UnitalSZ

Algorithms and library of abstract unitals and their embeddings

0.29

19 February 2018

Gábor Péter Nagy

Dávid Mezőfi

Gábor Péter Nagy
Email: nagyg@math.u-szeged.hu
Homepage: http://www.math.u-szeged.hu/~nagyg
Address: H-6720 Szeged, Aradi vértanúk tere 1

Dávid Mezőfi
Email: mezofi@math.u-szeged.hu
Homepage: http://www.math.u-szeged.hu/~mezofi
Address: H-6720 Szeged, Aradi vértanúk tere 1

Contents

Chapter 1

Abstract unitals

1.1 Global functions for internal usage

1.1.1 AU_UnitalBlistList_axiomcheck

▷ `AU_UnitalBlistList_axiomcheck(`*`bmat`*`)` (function)

Returns: `true` if *`bmat`* is the blist list of an abstract unital.

Each row of *`bmat`* corresponds to a block of the unital. We check the sizes of the blocks and the sizes of the intersections of the dual blocks.

1.1.2 AU_IsUnitalBlistList

▷ `AU_IsUnitalBlistList(`*`bmat`*`)` (function)

Returns: `true` if *`bmat`* is the blist list of an abstract unital.

Each row of *`bmat`* corresponds to a block of the unital. We check the sizes of the blocks and the sizes of the intersections of the dual blocks. Wrong *`bmat`* matrix size drops error.

1.1.3 AU_IsUnitalIncidenceMatrix

▷ `AU_IsUnitalIncidenceMatrix(`*`incmat`*`)` (function)

Returns: `true` if *`incmat`* is the incidence matrix of an abstract unital.

Each row of *`incmat`* corresponds to a block of the unital. We check the sizes of the blocks and the sizes of the intersections of the dual blocks. Wrong *`incmat`* matrix size drops error.

1.1.4 AU_IsUnitalBlockDesign

▷ `AU_IsUnitalBlockDesign(`*`blocklist`*`)` (function)

Returns: `true` if *`blocklist`* is the list of blocks of an abstract unital.

We check the sizes of the blocks and the sizes of the intersections of the dual blocks. Wrong number of blocks or wrong number of points (union of the blocks in *`blocklist`*) drops error.

1.1.5 AU_UnitalByBlistListNC

▷ `AU_UnitalByBlistListNC(`*`bmat`*`)` (function)

Returns: The unital object corresponding to the blist list *`bmat`*.

The function stores `bmat` and sets the order of the unital. The function *do not check* the necessary conditions (the size of bmat, the sizes of the blocks and their intersections).

1.2 Constructing abstract unitals

1.2.1 AbstractUnitalByBlistList

▷ `AbstractUnitalByBlistList(bmat)` (function)

Returns: The unital object corresponding to the blist list `bmat`.

Each row of `bmat` corresponds to a block of the unital. We check the sizes of the blocks and the sizes of the intersections of the dual blocks. Wrong `bmat` matrix size drops error. The function stores `bmat` and sets the `Order` of the unital.

1.2.2 AbstractUnitalByDesignBlocks

▷ `AbstractUnitalByDesignBlocks(blocklist)` (function)

Returns: The unital object corresponding to the list of blocks `blocklist`. We check the sizes of the blocks and the sizes of the intersections of the dual blocks. Wrong number of blocks or wrong number of points (union of the blocks in `blocklist`) drops error. The function stores `bmat`, which is based on `blocklist`, sets the `Order` of the unital and sets the names of the points, `PointNamesOfUnital` of the unital.

1.2.3 AbstractUnitalByIncidenceMatrix

▷ `AbstractUnitalByIncidenceMatrix(incmat)` (function)

Returns: The unital object corresponding to the incidence matrix `incmat`.

Each row of `incmat` corresponds to a block of the unital. We check the sizes of the blocks and the sizes of the intersections of the dual blocks. Wrong `incmat` matrix size drops error. The function stores `bmat`, which is based on `incmat` and sets the `Order` of the unital.

1.3 Methods for abstract unitals

1.3.1 PointsOfUnital (for IsAbstractUnitalDesign)

▷ `PointsOfUnital(u)` (attribute)

Returns: The range `[ 1..q^3 + 1 ]`.

If `u` is a unital of order q, then `u` has q^3+1 points.

1.3.2 BlocksOfUnital (for IsAbstractUnitalDesign)

▷ `BlocksOfUnital(u)` (attribute)

Returns: The blocks of the unital `u`.

If `u` is a unital of order q, then each block is a subset of the points of the unital with $q+1$ points. The blocks of an abstract unital form a $2-(q^3+1, q+1, 1)$ design.

1.3.3 PointNamesOfUnital (for IsAbstractUnitalDesign)

▷ `PointNamesOfUnital(`*`u`*`)` (attribute)

Returns: The names of the $q^3 + 1$ points of *`u`*.

The names of the points of *`u`* is a list of length $q^3 + 1$ of arbitrary GAP objects. It may be set by `SetPointNamesOfUnital`. The default is the range `[ 1..q^3 + 1 ]`.

1.3.4 IncidenceDigraph (for IsAbstractUnitalDesign)

▷ `IncidenceDigraph(`*`u`*`)` (attribute)

Returns: The (bipartite) digraph constructed from the boolean incidence matrix `bmat` of the unital *`u`*.

1.3.5 AutomorphismGroup (for IsAbstractUnitalDesign)

▷ `AutomorphismGroup(`*`u`*`)` (attribute)

Returns: The automorphism group of the unital *`u`*.

The function computes the automorphism group of *`u`* with the help of its incidence digraph.

