 int elem::index

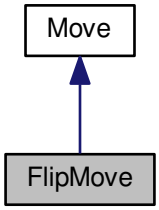
The documentation for this struct was generated from the following file:

- BP_Data.hpp

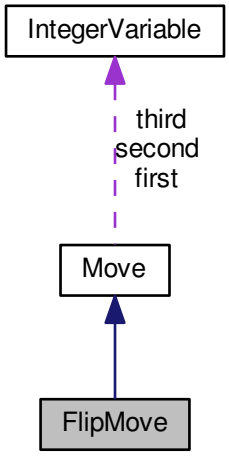
4.9 FlipMove Class Reference

`#include <FlipMove.hpp>`

Inheritance diagram for FlipMove:



Collaboration diagram for FlipMove:



Public Member Functions

- FlipMove (IntegerVariable *iv)
- FlipMove (const FlipMove &orig)
- virtual ~FlipMove ()
- FlipMove & operator= (const FlipMove &a)

Additional Inherited Members

4.9.1 Constructor & Destructor Documentation

4.9.1.1 FlipMove::FlipMove (IntegerVariable * *iv*)

4.9.1.2 FlipMove::FlipMove (const FlipMove & *orig*)

4.9.1.3 FlipMove::~FlipMove ( ) [virtual]

4.9.2 Member Function Documentation

4.9.2.1 FlipMove::operator= (const FlipMove & *a*)

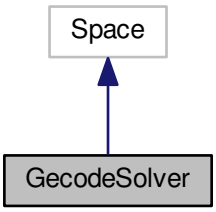
The documentation for this class was generated from the following files:

- FlipMove.hpp
- FlipMove.cpp

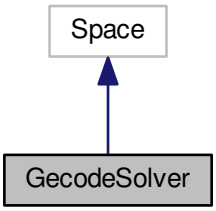
4.10 GecodeSolver Class Reference

`#include <GecodeSolver.hpp>`

Inheritance diagram for GecodeSolver:



Collaboration diagram for GecodeSolver:



Public Member Functions

- GecodeSolver (std::shared_ptr< Model > model)
- virtual ~GecodeSolver ()

- void branch ()
- bool initialize (int TimeForGecode, bool fix)
- bool FindSolution (int TimeForGecode, bool fix)
- void linear (std::vector< int > &coefficients, const std::vector< IntegerVariable * > &variables, int relation, int upperbound)
- void createGecodeVariable (int lb, int ub)
- void SetValues (Gecode::IntVarArray vars)
- void fixVariables ()
- void printSpaceStatus ()
- void createArray ()
- void print_stats (Gecode::Search::Statistics &stat)
- void print (std::ostream &os) const
- GecodeSolver (bool share, GecodeSolver &s)
- void postCovSol ()
- virtual Space * copy (bool share)

Protected Attributes

- std::shared_ptr< Model > model
- Gecode::IntVarArray IntVars
- Gecode::IntVarArgs tmpVars
- Gecode::IntVarArgs binVars

4.10.1 Constructor & Destructor Documentation

4.10.1.1 GecodeSolver::GecodeSolver (std::shared_ptr< Model > *model*)

4.10.1.2 GecodeSolver::~GecodeSolver ( ) [virtual]

4.10.1.3 GecodeSolver::GecodeSolver (bool *share*, GecodeSolver & *s*)

4.10.2 Member Function Documentation

4.10.2.1 void GecodeSolver::branch ( )

4.10.2.2 Gecode::Space * GecodeSolver::copy (bool *share*) [virtual]

4.10.2.3 void GecodeSolver::createArray ( )

4.10.2.4 void GecodeSolver::createGecodeVariable (int *lb*, int *ub*)

4.10.2.5 bool GecodeSolver::FindSolution (int *TimeForGecode*, bool *fix*)

4.10.2.6 void GecodeSolver::fixVariables ( )

4.10.2.7 bool GecodeSolver::initialize (int *TimeForGecode*, bool *fix*)

4.10.2.8 void GecodeSolver::linear (std::vector< int > & *coefficients*, const std::vector< IntegerVariable * > & *variables*, int *relation*, int *upperbound*)

4.10.2.9 void GecodeSolver::postCovSol ( )

4.10.2.10 void GecodeSolver::print (std::ostream & *os*) const

4.10.2.11 void GecodeSolver::print_stats (Gecode::Search::Statistics & *stat*)

4.10.2.12 void GecodeSolver::printSpaceStatus ( )

4.10.2.13 void GecodeSolver::SetValues (Gecode::IntVarArray *vars*)

4.10.3 Member Data Documentation

4.10.3.1 Gecode::IntVarArgs GecodeSolver::binVars [protected]

4.10.3.2 Gecode::IntVarArray GecodeSolver::IntVars [protected]

4.10.3.3 std::shared_ptr<Model> GecodeSolver::model [protected]

4.10.3.4 Gecode::IntVarArgs GecodeSolver::tmpVars [protected]

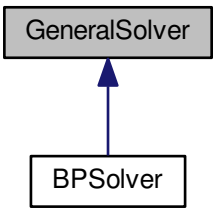
The documentation for this class was generated from the following files:

- GecodeSolver.hpp
- GecodeSolver.cpp

4.11 GeneralSolver Class Reference

`#include <GeneralSolver.hpp>`

Inheritance diagram for GeneralSolver:



Public Member Functions

- GeneralSolver ()
- ~GeneralSolver ()
- GeneralSolver & operator= (const GeneralSolver &a)
- void linear (std::vector< int > &coefficients, std::vector< IntegerVariable * > &variables, int relation, int ub, unsigned priority)
- void createIntVars (unsigned numberOfVariables, int lb, int ub)
- void createIntVar (int lb, int ub)

  Create a single variable with given lower and upper bound.
- std::vector< IntegerVariable * > & getAllVariables ()
- void print (std::vector< IntegerVariable > &IntegerVariables)

  Only for testing, should be removed.

- void InitialSolution (int TimeForGecode)
- void relax (int timesRelaxed)
- void simpleRelax (int timesRelaxed)
- void initializeLS ()
- bool canBeMadeOneway (IntegerVariable *iv, constraint cons)
- void makeOneway (IntegerVariable *iv, constraint cons, int coeff)
- int getInitialValue ()
- void optimizeSolution (int time)
- void printCurrent ()

Private Member Functions

- void print_stats (Gecode::Search::Statistics &stat)

Private Attributes

- std::shared_ptr< Model > model = std::make_shared<Model> ()
- std::shared_ptr< State > st = std::make_shared<State>(model)
- std::unique_ptr< LSSpace > LS = std::unique_ptr<LSSpace> (new LSSpace(model))
- std::unique_ptr< GecodeSolver > GS = std::unique_ptr<GecodeSolver> (new GecodeSolver(model))

Friends

- class Test

4.11.1 Constructor & Destructor Documentation

4.11.1.1 GeneralSolver::GeneralSolver ( ) [inline]

4.11.1.2 GeneralSolver::~GeneralSolver ( ) [inline]

4.11.2 Member Function Documentation

4.11.2.1 bool GeneralSolver::canBeMadeOneway (IntegerVariable * *iv*, constraint *cons*) [inline]

4.11.2.2 void GeneralSolver::createIntVar (int *lb*, int *ub*) [inline]

Create a single variable with given lower and upper bound.

4.11.2.3 void GeneralSolver::createIntVars (unsigned *numberOfVariables*, int *lb*, int *ub*) [inline]

4.11.2.4 std::vector<IntegerVariable*>& GeneralSolver::getAllVariables ( ) [inline]

4.11.2.5 int GeneralSolver::getInitialValue ( ) [inline]

4.11.2.6 void GeneralSolver::initializeLS ( ) [inline]

Sort constraints a variable is part of in decreasing order according to domain

Sort integer variables decreasing order according to number of constraints they are involved

Omskriv det hele. Alt skal bygges op model->invariant fungerer som en prioritet invariant skal opdateres i. Hver variable har deres egen kø.

4.11.2.7 void GeneralSolver::InitialSolution (int *TimeForGecode*) [inline]

Uses Gecode to find initial solution returns a new GeneralSolver with the initial solution the old (the one this method is called from) is not updated with new solution.

4.11.2.8 void GeneralSolver::linear (std::vector< int > & *coefficients,* std::vector< **IntegerVariable** * > & *variables,* int *relation,* int *ub,* unsigned *priority*) [inline]

4.11.2.9 void GeneralSolver::makeOneway (**IntegerVariable** * *iv,* **constraint** *cons,* int *coeff*) [inline]

4.11.2.10 GeneralSolver& GeneralSolver::operator= (const GeneralSolver & *a*) [inline]

4.11.2.11 void GeneralSolver::optimizeSolution (int *time*) [inline]

4.11.2.12 void GeneralSolver::print (std::vector< **IntegerVariable** > & *IntegerVariables*) [inline]

Only for testing, should be removed.

4.11.2.13 void GeneralSolver::print_stats (Gecode::Search::Statistics & *stat*) [inline], [private]

4.11.2.14 void GeneralSolver::printCurrent ( ) [inline]

4.11.2.15 void GeneralSolver::relax (int *timesRelaxed*) [inline]

relaxes the space (reduce the number of constraints). Used when Gecode cant find a solution in time. Only works for binary Different relaxation can be chosen (not atm) needs to create a new GecodeSolver (Space) and recreate some of the calls the user made (those that should not be relaxed).

4.11.2.16 void GeneralSolver::simpleRelax (int *timesRelaxed*) [inline]

4.11.3 Friends And Related Function Documentation

4.11.3.1 friend class Test [friend]

4.11.4 Member Data Documentation

4.11.4.1 std::unique_ptr<**GecodeSolver**> GeneralSolver::GS = std::unique_ptr<**GecodeSolver**> (new **GecodeSolver(model)**) [private]

4.11.4.2 std::unique_ptr<**LSSpace**> GeneralSolver::LS = std::unique_ptr<**LSSpace**> (new **LSSpace(model)**) [private]

4.11.4.3 std::shared_ptr<**Model**> GeneralSolver::model = std::make_shared<**Model**> () [private]

4.11.4.4 std::shared_ptr<**State**> GeneralSolver::st = std::make_shared<**State**>(**model**) [private]

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

- GeneralSolver.hpp

4.12 IntegerVariable Class Reference

`#include <IntegerVariable.hpp>`

Public Member Functions

- VariableInConstraints & usedInConstraints ()
- void addToUsedInConstraints (std::shared_ptr< Constraint > constraint)
- invariant getOneway ()
- void setDefinedBy (invariant invar, constraint cons)
- IntegerVariable (int lowerbound, int upperbound, int id)
- void setCurrentValue (int val)
- bool isIntegerVariable ()
- int getCurrentValue ()
- void addToUpdate (updateType invariant)
- InvariantContainer & getUpdateVector ()
- int getID ()
- Gecode::IntVar * getVariablePointer ()
- void setVariablePointer (Gecode::IntVar &gecodeVar)
- int getLowerBound ()
- int getUpperBound ()
- ~IntegerVariable ()

Public Attributes

- int usedIn = 0

  Only used for testing instances.

Protected Member Functions

- void clearUpdateVector ()

Protected Attributes

- int lowerBound
- int upperBound
- int vectorID
- int value = 0
- bool isInteger = false
- bool isDefined = false
- VariableInConstraints constraints
- invariant oneway
- constraint definedByCons
- updateVector update
- Gecode::IntVarArray * ArrayPointer
- Gecode::IntVar * VariablePointer

Friends

- class GeneralSolver

4.12.1 Constructor & Destructor Documentation

4.12.1.1 IntegerVariable::IntegerVariable (int *lowerbound,* int *upperbound,* int *id*) [inline]

4.12.1.2 IntegerVariable::∼IntegerVariable ( ) [inline]

4.12.2 Member Function Documentation

4.12.2.1 void IntegerVariable::addToUpdate (updateType *invariant*) [inline]

4.12.2.2 void IntegerVariable::addToUsedInConstraints (std::shared_ptr< Constraint > *constraint*) [inline]

4.12.2.3 void IntegerVariable::clearUpdateVector ( ) [inline], [protected]

4.12.2.4 int IntegerVariable::getCurrentValue ( ) [inline]

4.12.2.5 int IntegerVariable::getID ( ) [inline]

4.12.2.6 int IntegerVariable::getLowerBound ( ) [inline]

4.12.2.7 invariant IntegerVariable::getOneway ( ) [inline]

4.12.2.8 InvariantContainer& IntegerVariable::getUpdateVector ( ) [inline]

4.12.2.9 int IntegerVariable::getUpperBound ( ) [inline]

4.12.2.10 Gecode::IntVar∗ IntegerVariable::getVariablePointer ( ) [inline]

4.12.2.11 bool IntegerVariable::isIntegerVariable ( ) [inline]

4.12.2.12 void IntegerVariable::setCurrentValue (int *val*) [inline]

4.12.2.13 void IntegerVariable::setDefinedBy (invariant *invar,* constraint *cons*) [inline]

4.12.2.14 void IntegerVariable::setVariablePointer (Gecode::IntVar & *gecodeVar*) [inline]

4.12.2.15 VariableInConstraints& IntegerVariable::usedInConstraints ( ) [inline]

4.12.3 Friends And Related Function Documentation

4.12.3.1 friend class GeneralSolver [friend]

4.12.4 Member Data Documentation

4.12.4.1 Gecode::IntVarArray∗ IntegerVariable::ArrayPointer [protected]

4.12.4.2 VariableInConstraints IntegerVariable::constraints [protected]

4.12.4.3 constraint IntegerVariable::definedByCons [protected]

4.12.4.4 bool IntegerVariable::isDefined = false [protected]

4.12.4.5 bool IntegerVariable::isInteger = false [protected]

4.12.4.6 int IntegerVariable::lowerBound [protected]

4.12.4.7 invariant IntegerVariable::oneway [protected]

4.12.4.8 updateVector IntegerVariable::update [protected]

4.12.4.9 int IntegerVariable::upperBound [protected]

4.12.4.10 int IntegerVariable::usedIn = 0

Only used for testing instances.

