

Table Of Content

ConstraintNetwork	2
ConstraintNetworkFromCSPif	4
ConstraintNetworkFromInequalityRelations	6
ConstraintNetworkFromInequalityRelationsPretend	8
EltIterator	9
Factor	11
FactorCR	15
FactorCRPretend	17
FactorSumOut	19
FactorSumOutPretend	20
FactorTimes	22
Query	23
QueryPretend	26
SetConstant	28
SetConstantStandard	30
Tuple	32
Variable	33
_Paper	36
_Test	38
Index	40

Class ConstraintNetwork

```
java.lang.Object
|
+--ca.ubc.cs.kisynski.ve_in_hcsp.ConstraintNetwork
```

Direct Known Subclasses:

[ConstraintNetworkFromInequalityRelations](#), [ConstraintNetworkFromInequalityRelationsPretend](#)

< [Methods](#) >

```
public abstract class ConstraintNetwork
extends java.lang.Object
```

A Constraint Network contains a tuple of {@link Variable}s and a set of constraints, each represented as a {@link FactorCR}.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Methods

getFactors

```
public java.util.Iterator getFactors()
```

Returns:

an {@link java.util.Iterator} over the factors of the constraint network.

getFactorsNum

```
public int getFactorsNum()
```

Returns:

number of the factors in the constraint network.

getVariables

```
public java.util.Iterator getVariables()
```

Returns:

an {@link java.util.Iterator} over the variables of the constraint network.

getVariablesNum

```
public int getVariablesNum()
```

Returns:

number of the variables in the constraint network.

print

```
public void print()
```

Prints the constraint network.

print

```
public void print(java.io.PrintStream output)
```

Prints the constraint network to the given output stream.

Parameters:

output - stream.

printBrief

```
public void printBrief()
```

Prints a summary of the constraint network.

printBrief

```
public void printBrief(java.io.PrintStream output)
```

Prints a summary of the constraint network to the given output stream.

Parameters:

output - stream.

Class ConstraintNetworkFromCSPif

```
java.lang.Object
|
+--ConstraintNetwork
    |
    +--ConstraintNetworkFromInequalityRelations
        |
        +--ca.ubc.cs.kisynski.ve_in_hcsp.ConstraintNetworkFromCSPif
```

< [Methods](#) >

```
public class ConstraintNetworkFromCSPif
extends ConstraintNetworkFromInequalityRelations
```

A XML file parser to translate CSPs in CISpace CSPIF format to IN #CSP packages internal format. It is not a very clean and efficiently written code.

Properties:

- Changes by Jacek Kisynski:

- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Leif Chang (original code) & Jacek Kisynski (adaptation for #CSP packages)

Version:

1.2 26-October-2007

Methods

construct

```
public static ConstraintNetworkFromCSPif construct( java.io.InputStream stream)
                                                    throws
java.io.IOException,
                                                    java.lang.RuntimeException,
                                                    java.lang.Exception
```

Constructs a Constraint Network.

Can be used to read network from a String if one uses the following trick:

```
"ByteArrayInputStream inputStream = new ByteArrayInputStream(string.getBytes());".
```

Parameters:

stream - InputStream with Constraint Network in CSPif format.

Returns:

Constraint Network.

Throws:

java.io.IOException - if file not readable;
java.lang.RuntimeException - if file content is corrupted;
java.lang.Exception - if some other bad thing happens to happen.

construct

```
public static ConstraintNetworkFromCSPif construct(java.lang.String filename)
                                                    throws
java.io.IOException,
                                                    java.lang.RuntimeException,
                                                    java.lang.Exception
```

Constructs a Constraint Network.

Parameters:

filename - with Constraint Network in CSPif format.

Returns:

Constraint Network.

Throws:

java.io.IOException - if file not readable.

java.lang.RuntimeException - if file content is corrupted.

java.lang.Exception - if some other bad thing happens to happen.

