Smi

0.95

Generated by Doxygen 1.8.9.1

Thu Oct 8 2015 22:44:46

Contents

1	Hiera	archical Index	1
	1.1	Class Hierarchy	1
2	Clas	s Index	13
	2.1	Class List	13
3	File I	Index	15
	3.1	File List	15
4	Clas	s Documentation	17
	4.1	SmiCoreCombineAdd Class Reference	17
	4.2	SmiCoreCombineReplace Class Reference	17
		4.2.1 Detailed Description	17
	4.3	SmiCoreCombineRule Class Reference	17
		4.3.1 Detailed Description	18
	4.4	SmiCoreData Class Reference	18
		4.4.1 Detailed Description	18
	4.5	SmiDiscreteDistribution Class Reference	18
		4.5.1 Detailed Description	19
	4.6	SmiDiscreteEvent Class Reference	19
		4.6.1 Detailed Description	19
	4.7	SmiDiscreteRV Class Reference	19
		4.7.1 Detailed Description	19
	4.8	SmiLinearData Class Reference	19
		4.8.1 Detailed Description	20
	4.9	SmiMessage Class Reference	20
		4.9.1 Detailed Description	
	4.10	SmiNodeData Class Reference	
		4.10.1 Detailed Description	20

iv CONTENTS

4.11	SmiQuadraticData Class Reference	20
	4.11.1 Detailed Description	21
4.12	SmiQuadraticDataDC Class Reference	21
	4.12.1 Detailed Description	21
4.13	$SmiScenarioTree < T > Class \ Template \ Reference \\ \ \ldots \\ \ \ldots$	21
	4.13.1 Detailed Description	22
	4.13.2 Member Function Documentation	22
	4.13.2.1 scenBegin	22
	4.13.2.2 getRoot	22
	4.13.2.3 getLeaf	22
	4.13.2.4 find	23
	4.13.2.5 addPathtoLeaf	23
	4.13.2.6 addPathtoLeaf	23
4.14	SmiScnModel Class Reference	23
	4.14.1 Detailed Description	26
	4.14.2 Member Function Documentation	26
	4.14.2.1 generateScenario	26
	4.14.2.2 generateScenario	26
	4.14.2.3 generateScenario	27
	4.14.2.4 generateScenario	27
4.15	SmiScnModelAddNode Class Reference	27
	4.15.1 Detailed Description	27
4.16	SmiScnModelDeleteNode Class Reference	27
	4.16.1 Detailed Description	27
4.17	SmiScnNode Class Reference	28
	4.17.1 Detailed Description	28
4.18	SmiSmpsCardReader Class Reference	28
	4.18.1 Detailed Description	28
4.19	SmiSmpsIO Class Reference	28
	4.19.1 Detailed Description	28
4.20	SmiTreeNode< T > Class Template Reference	29
	4.20.1 Detailed Description	29
Index		31

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

```
_EKKfactinfo[external]
AbcDualRowPivot[external]
         AbcDualRowDantzig[external]
          AbcDualRowSteepest[external]
AbcMatrix[external]
AbcMatrix2[external]
AbcMatrix3[external]
AbcNonLinearCost[external]
AbcPrimalColumnPivot[external]
         AbcPrimalColumnDantzig[external]
          AbcPrimalColumnSteepest[external]
AbcSimplexFactorization[external]
AbcTolerancesEtc [external]
AbcWarmStartOrganizer[external]
forcing constraint action::action[external]
doubleton_action::action[external]
tripleton action::action[external]
remove_fixed_action::action[external]
std::allocator< T >
ampl info[external]
OsiSolverInterface::ApplyCutsReturnCode \cite{Continuous} and \c
std::array< T >
std::auto_ptr< T >
auxiliary_graph[external]
std::basic\_string < Char >
         std::string
         std::wstring
std::basic string< char >
std::basic_string< wchar_t >
 std::bitset < Bits >
BitVector128 [external]
blockStruct[external]
blockStruct3[external]
 ClpNode::branchState [external]
```

2 Hierarchical Index

```
CbcOrClpParam[external]
Cgl012Cut[external]
cgl arc[external]
cgl graph[external]
cgl_node[external]
CglBK[external]
CglCutGenerator[external]
   CglAllDifferent[external]
   CglClique[external]
     CglFakeClique[external]
   CglDuplicateRow [external]
   CglFlowCover[external]
   Cg|GMI[external]
   CglGomory[external]
   CglImplication [external]
   CglKnapsackCover[external]
   CglLandP[external]
   CglLiftAndProject[external]
   CglMixedIntegerRounding[external]
   CglMixedIntegerRounding2[external]
   CglOddHole[external]
   CglProbing[external]
   CglRedSplit[external]
   CglRedSplit2[external]
   CglResidualCapacity [external]
   CglSimpleRounding[external]
   CglStored[external]
  CglTwomir[external]
   CglZeroHalf[external]
CglFlowVUB[external]
CglHashLink[external]
LAP::CglLandPSimplex[external]
CglMixIntRoundVUB[external]
CglMixIntRoundVUB2[external]
CglParam[external]
  CglGMIParam [external]
   CglLandP::Parameters [external]
   CglRedSplit2Param[external]
   CglRedSplitParam [external]
CglPreProcess [external]
CglTreeInfo[external]
   CglTreeProbingInfo[external]
CglUniqueRowCuts[external]
CliqueEntry[external]
CglProbing::CliqueType [external]
ClpCholeskyBase[external]
   ClpCholeskyDense[external]
   ClpCholeskyMumps[external]
   ClpCholeskyTaucs[external]
  ClpCholeskyUfl[external]
  ClpCholeskyWssmp[external]
   ClpCholeskyWssmpKKT[external]
ClpCholeskyDenseC[external]
ClpConstraint[external]
```

