

Ensemble Clustering

Generated by Doxygen 1.8.2

Thu Dec 13 2012 16:53:36

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	EnsembleClustering Namespace Reference	9
5.1.1	Typedef Documentation	10
5.1.1.1	cluster	10
5.1.1.2	Cluster	10
5.1.1.3	Clustering	10
5.1.1.4	Edge	10
5.1.1.5	edge	10
5.1.1.6	Node	10
5.1.1.7	node	11
5.1.2	Function Documentation	11
5.1.2.1	TEST_F	11
5.1.2.2	TEST_F	11
5.1.2.3	TEST_F	12
6	Class Documentation	13
6.1	EnsembleClustering::ClusterContracter Class Reference	13
6.1.1	Detailed Description	14
6.1.2	Constructor & Destructor Documentation	14
6.1.2.1	ClusterContracter	14
6.1.2.2	~ClusterContracter	14

6.2	EnsembleClustering::Clusterer Class Reference	14
6.2.1	Detailed Description	15
6.2.2	Constructor & Destructor Documentation	15
6.2.2.1	Clusterer	15
6.2.2.2	~Clusterer	15
6.2.3	Member Function Documentation	15
6.2.3.1	run	15
6.3	EnsembleClustering::Clustering Class Reference	15
6.3.1	Detailed Description	16
6.3.2	Constructor & Destructor Documentation	17
6.3.2.1	Clustering	17
6.3.2.2	~Clustering	17
6.3.3	Member Function Documentation	17
6.3.3.1	addToCluster	17
6.3.3.2	clusterOf	17
6.3.3.3	firstCluster	17
6.3.3.4	isProper	17
6.3.3.5	lastCluster	17
6.3.3.6	mergeClusters	18
6.3.3.7	moveToCluster	18
6.3.3.8	operator[]	18
6.3.3.9	operator[]	18
6.3.3.10	toSingleton	18
6.3.4	Member Data Documentation	18
6.3.4.1	nextCluster	18
6.4	EnsembleClustering::ClusteringGenerator Class Reference	19
6.4.1	Detailed Description	19
6.4.2	Constructor & Destructor Documentation	19
6.4.2.1	ClusteringGenerator	19
6.4.2.2	~ClusteringGenerator	19
6.4.3	Member Function Documentation	19
6.4.3.1	makeOneClustering	19
6.4.3.2	makeSingletonClustering	20
6.5	EnsembleClustering::ClusteringTest Class Reference	21
6.5.1	Detailed Description	21
6.6	EnsembleClustering::Contracter Class Reference	21
6.6.1	Detailed Description	22
6.6.2	Constructor & Destructor Documentation	22
6.6.2.1	Contracter	22
6.6.2.2	~Contracter	22

6.6.3	Member Function Documentation	22
6.6.3.1	contract	22
6.7	EnsembleClustering::EdgeScoring Class Reference	23
6.7.1	Detailed Description	23
6.7.2	Constructor & Destructor Documentation	23
6.7.2.1	EdgeScoring	23
6.7.2.2	~EdgeScoring	23
6.7.3	Member Function Documentation	23
6.7.3.1	scoreEdge	23
6.8	EnsembleClustering::EnsembleClusterer Class Reference	24
6.8.1	Detailed Description	24
6.8.2	Constructor & Destructor Documentation	24
6.8.2.1	EnsembleClusterer	24
6.8.2.2	~EnsembleClusterer	24
6.9	EnsembleClustering::Graph Class Reference	24
6.9.1	Detailed Description	25
6.9.2	Constructor & Destructor Documentation	25
6.9.2.1	Graph	25
6.9.2.2	Graph	26
6.9.2.3	~Graph	26
6.9.3	Member Function Documentation	26
6.9.3.1	asSTINGER	26
6.9.3.2	degree	26
6.9.3.3	firstNode	26
6.9.3.4	forallEdges	26
6.9.3.5	hasEdge	26
6.9.3.6	insertEdge	26
6.9.3.7	lastNode	27
6.9.3.8	numberOfEdges	27
6.9.3.9	numberOfNodes	27
6.9.3.10	totalEdgeWeight	27
6.9.3.11	weight	27
6.9.3.12	weight	28
6.9.3.13	weight	28
6.9.4	Member Data Documentation	28
6.9.4.1	defaultEdgeType	28
6.9.4.2	defaultEdgeWeight	28
6.9.4.3	defaultTimeStamp	28
6.9.4.4	stingerG	28
6.10	EnsembleClustering::GraphGenerator Class Reference	28

6.10.1 Detailed Description	29
6.10.2 Constructor & Destructor Documentation	29
6.10.2.1 GraphGenerator	29
6.10.2.2 ~GraphGenerator	29
6.10.3 Member Function Documentation	29
6.10.3.1 makeCircularGraph	29
6.10.3.2 makeCompleteGraph	29
6.10.3.3 makeErdosRenyiGraph	30
6.11 EnsembleClustering::GraphGTest Class Reference	30
6.11.1 Detailed Description	31
6.11.2 Member Function Documentation	31
6.11.2.1 SetUp	32
6.11.2.2 TearDown	32
6.11.3 Member Data Documentation	32
6.11.3.1 gen	32
6.12 GTestTest Class Reference	32
6.12.1 Detailed Description	33
6.12.2 Member Function Documentation	33
6.12.2.1 SetUp	33
6.13 EnsembleClustering::IndexMap< I, T > Class Template Reference	33
6.13.1 Detailed Description	34
6.13.2 Constructor & Destructor Documentation	34
6.13.2.1 IndexMap	34
6.13.2.2 IndexMap	35
6.13.2.3 ~IndexMap	35
6.13.3 Member Function Documentation	35
6.13.3.1 operator[]	35
6.13.3.2 operator[]	35
6.13.4 Member Data Documentation	35
6.13.4.1 array	35
6.13.4.2 defaultValue	35
6.13.4.3 n	36
6.14 EnsembleClustering::InputGTest Class Reference	36
6.14.1 Detailed Description	36
6.15 EnsembleClustering::LabelPropagation Class Reference	37
6.15.1 Detailed Description	37
6.15.2 Constructor & Destructor Documentation	38
6.15.2.1 LabelPropagation	38
6.15.2.2 ~LabelPropagation	38
6.15.3 Member Function Documentation	38

6.15.3.1	run	38
6.16	EnsembleClustering::Matcher Class Reference	39
6.16.1	Detailed Description	39
6.16.2	Constructor & Destructor Documentation	39
6.16.2.1	Matcher	39
6.16.2.2	~Matcher	39
6.16.3	Member Function Documentation	39
6.16.3.1	run	39
6.17	EnsembleClustering::Matching Class Reference	40
6.17.1	Detailed Description	41
6.17.2	Constructor & Destructor Documentation	41
6.17.2.1	Matching	41
6.17.2.2	~Matching	41
6.17.3	Member Function Documentation	41
6.17.3.1	areMatched	41
6.17.3.2	clone	41
6.17.3.3	dispose	41
6.17.3.4	isMatched	42
6.17.3.5	isProper	42
6.17.3.6	match	42
6.17.3.7	operator=	42
6.17.3.8	unmatch	43
6.18	EnsembleClustering::MatchingContracter Class Reference	43
6.18.1	Detailed Description	43
6.18.2	Constructor & Destructor Documentation	43
6.18.2.1	MatchingContracter	43
6.18.2.2	~MatchingContracter	43
6.19	EnsembleClustering::METISParser Class Reference	44
6.19.1	Detailed Description	44
6.19.2	Constructor & Destructor Documentation	44
6.19.2.1	METISParser	44
6.19.2.2	~METISParser	44
6.19.3	Member Function Documentation	44
6.19.3.1	close	44
6.19.3.2	getHeader	45
6.19.3.3	getNext	45
6.19.3.4	hasNext	45
6.19.3.5	open	45
6.19.4	Member Data Documentation	45
6.19.4.1	graphFile	45

6.19.4.2	graphPath	45
6.19.4.3	line	45
6.19.4.4	nodeCount	45
6.20	EnsembleClustering::METISToSTINGER Class Reference	45
6.20.1	Detailed Description	46
6.20.2	Constructor & Destructor Documentation	46
6.20.2.1	METISToSTINGER	46
6.20.2.2	~METISToSTINGER	46
6.20.3	Member Function Documentation	46
6.20.3.1	read	46
6.21	EnsembleClustering::Modularity Class Reference	47
6.21.1	Detailed Description	49
6.21.2	Constructor & Destructor Documentation	49
6.21.2.1	Modularity	49
6.21.2.2	~Modularity	49
6.21.3	Member Function Documentation	49
6.21.3.1	getQuality	49
6.21.3.2	precompute	50
6.21.4	Member Data Documentation	50
6.21.4.1	incidentWeight	51
6.22	EnsembleClustering::ModularityScoring Class Reference	51
6.22.1	Detailed Description	52
6.22.2	Constructor & Destructor Documentation	52
6.22.2.1	ModularityScoring	52
6.22.2.2	~ModularityScoring	52
6.22.3	Member Function Documentation	52
6.22.3.1	cutweight	52
6.22.3.2	deltaMod	52
6.22.3.3	mod	52
6.22.3.4	scoreEdge	52
6.22.3.5	weight	52
6.23	EnsembleClustering::NodeMap< T > Class Template Reference	53
6.23.1	Detailed Description	53
6.23.2	Constructor & Destructor Documentation	54
6.23.2.1	NodeMap	54
6.23.2.2	NodeMap	54
6.23.2.3	~NodeMap	54
6.23.3	Member Function Documentation	54
6.23.3.1	operator[]	54
6.23.3.2	operator[]	54

6.23.4	Member Data Documentation	54
6.23.4.1	array	54
6.23.4.2	defaultValue	55
6.23.4.3	n	55
6.24	Noise Class Reference	55
6.24.1	Detailed Description	55
6.24.2	Constructor & Destructor Documentation	55
6.24.2.1	Noise	55
6.24.2.2	~Noise	56
6.24.3	Member Function Documentation	56
6.24.3.1	add	56
6.24.4	Member Data Documentation	56
6.24.4.1	lowerBound	56
6.24.4.2	randomEngine	56
6.24.4.3	uniform	56
6.24.4.4	upperBound	56
6.25	EnsembleClustering::Overlapper Class Reference	56
6.25.1	Detailed Description	57
6.25.2	Constructor & Destructor Documentation	57
6.25.2.1	Overlapper	57
6.25.2.2	~Overlapper	57
6.26	EnsembleClustering::ParallelMatcher Class Reference	57
6.26.1	Detailed Description	58
6.26.2	Constructor & Destructor Documentation	58
6.26.2.1	ParallelMatcher	58
6.26.2.2	~ParallelMatcher	59
6.26.3	Member Function Documentation	59
6.26.3.1	run	59
6.27	EnsembleClustering::QualityMeasure Class Reference	59
6.27.1	Detailed Description	60
6.27.2	Constructor & Destructor Documentation	61
6.27.2.1	QualityMeasure	61
6.27.2.2	~QualityMeasure	61
6.27.3	Member Function Documentation	61
6.27.3.1	getQuality	61
6.27.4	Member Data Documentation	61
6.27.4.1	G	61
6.28	RandomProbability Class Reference	61
6.28.1	Detailed Description	61
6.28.2	Constructor & Destructor Documentation	62

6.28.2.1	RandomProbability	62
6.28.2.2	~RandomProbability	62
6.28.3	Member Function Documentation	62
6.28.3.1	generate	62
6.28.4	Member Data Documentation	62
6.28.4.1	randomEngine	62
6.28.4.2	uniform	62
6.29	EnsembleClustering::RegionGrowingOverlapper Class Reference	62
6.29.1	Detailed Description	63
6.29.2	Constructor & Destructor Documentation	63
6.29.2.1	RegionGrowingOverlapper	63
6.29.2.2	~RegionGrowingOverlapper	63
6.30	EnsembleClustering::ScoreMatchContract Class Reference	63
6.30.1	Detailed Description	64
6.30.2	Constructor & Destructor Documentation	64
6.30.2.1	ScoreMatchContract	64
6.30.2.2	~ScoreMatchContract	65
6.31	EnsembleClustering::STINGERFromAdjacencies Class Reference	65
6.31.1	Detailed Description	66
6.31.2	Constructor & Destructor Documentation	66
6.31.2.1	STINGERFromAdjacencies	66
6.31.2.2	~STINGERFromAdjacencies	66
6.31.3	Member Function Documentation	66
6.31.3.1	addAdjacencies	66
6.31.3.2	createGraph	66
6.31.3.3	getGraph	66
6.31.3.4	getSTINGER	66
6.31.4	Member Data Documentation	67
6.31.4.1	currentNode	67
6.31.4.2	G	67
6.32	Timer Class Reference	67
6.32.1	Detailed Description	67
6.32.2	Constructor & Destructor Documentation	67
6.32.2.1	Timer	67
6.32.2.2	~Timer	68
7	File Documentation	69
7.1	src/aux/IndexMap.h File Reference	69
7.2	src/aux/log.h File Reference	70
7.2.1	Macro Definition Documentation	70

7.2.1.1	DEBUG	70
7.2.1.2	ERROR	70
7.2.1.3	FATAL	71
7.2.1.4	INFO	71
7.2.1.5	LOCATION	71
7.2.1.6	LOGGER	71
7.2.1.7	TRACE	71
7.2.1.8	WARN	71
7.3	src/aux/Noise.cpp File Reference	71
7.4	src/aux/Noise.h File Reference	72
7.5	src/aux/RandomProbability.cpp File Reference	72
7.6	src/aux/RandomProbability.h File Reference	73
7.7	src/aux/Timer.cpp File Reference	74
7.8	src/aux/Timer.h File Reference	75
7.9	src/clustering/Clusterer.cpp File Reference	76
7.10	src/clustering/Clusterer.h File Reference	76
7.11	src/clustering/Clustering.cpp File Reference	78
7.12	src/clustering/Clustering.h File Reference	78
7.13	src/clustering/ClusteringGenerator.cpp File Reference	80
7.14	src/clustering/ClusteringGenerator.h File Reference	80
7.15	src/clustering/LabelPropagation.cpp File Reference	82
7.16	src/clustering/LabelPropagation.h File Reference	82
7.17	src/clustering/Modularity.cpp File Reference	84
7.18	src/clustering/Modularity.h File Reference	84
7.19	src/clustering/QualityMeasure.cpp File Reference	86
7.20	src/clustering/QualityMeasure.h File Reference	86
7.21	src/clustering/ScoreMatchContract.cpp File Reference	88
7.22	src/clustering/ScoreMatchContract.h File Reference	89
7.23	src/clustering/test/ClusteringTest.cpp File Reference	90
7.24	src/clustering/test/ClusteringTest.h File Reference	91
7.25	src/coarsening/ClusterContracter.cpp File Reference	92
7.26	src/coarsening/ClusterContracter.h File Reference	92
7.27	src/coarsening/Contracter.cpp File Reference	93
7.28	src/coarsening/Contracter.h File Reference	94
7.29	src/coarsening/MatchingContracter.cpp File Reference	95
7.30	src/coarsening/MatchingContracter.h File Reference	96
7.31	src/ensemble/EnsembleClusterer.cpp File Reference	96
7.32	src/ensemble/EnsembleClusterer.h File Reference	97
7.33	src/EnsembleClustering.cpp File Reference	97
7.33.1	Function Documentation	98

7.33.1.1	configureLogging	98
7.33.1.2	main	98
7.33.1.3	makeCompleteGraph	98
7.33.1.4	testMatching	99
7.33.1.5	testMETISToSTINGER	99
7.34	src/graph/Graph.cpp File Reference	100
7.35	src/graph/Graph.h File Reference	101
7.35.1	Macro Definition Documentation	102
7.35.1.1	EDGE_DEST	102
7.35.1.2	EDGE_SOURCE	102
7.35.1.3	FORALL_EDGES_BEGIN	102
7.35.1.4	FORALL_EDGES_END	102
7.35.1.5	FORALL_EDGES_OF_NODE_BEGIN	102
7.35.1.6	FORALL_EDGES_OF_NODE_END	102
7.35.1.7	PARALLEL_FORALL_EDGES_BEGIN	102
7.35.1.8	PARALLEL_FORALL_EDGES_END	103
7.35.1.9	READ_ONLY_FORALL_EDGES_BEGIN	103
7.35.1.10	READ_ONLY_FORALL_EDGES_END	103
7.35.1.11	READ_ONLY_FORALL_EDGES_OF_NODE_BEGIN	103
7.35.1.12	READ_ONLY_FORALL_EDGES_OF_NODE_END	103
7.35.1.13	READ_ONLY_PARALLEL_FORALL_EDGES_BEGIN	103
7.35.1.14	READ_ONLY_PARALLEL_FORALL_EDGES_END	103
7.36	src/graph/GraphGenerator.cpp File Reference	104
7.37	src/graph/GraphGenerator.h File Reference	104
7.38	src/graph/NodeMap.h File Reference	105
7.39	src/graph/test/GraphGTest.cpp File Reference	106
7.40	src/graph/test/GraphGTest.h File Reference	107
7.41	src/input/METISParser.cpp File Reference	108
7.42	src/input/METISParser.h File Reference	109
7.43	src/input/METISToSTINGER.cpp File Reference	109
7.44	src/input/METISToSTINGER.h File Reference	110
7.45	src/input/STINGERFromAdjacencies.cpp File Reference	111
7.46	src/input/STINGERFromAdjacencies.h File Reference	112
7.47	src/input/test/InputGTest.cpp File Reference	113
7.48	src/input/test/InputGTest.h File Reference	113
7.49	src/matching/Matcher.cpp File Reference	114
7.50	src/matching/Matcher.h File Reference	115
7.51	src/matching/Matching.cpp File Reference	117
7.52	src/matching/Matching.h File Reference	117
7.53	src/matching/ParallelMatcher.cpp File Reference	119

7.54	src/matching/ParallelMatcher.h File Reference	119
7.55	src/overlap/Overlapper.cpp File Reference	121
7.56	src/overlap/Overlapper.h File Reference	121
7.57	src/overlap/RegionGrowingOverlapper.cpp File Reference	123
7.58	src/overlap/RegionGrowingOverlapper.h File Reference	124
7.59	src/scoring/EdgeScoring.cpp File Reference	125
7.60	src/scoring/EdgeScoring.h File Reference	126
7.61	src/scoring/ModularityScoring.cpp File Reference	127
7.62	src/scoring/ModularityScoring.h File Reference	127
7.63	src/test/TestGTest.h File Reference	128
7.63.1	Function Documentation	129
7.63.1.1	TEST_F	129

Index**129**

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

EnsembleClustering	9
------------------------------------	---

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

EnsembleClustering::Clusterer	14
EnsembleClustering::LabelPropagation	37
EnsembleClustering::ScoreMatchContract	63
EnsembleClustering::ClusteringGenerator	19
EnsembleClustering::Contracter	21
EnsembleClustering::ClusterContracter	13
EnsembleClustering::EdgeScoring	23
EnsembleClustering::ModularityScoring	51
EnsembleClustering::EnsembleClusterer	24
EnsembleClustering::Graph	24
EnsembleClustering::GraphGenerator	28
EnsembleClustering::IndexMap< I, T >	33
EnsembleClustering::Matcher	39
EnsembleClustering::ParallelMatcher	57
EnsembleClustering::MatchingContracter	43
EnsembleClustering::METISParser	44
EnsembleClustering::METIStoSTINGER	45
EnsembleClustering::NodeMap< T >	53
EnsembleClustering::NodeMap< cluster >	53
EnsembleClustering::Clustering	15
EnsembleClustering::NodeMap< double >	53
EnsembleClustering::NodeMap< node >	53
EnsembleClustering::Matching	40
Noise	55
EnsembleClustering::Overlapper	56
EnsembleClustering::RegionGrowingOverlapper	62
EnsembleClustering::QualityMeasure	59
EnsembleClustering::Modularity	47
RandomProbability	61
EnsembleClustering::STINGERFromAdjacencies	65
Test	
EnsembleClustering::ClusteringTest	21
EnsembleClustering::GraphGTest	30
EnsembleClustering::InputGTest	36
GTestTest	32
Timer	67

Chapter 3

Class Index

3.1 Class List

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

EnsembleClustering::ClusterContracter	13
EnsembleClustering::Clusterer	14
EnsembleClustering::Clustering	15
EnsembleClustering::ClusteringGenerator	19
EnsembleClustering::ClusteringTest	21
EnsembleClustering::Contracter	21
EnsembleClustering::EdgeScoring	23
EnsembleClustering::EnsembleClusterer	24
EnsembleClustering::Graph	
Graph interface	24
EnsembleClustering::GraphGenerator	28
EnsembleClustering::GraphGTest	30
GTestTest	32
EnsembleClustering::IndexMap< I, T >	
An IndexMap implements a 1-based mapping from an integer index type to an arbitray value type	33
EnsembleClustering::InputGTest	36
EnsembleClustering::LabelPropagation	
As described in Ovelgoenne et al: An Ensemble Learning Strategy for Graph Clustering Ragha- van et al	37
EnsembleClustering::Matcher	39
EnsembleClustering::Matching	40
EnsembleClustering::MatchingContracter	43
EnsembleClustering::METISParser	44
EnsembleClustering::METIStoSTINGER	
This class provides a user interface for reading a METIS graph file and returning a STINGER- based graph object	45
EnsembleClustering::Modularity	47
EnsembleClustering::ModularityScoring	51
EnsembleClustering::NodeMap< T >	53
Noise	
Noise is random addition to a signal	55
EnsembleClustering::Overlapper	56
EnsembleClustering::ParallelMatcher	57
EnsembleClustering::QualityMeasure	
Abstract base class for all clustering quality measures	59
RandomProbability	61
EnsembleClustering::RegionGrowingOverlapper	62
EnsembleClustering::ScoreMatchContract	63

[EnsembleClustering::STINGERFromAdjacencies](#)

A 'builder' which constructs a STINGER-based graph from adjacencies [65](#)

[Timer](#)

TODO: Platform-agnostic timer class [67](#)

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

src/EnsembleClustering.cpp	97
src/aux/IndexMap.h	69
src/aux/log.h	70
src/aux/Noise.cpp	71
src/aux/Noise.h	72
src/aux/RandomProbability.cpp	72
src/aux/RandomProbability.h	73
src/aux/Timer.cpp	74
src/aux/Timer.h	75
src/clustering/Clusterer.cpp	76
src/clustering/Clusterer.h	76
src/clustering/Clustering.cpp	78
src/clustering/Clustering.h	78
src/clustering/ClusteringGenerator.cpp	80
src/clustering/ClusteringGenerator.h	80
src/clustering/LabelPropagation.cpp	82
src/clustering/LabelPropagation.h	82
src/clustering/Modularity.cpp	84
src/clustering/Modularity.h	84
src/clustering/QualityMeasure.cpp	86
src/clustering/QualityMeasure.h	86
src/clustering/ScoreMatchContract.cpp	88
src/clustering/ScoreMatchContract.h	89
src/clustering/test/ClusteringTest.cpp	90
src/clustering/test/ClusteringTest.h	91
src/coarsening/ClusterContracter.cpp	92
src/coarsening/ClusterContracter.h	92
src/coarsening/Contracter.cpp	93
src/coarsening/Contracter.h	94
src/coarsening/MatchingContracter.cpp	95
src/coarsening/MatchingContracter.h	96
src/ensemble/EnsembleClusterer.cpp	96
src/ensemble/EnsembleClusterer.h	97
src/graph/Graph.cpp	100
src/graph/Graph.h	101
src/graph/GraphGenerator.cpp	104
src/graph/GraphGenerator.h	104
src/graph/NodeMap.h	105

src/graph/test/GraphGTest.cpp	106
src/graph/test/GraphGTest.h	107
src/input/METISParser.cpp	108
src/input/METISParser.h	109
src/input/METISToSTINGER.cpp	109
src/input/METISToSTINGER.h	110
src/input/STINGERFromAdjacencies.cpp	111
src/input/STINGERFromAdjacencies.h	112
src/input/test/InputGTest.cpp	113
src/input/test/InputGTest.h	113
src/matching/Matcher.cpp	114
src/matching/Matcher.h	115
src/matching/Matching.cpp	117
src/matching/Matching.h	117
src/matching/ParallelMatcher.cpp	119
src/matching/ParallelMatcher.h	119
src/overlap/Overlapper.cpp	121
src/overlap/Overlapper.h	121
src/overlap/RegionGrowingOverlapper.cpp	123
src/overlap/RegionGrowingOverlapper.h	124
src/scoring/EdgeScoring.cpp	125
src/scoring/EdgeScoring.h	126
src/scoring/ModularityScoring.cpp	127
src/scoring/ModularityScoring.h	127
src/test/TestGTest.h	128

Chapter 5

Namespace Documentation

5.1 EnsembleClustering Namespace Reference

Classes

- class [IndexMap](#)

An [IndexMap](#) implements a 1-based mapping from an integer index type to an arbitray value type.

- class [Clusterer](#)
- class [Clustering](#)
- class [ClusteringGenerator](#)
- class [LabelPropagation](#)

As described in Ovelgoenne et al: An Ensemble Learning Strategy for [Graph Clustering](#) Raghavan et al.

- class [Modularity](#)
- class [QualityMeasure](#)

Abstract base class for all clustering quality measures.

- class [ScoreMatchContract](#)
- class [ClusteringTest](#)
- class [ClusterContracter](#)
- class [Contracter](#)
- class [MatchingContracter](#)
- class [EnsembleClusterer](#)
- class [Graph](#)

[Graph](#) interface.

- class [GraphGenerator](#)
- class [NodeMap](#)
- class [GraphGTest](#)
- class [METISParser](#)
- class [METIStoSTINGER](#)

This class provides a user interface for reading a METIS graph file and returning a STINGER-based graph object.

- class [STINGERFromAdjacencies](#)

A 'builder' which constructs a STINGER-based graph from adjacencies.

- class [InputGTest](#)
- class [Matcher](#)
- class [Matching](#)
- class [ParallelMatcher](#)
- class [Overlapper](#)
- class [RegionGrowingOverlapper](#)
- class [EdgeScoring](#)
- class [ModularityScoring](#)

Typedefs

- typedef int64_t [cluster](#)
cluster is represented as a 1-based index
- typedef int64_t [node](#)
Typedefs.
- typedef std::pair< [node](#), [node](#) > [edge](#)
an undirected edge is a pair of nodes (indices)
- typedef int [Node](#)
- typedef int [Edge](#)
- typedef int [Clustering](#)
- typedef int [Cluster](#)

Functions

- [TEST_F](#) ([ClusteringTest](#), testModularity)
- [TEST_F](#) ([GraphGTest](#), testIteration)
- [TEST_F](#) ([InputGTest](#), testMETISParser)

5.1.1 Typedef Documentation

5.1.1.1 typedef int64_t EnsembleClustering::cluster

cluster is represented as a 1-based index

Definition at line 15 of file Clustering.h.

5.1.1.2 typedef int EnsembleClustering::Cluster

Definition at line 21 of file ModularityScoring.h.

5.1.1.3 typedef int EnsembleClustering::Clustering

Definition at line 20 of file ModularityScoring.h.

5.1.1.4 typedef int EnsembleClustering::Edge

Definition at line 17 of file ModularityScoring.h.

5.1.1.5 typedef std::pair<node, node> EnsembleClustering::edge

an undirected edge is a pair of nodes (indices)

Definition at line 26 of file Graph.h.

5.1.1.6 typedef int EnsembleClustering::Node

Definition at line 14 of file EdgeScoring.h.

5.1.1.7 typedef int64_t EnsembleClustering::node

Typedefs.

a node is an integer logical index. it is 1-based!

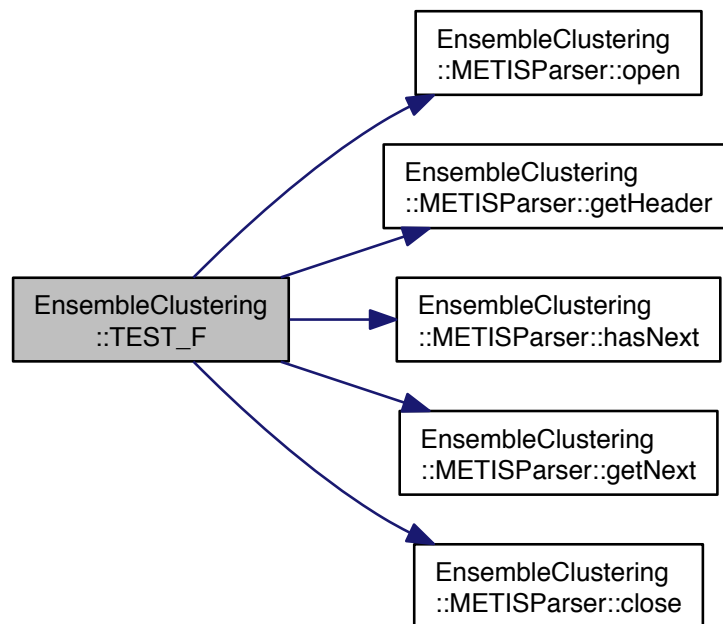
Definition at line 25 of file Graph.h.