Class

ConstraintNetworkFromInequalityRelations

```
java.lang.Object
|
+--ConstraintNetwork
    |
    +--ca.ubc.cs.kisynski.ve_in_hcsp.ConstraintNetworkFromInequalityRelations
```

Direct Known Subclasses:

[ConstraintNetworkFromCSPif](#)

< [Constructors](#) >

```
public class ConstraintNetworkFromInequalityRelations
extends ConstraintNetwork
```

Constraint Network created from arrays representing domains and constraints.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2

of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Constructors

ConstraintNetworkFromInequalityRelations

```
public ConstraintNetworkFromInequalityRelations(java.lang.String[] variables,  
                                               java.lang.String[][] domains,  
                                               java.lang.String[][] unary,  
                                               int[][] binary)  
  
    throws null,  
                                               null
```

Constructor.

Parameters:

variables - - names of the variables;
domains - - array with (String) domains of corresponding variables from array variables;
unary - - array with elements excluded from domains of corresponding variables;
binary - constraints Arrays should have the same length, elements should belong to the corresponding domains, indexes should correspond to respective variables. Binary constraints should be listed once, in arrays corresponding to the predeceasing variable. Domains do not need to be sorted - we do this (and modify input arrays). Names of variables are not used internally.

Throws:

null - if one of the above requirements is broken.
null - if after processing unary constraints one of the variables has empty domain.

Class

ConstraintNetworkFromInequalityRelationsPretend

```
java.lang.Object
|
+--ConstraintNetwork
|
+--ca.ubc.cs.kisynski.ve_in_hcsp.ConstraintNetworkFromInequalityRelationsPretend
```

< [Constructors](#) >

```
public class ConstraintNetworkFromInequalityRelationsPretend
extends ConstraintNetwork
```

Constraint Network created from arrays representing domains and constraints. We only pretend to store them. This is useful for, for example, finding the elimination ordering an algorithm would have used.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Constructors

ConstraintNetworkFromInequalityRelationsPretend

```
public ConstraintNetworkFromInequalityRelationsPretend( java.lang.String[]
variables,
                                                    java.lang.String[][]
domains,
                                                    java.lang.String[][]
unary,
                                                    int[][] binary)
    throws null,
                                                    null
```

Constructor.

Parameters:

variables - - names of the variables;
domains - - array with (String) domains of corresponding variables from array variables;
unary - - array with elements excluded from domains of corresponding variables;
binary - constraints Arrays should have the same length, elements should belong to the corresponding domains, indexes should correspond to respective variables. Binary constraints should be listed once, in arrays corresponding to the predeceasing variable. Domains do not need to be sorted - we do this (and modify input arrays). Names of variables are not used internally.

Throws:

null - if one of the above requirements is broken;
null - if after processing unary constraints one of the variables has empty domain.

Interface EltsIterator

< [Methods](#) >

```
public interface EltsIterator
```

An iterator over the Tuples of a Factor. This follows the basic abstract Iterator interface. I don't use the standard interface because I extend it to go back to a position. This is needed for {@link FactorExpand}.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.FOve_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.FOve_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A

PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Methods

backTo

```
public void backTo(int pos)
    throws java.lang.IllegalArgumentException
```

Goes back to the given position in the iteration.

Parameters:

pos - the position in the iteration returned by currPos

Throws:

java.lang.IllegalArgumentException -

currPos

```
public int currPos()
```

Returns the current position in the iteration. The position is the index in the sequence of values. That is the position where the next value will be taken out of. This will even be true when the factor isn't represented as an array.

Returns:

the current position in the iteration.

hasNext

```
public boolean hasNext()
```

Returns true if the iterator has more elements.

Returns:

true if the iterator has more elements.

next

```
public Tuple next()
                                     throws
java.util.NoSuchElementException
```

Returns the next element in the iteration.

Returns:

the next element in the iteration.

Throws:

java.util.NoSuchElementException -

Class Factor

```
java.lang.Object
|
+--ca.ubc.cs.kisynski.ve_in_hcsp.Factor
```

Direct Known Subclasses:

[FactorTimes](#)

< [Methods](#) >

```
public abstract class Factor
extends java.lang.Object
```

A Factor is a table that given a {@link Tuple} of values returns a value. This is an abstract class that can be instantiated in many ways, for example the {@link FactorTimes}. Factor that is the product of other Factors.

Algorithm avoids storing tuples zero-valued Tuples, so Factors with only empty Tuples might be created during computation.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Methods

contains

```
public boolean contains(Variable variable)
```

Returns true if the Factor contains given variable.