1.1 Class Hierarchy 3

```
ClpConstraintLinear[external]
   ClpConstraintQuadratic[external]
ClpDataSave[external]
ClpDisasterHandler[external]
   OsiClpDisasterHandler[external]
ClpDualRowPivot[external]
  ClpDualRowDantzig[external]
   ClpDualRowSteepest[external]
ClpEventHandler[external]
   MyEventHandler[external]
ClpFactorization[external]
ClpHashValue[external]
ClpLsqr[external]
ClpMatrixBase [external]
   ClpDummyMatrix[external]
   ClpNetworkMatrix[external]
  ClpPackedMatrix[external]
      ClpDynamicMatrix[external]
        ClpDynamicExampleMatrix[external]
     ClpGubMatrix[external]
        ClpGubDynamicMatrix[external]
   ClpPlusMinusOneMatrix[external]
ClpModel[external]
  ClpInterior[external]
     ClpPdco[external]
      ClpPredictorCorrector[external]
  ClpSimplex[external]
     AbcSimplex [external]
        AbcSimplexDual[external]
        AbcSimplexPrimal[external]
     ClpSimplexDual[external]
     ClpSimplexOther[external]
     ClpSimplexPrimal[external]
        ClpSimplexNonlinear[external]
ClpNetworkBasis [external]
ClpNode [external]
ClpNodeStuff[external]
ClpNonLinearCost[external]
ClpObjective [external]
   ClpLinearObjective [external]
   ClpQuadraticObjective [external]
ClpPackedMatrix2[external]
ClpPackedMatrix3[external]
ClpPdcoBase[external]
ClpPresolve[external]
ClpPrimalColumnPivot[external]
   ClpPrimalColumnDantzig[external]
  ClpPrimalColumnSteepest[external]
   ClpPrimalQuadraticDantzig[external]
ClpSimplexProgress[external]
ClpSolve[external]
ClpTrustedData[external]
CoinAbcAnyFactorization[external]
   CoinAbcDenseFactorization[external]
```

4 Hierarchical Index

```
CoinAbcTypeFactorization[external]
CoinAbcStack[external]
CoinAbcStatistics [external]
CoinAbsFltEq[external]
CoinArrayWithLength [external]
   CoinArbitraryArrayWithLength [external]
   CoinBigIndexArrayWithLength [external]
   CoinDoubleArrayWithLength [external]
   CoinFactorizationDoubleArrayWithLength [external]
   CoinFactorizationLongDoubleArrayWithLength [external]
   CoinIntArrayWithLength [external]
   CoinUnsignedIntArrayWithLength [external]
   CoinVoidStarArrayWithLength [external]
CoinBaseModel[external]
   CoinModel[external]
   CoinStructuredModel[external]
CoinBuild [external]
CoinDenseVector< T > [external]
CoinError[external]
   CglLandP::NoBasisError[external]
   CglLandP::SimplexInterfaceError[external]
CoinExternalVectorFirstGreater_2< class, class, class > [external]
CoinExternalVectorFirstGreater_3 < class, class, class, class, class > [external]
CoinExternalVectorFirstLess 2< class, class, class > [external]
CoinExternalVectorFirstLess 3< class, class, class, class > [external]
CoinFactorization [external]
CoinFileIOBase [external]
   CoinFileInput[external]
   CoinFileOutput[external]
CoinFirstAbsGreater_2< class, class > [external]
CoinFirstAbsGreater 3< class, class, class > [external]
CoinFirstAbsLess 2 < class, class > [external]
CoinFirstAbsLess_3< class, class, class > [external]
CoinFirstGreater 2< class, class > [external]
CoinFirstGreater_3< class, class, class > [external]
CoinFirstLess 2< class, class > [external]
CoinFirstLess 3< class, class, class > [external]
CoinLpIO::CoinHashLink[external]
CoinMpsIO::CoinHashLink[external]
ClpHashValue::CoinHashLink[external]
CoinIndexedVector[external]
   CoinPartitionedVector[external]
   LAP::TabRow[external]
CoinLpIO [external]
CoinMessageHandler[external]
   MyMessageHandler[external]
CoinMessages [external]
   CglMessage[external]
   ClpMessage [external]
   CoinMessage[external]
   LAP::LandPMessages[external]
   LAP::LapMessages[external]
   SmiMessage . . . . . . .
CoinModelHash[external]
```

1.1 Class Hierarchy 5

CoinModelHash2[external] CoinModelHashLink[external] CoinModelInfo2[external] CoinModelLink[external] CoinModelLinkedList[external] CoinModelTriple[external] CoinModelTriple[external]
SmiSmpsCardReader
SmiSmpsIO
CoinOneMessage [external]
CoinOtherFactorization [external]
CoinDenseFactorization[external]
CoinOslFactorization [external]
CoinSimpFactorization[external]
CoinPackedMatrix[external]
CoinPackedVectorBase[external]
CoinPackedVector[external]
CoinShallowPackedVector[external]
<pre>CoinPair< S, T > [external]</pre>
CoinParam [external]
CoinPrePostsolveMatrix[external]
CoinPostsolveMatrix[external]
CoinPresolveMatrix [external]
CoinPresolveAction[external]
<pre>do_tighten_action[external]</pre>
<pre>doubleton_action[external]</pre>
<pre>drop_empty_cols_action[external]</pre>
<pre>drop_empty_rows_action[external]</pre>
<pre>drop_zero_coefficients_action[external]</pre>
<pre>dupcol_action[external]</pre>
<pre>duprow3_action[external]</pre>
<pre>duprow_action[external]</pre>
<pre>forcing_constraint_action[external]</pre>
<pre>gubrow_action[external]</pre>
<pre>implied_free_action[external]</pre>
<pre>isolated_constraint_action[external]</pre>
<pre>make_fixed_action[external]</pre>
<pre>remove_dual_action[external]</pre>
<pre>remove_fixed_action[external]</pre>
<pre>slack_doubleton_action[external]</pre>
<pre>slack_singleton_action[external]</pre>
<pre>subst_constraint_action[external]</pre>
<pre>tripleton_action[external]</pre>
<pre>twoxtwo_action[external]</pre>
<pre>useless_constraint_action[external]</pre>
CoinPresolveMonitor[external]
CoinRational[external]
CoinRelFltEq[external]
CoinSearchTreeBase[external]
CoinSearchTree < class > [external]
CoinSearchTreeCompareBest[external]
CoinSearchTreeCompareBreadth[external]
CoinSearchTreeCompareDepth [external]

