NAME

PathLengthFingerprints

SYNOPSIS

use Fingerprints::PathLengthFingerprints;

use Fingerprints::PathLengthFingerprints qw(:all);

DESCRIPTION

PathLengthFingerprints class provides the following methods:

new, GenerateFingerprints, , GetDescription, SetAtomIdentifierType, SetAtomicInvariantsToUse, SetFunctionalClassesToUse, SetMaxLength, SetMinLength, SetNumOfBitsToSetPerPath, SetType, StringifyPathLengthFingerprints

PathLengthFingerprints is derived from Fingerprints class which in turn is derived from ObjectProperty base class that provides methods not explicitly defined in PathLengthFingerprints, 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 AtomTypesFingerpritns corresponding to following AtomtomIdentifierTypes:

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

Based on the values specified for Type, AtomtomIdentifierTypes, MinPathLength and MaxPathLength, all appropriate atom paths are generated for each atom in the molecule and collected in a list and the list is filtered to remove any structurally duplicate paths as indicated by the value of UseUniquePaths.

For molecules containing rings, atom paths starting from each atom can be traversed in four different ways:

```
o Atom paths without any rings and sharing of bonds in traversed paths.
o Atom paths containing rings and without any sharing of bonds in
  traversed paths
o All possible atom paths without any rings and sharing of bonds in
  traversed paths
o All possible atom paths containing rings and with sharing of bonds in
  traversed paths.
```

Atom path traversal is terminated at the last ring atom. For molecules containing no rings, first two and last two types described above are equivalent.

AllowSharedBonds and AllowRings allow generation of different types of paths to be used for fingerprints generation.

The combination of AllowSharedBonds, AllowRings, and UseBondSymbols allows generation of 8 different types of path length fingerprints:

```
AllowSharedBonds AllowRings UseBondSymbols

0                0          1   - AtomPathsNoCyclesWithBondSymbols
0                1          1   - AtomPathsWithCyclesWithBondSymbols

1                0          1   - AllAtomPathsNoCyclesWithBondSymbols
1                1          1   - AllAtomPathsWithCyclesWithBondSymbols
                                    [ DEFAULT ]

0                0          0   - AtomPathsNoCyclesNoBondSymbols
0                1          0   - AtomPathsWithCyclesNoBondSymbols

1                0          0   - AllAtomPathsNoCyclesNoBondSymbols
```

```
1               1          0    - AllAtomPathsWithCyclesNoWithBondSymbols
```

Additionally, possible values for option --AtomIdentifierType in conjunction with corresponding specified values for AtomicInvariantsToUse and FunctionalClassesToUse changes the nature of atom path length strings and the fingerprints.

For each atom path in the filtered atom paths list, an atom path string is created using value of AtomIdentifierType and specified values to use for a particular atom identifier type. Value of UseBondSymbols controls whether bond order symbols are used during generation of atom path string. Atom symbol corresponds to element symbol and characters used to represent bond order are: *1 - None; 2 - '='; 3 - '#'; 1.5 or aromatic - ':'; others: bond order value*. By default, bond symbols are included in atom path strings. Exclusion of bond symbols in atom path strings results in fingerprints which correspond purely to atom paths without considering bonds.

UseUniquePaths controls the removal of structurally duplicate atom path strings are removed from the list.

For *PathLengthBits* value of Type, each atom path is hashed to a 32 bit unsigned integer key using TextUtil::HashCode function. Using the hash key as a seed for a random number generator, a random integer value between 0 and Size is used to set corresponding bits in the fingerprint bit-vector string. Value of NumOfBitsToSetPerPaths option controls the number of time a random number is generated to set corresponding bits.

For *PathLengthCount* value of Typen, the number of times an atom path appears is tracked and a fingerprints count-string corresponding to count of atom paths is generated.