5.1.2 Function Documentation

5.1.2.1 EnsembleClustering::TEST_F (InputGTest , testMETISParser)

Definition at line 22 of file InputGTest.h.

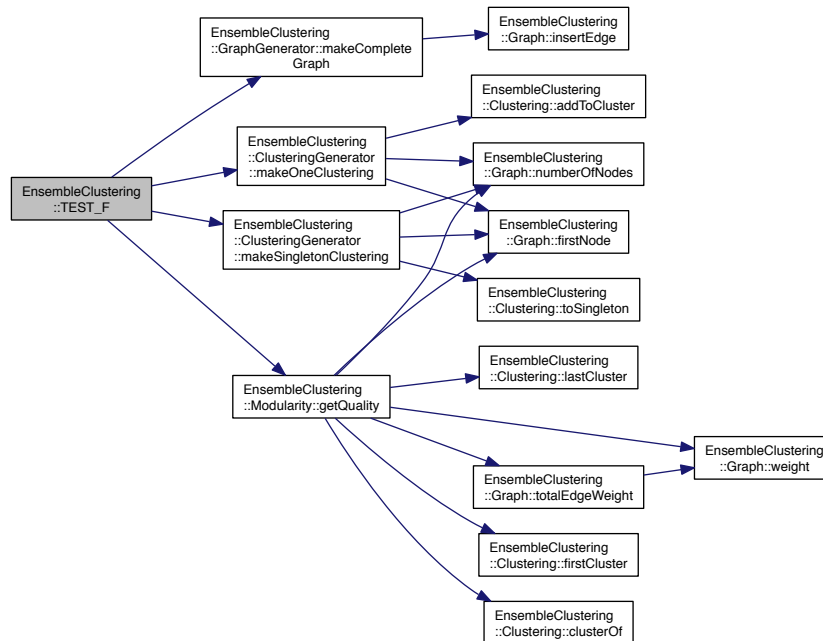
Here is the call graph for this function:



5.1.2.2 EnsembleClustering::TEST_F (ClusteringTest , testModularity)

Definition at line 29 of file ClusteringTest.h.

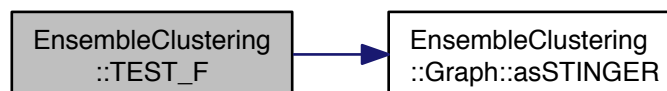
Here is the call graph for this function:



5.1.2.3 EnsembleClustering::TEST_F (GraphGTest , testIteration)

Definition at line 34 of file GraphGTest.h.

Here is the call graph for this function:



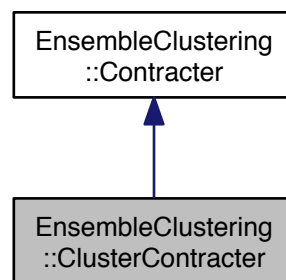
Chapter 6

Class Documentation

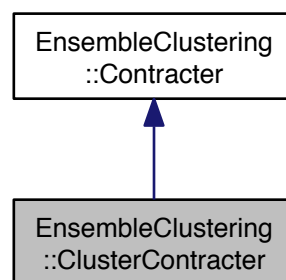
6.1 EnsembleClustering::ClusterContracter Class Reference

```
#include <ClusterContracter.h>
```

Inheritance diagram for EnsembleClustering::ClusterContracter:



Collaboration diagram for EnsembleClustering::ClusterContracter:



Public Member Functions

- [ClusterContracter](#) ()
- virtual [~ClusterContracter](#) ()

6.1.1 Detailed Description

Definition at line 15 of file ClusterContracter.h.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 EnsembleClustering::ClusterContracter::ClusterContracter ()

Definition at line 12 of file ClusterContracter.cpp.

6.1.2.2 EnsembleClustering::ClusterContracter::~~ClusterContracter () [virtual]

Definition at line 17 of file ClusterContracter.cpp.

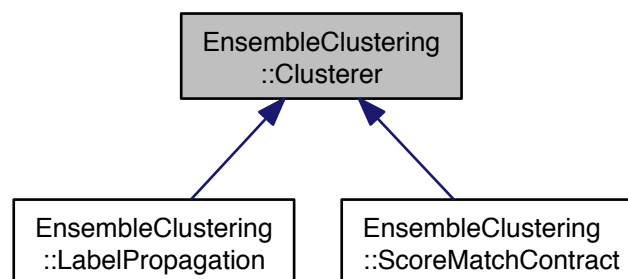
The documentation for this class was generated from the following files:

- src/coarsening/[ClusterContracter.h](#)
- src/coarsening/[ClusterContracter.cpp](#)

6.2 EnsembleClustering::Clusterer Class Reference

```
#include <Clusterer.h>
```

Inheritance diagram for EnsembleClustering::Clusterer:



Public Member Functions

- [Clusterer](#) ()
- virtual [~Clusterer](#) ()
- virtual [Clustering](#) & [run](#) ([Graph](#) &G)=0

6.2.1 Detailed Description

Definition at line 17 of file Clusterer.h.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 EnsembleClustering::Clusterer::Clusterer ()

Definition at line 12 of file Clusterer.cpp.

6.2.2.2 EnsembleClustering::Clusterer::~~Clusterer () [virtual]

Definition at line 17 of file Clusterer.cpp.

6.2.3 Member Function Documentation

6.2.3.1 virtual Clustering& EnsembleClustering::Clusterer::run (Graph & G) [pure virtual]

Implemented in [EnsembleClustering::LabelPropagation](#).

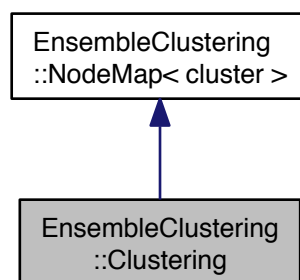
The documentation for this class was generated from the following files:

- [src/clustering/Clusterer.h](#)
- [src/clustering/Clusterer.cpp](#)

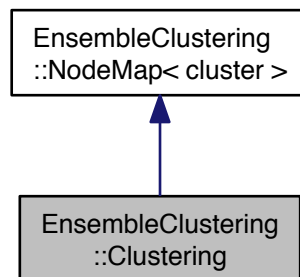
6.3 EnsembleClustering::Clustering Class Reference

```
#include <Clustering.h>
```

Inheritance diagram for EnsembleClustering::Clustering:



Collaboration diagram for EnsembleClustering::Clustering:



Public Member Functions

- [Clustering](#) (int64_t n)
Construct new clustering.
- virtual [~Clustering](#) ()
- [cluster & operator\[\]](#) (const [node](#) &u)
Index operator.
- const [cluster & operator\[\]](#) (const [node](#) &u) const
Index operator for const instances of this class.
- [cluster & clusterOf](#) ([node](#) u)
Return the cluster (id) in which a node is contained.
- void [addToCluster](#) ([cluster](#) c, [node](#) u)
Add a (previously unassigned) node to a cluster.
- void [moveToCluster](#) ([cluster](#) c, [node](#) u)
Move a (previously assigned) node to a cluster.
- void [toSingleton](#) ([node](#) u)
Creates a singleton cluster containing the node.
- void [mergeClusters](#) ([cluster](#) c, [cluster](#) d)
Assigns the nodes from both clusters to a new cluster.
- bool [isProper](#) (const [Graph](#) &G)
Check whether this clustering is a proper clustering of the graph, i.e.
- [cluster firstCluster](#) ()
Get the lowest cluster id;.
- [cluster lastCluster](#) ()
Get the highest cluster id that has been assigned.

Protected Attributes

- [cluster nextCluster](#)
next free cluster id for new cluster

6.3.1 Detailed Description

Definition at line 17 of file Clustering.h.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 EnsembleClustering::Clustering (int64_t *n*)

Construct new clustering.

Parameters

<i>in</i>	<i>n</i>	number of nodes
-----------	----------	-----------------

< first cluster index is 1

Definition at line 12 of file Clustering.cpp.

6.3.2.2 EnsembleClustering::Clustering::~~Clustering () [virtual]

Definition at line 16 of file Clustering.cpp.

6.3.3 Member Function Documentation

6.3.3.1 void EnsembleClustering::Clustering::addToCluster (cluster *c*, node *u*)

Add a (previously unassigned) node to a cluster.

Definition at line 22 of file Clustering.cpp.

6.3.3.2 cluster& EnsembleClustering::Clustering::clusterOf (node *u*) [inline]

Return the cluster (id) in which a node is contained.

Definition at line 55 of file Clustering.h.

6.3.3.3 cluster EnsembleClustering::Clustering::firstCluster ()

Get the lowest cluster id;.

Definition at line 53 of file Clustering.cpp.

6.3.3.4 bool EnsembleClustering::Clustering::isProper (const Graph & *G*)

Check whether this clustering is a proper clustering of the graph, i.e.

a disjoint partition of the whole node set.

Definition at line 49 of file Clustering.cpp.

6.3.3.5 cluster EnsembleClustering::Clustering::lastCluster ()

Get the highest cluster id that has been assigned.

This gives an upper bound for the number of clusters in this clustering, although not the actual number of clusters since clusters can become empty.

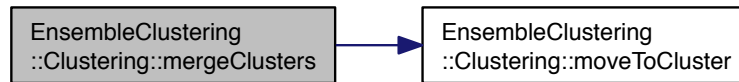
Definition at line 57 of file Clustering.cpp.

6.3.3.6 void EnsembleClustering::Clustering::mergeClusters (cluster *c*, cluster *d*)

Assigns the nodes from both clusters to a new cluster.

Definition at line 37 of file Clustering.cpp.

Here is the call graph for this function:



6.3.3.7 void EnsembleClustering::Clustering::moveToCluster (cluster *c*, node *u*)

Move a (previously assigned) node to a cluster.

Definition at line 32 of file Clustering.cpp.

6.3.3.8 cluster& EnsembleClustering::Clustering::operator[] (const node & *u*) [inline]

Index operator.

Parameters

in	<i>u</i>	a node
----	----------	--------

Definition at line 39 of file Clustering.h.

6.3.3.9 const cluster& EnsembleClustering::Clustering::operator[] (const node & *u*) const [inline]

Index operator for const instances of this class.

Parameters

in	<i>u</i>	a node
----	----------	--------

Definition at line 47 of file Clustering.h.

6.3.3.10 void EnsembleClustering::Clustering::toSingleton (node *u*)

Creates a singleton cluster containing the node.

Definition at line 27 of file Clustering.cpp.

6.3.4 Member Data Documentation

6.3.4.1 cluster EnsembleClustering::Clustering::nextCluster [protected]

next free cluster id for new cluster

Definition at line 21 of file Clustering.h.

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

- src/clustering/Clustering.h
- src/clustering/Clustering.cpp

6.4 EnsembleClustering::ClusteringGenerator Class Reference

```
#include <ClusteringGenerator.h>
```

Public Member Functions

- [ClusteringGenerator](#) ()
- virtual [~ClusteringGenerator](#) ()
- virtual [Clustering](#) & [makeSingletonClustering](#) (const [Graph](#) &G)
Make a singleton clustering of G, i.e.
- virtual [Clustering](#) & [makeOneClustering](#) (const [Graph](#) &G)
Make a 1-clustering of G, i.e.

6.4.1 Detailed Description

Definition at line 15 of file ClusteringGenerator.h.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 EnsembleClustering::ClusteringGenerator::ClusteringGenerator ()

Definition at line 12 of file ClusteringGenerator.cpp.

6.4.2.2 EnsembleClustering::ClusteringGenerator::~~ClusteringGenerator () [virtual]

Definition at line 17 of file ClusteringGenerator.cpp.

6.4.3 Member Function Documentation

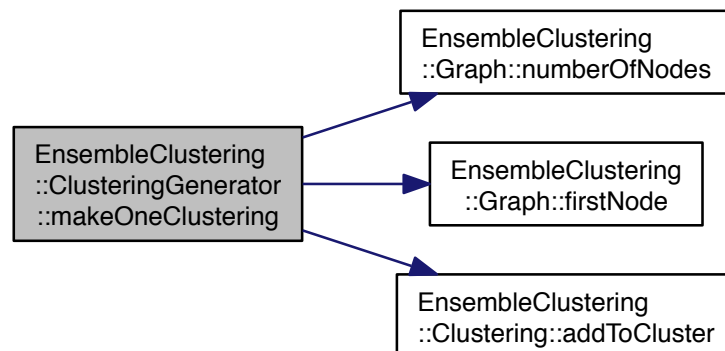
6.4.3.1 Clustering & EnsembleClustering::ClusteringGenerator::makeOneClustering (const Graph & G) [virtual]

Make a 1-clustering of G, i.e.

a clustering in which all nodes belong to the same cluster.

Definition at line 30 of file ClusteringGenerator.cpp.

Here is the call graph for this function:



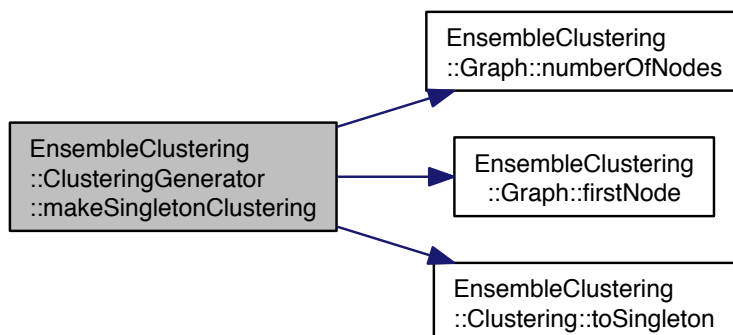
6.4.3.2 Clustering & EnsembleClustering::ClusteringGenerator::makeSingletonClustering (const Graph & G) [virtual]

Make a singleton clustering of G, i.e.

a clustering in which every node belongs to its own cluster.

Definition at line 21 of file ClusteringGenerator.cpp.

Here is the call graph for this function:



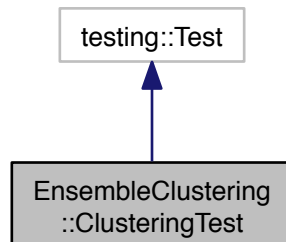
The documentation for this class was generated from the following files:

- [src/clustering/ClusteringGenerator.h](#)
- [src/clustering/ClusteringGenerator.cpp](#)

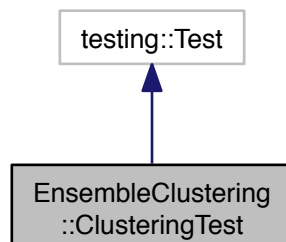
6.5 EnsembleClustering::ClusteringTest Class Reference

```
#include <ClusteringTest.h>
```

Inheritance diagram for EnsembleClustering::ClusteringTest:



Collaboration diagram for EnsembleClustering::ClusteringTest:



6.5.1 Detailed Description

Definition at line 22 of file ClusteringTest.h.

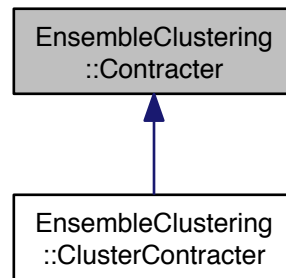
The documentation for this class was generated from the following file:

- [src/clustering/test/ClusteringTest.h](#)

6.6 EnsembleClustering::Contracter Class Reference

```
#include <Contracter.h>
```

Inheritance diagram for EnsembleClustering::Contracter:



Public Member Functions

- [Contracter](#) ()
- virtual [~Contracter](#) ()
- virtual [node contract](#) ([node u](#), [node v](#))

6.6.1 Detailed Description

Definition at line 17 of file [Contracter.h](#).

6.6.2 Constructor & Destructor Documentation

6.6.2.1 EnsembleClustering::Contracter::Contracter ()

Definition at line 12 of file [Contracter.cpp](#).

6.6.2.2 EnsembleClustering::Contracter::~~Contracter () [virtual]

Definition at line 17 of file [Contracter.cpp](#).

6.6.3 Member Function Documentation

6.6.3.1 [node](#) EnsembleClustering::Contracter::contract ([node u](#), [node v](#)) [virtual]

Definition at line 21 of file [Contracter.cpp](#).

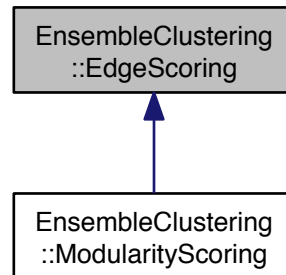
The documentation for this class was generated from the following files:

- [src/coarsening/Contracter.h](#)
- [src/coarsening/Contracter.cpp](#)

6.7 EnsembleClustering::EdgeScoring Class Reference

```
#include <EdgeScoring.h>
```

Inheritance diagram for EnsembleClustering::EdgeScoring:



Public Member Functions

- [EdgeScoring](#) ()
- virtual [~EdgeScoring](#) ()
- virtual double [scoreEdge](#) ([Node](#) u, [Node](#) v)=0

6.7.1 Detailed Description

Definition at line 16 of file [EdgeScoring.h](#).

6.7.2 Constructor & Destructor Documentation

6.7.2.1 EnsembleClustering::EdgeScoring::EdgeScoring ()

Definition at line 12 of file [EdgeScoring.cpp](#).

6.7.2.2 EnsembleClustering::EdgeScoring::~~EdgeScoring () [virtual]

Definition at line 17 of file [EdgeScoring.cpp](#).

6.7.3 Member Function Documentation

6.7.3.1 virtual double EnsembleClustering::EdgeScoring::scoreEdge ([Node](#) u, [Node](#) v) [pure virtual]

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

- [src/scoring/EdgeScoring.h](#)
- [src/scoring/EdgeScoring.cpp](#)

6.8 EnsembleClustering::EnsembleClusterer Class Reference

```
#include <EnsembleClusterer.h>
```

Public Member Functions

- [EnsembleClusterer](#) ()
- virtual [~EnsembleClusterer](#) ()

6.8.1 Detailed Description

Definition at line 13 of file EnsembleClusterer.h.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 EnsembleClustering::EnsembleClusterer::EnsembleClusterer ()

Definition at line 12 of file EnsembleClusterer.cpp.

6.8.2.2 EnsembleClustering::EnsembleClusterer::~~EnsembleClusterer () [virtual]

Definition at line 17 of file EnsembleClusterer.cpp.

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

- src/ensemble/[EnsembleClusterer.h](#)
- src/ensemble/[EnsembleClusterer.cpp](#)

6.9 EnsembleClustering::Graph Class Reference

[Graph](#) interface.

```
#include <Graph.h>
```

Public Member Functions

- [Graph](#) ()
methods
- [Graph](#) (stinger *stingerG)
Initialize with STINGER graph.
- [~Graph](#) ()
- stinger * [asSTINGER](#) () const
Return the internal STINGER data structure.
- void [insertEdge](#) (node u, node v, double weight=defaultEdgeWeight, int64_t type=defaultEdgeType, int64_t timestamp=defaultTimeStamp)
Insert a weighted, undirected edge.
- bool [hasEdge](#) (node u, node v) const
Check if undirected edge {u,v} exists in G.
- double [weight](#) (node v) const
Return node weight.
- double [weight](#) (edge uv) const

- *Return edge weight.*
double [weight](#) ([node](#) u, [node](#) v) const
- *Return edge weight.*
double [totalEdgeWeight](#) () const
- *Get the sum of the weight of all edges.*
int64_t [degree](#) ([node](#) u) const
- *Return the degree (number of incident edges).*
int64_t [numberOfEdges](#) () const
- *Return the number of edges in the graph.*
int64_t [numberOfNodes](#) () const
- *Return the number of (non-isolated) nodes in the graph.*
[node](#) [firstNode](#) () const
- *Get the first node index (for iteration over all nodes)*
[node](#) [lastNode](#) () const
- *Get the last node index (for iteration over all nodes).*
template<typename Callback >
void [forallEdges](#) (bool parallel, Callback callback)

Static Public Attributes

- static constexpr double [defaultEdgeWeight](#) = 1.0
default parameters
- static const int64_t [defaultEdgeType](#) = 0
- static const int64_t [defaultTimeStamp](#) = 0

Protected Attributes

- stinger * [stingerG](#)

6.9.1 Detailed Description

[Graph](#) interface.

[Graph](#) encapsulates a STINGER graph object and provides a more concise interface to it.

The graph concept modelled is

- undirected
- weighted
- without self-loops (use node weights instead)

Definition at line 77 of file Graph.h.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 EnsembleClustering::Graph::Graph ()

methods

Construct [Graph](#) object with new STINGER graph inside.

Definition at line 13 of file Graph.cpp.

6.9.2.2 EnsembleClustering::Graph::Graph (stinger * stingerG)

Initialize with STINGER graph.

Parameters

<i>in</i>	<i>stingerG</i>	a STINGER graph struct
-----------	-----------------	------------------------

Definition at line 20 of file Graph.cpp.

6.9.2.3 EnsembleClustering::Graph::~~Graph ()

Definition at line 17 of file Graph.cpp.

6.9.3 Member Function Documentation

6.9.3.1 stinger * EnsembleClustering::Graph::asSTINGER () const

Return the internal STINGER data structure.

Definition at line 26 of file Graph.cpp.

6.9.3.2 int64_t EnsembleClustering::Graph::degree (node *u*) const

Return the degree (number of incident edges).

Definition at line 69 of file Graph.cpp.

6.9.3.3 node EnsembleClustering::Graph::firstNode () const

Get the first node index (for iteration over all nodes)

Definition at line 65 of file Graph.cpp.

6.9.3.4 template<typename Callback > void EnsembleClustering::Graph::forallEdges (bool *parallel*, Callback *callback*) [inline]

Definition at line 195 of file Graph.h.

6.9.3.5 bool EnsembleClustering::Graph::hasEdge (node *u*, node *v*) const

Check if undirected edge {*u*,*v*} exists in G.

Definition at line 36 of file Graph.cpp.

6.9.3.6 void EnsembleClustering::Graph::insertEdge (node *u*, node *v*, double *weight* = defaultEdgeWeight, int64_t *type* = defaultEdgeType, int64_t *timestamp* = defaultTimeStamp)

Insert a weighted, undirected edge.

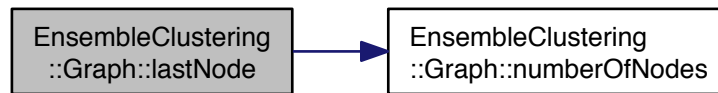
Definition at line 30 of file Graph.cpp.

6.9.3.7 node EnsembleClustering::Graph::lastNode () const

Get the last node index (for iteration over all nodes).

Definition at line 85 of file Graph.cpp.

Here is the call graph for this function:

**6.9.3.8 int64_t EnsembleClustering::Graph::numberOfEdges () const**

Return the number of edges in the graph.

Definition at line 54 of file Graph.cpp.

6.9.3.9 int64_t EnsembleClustering::Graph::numberOfNodes () const

Return the number of (non-isolated) nodes in the graph.

TODO: Maybe this should be changed to support isolated nodes.

Definition at line 59 of file Graph.cpp.

6.9.3.10 double EnsembleClustering::Graph::totalEdgeWeight () const

Get the sum of the weight of all edges.

Definition at line 76 of file Graph.cpp.

Here is the call graph for this function:

**6.9.3.11 double EnsembleClustering::Graph::weight (node v) const**

Return node weight.

Definition at line 43 of file Graph.cpp.

6.9.3.12 double EnsembleClustering::Graph::weight (edge *uv*) const

Return edge weight.

Definition at line 47 of file Graph.cpp.

6.9.3.13 double EnsembleClustering::Graph::weight (node *u*, node *v*) const [inline]

Return edge weight.

Equivalent to getWeight(edge uv)

Definition at line 144 of file Graph.h.

6.9.4 Member Data Documentation

6.9.4.1 const int64_t EnsembleClustering::Graph::defaultEdgeType = 0 [static]

Definition at line 92 of file Graph.h.

6.9.4.2 constexpr double EnsembleClustering::Graph::defaultEdgeWeight = 1.0 [static]

default parameters

Definition at line 91 of file Graph.h.

6.9.4.3 const int64_t EnsembleClustering::Graph::defaultTimeStamp = 0 [static]

Definition at line 93 of file Graph.h.

6.9.4.4 stinger* EnsembleClustering::Graph::stingerG [protected]

Definition at line 81 of file Graph.h.

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

- [src/graph/Graph.h](#)
- [src/graph/Graph.cpp](#)

6.10 EnsembleClustering::GraphGenerator Class Reference

```
#include <GraphGenerator.h>
```

Public Member Functions

- [GraphGenerator](#) ()
- virtual [~GraphGenerator](#) ()
- [Graph](#) & [makeErdosRenyiGraph](#) (int64_t n, double p)
Generate a random graph according to the Erdos-Renyi model.
- [Graph](#) & [makeCircularGraph](#) (int64_t n)
Generate a graph whose nodes and edges form a circle.
- [Graph](#) & [makeCompleteGraph](#) (int64_t n)

6.10.1 Detailed Description

Definition at line 16 of file GraphGenerator.h.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 EnsembleClustering::GraphGenerator::GraphGenerator ()

Definition at line 12 of file GraphGenerator.cpp.

6.10.2.2 EnsembleClustering::GraphGenerator::~~GraphGenerator () [virtual]

Definition at line 17 of file GraphGenerator.cpp.

6.10.3 Member Function Documentation

6.10.3.1 Graph & EnsembleClustering::GraphGenerator::makeCircularGraph (int64_t *n*)

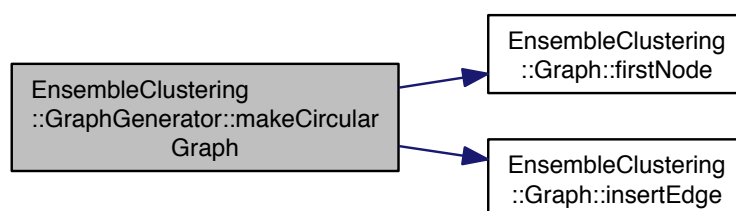
Generate a graph whose nodes and edges form a circle.

Parameters

<i>in</i>	<i>n</i>	number of nodes
-----------	----------	-----------------

Definition at line 37 of file GraphGenerator.cpp.

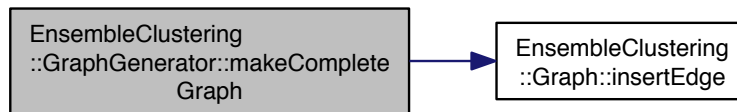
Here is the call graph for this function:



6.10.3.2 Graph & EnsembleClustering::GraphGenerator::makeCompleteGraph (int64_t *n*)

Definition at line 45 of file GraphGenerator.cpp.

Here is the call graph for this function:



6.10.3.3 Graph & EnsembleClustering::GraphGenerator::makeErdosRenyiGraph (`int64_t n`, `double p`)

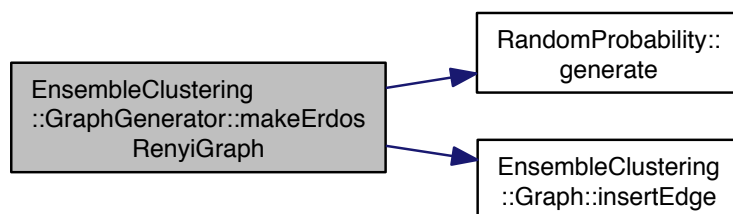
Generate a random graph according to the Erdos-Renyi model.

Parameters

<code>in</code>	<code>n</code>	number of nodes
<code>in</code>	<code>p</code>	edge probability

Definition at line 25 of file `GraphGenerator.cpp`.

Here is the call graph for this function:



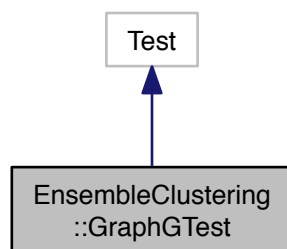
The documentation for this class was generated from the following files:

- [src/graph/GraphGenerator.h](#)
- [src/graph/GraphGenerator.cpp](#)

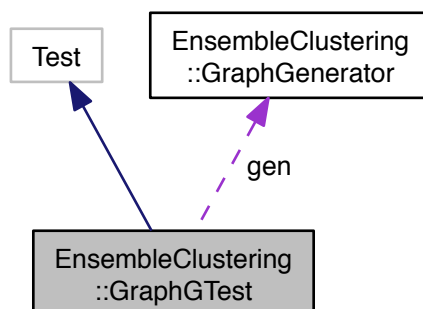
6.11 EnsembleClustering::GraphGTest Class Reference

```
#include <GraphGTest.h>
```

Inheritance diagram for EnsembleClustering::GraphGTest:



Collaboration diagram for EnsembleClustering::GraphGTest:



Public Member Functions

- virtual void [SetUp](#) ()
- virtual void [TearDown](#) ()

Protected Attributes

- [GraphGenerator](#) *gen*

6.11.1 Detailed Description

Definition at line 20 of file GraphGTest.h.

6.11.2 Member Function Documentation

6.11.2.1 void EnsembleClustering::GraphGTest::SetUp () [virtual]

Definition at line 14 of file GraphGTest.cpp.

6.11.2.2 void EnsembleClustering::GraphGTest::TearDown () [virtual]

Definition at line 17 of file GraphGTest.cpp.