6 Hierarchical Index

```
CoinSearchTreeComparePreferred[external]
CoinSearchTreeManager[external]
CoinSet[external]
   CoinSosSet[external]
CoinSnapshot[external]
CoinThreadRandom [external]
CoinTimer[external]
CoinTreeNode[external]
CoinTreeSiblings [external]
CoinTriple < S, T, U > [external]
CoinWarmStart[external]
   CoinWarmStartBasis [external]
      AbcWarmStart[external]
   CoinWarmStartDual[external]
   CoinWarmStartPrimalDual[external]
   CoinWarmStartVector< T > [external]
   CoinWarmStartVector < double > [external]
   CoinWarmStartVector< U > [external]
   CoinWarmStartVectorPair< T, U > [external]
CoinWarmStartDiff[external]
   CoinWarmStartBasisDiff[external]
   CoinWarmStartDualDiff[external]
   CoinWarmStartPrimalDualDiff[external]
   CoinWarmStartVectorDiff< T > [external]
   CoinWarmStartVectorDiff< double > [external]
   CoinWarmStartVectorDiff< U > [external]
   CoinWarmStartVectorPairDiff< T, U > [external]
CoinYacc[external]
std::complex
OsiCuts::const_iterator[external]
std::multiset < K >::const iterator
std::unordered set< K >::const iterator
std::map< K, T >::const_iterator
std::basic string< Char >::const iterator
std::string::const_iterator
std::multimap< K, T >::const iterator
std::wstring::const iterator
std::deque< T >::const iterator
std::list< T >::const iterator
std::unordered map< K, T>::const iterator
std::forward_list< T >::const_iterator
std::unordered multimap< K, T >::const iterator
std::set< K >::const iterator
std::unordered multiset< K >::const iterator
std::vector< T >::const_iterator
std::vector< T >::const_reverse_iterator
std::unordered_multiset< K >::const_reverse iterator
std::set< K >::const reverse iterator
std::unordered multimap< K, T>::const reverse iterator
std::basic_string< Char >::const_reverse_iterator
std::wstring::const reverse iterator
std::deque < T >::const_reverse_iterator
std::map < K, T >::const reverse iterator
std::forward list< T >::const reverse iterator
```

1.1 Class Hierarchy 7

```
std::unordered_map< K, T >::const_reverse_iterator
std::multimap< K, T >::const reverse iterator
std:: list < T > :: const\_reverse\_iterator
std::unordered set< K >::const reverse iterator
std::multiset < K > ::const\_reverse\_iterator
std::string::const_reverse_iterator
cut[external]
cut list[external]
cutParams[external]
LAP::Cuts[external]
cycle [external]
cycle_list[external]
std::deque< T >
std::deque < StdVectorDouble >
DGG_constraint_t[external]
DGG_data_t[external]
DGG_list_t[external]
disaggregationAction[external]
dropped zero[external]
dualColumnResult[external]
edge[external]
EKKHlink[external]
std::error_category
std::error code
std::error condition
std::exception
   std::bad alloc
   std::bad_cast
   std::bad_exception
   std::bad_typeid
   std::ios_base::failure
   std::logic error
      std::domain_error
      std::invalid argument
      std::length_error
      std::out of range
   std::runtime error
      std::overflow error
      std::range_error
      std::underflow error
FactorPointers [external]
std::forward list< T >
glp prob[external]
ldiot[external]
IdiotResult[external]
ilp[external]
Info[external]
info_weak[external]
std::ios base
   basic_ios < char >
   basic ios < wchar t >
   std::basic_ios
      basic istream < char >
      basic istream< wchar t>
```

8 Hierarchical Index

```
basic_ostream< char >
      basic ostream< wchar t>
      std::basic istream
          basic ifstream < char >
          basic_ifstream< wchar_t >
          basic iostream < char >
          basic iostream< wchar t >
          basic istringstream < char >
          basic istringstream< wchar t >
          std::basic ifstream
             std::ifstream
             std::wifstream
          std::basic iostream
             basic_fstream < char >
             basic fstream< wchar t >
             basic_stringstream< char >
             basic_stringstream< wchar_t >
             std::basic_fstream
                 std::fstream
                 std::wfstream
             std::basic stringstream
                 std::stringstream
                 std::wstringstream
          std::basic istringstream
             std::istringstream
             std::wistringstream
          std::istream
          std::wistream
      std::basic_ostream
          basic_iostream< char >
          basic_iostream< wchar_t >
          basic ofstream < char >
          basic_ofstream< wchar_t >
          basic ostringstream < char >
          basic_ostringstream< wchar_t >
          std::basic iostream
          std::basic ofstream
             std::ofstream
             std::wofstream
          std::basic_ostringstream
             std::ostringstream
             std::wostringstream
          std::ostream
          std::wostream
      std::ios
      std::wios
OsiCuts::iterator[external]
std::multiset< K >::iterator
std::set < K >::iterator
std::vector< T >::iterator
std::unordered multiset< K >::iterator
std::unordered_multimap< K, T >::iterator
std::unordered set< K >::iterator
std::string::iterator
```