Parameters:

variable -

Returns:

true if par is in the current factor.

getColumnSpacingWidth

```
public static int getColumnSpacingWidth()
```

Returns:

width of the column spacing in the string representation of the Factor.

getSavingForTracing

```
public static boolean getSavingForTracing()
```

Returns:

true if the intermediate Factors are being saved to allow for tracing of how a value was computed.

getVariable

```
public final Variable getVariable(int i)
```

Parameters:

i - index of the variable to be returned.

Returns:

i-th variable from the factor.

getVariables

```
public java.util.Iterator getVariables()
```

Returns:

iterator over variables.

getVariablesNum

```
public int getVariablesNum()
```

Returns:

number of variables.

iterator

```
public abstract EltsIterator iterator()
```

Returns:

an iterator over the Tuples of the Factor.

print

```
public void print()
```

Prints the Factor in a table form.

print

```
public void print(java.lang.String indent)
```

Prints the Factor in a table form.

Parameters:

indent - - String printed at the start of each line.

properlyOrdered

```
public boolean properlyOrdered()
```

Returns:

true if Factor Variables are in order.

setSavingForTracing

```
public static void setSavingForTracing(boolean val)
```

Sets the property that the intermediate Factors are being saved to allow for tracing of how a value was computed.

Parameters:

val - if true uses more space but allows for tracing; if false allows the intermediate Factors to be garbage collected.

size

```
public long size()
```

Returns:

the size of a factor table.

toString

```
public java.lang.String toString()
```

Overrides:

toString in class java.lang.Object

valuesIterator

```
public java.util.Iterator valuesIterator()
```

Returns:

an iterator over the values of the Factor.

valuesNumber

```
public int valuesNumber()
```

Returns:

number of values in the Factor

Class FactorCR

```
java.lang.Object
|
+--Factor
    |
    +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorStored
        |
        +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorCR
```

< [Constructors](#) >

```
public class FactorCR
extends ca.ubc.cs.kisynski.ve_in_hcsp.FactorStored
```

Factors that represent constraint relations. These explicitly store the Tuples in a table.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Constructors

FactorCR

```
public FactorCR(Variable variable)
    throws java.lang.IllegalArgumentException
```

Constructor.

Parameters:

variable - of the factor, which corresponds to a separated component of the Constraint Network

Throws:

java.lang.IllegalArgumentException - if Variable does not belong to a separated component of the Constraint Network.

FactorCR

```
public FactorCR(Variable variable0,
                Variable variable1)
    throws java.lang.IllegalArgumentException,
           java.lang.Error
```

Constructor.

Parameters:

variable0 - of the factor;
variable1 - of the factor.

Throws:

java.lang.IllegalArgumentException - if Variables are not in order;
java.lang.Error - if Variables do not belong to the same connected component of the Constraint Network.

FactorCR

```
public FactorCR(Variable variable0,  
               Variable variable1,  
               boolean sort)  
    throws java.lang.Error
```

Constructor.

Parameters:

variable0 - of the factor;
variable1 - of the factor;
sort - if true - sort Variables, otherwise allow invalid ordering (should be used with care!).

Throws:

java.lang.Error - if Variables do not belong to the same connected component of the Constraint Network.

Class FactorCRPretend

```
java.lang.Object  
|  
+--Factor  
    |  
    +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorStoredPretend  
        |  
        +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorCRPretend
```

< [Constructors](#) >

```
public class FactorCRPretend  
    extends ca.ubc.cs.kisynski.ve_in_hcsp.FactorStoredPretend
```

Factors that represent constraint relations, but we only pretend to store them. This is useful for, for example, finding the elimination ordering an algorithm would have used.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Constructors

FactorCRPretend

```
public FactorCRPretend(Variable variable)
```

Constructor.

Parameters:

variable - of the factor, which corresponds to separate component of the Constraint Network

FactorCRPretend

```
public FactorCRPretend(Variable variable1,  
                       Variable variable2)  
    throws java.lang.IllegalArgumentException
```

Constructor.

Parameters:

variable1 - of the factor;
variable2 - of the factor.

Throws:

java.lang.IllegalArgumentException - if Variables are not in order.

FactorCRPretend

```
public FactorCRPretend(Variable variable1,  
                       Variable variable2,  
                       boolean sort)
```

Constructor.

