NAME

TopologicalAtomTorsionsFingerprints

SYNOPSIS

use Fingerprints::TopologicalAtomTorsionsFingerprints;

use Fingerprints::TopologicalAtomTorsionsFingerprints qw(:all);

DESCRIPTION

TopologicalAtomTorsionsFingerprints class provides the following methods:

new, GenerateFingerprints, GetAtomTorsionsIDs, GetDescription, SetAtomIdentifierType, SetAtomicInvariantsToUse, SetFunctionalClassesToUse, StringifyTopologicalAtomTorsionsFingerprints

TopologicalAtomTorsionsFingerprints is derived from Fingerprints class which in turn is derived from ObjectProperty base class that provides methods not explicitly defined in TopologicalAtomTorsionsFingerprints, Fingerprints or ObjectProperty classes using Perl's AUTOLOAD functionality. These methods are generated on-the-fly for a specified object property:

```
Set<PropertyName>(<PropertyValue>);
$PropertyValue = Get<PropertyName>();
Delete<PropertyName>();
```

The current release of MayaChemTools supports generation of TopologicalAtomTorsionsFingerprints corresponding to following AtomtomIdentifierTypes:

```
AtomicInvariantsAtomTypes, DREIDINGAtomTypes, EStateAtomTypes,
FunctionalClassAtomTypes, MMFF94AtomTypes, SLogPAtomTypes,
SYBYLAtomTypes, TPSAAtomTypes, UFFAtomTypes
```

Based on the values specified for AtomIdentifierType along with other specified parameters such as AtomicInvariantsToUse and FunctionalClassesToUse, initial atom types are assigned to all non-hydrogen in a molecule. All unique atom torsions are identified and an atom torsion identifier is generated; the format of atom torsion identifier is:

```
<AtomType1>-<AtomType2>-<AtomType3>-<AtomType4>

AtomType1, AtomType2, AtomType3, AtomTyp4: Assigned atom types

where AtomType1 <= AtomType2 <= AtomType3 <= AtomType4
```

The atom torsion identifiers for all unique atom torsions corresponding to non-hydrogen atoms constitute topological atom torsions fingerprints of the molecule.