1.1 Class Hierarchy 9

```
std::deque< T >::iterator
std::map < K, T >::iterator
std::forward list< T >::iterator
std::unordered map < K, T >::iterator
std::list< T >::iterator
std::wstring::iterator
std::multimap< K, T >::iterator
std::basic string< Char >::iterator
std::list< T >
log_var[external]
std::map< K, T>
std::map< int, CoinPackedVector * >
std::map< int, double * >
std::map< int, SmiTreeNode< SmiScnNode * > * >
std::map< int, SmiTreeNode< T > * >
std::map< string, int >
std::multimap< K, T >
std::multiset< K >
Options [external]
OsiAuxInfo[external]
   OsiBabSolver[external]
OsiBranchingInformation [external]
OsiBranchingObject [external]
   OsiTwoWayBranchingObject[external]
      OsiIntegerBranchingObject[external]
      OsiLotsizeBranchingObject [external]
      OsiSOSBranchingObject[external]
OsiChooseVariable [external]
   OsiChooseStrong[external]
OsiCut[external]
   OsiColCut[external]
   OsiRowCut[external]
      OsiRowCut2[external]
OsiCuts [external]
OsiHotInfo[external]
OsiObject[external]
   OsiObject2[external]
      OsiLotsize [external]
      OsiSimpleInteger[external]
      OsiSOS[external]
OsiPresolve [external]
OsiPseudoCosts [external]
OsiRowCutDebugger[external]
OsiSolverBranch [external]
OsiSolverInterface [external]
   OsiClpSolverInterface [external]
   OsiCpxSolverInterface [external]
   OsiGlpkSolverInterface [external]
   OsiGrbSolverInterface [external]
   OsiMskSolverInterface [external]
   OsiSpxSolverInterface [external]
   OsiXprSolverInterface [external]
OsiSolverResult[external]
Outfo [external]
```

10 Hierarchical Index

ClpSimplexOther::parametricsData[external]
<pre>parity_ilp [external]</pre>
AbcSimplexPrimal::pivotStruct[external]
<pre>pool_cut[external]</pre>
<pre>pool_cut_list[external]</pre>
<pre>presolvehlink[external]</pre>
std::priority_queue< T >
std::queue < T >
Coin::ReferencedObject[external]
std::vector < T >::reverse_iterator
std::string::reverse_iterator
std::unordered_multimap< K, T >::reverse_iterator
std::unordered_multiset< K >::reverse_iterator
std::multiset< K >::reverse_iterator
std::unordered_set< K >::reverse_iterator
std::multimap< K, T >::reverse_iterator
std::forward_list< T >::reverse_iterator
std::unordered_map< K, T >::reverse_iterator
std::list< T >::reverse_iterator
std::deque < T >::reverse_iterator
std::map< K, T >::reverse_iterator
std::wstring::reverse_iterator
std::basic_string< Char >::reverse_iterator
std::set < K >::reverse_iterator
scatterStruct[external]
select_cut[external]
<pre>separation_graph[external] stduagt < K ></pre>
std::set < K >
<pre>short_path_node[external] add::amount_ntr < T ></pre>
<pre>std::smart_ptr< T > Coin::SmartPtr< T > [external]</pre>
SmiCoreCombineRule
SmiCoreCombineAdd
SmiCoreCombineReplace
SmiCoreData
SmiDiscreteDistribution
SmiDiscreteRV
SmiLinearData
SmiDiscreteEvent
SmiNodeData
SmiQuadraticData
SmiQuadraticDataDC
SmiScenarioTree < T >
SmiScenarioTree < SmiScnNode * >
SmiScnModel
SmiScnModelAddNode
SmiScnModelDeleteNode
SmiSchNode
SmiTreeNode < T >
SmiTreeNode < SmiScnNode * >
std::stack< T >
<pre>symrec[external]</pre>
std::system_error

1.1 Class Hierarchy 11

```
OsiUnitTest::TestOutcome[external]
OsiUnitTest::TestOutcomes [external]
std::thread
std::unique_ptr< T >
std::unordered_map< K, T >
std::unordered_multimap< K, T >
std::unordered multiset< K >
std::unordered_set< K>
std::valarray< T >
LAP::Validator[external]
std::vector< T >
std::vector< ColumnSelectionStrategy >
std::vector< double *>
std::vector< double >
std::vector< int >
std::vector < RowSelectionStrategy >
std::vector < SmiDiscreteEvent * >
std::vector< SmiDiscreteRV *>
std::vector < SmiNodeData * >
std::vector < SmiScnNode * >
std::vector< SmiTreeNode< SmiScnNode * > * >
std::vector < SmiTreeNode < T > * >
std::vector< std::string >
std::vector< std::vector< int >>
std::weak_ptr< T >
Κ
S
Т
U
```

12 **Hierarchical Index**

Chapter 2

Class Index

2.1 Class List

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

SmiCoreCombineAdd
SmiCoreCombineReplace
SmiCoreCombineRule
This deals with combining Core and Stochastic data
SmiCoreData
SmiDiscreteDistribution
SmiDiscreteEvent
SmiDiscreteRV 19
SmiLinearData
SmiMessage
This deals with Clp messages (as against Osi messages etc)
SmiNodeData
SmiQuadraticData
SmiQuadraticDataDC
$SmiScenarioTree < T > \dots \dots$
SmiScnModel
SmiScnModel: COIN-SMI Scenario Model Class
SmiScnModelAddNode
SmiScnModelDeleteNode
SmiScnNode
SmiSmpsCardReader
SmiSmpsIO
SmiTreeNode< T >
Scenario Tree