Parameters:

variable1 - of the factor;
variable2 - of the factor;
sort - if true - sort Variables, otherwise allows invalid ordering (should be used with care!).

Class FactorSumOut

```
java.lang.Object
|
+--Factor
    |
    +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorStored
        |
        +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorSumOut
```

< [Methods](#) >

```
public class FactorSumOut
extends ca.ubc.cs.kisynski.ve_in_hcsp.FactorStored
```

This the the class of Factors that are created by summing out a set of Variables from another Factor.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Methods

getTheFactor

```
public Factor getTheFactor()
    throws
    java.lang.UnsupportedOperationException
```

Returns:

the factor that the variables are being summed out from

Throws:

java.lang.UnsupportedOperationException - if Factor.getSavingForTracing() is false.

getTheVariables

```
public ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] getTheVariables()
    throws
    java.lang.UnsupportedOperationException
```

Returns:

the variables being summed out.

Throws:

java.lang.UnsupportedOperationException - if Factor.getSavingForTracing() is false.

Class FactorSumOutPretend

```
java.lang.Object
|
+--Factor
    |
    +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorStoredPretend
        |
        +--ca.ubc.cs.kisynski.ve_in_hcsp.FactorSumOutPretend
```

< [Methods](#) >

```
public class FactorSumOutPretend
    extends ca.ubc.cs.kisynski.ve_in_hcsp.FactorStoredPretend
```

This the the class of Factors that are created by summing out a set of Variables from another Factor, but we only pretend to store them. This is useful for, for example, finding the elimination ordering an algorithm would have used.

Properties:

- Changes by Jacek Kisynski:

- use of generics (update to J2SE 5.0).

*

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Methods

getTheFactor

```
public Factor getTheFactor()
                               throws
java.lang.UnsupportedOperationException
```

Returns:

the factor that the variables are being summed out from

Throws:

java.lang.UnsupportedOperationException - if Factor.getSavingForTracing() is false.

getTheVariables

```
public ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] getTheVariables()
                                                throws
java.lang.UnsupportedOperationException
```

Returns:

the variables being summed out.

Throws:

java.lang.UnsupportedOperationException - if Factor.getSavingForTracing() is false.

Class FactorTimes

```
java.lang.Object
|
+--Factor
|
+--ca.ubc.cs.kisynski.ve_in_hcsp.FactorTimes
```

< [Methods](#) >

public class **FactorTimes**
extends [Factor](#)

This is the class of Factors that are the product of other Factors. It is lazy in that it doesn't explicitly store the Factors. The Tuples can be accessed as needed using the {@link ca.ubc.cs.kisynski.ve_in_hcsp.EltsIterator}.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Methods

getTheFirstFactor

```
public Factor getTheFirstFactor()
```

Returns:

the first Factor in the product.

getTheSecondFactor

```
public Factor getTheSecondFactor()
```

Returns:

the second Factor in the product.

iterator

```
public EltsIterator iterator()
```

Returns:

an iterator over the Tuples of the Factor.

Overrides:

[iterator](#) in class [Factor](#)

Class Query

```
java.lang.Object
|
+--ca.ubc.cs.kisynski.ve_in_hcsp.Query
```

< [Constructors](#) > < [Methods](#) >

```
public class Query
extends java.lang.Object
```

A Factor that is the result of a query.

Algorithm avoids storing zero-valued Tuples, so Factor with only empty tuples might be returned as the result.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

1.2 26-October-2007

Constructors

Query

```
public Query(ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] queryVars,
             ConstraintNetwork cnet,
             ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] eo)
    throws java.lang.IllegalArgumentException
```

Constructor.

Parameters:

queryVars - the array of parameters being queried. These are assumed to be in order and must be a subset of the Variables in the Constraint Network;
cnet - a Constraint Network;
eo - elimination ordering.

Throws:

java.lang.IllegalArgumentException - if there are no Factors in the Network, queryVars are not in order or not a subset of Variables in the ConstraintNetwork or elimination ordering is inconsistent with queryVars.

Query

```
public Query(ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] queryVars,  
            ConstraintNetwork cnet,  
            java.lang.String how)  
    throws java.lang.IllegalArgumentException
```

Constructor.