4.12.4.11 int IntegerVariable::value = 0 [protected]

4.12.4.12 Gecode::IntVar* IntegerVariable::VariablePointer [protected]

4.12.4.13 int IntegerVariable::vectorID [protected]

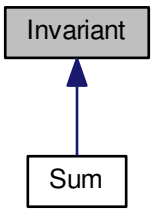
The documentation for this class was generated from the following file:

- IntegerVariable.hpp

4.13 Invariant Class Reference

`#include <Invariant.hpp>`

Inheritance diagram for Invariant:



Public Member Functions

- Invariant ()
- Invariant (const Invariant &a)
- virtual void addChange (int variableNumber, int change)
- virtual int calculateDeltaValue ()
- virtual void initialize ()
- int getDeltaValue ()
- virtual bool test ()
- virtual ~Invariant ()
- int getCurrentValue ()
- void updateValue ()
- void setUsedByConstraint (int constraint, int priority)

- void setUsedByObjective (int objective)
- int getConstraintNumber ()
- int getUsedInObjective ()
- unsigned getPriority ()
- int getType ()
- InvariantContainer & getUpdateVector ()
- void addToUpdate (invariant invar)
- std::vector< IntegerVariable * > & getVariables ()
- std::unordered_map< int, coefType > getCoefficients ()

  Passing by value.
- int getVariableID ()

  Only used when the invariant defines a variable through a oneway constraint.

Protected Attributes

- std::vector< IntegerVariable * > VariablePointers

  Correspond to the priority the invariant should be updated.
- double CurrentValue = 0
- double DeltaValue = 0
- unsigned constraintPriority
- int usedInConstraintNr
- int usedInObjectiveNr
- std::unordered_map< int, coefType > coefficients
- int type
- int startValue = 0

  Should be defined when creating oneway constraints that define (integer)variables.
- int variableID
- InvariantContainer invariants
- InvariantContainer update

Friends

- class GeneralSolver

4.13.1 Constructor & Destructor Documentation

4.13.1.1 Invariant::Invariant ( ) [inline]

4.13.1.2 Invariant::Invariant (const Invariant & *a*) [inline]

4.13.1.3 virtual Invariant::~Invariant ( ) [inline], [virtual]

4.13.2 Member Function Documentation

4.13.2.1 virtual void Invariant::addChange (int *variableNumber,* int *change*) [inline], [virtual]

Reimplemented in Sum.

4.13.2.2 void Invariant::addToUpdate (invariant *invar*) [inline]

4.13.2.3 virtual int Invariant::calculateDeltaValue ( ) [inline], [virtual]

Reimplemented in Sum.

4.13.2.4 std::unordered_map<int, coefType> Invariant::getCoefficients ( ) [inline]

Passing by value.

4.13.2.5 int Invariant::getConstraintNumber ( ) [inline]

4.13.2.6 int Invariant::getCurrentValue ( ) [inline]

4.13.2.7 int Invariant::getDeltaValue ( ) [inline]

4.13.2.8 unsigned Invariant::getPriority ( ) [inline]

4.13.2.9 int Invariant::getType ( ) [inline]

4.13.2.10 InvariantContainer& Invariant::getUpdateVector ( ) [inline]

4.13.2.11 int Invariant::getUsedInObjective ( ) [inline]

4.13.2.12 int Invariant::getVariableID ( ) [inline]

Only used when the invariant defines a variable through a oneway constraint.

4.13.2.13 std::vector<IntegerVariable*>& Invariant::getVariables ( ) [inline]

4.13.2.14 virtual void Invariant::initialize ( ) [inline], [virtual]

4.13.2.15 void Invariant::setUsedByConstraint (int *constraint,* int *priority*) [inline]

4.13.2.16 void Invariant::setUsedByObjective (int *objective*) [inline]

4.13.2.17 virtual bool Invariant::test ( ) [inline], [virtual]

Reimplemented in Sum.

4.13.2.18 void Invariant::updateValue ( ) [inline]

4.13.3 Friends And Related Function Documentation

4.13.3.1 friend class GeneralSolver [friend]

4.13.4 Member Data Documentation

4.13.4.1 std::unordered_map<int, coefType> Invariant::coefficients [protected]

4.13.4.2 unsigned Invariant::constraintPriority [protected]

4.13.4.3 double Invariant::CurrentValue = 0 [protected]

4.13.4.4 double Invariant::DeltaValue = 0 [protected]

4.13.4.5 InvariantContainer Invariant::invariants [protected]

4.13.4.6 int Invariant::startValue = 0 [protected]

Should be defined when creating oneway constraints that define (integer)variables.

4.13.4.7 int Invariant::type [protected]

4.13.4.8 InvariantContainer Invariant::update [protected]

4.13.4.9 int Invariant::usedInConstraintNr [protected]

4.13.4.10 int Invariant::usedInObjectiveNr [protected]

4.13.4.11 int Invariant::variableID [protected]

4.13.4.12 std::vector<IntegerVariable*> Invariant::VariablePointers [protected]

Correspond to the priority the invariant should be updated.

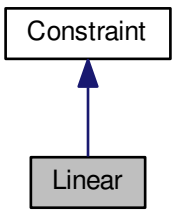
The documentation for this class was generated from the following file:

- Invariant.hpp

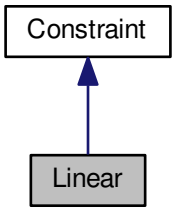
4.14 Linear Class Reference

`#include <Linear.hpp>`

Inheritance diagram for Linear:



Collaboration diagram for Linear:



Public Member Functions

- Linear (std::vector< int > &coefficients, std::vector< IntegerVariable * > &variables, int ub, int relation)

  Used to create the original (given by user) constraints.
- ~Linear ()

  Used to create the Linear constraint used by local search.
- int setDeltaViolation ()
- int setDeltaViolationDegree ()
- int updateViolation ()
- int updateViolationDegree ()
- bool testCons ()
- bool testObj ()

Protected Attributes

- int rhs
- int relation

4.14.1 Constructor & Destructor Documentation

4.14.1.1 Linear::Linear (std::vector< int > & *coefficients*, std::vector< IntegerVariable * > & *variables*, int *ub*, int *relation*) [inline]

Used to create the original (given by user) constraints.

4.14.1.2 Linear::~Linear ( ) [inline]

Used to create the Linear constraint used by local search.

4.14.2 Member Function Documentation

4.14.2.1 int Linear::setDeltaViolation ( ) [inline], [virtual]

Reimplemented from Constraint.

4.14.2.2 int Linear::setDeltaViolationDegree ( ) [inline], [virtual]

Reimplemented from Constraint.

4.14.2.3 bool Linear::testCons ( ) [inline], [virtual]

Reimplemented from Constraint.

4.14.2.4 bool Linear::testObj ( ) [inline], [virtual]

Reimplemented from Constraint.

4.14.2.5 int Linear::updateViolation ( ) [inline], [virtual]

Reimplemented from Constraint.

4.14.2.6 int Linear::updateViolationDegree ( ) [inline], [virtual]

Reimplemented from Constraint.

4.14.3 Member Data Documentation

4.14.3.1 int Linear::relation [protected]

4.14.3.2 int Linear::rhs [protected]

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

- Linear.hpp

4.15 LSSpace Class Reference

`#include <LSSpace.hpp>`

Public Member Functions

- LSSpace (std::shared_ptr< Model > model)
- void printCurrent ()
- ~LSSpace ()
- void optimizeSolution (int time, std::shared_ptr< State > st)
- void simpleMove (int variabelNr)
- bool bestImprovement ()

Private Attributes

- int Violations = 0
- int iterations = 0
- std::shared_ptr< Model > model

Friends

- class Test

4.15.1 Constructor & Destructor Documentation

4.15.1.1 LSSpace::LSSpace (std::shared_ptr< Model > *model*)

4.15.1.2 LSSpace::~LSSpace ( ) [inline]

4.15.2 Member Function Documentation

4.15.2.1 bool LSSpace::bestImprovement ( )

4.15.2.2 void LSSpace::optimizeSolution (int *time*, std::shared_ptr< State > *st*)

4.15.2.3 void LSSpace::printCurrent ( )

4.15.2.4 void LSSpace::simpleMove (int *variabelNr*)

4.15.3 Friends And Related Function Documentation

4.15.3.1 friend class Test [friend]

4.15.4 Member Data Documentation

4.15.4.1 int LSSpace::iterations = 0 [private]

4.15.4.2 std::shared_ptr<Model> LSSpace::model [private]

4.15.4.3 int LSSpace::Violations = 0 [private]

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

- LSSpace.hpp
- LSSpace.cpp

4.16 Model Class Reference

`#include <Model.hpp>`

Public Member Functions

- Model ()
- Model (const Model &orig)
- ~Model ()
- void addBinaryVariable (int lb, int ub)
- void addIntegerVariable (int lb, int ub)
- IntegerVariable * getMaskAt (int i)
- std::vector< IntegerVariable * > & getMask ()
- std::list< IntegerVariable * > & getIntegerVariables ()

  Returns all integer variables as a list. only used to make oneway constraints.
- variableContainer & getNonFixedBinaryVariables ()

- variableContainer & getAllVariables ()
- InvariantContainer & getInvariants ()
- constraintContainer getConstraintsWithPriority (int prio)
- allConstraints & getConstraints ()
- constraintContainer getObjectives ()
- IntegerVariable * getNonFixedBinaryVariable (int i)
- void updateIntegerVariable (int index, Gecode::IntVar &variable)
- void nonFixedVariables (std::vector< IntegerVariable * > *nonFixed)

Public Attributes

- int initialValue

  Should be moved to state.

Private Attributes

- variableContainer original

  All variables given by user.
- variableContainer nonFixedBinaryVariables

  Binary Variables that are not fixed by preprocessing.
- InvariantContainer Invariants

  Invariants for original model, before oneway constraints are made.
- std::list< IntegerVariable * > IntegerVariables

  All integer variables in the model.
- allConstraints Constraints
- std::vector< IntegerVariable * > mask

  All Constraints after some has been made oneway (oneway is not included, oneway is invariants)

4.16.1 Constructor & Destructor Documentation

4.16.1.1 Model::Model ( )

4.16.1.2 Model::Model (const Model & *orig*)

4.16.1.3 Model::~Model ( )

4.16.2 Member Function Documentation

4.16.2.1 void Model::addBinaryVariable (int *lb*, int *ub*)

4.16.2.2 void Model::addIntegerVariable (int *lb*, int *ub*)

4.16.2.3 variableContainer & Model::getAllVariables ( )

4.16.2.4 allConstraints & Model::getConstraints ( )

4.16.2.5 constraintContainer Model::getConstraintsWithPriority (int *prio*)

4.16.2.6 std::list< IntegerVariable * > & Model::getIntegerVariables ( )

Returns all integer variables as a list. only used to make oneway constraints.

4.16.2.7 InvariantContainer & Model::getInvariants ( )

4.16.2.8 std::vector< IntegerVariable * > & Model::getMask ( )

4.16.2.9 IntegerVariable * Model::getMaskAt (int *i*)

4.16.2.10 IntegerVariable * Model::getNonFixedBinaryVariable (int *i*)

4.16.2.11 variableContainer & Model::getNonFixedBinaryVariables ( )

4.16.2.12 constraintContainer Model::getObjectives ( )

4.16.2.13 void Model::nonFixedVariables (std::vector< IntegerVariable * > * *nonFixed*)

4.16.2.14 void Model::updateIntegerVariable (int *index*, Gecode::IntVar & *variable*)

4.16.3 Member Data Documentation

4.16.3.1 allConstraints Model::Constraints [private]

4.16.3.2 int Model::initialValue

Should be moved to state.

4.16.3.3 std::list<IntegerVariable*> Model::IntegerVariables [private]

All integer variables in the model.

4.16.3.4 InvariantContainer Model::Invariants [private]

Invariants for original model, before oneway constraints are made.

Invariants including oneway constraints but not the invariant from the constraints oneway is made

4.16.3.5 std::vector<IntegerVariable*> Model::mask [private]

All Constraints after some has been made oneway (oneway is not included, oneway is invariants)

4.16.3.6 variableContainer Model::nonFixedBinaryVariables [private]

Binary Variables that are not fixed by preprocessing.

4.16.3.7 variableContainer Model::original [private]

All variables given by user.

