CddInterface

Gap interface to Cdd package

2022.11.01

1 November 2022

Kamal Saleh

Kamal Saleh
Email: kamal.saleh@uni-siegen.de
Homepage: https://github.com/kamalsaleh
Address: Department Mathematik
        Universität Siegen
        Walter-Flex-Straße 3
        57072 Siegen
        Germany
# Contents

1 Introduction ............................... 3
   1.1 Why CddInterface ................................. 3
   1.2 H-representation and V-representation of polyhedra ................................. 3

2 Creating polyhedra and their Operations ............................... 5
   2.1 Creating a polyhedron ................................. 5
   2.2 Some operations on a polyhedron ................................. 6
   2.3 Some operations on two polyhedrons ................................. 8

3 Linear Programs ............................... 11
   3.1 Creating and solving linear programs ................................. 11

4 Attributes and properties ............................... 13
   4.1 Attributes and properties of polyhedron ................................. 13

Index ........................................ 19
Chapter 1

Introduction

1.1 Why CddInterface

We know that every convex polyhedron has two representations, one as the intersection of finite half-spaces and the other as Minkowski sum of the convex hull of finite points and the nonnegative hull of finite directions. These are called \( H \)-representation and \( V \)-representation, respectively. CddInterface is a gap interface to the C package cddlib which among other things can translate between these two representations.

1.2 H-representation and V-representation of polyhedra

Let us start by introducing the \( H \)-representation. Let \( A \) be \( m \times d \) matrix and let \( b \) be a column \( m \)-vector. The \( H \)-representation of the polyhedron defined by the system \( b + Ax \geq 0 \) of \( m \) inequalities and \( d \) variables \( x = (x_1, \ldots, x_d) \) is as follows:

\[
\begin{align*}
\text{Code} \quad \text{H-representation} \\
\text{linearity} t, [i_1, i_2, \ldots, i_t] \\
\text{begin} \\
\quad m \times (d+1) \text{ numbertype} \\
\quad b A \\
\text{end}
\end{align*}
\]

The linearity line is added when we want to specify that some rows of the system \( b + Ax \) are equalities. That is, \( k \in \{i_1, i_2, \ldots, i_t\} \) means that the row \( k \) of the system \( b + Ax \) is specified to be equality.

For example, the \( H \)-representation of the polyhedron defined by the following system:

\[
\begin{align*}
4 - 3x_1 + 6x_2 - 5x_4 &= 0, \\
1 + 2x_1 - 2x_2 - 7x_3 &\geq 0, \\
-3x_2 + 5x_4 &= 0;
\end{align*}
\]

is the following:

\[
\begin{align*}
\text{Code} \quad \text{H-representation} \\
\text{linearity} 2, [1, 3] \\
\text{begin} \\
\quad 3 \times 5 \text{ rational} \\
\quad 4 -3 6 0 -5 \\
\quad 1 2 -2 -7 0
\end{align*}
\]
Next we define Polyhedra $V$-format. Let $P$ be represented by $n$ generating points and $s$ generating directions (rays) as
\[ P = \text{conv}(v_1, \ldots, v_n) + \text{nonneg}(r_{n+1}, \ldots, r_{n+s}). \]

Then the Polyhedra $V$-format is for $P$ is:

```
V-representation
linearity t, [i_1, i_2, \ldots, i_t]
begin
(n+s) x (d+1) numbertype
  1  v_1
  :  :
  1  v_n
  0  r_{n+1}
  :  :
  0  r_{n+s}
end
```

In the above format the generating points and generating rays may appear mixed in arbitrary order. Linearity for $V$-representation specifies a subset of generators whose coefficients are relaxed to be free. That is, $k \in \{i_1, i_2, \ldots, i_t\}$ specifies that the $k$-th generator is specified to be free. This means for each such a ray $r_k$, the line generated by $r_k$ is in the polyhedron, and for each such a vertex $v_k$, its coefficient is no longer nonnegative but still the coefficients for all $v_i$'s must sum up to one.