Parameters:

queryVars - the array of Variables being queried. These are assumed to be in order and must be a subset of the Variables in the Constraint Network;
cnet - a Constraint Network;
how - a heuristic: random, sequential, min-factor-tuples (=min-weight-tuples), min-factor-values (=min-weight-values), min-degree (=min-size), min-fill (=min-deficiency, min-discrepancy) (DEFAULT), max-cardinality.

Throws:

java.lang.IllegalArgumentException - if there are no Factors in the Network, queryVars are not in order or not a subset of Variables in the ConstraintNetwork.

Methods

getEH

```
public java.lang.String getEH()
```

Returns:

elimination heuristic.

getEO

```
public ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] getEO()
```

Returns:

elimination ordering.

getMaxFactorSize

```
public long getMaxFactorSize()
```

Returns:

the size of the biggest factor.

getResult

```
public Factor getResult()
```

Returns:

the result of the query.

Class QueryPretend

```
java.lang.Object  
|  
+--ca.ubc.cs.kisynski.ve_in_hcsp.QueryPretend
```

< [Constructors](#) > < [Methods](#) >

```
public class QueryPretend  
extends java.lang.Object
```

A factor that is the result of a query, but we only pretend to compute it. This is useful for, for example, finding the elimination ordering an algorithm would have used.

Algorithm avoids storing zero-valued Tuples, so Factor with only empty tuples might be returned as the result.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski (adaptation for #CSP, update to Java2) & David Poole (original code)

Version:

Constructors

QueryPretend

```
public QueryPretend(ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] queryVars,  
                   ConstraintNetwork cnet,  
                   java.lang.String how)  
    throws java.lang.IllegalArgumentException
```

Constructor.

Parameters:

queryVars - the array of Variables being queried. These are assumed to be in order and must be a subset of the Variables in the Constraint Network;
cnet - a Constraint Network;
how - a heuristic: random, sequential, min-factor-tuples (=min-weight-tuples), min-factor-values (=min-weight-values), min-degree (=min-size), min-fill (=min-deficiency, min-discrepancy) (DEFAULT), max-cardinality.

Throws:

java.lang.IllegalArgumentException - if there are no Factors in the Network, queryVars are not in order or not a subset of Variables in the ConstraintNetwork.

Methods

getEH

```
public java.lang.String getEH()
```

Returns:

elimination heuristic.

getEO

```
public ca.ubc.cs.kisynski.ve_in_hcsp.Variable[] getEO()
```

Returns:

elimination ordering.

getMaxFactorSize

```
public long getMaxFactorSize()
```

Returns:

the size of the biggest factor.

getResult

```
public Factor getResult()
```

Returns:

the result of the query.

Class SetConstant

```
java.lang.Object  
|  
+--ca.ubc.cs.kisynski.ve_in_hcsp.SetConstant
```

All Implemented Interfaces:

java.lang.Comparable

Direct Known Subclasses:

[SetConstantStandard](#)

< [Constructors](#) > < [Methods](#) >

```
public abstract class SetConstant  
extends java.lang.Object  
implements java.lang.Comparable
```

Set constant representing subset of elements from Variable(s) domain(s). Subset is disjoint with other subsets.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY

WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Constructors

SetConstant

```
public SetConstant()
```

Constructor.

Methods

addElement

```
public abstract void addElement(java.lang.String element)
```

Parameters:

element - to be added to the subset represented by the SetConstant..

compareTo

```
public int compareTo(SetConstant object)
```

equals

```
public boolean equals(java.lang.Object object)
```

Overrides:

equals in class java.lang.Object

getCount

```
public abstract int getCount()
```

Returns:

number of the elements in the subset represented by the SetConstant.

getElements

```
public abstract ca.ubc.cs.kisynski.utils.ItrSafe getElements()
```

Returns:

an iterator over elements of the subset represented by the SetConstant.

hashCode

```
public int hashCode()
```

Overrides:

hashCode in class java.lang.Object

toString

```
public java.lang.String toString()
```

Overrides:

toString in class java.lang.Object

Class SetConstantStandard

```
java.lang.Object
|
+-- SetConstant
     |
     +-- ca.ubc.cs.kisynski.ve_in_hcsp.SetConstantStandard
```

All Implemented Interfaces:

java.lang.Comparable

< [Constructors](#) > < [Methods](#) >

```
public class SetConstantStandard
extends SetConstant
```

Set constant representing subset of elements from Variable(s) domain(s). Class created for tests. It is assumed that it won't be mixed with Set constants from different subclasses.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Constructors

SetConstantStandard

```
public SetConstantStandard()
```

Constructor.