6.11.3 Member Data Documentation

6.11.3.1 GraphGenerator EnsembleClustering::GraphGTest::gen [protected]

Definition at line 24 of file GraphGTest.h.

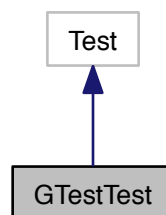
The documentation for this class was generated from the following files:

- src/graph/test/[GraphGTest.h](#)
- src/graph/test/[GraphGTest.cpp](#)

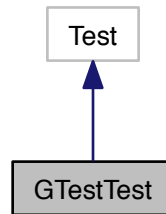
6.12 GTestTest Class Reference

```
#include <TestGTest.h>
```

Inheritance diagram for GTestTest:



Collaboration diagram for GTestTest:



Protected Member Functions

- virtual void [SetUp](#) ()

6.12.1 Detailed Description

Definition at line 13 of file TestGTest.h.

6.12.2 Member Function Documentation

6.12.2.1 virtual void GTestTest::SetUp () `[inline], [protected], [virtual]`

Definition at line 16 of file TestGTest.h.

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

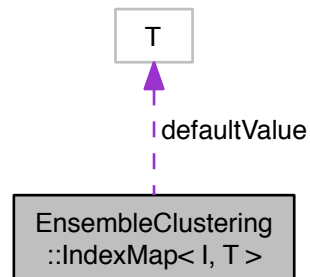
- src/test/[TestGTest.h](#)

6.13 EnsembleClustering::IndexMap< I, T > Class Template Reference

An [IndexMap](#) implements a 1-based mapping from an integer index type to an arbitray value type.

```
#include <IndexMap.h>
```

Collaboration diagram for EnsembleClustering::IndexMap< I, T >:



Public Member Functions

- [IndexMap](#) (int64_t n)
- [IndexMap](#) (int64_t n, T defaultValue)
Construct a new [IndexMap](#) which holds n entries .
- virtual [~IndexMap](#) ()
- T & [operator\[\]](#) (const I &index)
Index operator.
- const T & [operator\[\]](#) (const I &index) const
Index operator for const instances of this class.

Protected Attributes

- T * [array](#)
array of size (n+1). array[0] is not a valid entry, since node indices are 1-based
- T [defaultValue](#)
- int64_t n

6.13.1 Detailed Description

```
template<typename I, typename T>class EnsembleClustering::IndexMap< I, T >
```

An [IndexMap](#) implements a 1-based mapping from an integer index type to an arbitray value type.

Definition at line 17 of file IndexMap.h.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 `template<typename I, typename T > EnsembleClustering::IndexMap< I, T >::IndexMap (int64_t n)`
[inline]

Definition at line 60 of file IndexMap.h.

6.13.2.2 `template<typename I, typename T> EnsembleClustering::IndexMap< I, T >::IndexMap (int64_t n, T defaultValue) [inline]`

Construct a new [IndexMap](#) which holds n entries .

Parameters

<code>in</code>	<code>defaultValue</code>	all entries are initialized to this value
-----------------	---------------------------	---

Definition at line 67 of file IndexMap.h.

6.13.2.3 `template<typename I, typename T> EnsembleClustering::IndexMap< I, T >::~~IndexMap () [inline], [virtual]`

Definition at line 77 of file IndexMap.h.

6.13.3 Member Function Documentation

6.13.3.1 `template<typename I, typename T> T & EnsembleClustering::IndexMap< I, T >::operator[] (const I & index) [inline]`

Index operator.

Parameters

<code>in</code>	<code>u</code>	a node
-----------------	----------------	--------

Definition at line 82 of file IndexMap.h.

6.13.3.2 `template<typename I, typename T> const T & EnsembleClustering::IndexMap< I, T >::operator[] (const I & index) const [inline]`

Index operator for const instances of this class.

Parameters

<code>in</code>	<code>u</code>	a node
-----------------	----------------	--------

Definition at line 87 of file IndexMap.h.

6.13.4 Member Data Documentation

6.13.4.1 `template<typename I, typename T> T* EnsembleClustering::IndexMap< I, T >::array [protected]`

array of size (n+1). array[0] is not a valid entry, since node indices are 1-based

Definition at line 22 of file IndexMap.h.

6.13.4.2 `template<typename I, typename T> T EnsembleClustering::IndexMap< I, T >::defaultValue [protected]`

Definition at line 23 of file IndexMap.h.

6.13.4.3 `template<typename I, typename T> int64_t EnsembleClustering::IndexMap<I, T>::n` `[protected]`

Definition at line 24 of file `IndexMap.h`.

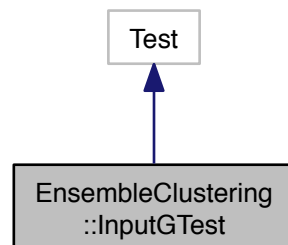
The documentation for this class was generated from the following file:

- `src/aux/IndexMap.h`

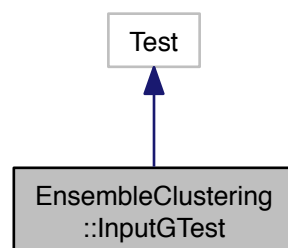
6.14 EnsembleClustering::InputGTest Class Reference

```
#include <InputGTest.h>
```

Inheritance diagram for `EnsembleClustering::InputGTest`:



Collaboration diagram for `EnsembleClustering::InputGTest`:



6.14.1 Detailed Description

Definition at line 18 of file `InputGTest.h`.

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

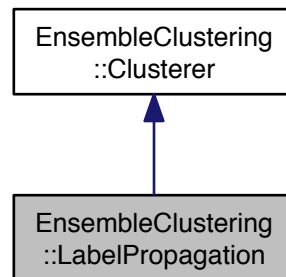
- `src/input/test/InputGTest.h`

6.15 EnsembleClustering::LabelPropagation Class Reference

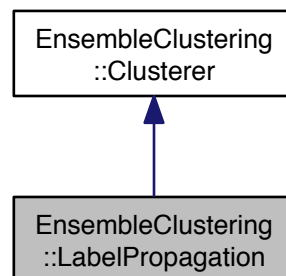
As described in Ovelgoenne et al: An Ensemble Learning Strategy for [Graph Clustering](#) Raghavan et al.

```
#include <LabelPropagation.h>
```

Inheritance diagram for EnsembleClustering::LabelPropagation:



Collaboration diagram for EnsembleClustering::LabelPropagation:



Public Member Functions

- [LabelPropagation](#) ()
- virtual [~LabelPropagation](#) ()
- virtual [Clustering & run](#) ([Graph](#) &G)

6.15.1 Detailed Description

As described in Ovelgoenne et al: An Ensemble Learning Strategy for [Graph Clustering](#) Raghavan et al.

proposed a label propagation algorithm for graph clustering. This algorithm initializes every vertex of a graph with a unique label. Then, in iterative sweeps over the set of vertices the vertex labels are updated. A vertex gets the label

that the maximum number of its neighbors have. The procedure is stopped when every vertex has the label that at least half of its neighbors have.

Definition at line 30 of file LabelPropagation.h.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 EnsembleClustering::LabelPropagation::LabelPropagation ()

Definition at line 12 of file LabelPropagation.cpp.

6.15.2.2 EnsembleClustering::LabelPropagation::~~LabelPropagation () [virtual]

Definition at line 17 of file LabelPropagation.cpp.

6.15.3 Member Function Documentation

6.15.3.1 Clustering & EnsembleClustering::LabelPropagation::run (Graph & G) [virtual]

< a label is the same as a cluster id

< neighborLabelCounts[v] maps label -> frequency in the neighbors of v

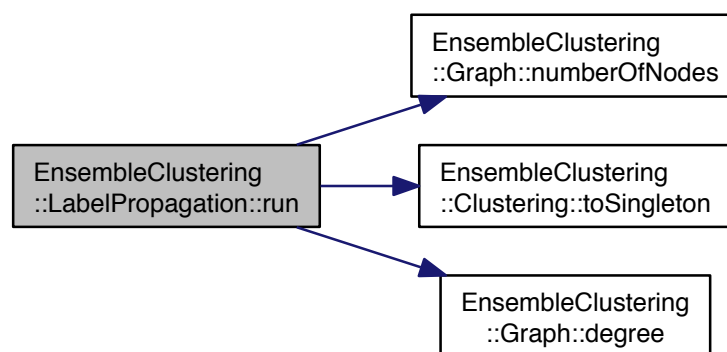
< number of nodes which already have the majority label

< number of iterations

Implements [EnsembleClustering::Clusterer](#).

Definition at line 21 of file LabelPropagation.cpp.

Here is the call graph for this function:



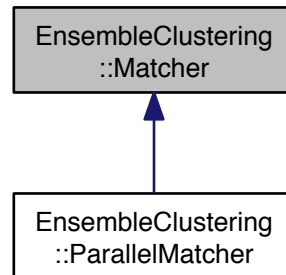
The documentation for this class was generated from the following files:

- [src/clustering/LabelPropagation.h](#)
- [src/clustering/LabelPropagation.cpp](#)

6.16 EnsembleClustering::Matcher Class Reference

```
#include <Matcher.h>
```

Inheritance diagram for EnsembleClustering::Matcher:



Public Member Functions

- [Matcher](#) ()
- virtual [~Matcher](#) ()
- virtual [Matching](#) & [run](#) (const [Graph](#) &G)=0

6.16.1 Detailed Description

Definition at line 16 of file [Matcher.h](#).

6.16.2 Constructor & Destructor Documentation

6.16.2.1 EnsembleClustering::Matcher::Matcher ()

Definition at line 12 of file [Matcher.cpp](#).

6.16.2.2 EnsembleClustering::Matcher::~~Matcher () [virtual]

Definition at line 17 of file [Matcher.cpp](#).

6.16.3 Member Function Documentation

6.16.3.1 virtual [Matching](#) & EnsembleClustering::Matcher::run (const [Graph](#) & G) [pure virtual]

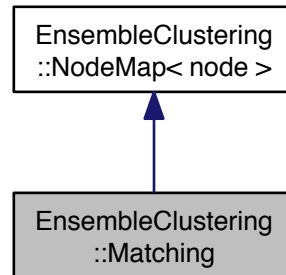
The documentation for this class was generated from the following files:

- [src/matching/Matcher.h](#)
- [src/matching/Matcher.cpp](#)

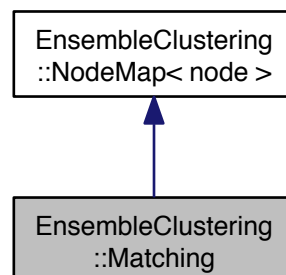
6.17 EnsembleClustering::Matching Class Reference

```
#include <Matching.h>
```

Inheritance diagram for EnsembleClustering::Matching:



Collaboration diagram for EnsembleClustering::Matching:



Public Member Functions

- `Matching` (`int64_t n`)
Construct new matching.
- virtual `~Matching` ()
Destructor.
- void `match` (const `node` &u, const `node` &v)
Set two nodes as eachothers matching partners.
- void `unmatch` (const `node` &u, const `node` &v)
Reset the two nodes to unmatched.
- bool `isMatched` (const `node` &u) const
Check if node is matched.
- bool `areMatched` (const `node` &u, const `node` &v) const

- Check if the two nodes are matched.*
- `bool isProper (Graph &G) const`
Check whether this is a proper matching in the graph, i.e.
- `Matching & operator= (const Matching &from)`
copy semantics
- `void clone (const Matching &from)`
Properly copy this object.
- `void dispose ()`
Properly destruct this object.

Additional Inherited Members

6.17.1 Detailed Description

Definition at line 17 of file Matching.h.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 EnsembleClustering::Matching::Matching (int64_t n)

Construct new matching.

Parameters

<code>in</code>	<code>n</code>	maximum number of nodes
-----------------	----------------	-------------------------

Definition at line 12 of file Matching.cpp.

6.17.2.2 EnsembleClustering::Matching::~Matching () [virtual]

Destructor.

Definition at line 19 of file Matching.cpp.

6.17.3 Member Function Documentation

6.17.3.1 bool EnsembleClustering::Matching::areMatched (const node & u, const node & v) const

Check if the two nodes are matched.

Definition at line 82 of file Matching.cpp.

6.17.3.2 void EnsembleClustering::Matching::clone (const Matching & from)

Properly copy this object.

Definition at line 78 of file Matching.cpp.

6.17.3.3 void EnsembleClustering::Matching::dispose ()

Properly destruct this object.

Definition at line 88 of file Matching.cpp.

6.17.3.4 `bool EnsembleClustering::Matching::isMatched (const node & u) const`

Check if node is matched.

Parameters

in	<i>u</i>	a node
out	<i>true</i>	if u is matched

Definition at line 23 of file Matching.cpp.

6.17.3.5 `bool EnsembleClustering::Matching::isProper (Graph & G) const`

Check whether this is a proper matching in the graph, i.e.

no two edges are adjacent.

[in] G a graph

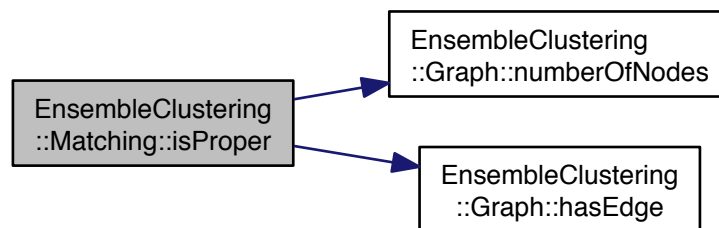
Parameters

out	<i>true</i>	if this is a proper matching
-----	-------------	------------------------------

The content of this data structure represents a matching iff (for all v in V : $M[v] = 0$ or $M[M[v]] = v$) and (for all (u,v) in M): (u,v) in E

Definition at line 27 of file Matching.cpp.

Here is the call graph for this function:



6.17.3.6 `void EnsembleClustering::Matching::match (const node & u, const node & v)`

Set two nodes as eachothers matching partners.

Definition at line 57 of file Matching.cpp.

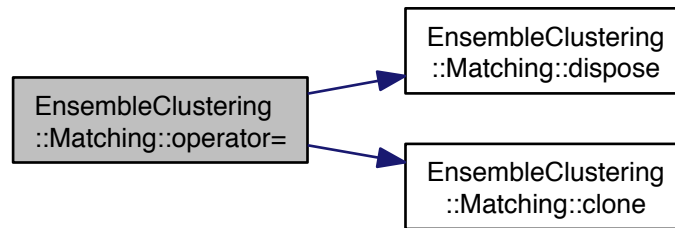
6.17.3.7 `Matching & EnsembleClustering::Matching::operator= (const Matching & from)`

copy semantics

Assignment operator.

Definition at line 69 of file Matching.cpp.

Here is the call graph for this function:



6.17.3.8 void EnsembleClustering::Matching::unmatch (const node & *u*, const node & *v*)

Reset the two nodes to unmatched.

Definition at line 63 of file Matching.cpp.

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

- [src/matching/Matching.h](#)
- [src/matching/Matching.cpp](#)

6.18 EnsembleClustering::MatchingContracter Class Reference

```
#include <MatchingContracter.h>
```

Public Member Functions

- [MatchingContracter](#) ()
- virtual [~MatchingContracter](#) ()

6.18.1 Detailed Description

Definition at line 13 of file MatchingContracter.h.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 EnsembleClustering::MatchingContracter::MatchingContracter ()

Definition at line 12 of file MatchingContracter.cpp.

6.18.2.2 EnsembleClustering::MatchingContracter::~~MatchingContracter () [virtual]

Definition at line 17 of file MatchingContracter.cpp.

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

- src/coarsening/[MatchingContracter.h](#)
- src/coarsening/[MatchingContracter.cpp](#)

6.19 EnsembleClustering::METISParser Class Reference

```
#include <METISParser.h>
```

Public Member Functions

- [METISParser](#) ()
- virtual [~METISParser](#) ()
- virtual void [open](#) (std::string [graphPath](#))
Open a METIS graph file.
- virtual std::pair< int, int > [getHeader](#) ()
Get the METIS graph file header.
- virtual bool [hasNext](#) ()
Test if graph file has a next line.
- virtual std::vector< [node](#) > [getNext](#) ()
Get adjacencies from the next line in the METIS graph file.
- virtual void [close](#) ()
Close input file and clean up.

Protected Attributes

- std::string [graphPath](#)
- std::ifstream [graphFile](#)
- std::string [line](#)
- int [nodeCount](#)

6.19.1 Detailed Description

Definition at line 25 of file METISParser.h.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 EnsembleClustering::METISParser::METISParser ()

Definition at line 39 of file METISParser.cpp.

6.19.2.2 EnsembleClustering::METISParser::~~METISParser () [virtual]

Definition at line 43 of file METISParser.cpp.

6.19.3 Member Function Documentation

6.19.3.1 void EnsembleClustering::METISParser::close () [virtual]

Close input file and clean up.

Definition at line 103 of file METISParser.cpp.

6.19.3.2 `std::pair< int, int > EnsembleClustering::METISParser::getHeader ()` [virtual]

Get the METIS graph file header.

Definition at line 62 of file METISParser.cpp.

6.19.3.3 `std::vector< node > EnsembleClustering::METISParser::getNext ()` [virtual]

Get adjacencies from the next line in the METIS graph file.

Definition at line 87 of file METISParser.cpp.

6.19.3.4 `bool EnsembleClustering::METISParser::hasNext ()` [virtual]

Test if graph file has a next line.

Definition at line 79 of file METISParser.cpp.

6.19.3.5 `void EnsembleClustering::METISParser::open (std::string graphPath)` [virtual]

Open a METIS graph file.

Definition at line 48 of file METISParser.cpp.

6.19.4 Member Data Documentation

6.19.4.1 `std::ifstream EnsembleClustering::METISParser::graphFile` [protected]

Definition at line 61 of file METISParser.h.

6.19.4.2 `std::string EnsembleClustering::METISParser::graphPath` [protected]

Definition at line 60 of file METISParser.h.

6.19.4.3 `std::string EnsembleClustering::METISParser::line` [protected]

Definition at line 62 of file METISParser.h.

6.19.4.4 `int EnsembleClustering::METISParser::nodeCount` [protected]

Definition at line 63 of file METISParser.h.

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

- [src/input/METISParser.h](#)
- [src/input/METISParser.cpp](#)

6.20 EnsembleClustering::METIStoSTINGER Class Reference

This class provides a user interface for reading a METIS graph file and returning a STINGER-based graph object.

```
#include <METIStoSTINGER.h>
```

Public Member Functions

- [METISToSTINGER](#) ()
- virtual [~METISToSTINGER](#) ()
- virtual [Graph](#) * [read](#) (std::string graphPath)

6.20.1 Detailed Description

This class provides a user interface for reading a METIS graph file and returning a STINGER-based graph object.
Definition at line 22 of file METISToSTINGER.h.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 EnsembleClustering::METISToSTINGER::METISToSTINGER ()

Definition at line 19 of file METISToSTINGER.cpp.

6.20.2.2 EnsembleClustering::METISToSTINGER::~~METISToSTINGER () [virtual]

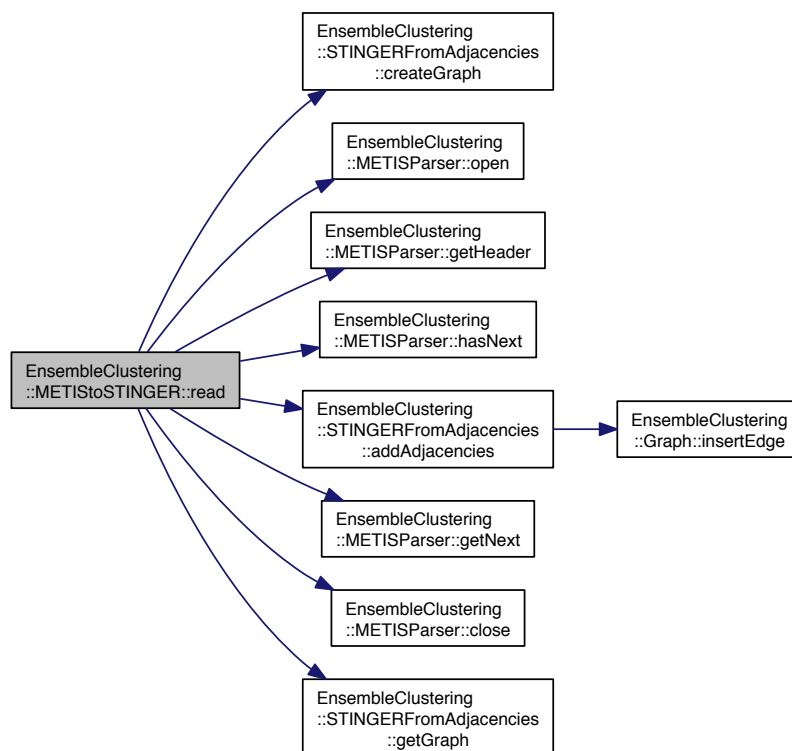
Definition at line 24 of file METISToSTINGER.cpp.

6.20.3 Member Function Documentation

6.20.3.1 [Graph](#) * EnsembleClustering::METISToSTINGER::read (std::string *graphPath*) [virtual]

Definition at line 28 of file METISToSTINGER.cpp.

Here is the call graph for this function:



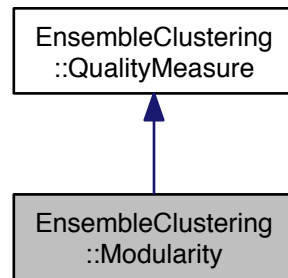
The documentation for this class was generated from the following files:

- `src/input/METISToSTINGER.h`
- `src/input/METISToSTINGER.cpp`

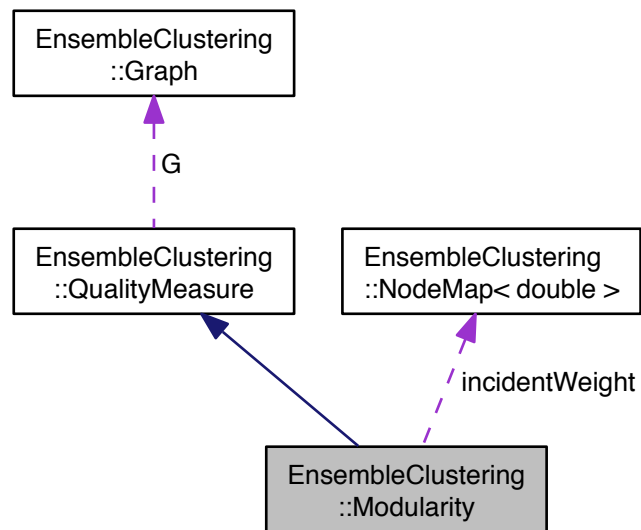
6.21 EnsembleClustering::Modularity Class Reference

```
#include <Modularity.h>
```

Inheritance diagram for EnsembleClustering::Modularity:



Collaboration diagram for EnsembleClustering::Modularity:



Public Member Functions

- [Modularity](#) ([Graph](#) &G)
- virtual [~Modularity](#) ()
- virtual double [getQuality](#) ([Clustering](#) &zeta)

Returns the [Modularity](#) of the given clustering with respect to the graph instance.

Protected Member Functions

- virtual void [precompute](#) ()

Precompute some values depending on the graph instance to be used in getQuality.

Protected Attributes

- [NodeMap](#)< double > * [incidentWeight](#)
node -> sum of the weight of incident edges

6.21.1 Detailed Description

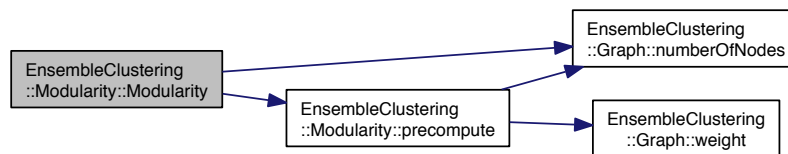
Definition at line 22 of file Modularity.h.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 EnsembleClustering::Modularity::Modularity (Graph & G)

Definition at line 14 of file Modularity.cpp.

Here is the call graph for this function:



6.21.2.2 EnsembleClustering::Modularity::~~Modularity () [virtual]

Definition at line 19 of file Modularity.cpp.

6.21.3 Member Function Documentation

6.21.3.1 double EnsembleClustering::Modularity::getQuality (Clustering & zeta) [virtual]

Returns the [Modularity](#) of the given clustering with respect to the graph instance.

[Modularity](#) is defined as:

$$mod(\zeta) := \frac{\sum_{C \in \zeta} \sum_{e \in E(C)} \omega(e)}{\sum_{e \in E} \omega(e)} - \frac{\sum_{C \in \zeta} (\sum_{v \in C} \omega(v))^2}{4 \left(\sum_{e \in E} \omega(e) \right)^2}$$

< term $\sum_{C \in \zeta} \sum_{e \in E(C)} \omega(e)$ >

< term $\sum_{C \in \zeta} (\sum_{v \in C} \omega(v))^2$ >

< cluster -> weight of its internal edges

< term $\sum_{C \in \zeta} \sum_{e \in E(C)} \omega(e)$ >

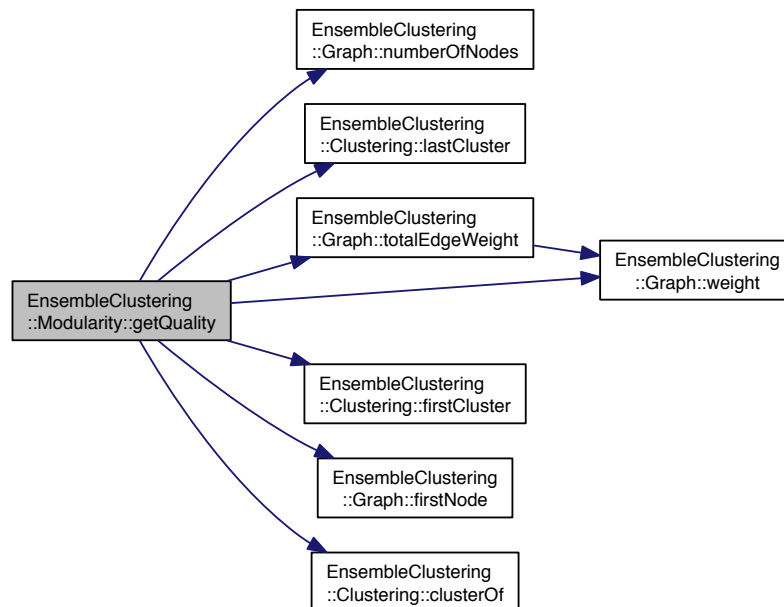
< cluster -> sum of the weights of incident edges for all nodes

< term $\sum_{C \in \zeta} (\sum_{v \in C} \omega(v))^2$ >

Implements [EnsembleClustering::QualityMeasure](#).

Definition at line 36 of file Modularity.cpp.

Here is the call graph for this function:

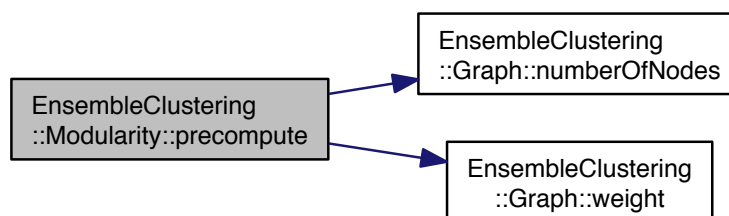


6.21.3.2 void EnsembleClustering::Modularity::precompute () [protected],[virtual]

Precompute some values depending on the graph instance to be used in `getQuality`.

Definition at line 23 of file Modularity.cpp.

Here is the call graph for this function:



6.21.4 Member Data Documentation

6.21.4.1 NodeMap<double>* EnsembleClustering::Modularity::incidentWeight [protected]

node -> sum of the weight of incident edges

Definition at line 26 of file Modularity.h.

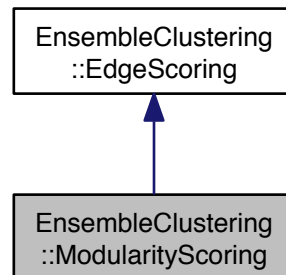
The documentation for this class was generated from the following files:

- [src/clustering/Modularity.h](#)
- [src/clustering/Modularity.cpp](#)

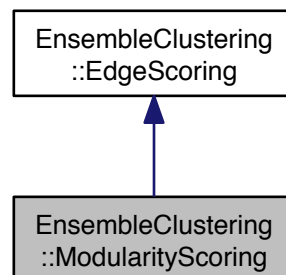
6.22 EnsembleClustering::ModularityScoring Class Reference

```
#include <ModularityScoring.h>
```

Inheritance diagram for EnsembleClustering::ModularityScoring:



Collaboration diagram for EnsembleClustering::ModularityScoring:



Public Member Functions

- [ModularityScoring\(\)](#)

- virtual `~ModularityScoring()`
- virtual double `scoreEdge(Edge uv)=0`
Returns an edge score for an edge (u,v) which expresses the modularity increase which can be gained by merging the clusters of u and v.
- virtual double `mod(Clustering clustering)=0`
Calculates the modularity of the given clustering;.
- virtual double `deltaMod(Cluster c, Cluster d)=0`
Calculates the difference in modularity that would result from a merger of two clusters.
- virtual double `cutweight(Cluster c, Cluster d)=0`
- virtual double `weight(Cluster c)=0`

6.22.1 Detailed Description

Definition at line 26 of file ModularityScoring.h.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 EnsembleClustering::ModularityScoring::ModularityScoring ()

Definition at line 13 of file ModularityScoring.cpp.