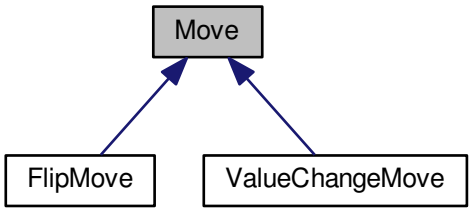
The documentation for this class was generated from the following files:

- Model.hpp
- Model.cpp

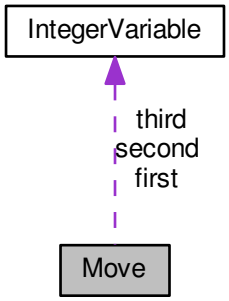
4.17 Move Class Reference

`#include <Move.hpp>`

Inheritance diagram for Move:



Collaboration diagram for Move:



Public Member Functions

- Move ()
- Move (IntegerVariable *var1, int delta1, int type)
- Move (IntegerVariable *var1, int delta1, IntegerVariable *var2, int delta2, int type)
- Move (IntegerVariable *var1, int delta1, IntegerVariable *var2, int delta2, IntegerVariable *var3, int delta3, int type)
- void flip ()
- ~Move ()
- Move (const Move &a)
- Move & operator= (const Move &a)
- void copy (Move *mv)

Public Attributes

- IntegerVariable * first

- int deltaValueFirst
- IntegerVariable * second
- int deltaValueSecond
- IntegerVariable * third
- int deltaValueThird
- int moveType

4.17.1 Constructor & Destructor Documentation

4.17.1.1 Move::Move ( ) [inline]

4.17.1.2 Move::Move (IntegerVariable * *var1*, int *delta1*, int *type*) [inline]

4.17.1.3 Move::Move (IntegerVariable * *var1*, int *delta1*, IntegerVariable * *var2*, int *delta2*, int *type*) [inline]

4.17.1.4 Move::Move (IntegerVariable * *var1*, int *delta1*, IntegerVariable * *var2*, int *delta2*, IntegerVariable * *var3*, int *delta3*, int *type*) [inline]

4.17.1.5 Move::~Move ( ) [inline]

4.17.1.6 Move::Move (const Move & *a*) [inline]

4.17.2 Member Function Documentation

4.17.2.1 void Move::copy (Move * *mv*) [inline]

4.17.2.2 void Move::flip ( ) [inline]

4.17.2.3 Move& Move::operator= (const Move & *a*) [inline]

4.17.3 Member Data Documentation

4.17.3.1 int Move::deltaValueFirst

4.17.3.2 int Move::deltaValueSecond

4.17.3.3 int Move::deltaValueThird

4.17.3.4 IntegerVariable* Move::first

4.17.3.5 int Move::moveType

4.17.3.6 IntegerVariable* Move::second

4.17.3.7 IntegerVariable* Move::third

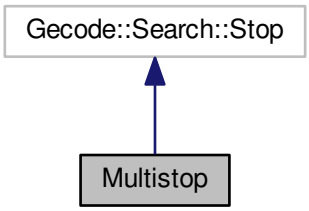
The documentation for this class was generated from the following file:

- Move.hpp

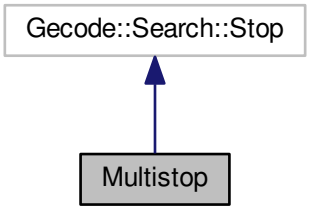
4.18 Multistop Class Reference

```
#include <Multistop.hpp>
```

Inheritance diagram for Multistop:



Collaboration diagram for Multistop:



Public Member Functions

- Multistop (unsigned node, unsigned fail, unsigned time)
- virtual bool stop (const Gecode::Search::Statistics &s, const Gecode::Search::Options &o)

  Return true if node, time or fail limit is exceeded.
- Multistop (const Multistop &orig)
- ~Multistop ()

Public Attributes

- int called = 0

Private Attributes

- Gecode::Search::NodeStop * ns

  Used node stop object.
- Gecode::Search::FailStop * fs

  Used fail stop object.
- Gecode::Search::TimeStop * ts

  Used time stop object.

4.18.1 Constructor & Destructor Documentation

4.18.1.1 Multistop::Multistop (unsigned *node,* unsigned *fail,* unsigned *time*)

Creates a Stop object with three stop criteria node, fail, and time. Giving 0 as an argument will ignorer that criteria. Time is in ms.

4.18.1.2 Multistop::Multistop (const Multistop & *orig*)

4.18.1.3 Multistop::∼Multistop ( )

4.18.2 Member Function Documentation

4.18.2.1 bool Multistop::stop (const Gecode::Search::Statistics & *s,* const Gecode::Search::Options & *o*) [virtual]

Return true if node, time or fail limit is exceeded.

4.18.3 Member Data Documentation

4.18.3.1 int Multistop::called = 0

4.18.3.2 Gecode::Search::FailStop∗ Multistop::fs [private]

Used fail stop object.

4.18.3.3 Gecode::Search::NodeStop∗ Multistop::ns [private]

Used node stop object.

4.18.3.4 Gecode::Search::TimeStop∗ Multistop::ts [private]

Used time stop object.

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

- Multistop.hpp
- Multistop.cpp

4.19 NeighborhoodExplorer Class Reference

`#include <NeighborhoodExplorer.hpp>`

Public Member Functions

- NeighborhoodExplorer (std::shared_ptr< Model > model)
- NeighborhoodExplorer (const NeighborhoodExplorer &orig)
- virtual ∼NeighborhoodExplorer ()
- void randomWalk (Move ∗mv, std::shared_ptr< State > st)
- bool bestImprovement (Move ∗mv, std::shared_ptr< State > st)
- void makeMove (Move ∗mv, std::shared_ptr< State > st)

Public Attributes

- std::shared_ptr< Model > model

Private Member Functions

- std::vector< int > calculateDeltaChange (Move *mv)

  Not using priority of constraints yet.
- void commitMove (Move *mv, std::shared_ptr< State > st)

4.19.1 Constructor & Destructor Documentation

4.19.1.1 NeighborhoodExplorer::NeighborhoodExplorer (std::shared_ptr< Model > *model*)

4.19.1.2 NeighborhoodExplorer::NeighborhoodExplorer (const NeighborhoodExplorer & *orig*)

4.19.1.3 NeighborhoodExplorer::~NeighborhoodExplorer ( ) [virtual]

4.19.2 Member Function Documentation

4.19.2.1 bool NeighborhoodExplorer::bestImprovement (Move * *mv*, std::shared_ptr< State > *st*)

4.19.2.2 std::vector< int > NeighborhoodExplorer::calculateDeltaChange (Move * *mv*) [private]

Not using priority of constraints yet.

4.19.2.3 void NeighborhoodExplorer::commitMove (Move * *mv*, std::shared_ptr< State > *st*) [private]

4.19.2.4 void NeighborhoodExplorer::makeMove (Move * *mv*, std::shared_ptr< State > *st*)

4.19.2.5 void NeighborhoodExplorer::randomWalk (Move * *mv*, std::shared_ptr< State > *st*)

4.19.3 Member Data Documentation

4.19.3.1 std::shared_ptr<Model> NeighborhoodExplorer::model

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

- NeighborhoodExplorer.hpp
- NeighborhoodExplorer.cpp

4.20 Options Class Reference

`#include <Options.hpp>`

Public Member Functions

- Options ()
- Options (const Options &orig)
- virtual ~Options ()

Private Attributes

- ínt timelimit = 0

4.20.1 Constructor & Destructor Documentation

4.20.1.1 Options::Options ( )

4.20.1.2 Options::Options (const Options & *orig*)

4.20.1.3 Options::∼Options ( ) [virtual]

4.20.2 Member Data Documentation

4.20.2.1 ínt Options::timelimit = 0 [private]

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

- Options.hpp
- Options.cpp

4.21 Random Class Reference

```
#include <Random.hpp>
```

Static Public Member Functions

- static int Integer (int lb, int ub)
- static int Integer (int ub)
- static double Double (double lb=0, double ub=1)
- static int Seed (int seed)

Static Public Attributes

- static std::mt19937 mt
- static int seed = std::random_device()()

4.21.1 Member Function Documentation

4.21.1.1 double Random::Double (double *lb* = 0, double *ub* = 1) [static]

4.21.1.2 int Random::Integer (int *lb*, int *ub*) [static]

4.21.1.3 int Random::Integer (int *ub*) [static]

4.21.1.4 int Random::Seed (int *seed*) [static]

4.21.2 Member Data Documentation

4.21.2.1 std::mt19937 Random::mt [static]

4.21.2.2 int Random::seed = std::random_device()() [static]

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

- Random.hpp
- Random.cpp

4.22 Constraint::SortGreater Struct Reference

`#include <Constraint.hpp>`

Public Member Functions

- bool operator() (const std::shared_ptr< Constraint > &cons1, const std::shared_ptr< Constraint > &cons2) const

4.22.1 Member Function Documentation

4.22.1.1 bool Constraint::SortGreater::operator() (const std::shared_ptr< Constraint > & *cons1*, const std::shared_ptr< Constraint > & *cons2*) const [inline]

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

- Constraint.hpp

4.23 State Class Reference

`#include <State.hpp>`

Public Member Functions

- State (std::shared_ptr< Model > model)
- State (const State &orig)
- virtual ~State ()
- void initializeInvariants ()

  Maybe all the initialize should be moved to model (again).
- void initializeConstraints ()
- void initializeObjective ()
- int getObjectiveValue ()
- void saveSolution ()
- std::vector< int > * getSolution ()
- int getSolutionValue ()
- void setSolution ()
- bool recalculateAll ()
- int maskAt (int i)

  Move this to model.
- void shuffleMask ()

  Move this to model.

Public Attributes

- int numberOfViolations

Private Attributes

- int solutionValue
- std::vector< int > * mask
- std::shared_ptr< Model > model
- std::vector< int > * solution

4.23.1 Constructor & Destructor Documentation

4.23.1.1 State::State (std::shared_ptr< Model > *model*)

4.23.1.2 State::State (const State & *orig*)

4.23.1.3 State::~State ( ) [virtual]

4.23.2 Member Function Documentation

4.23.2.1 int State::getObjectiveValue ( )

4.23.2.2 std::vector< int > * State::getSolution ( )

4.23.2.3 int State::getSolutionValue ( )

4.23.2.4 void State::initializeConstraints ( )

4.23.2.5 void State::initializeInvariants ( )

Maybe all the initialize should be moved to model (again).

4.23.2.6 void State::initializeObjective ( )

4.23.2.7 int State::maskAt (int *i*)

Move this to model.

4.23.2.8 bool State::recalculateAll ( )

4.23.2.9 void State::saveSolution ( )

4.23.2.10 void State::setSolution ( )

4.23.2.11 void State::shuffleMask ( )

Move this to model.

4.23.3 Member Data Documentation

4.23.3.1 std::vector<int>* State::mask [private]

4.23.3.2 std::shared_ptr<Model> State::model [private]

4.23.3.3 int State::numberOfViolations

4.23.3.4 std::vector<int>* State::solution [private]

4.23.3.5 int State::solutionValue [private]

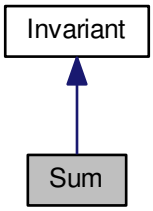
The documentation for this class was generated from the following files:

- State.hpp
- State.cpp

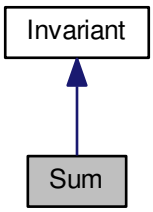
4.24 Sum Class Reference

`#include <Sum.hpp>`

Inheritance diagram for Sum:



Collaboration diagram for Sum:



Public Member Functions

- Sum (std::vector< IntegerVariable * > &vars, std::vector< int > &c)
- Sum (std::vector< IntegerVariable * > &vars, std::unordered_map< int, coefType > &map)

Construct that copies a Coefficient map. Size can be different when this sum contains invariants.

- Sum (const Sum &a)
- ~Sum ()
- Sum & operator= (const Sum &a)
- int calculateDeltaValue ()
- void addChange (int variableNumber, int changeInValue)
- bool test ()

  *update currentValue by adding currentValue*coeff of all variables and invariants*

Protected Attributes

- std::vector< int > VariableChange

4.24.1 Constructor & Destructor Documentation

4.24.1.1 Sum::Sum (std::vector< IntegerVariable * > & *vars*, std::vector< int > & *c*)

4.24.1.2 Sum::Sum (std::vector< IntegerVariable * > & *vars*, std::unordered_map< int, coefType > & *map*)

Construct that copies a Coefficient map. Size can be different when this sum contains invariants.

4.24.1.3 Sum::Sum (const Sum & *a*)

4.24.1.4 Sum::~Sum ( )

4.24.2 Member Function Documentation

4.24.2.1 void Sum::addChange (int *variableNumber*, int *changeInValue*) [virtual]

Reimplemented from Invariant.

4.24.2.2 int Sum::calculateDeltaValue ( ) [virtual]

Reimplemented from Invariant.

4.24.2.3 Sum& Sum::operator= (const Sum & *a*)

4.24.2.4 bool Sum::test ( ) [virtual]

update currentValue by adding currentValue*coeff of all variables and invariants

Reimplemented from Invariant.