SetConstantStandard

```
public SetConstantStandard(java.util.List elements)
```

Constructor.

Parameters:

elements - non-empty list of elements (that will be stored as Strings).

Methods

addElement

```
public void addElement(java.lang.String element)
```

Overrides:

[addElement](#) in class [SetConstant](#)

getCount

```
public int getCount()
```

Overrides:

[getCount](#) in class [SetConstant](#)

getElements

```
public ca.ubc.cs.kisynski.utils.ItrSafe getElements()
```

Overrides:

[getElements](#) in class [SetConstant](#)

Class Tuple

```
java.lang.Object  
|  
+--ca.ubc.cs.kisynski.ve_in_hcsp.Tuple
```

< [Methods](#) >

```
public class Tuple  
extends java.lang.Object
```

A Tuple is an element of the {@link Factor}. Tuples (and their members) are shared between different Factors, so they can not change internally. If this.values == null, then Tuple has zero value and this.partitions = null.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Methods

getValues

```
public ca.ubc.cs.kisynski.utils.ItrSafe getValues()
```

Returns:

iterator over values of the tuple.

size

```
public int size()
```

Returns:

number of values.

Class Variable

```
java.lang.Object  
|  
+--ca.ubc.cs.kisynski.ve_in_hcsp.Variable
```

All Implemented Interfaces:

java.lang.Comparable

< [Methods](#) >

```
public class Variable
extends java.lang.Object
implements java.lang.Comparable
```

Variable from the Constraint Network.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Methods

compareTo

```
public int compareTo(Variable object)
```

equals

```
public boolean equals(java.lang.Object object)
```

Overrides:

equals in class java.lang.Object

getId

```
public int getId()
```

Returns:

Id of the Variable.

getIndividuals

```
public java.util.Iterator getIndividuals()
```

Returns:

iterator over domain of the variable.

getName

```
public java.lang.String getName()
```

Returns:

name of the Variable.

getParametrizedDomain

```
public java.util.Iterator getParametrizedDomain()
```

Does not provide opportunity to remove elements from the container.

Returns:

Iterator over SetConstants representing domain of the Variable.

getParametrizedDomainSize

```
public int getParametrizedDomainSize()
```

Returns:

number of SetConstants representing domain of the Variable.

getSize

```
public int getSize()
```

Returns:

size of the domain of the Variable.

hashCode

```
public int hashCode()
```

Overrides:

hashCode in class java.lang.Object

toString

```
public java.lang.String toString()
```

Overrides:

toString in class java.lang.Object

Class `_Paper`

```
java.lang.Object
|
+--ca.ubc.cs.kisynski.ve_in_hcsp._Paper
```

< [Constructors](#) > < [Methods](#) >

```
public class _Paper
extends java.lang.Object
```

Example from the #CSP paper.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the

terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.ve_in_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Constructors

_Paper

```
public _Paper()
```

Methods

main

```
public static void main(java.lang.String[] args)
```

Parameters:

args - not used.

Class _Test

```
java.lang.Object
|
+-- java.awt.Component
|   |
|   +-- java.awt.Container
|       |
|       +-- java.awt.Panel
|           |
|           +-- java.applet.Applet
|               |
|               +-- javax.swing.JApplet
|                   |
|                   +-- ca.ubc.cs.kisynski.ve_in_hcsp._Test
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.image.ImageObserver, java.io.Serializable,
javax.accessibility.Accessible, javax.swing.RootPaneContainer,
javax.swing.TransferHandler.HasGetTransferHandler

< [Constructors](#) > < [Methods](#) >

```
public class _Test
extends javax.swing.JApplet
```

A simple Test of ca.ubc.cs.kisynski.ve_in_hcsp package.

Properties:

- Changes by Jacek Kisynski:
- use of generics (update to J2SE 5.0).

This file is part of ca.ubc.cs.kisynski.ve_in_hcsp package.

ca.ubc.cs.kisynski.ve_in_hcsp package is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

ca.ubc.cs.kisynski.FOve_hcsp package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with ca.ubc.cs.kisynski.ve_in_hcsp package; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Author:

Jacek Kisynski

Version:

1.2 26-October-2007

Constructors

_Test