The current release of MayaChemTools generates the following types of topological atom torsions fingerprints vector strings:

```
FingerprintsVector;TopologicalAtomTorsions:AtomicInvariantsAtomTypes;3
3;NumericalValues;IDsAndValuesString;C.X1.BO1.H3-C.X3.BO3.H1-C.X3.BO4-
C.X3.BO4 C.X1.BO1.H3-C.X3.BO3.H1-C.X3.BO4-N.X3.BO3 C.X2.BO2.H2-C.X2.BO
2.H2-C.X3.BO3.H1-C.X2.BO2.H2 C.X2.BO2.H2-C.X2.BO2.H2-C.X3.BO3.H1-O...;
2 2 1 1 2 2 1 1 3 4 4 8 4 2 2 6 2 2 1 2 1 1 2 1 1 2 6 2 4 2 1 3 1

FingerprintsVector;TopologicalAtomTorsions:AtomicInvariantsAtomTypes;3
3;NumericalValues;IDsAndValuesPairsString;C.X1.BO1.H3-C.X3.BO3.H1-C.X3
.BO4-C.X3.BO4 2 C.X1.BO1.H3-C.X3.BO3.H1-C.X3.BO4-N.X3.BO3 2 C.X2.BO2.H
2-C.X2.BO2.H2-C.X3.BO3.H1-C.X2.BO2.H2 1 C.X2.BO2.H2-C.X2.BO2.H2-C.X3.B
O3.H1-O.X1.BO1.H1 1 C.X2.BO2.H2-C.X2.BO2.H2-N.X3.BO3-C.X3.BO4 2 C.X2.B
O2.H2-C.X3.BO3.H1-C.X2.BO2.H2-C.X3.BO3.H1 2 C.X2.BO2.H2-C.X3.BO3.H1...

FingerprintsVector;TopologicalAtomTorsions:DREIDINGAtomTypes;27;Numeri
calValues;IDsAndValuesString;C_2-C_3-C_3-C_3 C_2-C_3-C_3-O_3 C_2-C_R-C
_R-C_3 C_2-C_R-C_R-C_R C_2-C_R-C_R-N_R C_2-N_3-C_R-C_R C_3-C_3-C_2-O_2
C_3-C_3-C_2-O_3 C_3-C_3-C_3-C_3 C_3-C_3-C_3-N_R C_3-C_3-C_3-O_3 C_...;
1 1 1 2 1 2 1 1 3 1 3 2 2 2 1 1 1 3 1 2 2 32 2 2 5 3 1

FingerprintsVector;TopologicalAtomTorsions:EStateAtomTypes;36;Numerica
lValues;IDsAndValuesString;aaCH-aaCH-aaCH-aaCH aaCH-aaCH-aaCH-aasC aaC
H-aaCH-aasC-aaCH aaCH-aaCH-aasC-aasC aaCH-aaCH-aasC-sF aaCH-aaCH-aasC-
ssNH aaCH-aasC-aasC-aasC aaCH-aasC-aasC-aasN aaCH-aasC-ssNH-dssC a...;
4 4 8 4 2 2 6 2 2 2 4 3 2 1 3 3 2 2 2 1 2 1 1 1 2 1 1 1 1 1 1 1 2 1 1 2

FingerprintsVector;TopologicalAtomTorsions:FunctionalClassAtomTypes;26
```

```
;NumericalValues;IDsAndValuesString;Ar-Ar-Ar-Ar Ar-Ar-Ar-Ar.HBA Ar-Ar-
Ar-HBD Ar-Ar-Ar-Hal Ar-Ar-Ar-None Ar-Ar-Ar.HBA-Ar Ar-Ar-Ar.HBA-None Ar
-Ar-HBD-None Ar-Ar-None-HBA Ar-Ar-None-HBD Ar-Ar-None-None Ar-Ar.H...;
32 5 2 2 3 3 3 2 2 2 2 1 2 1 1 1 2 1 1 1 1 3 1 1 1 3

FingerprintsVector;TopologicalAtomTorsions:MMFF94AtomTypes;43;Numerica
lValues;IDsAndValuesString;C5A-C5B-C5B-C5A C5A-C5B-C5B-C=ON C5A-C5B-C5
B-CB C5A-C5B-C=ON-NC=O C5A-C5B-C=ON-O=CN C5A-C5B-CB-CB C5A-CB-CB-CB C5
A-N5-C5A-C5B C5A-N5-C5A-CB C5A-N5-C5A-CR C5A-N5-CR-CR C5B-C5A-CB-C...;
1 1 1 1 1 2 2 2 1 1 2 2 2 2 1 1 2 1 1 2 1 2 1 1 1 2 1 1 1 2 18 2 2 1 1
1 1 2 1 1 3 1 3

FingerprintsVector;TopologicalAtomTorsions:SLogPAtomTypes;49;Numerical
Values;IDsAndValuesPairsString;C1-C10-N11-C20 1 C1-C10-N11-C21 1 C1-C1
1-C21-C21 2 C1-C11-C21-N11 2 C1-CS-C1-C10 1 C1-CS-C1-C5 1 C1-CS-C1-CS
2 C10-C1-CS-O2 1 C10-N11-C20-C20 2 C10-N11-C21-C11 1 C10-N11-C21-C21 1
C11-C21-C21-C20 1 C11-C21-C21-C5 1 C11-C21-N11-C20 1 C14-C18-C18-C20
2 C18-C14-C18-C18 2 C18-C18-C14-F 2 C18-C18-C18-C18 4 C18-C18-C18-C...

FingerprintsVector;TopologicalAtomTorsions:SYBYLAtomTypes;26;Numerical
Values;IDsAndValuesPairsString;C.2-C.3-C.3-C.3 1 C.2-C.3-C.3-O.3 1 C.2
-C.ar-C.ar-C.3 1 C.2-C.ar-C.ar-C.ar 2 C.2-C.ar-C.ar-N.ar 1 C.2-N.am-C.
ar-C.ar 2 C.3-C.3-C.2-O.co2 2 C.3-C.3-C.3-C.3 3 C.3-C.3-C.3-N.ar 1 C.3
-C.3-C.3-O.3 3 C.3-C.3-C.ar-C.ar 2 C.3-C.3-C.ar-N.ar 2 C.3-C.3-N.ar-C.
ar 2 C.3-C.ar-C.ar-C.ar 1 C.3-C.ar-N.ar-C.3 1 C.3-C.ar-N.ar-C.ar 1 ...

FingerprintsVector;TopologicalAtomTorsions:TPSAAtomTypes;8;NumericalVa
lues;IDsAndValuesPairsString;N21-None-None-None 9 N7-None-None-None 4
None-N21-None-None 10 None-N7-None-None 3 None-N7-None-O3 1 None-None-
None-None 44 None-None-None-O3 3 None-None-None-O4 5

FingerprintsVector;TopologicalAtomTorsions:UFFAtomTypes;27;NumericalVa
lues;IDsAndValuesPairsString;C_2-C_3-C_3-C_3 1 C_2-C_3-C_3-O_3 1 C_2-C
_R-C_R-C_3 1 C_2-C_R-C_R-C_R 2 C_2-C_R-C_R-N_R 1 C_2-N_3-C_R-C_R 2 C_3
-C_3-C_2-O_2 1 C_3-C_3-C_2-O_3 1 C_3-C_3-C_3-C_3 3 C_3-C_3-C_3-N_R 1 C
_3-C_3-C_3-O_3 3 C_3-C_3-C_R-C_R 2 C_3-C_3-C_R-N_R 2 C_3-C_3-N_R-C_R 2
 C_3-C_R-C_R-C_R 1 C_3-C_R-N_R-C_3 1 C_3-C_R-N_R-C_R 1 C_3-N_R-C_R-...
```