4.24.3 Member Data Documentation

4.24.3.1 std::vector<int> Sum::VariableChange [protected]

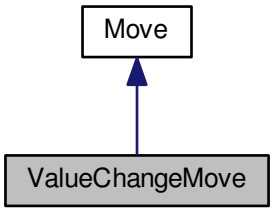
The documentation for this class was generated from the following files:

- Sum.hpp
- Sum.cpp

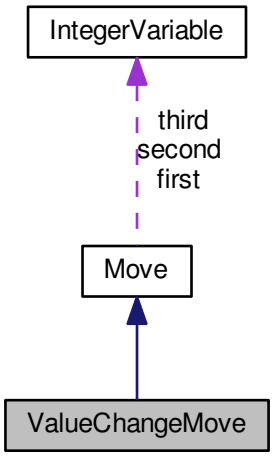
4.25 ValueChangeMove Class Reference

`#include <ValueChangeMove.hpp>`

Inheritance diagram for ValueChangeMove:



Collaboration diagram for ValueChangeMove:



Public Member Functions

- ValueChangeMove (IntegerVariable *iv, int newValue)
- ValueChangeMove (const ValueChangeMove &orig)
- ~ValueChangeMove ()
- ValueChangeMove & operator= (const ValueChangeMove &a)

Additional Inherited Members

4.25.1 Constructor & Destructor Documentation

4.25.1.1 ValueChangeMove::ValueChangeMove (IntegerVariable * *iv,* int *newValue*)

4.25.1.2 ValueChangeMove::ValueChangeMove (const ValueChangeMove & *orig*)

4.25.1.3 ValueChangeMove::~ValueChangeMove ( )

4.25.2 Member Function Documentation

4.25.2.1 ValueChangeMove & ValueChangeMove::operator= (const ValueChangeMove & *a*)

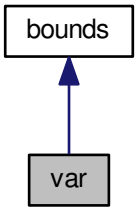
The documentation for this class was generated from the following files:

- ValueChangeMove.hpp
- ValueChangeMove.cpp

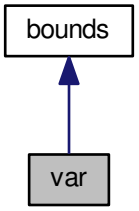
4.26 var Struct Reference

`#include <BP_Data.hpp>`

Inheritance diagram for var:



Collaboration diagram for var:



Public Member Functions

- ~var ()

Public Attributes

- double objcoeff
- int kind
- bool bin

4.26.1 Constructor & Destructor Documentation

4.26.1.1 var::~var ( ) [inline]

4.26.2 Member Data Documentation

4.26.2.1 bool var::bin

4.26.2.2 int var::kind

4.26.2.3 double var::objcoeff

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

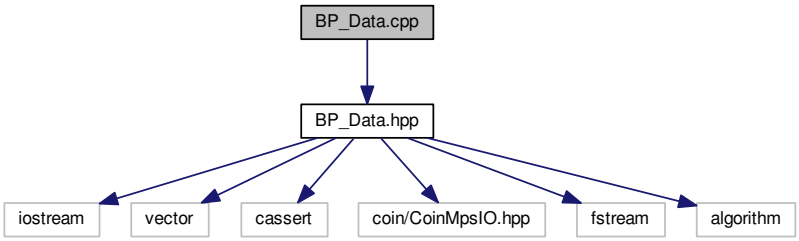
- BP_Data.hpp

Chapter 5

File Documentation

5.1 BP_Data.cpp File Reference

```
#include "BP_Data.hpp"
```
Include dependency graph for BP_Data.cpp:



Functions

- ostream & operator<< (ostream &os, const BP_Output &out)
- istream & operator>> (istream &is, BP_Output &out)

5.1.1 Function Documentation

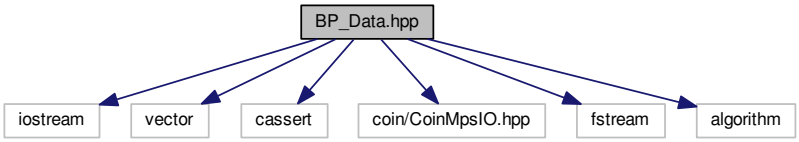
5.1.1.1 ostream& operator<< (ostream & *os,* const BP_Output & *out*)

5.1.1.2 istream& operator>> (istream & *is,* BP_Output & *out*)

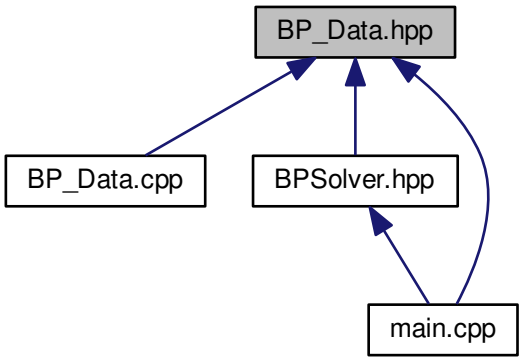
5.2 BP_Data.hpp File Reference

```
#include <iostream>
#include <vector>
#include <cassert>
#include "coin/CoinMpsIO.hpp"
#include <fstream>
#include <algorithm>
```

Include dependency graph for BP_Data.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct elem
- struct bounds
- struct var
- class BP_Input
- class BP_Output

Macros

- #define DEBUG(X) X

5.2.1 Macro Definition Documentation

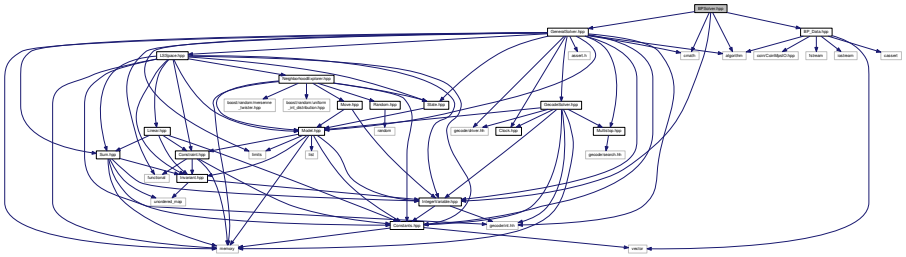
5.2.1.1 #define DEBUG(*X*) X

5.3 BPSolver.hpp File Reference

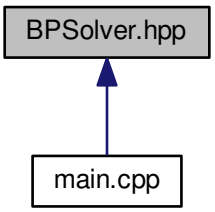
```
#include "BP_Data.hpp"
```

```
#include <cmath>
#include <algorithm>
#include "GeneralSolver.hpp"
#include "IntegerVariable.hpp"
```

Include dependency graph for BPSolver.hpp:



This graph shows which files directly or indirectly include this file:



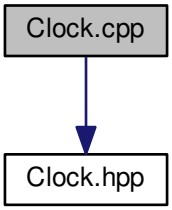
Classes

- class BPSolver

5.4 Clock.cpp File Reference

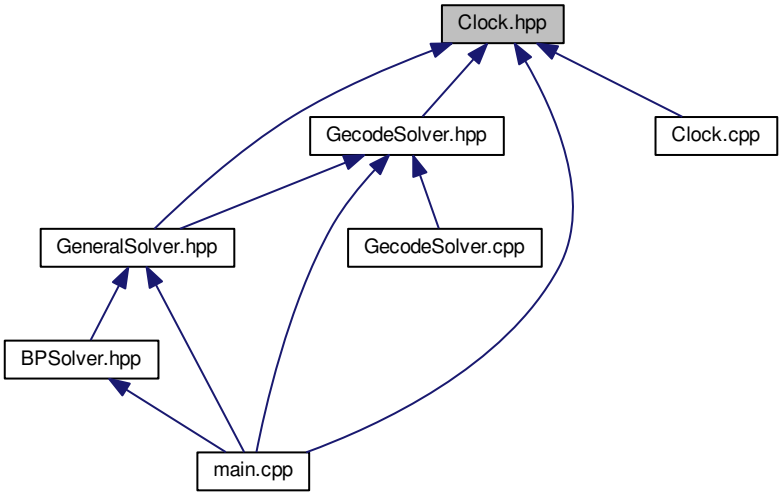
```
#include "Clock.hpp"
```

Include dependency graph for Clock.cpp:



5.5 Clock.hpp File Reference

This graph shows which files directly or indirectly include this file:



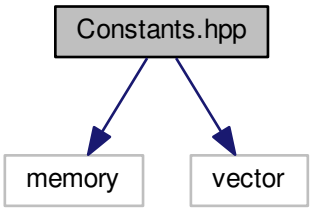
Classes

- class Clock

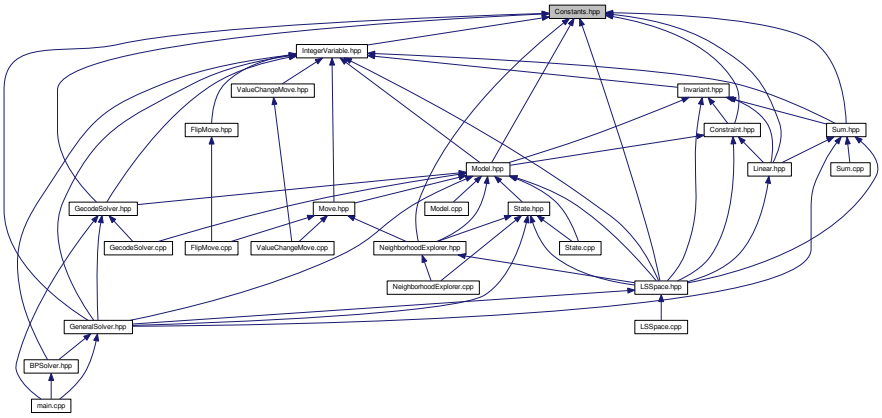
5.6 Constants.hpp File Reference

```
#include <memory>
#include <vector>
```

Include dependency graph for Constants.hpp:



This graph shows which files directly or indirectly include this file:



Macros

- #define nonTerm 0.000000000001
- #define weight 100
- #define tabulistLB 5
- #define tabulistUB 20
- #define maxIter 200000000
- #define iniTrails 20
- #define maxMoves 600
- #define RANDOMSEED 1337
- #define PRINT TRUE
- #define debug std::cout << __FILE__ << " " << __FUNCTION__ << " " << __LINE__ << std::endl;
- #define EQ 0
- #define LQ 1
- #define GQ 3
- #define GR 4
- #define LE 2
- #define OBJ 0
- #define HARD 1
- #define SOFT 2

- #define FLIP 1
- #define SWAP 2
- #define VALUECHANGE 3
- #define LINEAR 1
- #define SUM 1

Typedefs

- typedef double coefType
- typedef std::shared_ptr < Constraint > constraint
- typedef std::vector < IntegerVariable * > variableContainer
- typedef std::vector< constraint > VariableInConstraints
- typedef std::shared_ptr < Invariant > invariant
- typedef std::vector< invariant > InvariantContainer
- typedef InvariantContainer updateVector
- typedef std::shared_ptr < Invariant > updateType
- typedef std::shared_ptr < std::vector< constraint > > constraintContainer
- typedef std::vector < constraintContainer > allConstraints

5.6.1 Macro Definition Documentation

5.6.1.1 #define debug std::cout << __FILE__ << " " << __FUNCTION__ << " " << __LINE__ << std::endl;

5.6.1.2 #define EQ 0

5.6.1.3 #define FLIP 1

5.6.1.4 #define GQ 3

5.6.1.5 #define GR 4

5.6.1.6 #define HARD 1

5.6.1.7 #define iniTrails 20

5.6.1.8 #define LE 2

5.6.1.9 #define LINEAR 1

5.6.1.10 #define LQ 1

5.6.1.11 #define maxIter 200000000

5.6.1.12 #define maxMoves 600

5.6.1.13 #define nonTerm 0.000000000001

5.6.1.14 #define OBJ 0

5.6.1.15 #define PRINT TRUE

5.6.1.16 #define RANDOMSEED 1337

5.6.1.17 #define SOFT 2

5.6.1.18 #define SUM 1

5.6.1.19 #define SWAP 2

5.6.1.20 #define tabulistLB 5

5.6.1.21 #define tabulistUB 20

5.6.1.22 #define VALUECHANGE 3

5.6.1.23 #define weight 100

5.6.2 Typedef Documentation

5.6.2.1 typedef std::vector<constraintContainer> allConstraints

5.6.2.2 typedef double coefType

5.6.2.3 typedef std::shared_ptr<Constraint> constraint

5.6.2.4 typedef std::shared_ptr<std::vector<constraint> > constraintContainer

5.6.2.5 typedef std::shared_ptr<Invariant> invariant

5.6.2.6 typedef std::vector<invariant> InvariantContainer

5.6.2.7 typedef std::shared_ptr<Invariant> updateType

5.6.2.8 typedef InvariantContainer updateVector

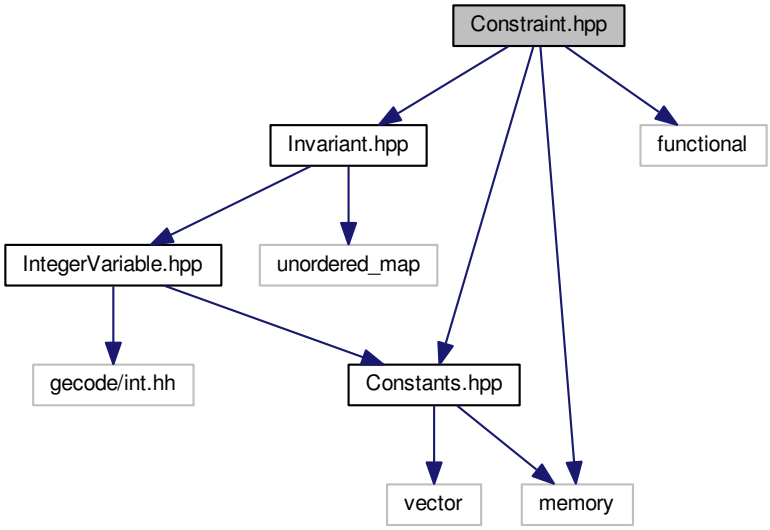
5.6.2.9 typedef std::vector<IntegerVariable*> variableContainer

5.6.2.10 typedef std::vector<constraint> VariableInConstraints

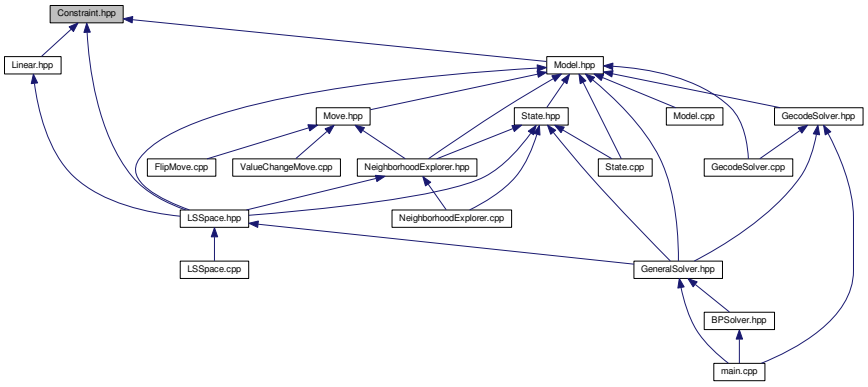
5.7 Constraint.hpp File Reference

```
#include "Invariant.hpp"
#include <memory>
#include <functional>
#include "Constants.hpp"
```

Include dependency graph for Constraint.hpp:



This graph shows which files directly or indirectly include this file:



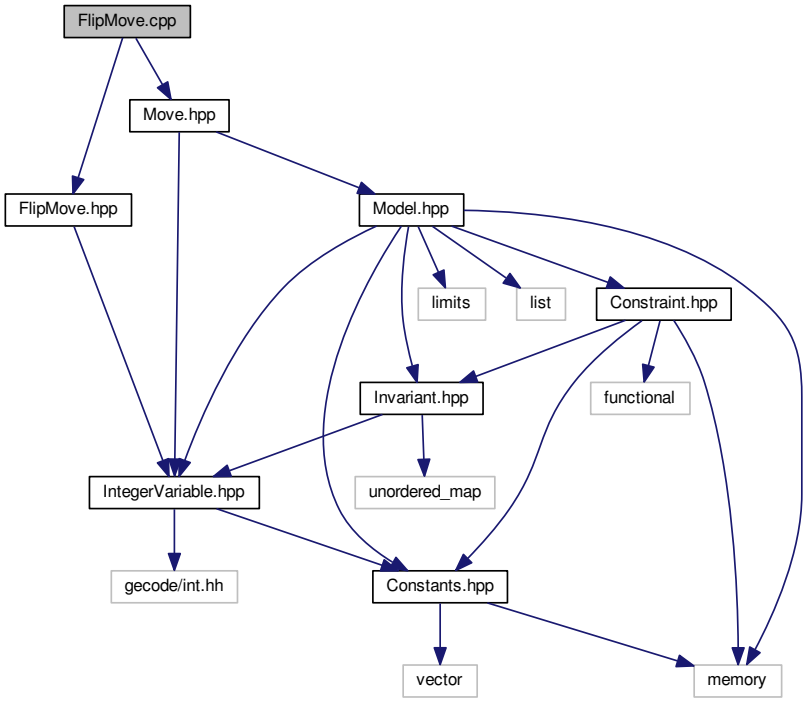
Classes

- class Constraint
- struct Constraint::SortGreater
- class ConstraintSorter

5.8 FlipMove.cpp File Reference

```
#include "FlipMove.hpp"
#include "Move.hpp"
```

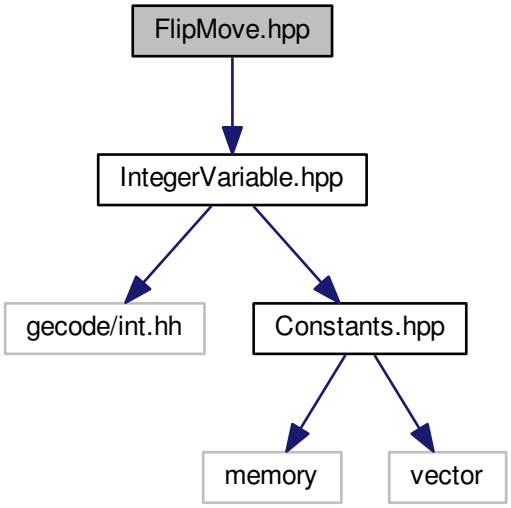
Include dependency graph for FlipMove.cpp:



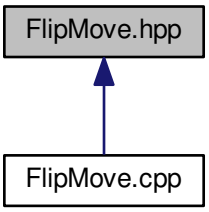
5.9 FlipMove.hpp File Reference

```
#include "IntegerVariable.hpp"
```

Include dependency graph for FlipMove.hpp:



This graph shows which files directly or indirectly include this file:



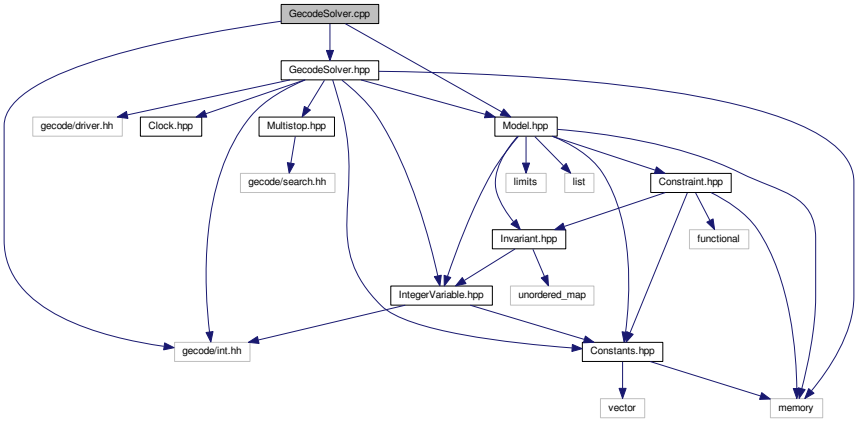
Classes

- class FlipMove

5.10 GecodeSolver.cpp File Reference

```
#include <gecode/int.hh>
#include "GecodeSolver.hpp"
#include "Model.hpp"
```

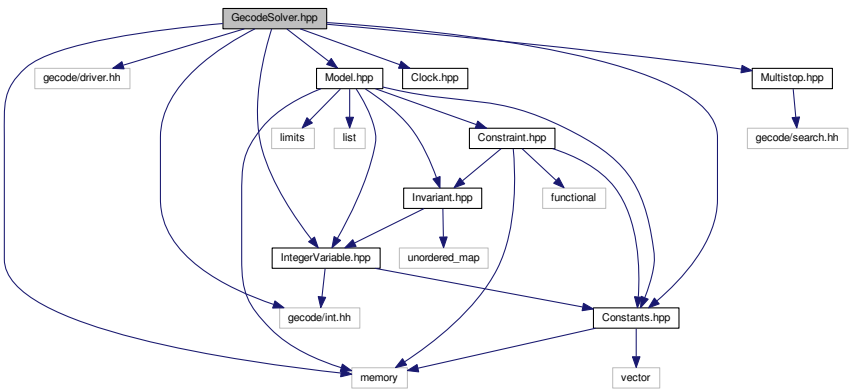
Include dependency graph for GecodeSolver.cpp:



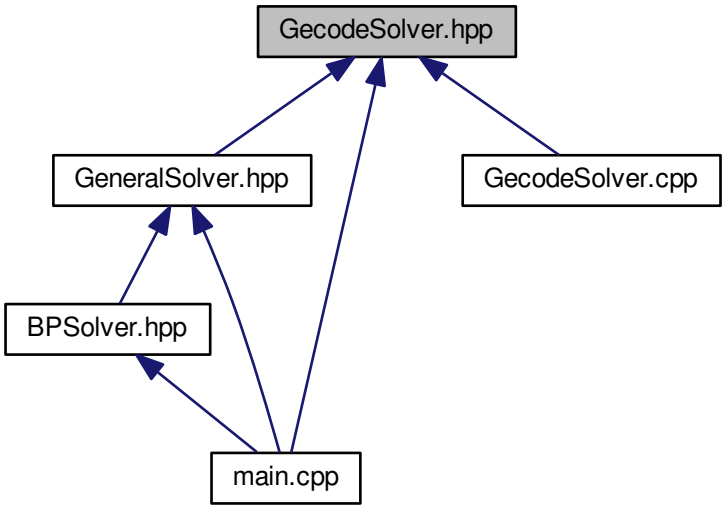
5.11 GecodeSolver.hpp File Reference

```
#include <memory>
#include <gecode/driver.hh>
#include <gecode/int.hh>
#include "IntegerVariable.hpp"
#include "Clock.hpp"
#include "Constants.hpp"
#include "Model.hpp"
#include "Multistop.hpp"
```

Include dependency graph for GecodeSolver.hpp:



This graph shows which files directly or indirectly include this file:



Classes

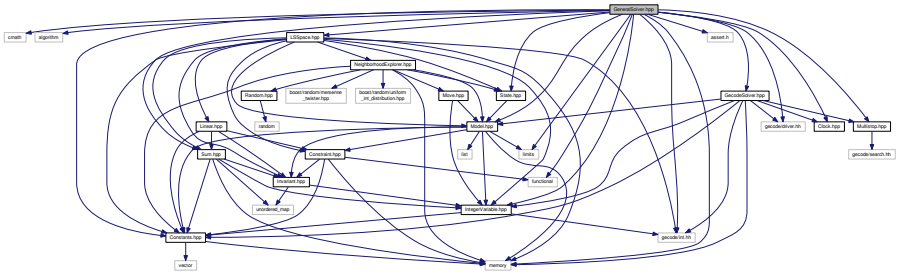
- class GecodeSolver

5.12 GeneralSolver.cpp File Reference

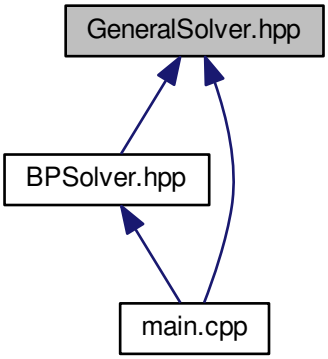
5.13 GeneralSolver.hpp File Reference

```
#include <cmath>
#include <algorithm>
#include "LSSpace.hpp"
#include <assert.h>
#include "Model.hpp"
#include <gecode/driver.hh>
#include <gecode/int.hh>
#include "Constants.hpp"
#include "GecodeSolver.hpp"
#include <limits>
#include "IntegerVariable.hpp"
#include "Multistop.hpp"
#include "Sum.hpp"
#include <memory>
#include "State.hpp"
#include <functional>
#include "Clock.hpp"
```

Include dependency graph for GeneralSolver.hpp:



This graph shows which files directly or indirectly include this file:

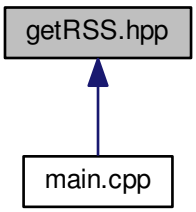


Classes

- class GeneralSolver

5.14 getRSS.hpp File Reference

This graph shows which files directly or indirectly include this file:



Functions

- size_t getPeakRSS ()
- size_t getCurrentRSS ()

5.14.1 Function Documentation

5.14.1.1 size_t getCurrentRSS ( )

Returns the current resident set size (physical memory use) measured in bytes, or zero if the value cannot be determined on this OS.

5.14.1.2 size_t getPeakRSS ( )

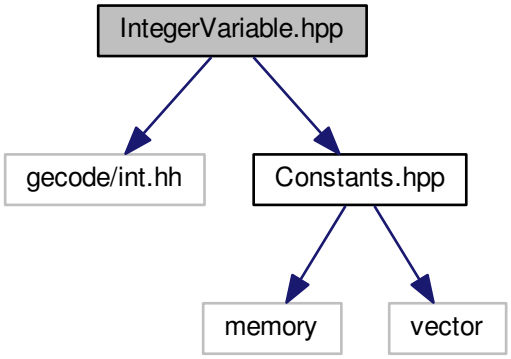
Returns the peak (maximum so far) resident set size (physical memory use) measured in bytes, or zero if the value cannot be determined on this OS.

5.15 IntegerVariable.cpp File Reference

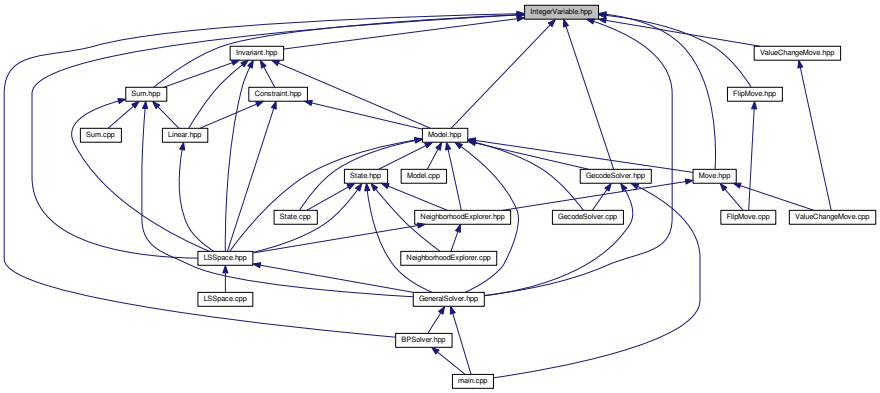
5.16 IntegerVariable.hpp File Reference

```
#include <gecode/int.hh>
#include "Constants.hpp"
```

Include dependency graph for IntegerVariable.hpp:



This graph shows which files directly or indirectly include this file:



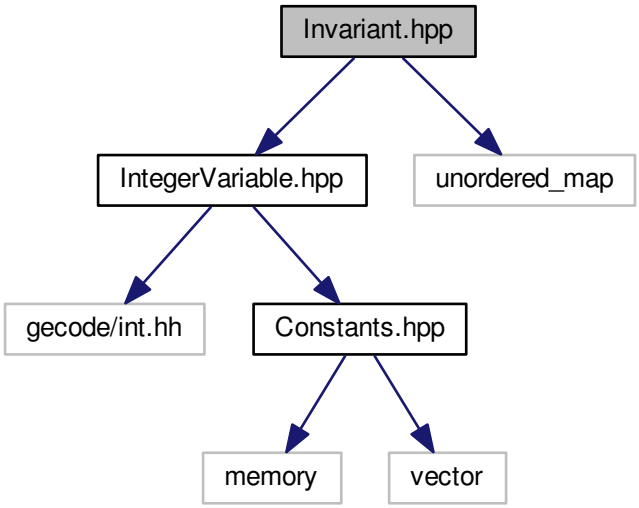
Classes

- class IntegerVariable

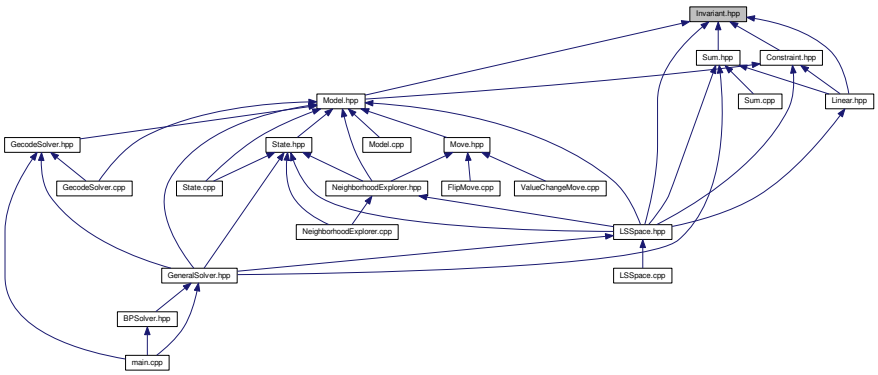
5.17 Invariant.hpp File Reference

```
#include "IntegerVariable.hpp"
#include <unordered_map>
```