6.22.2.2 EnsembleClustering::ModularityScoring::~~ModularityScoring () [virtual]

Definition at line 18 of file ModularityScoring.cpp.

6.22.3 Member Function Documentation

6.22.3.1 virtual double EnsembleClustering::ModularityScoring::cutweight (Cluster c, Cluster d) [pure virtual]

6.22.3.2 virtual double EnsembleClustering::ModularityScoring::deltaMod (Cluster c, Cluster d) [pure virtual]

Calculates the difference in modularity that would result from a merger of two clusters.

6.22.3.3 double EnsembleClustering::ModularityScoring::mod (Clustering clustering) [pure virtual]

Calculates the modularity of the given clustering;.

Definition at line 25 of file ModularityScoring.cpp.

6.22.3.4 virtual double EnsembleClustering::ModularityScoring::scoreEdge (Edge uv) [pure virtual]

Returns an edge score for an edge (u,v) which expresses the modularity increase which can be gained by merging the clusters of u and v.

Parameters

in	<i>u</i>	source node id
out	<i>v</i>	target node id

6.22.3.5 virtual double EnsembleClustering::ModularityScoring::weight (Cluster c) [pure virtual]

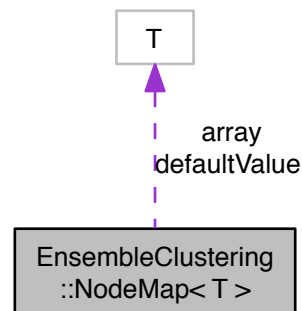
The documentation for this class was generated from the following files:

- [src/scoring/ModularityScoring.h](#)
- [src/scoring/ModularityScoring.cpp](#)

6.23 EnsembleClustering::NodeMap< T > Class Template Reference

```
#include <NodeMap.h>
```

Collaboration diagram for EnsembleClustering::NodeMap< T >:



Public Member Functions

- [NodeMap](#) (int64_t n)
- [NodeMap](#) (int64_t n, T defaultValue)
Construct a node map which holds n entries .
- virtual [~NodeMap](#) ()
- T & [operator\[\]](#) (const [node](#) &u)
Index operator.
- const T & [operator\[\]](#) (const [node](#) &u) const
Index operator for const instances of this class.

Protected Attributes

- T * [array](#)
array of size (n+1). array[0] is not a valid entry, since node indices are 1-based
- T [defaultValue](#)
- int64_t n

6.23.1 Detailed Description

```
template<class T>class EnsembleClustering::NodeMap< T >
```

Definition at line 15 of file NodeMap.h.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 `template<class T> EnsembleClustering::NodeMap< T>::NodeMap (int64_t n) [inline]`

Definition at line 57 of file NodeMap.h.

6.23.2.2 `template<class T> EnsembleClustering::NodeMap< T>::NodeMap (int64_t n, T defaultValue) [inline]`

Construct a node map which holds n entries .

Parameters

<i>in</i>	<i>defaultValue</i>	all entries are initialized to this value
-----------	---------------------	---

Definition at line 62 of file NodeMap.h.

6.23.2.3 `template<class T> EnsembleClustering::NodeMap< T>::~~NodeMap () [inline],[virtual]`

Definition at line 71 of file NodeMap.h.

6.23.3 Member Function Documentation

6.23.3.1 `template<class T> T & EnsembleClustering::NodeMap< T>::operator[] (const node & u) [inline]`

Index operator.

Parameters

<i>in</i>	<i>u</i>	a node
-----------	----------	--------

Definition at line 75 of file NodeMap.h.

6.23.3.2 `template<class T> const T & EnsembleClustering::NodeMap< T>::operator[] (const node & u) const [inline]`

Index operator for const instances of this class.

Parameters

<i>in</i>	<i>u</i>	a node
-----------	----------	--------

Definition at line 79 of file NodeMap.h.

6.23.4 Member Data Documentation

6.23.4.1 `template<class T> T* EnsembleClustering::NodeMap< T>::array [protected]`

array of size (n+1). array[0] is not a valid entry, since node indices are 1-based

Definition at line 19 of file NodeMap.h.

6.23.4.2 `template<class T> T EnsembleClustering::NodeMap< T >::defaultValue` [protected]

Definition at line 20 of file NodeMap.h.

6.23.4.3 `template<class T> int64_t EnsembleClustering::NodeMap< T >::n` [protected]

Definition at line 21 of file NodeMap.h.

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

- [src/graph/NodeMap.h](#)

6.24 Noise Class Reference

[Noise](#) is random addition to a signal.

```
#include <Noise.h>
```

Public Member Functions

- [Noise](#) (double l, double u)
- virtual [~Noise](#) ()
- double [add](#) (double x)

Add noise to double.

Public Attributes

- double [lowerBound](#)
- double [upperBound](#)

Protected Attributes

- `std::uniform_real_distribution`
`< double >` [uniform](#)
- `std::default_random_engine` [randomEngine](#)

6.24.1 Detailed Description

[Noise](#) is random addition to a signal.

This class provides methods which add random numbers to their inputs in order to enable randomization.

Definition at line 19 of file Noise.h.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 `Noise::Noise (double l, double u)`

Parameters

<code>in</code>	<code>l</code>	lower bound for added random number
<code>in</code>	<code>u</code>	upper bound for added random number

Definition at line 12 of file Noise.cpp.

6.24.2.2 Noise::~Noise () [virtual]

Definition at line 19 of file Noise.cpp.

6.24.3 Member Function Documentation

6.24.3.1 double Noise::add (double x)

Add noise to double.

Parameters

in	x	input
out	<i>input</i>	plus noise

Definition at line 23 of file Noise.cpp.

6.24.4 Member Data Documentation

6.24.4.1 double Noise::lowerBound

Definition at line 28 of file Noise.h.

6.24.4.2 std::default_random_engine Noise::randomEngine [protected]

Definition at line 24 of file Noise.h.

6.24.4.3 std::uniform_real_distribution<double> Noise::uniform [protected]

Definition at line 23 of file Noise.h.

6.24.4.4 double Noise::upperBound

Definition at line 29 of file Noise.h.

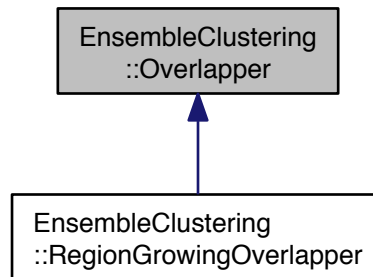
The documentation for this class was generated from the following files:

- [src/aux/Noise.h](#)
- [src/aux/Noise.cpp](#)

6.25 EnsembleClustering::Overlapper Class Reference

```
#include <Overlapper.h>
```

Inheritance diagram for EnsembleClustering::Overlapper:



Public Member Functions

- [Overlapper\(\)](#)
- virtual [~Overlapper\(\)](#)

6.25.1 Detailed Description

Definition at line 20 of file Overlapper.h.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 EnsembleClustering::Overlapper::Overlapper ()

Definition at line 12 of file Overlapper.cpp.

6.25.2.2 EnsembleClustering::Overlapper::~~Overlapper () [virtual]

Definition at line 17 of file Overlapper.cpp.

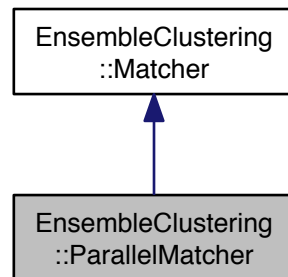
The documentation for this class was generated from the following files:

- [src/overlap/Overlapper.h](#)
- [src/overlap/Overlapper.cpp](#)

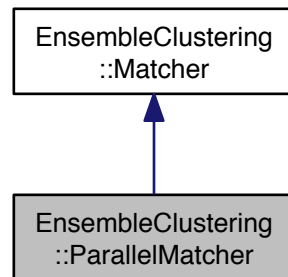
6.26 EnsembleClustering::ParallelMatcher Class Reference

```
#include <ParallelMatcher.h>
```

Inheritance diagram for EnsembleClustering::ParallelMatcher:



Collaboration diagram for EnsembleClustering::ParallelMatcher:



Public Member Functions

- [ParallelMatcher](#) ()
- virtual [~ParallelMatcher](#) ()
- virtual [Matching & run](#) ([Graph](#) &G)

Apply the parallel matching algorithm described by Manne/Bisseling Source: http://link.springer.com/chapter/10.1007%2F978-3-540-68111-3_74?LI=true#page-1.

6.26.1 Detailed Description

Definition at line 16 of file ParallelMatcher.h.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 EnsembleClustering::ParallelMatcher::ParallelMatcher ()

Definition at line 14 of file ParallelMatcher.cpp.

6.26.2.2 EnsembleClustering::ParallelMatcher::~~ParallelMatcher () [virtual]

Definition at line 19 of file ParallelMatcher.cpp.

6.26.3 Member Function Documentation

6.26.3.1 Matching & EnsembleClustering::ParallelMatcher::run (Graph & G) [virtual]

Apply the parallel matching algorithm described by Manne/Bisseling Source: http://link.springer.com/chapter/10.1007%2F978-3-540-68111-3_74?LI=true#page-1.

< candidate[v] is the preferred matching partner of v

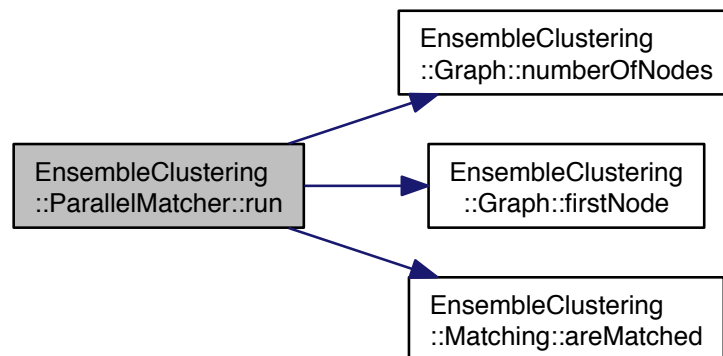
< S[v] is a set with the potential

< candidates of node v

< targets of dominating edges

Definition at line 23 of file ParallelMatcher.cpp.

Here is the call graph for this function:



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

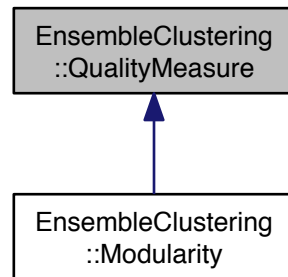
- [src/matching/ParallelMatcher.h](#)
- [src/matching/ParallelMatcher.cpp](#)

6.27 EnsembleClustering::QualityMeasure Class Reference

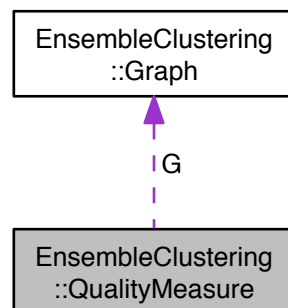
Abstract base class for all clustering quality measures.

```
#include <QualityMeasure.h>
```

Inheritance diagram for EnsembleClustering::QualityMeasure:



Collaboration diagram for EnsembleClustering::QualityMeasure:



Public Member Functions

- [QualityMeasure](#) ([Graph](#) &[G](#))
- virtual [~QualityMeasure](#) ()
- virtual double [getQuality](#) ([Clustering](#) &zeta)=0

Protected Attributes

- [Graph](#) * [G](#)

6.27.1 Detailed Description

Abstract base class for all clustering quality measures.

Definition at line 18 of file [QualityMeasure.h](#).

6.27.2 Constructor & Destructor Documentation

6.27.2.1 EnsembleClustering::QualityMeasure::QualityMeasure (Graph & G)

Definition at line 12 of file QualityMeasure.cpp.

6.27.2.2 EnsembleClustering::QualityMeasure::~~QualityMeasure () [virtual]

Definition at line 16 of file QualityMeasure.cpp.

6.27.3 Member Function Documentation

6.27.3.1 virtual double EnsembleClustering::QualityMeasure::getQuality (Clustering & zeta) [pure virtual]

Implemented in [EnsembleClustering::Modularity](#).

6.27.4 Member Data Documentation

6.27.4.1 Graph* EnsembleClustering::QualityMeasure::G [protected]

Definition at line 22 of file QualityMeasure.h.

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

- [src/clustering/QualityMeasure.h](#)
- [src/clustering/QualityMeasure.cpp](#)

6.28 RandomProbability Class Reference

```
#include <RandomProbability.h>
```

Public Member Functions

- [RandomProbability](#) ()
- virtual [~RandomProbability](#) ()
- virtual double [generate](#) ()

Protected Attributes

- std::uniform_real_distribution
< double > [uniform](#)
- std::default_random_engine [randomEngine](#)

6.28.1 Detailed Description

Definition at line 13 of file RandomProbability.h.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 RandomProbability::RandomProbability ()

Definition at line 10 of file RandomProbability.cpp.

6.28.2.2 RandomProbability::~~RandomProbability () [virtual]

Definition at line 15 of file RandomProbability.cpp.

6.28.3 Member Function Documentation

6.28.3.1 double RandomProbability::generate () [virtual]

Definition at line 19 of file RandomProbability.cpp.

6.28.4 Member Data Documentation

6.28.4.1 std::default_random_engine RandomProbability::randomEngine [protected]

Definition at line 18 of file RandomProbability.h.

6.28.4.2 std::uniform_real_distribution<double> RandomProbability::uniform [protected]

Definition at line 17 of file RandomProbability.h.

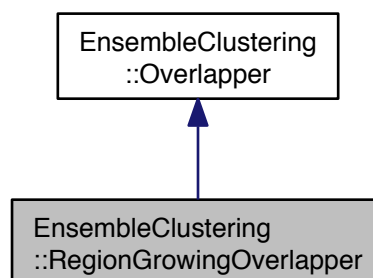
The documentation for this class was generated from the following files:

- [src/aux/RandomProbability.h](#)
- [src/aux/RandomProbability.cpp](#)

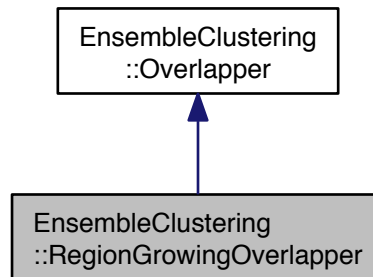
6.29 EnsembleClustering::RegionGrowingOverlapper Class Reference

```
#include <RegionGrowingOverlapper.h>
```

Inheritance diagram for EnsembleClustering::RegionGrowingOverlapper:



Collaboration diagram for EnsembleClustering::RegionGrowingOverlapper:



Public Member Functions

- [RegionGrowingOverlapper\(\)](#)
- virtual [~RegionGrowingOverlapper\(\)](#)

6.29.1 Detailed Description

Definition at line 15 of file RegionGrowingOverlapper.h.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 EnsembleClustering::RegionGrowingOverlapper::RegionGrowingOverlapper ()

Definition at line 12 of file RegionGrowingOverlapper.cpp.

6.29.2.2 EnsembleClustering::RegionGrowingOverlapper::~~RegionGrowingOverlapper () [virtual]

Definition at line 17 of file RegionGrowingOverlapper.cpp.

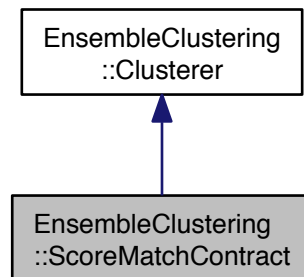
The documentation for this class was generated from the following files:

- [src/overlap/RegionGrowingOverlapper.h](#)
- [src/overlap/RegionGrowingOverlapper.cpp](#)

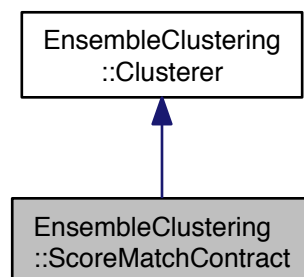
6.30 EnsembleClustering::ScoreMatchContract Class Reference

```
#include <ScoreMatchContract.h>
```

Inheritance diagram for EnsembleClustering::ScoreMatchContract:



Collaboration diagram for EnsembleClustering::ScoreMatchContract:



Public Member Functions

- [ScoreMatchContract\(\)](#)
- virtual [~ScoreMatchContract\(\)](#)

6.30.1 Detailed Description

Definition at line 15 of file ScoreMatchContract.h.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 EnsembleClustering::ScoreMatchContract::ScoreMatchContract ()

Definition at line 12 of file ScoreMatchContract.cpp.

6.30.2.2 EnsembleClustering::ScoreMatchContract::~~ScoreMatchContract () [virtual]

Definition at line 17 of file ScoreMatchContract.cpp.

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

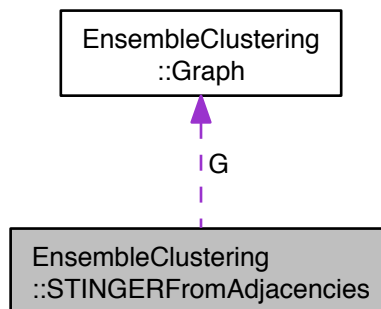
- src/clustering/ScoreMatchContract.h
- src/clustering/ScoreMatchContract.cpp

6.31 EnsembleClustering::STINGERFromAdjacencies Class Reference

A 'builder' which constructs a STINGER-based graph from adjacencies.

```
#include <STINGERFromAdjacencies.h>
```

Collaboration diagram for EnsembleClustering::STINGERFromAdjacencies:



Public Member Functions

- [STINGERFromAdjacencies](#) ()
- virtual [~STINGERFromAdjacencies](#) ()
- virtual void [createGraph](#) ()
Create new STINGER instance.
- virtual void [addAdjacencies](#) (std::vector< [node](#) > adj)
Add next node and its adjacent edges.
- virtual stinger * [getSTINGER](#) ()
- virtual [Graph](#) * [getGraph](#) ()

Protected Attributes

- [Graph](#) * G
- [node](#) currentNode

6.31.1 Detailed Description

A 'builder' which constructs a STINGER-based graph from adjacencies.

An adjacency is a collection of node ids which represent a new node as well as its incident edges.

Definition at line 26 of file STINGERFromAdjacencies.h.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 EnsembleClustering::STINGERFromAdjacencies::STINGERFromAdjacencies ()

Definition at line 18 of file STINGERFromAdjacencies.cpp.

6.31.2.2 EnsembleClustering::STINGERFromAdjacencies::~~STINGERFromAdjacencies () [virtual]

Definition at line 22 of file STINGERFromAdjacencies.cpp.

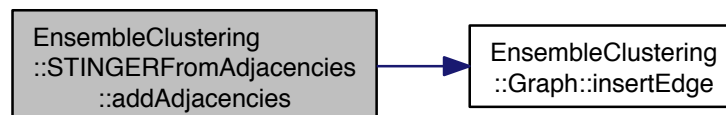
6.31.3 Member Function Documentation

6.31.3.1 void EnsembleClustering::STINGERFromAdjacencies::addAdjacencies (std::vector< node > adj) [virtual]

Add next node and its adjacent edges.

Definition at line 31 of file STINGERFromAdjacencies.cpp.

Here is the call graph for this function:



6.31.3.2 void EnsembleClustering::STINGERFromAdjacencies::createGraph () [virtual]

Create new STINGER instance.

Definition at line 26 of file STINGERFromAdjacencies.cpp.

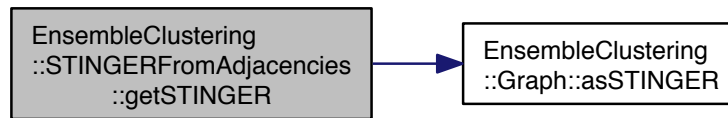
6.31.3.3 Graph * EnsembleClustering::STINGERFromAdjacencies::getGraph () [virtual]

Definition at line 47 of file STINGERFromAdjacencies.cpp.

6.31.3.4 stinger * EnsembleClustering::STINGERFromAdjacencies::getSTINGER () [virtual]

Definition at line 43 of file STINGERFromAdjacencies.cpp.

Here is the call graph for this function:



6.31.4 Member Data Documentation

6.31.4.1 `node EnsembleClustering::STINGERFromAdjacencies::currentNode` [protected]

Definition at line 54 of file STINGERFromAdjacencies.h.

6.31.4.2 `Graph* EnsembleClustering::STINGERFromAdjacencies::G` [protected]

Definition at line 50 of file STINGERFromAdjacencies.h.

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

- [src/input/STINGERFromAdjacencies.h](#)
- [src/input/STINGERFromAdjacencies.cpp](#)

6.32 Timer Class Reference

TODO: Platform-agnostic timer class.

```
#include <Timer.h>
```

Public Member Functions

- [Timer](#) ()
- virtual [~Timer](#) ()

6.32.1 Detailed Description

TODO: Platform-agnostic timer class.

Definition at line 42 of file Timer.h.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 `Timer::Timer ()`

Definition at line 11 of file Timer.cpp.

6.32.2.2 `Timer::~Timer ()` [virtual]

Definition at line 16 of file `Timer.cpp`.

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

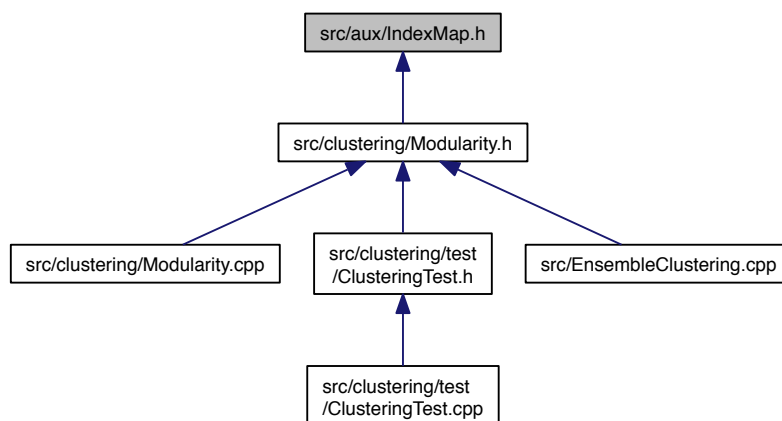
- `src/aux/Timer.h`
- `src/aux/Timer.cpp`

Chapter 7

File Documentation

7.1 src/aux/IndexMap.h File Reference

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



Classes

- class [EnsembleClustering::IndexMap< I, T >](#)

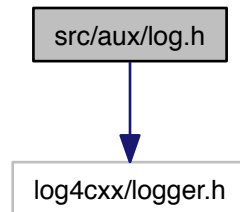
An [IndexMap](#) implements a 1-based mapping from an integer index type to an arbitrary value type.

Namespaces

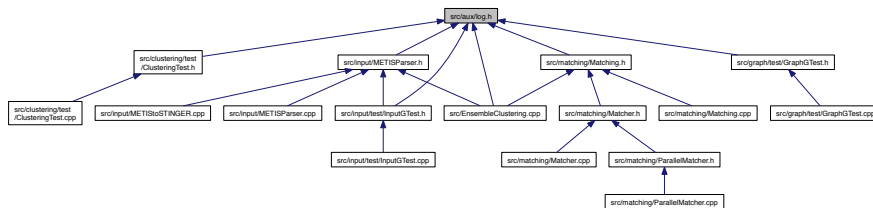
- namespace [EnsembleClustering](#)

7.2 src/aux/log.h File Reference

```
#include "log4cxx/logger.h"
Include dependency graph for log.h:
```



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



Macros

- `#define LOCATION "in " << __PRETTY_FUNCTION__ << ": "`
- `#define LOGGER log4cxx::Logger::getRootLogger()`
- `#define FATAL(X) LOG4CXX_FATAL(LOGGER, LOCATION << X)`
- `#define ERROR(X) LOG4CXX_ERROR(LOGGER, LOCATION << X)`
- `#define WARN(X) LOG4CXX_WARN(LOGGER, LOCATION << X)`
- `#define INFO(X) LOG4CXX_INFO(LOGGER, LOCATION << X)`
- `#define DEBUG(X) LOG4CXX_DEBUG(LOGGER, LOCATION << X);`
- `#define TRACE(X) LOG4CXX_TRACE(LOGGER, LOCATION << X)`

7.2.1 Macro Definition Documentation

7.2.1.1 `#define DEBUG(X) LOG4CXX_DEBUG(LOGGER, LOCATION << X);`

Definition at line 23 of file log.h.

7.2.1.2 `#define ERROR(X) LOG4CXX_ERROR(LOGGER, LOCATION << X)`

Definition at line 20 of file log.h.

7.2.1.3 `#define FATAL(X) LOG4CXX_FATAL(LOGGER, LOCATION << X)`

Definition at line 19 of file log.h.

7.2.1.4 `#define INFO(X) LOG4CXX_INFO(LOGGER, LOCATION << X)`

Definition at line 22 of file log.h.

7.2.1.5 `#define LOCATION "in" << __PRETTY_FUNCTION__ << ": "`

Definition at line 14 of file log.h.

7.2.1.6 `#define LOGGER log4cxx::Logger::getRootLogger()`

Definition at line 15 of file log.h.

7.2.1.7 `#define TRACE(X) LOG4CXX_TRACE(LOGGER, LOCATION << X)`

Definition at line 24 of file log.h.

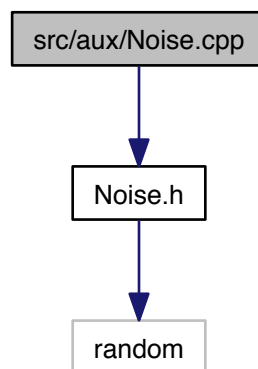
7.2.1.8 `#define WARN(X) LOG4CXX_WARN(LOGGER, LOCATION << X)`

Definition at line 21 of file log.h.

7.3 src/aux/Noise.cpp File Reference

```
#include "Noise.h"
```

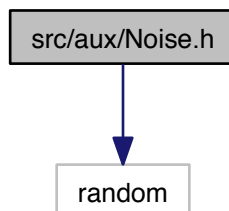
Include dependency graph for Noise.cpp:



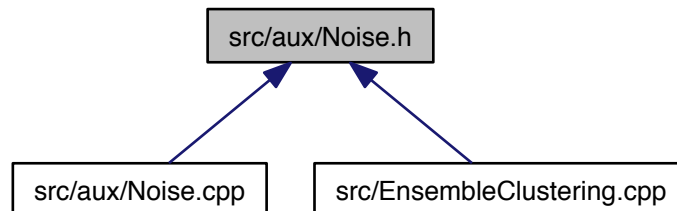
7.4 src/aux/Noise.h File Reference

```
#include <random>
```

Include dependency graph for Noise.h:



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



Classes

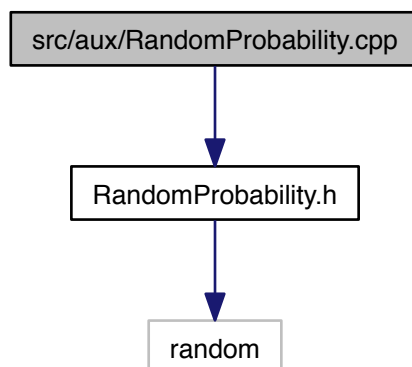
- class [Noise](#)

[Noise](#) is random addition to a signal.