For example the $V$-representation of the polyhedron defined as
\[ P := \text{conv}((2,3),(-2,-3),(-1,2)) + \text{nonneg}( (1,2),(-1,-2),(1,1) ) \]

is:

```
V-representation
linearity 2, [ 1, 3 ]
begin
  4 x 3 rational
  1  2  3
  1 -1  2
  0  1  2
  0  1  1
end
```
Chapter 2

Creating polyhedra and their Operations

2.1 Creating a polyhedron

2.1.1 Cdd_PolyhedronByInequalities

\[ \text{Cdd}\text{\_PolyhedronByInequalities}\left(\text{ineq[, linearities\_list]}\right) \]

**Returns:** a CddPolyhedron

The function takes a list in which every entry represents an inequality (or equality). In case we want some entries to represent equalities we should refer in a second list to their indices.

Example

```gap
gap> A:= Cdd_PolyhedronByInequalities( \[ \[ 0, 1, 0 \], \[ 0, 1, -1 \] \] );
<\text{Polyhedron given by its H-representation}>
\n\text{gap\_} \text{Display( A );}
\text{H-representation}
\begin{equation}
2 \times 3 \text{ rational}
\end{equation}
0 1 0
0 1 -1
end
\n\text{gap\_} B:= Cdd_PolyhedronByInequalities( \[ \[ 0, 1, 0 \], \[ 0, 1, -1 \] \], [2] );
<\text{Polyhedron given by its H-representation}>
\n\text{gap\_} \text{Display( B );}
\text{H-representation}
lin\text{earity 1, [2]}
begin
2 \times 3 \text{ rational}
0 1 0
0 1 -1
end
```

2.1.2 Cdd_PolyhedronByGenerators

\[ \text{Cdd}\text{\_PolyhedronByGenerators}\left(\text{genes[, linearities\_list]}\right) \]

**Returns:** a CddPolyhedron
The function takes a list in which every entry represents a vertex in the ambient vector space. In case we want some vertices to be free (the vertex and its negative belong to the polyhedron) we should refer in a second list to their indices.

```
gap> A := Cdd_PolyhedronByGenerators( [ [ 0, 1, 3 ], [ 1, 4, 5 ] ] );
<Polyhedron given by its V-representation>
gap> Display( A );
V-representation
begin
  2 X 3 rational
  0 1 3
  1 4 5
end
```

```
gap> A := Cdd_PolyhedronByGenerators( [ [ 0, 1, 3 ] ], [ 1 ] );
<Polyhedron given by its V-representation>
gap> Display( B );
V-representation
linearity 1, [ 1 ]
begin
  1 X 3 rational
  0 1 3
end
```

## 2.2 Some operations on a polyhedron

### 2.2.1 Cdd_FourierProjection (for IsCddPolyhedron, IsInt)

```
▷ Cdd_FourierProjection(P, i)  
    (operation)

    Returns: a CddPolyhedron

    The function returns the Fourier projection of the polyhedron in the subspace \((O, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)\) after applying the Fourier elimination algorithm to get rid of the variable \(x_i\).

    To illustrate this projection, let \(P = \text{conv}( (1,2), (4,5) ) \) in \(\mathbb{Q}^2\).
```

To find its projection on the subspace \((O,x_1)\), we apply the Fourier elimination to get rid of \(x_2\).

```
gap> P := Cdd_PolyhedronByGenerators( [ [ 1, 1, 2 ], [ 1, 4, 5 ] ] );
<Polyhedron given by its V-representation>
gap> H := Cdd_H_Rep( P );
<Polyhedron given by its H-representation>
gap> Display( H );
H-representation
linearity 1, [ 3 ]
begin
  3 X 3 rational
  4 -1 0
 -1 1 0
-1 -1 1
```
end

gap> P_x1 := Cdd_FourierProjection( H, 2);
<Polyhedron given by its H-representation>
gap> Display( P_x1 );
H-representation
linearity 1, [ 3 ]
begin
  3 X 3 rational
   4  -1  0
   -1  1  0
   0   0  1
end
gap> Display( Cdd_V_Rep( P_x1 ) );
V-representation
begin
  2 X 3 rational
   1   1  0
   1   4  0
end