Include dependency graph for Invariant.hpp:



This graph shows which files directly or indirectly include this file:



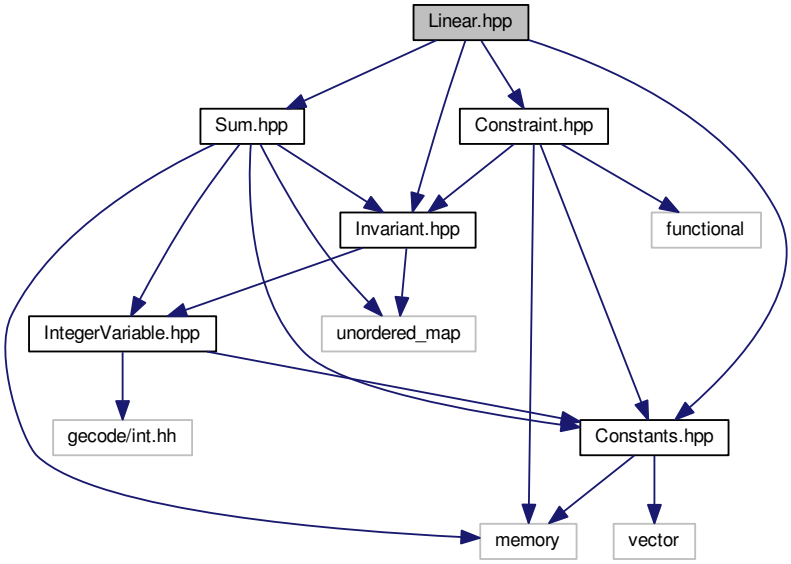
Classes

- class Invariant

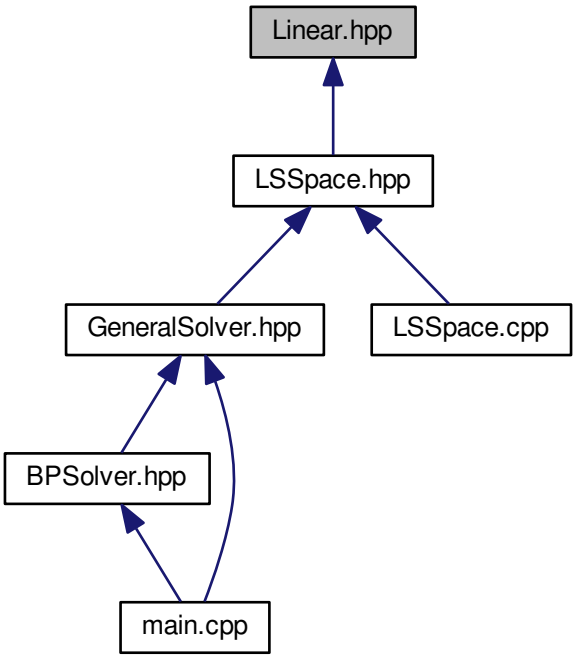
5.18 Linear.hpp File Reference

```
#include "Sum.hpp"
#include "Invariant.hpp"
#include "Constraint.hpp"
#include "Constants.hpp"
```

Include dependency graph for Linear.hpp:



This graph shows which files directly or indirectly include this file:



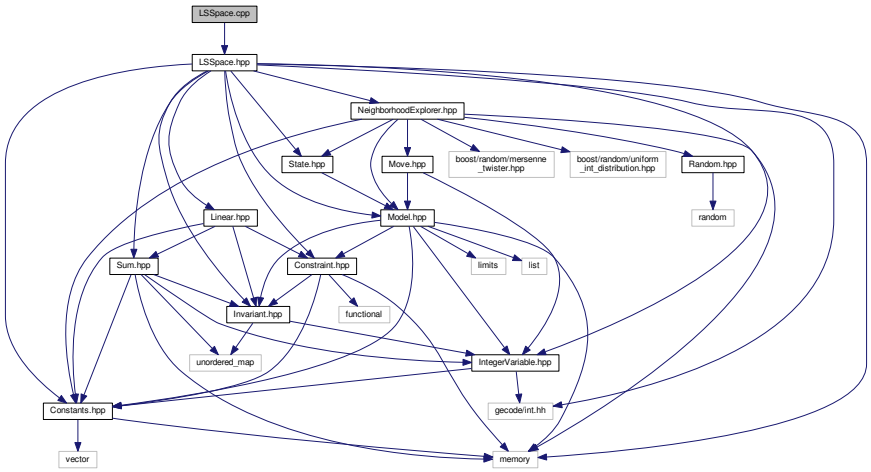
Classes

- class Linear

5.19 LSSpace.cpp File Reference

```
#include "LSSpace.hpp"
```

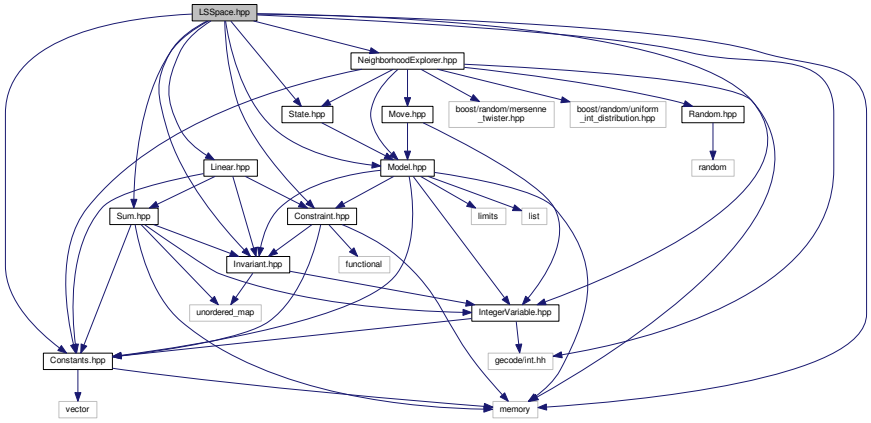
Include dependency graph for LSSpace.cpp:



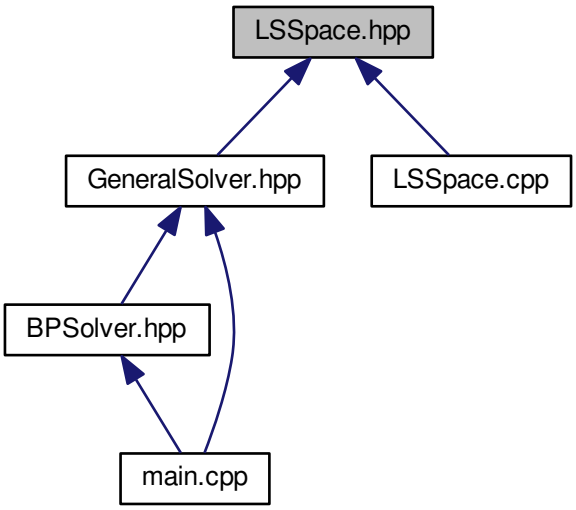
5.20 LSSpace.hpp File Reference

```
#include <gecode/int.hh>
#include "IntegerVariable.hpp"
#include "Invariant.hpp"
#include "Constraint.hpp"
#include "Sum.hpp"
#include "Linear.hpp"
#include "Model.hpp"
#include "NeighborhoodExplorer.hpp"
#include "Constants.hpp"
#include <memory>
#include "State.hpp"
```

Include dependency graph for LSSpace.hpp:



This graph shows which files directly or indirectly include this file:



Classes

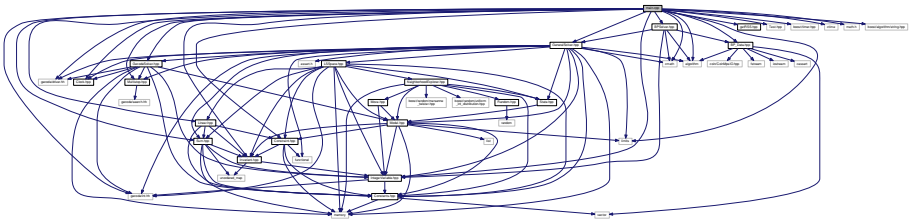
- class LSSpace

5.21 main.cpp File Reference

```
#include <gecode/driver.hh>
```

```
#include <gecode/int.hh>
#include "BP_Data.hpp"
#include <cmath>
#include <algorithm>
#include <limits>
#include "BPSolver.hpp"
#include "Clock.hpp"
#include "getRSS.hpp"
#include "GeneralSolver.hpp"
#include "GecodeSolver.hpp"
#include "Test.hpp"
#include "Multistop.hpp"
#include <boost/algorithm/string.hpp>
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗argv[])

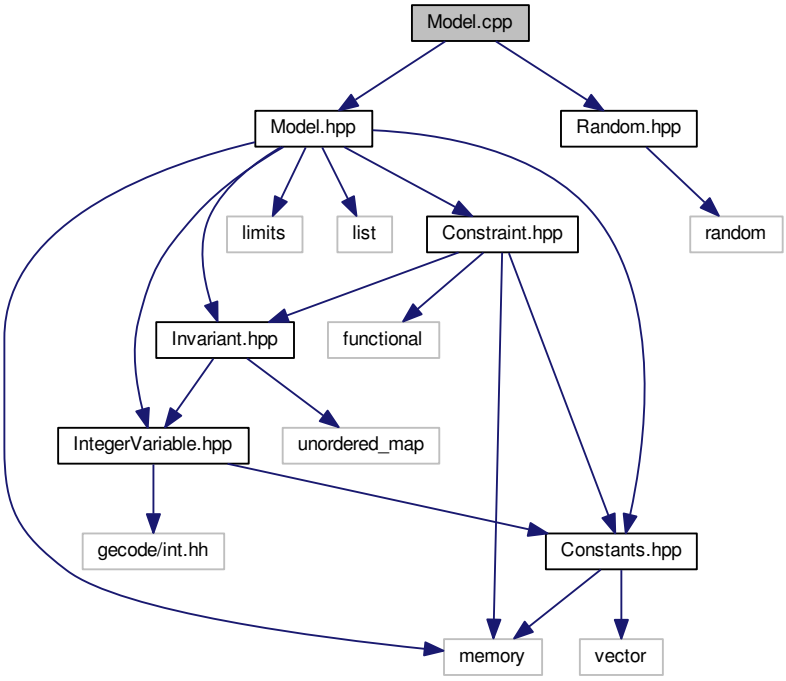
5.21.1 Function Documentation

5.21.1.1 int main (int *argc*, char ∗ *argv[]*)

5.22 Model.cpp File Reference

```
#include "Model.hpp"
#include "Random.hpp"
```

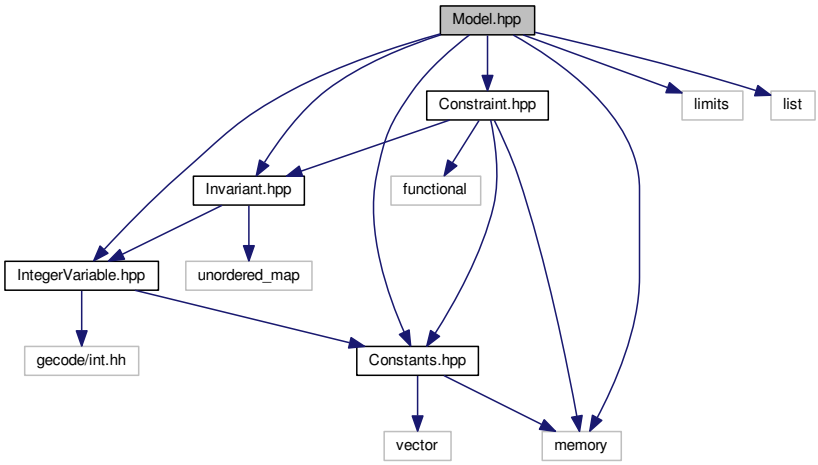
Include dependency graph for Model.cpp:



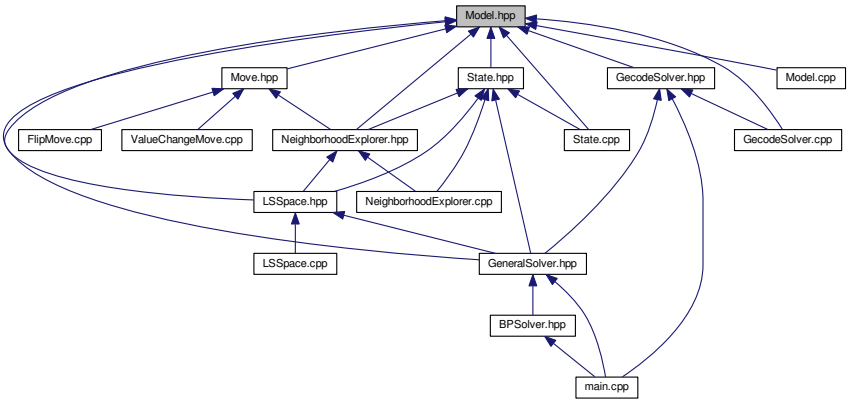
5.23 Model.hpp File Reference