14 **Class Index**

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

config_smi_default.h	 '?
SmiConfig.h	 ?
SmiCoreCombineRule.hpp	 ?
SmiDiscreteDistribution.hpp	 ?
SmiLinearData.hpp	 ?
SmiMessage.hpp	 ?
SmiQuadratic.hpp	 ?
SmiScenarioTree.hpp	 ?
SmiScnData.hpp	
SmiScnModel.hpp	 ?
SmiSmpsIQ.hpp	

16 File Index

Chapter 4

Class Documentation

4.1 SmiCoreCombineAdd Class Reference

Inheritance diagram for SmiCoreCombineAdd:

4.2 SmiCoreCombineReplace Class Reference

Inheritance diagram for SmiCoreCombineReplace:

Collaboration diagram for SmiCoreCombineReplace:

Public Member Functions

virtual void Process (double *d1, int o1, const CoinPackedVector &cpv2, char *type=0)
 Process.

4.2.1 Detailed Description

Definition at line 54 of file SmiCoreCombineRule.hpp.

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

SmiCoreCombineRule.hpp

4.3 SmiCoreCombineRule Class Reference

This deals with combining Core and Stochastic data.

#include <SmiCoreCombineRule.hpp>

Inheritance diagram for SmiCoreCombineRule:

Public Member Functions

Virtual Functions: Process and Diff

- virtual void Process (double *d1, int o1, const CoinPackedVector &cpv2, char *type=0)=0
 Process.
- virtual void Process (double *d1, int o1, const int len, const int *inds, const double *dels, char *type=0)=0
- virtual CoinPackedVector * Process (CoinPackedVector *cpv1, CoinPackedVector *cpv2, char *type=0)=0
- virtual int Process (double *dr, const int dr_len, CoinPackedVector *cpv, double *dels, int *indx)=0
- virtual int Process (double *dr, const int dr_len, const int cpv_nels, const int *cpv_ind, const double *cpv_els, double *dels, int *indx)=0
- virtual ∼SmiCoreCombineRule ()

4.3.1 Detailed Description

This deals with combining Core and Stochastic data.

In the Stochastic MPS standard, stochastic data updates the underlying core lp data. To specify a new scenario, one only has to identify those data that are different. So, in a sense, the stochastic data is really a "diff" between the scenario and the core data. This class specifies how to perform the "undiff", that is, how to combine core and stochastic data.

And of course, a complete implementation specifies the "diff" part as well. Now during a fit of original confusion in the birth of the SMPS standard, we decided to make default combine rule "replace", which has a rather special "diff", but we've learned to live with it.

There only needs to be one of these classes. so they're singletons.

Definition at line 36 of file SmiCoreCombineRule.hpp.

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

· SmiCoreCombineRule.hpp

4.4 SmiCoreData Class Reference

Public Member Functions

void addQuadraticObjectiveToCore (int *starts, int *indx, double *dels)
 Adds QP data after the constructor has been called.

4.4.1 Detailed Description

Definition at line 202 of file SmiScnData.hpp.

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

SmiScnData.hpp

4.5 SmiDiscreteDistribution Class Reference

Public Member Functions

 void addDiscreteRV (SmiDiscreteRV *s) add discrete RV • SmiDiscreteRV * getDiscreteRV (int i)

get discrete RV

· int getNumRV ()

get number of RV

SmiCoreData * getCore ()

get core model

void setCombineWithCoreRule (SmiCoreCombineRule *r)

set combine rule

SmiCoreCombineRule * getCombineWithCoreRule ()

get combine rule

SmiDiscreteDistribution (SmiCoreData *c, SmiCoreCombineRule *r=SmiCoreCombineReplace::Instance())
 constructor requires core data and combine rule

4.5.1 Detailed Description

Definition at line 24 of file SmiDiscreteDistribution.hpp.

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

· SmiDiscreteDistribution.hpp

4.6 SmiDiscreteEvent Class Reference

Inheritance diagram for SmiDiscreteEvent:

Collaboration diagram for SmiDiscreteEvent:

4.6.1 Detailed Description

Definition at line 69 of file SmiDiscreteDistribution.hpp.

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

SmiDiscreteDistribution.hpp

4.7 SmiDiscreteRV Class Reference

4.7.1 Detailed Description

Definition at line 82 of file SmiDiscreteDistribution.hpp.

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

· SmiDiscreteDistribution.hpp

4.8 SmiLinearData Class Reference

Inheritance diagram for SmiLinearData:

4.8.1 Detailed Description

Definition at line 23 of file SmiLinearData.hpp.

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

· SmiLinearData.hpp

4.9 SmiMessage Class Reference

This deals with Clp messages (as against Osi messages etc)

```
#include <SmiMessage.hpp>
```

Inheritance diagram for SmiMessage:

Collaboration diagram for SmiMessage:

Public Member Functions

Constructors etc

• SmiMessage (Language language=us_en)

Constructor.

4.9.1 Detailed Description

This deals with Clp messages (as against Osi messages etc)

Definition at line 20 of file SmiMessage.hpp.

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

· SmiMessage.hpp

4.10 SmiNodeData Class Reference

4.10.1 Detailed Description

Definition at line 27 of file SmiScnData.hpp.

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

· SmiScnData.hpp

4.11 SmiQuadraticData Class Reference

Inheritance diagram for SmiQuadraticData:

4.11.1 Detailed Description

Definition at line 21 of file SmiQuadratic.hpp.

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

SmiQuadratic.hpp

4.12 SmiQuadraticDataDC Class Reference

Inheritance diagram for SmiQuadraticDataDC:

Collaboration diagram for SmiQuadraticDataDC:

4.12.1 Detailed Description

Definition at line 55 of file SmiQuadratic.hpp.