Let again $Q = \text{Conv}((2,3,4),(2,4,5)) + \text{nonneg}((1,1,1))$, and let us compute its projection on $(O, x_2, x_3)$

gap> Q := Cdd_PolyhedronByGenerators( [ [ 1, 2, 3, 4 ], [ 1, 2, 4, 5 ], [ 0, 1, 1, 1 ] ] );
<Polyhedron given by its V-representation>
gap> R := Cdd_H_Rep( Q );
<Polyhedron given by its H-representation>
gap> Display( R );
H-representation
linearity 1, [ 4 ]
begin
  4 X 4 rational
   2   1 -1  0
  -2   1  0  0
  -1  -1  1  0
  -1   0 -1  1
end
gap> P_x2_x3 := Cdd_FourierProjection( R, 1);
<Polyhedron given by its H-representation>
gap> Display( P_x2_x3 );
H-representation
linearity 2, [ 1, 3 ]
begin
  3 X 4 rational
   -1  0 -1  1
  -3  0  1  0
   0  1  0  0
end
2.3 Some operations on two polyhedrons

2.3.1 Cdd_IsContained (for IsCddPolyhedron, IsCddPolyhedron)

\[ \text{\texttt{Cdd\_IsContained}}(P1, P2) \] (operation)

\textbf{Returns:} true or false

The function returns \texttt{true} if \( P1 \) is contained in \( P2 \), otherwise returns \texttt{false}.

\textbf{Example}

\[
gap> A := \text{Cdd\_PolyhedronByInequalities}([ [10, -1, 1, 0], \]
\quad \text{\texttt{\_[1]} = Cdd\_PolyhedronByInequalities}\):
\[
\quad -24, 9, 2, 0 ], [ 1, 1, -1, 0 ], [ -23, -12, 1, 11 ] ], [ 4 ]);\]
\[
\text{\texttt{\_\_Polyhedron given by its H-representation}};\]
\[
gap> B := \text{Cdd\_PolyhedronByInequalities}([ [1, 0, 0, 0],}
\quad \text{\texttt{\_[1]} = Cdd\_PolyhedronByInequalities}\):
\[
\quad -4, 1, 0, 0 ], [ 10, -1, 1, 0 ], [ -3, -1, 0, 1 ] ], [ 3, 4 ]);\]
\[
\text{\texttt{\_\_Polyhedron given by its H-representation}};\]
\[
gap> \text{Cdd\_IsContained}( B, A );\]
\texttt{true}
\[
gap> \text{Display( Cdd\_V\_Rep( A ) );}\]
\texttt{V-representation}
\[
\begin{array}{c}
3 & 4 \\
1 & 2 & 3 & 4 \\
1 & 4 & -6 & 7 \\
0 & 1 & 1 & 1 \\
\end{array}
\]
\[
gap> \text{Display( Cdd\_V\_Rep( B ) );}\]
\texttt{V-representation}
\[
\begin{array}{c}
2 & 4 \\
1 & 4 & -6 & 7 \\
0 & 1 & 1 & 1 \\
\end{array}
\]

2.3.2 Cdd\_Intersection (for IsCddPolyhedron, IsCddPolyhedron)

\[ \text{\texttt{Cdd\_Intersection}}(P1, P2) \] (operation)

\textbf{Returns:} a CddPolyhedron

The function returns the intersection of \( P1 \) and \( P2 \).
Example

```
gap> A := Cdd_PolyhedronByInequalities([ [ 3, 4, 5 ] ], [ 1 ]);;
gap> B := Cdd_PolyhedronByInequalities([ [ 9, 7, 2 ] ], [ 1 ]);;
gap> C := Cdd_Intersection(A, B);
V-representation
linearity 1, [ 2 ]
begin
  2 X 3 rational
  1  -3/4  0
  0   -5   4
end
```

```
gap> Display( Cdd_V_Rep( A ) );
V-representation
begin
  2 X 3 rational
  1  -3/4  0
  0   -5   4
end
```

```
gap> Display( Cdd_V_Rep( B ) );
V-representation
linearity 1, [ 2 ]
begin
  2 X 3 rational
  1  -9/7  0
  0   -2   7
end
```

```
gap> Display( Cdd_V_Rep( C ) );
V-representation
begin
  1 X 3 rational
  1   -13/9  5/9
end
```