```
#include "IntegerVariable.hpp"
#include "Invariant.hpp"
#include "Constraint.hpp"
#include "Constants.hpp"
#include <limits>
#include <memory>
#include <list>
```

Include dependency graph for Model.hpp:



This graph shows which files directly or indirectly include this file:



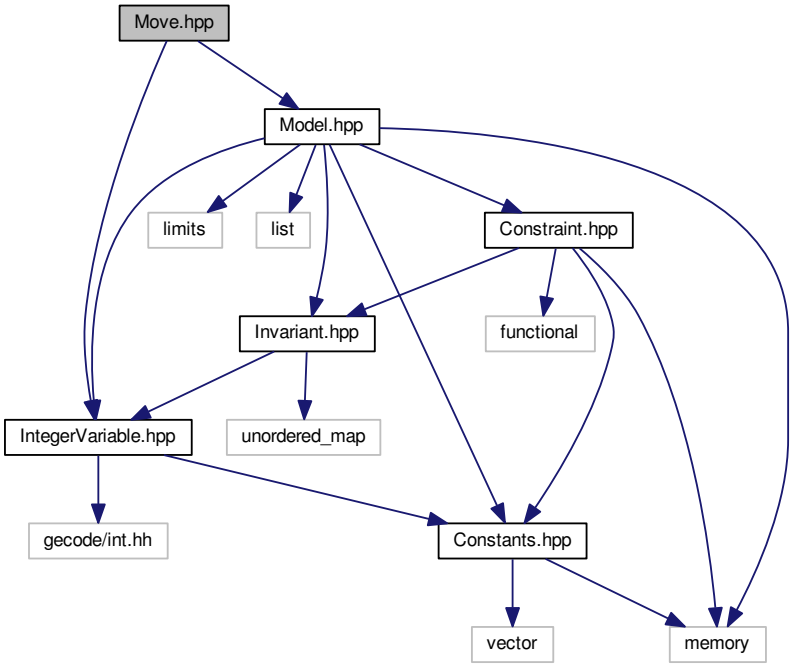
Classes

- class Model

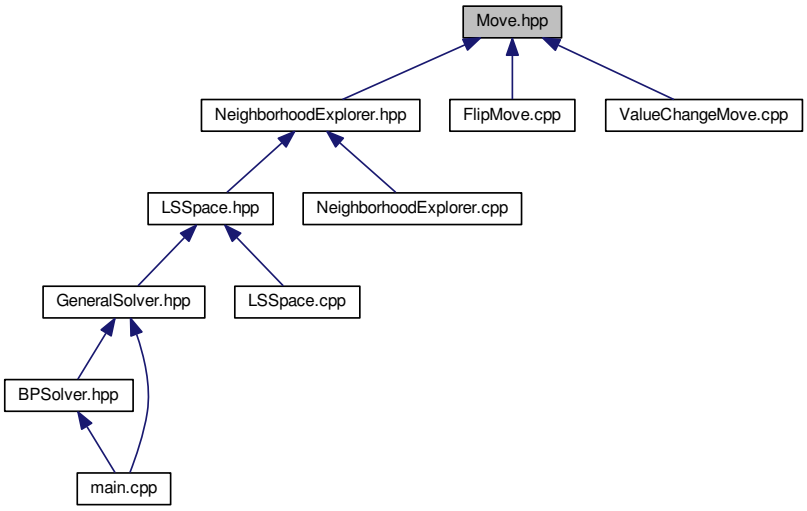
5.24 Move.hpp File Reference

```
#include "Model.hpp"
#include "IntegerVariable.hpp"
```

Include dependency graph for Move.hpp:



This graph shows which files directly or indirectly include this file:



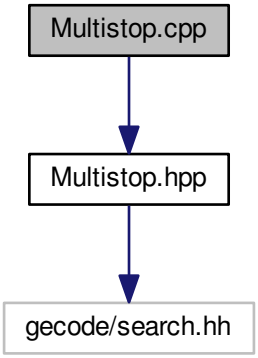
Classes

- class Move

5.25 Multistop.cpp File Reference

```
#include "Multistop.hpp"
```

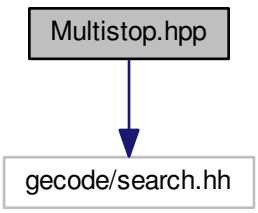
Include dependency graph for Multistop.cpp:



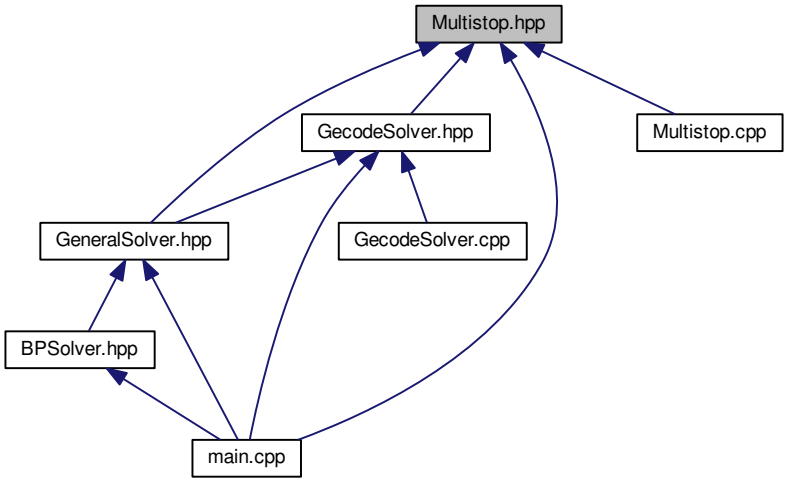
5.26 Multistop.hpp File Reference

```
#include <gecode/search.hh>
```

Include dependency graph for Multistop.hpp:



This graph shows which files directly or indirectly include this file:



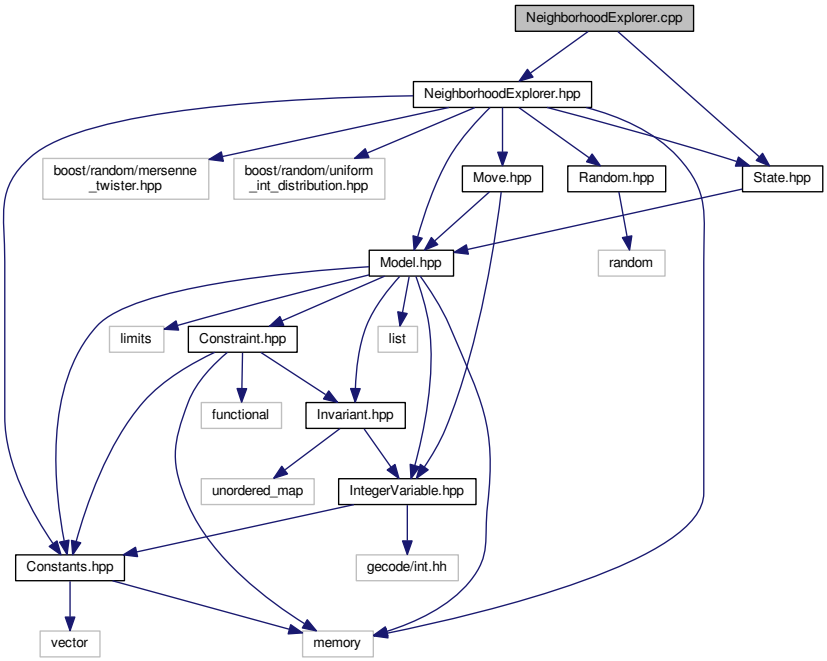
Classes

- class Multistop

5.27 NeighborhoodExplorer.cpp File Reference

```
#include "NeighborhoodExplorer.hpp"
#include "State.hpp"
```

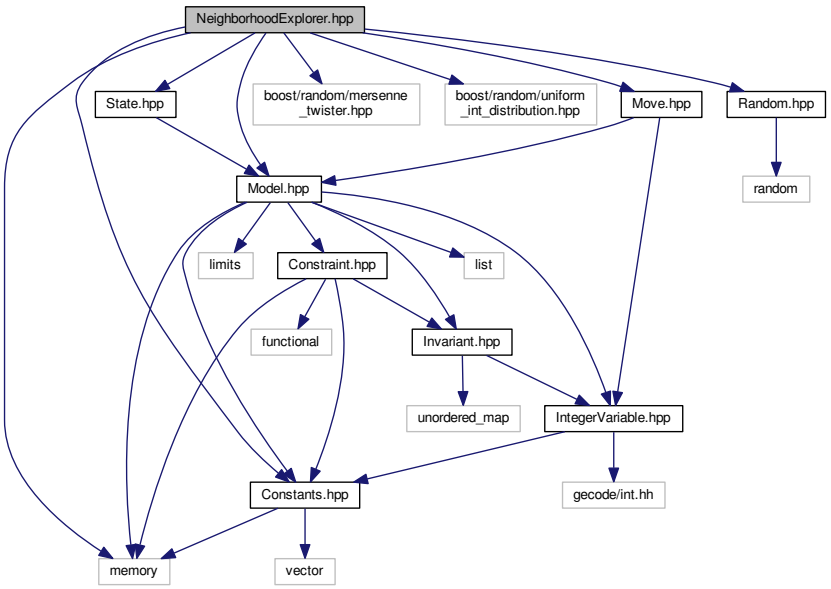
Include dependency graph for NeighborhoodExplorer.cpp:



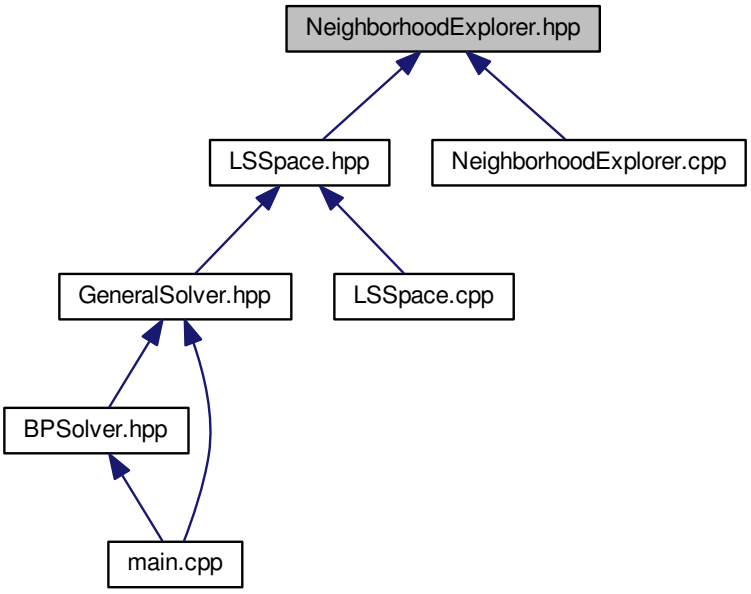
5.28 NeighborhoodExplorer.hpp File Reference

```
#include "Move.hpp"
#include "Model.hpp"
#include "Constants.hpp"
#include <memory>
#include "boost/random/mersenne_twister.hpp"
#include "boost/random/uniform_int_distribution.hpp"
#include "Random.hpp"
#include "State.hpp"
```

Include dependency graph for NeighborhoodExplorer.hpp:



This graph shows which files directly or indirectly include this file:



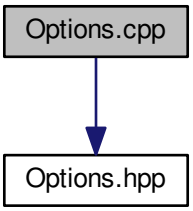
Classes

- class NeighborhoodExplorer

5.29 Options.cpp File Reference

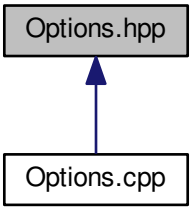
```
#include "Options.hpp"
```

Include dependency graph for Options.cpp:



5.30 Options.hpp File Reference

This graph shows which files directly or indirectly include this file:



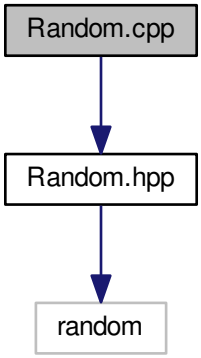
Classes

- class Options

5.31 Random.cpp File Reference

```
#include "Random.hpp"
```

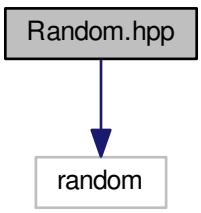
Include dependency graph for Random.cpp:



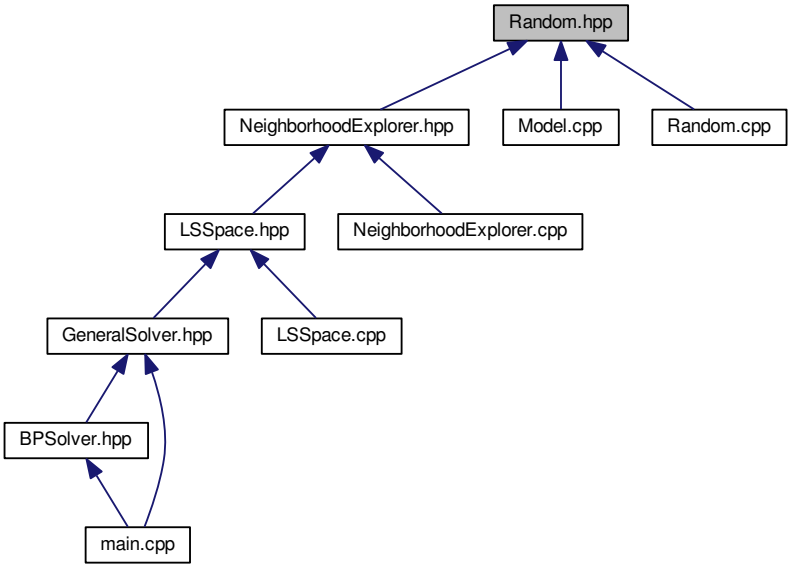
5.32 Random.hpp File Reference