```
public _Test()
```

Methods

init

```
public void init()
```

This method initializes this applet.

Overrides:

init in class java.applet.Applet

main

```
public static void main(java.lang.String[] args)
```

Main method so it can be run as an application.

Parameters:

args - not used.

INDEX

A

[addElement](#) ... 29
[addElement](#) ... 32

B

[backTo](#) ... 10

C

[compareTo](#) ... 29
[compareTo](#) ... 34
[construct](#) ... 5
[construct](#) ... 6
[contains](#) ... 12
[currPos](#) ... 10
[ConstraintNetwork](#) ... 2
[ConstraintNetworkFromCSPif](#) ... 4
[ConstraintNetworkFromInequalityRelations](#) ... 6
[ConstraintNetworkFromInequalityRelations](#) ... 7
[ConstraintNetworkFromInequalityRelationsPretend](#)
... 8
[ConstraintNetworkFromInequalityRelationsPretend](#)
... 9

E

[equals](#) ... 29
[equals](#) ... 34
[EltIterator](#) ... 9

F

[Factor](#) ... 11
[FactorCR](#) ... 15
[FactorCR](#) ... 16
[FactorCR](#) ... 16
[FactorCR](#) ... 17
[FactorCRPretend](#) ... 17
[FactorCRPretend](#) ... 18
[FactorCRPretend](#) ... 18
[FactorCRPretend](#) ... 18
[FactorSumOut](#) ... 19
[FactorSumOutPretend](#) ... 20
[FactorTimes](#) ... 22

G

[getColumnSpacingWidth](#) ... 12
[getCount](#) ... 30
[getCount](#) ... 32
[getEH](#) ... 25
[getEH](#) ... 27
[getElements](#) ... 30
[getElements](#) ... 32
[getEQ](#) ... 25
[getEQ](#) ... 27
[getFactors](#) ... 3
[getFactorsNum](#) ... 3
[getId](#) ... 35
[getIndividuals](#) ... 35
[getMaxFactorSize](#) ... 25
[getMaxFactorSize](#) ... 28
[getName](#) ... 35
[getParametrizedDomain](#) ... 35
[getParametrizedDomainSize](#) ... 35
[getResult](#) ... 26
[getResult](#) ... 28
[getSavingForTracing](#) ... 12
[getSize](#) ... 36
[getTheFactor](#) ... 20
[getTheFactor](#) ... 21
[getTheFirstFactor](#) ... 23
[getTheSecondFactor](#) ... 23
[getTheVariables](#) ... 20
[getTheVariables](#) ... 21
[getValues](#) ... 33
[getVariable](#) ... 13
[getVariables](#) ... 3
[getVariables](#) ... 13
[getVariablesNum](#) ... 3
[getVariablesNum](#) ... 13

H

[hashCode](#) ... 30
[hashCode](#) ... 36
[hasNext](#) ... 10

I

[init](#) ... 39
[iterator](#) ... 13
[iterator](#) ... 23

M

[main](#) ... 37
[main](#) ... 39

N

[next](#) ... 11

P

[print](#) ... 3
[print](#) ... 4
[print](#) ... 13
[print](#) ... 14
[printBrief](#) ... 4
[printBrief](#) ... 4
[properlyOrdered](#) ... 14

Q

[Query](#) ... 23
[Query](#) ... 24
[Query](#) ... 25
[QueryPretend](#) ... 26
[QueryPretend](#) ... 27

S

[setSavingForTracing](#) ... 14
[size](#) ... 14
[size](#) ... 33
[SetConstant](#) ... 28
[SetConstant](#) ... 29
[SetConstantStandard](#) ... 30
[SetConstantStandard](#) ... 31
[SetConstantStandard](#) ... 31

T

[toString](#) ... 14
[toString](#) ... 30
[toString](#) ... 36
[Tuple](#) ... 32

V

[valuesIterator](#) ... 15
[valuesNumber](#) ... 15
[Variable](#) ... 33