### 2.3.3 \+ (for IsCddPolyhedron, IsCddPolyhedron)

\textbf{\+)
\((P_1, P_2)\)

\textbf{Returns:} a \texttt{CddPolyhedron}

The function returns the Minkowski sum of \(P_1\) and \(P_2\).

```
Example

```
```
gap> P := Cdd_PolyhedronByGenerators([ [ 1, 2, 5 ], [ 0, 1, 2 ] ]);;
< Polyhedron given by its V-representation >
gap> Q := Cdd_PolyhedronByGenerators([ [ 1, 4, 6 ], [ 1, 3, 7 ], [ 0, 3, 1 ] ]);;
< Polyhedron given by its V-representation >
gap> S := P+Q;
< Polyhedron given by its H-representation >
gap> V := Cdd_V_Rep(S);
< Polyhedron given by its V-representation >
gap> Display(V);
V-representation
begin
  4 X 3 rational
  0   3   1
  1   6  11
  1   5  12
end
```
end

gap> Cdd_GeneratingVertices( P );
[ [ 2, 5 ] ]
gap> Cdd_GeneratingVertices( Q );
[ [ 3, 7 ], [ 4, 6 ] ]
gap> Cdd_GeneratingVertices( S );
[ [ 5, 12 ], [ 6, 11 ] ]
gap> Cdd_GeneratingRays( P );
[ [ 1, 2 ] ]
gap> Cdd_GeneratingRays( Q );
[ [ 3, 1 ] ]
gap> Cdd_GeneratingRays( S );
[ [ 1, 2 ], [ 3, 1 ] ]
Chapter 3

Linear Programs

3.1 Creating and solving linear programs

3.1.1 Cdd_LinearProgram (for IsCddPolyhedron, IsString, IsList)

\[ \text{Cdd\_LinearProgram}(P, \text{str}, \text{obj}) \]

\textbf{Operation}

\textbf{Returns:} a \textit{CddLinearProgram} Object

The function takes three variables. The first is a polyhedron \textit{poly}, the second \textit{str} should be "max" or "min" and the third \textit{obj} is the objective function.

3.1.2 Cdd_SolveLinearProgram (for IsCddLinearProgram)

\[ \text{Cdd\_SolveLinearProgram}(\text{lp}) \]

\textbf{Operation}

\textbf{Returns:} a list if the program is optimal, otherwise returns the value 0

The function takes a linear program. If the program is optimal, the function returns a list of two entries, the solution vector and the optimal value of the objective, otherwise it returns \textit{fail}.

To illustrate the using of these functions, let us solve the linear program given by:

\[
\text{Maximize } P(x, y) = 1 - 2x + 5y, \text{ with }
\]

\[
100 \leq x \leq 200, 80 \leq y \leq 170, y \geq -x + 200.
\]

We bring the inequalities to the form \( b + AX \geq 0 \) and get:

\[
-100 + x \geq 0, 200 - x \geq 0, -80 + y \geq 0, 170 - y \geq 0, -200 + x + y \geq 0.
\]

\texttt{gap} > A:= Cdd\_PolyhedronByInequalities( \[ \[ -100, 1, 0 \], [ 200, -1, 0 ] ,
> \[ -80, 0, 1 \], [ 170, 0, -1 ], [ -200, 1, 1 ] \] );
\texttt{<Polyhedron given by its H-representation>}
\texttt{gap} > lp1:= Cdd\_LinearProgram( A, "max", [1, -2, 5 ] );
\texttt{<Linear program>}
\texttt{gap} > Display( lp1 );
Linear program given by:
H-representation
begin
   5 X 3 rational
\begin{verbatim}
max [ 1, -2, 5 ]
gap> Cdd_SolveLinearProgram( lp1 );
[ [ 100, 170 ], 651 ]
gap> lp2:= Cdd_LinearProgram( A, "min", [ 1, -2, 5 ] );
<Linear program>
\end{verbatim}
\begin{verbatim}
gap> Display( lp2 );
Linear program given by:
H-representation
begin
  5 X 3 rational