```
#include <random>
```

Include dependency graph for Random.hpp:



This graph shows which files directly or indirectly include this file:



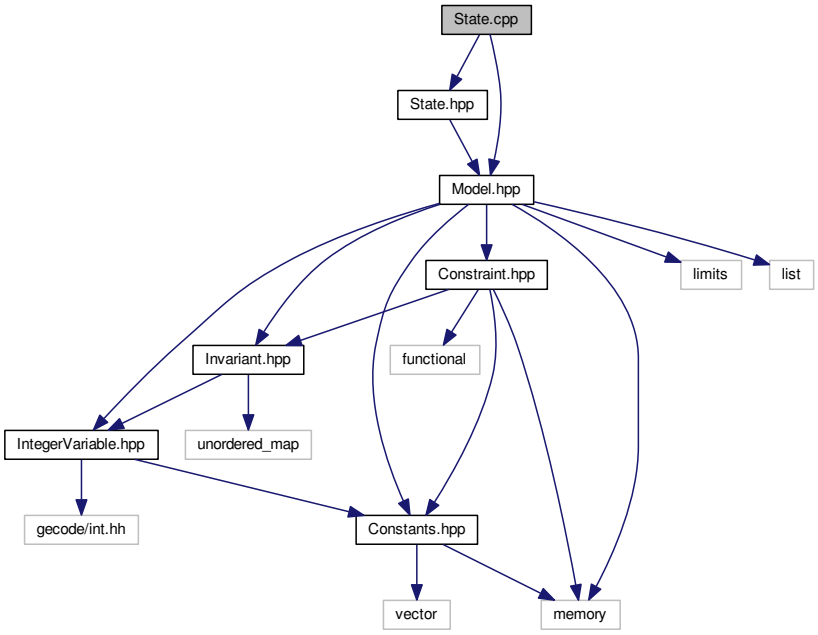
Classes

- class Random

5.33 State.cpp File Reference

```
#include "State.hpp"
#include "Model.hpp"
```

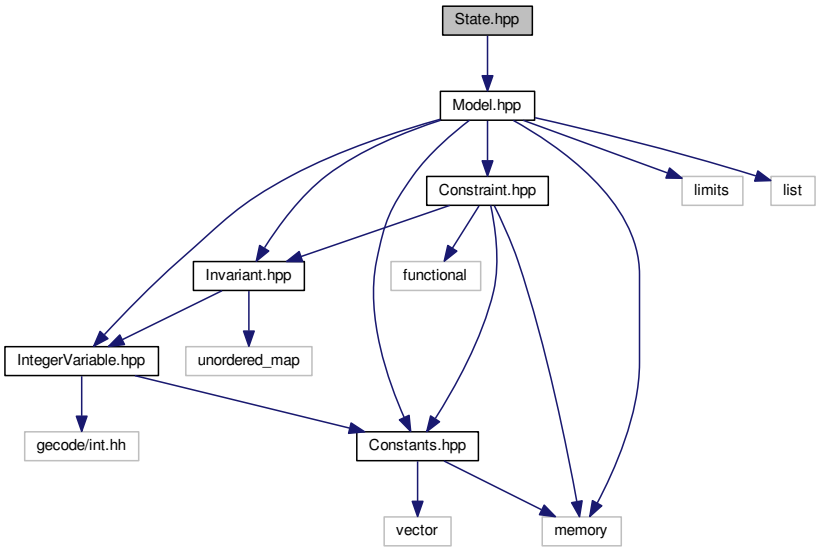
Include dependency graph for State.cpp:



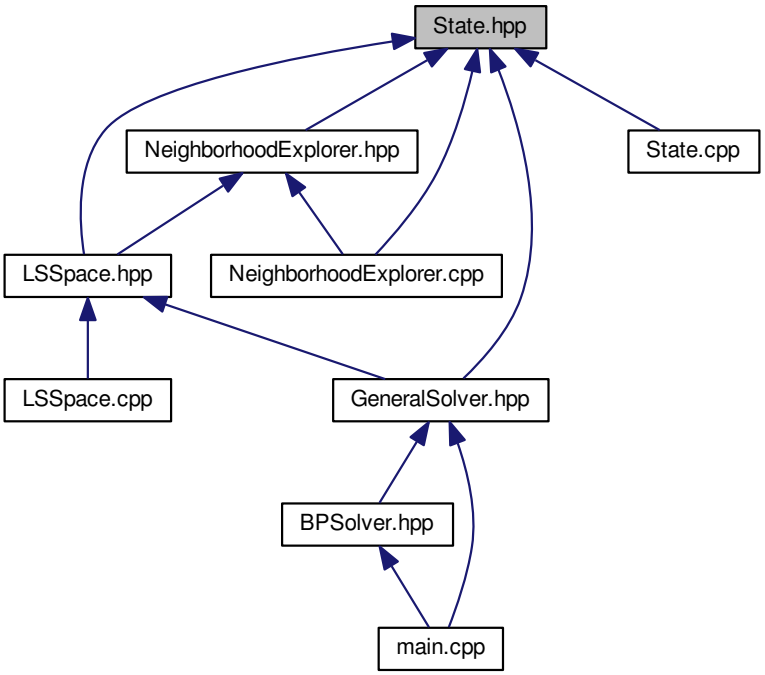
5.34 State.hpp File Reference

```
#include "Model.hpp"
```

Include dependency graph for State.hpp:



This graph shows which files directly or indirectly include this file:



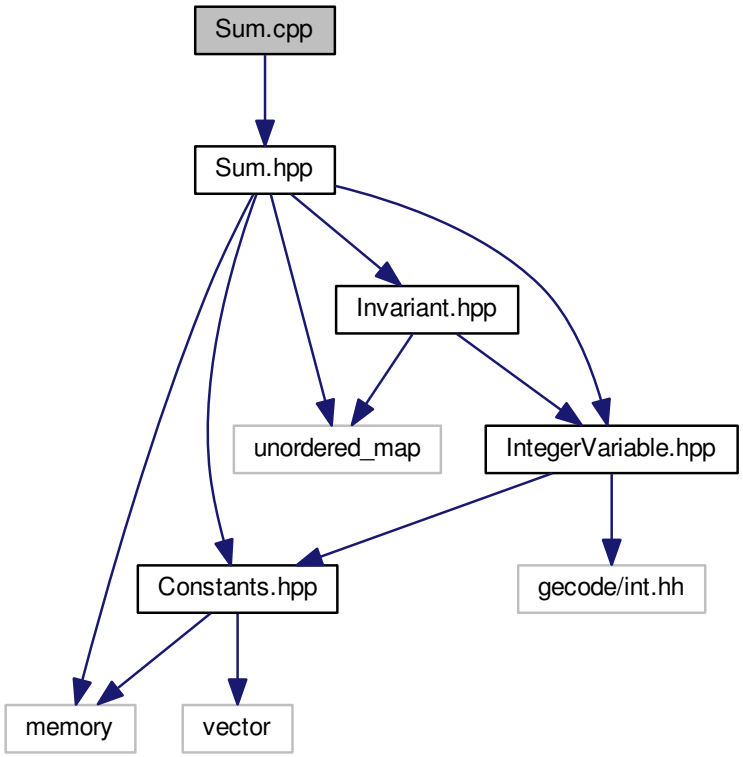
Classes

- class State

5.35 Sum.cpp File Reference

```
#include "Sum.hpp"
```

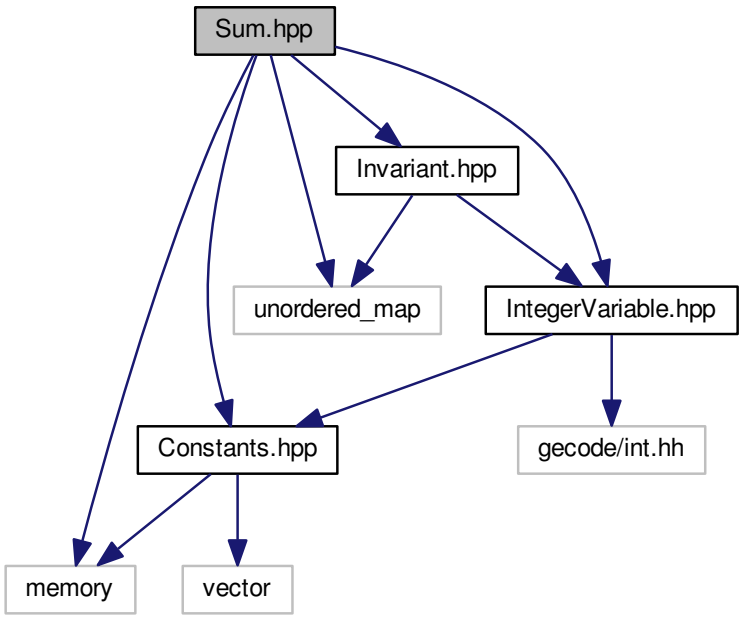
Include dependency graph for Sum.cpp:



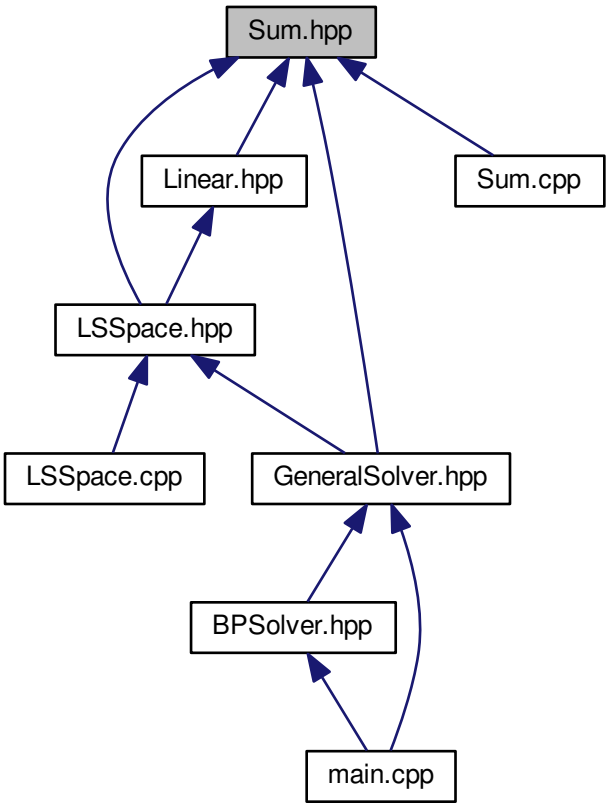
5.36 Sum.hpp File Reference

```
#include <memory>
#include "Constants.hpp"
#include <unordered_map>
#include "IntegerVariable.hpp"
#include "Invariant.hpp"
```

Include dependency graph for Sum.hpp:



This graph shows which files directly or indirectly include this file:



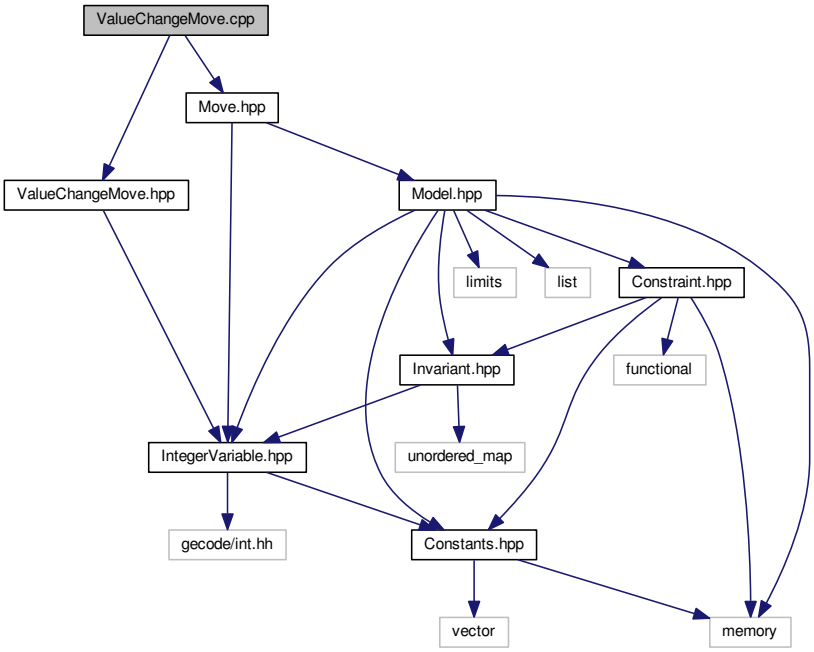
Classes

- class Sum

5.37 ValueChangeMove.cpp File Reference

```
#include "ValueChangeMove.hpp"
#include "Move.hpp"
```

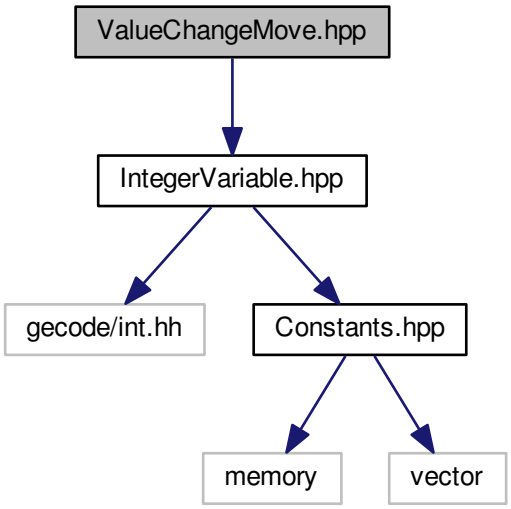
Include dependency graph for ValueChangeMove.cpp:



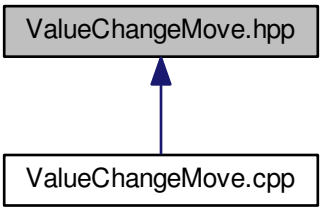
5.38 ValueChangeMove.hpp File Reference

```
#include "IntegerVariable.hpp"
```

Include dependency graph for ValueChangeMove.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ValueChangeMove

Index