The current release of MayaChemTools generates the following types of path length fingerprints bit-vector and vector strings:

```
FingerprintsBitVector;PathLengthBits:AtomicInvariantsAtomTypes:MinLeng
th1:MaxLength8;1024;BinaryString;Ascending;0010000100110101010110001100
0100010101011000101001011100110001000010001001101000001001001001001000
0010110100000111001001000001001010100100100000000011000000101001011100
0010000001000101010100000100111100110111011011011000000010110111001101
0101100011000000010001000011000010100011101100001000001000100000000...

FingerprintsBitVector;PathLengthBits:AtomicInvariantsAtomTypes:MinLeng
th1:MaxLength8;1024;HexadecimalString;Ascending;48caa1315d82d91122b029
42861c9409a4208182d12015509767bd0867653604481a8b1288000056090583603078
9cedae54e26596889ab1213098009004905152242084215021 20a0dd9200509723ae89
00024181b86c0122821d4e4880c38620dab280824b455404009f082003d52c212b4e6d
6ea05280140069c780290c43

FingerprintsVector;PathLengthCount:AtomicInvariantsAtomTypes:MinLength
1:MaxLength8;432;NumericalValues;IDsAndValuesPairsString;C.X1.BO1.H3 2
C.X2.BO2.H2 4 C.X2.BO3.H1 14 C.X3.BO3.H1 3 C.X3.BO4 10 F.X1.BO1 1 N.X
2.BO2.H1 1 N.X3.BO3 1 O.X1.BO1.H1 3 O.X1.BO2 2 C.X1.BO1.H3C.X3.BO3.H1
2 C.X2.BO2.H2C.X2.BO2.H2 1 C.X2.BO2.H2C.X3.BO3.H1 4 C.X2.BO2.H2C.X3.BO
4 1 C.X2.BO2.H2N.X3.BO3 1 C.X2.BO3.H1:C.X2.BO3.H1 10 C.X2.BO3.H1:C....

FingerprintsVector;PathLengthCount:DREIDINGAtomTypes:MinLength1:MaxLen
gth8;410;NumericalValues;IDsAndValuesPairsString;C_2 2 C_3 9 C_R 22 F_
1 N_3 1 N_R 1 O_2 2 O_3 3 C_2=O_2 2 C_2C_3 1 C_2C_R 1 C_2N_3 1 C_2O_3
1 C_3C_3 7 C_3C_R 1 C_3N_R 1 C_3O_3 2 C_R:C_R 21 C_R:N_R 2 C_RC_R 2 C
_RF_ 1 C_RN_3 1 C_2C_3C_3 1 C_2C_R:C_R 2 C_2N_3C_R 1 C_3C_2=O_2 1 C_3C
_2O_3 1 C_3C_3C_3 5 C_3C_3C_R 2 C_3C_3N_R 1 C_3C_3O_3 4 C_3C_R:C_R ...

FingerprintsVector;PathLengthCount:EStateAtomTypes:MinLength1:MaxLengt
h8;454;NumericalValues;IDsAndValuesPairsString;aaCH 14 aasC 8 aasN 1 d
O 2 dssC 2 sCH3 2 sF 1 sOH 3 ssCH2 4 ssNH 1 sssCH 3 aaCH:aaCH 10 aaCH:
aasC 8 aasC:aasC 3 aasC:aasN 2 aasCaasC 2 aasCdssC 1 aasCsF 1 aasCssNH
1 aasCsssCH 1 aasNssCH2 1 dO=dssC 2 dssCsOH 1 dssCssCH2 1 dssCssNH 1
sCH3sssCH 2 sOHsssCH 2 ssCH2ssCH2 1 ssCH2sssCH 4 aaCH:aaCH:aaCH 6 a...

FingerprintsVector;PathLengthCount:FunctionalClassAtomTypes:MinLength1