-100 1  0
  200 -1  0
-200 1  1
  170 0  -1
  200 0  -1
end
min [ 1, -2, 5 ]
gap> Cdd_SolveLinearProgram( lp2 );
[ [ 200, 80 ], 1 ]
gap> B:= Cdd_V_Rep( A );
<Polyhedron given by its V-representation>
\end{verbatim}

So the optimal solution for \( lp1 \) is \((x = 100, y = 170)\) with optimal value \( p = 1 - 2(100) + 5(170) = 651 \) and for \( lp2 \) is \((x = 200, y = 80)\) with optimal value \( p = 1 - 2(200) + 5(80) = 1 \).
Chapter 4

Attributes and properties

4.1 Attributes and properties of polyhedron

4.1.1 Cdd_Canonicalize (for IsCddPolyhedron)

\[
\text{Cdd}\_\text{Canonicalize}(P) \quad \text{(attribute)}
\]

Returns: a CddPolyhedron

The function takes a polyhedron and reduces its defining inequalities (generators set) by deleting all redundant inequalities (generators).

Example

\[
\text{gap> A:= Cdd\_PolyhedronByInequalities( \{ [0, 2, 6], [0, 1, 3], [1, 4, 10] \} );}
\]
\(<\text{Polyhedron given by its H-representation}>
\text{gap> B:= Cdd\_Canonicalize( A );}
\text{gap> Display( B );}
\text{H-representation}
\text{begin}
\begin{array}{ccc}
0 & 1 & 3 \\
1 & 4 & 10 \\
\end{array}
\text{end}
\]

4.1.2 Cdd_V_Rep (for IsCddPolyhedron)

\[
\text{Cdd}\_\text{V}\_\text{Rep}(P) \quad \text{(attribute)}
\]

Returns: a CddPolyhedron

The function takes a polyhedron and returns its reduced V-representation.

4.1.3 Cdd_H_Rep (for IsCddPolyhedron)

\[
\text{Cdd}\_\text{H}\_\text{Rep}(P) \quad \text{(attribute)}
\]

Returns: a CddPolyhedron

The function takes a polyhedron and returns its reduced H-representation.
CddInterface

```gap
gap> B := Cdd_V_Rep( A );
<Polyhedron given by its V-representation>
gap> Display( B );
V-representation
linearity 1, [ 2 ]
begin
  2 X 3 rational

  0  1  0
  0 -1  1
end
gap> C := Cdd_H_Rep( B );
<Polyhedron given by its H-representation>
gap> Display( C );
H-representation
begin
  1 X 3 rational

  0  1  1
end
gap> D := Cdd_PolyhedronByInequalities( [ [ 0, 1, 1, 34, 22, 43 ],
     [ 11, 2, 2, 54, 53, 221 ], [33, 23, 45, 2, 40, 11 ] ] );
<Polyhedron given by its H-representation>
gap> Cdd_V_Rep( D );
<Polyhedron given by its V-representation>
gap> Display( last );
V-representation
linearity 2, [ 5, 6 ]
begin
  6 X 6 rational

  1  -743/14  369/14  11/14  0  0
  0  -1213   619   22   0  0
  0   -1     1    0   0  0
  0   764   -390  -11   0  0
  0  -13526  6772   99  154 0
  0  -116608 59496 1485  0  154
end
```

4.1.4 Cdd_AmbientSpaceDimension (for IsCddPolyhedron)

```gap
▷ Cdd_AmbientSpaceDimension(P)  (attribute)

Returns:  The dimension of the ambient space of the polyhedron(i.e., the space that contains P).
```

4.1.5 Cdd_Dimension (for IsCddPolyhedron)

```gap
▷ Cdd_Dimension(P)  (attribute)

Returns:  The dimension of the polyhedron, where the dimension, dim(P), of a polyhedron P is the maximum number of affinely independent points in P minus 1.
```
4.1.6 Cdd_GeneratingVertices (for IsCddPolyhedron)

\[ \text{Cdd\_GeneratingVertices}(P) \]

**Returns:** The reduced generating vertices of the polyhedron

4.1.7 Cdd_GeneratingRays (for IsCddPolyhedron)

\[ \text{Cdd\_GeneratingRays}(P) \]

**Returns:** list

The output is the reduced generating rays of the polyhedron

4.1.8 Cdd_Equalities (for IsCddPolyhedron)

\[ \text{Cdd\_Equalities}(P) \]

**Returns:** a list

The output is the reduced equalities of the polyhedron.