1.3.6 Isomorphism (for IsAbstractUnitalDesign, IsAbstractUnitalDesign)

▷ `Isomorphism(`*`u1, u2`*`)` (operation)

Returns: An isomorphism between the unitals *`u1`* and *`u1`* if they are isomorphic, and `fail` otherwise.

The isomorphism is a permutation which sends the points of the unital *`u1`* to the points of the unital *`u2`* such that the it preserves the incidence between the points and the blocks. The function computes the isomorphism with the help of the incidence digraphs of the unitals *`u1`* and *`u2`*.

Chapter 2

Libraries and classes of abstract unitals

2.1 Classes of abstract unitals

2.1.1 HermitianAbstractUnital

▷ `HermitianAbstractUnital(q)` (function)

Returns: The classical unital object, which is the abstract unital of order `q` isomorphic to the Hermitian curve in the classical projective plane.

The Hermitian curve has the following canonical equation: $X_0^{q+1} + X_1^{q+1} + X_2^{q+1} = 0$. The function computes the blocks of the unital with the help of `PGU(3,q)` and calls `AbstractUnitalByDesignBlocks`. The `Name` of the unital is set as `HermitianAbstractUnital(q)`.

2.2 Global functions for internal usage

2.2.1 AU_ReadLibraryDataFromFiles

▷ `AU_ReadLibraryDataFromFiles(nr, q, filename)` (function)

Returns: The list of boolean incidence matrices of size $(q^3+1) \times q^2(q^2-q+1)$ read from `filename`.

The file `filename` must be gzipped and must contain `nr` matrices of dimension mentioned above. The matrices must be 0-1 matrices without any whitespace between the entries in one row and there must not be any empty lines between matrices.

2.2.2 AU_InitLibraryData

▷ `AU_InitLibraryData()` (function)

Returns:

Reads in the incidence matrices from the libraries of unitals shipped with the package.

2.2.3 BBTAbstractUnital

▷ `BBTAbstractUnital(n)` (function)

Returns: The `n`th (abstract) unital of order 3 of the unitals by Betten, Betten and Tonchev.

In the paper Unitals and codes by Anton Betten, Dieter Betten and Vladimir D. Tonchev (Discrete Mathematics 267, 2003, 23-33.) 909 unitals of order 3 were constructed. The incidence matrices of these unitals are shipped with the package.

2.3 Libraries

2.3.1 KNPAbstractUnital

▷ `KNPAbstractUnital(n)` (function)

Returns: The *n*th (abstract) unital of order 4 of the unitals by Krčadinac, Nakić and Pavčević.

In the paper The Kramer-Mesner method with tactical decompositions: some new unitals on 65 points by Vedran Krčadinac, Anamari Nakić and Mario Osvin Pavčević (Journal of Combinatorial Designs 19, 2011, 290-303.) 1777 unitals of order 4 were constructed. The incidence matrices of these unitals are shipped with the package.

2.3.2 KrcadinacAbstractUnital

▷ `KrcadinacAbstractUnital(n)` (function)

Returns: The *n*th (abstract) unital of order 3 of the unitals by Krčadinac.

In the paper Steiner 2-designs S(2, 4, 28) with nontrivial automorphisms by Vedran Krčadinac (Glasnik Matematički, Vol. 37 (57), 2002, 259-268.) 4466 unitals of order 3 were constructed. This library contains all the unitals of order 3 with nontrivial automorphism group. The incidence matrices of these unitals are shipped with the package.

2.3.3 AbstractUnitalLibraryInfo

▷ `AbstractUnitalLibraryInfo()` (function)

Returns:

The function prints the information about the available libraries of unitals.

Chapter 3

Full points and perspectivities

3.1 Full points of unitals

3.1.1 FullPointsOfUnitalsBlocks (for IsAbstractUnitalDesign, IsPosInt, IsPosInt)

▷ `FullPointsOfUnitalsBlocks(u, b1, b2)` (operation)

Returns: The list full point of `u` w.r.t. the blocks `b1,b2`. The arguments `b1,b2` are either blocks of the unital `u`, or indices of blocks in `BlocksOfUnital( u )`.

The point P is a *full point* of the unital U w.r.t. the blocks b_1, b_2 if P is not contained in b_1 or b_2, and, the projection with center P from b_1 to b_2 is a well-defined bijection.

3.1.2 FullPointsOfUnitalRepresentatives (for IsAbstractUnitalDesign)

▷ `FullPointsOfUnitalRepresentatives(u)` (attribute)

Returns: A list of records `r` containing the fields `r.block1, r.block2, r.fullpts`, where `r.fullpts` is the set of full point of `u` w.r.t. the blocks `r.block1, r.block2`. The returned list contains all possible full points of `u` up to the automorphism group of `u`. That is, if P is a full point w.r.t. the blocks b_1, b_2, then there is an automorphism α of U such that $P^\alpha, b_1^\alpha, b_2^\alpha$ are in the list.

3.2 Group of perspectivities

3.2.1 PerspectivityGroupOfUnitalsBlocks (for IsAbstractUnitalDesign, IsList, IsList, IsList)

▷ `PerspectivityGroupOfUnitalsBlocks(u, b1, b2[, fullpts])` (operation)

Returns: The group generated by perspectivies from block `b1` to block `b2` of the unital `u`. Notice that the returned group consists of permutations of `[1..Order(u)+1]`. A list of full points can be given as 4th argument. It is not checked if the elements of `fullpts` are full points.

Perspectivities between blocks b_1, b_2 of an abstract unital U are projections from b_1 to b_2 from a center P. In order the perspectivity be well-defined, P must be a full point w.r.t. b_1, b_2.

Index