7.5 src/aux/RandomProbability.cpp File Reference

```
#include "RandomProbability.h"
```

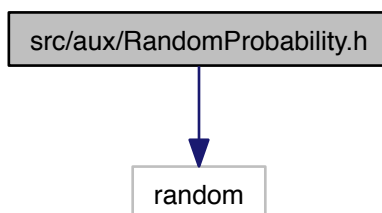
Include dependency graph for RandomProbability.cpp:



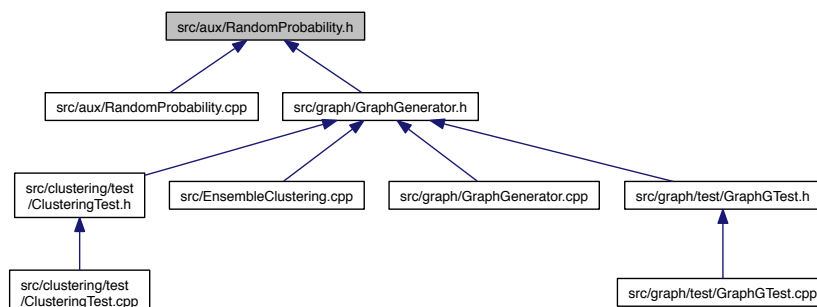
7.6 src/aux/RandomProbability.h File Reference

```
#include <random>
```

Include dependency graph for RandomProbability.h:



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



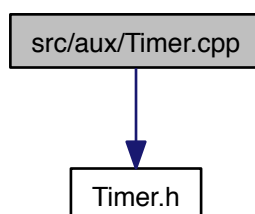
Classes

- class [RandomProbability](#)

7.7 src/aux/Timer.cpp File Reference

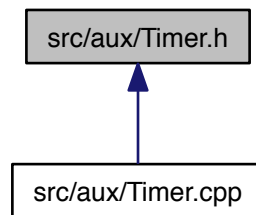
```
#include "Timer.h"
```

Include dependency graph for Timer.cpp:



7.8 src/aux/Timer.h File Reference

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



Classes

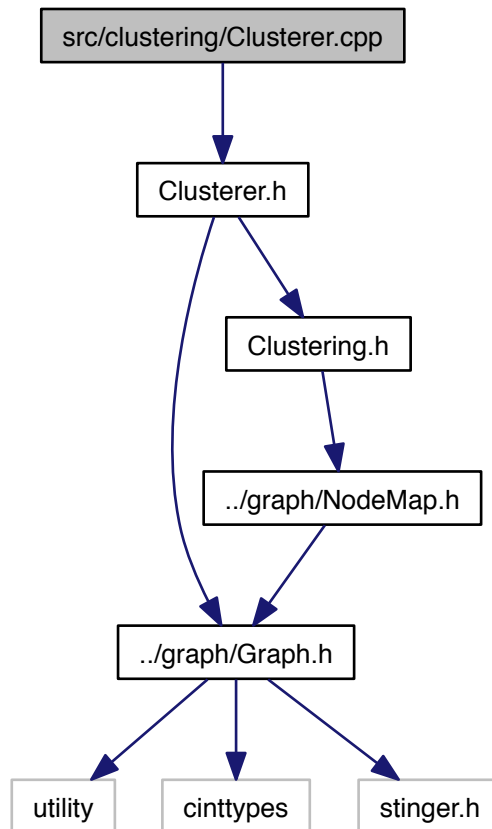
- class [Timer](#)

TODO: Platform-agnostic timer class.

7.9 src/clustering/Clusterer.cpp File Reference

```
#include "Clusterer.h"
```

Include dependency graph for Clusterer.cpp:



Namespaces

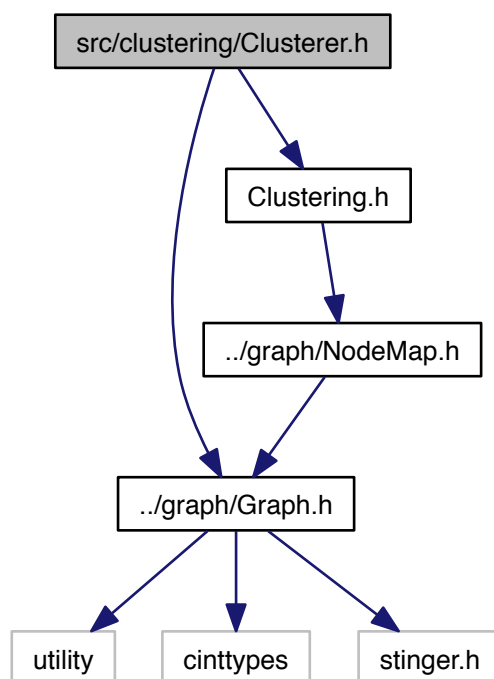
- namespace [EnsembleClustering](#)

7.10 src/clustering/Clusterer.h File Reference

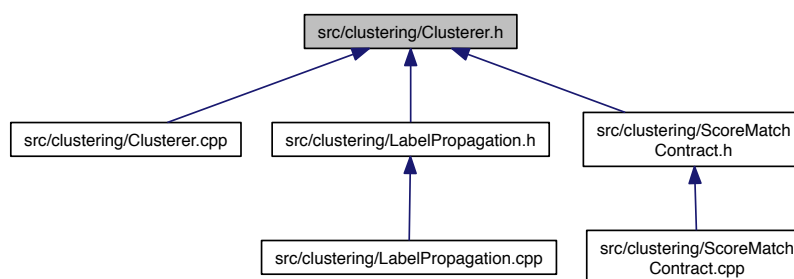
```
#include "../graph/Graph.h"
```

```
#include "Clustering.h"
```

Include dependency graph for Clusterer.h:



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



Classes

- class [EnsembleClustering::Clusterer](#)

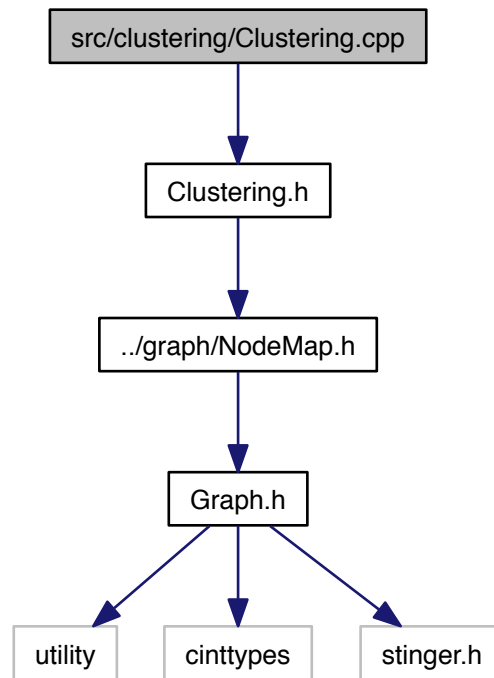
Namespaces

- namespace [EnsembleClustering](#)

7.11 src/clustering/Clustering.cpp File Reference

```
#include "Clustering.h"
```

Include dependency graph for Clustering.cpp:



Namespaces

- namespace [EnsembleClustering](#)

7.12 src/clustering/Clustering.h File Reference

```
#include "../graph/NodeMap.h"
```

```
graph TD; A[src/clustering/Clustering.h] --> B[../graph/NodeMap.h]; B --> C[Graph.h]; C --> D[utility]; C --> E[cinttypes]; C --> F[stinger.h];
```

- class EnsembleClustering::Clustering

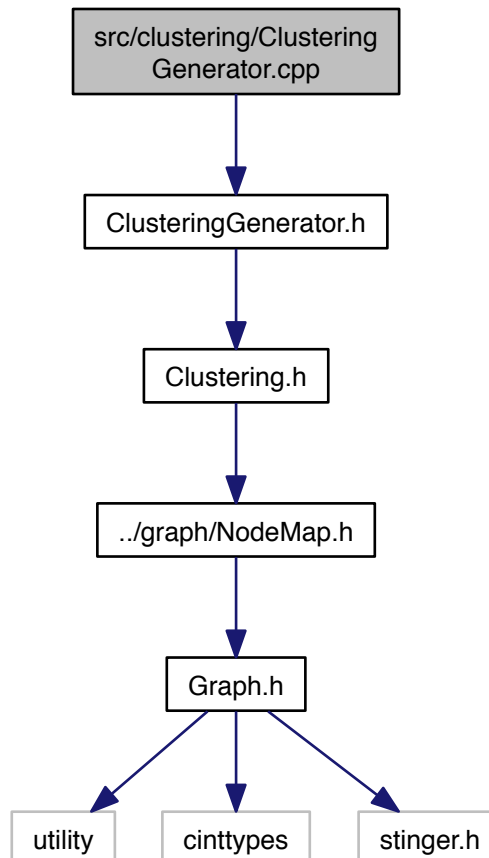
- namespace **EnsembleClustering**

- typedef int64_t EnsembleClustering::cluster
cluster is represented as a 1-based index

7.13 src/clustering/ClusteringGenerator.cpp File Reference

```
#include "ClusteringGenerator.h"
```

Include dependency graph for ClusteringGenerator.cpp:



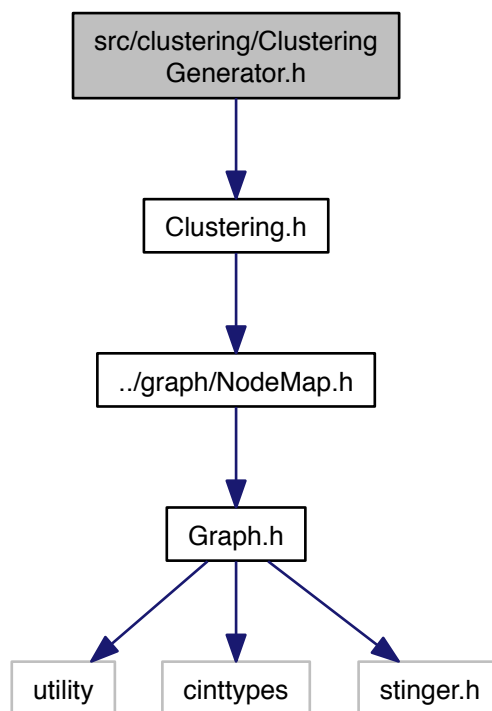
Namespaces

- namespace [EnsembleClustering](#)

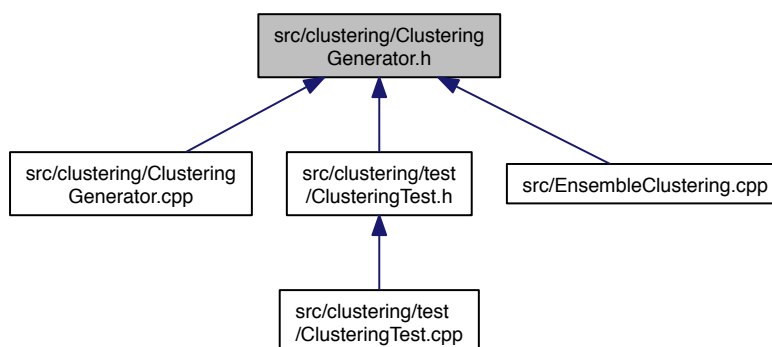
7.14 src/clustering/ClusteringGenerator.h File Reference

```
#include "Clustering.h"
```

Include dependency graph for ClusteringGenerator.h:



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



Classes

- class [EnsembleClustering::ClusteringGenerator](#)

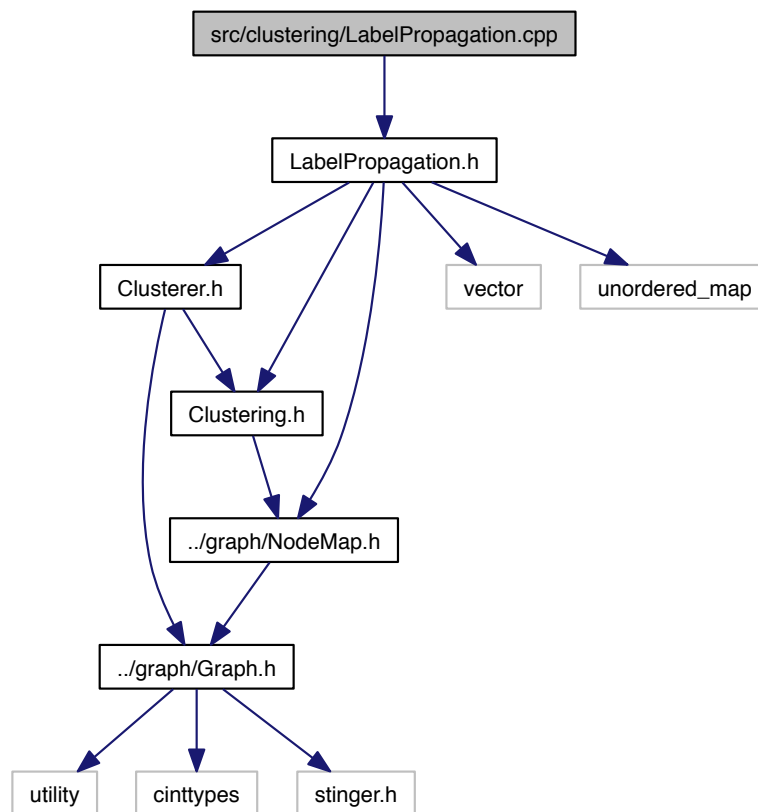
Namespaces

- namespace [EnsembleClustering](#)

7.15 src/clustering/LabelPropagation.cpp File Reference

```
#include "LabelPropagation.h"
```

Include dependency graph for LabelPropagation.cpp:



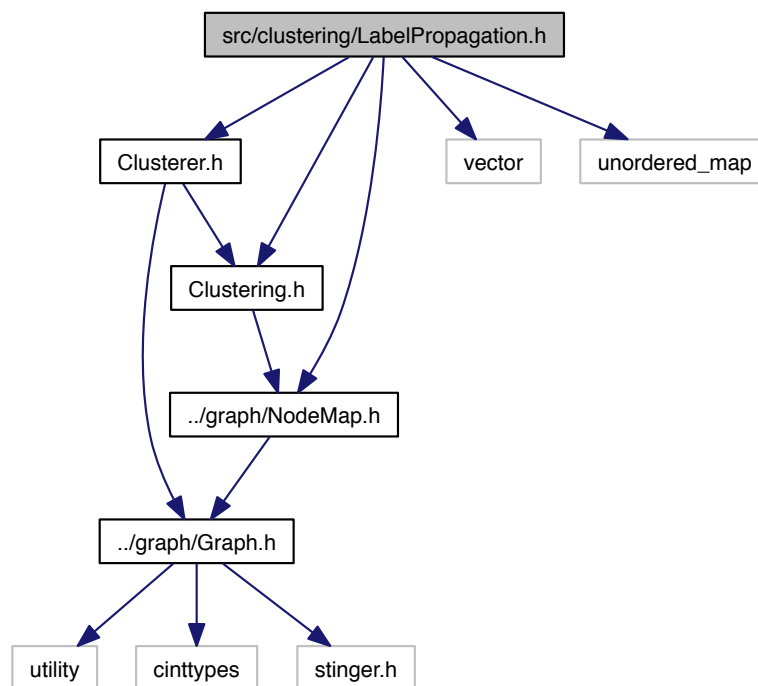
Namespaces

- namespace [EnsembleClustering](#)

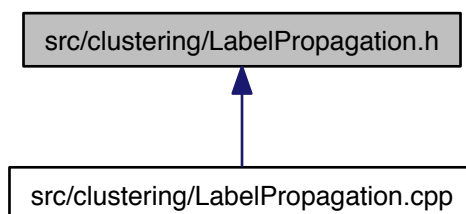
7.16 src/clustering/LabelPropagation.h File Reference

```
#include "Clusterer.h"
#include "Clustering.h"
#include <vector>
#include <unordered_map>
#include "../graph/NodeMap.h"
```


Include dependency graph for LabelPropagation.h:



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



Classes

- class [EnsembleClustering::LabelPropagation](#)

As described in Ovelgoenne et al: An Ensemble Learning Strategy for [Graph Clustering](#) Raghavan et al.

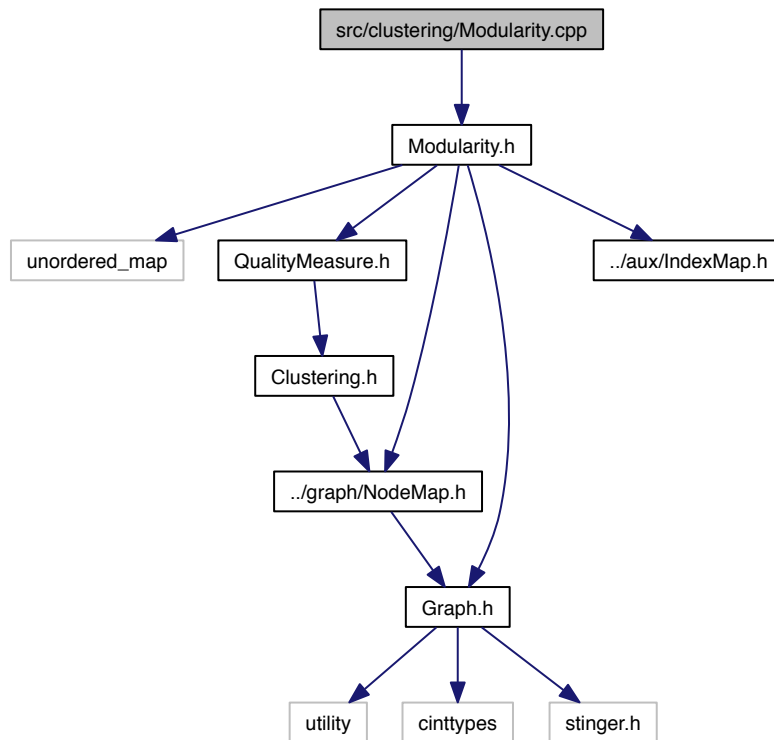
Namespaces

- namespace [EnsembleClustering](#)

7.17 src/clustering/Modularity.cpp File Reference

```
#include "Modularity.h"
```

Include dependency graph for Modularity.cpp:



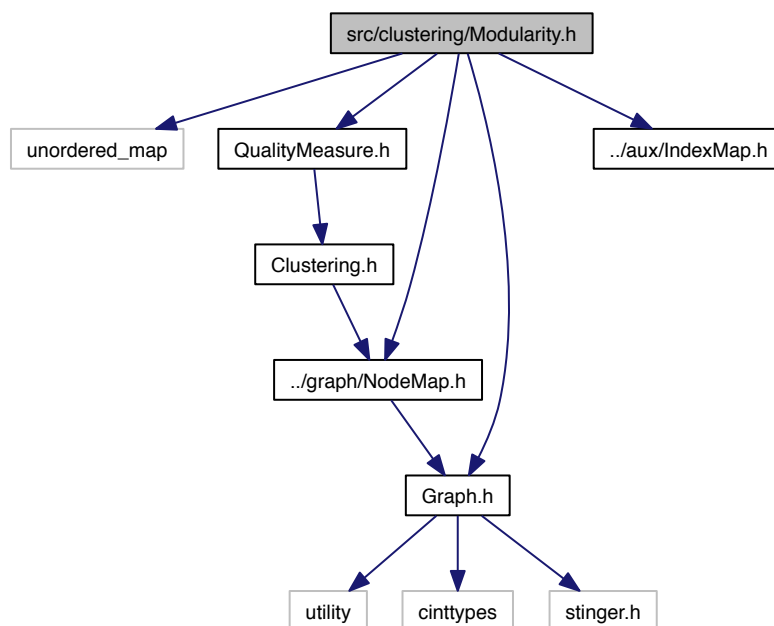
Namespaces

- namespace [EnsembleClustering](#)

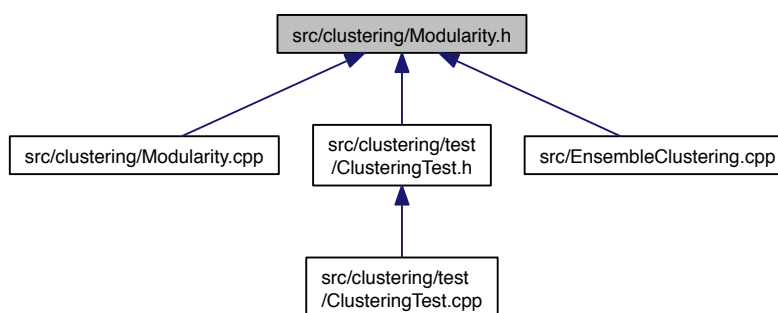
7.18 src/clustering/Modularity.h File Reference

```
#include <unordered_map>
#include "QualityMeasure.h"
#include "../aux/IndexMap.h"
#include "../graph/Graph.h"
#include "../graph/NodeMap.h"
```

Include dependency graph for Modularity.h:



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



Classes

- class [EnsembleClustering::Modularity](#)

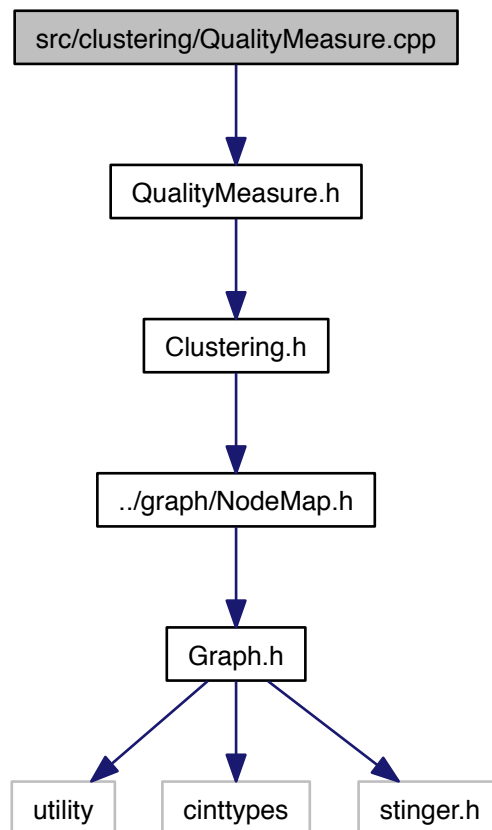
Namespaces

- namespace [EnsembleClustering](#)

7.19 src/clustering/QualityMeasure.cpp File Reference

```
#include "QualityMeasure.h"
```

Include dependency graph for QualityMeasure.cpp:



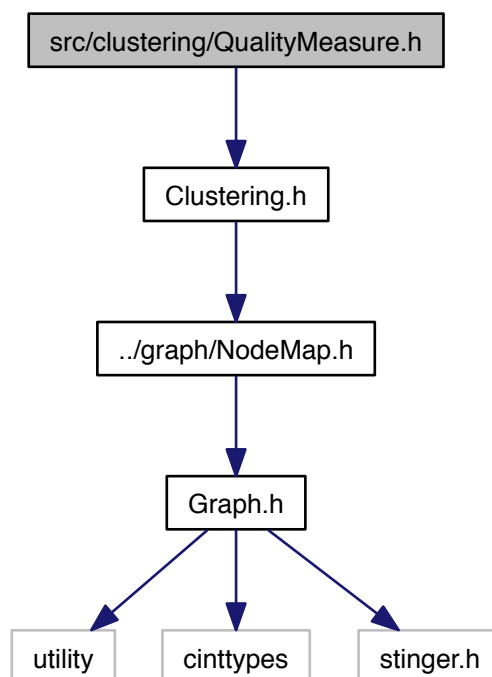
Namespaces

- namespace [EnsembleClustering](#)

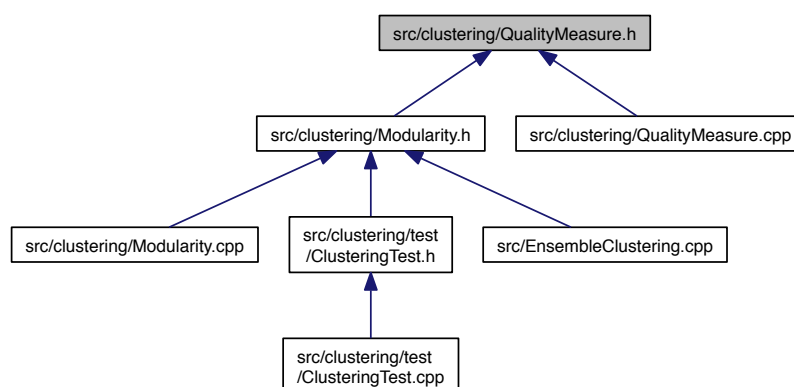
7.20 src/clustering/QualityMeasure.h File Reference

```
#include "Clustering.h"
```

Include dependency graph for QualityMeasure.h:



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



Classes

- class [EnsembleClustering::QualityMeasure](#)

Abstract base class for all clustering quality measures.

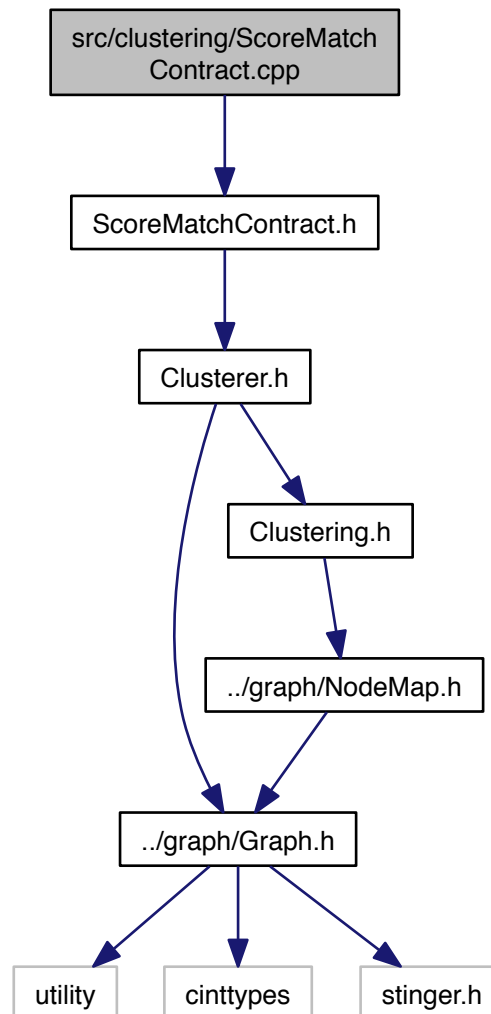
Namespaces

- namespace [EnsembleClustering](#)

7.21 src/clustering/ScoreMatchContract.cpp File Reference

```
#include "ScoreMatchContract.h"
```

Include dependency graph for ScoreMatchContract.cpp:



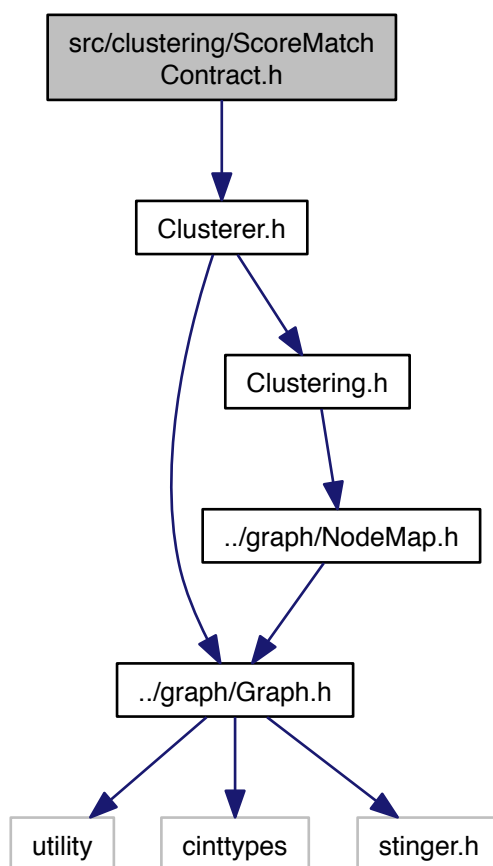
Namespaces

- namespace [EnsembleClustering](#)

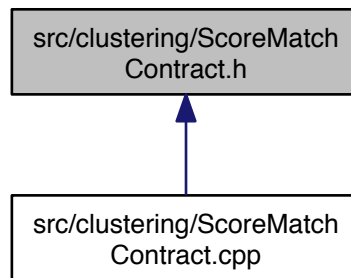
7.22 src/clustering/ScoreMatchContract.h File Reference

```
#include "Clusterer.h"
```

Include dependency graph for ScoreMatchContract.h:



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



Classes

- class [EnsembleClustering::ScoreMatchContract](#)

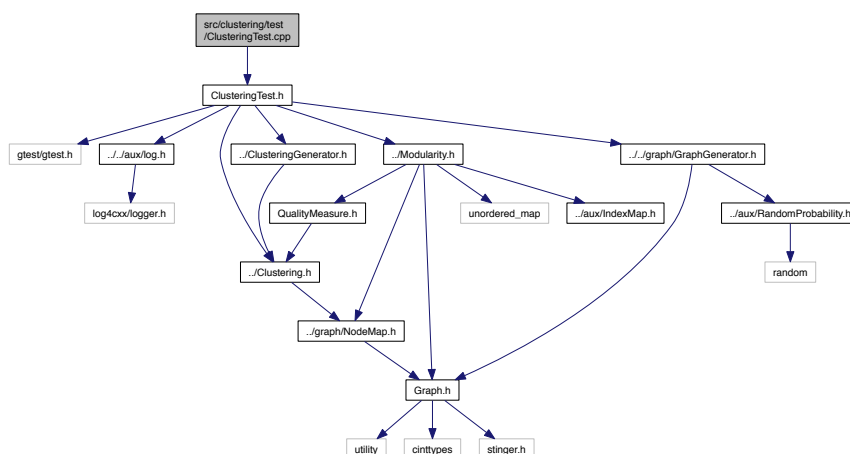
Namespaces

- namespace [EnsembleClustering](#)

7.23 src/clustering/test/ClusteringTest.cpp File Reference

```
#include "ClusteringTest.h"
```

Include dependency graph for `ClusteringTest.cpp`:

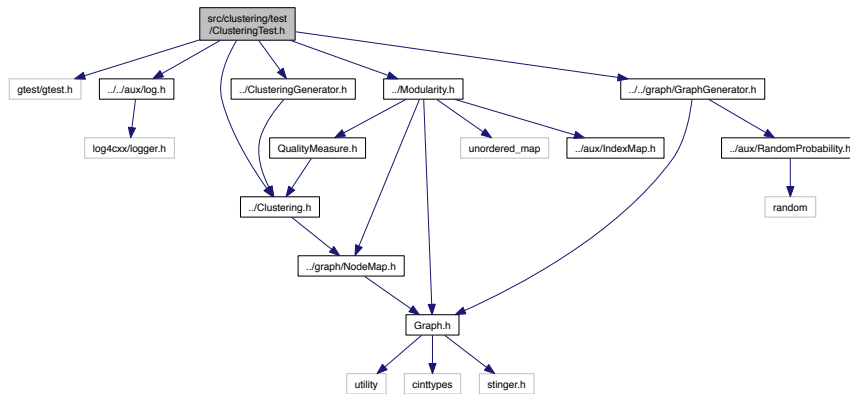


Namespaces

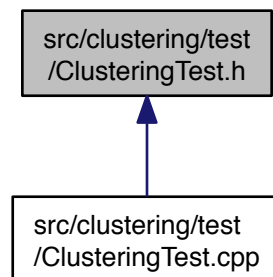
- namespace [EnsembleClustering](#)

7.24 src/clustering/test/ClusteringTest.h File Reference

```
#include <gtest/gtest.h>
#include "../aux/log.h"
#include "../Clustering.h"
#include "../Modularity.h"
#include "../ClusteringGenerator.h"
#include "../../graph/GraphGenerator.h"
Include dependency graph for ClusteringTest.h:
```



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



Classes

- class [EnsembleClustering::ClusteringTest](#)

Namespaces

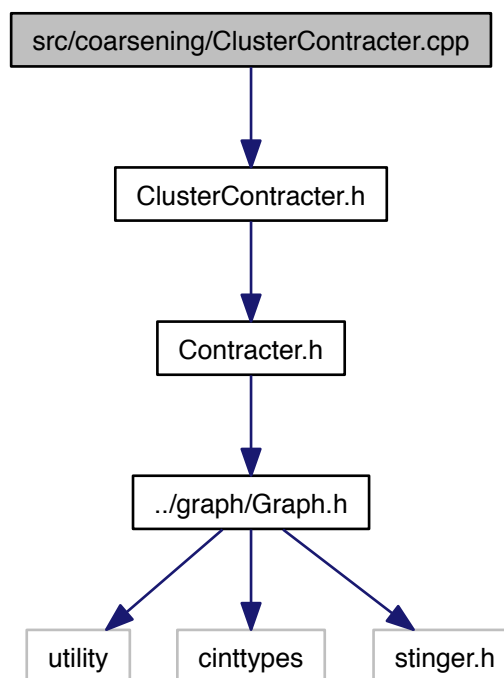
- namespace [EnsembleClustering](#)

Functions

- [EnsembleClustering::TEST_F](#) (ClusteringTest, testModularity)

7.25 src/coarsening/ClusterContracter.cpp File Reference

```
#include "ClusterContracter.h"  
Include dependency graph for ClusterContracter.cpp:
```



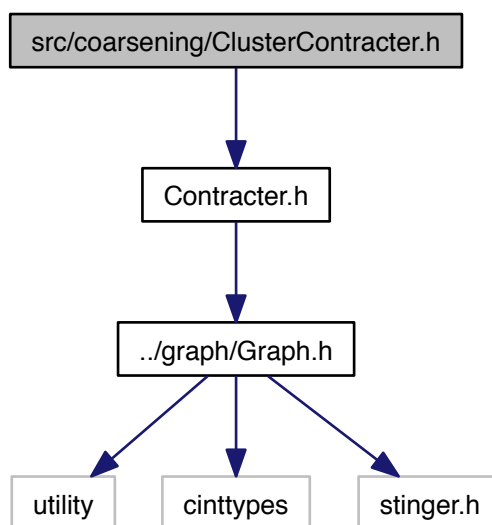
Namespaces

- namespace [EnsembleClustering](#)

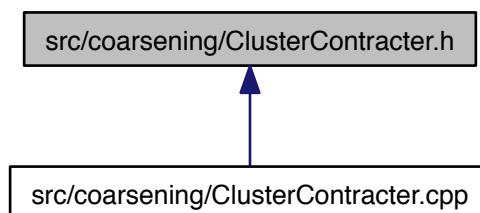
7.26 src/coarsening/ClusterContracter.h File Reference

```
#include "Contracter.h"
```

Include dependency graph for ClusterContracter.h:



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



Classes

- class [EnsembleClustering::ClusterContracter](#)

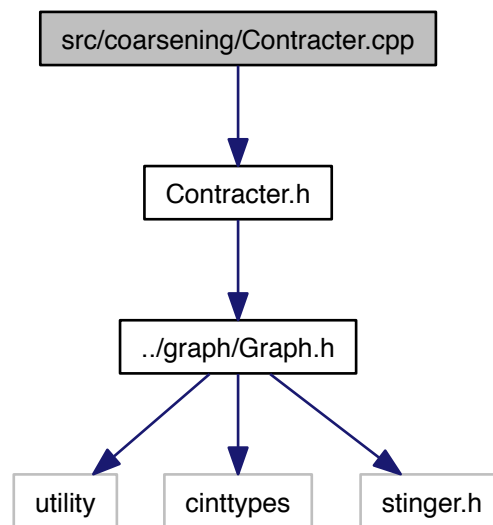
Namespaces

- namespace [EnsembleClustering](#)

7.27 src/coarsening/Contracter.cpp File Reference

```
#include "Contracter.h"
```

Include dependency graph for Contractor.cpp:



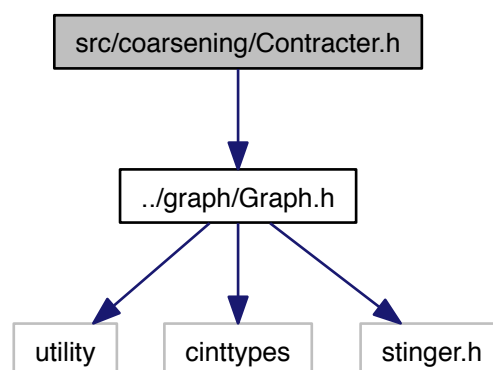
Namespaces

- namespace [EnsembleClustering](#)

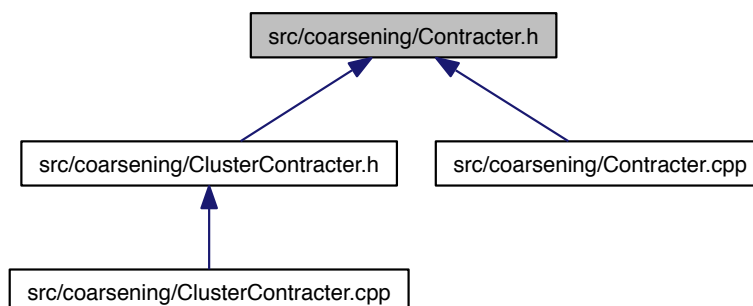
7.28 src/coarsening/Contractor.h File Reference

```
#include "../graph/Graph.h"
```

Include dependency graph for Contractor.h:



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



Classes

- class [EnsembleClustering::Contracter](#)

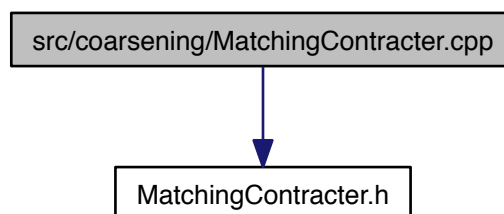
Namespaces

- namespace [EnsembleClustering](#)

7.29 src/coarsening/MatchingContracter.cpp File Reference

```
#include "MatchingContracter.h"
```

Include dependency graph for MatchingContracter.cpp:

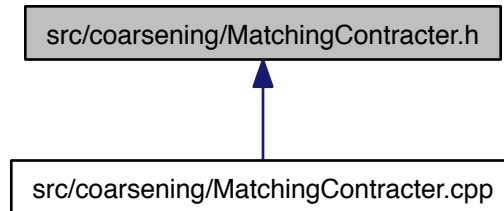


Namespaces

- namespace [EnsembleClustering](#)

7.30 src/coarsening/MatchingContracter.h File Reference

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



Classes

- class [EnsembleClustering::MatchingContracter](#)

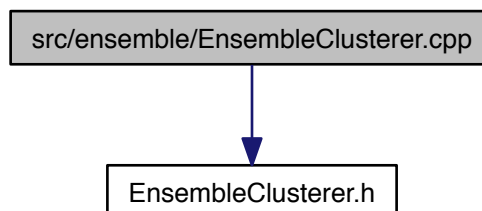
Namespaces

- namespace [EnsembleClustering](#)

7.31 src/ensemble/EnsembleClusterer.cpp File Reference

```
#include "EnsembleClusterer.h"
```

Include dependency graph for `EnsembleClusterer.cpp`:

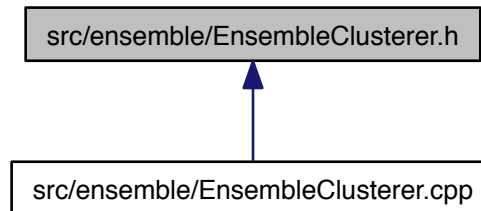


Namespaces

- namespace [EnsembleClustering](#)

7.32 src/ensemble/EnsembleClusterer.h File Reference

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



Classes

- class [EnsembleClustering::EnsembleClusterer](#)

Namespaces

- namespace [EnsembleClustering](#)

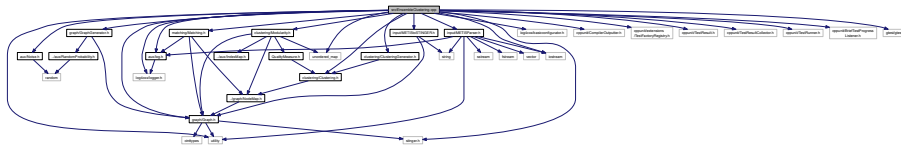
7.33 src/EnsembleClustering.cpp File Reference

```

#include <iostream>
#include <utility>
#include <unordered_map>
#include "log4cxx/logger.h"
#include "log4cxx/basicconfigurator.h"
#include <cppunit/CompilerOutputter.h>
#include <cppunit/extensions/TestFactoryRegistry.h>
#include <cppunit/TestResult.h>
#include <cppunit/TestResultCollector.h>
#include <cppunit/TestRunner.h>
#include <cppunit/BriefTestProgressListener.h>
#include "gtest/gtest.h"
#include "aux/log.h"
#include "aux/Noise.h"
#include "graph/Graph.h"
#include "input/METISParser.h"
#include "input/METISToSTINGER.h"
#include "matching/Matching.h"
#include "clustering/Clustering.h"
#include "clustering/ClusteringGenerator.h"
#include "graph/GraphGenerator.h"
#include "clustering/Modularity.h"
#include "stinger.h"

```

Include dependency graph for EnsembleClustering.cpp:



Functions

- void `testMETIStoSTINGER` ()
- void `testMatching` ()
- `Graph` & `makeCompleteGraph` (int n)
Make a complete graph with n vertices.
- void `configureLogging` ()
Call this first to configure logging output.
- int `main` (int argc, char **argv)

7.33.1 Function Documentation

7.33.1.1 void `configureLogging` ()

Call this first to configure logging output.

Definition at line 112 of file EnsembleClustering.cpp.

7.33.1.2 int `main` (int *argc*, char ** *argv*)

Definition at line 121 of file EnsembleClustering.cpp.

Here is the call graph for this function:

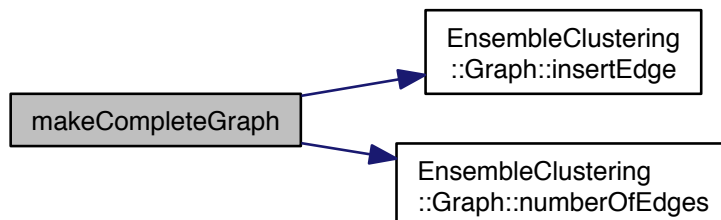


7.33.1.3 `Graph`& `makeCompleteGraph` (int *n*)

Make a complete graph with n vertices.

Definition at line 93 of file EnsembleClustering.cpp.

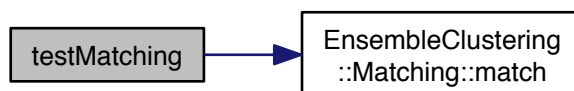
Here is the call graph for this function:



7.33.1.4 void testMatching ()

Definition at line 70 of file EnsembleClustering.cpp.

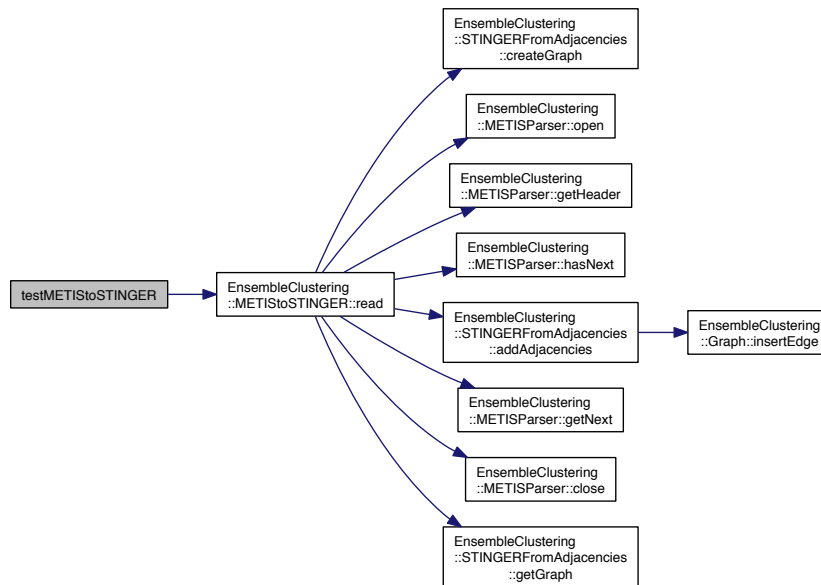
Here is the call graph for this function:



7.33.1.5 void testMETIstoSTINGER ()

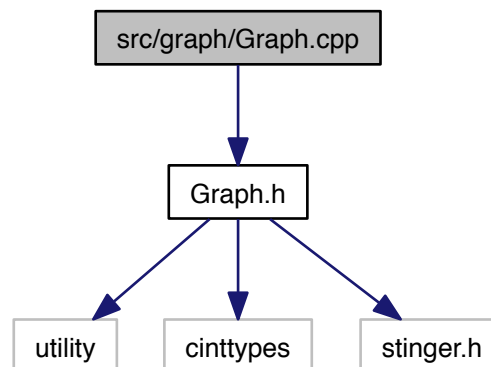
Definition at line 53 of file EnsembleClustering.cpp.

Here is the call graph for this function:



7.34 src/graph/Graph.cpp File Reference

```
#include "Graph.h"
Include dependency graph for Graph.cpp:
```

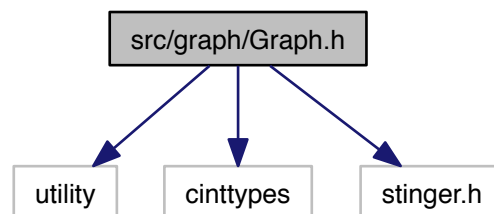


Namespaces

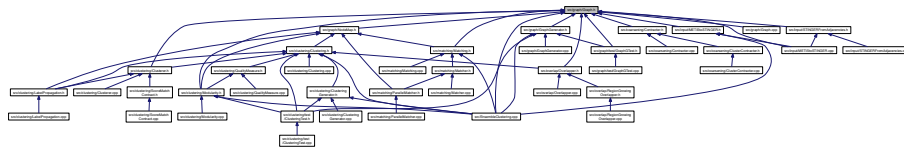
- namespace [EnsembleClustering](#)

7.35 src/graph/Graph.h File Reference

```
#include <utility>
#include <cinttypes>
#include "stinger.h"
Include dependency graph for Graph.h:
```



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



Classes

- class [EnsembleClustering::Graph](#)
Graph interface.

Namespaces

- namespace [EnsembleClustering](#)

Macros

- #define [FORALL_EDGES_BEGIN](#)(G) STINGER_FORALL_EDGES_BEGIN(G.asSTINGER(), G.defaultEdgeType)
Traversal macros.
- #define [FORALL_EDGES_END](#)() STINGER_FORALL_EDGES_END()
- #define [PARALLEL_FORALL_EDGES_BEGIN](#)(G) STINGER_PARALLEL_FORALL_EDGES_BEGIN(G.asSTINGER(), G.defaultEdgeType)
- #define [PARALLEL_FORALL_EDGES_END](#)() STINGER_PARALLEL_FORALL_EDGES_END()
- #define [READ_ONLY_FORALL_EDGES_BEGIN](#)(G) STINGER_READ_ONLY_FORALL_EDGES_BEGIN(G.asSTINGER(), G.defaultEdgeType)
- #define [READ_ONLY_FORALL_EDGES_END](#)() STINGER_READ_ONLY_FORALL_EDGES_END()
- #define [READ_ONLY_PARALLEL_FORALL_EDGES_BEGIN](#)(G) STINGER_READ_ONLY_PARALLEL_FORALL_EDGES_BEGIN(G.asSTINGER(), G.defaultEdgeType)

- `#define READ_ONLY_PARALLEL_FORALL_EDGES_END()` `STINGER_READ_ONLY_PARALLEL_FORALL_EDGES_END()`
- `#define EDGE_SOURCE` `STINGER_EDGE_SOURCE`
- `#define EDGE_DEST` `STINGER_EDGE_DEST`
- `#define FORALL_EDGES_OF_NODE_BEGIN(G, V)` `STINGER_FORALL_EDGES_OF_VTX_BEGIN(G.asSTINGER(), V)`
- `#define FORALL_EDGES_OF_NODE_END()` `STINGER_FORALL_EDGES_OF_VTX_END()`
- `#define READ_ONLY_FORALL_EDGES_OF_NODE_BEGIN(G, V)` `STINGER_READ_ONLY_FORALL_EDGES_OF_VTX_BEGIN(G.asSTINGER(), V)`
- `#define READ_ONLY_FORALL_EDGES_OF_NODE_END()` `STINGER_READ_ONLY_FORALL_EDGES_OF_VTX_END()`

Typedefs

- `typedef int64_t EnsembleClustering::node`
Typedefs.
- `typedef std::pair< node, node > EnsembleClustering::edge`
an undirected edge is a pair of nodes (indices)

7.35.1 Macro Definition Documentation

7.35.1.1 `#define EDGE_DEST` `STINGER_EDGE_DEST`

Definition at line 51 of file Graph.h.

7.35.1.2 `#define EDGE_SOURCE` `STINGER_EDGE_SOURCE`

Definition at line 50 of file Graph.h.

7.35.1.3 `#define FORALL_EDGES_BEGIN(G)` `STINGER_FORALL_EDGES_BEGIN(G.asSTINGER(), G.defaultEdgeType)`

Traversal macros.

These are modified versions of the macros defined in `stinger/include/stinger-traversal.h`

Definition at line 38 of file Graph.h.

7.35.1.4 `#define FORALL_EDGES_END()` `STINGER_FORALL_EDGES_END()`

Definition at line 39 of file Graph.h.

7.35.1.5 `#define FORALL_EDGES_OF_NODE_BEGIN(G, V)` `STINGER_FORALL_EDGES_OF_VTX_BEGIN(G.asSTINGER(), V)`

Definition at line 53 of file Graph.h.

7.35.1.6 `#define FORALL_EDGES_OF_NODE_END()` `STINGER_FORALL_EDGES_OF_VTX_END()`

Definition at line 54 of file Graph.h.

7.35.1.7 `#define PARALLEL_FORALL_EDGES_BEGIN(G)` `STINGER_PARALLEL_FORALL_EDGES_BEGIN(G.asSTINGER(), G.defaultEdgeType)`

Definition at line 41 of file Graph.h.

7.35.1.8 `#define PARALLEL_FORALL_EDGES_END() STINGER_PARALLEL_FORALL_EDGES_END()`

Definition at line 42 of file Graph.h.

7.35.1.9 `#define READ_ONLY_FORALL_EDGES_BEGIN(G) STINGER_READ_ONLY_FORALL_EDGES_BEGIN(G.asSTINGER(),
G.defaultEdgeType)`

Definition at line 44 of file Graph.h.

7.35.1.10 `#define READ_ONLY_FORALL_EDGES_END() STINGER_READ_ONLY_FORALL_EDGES_END()`

Definition at line 45 of file Graph.h.

7.35.1.11 `#define READ_ONLY_FORALL_EDGES_OF_NODE_BEGIN(G, V) STINGER_READ_ONLY_FORALL_EDGES_OF_VTX_-
BEGIN(G.asSTINGER(), V)`

Definition at line 56 of file Graph.h.

7.35.1.12 `#define READ_ONLY_FORALL_EDGES_OF_NODE_END() STINGER_READ_ONLY_FORALL_EDGES_OF_VTX_END()`

Definition at line 57 of file Graph.h.

7.35.1.13 `#define READ_ONLY_PARALLEL_FORALL_EDGES_BEGIN(G) STINGER_READ_ONLY_PARALLEL_FORALL_EDGES-
_BEGIN(G.asSTINGER(), G.defaultEdgeType)`

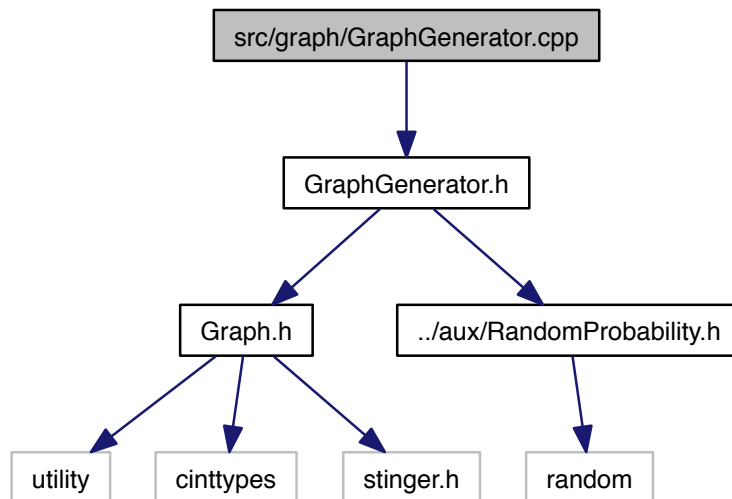
Definition at line 47 of file Graph.h.

7.35.1.14 `#define READ_ONLY_PARALLEL_FORALL_EDGES_END() STINGER_READ_ONLY_PARALLEL_FORALL_EDGES_EN-
D()`

Definition at line 48 of file Graph.h.

7.36 src/graph/GraphGenerator.cpp File Reference

```
#include "GraphGenerator.h"  
Include dependency graph for GraphGenerator.cpp:
```



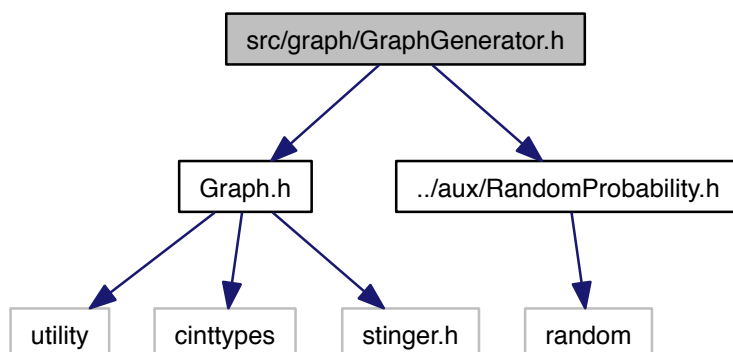
Namespaces

- namespace [EnsembleClustering](#)

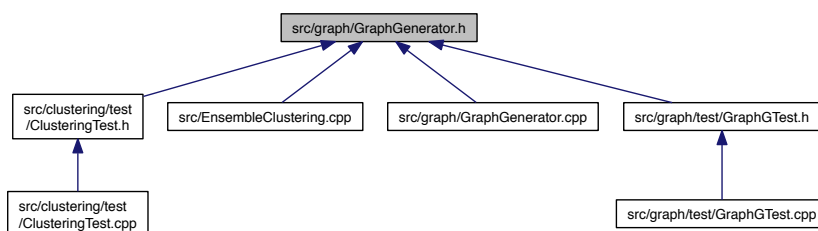
7.37 src/graph/GraphGenerator.h File Reference

```
#include "Graph.h"  
#include "../aux/RandomProbability.h"
```

Include dependency graph for GraphGenerator.h:



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



Classes

- class [EnsembleClustering::GraphGenerator](#)

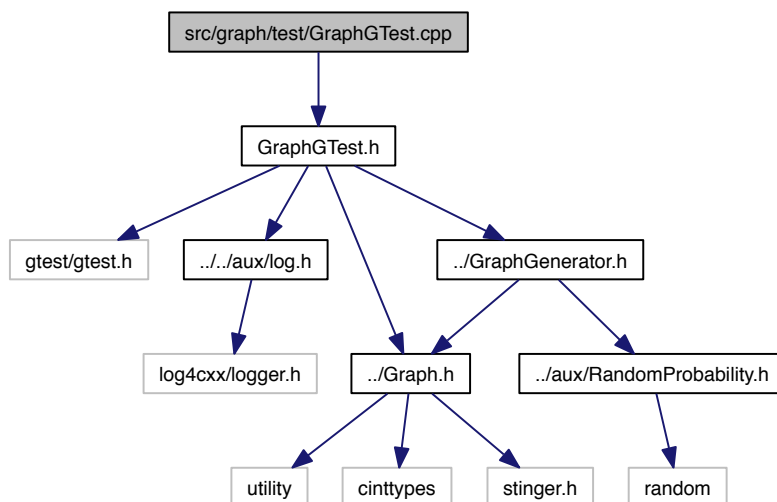
Namespaces

- namespace [EnsembleClustering](#)

7.38 src/graph/NodeMap.h File Reference

```
#include "Graph.h"
```


Include dependency graph for GraphGTest.cpp:



Namespaces

- namespace [EnsembleClustering](#)

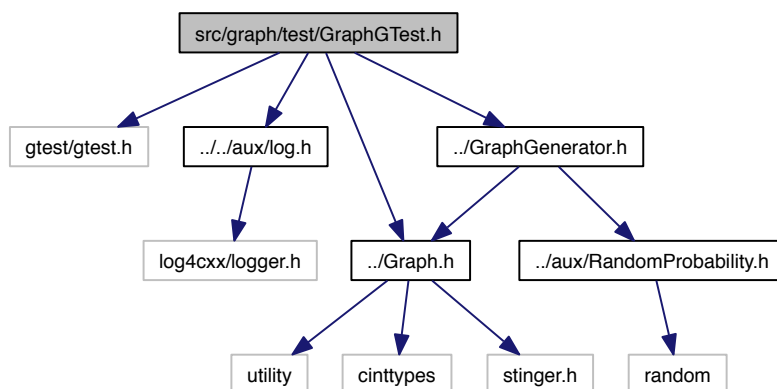
7.40 src/graph/test/GraphGTest.h File Reference

```

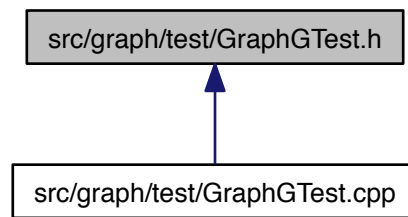
#include <gtest/gtest.h>
#include "../aux/log.h"
#include "../Graph.h"
#include "../GraphGenerator.h"

```

Include dependency graph for GraphGTest.h:



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



Classes

- class [EnsembleClustering::GraphGTest](#)

Namespaces

- namespace [EnsembleClustering](#)

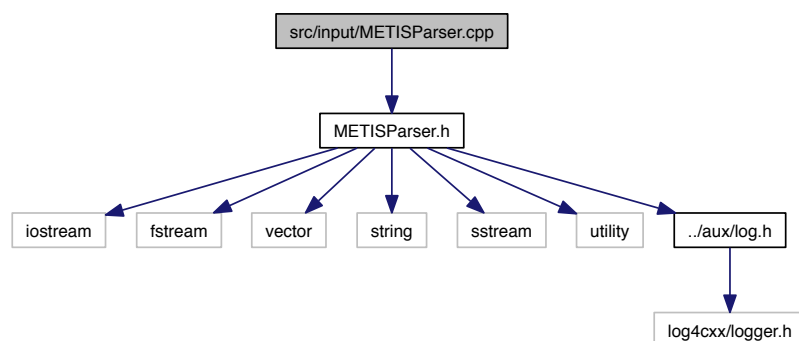
Functions

- [EnsembleClustering::TEST_F](#) (GraphGTest, testIteration)

7.41 src/input/METISParser.cpp File Reference

```
#include "METISParser.h"
```

Include dependency graph for METISParser.cpp:



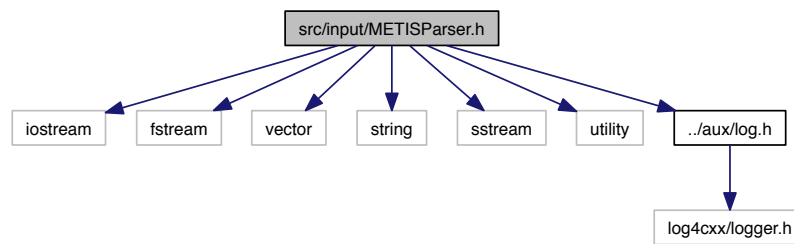
Namespaces

- namespace [EnsembleClustering](#)