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

· SmiQuadratic.hpp

4.13 SmiScenarioTree < T > Class Template Reference

Inheritance diagram for SmiScenarioTree< T >:

Public Member Functions

Iterators

```
    std::vector < T >::iterator treeBegin ()
        begin
    std::vector < T >::iterator treeEnd ()
        end
    std::vector < T > & wholeTree ()
        whole tree
    std::vector < T >::iterator scenBegin (int s)
        scenario iterators TODO: native code for these iterators that does not depend on copying.
    std::vector < T >::iterator scenEnd (int s)
```

Query members

```
    SmiTreeNode< T > * getRoot ()
        Get root node.
    SmiTreeNode< T > * getLeaf (int scn)
        Get leaf node.
    SmiTreeNode< T > * find (unsigned int scenario, int stage)
        Get node identified by scenario/stage.
    int getNumScenarios ()
        get number of scenarios
```

```
    SmiTreeNode< T > * find (std::vector< int > &label)
```

Get node identified by longest match to array of labels.

SmiTreeNode< T > * find (int *label, unsigned int sz)

Get node identified by longest match to array of labels.

std::vector< T > & getScenario (int scenario)

Get vector of node data for given scenario.

Tree modification members

- int addPathtoLeaf (int brscenario, int stage, std::vector< T > &pathdata, unsigned int start=0)
 Add new path from branching node to leaf.
- $\bullet \ \ \text{void setChildLabels (SmiTreeNode} < T > *n, \ \text{std::vector} < \ \text{int} > \ \text{labels)} \\$

Set child labels.

• int addPathtoLeaf (std::vector< int > &labels, std::vector< T > &pathdata)

Add new path using labels to find branching node.

SmiTreeNode< T > * addNodesToTree (SmiTreeNode< T > *parent, int scenario, std::vector< T > &path-data, int start)

Constructors, destructors and major modifying methods

• SmiScenarioTree ()

Default Constructor creates an empty scenario tree.

virtual ∼SmiScenarioTree ()

Destructor.

4.13.1 Detailed Description

template < class T> class SmiScenarioTree < T>

Definition at line 196 of file SmiScenarioTree.hpp.

4.13.2 Member Function Documentation

```
4.13.2.1 template < class T > std::vector < T >::iterator SmiScenarioTree < T >::scenBegin ( int s ) [inline]
```

scenario iterators TODO: native code for these iterators that does not depend on copying.

Definition at line 221 of file SmiScenarioTree.hpp.

```
4.13.2.2 template < class T > SmiTreeNode < T > * SmiScenarioTree < T >::getRoot() [inline]
```

Get root node.

Definition at line 236 of file SmiScenarioTree.hpp.

```
4.13.2.3 template < class T > SmiTreeNode < T > * SmiScenarioTree < T >::getLeaf ( int scn ) [inline]
```

Get leaf node.

Definition at line 241 of file SmiScenarioTree.hpp.

4.13.2.4 template < class T> SmiTreeNode < T>* SmiScenarioTree < T>::find (unsigned int scenario, int stage)

Get node identified by scenario/stage.

Definition at line 246 of file SmiScenarioTree.hpp.

4.13.2.5 template < class T > int SmiScenarioTree < T >::addPathtoLeaf (int brscenario, int stage, std::vector < T > & pathdata, unsigned int start = 0) [inline]

Add new path from branching node to leaf.

The branching node is the one belonging to "brscenario" at depth "stage". Length of incoming "pathdata" vector is leaf->depth() - stage. Responsibility for memory management of SmiTreeNodeData elements is assigned to Smi← ScenarioTree. SmiTreeNodeData elements must be created with "new" operator.

Definition at line 322 of file SmiScenarioTree.hpp.

4.13.2.6 template < class T > int SmiScenarioTree < T >::addPathtoLeaf (std::vector < int > & labels, std::vector < T > & pathdata) [inline]

Add new path using labels to find branching node.

The length of the incoming path is leaf.depth(). Responsibility for memory management of SmiTreeNodeData elements is assigned to SmiScenarioTree. SmiTreeNodeData elements must be created with "new" operator.

Definition at line 353 of file SmiScenarioTree.hpp.

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

SmiScenarioTree.hpp

4.14 SmiScnModel Class Reference

SmiScnModel: COIN-SMI Scenario Model Class.

#include <SmiScnModel.hpp>

Public Member Functions

Read SMPS files.

There should be three files: {name}.

[core, time, stoch]. If you have different extension conventions, then you can hack the method yourself. The files can be compressed. The object that reads the files is derived from **CoinMpsIO**.

The optional argument SmiCoreCombineRule allows user to pass in a class to override the default methods to combine core and stochastic data.

int readSmps (const char *name, SmiCoreCombineRule *r=NULL)

Writes SMPS files.

This method generates three files {name}.

[core, time, stoch] or {name}.[cor, tim, sto] (see second parameter).

Parameters

name	The name for the model and the written files
winFile←	optional; false by default, so extensions will be [core, time, stoch]. With this parameter set to
Extensions	true, it will be [cor, tim, sto].
strictFormat	optional, true by default. Set to false if SMPS files should be written in free format.

Returns

- -1 in case of no existing SMI model, otherwise 0
- int writeSmps (const char *name, bool winFileExtensions=false, bool strictFormat=true)
- SmiCoreData * getCore ()

Direct methods.

Direct methods require the user to create instances of Core data and Scenario data.

Currently, the dimension of the core nodes determines the dimension of the scenario nodes, but this is something that could easily be changed.