METHODS

new

```
$NewTopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                                                %NamesAndValues);
```

Using specified *TopologicalAtomTorsionsFingerprints* property names and values hash, new method creates a new object and returns a reference to newly created TopologicalAtomTorsionsFingerprints object. By default, the following properties are initialized:

```
Molecule = ''
Type = 'TopologicalAtomTorsions'
AtomIdentifierType = ''
AtomicInvariantsToUse = ['AS', 'X', 'BO', 'H', 'FC']
FunctionalClassesToUse = ['HBD', 'HBA', 'PI', 'NI', 'Ar', 'Hal']
```

Examples:

```
$TopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                            'Molecule' => $Molecule,
                            'AtomIdentifierType' =>
                                        'AtomicInvariantsAtomTypes');

$TopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                            'Molecule' => $Molecule,
                            'AtomIdentifierType' =>
                                        'AtomicInvariantsAtomTypes',
                            'AtomicInvariantsToUse' =>
                                        ['AS', 'X', 'BO', 'H', 'FC'] );

$TopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                            'Molecule' => $Molecule,
                            'AtomIdentifierType' =>
                                        'DREIDINGAtomTypes');
```

```
$TopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                          'Molecule' => $Molecule,
                          'AtomIdentifierType' =>
                                        'SYBYLAtomTypes');

$TopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                          'Molecule' => $Molecule,
                          'AtomIdentifierType' =>
                                        'SLogPAtomTypes');

$TopologicalAtomTorsionsFingerprints = new TopologicalAtomTorsionsFingerprints(
                          'Molecule' => $Molecule,
                          'AtomIdentifierType' =>
                                        'FunctionalClassAtomTypes',
                          'FunctionalClassesToUse' =>
                                        ['HBD', 'HBA', 'PI', 'NI', 'Ar', 'Hal'] );

$TopologicalAtomTorsionsFingerprints->GenerateFingerprints();
print "$TopologicalAtomTorsionsFingerprints\n";
```

GetDescription

```
$Description = $TopologicalAtomTorsionsFingerprints->GetDescription();
```

Returns a string containing description of topological atom torsions fingerprints.

GenerateFingerprints

```
$TopologicalAtomTorsionsFingerprints->GenerateFingerprints();
```

Generates topological atom torsions fingerprints and returns *TopologicalAtomTorsionsFingerprints*.

GetAtomTorsionsIDs

```
$AtomPairIDsRef = $TopologicalAtomTorsionsFingerprints->GetAtomTorsionsIDs();
@AtomPairIDs = $TopologicalAtomTorsionsFingerprints->GetAtomTorsionsIDs();
```

Returns atom torsion IDs corresponding to atom torsion count values in topological atom torsions fingerprints vector as an array or reference to an array.

SetAtomIdentifierType

```
$TopologicalAtomTorsionsFingerprints->SetAtomIdentifierType($IdentifierType);
```

Sets atom *IdentifierType* to use during atom torsions fingerprints generation and returns *TopologicalAtomTorsionsFingerprints*.

Possible values: *AtomicInvariantsAtomTypes*, *DREIDINGAtomTypes*, *EStateAtomTypes*, *FunctionalClassAtomTypes*, *MMFF94AtomTypes*, *SLogPAtomTypes*, *SYBYLAtomTypes*, *TPSAAtomTypes*, *UFFAtomTypes*.

SetAtomicInvariantsToUse

```
$TopologicalAtomTorsionsFingerprints->SetAtomicInvariantsToUse($ValuesRef);
$TopologicalAtomTorsionsFingerprints->SetAtomicInvariantsToUse(@Values);
```

Sets atomic invariants to use during *AtomicInvariantsAtomTypes* value of *AtomIdentifierType* for topological atom torsions fingerprints generation and returns *TopologicalAtomTorsionsFingerprints*.

Possible values for atomic invariants are: *AS*, *X*, *BO*, *LBO*, *SB*, *DB*, *TB*, *H*, *Ar*, *RA*, *FC*, *MN*, *SM*. Default value: *AS,X,BO,H,FC*.