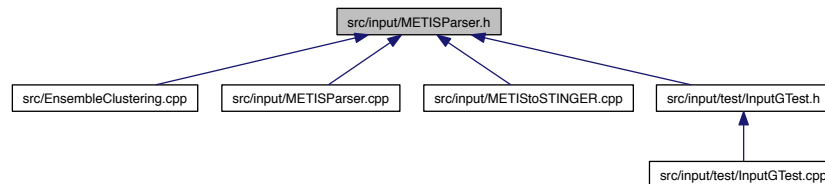
7.42 src/input/METISParser.h File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <sstream>
#include <utility>
#include "../aux/log.h"
```

Include dependency graph for METISParser.h:



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



Classes

- class [EnsembleClustering::METISParser](#)

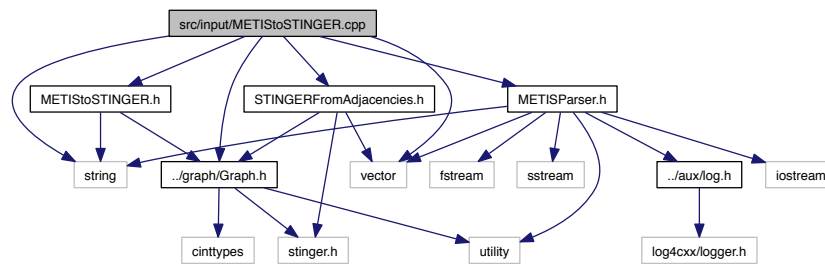
Namespaces

- namespace [EnsembleClustering](#)

7.43 src/input/METISStoSTINGER.cpp File Reference

```
#include "METISStoSTINGER.h"
#include <string>
#include <vector>
#include "../graph/Graph.h"
#include "STINGERFromAdjacencies.h"
#include "METISParser.h"
```

Include dependency graph for METISToSTINGER.cpp:

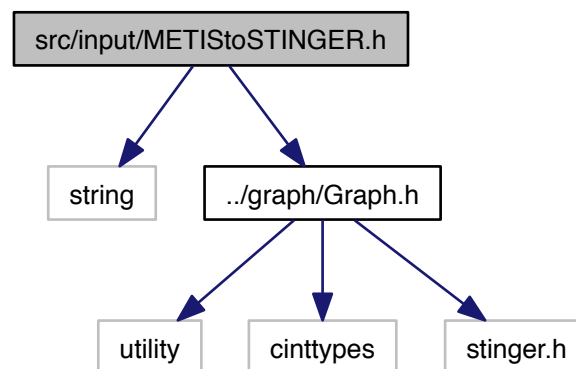


Namespaces

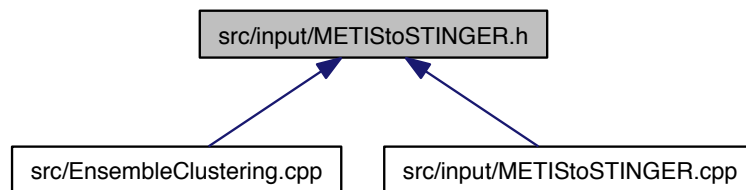
- namespace [EnsembleClustering](#)

7.44 src/input/METISToSTINGER.h File Reference

```
#include <string>
#include "../graph/Graph.h"
Include dependency graph for METISToSTINGER.h:
```



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



Classes

- class [EnsembleClustering::METISToSTINGER](#)

This class provides a user interface for reading a METIS graph file and returning a STINGER-based graph object.

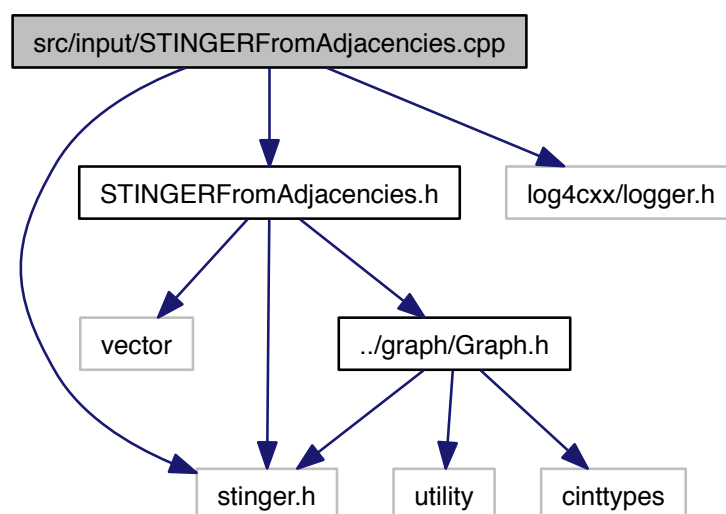
Namespaces

- namespace [EnsembleClustering](#)

7.45 src/input/STINGERFromAdjacencies.cpp File Reference

```
#include "STINGERFromAdjacencies.h"
#include "log4cxx/logger.h"
#include "stinger.h"
```

Include dependency graph for STINGERFromAdjacencies.cpp:



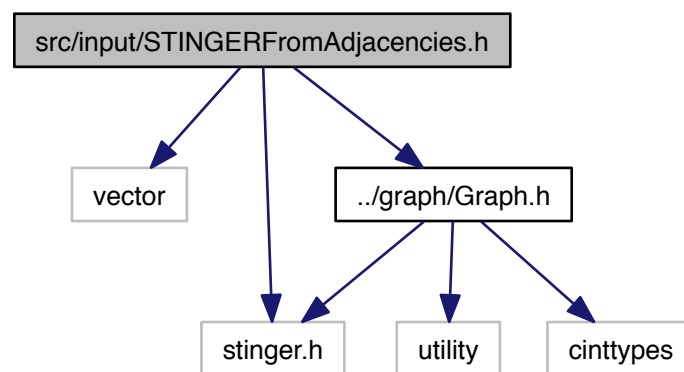
Namespaces

- namespace [EnsembleClustering](#)

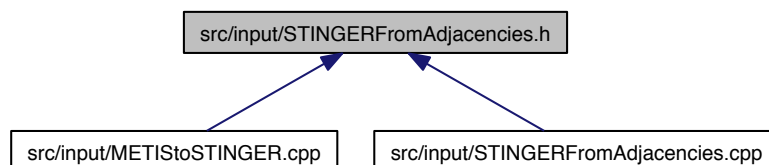
7.46 src/input/STINGERFromAdjacencies.h File Reference

```
#include <vector>
#include "stinger.h"
#include "../graph/Graph.h"
```

Include dependency graph for STINGERFromAdjacencies.h:



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



Classes

- class [EnsembleClustering::STINGERFromAdjacencies](#)
A 'builder' which constructs a *STINGER*-based graph from adjacencies.

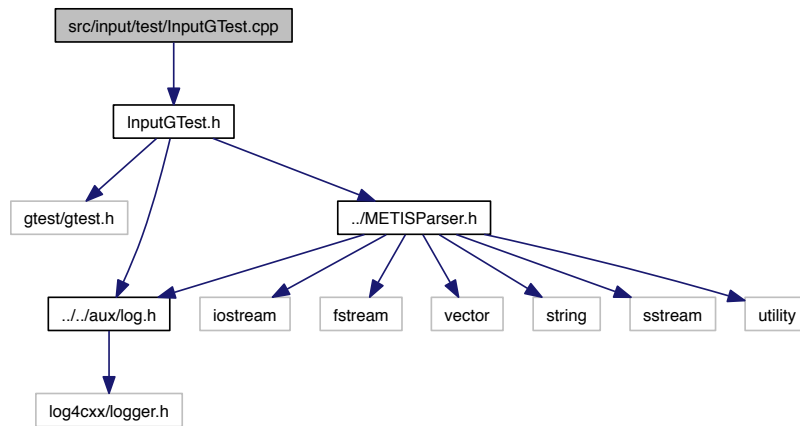
Namespaces

- namespace [EnsembleClustering](#)

7.47 src/input/test/InputGTest.cpp File Reference

```
#include "InputGTest.h"
```

Include dependency graph for InputGTest.cpp:



Namespaces

- namespace [EnsembleClustering](#)

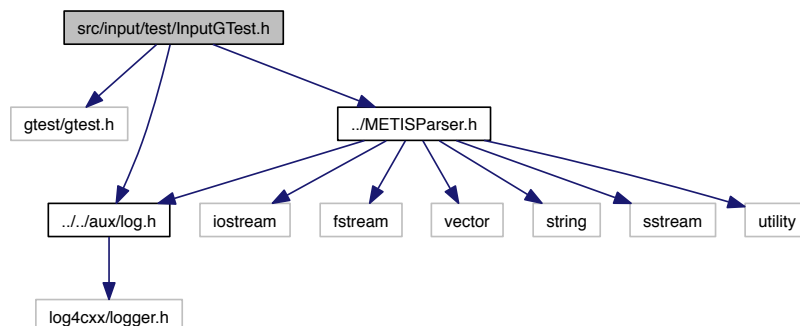
7.48 src/input/test/InputGTest.h File Reference

```
#include <gtest/gtest.h>
```

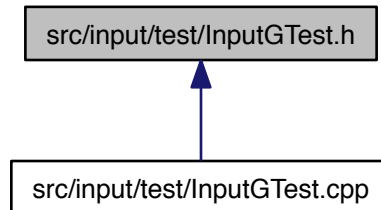
```
#include "../aux/log.h"
```

```
#include "../METISParser.h"
```

Include dependency graph for InputGTest.h:



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



Classes

- class [EnsembleClustering::InputGTest](#)

Namespaces

- namespace [EnsembleClustering](#)

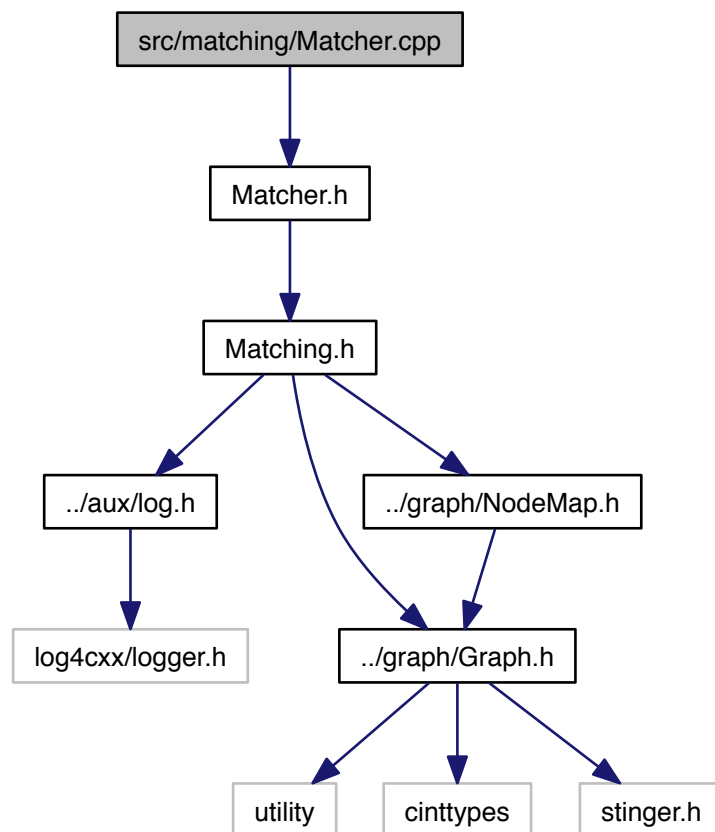
Functions

- [EnsembleClustering::TEST_F](#) (InputGTest, testMETISParser)

7.49 src/matching/Matcher.cpp File Reference

```
#include "Matcher.h"
```


Include dependency graph for Matcher.cpp:



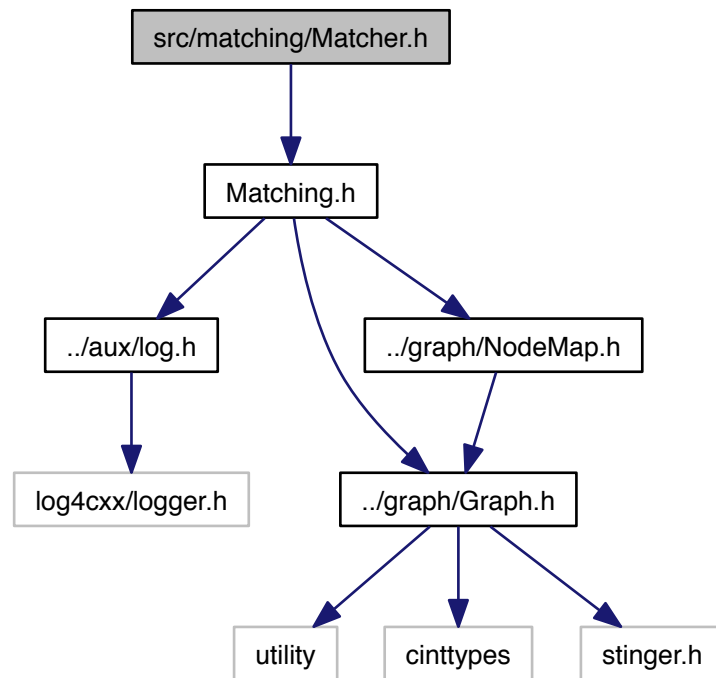
Namespaces

- namespace [EnsembleClustering](#)

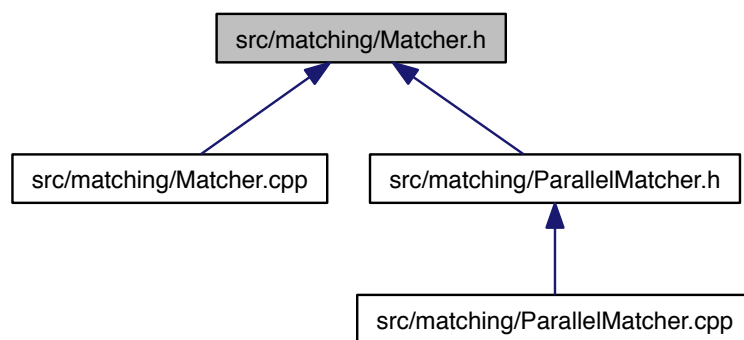
7.50 src/matching/Matcher.h File Reference

```
#include "Matching.h"
```

Include dependency graph for `Matcher.h`:



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



Classes

- class [EnsembleClustering::Matcher](#)

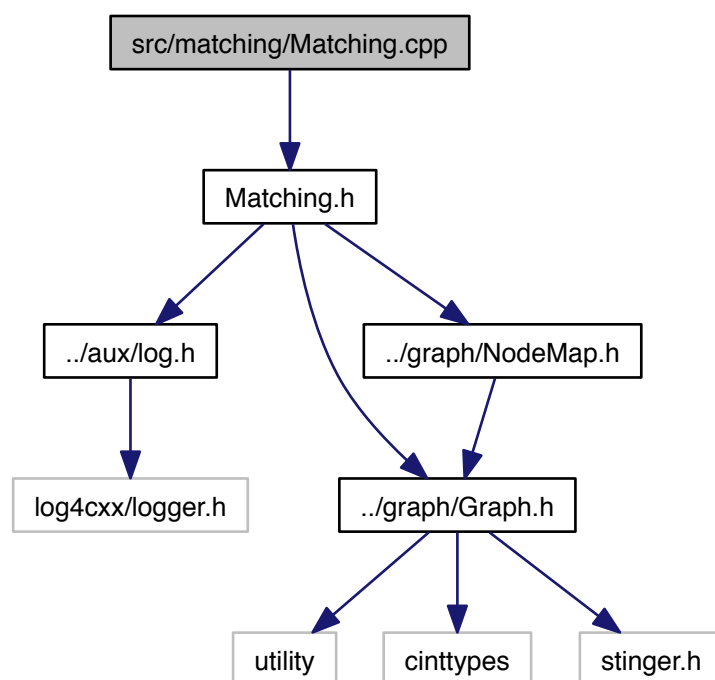
Namespaces

- namespace [EnsembleClustering](#)

7.51 src/matching/Matching.cpp File Reference

```
#include "Matching.h"
```

Include dependency graph for Matching.cpp:



Namespaces

- namespace [EnsembleClustering](#)

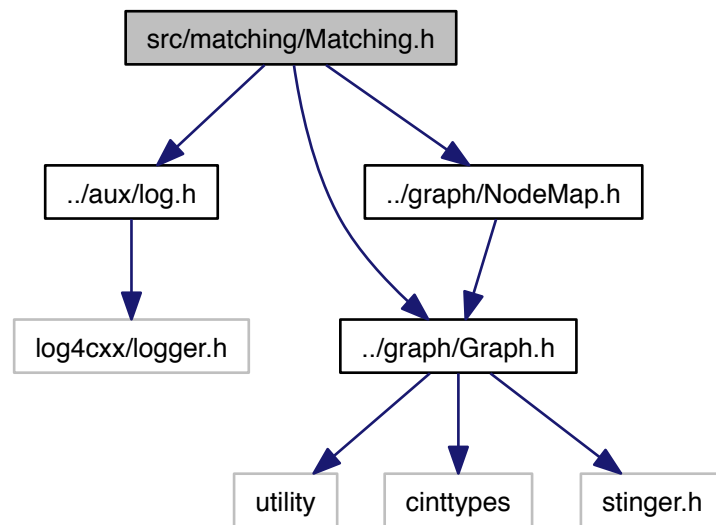
7.52 src/matching/Matching.h File Reference

```
#include "../aux/log.h"
```

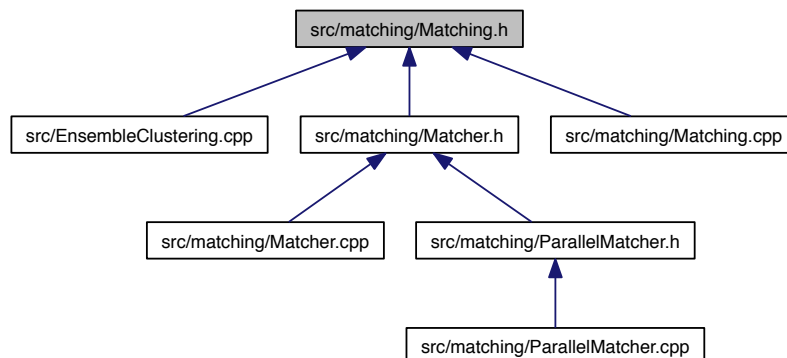
```
#include "../graph/Graph.h"
```

```
#include "../graph/NodeMap.h"
```

Include dependency graph for Matching.h:



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



Classes

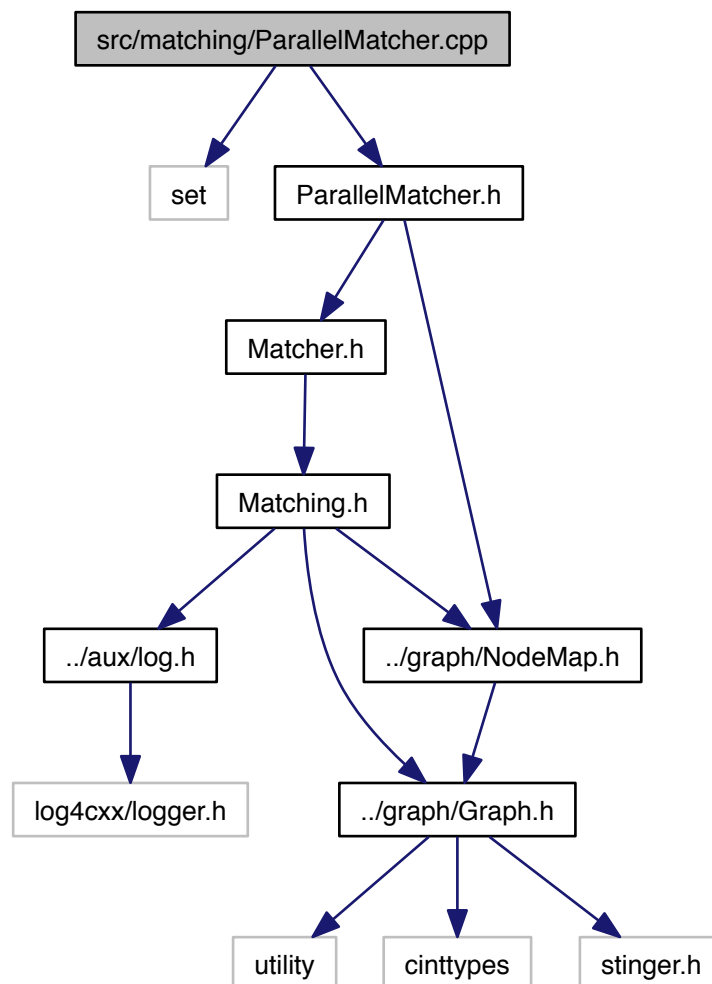
- class [EnsembleClustering::Matching](#)

Namespaces

- namespace [EnsembleClustering](#)

7.53 src/matching/ParallelMatcher.cpp File Reference

```
#include <set>
#include "ParallelMatcher.h"
Include dependency graph for ParallelMatcher.cpp:
```



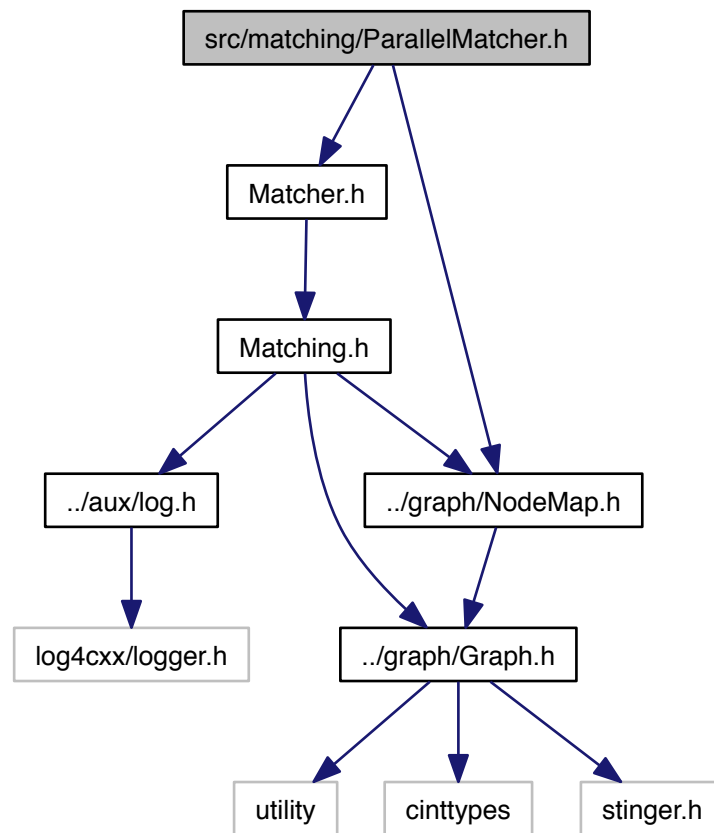
Namespaces

- namespace [EnsembleClustering](#)

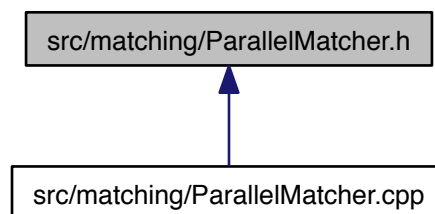
7.54 src/matching/ParallelMatcher.h File Reference

```
#include "Matcher.h"
#include "../graph/NodeMap.h"
```

Include dependency graph for ParallelMatcher.h:



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



Classes

- class [EnsembleClustering::ParallelMatcher](#)

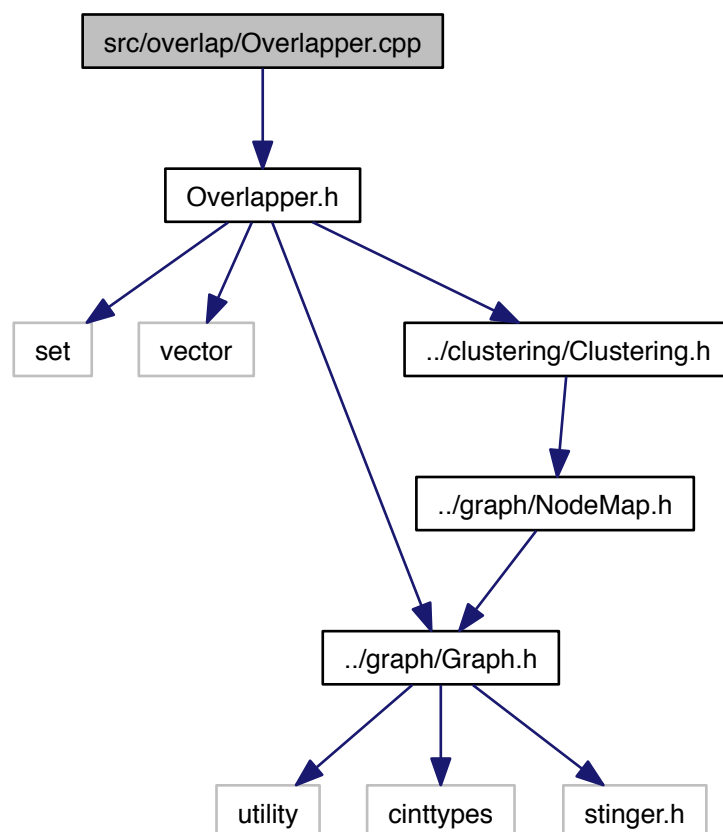
Namespaces

- namespace [EnsembleClustering](#)

7.55 src/overlap/Overlapper.cpp File Reference

```
#include "Overlapper.h"
```

Include dependency graph for Overlapper.cpp:



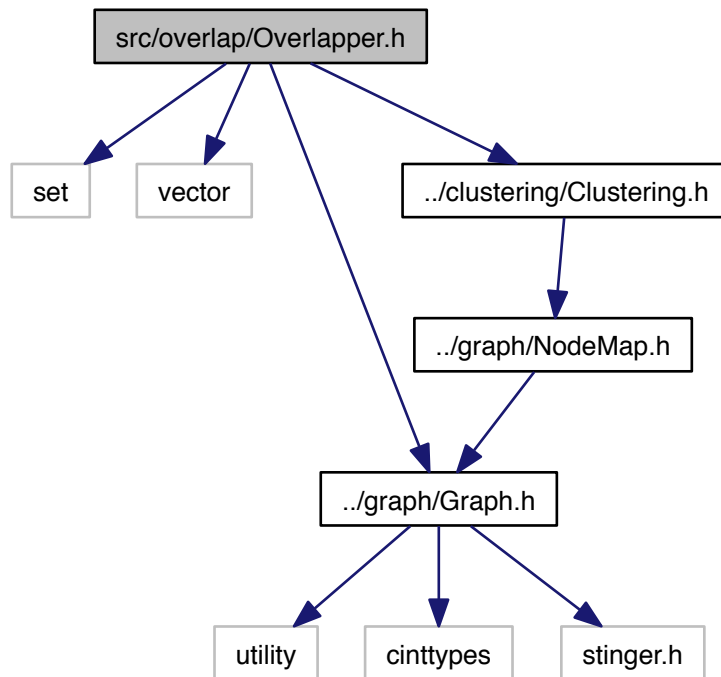
Namespaces

- namespace [EnsembleClustering](#)

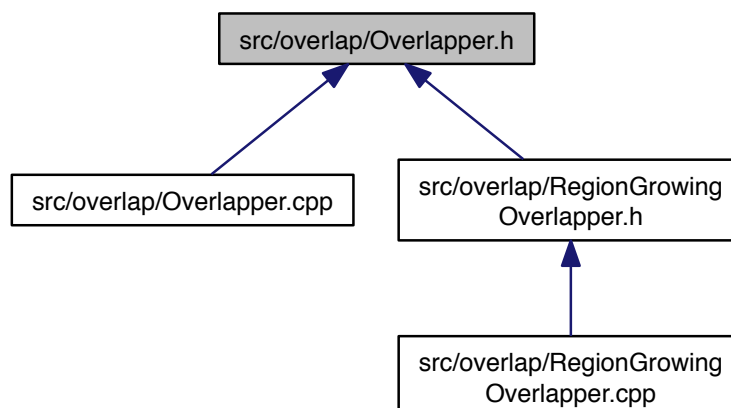
7.56 src/overlap/Overlapper.h File Reference