4.1.9 Cdd_Inequalities (for IsCddPolyhedron)

\[ \text{Cdd\_Inequalities}(P) \]

**Returns:** a list

The output is the reduced inequalities of the polyhedron.

4.1.10 Cdd_InteriorPoint (for IsCddPolyhedron)

\[ \text{Cdd\_InteriorPoint}(P) \]

**Returns:** a list

The output is an interior point in the polyhedron.

4.1.11 Cdd_Faces (for IsCddPolyhedron)

\[ \text{Cdd\_Faces}(P) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and returns a list. Every entry in this list is again a list, contains the dimension and linearity of the face defined as a polyhedron over the same system of inequalities.

4.1.12 Cdd_FacesWithFixedDimension (for IsCddPolyhedron, IsInt)

\[ \text{Cdd\_FacesWithFixedDimension}(P, d) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and a positive integer \( d \). The output is a list. Every entry in this list is the linearity of an \( d \)-dimensional face of \( P \) defined as a polyhedron over the same system of inequalities.
4.1.13 Cdd_FacesWithInteriorPoints (for IsCddPolyhedron)

\[ \text{Cdd_FacesWithInteriorPoints}(P) \]

**Attributes**

\[ \text{Cdd_FacesWithInteriorPoints}(P) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and returns a list. Every entry in this list is a again a list, contains the dimension, linearity of the face defined as a polyhedron over the same system of inequalities and an interior point in the face.

4.1.14 Cdd_FacesWithFixedDimensionAndInteriorPoints (for IsCddPolyhedron, IsInt)

\[ \text{Cdd_FacesWithFixedDimensionAndInteriorPoints}(P, d) \]

**Operation**

\[ \text{Cdd_FacesWithFixedDimensionAndInteriorPoints}(P, d) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and a positive integer \( d \). The output is a list. Every entry in this list is a again a list, contains the linearity of the face defined as a polyhedron over the same system of inequalities and an interior point in this face.

4.1.15 Cdd_Facets (for IsCddPolyhedron)

\[ \text{Cdd_Facets}(P) \]

**Attributes**

\[ \text{Cdd_Facets}(P) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and returns a list. Every entry in this list is the linearity of a facet defined as a polyhedron over the same system of inequalities.

4.1.16 Cdd_Lines (for IsCddPolyhedron)

\[ \text{Cdd_Lines}(P) \]

**Attributes**

\[ \text{Cdd_Lines}(P) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and returns a list. Every entry in this list is the linearity of a ray (1-dimensional face) defined as a polyhedron over the same system of inequalities.

4.1.17 Cdd_Vertices (for IsCddPolyhedron)

\[ \text{Cdd_Vertices}(P) \]

**Attributes**

\[ \text{Cdd_Vertices}(P) \]

**Returns:** a list

This function takes a \( H \)-represented polyhedron \( P \) and returns a list. Every entry in this list is the linearity of a vertex defined as a polyhedron over the same system of inequalities.

4.1.18 Cdd_IsEmpty (for IsCddPolyhedron)

\[ \text{Cdd_IsEmpty}(P) \]

**Property**

\[ \text{Cdd_IsEmpty}(P) \]

**Returns:** true or false

The output is \text{true} if the polyhedron is empty and \text{false} otherwise.

4.1.19 Cdd_IsCone (for IsCddPolyhedron)

\[ \text{Cdd_IsCone}(P) \]

**Property**

\[ \text{Cdd_IsCone}(P) \]

**Returns:** true or false

The output is \text{true} if the polyhedron is cone and \text{false} otherwise.
4.1.20  Cdd_IsPointed (for IsCddPolyhedron)

- **Cdd_IsPointed(P)**
  - **Returns:** true or false
  - The output is true if the polyhedron is pointed and false otherwise