- void processDiscreteDistributionIntoScenarios (SmiDiscreteDistribution *s, bool test=false)
 generate scenarios from discrete distribution
- void setModelProb (double p)
- int addNodeToSubmodel (SmiScnNode *smiScnNode)
- SmiScenarioIndex generateScenario (SmiCoreData *core, CoinPackedMatrix *matrix, CoinPackedVector *dclo, CoinPackedVector *dclo, CoinPackedVector *dclo, CoinPackedVector *drlo, CoinPackedVector *drup, SmiStageIndex branch, SmiScenarioIndex anc, double prob, SmiCoreCombineRule *r=SmiCore CombineReplace::Instance())
 - generate scenario with ancestor/branch node identification
- SmiScenarioIndex generateScenario (SmiCoreData *core, CoinPackedMatrix *matrix, CoinPackedVector *dclo, CoinPackedVector *dclo, CoinPackedVector *dclo, CoinPackedVector *drlo, CoinPackedVector *drup, std::vector < int >labels, double prob, SmiCoreCombineRule *r=SmiCoreCombineReplace::Instance()) generate scenario with labels information
- - generate scenario with ancestor/branch node identification
- SmiScenarioIndex **generateScenarioFromCore** (SmiCoreData *core, double prob, SmiCoreCombineRule *r=SmiCoreCombineReplace::Instance())
- SmiScenarioIndex generateScenarioFromCore (double prob, SmiCoreCombineRule *r=SmiCoreCombine ← Replace::Instance())

loadOsiSolverData

Loads deterministic equivalent model into internal osi data structures and return handle.

Note: this uses a callback class SmiCoreCombineRule to decide how to combine the core and stochastic data. The user can override the default methods when the scenario is generated (see SmiScnModel::generateScenario) or when SMPS files are processed (see SmiScnModel::readSmps).

OsiSolverInterface * loadOsiSolverData ()

- OsiSolverInterface * loadOsiSolverDataForSubproblem (int stage, int scenStart)
- std::vector< std::pair< double, double >> solveWS (OsiSolverInterface *osiSolver, double objSense)
- std::pair< double, double * > solveEV (OsiSolverInterface *osiSolver, double objSense)
- double solveEEV (OsiSolverInterface *osiSolver, double objSense)
- double getWSValue (OsiSolverInterface *osiSolver, double objSense)
- double getEVValue (OsiSolverInterface *osiSolver, double objSense)
- double getEEVValue (OsiSolverInterface *osiSolver, double objSense)
- void setCore (SmiCoreData *val)
- SmiScenarioIndex getNumScenarios ()
- double getScenarioProb (SmiScenarioIndex ns)
- SmiScnNode * getLeafNode (SmiScenarioIndex i)
- SmiScnNode * getRootNode ()
- int * getIntegerInd ()
- int getIntegerLen ()
- int * getBinaryInd ()
- int getBinaryLen ()
- std::vector< int > getIntIndices ()
- void addIntIndice (int indice)
- double getObjectiveValue (SmiScenarioIndex ns)
- double * getColSolution (SmiScenarioIndex ns, int *length)
- double * getRowSolution (SmiScenarioIndex ns, int *length)
- double * getRowDuals (SmiScenarioIndex ns, int *length)
- double getColSolution (SmiScenarioIndex ns, int stage, int colIndex)
- double **getRowSolution** (SmiScenarioIndex ns, int stage, int rowIndex)
- double getRowDuals (SmiScenarioIndex ns, int stage, int rowIndex)
- double * getColValue (const double *d, SmiScenarioIndex ns, int *length)
- double getColValue (const double *d, SmiScenarioIndex ns, int stage, int rowIndex)
- double * getRowValue (const double *d, SmiScenarioIndex ns, int *length, bool isDual)
- double getRowValue (const double *d, SmiScenarioIndex ns, int stage, int rowIndex, bool isDual)
- void setOsiSolverHandle (OsiSolverInterface &osi)
- void setOsiSolverHandle (OsiSolverInterface *osi)
- OsiSolverInterface * getOsiSolverInterface ()
- void releaseSolver ()
- void releaseCore ()
- void setQuadraticSolver (ClpModel *clp)
- ClpModel * getQuadraticSolver ()
- ClpModel * loadQuadraticSolverData ()
- · SmiScnModel ()
- ∼SmiScnModel ()
- void addNode (SmiScnNode *node, bool notDetEq=false)
- void deleteNode (SmiScnNode *tnode)
- void addNode (SmiNodeData *node)
- SmiScenarioTree < SmiScnNode * > * getSmiTree ()

4.14.1 Detailed Description

SmiScnModel: COIN-SMI Scenario Model Class.

Concrete class for generating scenario stochastic linear programs.

This class implements the Scenarios format of the Stochastic MPS modeling system (TODO: web pointer). Core data and Scenarios data can be passed using COIN/OSI structures, or can be read from SMPS formatted files.

Typical driver fragment looks like this

```
SmiScnModel smi;
smi.readSmps("app0110R");
smi.setOsiSolverHandle(new OsiClpSolverInterface());
OsiSolverInterface *osiStoch = smi.loadOsiSolverData();
osiStoch->initialSolve();
```

The setOsiSolverHandle method allows the user to pass in any OSI compatible solver.

Definition at line 49 of file SmiScnModel.hpp.

4.14.2 Member Function Documentation

4.14.2.1 SmiScenarioIndex SmiScnModel::generateScenario (SmiCoreData * core, CoinPackedMatrix * matrix, CoinPackedVector * dclo, CoinPackedVector * dcup, CoinPackedVector * dobj, CoinPackedVector * drlo, CoinPackedVector * drup, SmiStageIndex branch, SmiScenarioIndex anc, double prob, SmiCoreCombineRule * r = SmiCoreCombineReplace::Instance())

generate scenario with ancestor/branch node identification

Core argument must be supplied. Data values combine with corresponding core values, if found, or creates them if not.

Scenario nodes need to have same dimensions as core nodes.