```
#include <set>
#include <vector>
#include "../graph/Graph.h"
#include "../clustering/Clustering.h"
```

Include dependency graph for Overlapper.h:



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



Classes

- class [EnsembleClustering::Overlapper](#)

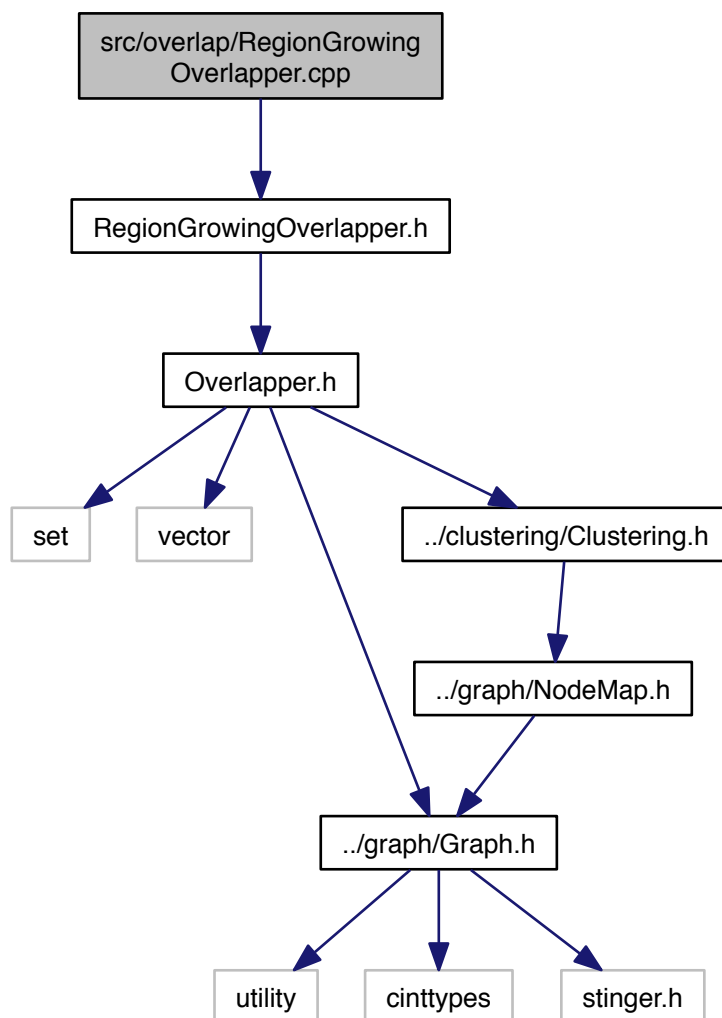
Namespaces

- namespace [EnsembleClustering](#)

7.57 src/overlap/RegionGrowingOverlapper.cpp File Reference

```
#include "RegionGrowingOverlapper.h"
```

Include dependency graph for RegionGrowingOverlapper.cpp:



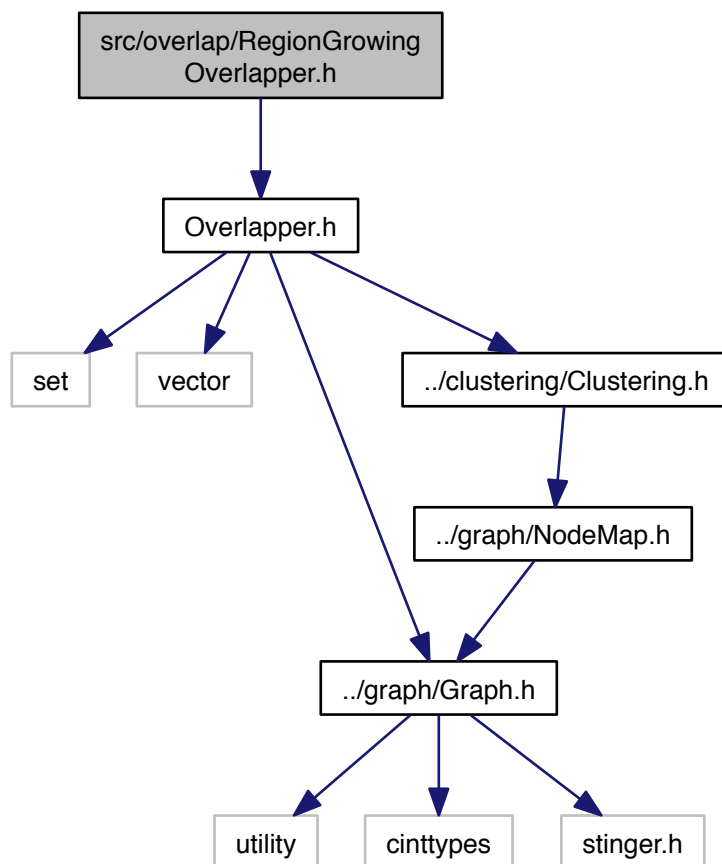
Namespaces

- namespace [EnsembleClustering](#)

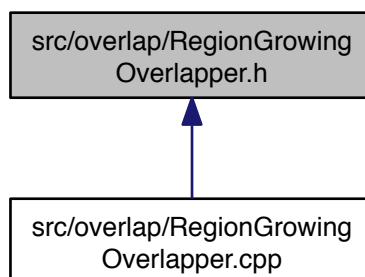
7.58 src/overlap/RegionGrowingOverlapper.h File Reference

```
#include "Overlapper.h"
```

Include dependency graph for RegionGrowingOverlapper.h:



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



Classes

- class [EnsembleClustering::RegionGrowingOverlapper](#)

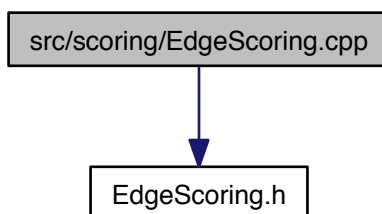
Namespaces

- namespace [EnsembleClustering](#)

7.59 src/scoring/EdgeScoring.cpp File Reference

```
#include "EdgeScoring.h"
```

Include dependency graph for EdgeScoring.cpp:

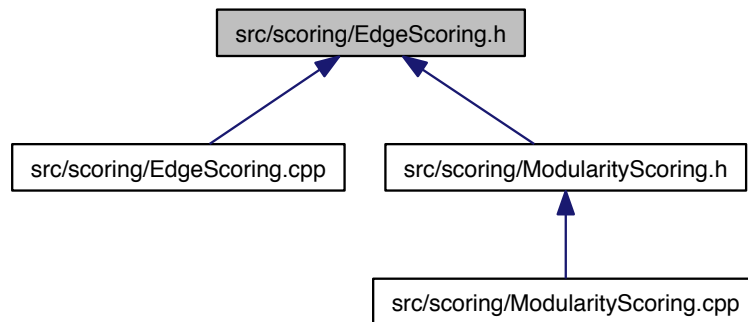


Namespaces

- namespace [EnsembleClustering](#)

7.60 src/scoring/EdgeScoring.h File Reference

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



Classes

- class [EnsembleClustering::EdgeScoring](#)

Namespaces

- namespace [EnsembleClustering](#)

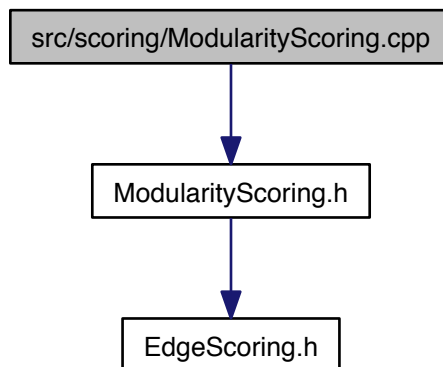
Typedefs

- typedef int [EnsembleClustering::Node](#)

7.61 src/scoring/ModularityScoring.cpp File Reference

```
#include "ModularityScoring.h"
```

Include dependency graph for ModularityScoring.cpp:



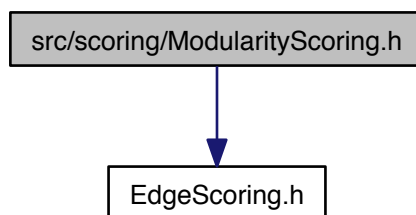
Namespaces

- namespace [EnsembleClustering](#)

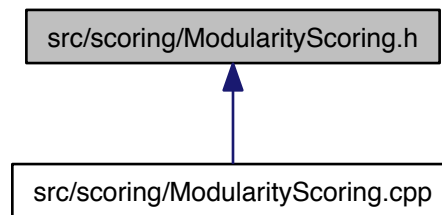
7.62 src/scoring/ModularityScoring.h File Reference

```
#include "EdgeScoring.h"
```

Include dependency graph for ModularityScoring.h:



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



Classes

- class [EnsembleClustering::ModularityScoring](#)

Namespaces

- namespace [EnsembleClustering](#)

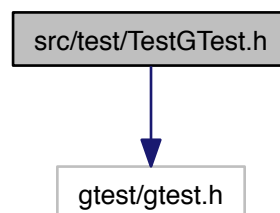
Typedefs

- typedef int [EnsembleClustering::Edge](#)
- typedef int [EnsembleClustering::Clustering](#)
- typedef int [EnsembleClustering::Cluster](#)

7.63 src/test/TestGTest.h File Reference

```
#include "gtest/gtest.h"
```

Include dependency graph for TestGTest.h:



Classes

- class [GTestTest](#)

Functions

- [TEST_F](#) ([GTestTest](#), myFirstTest)

7.63.1 Function Documentation

7.63.1.1 [TEST_F](#) ([GTestTest](#) , myFirstTest)

Definition at line 23 of file TestGTest.h.

Index

- ~ClusterContracter
 - EnsembleClustering::ClusterContracter, [14](#)
- ~Clusterer
 - EnsembleClustering::Clusterer, [15](#)
- ~Clustering
 - EnsembleClustering::Clustering, [17](#)
- ~ClusteringGenerator
 - EnsembleClustering::ClusteringGenerator, [19](#)
- ~Contracter
 - EnsembleClustering::Contracter, [22](#)
- ~EdgeScoring
 - EnsembleClustering::EdgeScoring, [23](#)
- ~EnsembleClusterer
 - EnsembleClustering::EnsembleClusterer, [24](#)
- ~Graph
 - EnsembleClustering::Graph, [26](#)
- ~GraphGenerator
 - EnsembleClustering::GraphGenerator, [29](#)
- ~IndexMap
 - EnsembleClustering::IndexMap, [35](#)
- ~LabelPropagation
 - EnsembleClustering::LabelPropagation, [38](#)
- ~METISParser
 - EnsembleClustering::METISParser, [44](#)
- ~METIStoSTINGER
 - EnsembleClustering::METIStoSTINGER, [46](#)
- ~Matcher
 - EnsembleClustering::Matcher, [39](#)
- ~Matching
 - EnsembleClustering::Matching, [41](#)
- ~MatchingContracter
 - EnsembleClustering::MatchingContracter, [43](#)
- ~Modularity
 - EnsembleClustering::Modularity, [49](#)
- ~ModularityScoring
 - EnsembleClustering::ModularityScoring, [52](#)
- ~NodeMap
 - EnsembleClustering::NodeMap, [54](#)
- ~Noise
 - Noise, [56](#)
- ~Overlapper
 - EnsembleClustering::Overlapper, [57](#)
- ~ParallelMatcher
 - EnsembleClustering::ParallelMatcher, [58](#)
- ~QualityMeasure
 - EnsembleClustering::QualityMeasure, [61](#)
- ~RandomProbability
 - RandomProbability, [62](#)
- ~RegionGrowingOverlapper
 - EnsembleClustering::RegionGrowingOverlapper, [63](#)
- ~STINGERFromAdjacencies
 - EnsembleClustering::STINGERFromAdjacencies, [66](#)
- ~ScoreMatchContract
 - EnsembleClustering::ScoreMatchContract, [64](#)
- ~Timer
 - Timer, [67](#)
- add
 - Noise, [56](#)
- addAdjacencies
 - EnsembleClustering::STINGERFromAdjacencies, [66](#)
- addToCluster
 - EnsembleClustering::Clustering, [17](#)
- areMatched
 - EnsembleClustering::Matching, [41](#)
- array
 - EnsembleClustering::IndexMap, [35](#)
 - EnsembleClustering::NodeMap, [54](#)
- asSTINGER
 - EnsembleClustering::Graph, [26](#)
- clone
 - EnsembleClustering::Matching, [41](#)
- close
 - EnsembleClustering::METISParser, [44](#)
- Cluster
 - EnsembleClustering, [10](#)
- cluster
 - EnsembleClustering, [10](#)
- ClusterContracter
 - EnsembleClustering::ClusterContracter, [14](#)
- clusterOf
 - EnsembleClustering::Clustering, [17](#)
- Clusterer
 - EnsembleClustering::Clusterer, [15](#)
- Clustering
 - EnsembleClustering, [10](#)
 - EnsembleClustering::Clustering, [17](#)
- ClusteringGenerator
 - EnsembleClustering::ClusteringGenerator, [19](#)
- configureLogging
 - EnsembleClustering.cpp, [98](#)
- contract
 - EnsembleClustering::Contracter, [22](#)
- Contracter
 - EnsembleClustering::Contracter, [22](#)

- createGraph
 - EnsembleClustering::STINGERFromAdjacencies, 66
- currentNode
 - EnsembleClustering::STINGERFromAdjacencies, 67
- cutweight
 - EnsembleClustering::ModularityScoring, 52
- DEBUG
 - log.h, 70
- defaultEdgeType
 - EnsembleClustering::Graph, 28
- defaultEdgeWeight
 - EnsembleClustering::Graph, 28
- defaultTimeStamp
 - EnsembleClustering::Graph, 28
- defaultValue
 - EnsembleClustering::IndexMap, 35
 - EnsembleClustering::NodeMap, 54
- degree
 - EnsembleClustering::Graph, 26
- deltaMod
 - EnsembleClustering::ModularityScoring, 52
- dispose
 - EnsembleClustering::Matching, 41
- EDGE_DEST
 - Graph.h, 102
- EDGE_SOURCE
 - Graph.h, 102
- ERROR
 - log.h, 70
- Edge
 - EnsembleClustering, 10
- edge
 - EnsembleClustering, 10
- EdgeScoring
 - EnsembleClustering::EdgeScoring, 23
- EnsembleClusterer
 - EnsembleClustering::EnsembleClusterer, 24
- EnsembleClustering, 9
 - Cluster, 10
 - cluster, 10
 - Clustering, 10
 - Edge, 10
 - edge, 10
 - Node, 10
 - node, 10
 - TEST_F, 11, 12
- EnsembleClustering.cpp
 - configureLogging, 98
 - main, 98
 - makeCompleteGraph, 98
 - testMETIStoSTINGER, 99
 - testMatching, 99
- EnsembleClustering::ClusterContracter, 13
 - ~ClusterContracter, 14
 - ClusterContracter, 14
- EnsembleClustering::Clusterer, 14
 - ~Clusterer, 15
 - Clusterer, 15
 - run, 15
- EnsembleClustering::Clustering, 15
 - ~Clustering, 17
 - addToCluster, 17
 - clusterOf, 17
 - Clustering, 17
 - firstCluster, 17
 - isProper, 17
 - lastCluster, 17
 - mergeClusters, 17
 - moveToCluster, 18
 - nextCluster, 18
 - toSingleton, 18
- EnsembleClustering::ClusteringGenerator, 19
 - ~ClusteringGenerator, 19
 - ClusteringGenerator, 19
 - makeOneClustering, 19
 - makeSingletonClustering, 20
- EnsembleClustering::ClusteringTest, 21
- EnsembleClustering::Contracter, 21
 - ~Contracter, 22
 - contract, 22
 - Contracter, 22
- EnsembleClustering::EdgeScoring, 23
 - ~EdgeScoring, 23
 - EdgeScoring, 23
 - scoreEdge, 23
- EnsembleClustering::EnsembleClusterer, 24
 - ~EnsembleClusterer, 24
 - EnsembleClusterer, 24
- EnsembleClustering::Graph, 24
 - ~Graph, 26
 - asSTINGER, 26
 - defaultEdgeType, 28
 - defaultEdgeWeight, 28
 - defaultTimeStamp, 28
 - degree, 26
 - firstNode, 26
 - forallEdges, 26
 - Graph, 25
 - hasEdge, 26
 - insertEdge, 26
 - lastNode, 26
 - numberOfEdges, 27
 - numberOfNodes, 27
 - stingerG, 28
 - totalEdgeWeight, 27
 - weight, 27, 28
- EnsembleClustering::GraphGTest, 30
 - gen, 32
 - SetUp, 31
 - TearDown, 32
- EnsembleClustering::GraphGenerator, 28
 - ~GraphGenerator, 29
 - GraphGenerator, 29

- makeCircularGraph, 29
- makeCompleteGraph, 29
- makeErdosRenyiGraph, 30
- EnsembleClustering::IndexMap
 - ~IndexMap, 35
 - array, 35
 - defaultValue, 35
 - IndexMap, 34
 - n, 35
- EnsembleClustering::IndexMap< I, T >, 33
- EnsembleClustering::InputGTest, 36
- EnsembleClustering::LabelPropagation, 37
 - ~LabelPropagation, 38
 - LabelPropagation, 38
 - run, 38
- EnsembleClustering::METISParser, 44
 - ~METISParser, 44
 - close, 44
 - getHeader, 44
 - getNext, 45
 - graphFile, 45
 - graphPath, 45
 - hasNext, 45
 - line, 45
 - METISParser, 44
 - nodeCount, 45
 - open, 45
- EnsembleClustering::METISToSTINGER, 45
 - read, 46
- EnsembleClustering::Matcher, 39
 - ~Matcher, 39
 - Matcher, 39
 - run, 39
- EnsembleClustering::Matching, 40
 - ~Matching, 41
 - areMatched, 41
 - clone, 41
 - dispose, 41
 - isMatched, 41
 - isProper, 42
 - match, 42
 - Matching, 41
 - operator=, 42
 - unmatch, 43
- EnsembleClustering::MatchingContracter, 43
 - ~MatchingContracter, 43
 - MatchingContracter, 43
- EnsembleClustering::Modularity, 47
 - ~Modularity, 49
 - getQuality, 49
 - incidentWeight, 50
 - Modularity, 49
 - precompute, 50
- EnsembleClustering::ModularityScoring, 51
 - ~ModularityScoring, 52
 - cutweight, 52
 - deltaMod, 52
 - mod, 52
- ModularityScoring, 52
 - scoreEdge, 52
 - weight, 52
- EnsembleClustering::NodeMap
 - ~NodeMap, 54
 - array, 54
 - defaultValue, 54
 - n, 55
 - NodeMap, 54
- EnsembleClustering::NodeMap< T >, 53
- EnsembleClustering::Overlapper, 56
 - ~Overlapper, 57
 - Overlapper, 57
- EnsembleClustering::ParallelMatcher, 57
 - ~ParallelMatcher, 58
 - ParallelMatcher, 58
 - run, 59
- EnsembleClustering::QualityMeasure, 59
 - ~QualityMeasure, 61
 - G, 61
 - getQuality, 61
 - QualityMeasure, 61
- EnsembleClustering::RegionGrowingOverlapper, 62
 - ~RegionGrowingOverlapper, 63
 - RegionGrowingOverlapper, 63
- EnsembleClustering::STINGERFromAdjacencies, 65
 - addAdjacencies, 66
 - createGraph, 66
 - currentNode, 67
 - G, 67
 - getGraph, 66
 - getSTINGER, 66
 - STINGERFromAdjacencies, 66
- EnsembleClustering::ScoreMatchContract, 63
 - ~ScoreMatchContract, 64
 - ScoreMatchContract, 64
- FATAL
 - log.h, 70
- FORALL_EDGES_BEGIN
 - Graph.h, 102
- FORALL_EDGES_END
 - Graph.h, 102
- firstCluster
 - EnsembleClustering::Clustering, 17
- firstNode
 - EnsembleClustering::Graph, 26
- forallEdges
 - EnsembleClustering::Graph, 26
- G
 - EnsembleClustering::QualityMeasure, 61
 - EnsembleClustering::STINGERFromAdjacencies, 67
- GTestTest, 32
 - SetUp, 33
- gen
 - EnsembleClustering::GraphGTest, 32
- generate

- RandomProbability, 62
- getGraph
 - EnsembleClustering::STINGERFromAdjacencies, 66
- getHeader
 - EnsembleClustering::METISParser, 44
- getNext
 - EnsembleClustering::METISParser, 45
- getQuality
 - EnsembleClustering::Modularity, 49
 - EnsembleClustering::QualityMeasure, 61
- getSTINGER
 - EnsembleClustering::STINGERFromAdjacencies, 66
- Graph
 - EnsembleClustering::Graph, 25
- Graph.h
 - EDGE_DEST, 102
 - EDGE_SOURCE, 102
 - FORALL_EDGES_BEGIN, 102
 - FORALL_EDGES_END, 102
- graphFile
 - EnsembleClustering::METISParser, 45
- GraphGenerator
 - EnsembleClustering::GraphGenerator, 29
- graphPath
 - EnsembleClustering::METISParser, 45
- hasEdge
 - EnsembleClustering::Graph, 26
- hasNext
 - EnsembleClustering::METISParser, 45
- INFO
 - log.h, 71
- incidentWeight
 - EnsembleClustering::Modularity, 50
- IndexMap
 - EnsembleClustering::IndexMap, 34
- insertEdge
 - EnsembleClustering::Graph, 26
- isMatched
 - EnsembleClustering::Matching, 41
- isProper
 - EnsembleClustering::Clustering, 17
 - EnsembleClustering::Matching, 42
- LOCATION
 - log.h, 71
- LOGGER
 - log.h, 71
- LabelPropagation
 - EnsembleClustering::LabelPropagation, 38
- lastCluster
 - EnsembleClustering::Clustering, 17
- lastNode
 - EnsembleClustering::Graph, 26
- line
 - EnsembleClustering::METISParser, 45
- log.h
 - DEBUG, 70
 - ERROR, 70
 - FATAL, 70
 - INFO, 71
 - LOCATION, 71
 - LOGGER, 71
 - TRACE, 71
 - WARN, 71
- lowerBound
 - Noise, 56
- METISParser
 - EnsembleClustering::METISParser, 44
- METISToSTINGER
 - EnsembleClustering::METISToSTINGER, 46
- main
 - EnsembleClustering.cpp, 98
- makeCircularGraph
 - EnsembleClustering::GraphGenerator, 29
- makeCompleteGraph
 - EnsembleClustering.cpp, 98
 - EnsembleClustering::GraphGenerator, 29
- makeErdosRenyiGraph
 - EnsembleClustering::GraphGenerator, 30
- makeOneClustering
 - EnsembleClustering::ClusteringGenerator, 19
- makeSingletonClustering
 - EnsembleClustering::ClusteringGenerator, 20
- match
 - EnsembleClustering::Matching, 42
- Matcher
 - EnsembleClustering::Matcher, 39
- Matching
 - EnsembleClustering::Matching, 41
- MatchingContracter
 - EnsembleClustering::MatchingContracter, 43
- mergeClusters
 - EnsembleClustering::Clustering, 17
- mod
 - EnsembleClustering::ModularityScoring, 52
- Modularity
 - EnsembleClustering::Modularity, 49
- ModularityScoring
 - EnsembleClustering::ModularityScoring, 52
- moveToCluster
 - EnsembleClustering::Clustering, 18
- n
 - EnsembleClustering::IndexMap, 35
 - EnsembleClustering::NodeMap, 55
- nextCluster
 - EnsembleClustering::Clustering, 18
- Node
 - EnsembleClustering, 10
- node
 - EnsembleClustering, 10
- nodeCount
 - EnsembleClustering::METISParser, 45

- NodeMap
 - EnsembleClustering::NodeMap, 54
- Noise, 55
 - ~Noise, 56
 - add, 56
 - lowerBound, 56
 - Noise, 55
 - randomEngine, 56
 - uniform, 56
 - upperBound, 56
- numberOfEdges
 - EnsembleClustering::Graph, 27
- numberOfNodes
 - EnsembleClustering::Graph, 27
- open
 - EnsembleClustering::METISParser, 45
- operator=
 - EnsembleClustering::Matching, 42
- Overlapper
 - EnsembleClustering::Overlapper, 57
- ParallelMatcher
 - EnsembleClustering::ParallelMatcher, 58
- precompute
 - EnsembleClustering::Modularity, 50
- QualityMeasure
 - EnsembleClustering::QualityMeasure, 61
- randomEngine
 - Noise, 56
 - RandomProbability, 62
- RandomProbability, 61
 - ~RandomProbability, 62
 - generate, 62
 - randomEngine, 62
 - RandomProbability, 62
 - RandomProbability, 62
 - uniform, 62
- read
 - EnsembleClustering::METISToSTINGER, 46
- RegionGrowingOverlapper
 - EnsembleClustering::RegionGrowingOverlapper, 63
- run
 - EnsembleClustering::Clusterer, 15
 - EnsembleClustering::LabelPropagation, 38
 - EnsembleClustering::Matcher, 39
 - EnsembleClustering::ParallelMatcher, 59
- STINGERFromAdjacencies
 - EnsembleClustering::STINGERFromAdjacencies, 66
- scoreEdge
 - EnsembleClustering::EdgeScoring, 23
 - EnsembleClustering::ModularityScoring, 52
- ScoreMatchContract
 - EnsembleClustering::ScoreMatchContract, 64
- SetUp
 - EnsembleClustering::GraphGTest, 31
 - GTestTest, 33
 - src/EnsembleClustering.cpp, 97
 - src/aux/IndexMap.h, 69
 - src/aux/Noise.cpp, 71
 - src/aux/Noise.h, 72
 - src/aux/RandomProbability.cpp, 72
 - src/aux/RandomProbability.h, 73
 - src/aux/Timer.cpp, 74
 - src/aux/Timer.h, 75
 - src/aux/log.h, 70
 - src/clustering/Clusterer.cpp, 76
 - src/clustering/Clusterer.h, 76
 - src/clustering/Clustering.cpp, 78
 - src/clustering/Clustering.h, 78
 - src/clustering/ClusteringGenerator.cpp, 80
 - src/clustering/ClusteringGenerator.h, 80
 - src/clustering/LabelPropagation.cpp, 82
 - src/clustering/LabelPropagation.h, 82
 - src/clustering/Modularity.cpp, 84
 - src/clustering/Modularity.h, 84
 - src/clustering/QualityMeasure.cpp, 86
 - src/clustering/QualityMeasure.h, 86
 - src/clustering/ScoreMatchContract.cpp, 88
 - src/clustering/ScoreMatchContract.h, 89
 - src/clustering/test/ClusteringTest.cpp, 90
 - src/clustering/test/ClusteringTest.h, 91
 - src/coarsening/ClusterContracter.cpp, 92
 - src/coarsening/ClusterContracter.h, 92
 - src/coarsening/Contracter.cpp, 93
 - src/coarsening/Contracter.h, 94
 - src/coarsening/MatchingContracter.cpp, 95
 - src/coarsening/MatchingContracter.h, 96
 - src/ensemble/EnsembleClusterer.cpp, 96
 - src/ensemble/EnsembleClusterer.h, 97
 - src/graph/Graph.cpp, 100
 - src/graph/Graph.h, 101
 - src/graph/GraphGenerator.cpp, 104
 - src/graph/GraphGenerator.h, 104
 - src/graph/NodeMap.h, 105
 - src/graph/test/GraphGTest.cpp, 106
 - src/graph/test/GraphGTest.h, 107
 - src/input/METISParser.cpp, 108
 - src/input/METISParser.h, 109
 - src/input/METISToSTINGER.cpp, 109
 - src/input/METISToSTINGER.h, 110
 - src/input/STINGERFromAdjacencies.cpp, 111
 - src/input/STINGERFromAdjacencies.h, 112
 - src/input/test/InputGTest.cpp, 113
 - src/input/test/InputGTest.h, 113
 - src/matching/Matcher.cpp, 114
 - src/matching/Matcher.h, 115
 - src/matching/Matching.cpp, 117
 - src/matching/Matching.h, 117
 - src/matching/ParallelMatcher.cpp, 119
 - src/matching/ParallelMatcher.h, 119
 - src/overlap/Overlapper.cpp, 121

- src/overlap/Overlapper.h, [121](#)
- src/overlap/RegionGrowingOverlapper.cpp, [123](#)
- src/overlap/RegionGrowingOverlapper.h, [124](#)
- src/scoring/EdgeScoring.cpp, [125](#)
- src/scoring/EdgeScoring.h, [126](#)
- src/scoring/ModularityScoring.cpp, [127](#)
- src/scoring/ModularityScoring.h, [127](#)
- src/test/TestGTest.h, [128](#)
- stingerG
 - EnsembleClustering::Graph, [28](#)
- TEST_F
 - EnsembleClustering, [11](#), [12](#)
 - TestGTest.h, [129](#)
- TRACE
 - log.h, [71](#)
- TearDown
 - EnsembleClustering::GraphGTest, [32](#)
- TestGTest.h
 - TEST_F, [129](#)
- testMETIStoSTINGER
 - EnsembleClustering.cpp, [99](#)
- testMatching
 - EnsembleClustering.cpp, [99](#)
- Timer, [67](#)
 - ~Timer, [67](#)
 - Timer, [67](#)
- toSingleton
 - EnsembleClustering::Clustering, [18](#)
- totalEdgeWeight
 - EnsembleClustering::Graph, [27](#)
- uniform
 - Noise, [56](#)
 - RandomProbability, [62](#)
- unmatch
 - EnsembleClustering::Matching, [43](#)
- upperBound
 - Noise, [56](#)
- WARN
 - log.h, [71](#)
- weight
 - EnsembleClustering::Graph, [27](#), [28](#)
 - EnsembleClustering::ModularityScoring, [52](#)