**Example**

```gap
poly := Cdd_PolyhedronByInequalities( [[ 1, 3, 4, 5, 7 ], [ 1, 3, 5, 12, 34 ],
> [ 9, 3, 0, 2, 13 ] ], [ 1 ] );
Print( "Polyhedron given by its H-representation" );
gap> Cdd_InteriorPoint( poly );
[ -194/75, 46/25, -3/25, 0 ]
gap> Cdd_FacesWithInteriorPoints( poly );
[ [ 3, [ 1 ] ], [ -194/75, 46/25, -3/25, 0 ] ], [ 2, [ 1, 2 ],
[ -62/25, 49/25, -7/25, 0 ] ], [ 1, [ 1, 2, 3 ],
[ -209/75, 56/25, -8/25, 0 ] ], [ 2, [ 1, 3 ], [ -217/75, 53/25, -4/25, 0 ] ]
gap> Cdd_Dimension( poly );
3
gap> Cdd_IsPointed( poly );
false
gap> Cdd_IsEmpty( poly );
false
gap> Cdd_Faces( poly );
[ [ 3, [ 1 ] ], [ 2, [ 1, 2 ] ], [ 1, [ 1, 2, 3 ] ], [ 2, [ 1, 3 ] ]
gap> poly1 := Cdd_ExtendLinearity( poly, [ 1, 2, 3 ] );
<Polyhedron given by its H-representation>
gap> Cdd_Dimension( poly1 );
1
gap> Cdd_Facets( poly );
[ [ 1, 2 ], [ 1, 3 ] ]
gap> Cdd_GeneratingVertices( poly );
[ [ -209/75, 56/25, -8/25, 0 ] ]
gap> Cdd_GeneratingRays( poly );
[ [ -23, -21, 3, 0 ], [ -97, -369, 342, -75 ] ]
gap> Cdd_Inequalities( poly );
[ [ 1, 3, 5, 12, 34 ], [ -209/75, 56/25, -8/25, 0 ] ]
gap> Cdd_Equalities( poly );
[ [ 1, 3, 4, 5, 7 ] ]
gap> P := Cdd_FourierProjection( poly, 2);
<Polyhedron given by its H-representation>
gap> Display( P );
H-representation
linearity 1, [ 3 ]
```
begin
3 X 5 rational

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>-1</td>
<td>-3</td>
<td>0</td>
<td>23</td>
<td>101</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
end
Index

for IsCddPolyhedron, IsCddPolyhedron, 9
Cdd_AmbientSpaceDimension
for IsCddPolyhedron, 14
Cdd_Canonicalize
for IsCddPolyhedron, 13
Cdd_Dimension
for IsCddPolyhedron, 14
Cdd_Equalities
for IsCddPolyhedron, 15
Cdd_Faces
for IsCddPolyhedron, 15
Cdd_FacesWithFixedDimension
for IsCddPolyhedron, IsInt, 15
Cdd_FacesWithFixedDimensionAndInteriorPoints
for IsCddPolyhedron, IsInt, 16
Cdd_FacesWithInteriorPoints
for IsCddPolyhedron, 16
Cdd_Facets
for IsCddPolyhedron, 16
Cdd_FourierProjection
for IsCddPolyhedron, IsInt, 6
Cdd_GeneratingRays
for IsCddPolyhedron, 15
Cdd_GeneratingVertices
for IsCddPolyhedron, 15
Cdd_H_Rep
for IsCddPolyhedron, 13
Cdd_Inequalities
for IsCddPolyhedron, 15
Cdd_InteriorPoint
for IsCddPolyhedron, 15
Cdd_Intersection
for IsCddPolyhedron, IsCddPolyhedron, 8
Cdd_IsCone
for IsCddPolyhedron, 16
Cdd_IsEmpty
for IsCddPolyhedron, 16
Cdd_IsPointed
for IsCddPolyhedron, 17
Cdd_LinearProgram
for IsCddPolyhedron, IsString, IsList, 11
Cdd_Lines
for IsCddPolyhedron, 16
Cdd_PolyhedronByGenerators, 5
Cdd_PolyhedronByInequalities, 5
Cdd_SolveLinearProgram
for IsCddLinearProgram, 11
Cdd_Vertices
for IsCddPolyhedron, 16
Cdd_V_Rep
for IsCddPolyhedron, 13