The atomic invariants abbreviations correspond to:

```
AS = Atom symbol corresponding to element symbol

X<n>   = Number of non-hydrogen atom neighbors or heavy atoms
BO<n> = Sum of bond orders to non-hydrogen atom neighbors or heavy atoms
LBO<n> = Largest bond order of non-hydrogen atom neighbors or heavy atoms
SB<n> = Number of single bonds to non-hydrogen atom neighbors or heavy atoms
DB<n> = Number of double bonds to non-hydrogen atom neighbors or heavy atoms
TB<n> = Number of triple bonds to non-hydrogen atom neighbors or heavy atoms
H<n>   = Number of implicit and explicit hydrogens for atom
Ar     = Aromatic annotation indicating whether atom is aromatic
RA     = Ring atom annotation indicating whether atom is a ring
FC<+n/-n> = Formal charge assigned to atom
MN<n> = Mass number indicating isotope other than most abundant isotope
SM<n> = Spin multiplicity of atom. Possible values: 1 (singlet), 2 (doublet) or
        3 (triplet)
```

Atom type generated by AtomTypes::AtomicInvariantsAtomTypes class corresponds to:

```
AS.X<n>.BO<n>.LBO<n>.<SB><n>.<DB><n>.<TB><n>.H<n>.Ar.RA.FC<+n/-n>.MN<n>.SM<n>
```

Except for AS which is a required atomic invariant in atom types, all other atomic invariants are optional. Atom type specification doesn't include atomic invariants with zero or undefined values.

In addition to usage of abbreviations for specifying atomic invariants, the following descriptive words are also allowed:

```
X : NumOfNonHydrogenAtomNeighbors or NumOfHeavyAtomNeighbors
BO : SumOfBondOrdersToNonHydrogenAtoms or SumOfBondOrdersToHeavyAtoms
LBO : LargestBondOrderToNonHydrogenAtoms or LargestBondOrderToHeavyAtoms
SB :  NumOfSingleBondsToNonHydrogenAtoms or NumOfSingleBondsToHeavyAtoms
DB : NumOfDoubleBondsToNonHydrogenAtoms or NumOfDoubleBondsToHeavyAtoms
TB : NumOfTripleBondsToNonHydrogenAtoms or NumOfTripleBondsToHeavyAtoms
H :  NumOfImplicitAndExplicitHydrogens
Ar : Aromatic
RA : RingAtom
FC : FormalCharge
MN : MassNumber
SM : SpinMultiplicity
```

AtomTypes::AtomicInvariantsAtomTypes module is used to assign atomic invariant atom types.

SetFunctionalClassesToUse

```
$TopologicalTorsionsFingerprints->SetFunctionalClassesToUse($ValuesRef);
$TopologicalTorsionsFingerprints->SetFunctionalClassesToUse(@Values);
```

Sets functional classes invariants to use during *FunctionalClassAtomTypes* value of *AtomIdentifierType* for topological atom torsions fingerprints generation and returns *TopologicalAtomTorsionsFingerprints*.

Possible values for atom functional classes are: *Ar, CA, H, HBA, HBD, Hal, NI, PI, RA*. Default value [Ref 24]: *HBD,HBA,PI,NI,Ar,Hal*.

The functional class abbreviations correspond to:

```
HBD: HydrogenBondDonor
HBA: HydrogenBondAcceptor
PI :  PositivelyIonizable
NI : NegativelyIonizable
Ar : Aromatic
Hal : Halogen
H : Hydrophobic
RA : RingAtom
CA : ChainAtom
```

```
Functional class atom type specification for an atom corresponds to:

Ar.CA.H.HBA.HBD.Hal.NI.PI.RA or None
```

AtomTypes::FunctionalClassAtomTypes module is used to assign functional class atom types. It uses following definitions [Ref 60-61, Ref 65-66]:

```
HydrogenBondDonor: NH, NH2, OH
HydrogenBondAcceptor: N[!H], O
PositivelyIonizable: +, NH2
NegativelyIonizable: -, C(=O)OH, S(=O)OH, P(=O)OH
```

StringifyTopologicalAtomTorsionsFingerprints

```
$String = $TopologicalAtomTorsionsFingerprints->
              StringifyTopologicalAtomTorsionsFingerprints();
```

Returns a string containing information about *TopologicalAtomTorsionsFingerprints* object.

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

Fingerprints.pm, FingerprintsStringUtil.pm, AtomNeighborhoodsFingerprints.pm, AtomTypesFingerprints.pm, EStateIndiciesFingerprints.pm, ExtendedConnectivityFingerprints.pm, MACCSKeys.pm, PathLengthFingerprints.pm, TopologicalAtomPairsFingerprints.pm, TopologicalAtomTripletsFingerprints.pm, TopologicalPharmacophoreAtomPairsFingerprints.pm, TopologicalPharmacophoreAtomTripletsFingerprints.pm

COPYRIGHT

Copyright (C) 2017 Manish Sud. All rights reserved.

This file is part of MayaChemTools.

MayaChemTools is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.