Data field arguments can be NULL, or empty.

branch, anc, arguments must be supplied. These identify the branching node according to the Stochastic MPS standard. prob is unconditional probability of scenario

```
4.14.2.2 SmiScenarioIndex SmiScnModel::generateScenario ( SmiCoreData * core, CoinPackedMatrix * matrix, CoinPackedVector * dclo, CoinPackedVector * dcup, CoinPackedVector * dobj, CoinPackedVector * drlo, CoinPackedVector * drup, std::vector < int > labels, double prob, SmiCoreCombineRule * r = SmiCoreCombineReplace::Instance())
```

generate scenario with labels information

 $Core\ argument\ must\ be\ supplied.\ Data\ values\ combine\ with\ corresponding\ core\ values,\ if\ found,\ or\ creates\ them\ if\ not.$

Scenario nodes need to have same dimensions as core nodes.

Data field arguments can be NULL, or empty.

Labels are passed as vector<int> array. Adds new path using labels to find branching node. The depth (root to leaf) of new path is labels.size().

4.14.2.3 SmiScenarioIndex SmiScnModel::generateScenario (CoinPackedMatrix * matrix, CoinPackedVector * dclo, CoinPackedVector * dclo, CoinPackedVector * drlo, CoinPackedVector * drlo, CoinPackedVector * drup, SmiStageIndex branch, SmiScenarioIndex anc, double prob, SmiCoreCombineRule * r = SmiCoreCombineReplace::Instance())

generate scenario with ancestor/branch node identification

Core argument must be supplied. Data values combine with corresponding core values, if found, or creates them if not. Scenario nodes need to have same dimensions as core nodes.

Data field arguments can be NULL, or empty.

branch, anc, arguments must be supplied. These identify the branching node according to the Stochastic MPS standard. prob is unconditional probability of scenario

4.14.2.4 SmiScenarioIndex SmiScnModel::generateScenario (CoinPackedMatrix * matrix, CoinPackedVector * dclo, CoinPackedVector * dclo, CoinPackedVector * dclo, CoinPackedVector * drlo, CoinPackedVector * drup, std::vector < int > labels, double prob, SmiCoreCombineRule * r = SmiCoreCombineReplace::Instance())

generate scenario with labels information

Core argument must be supplied. Data values combine with corresponding core values, if found, or creates them if not. Scenario nodes need to have same dimensions as core nodes.

Data field arguments can be NULL, or empty.

Labels are passed as vector<int> array. Adds new path using labels to find branching node. The depth (root to leaf) of new path is labels.size().

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

• SmiScnModel.hpp

4.15 SmiScnModelAddNode Class Reference

4.15.1 Detailed Description

Definition at line 440 of file SmiScnModel.hpp.

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

SmiScnModel.hpp

4.16 SmiScnModelDeleteNode Class Reference

4.16.1 Detailed Description

Definition at line 455 of file SmiScnModel.hpp.

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

SmiScnModel.hpp

4.17 SmiScnNode Class Reference

Friends

· class SmiScnModel

4.17.1 Detailed Description

Definition at line 360 of file SmiScnModel.hpp.

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

· SmiScnModel.hpp

4.18 SmiSmpsCardReader Class Reference

Inheritance diagram for SmiSmpsCardReader:

Collaboration diagram for SmiSmpsCardReader:

Public Member Functions

• SmiSmpsCardReader (CoinFileInput *input, CoinMpsIO *reader)

Constructor expects file to be open This one takes gzFile if fp null.

4.18.1 Detailed Description

Definition at line 41 of file SmiSmpsIO.hpp.

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

· SmiSmpsIO.hpp

4.19 SmiSmpsIO Class Reference

Inheritance diagram for SmiSmpsIO:

Collaboration diagram for SmiSmpsIO:

4.19.1 Detailed Description

Definition at line 76 of file SmiSmpsIO.hpp.

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

SmiSmpsIO.hpp

4.20 SmiTreeNode < T > Class Template Reference

Scenario Tree.

#include <SmiScenarioTree.hpp>
Inheritance diagram for SmiTreeNode< T >:

Public Member Functions

Constructors, destructors and major modifying methods

• SmiTreeNode ()

Default Constructor creates an empty node.

• SmiTreeNode (T p)

Constructor from P.

∼SmiTreeNode ()

Destructor.

4.20.1 Detailed Description

template < class T > class SmiTreeNode < T >

Scenario Tree.

This class is used for storing and accessing scenario trees. SmiTreeNode template class.

Manages pointers to parent, child and sibling for tree navigation. Template class instance is a pointer to an object that must be created with "new" operator.

Definition at line 27 of file SmiScenarioTree.hpp.

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

SmiScenarioTree.hpp

File Documentation

Index

addPathtoLeat SmiScenarioTree, 23
find
SmiScenarioTree, 22
generateScenario
SmiScnModel, 26, 27
getLeaf
SmiScenarioTree, 22
getRoot
SmiScenarioTree, 22
scenBegin
SmiScenarioTree, 22
SmiCoreCombineAdd, 17
SmiCoreCombineReplace, 17
SmiCoreCombineRule, 17
SmiCoreData, 18
SmiDiscreteDistribution, 18
SmiDiscreteEvent, 19
SmiDiscreteRV, 19
SmiLinearData, 19
SmiMessage, 20
SmiNodeData, 20
SmiQuadraticData, 20
SmiQuadraticDataDC, 21
SmiScenarioTree
addPathtoLeaf, 23
find, 22
getLeaf, 22
getRoot, 22
scenBegin, 22
SmiScenarioTree< T >, 21
SmiScnModel, 23
generateScenario, 26, 27
SmiScnModelAddNode, 27 SmiScnModelDeleteNode, 27
SmiSchNode, 28
SmiSmpsCardReader, 28
SmiSmpsIO, 28
CCpo.C.,

SmiTreeNode< T>, 29