:MaxLength8;404;NumericalValues;IDsAndValuesPairsString;Ar 22 Ar.HBA 1
HBA 2 HBA.HBD 3 HBD 1 Hal 1 NI 1 None 10 Ar.HBA:Ar 2 Ar.HBANone 1 Ar:
Ar 21 ArAr 2 ArHBD 1 ArHal 1 ArNone 2 HBA.HBDNI 1 HBA.HBDNone 2 HBA=NI
```

```
1 HBA=None 1 HBDNone 1 NINone 1 NoneNone 7 Ar.HBA:Ar:Ar 2 Ar.HBA:ArAr
1 Ar.HBA:ArNone 1 Ar.HBANoneNone 1 Ar:Ar.HBA:Ar 1 Ar:Ar.HBANone 2 ...

FingerprintsVector;PathLengthCount:MMFF94AtomTypes:MinLength1:MaxLengt
h8;463;NumericalValues;IDsAndValuesPairsString;C5A 2 C5B 2 C=ON 1 CB 1
8 COO 1 CR 9 F 1 N5 1 NC=O 1 O=CN 1 O=CO 1 OC=O 1 OR 2 C5A:C5B 2 C5A:N
5 2 C5ACB 1 C5ACR 1 C5B:C5B 1 C5BC=ON 1 C5BCB 1 C=ON=O=CN 1 C=ONNC=O 1
CB:CB 18 CBF 1 CBNC=O 1 COO=O=CO 1 COOCR 1 COOOC=O 1 CRCR 7 CRN5 1 CR
OR 2 C5A:C5B:C5B 2 C5A:C5BC=ON 1 C5A:C5BCB 1 C5A:N5:C5A 1 C5A:N5CR ...

FingerprintsVector;PathLengthCount:SLogPAtomTypes:MinLength1:MaxLength
8;518;NumericalValues;IDsAndValuesPairsString;C1 5 C10 1 C11 1 C14 1 C
18 14 C20 4 C21 2 C22 1 C5 2 CS 2 F 1 N11 1 N4 1 O10 1 O2 3 O9 1 C10C1
1 C10N11 1 C11C1 2 C11C21 1 C14:C18 2 C14F 1 C18:C18 10 C18:C20 4 C18
:C22 2 C1C5 1 C1CS 4 C20:C20 1 C20:C21 1 C20:N11 1 C20C20 2 C21:C21 1
C21:N11 1 C21C5 1 C22N4 1 C5=O10 1 C5=O9 1 C5N4 1 C5O2 1 CSO2 2 C10...

FingerprintsVector;PathLengthCount:SYBYLAtomTypes:MinLength1:MaxLength
8;412;NumericalValues;IDsAndValuesPairsString;C.2 2 C.3 9 C.ar 22 F 1
N.am 1 N.ar 1 O.2 1 O.3 2 O.co2 2 C.2=O.2 1 C.2=O.co2 1 C.2C.3 1 C.2C.
ar 1 C.2N.am 1 C.2O.co2 1 C.3C.3 7 C.3C.ar 1 C.3N.ar 1 C.3O.3 2 C.ar:C
.ar 21 C.ar:N.ar 2 C.arC.ar 2 C.arF 1 C.arN.am 1 C.2C.3C.3 1 C.2C.ar:C
.ar 2 C.2N.amC.ar 1 C.3C.2=O.co2 1 C.3C.2O.co2 1 C.3C.3C.3 5 C.3C.3...

FingerprintsVector;PathLengthCount:TPSAAtomTypes:MinLength1:MaxLength8
;331;NumericalValues;IDsAndValuesPairsString;N21 1 N7 1 None 34 O3 2 O
4 3 N21:None 2 N21None 1 N7None 2 None:None 21 None=O3 2 NoneNone 13 N
oneO4 3 N21:None:None 2 N21:NoneNone 2 N21NoneNone 1 N7None:None 2 N7N
one=O3 1 N7NoneNone 1 None:N21:None 1 None:N21None 2 None:None:None 20
None:NoneNone 12 NoneN7None 1 NoneNone=O3 2 NoneNoneNone 8 NoneNon...

FingerprintsVector;PathLengthCount:UFFAtomTypes:MinLength1:MaxLength8;
410;NumericalValues;IDsAndValuesPairsString;C_2 2 C_3 9 C_R 22 F_ 1 N_
3 1 N_R 1 O_2 2 O_3 3 C_2=O_2 2 C_2C_3 1 C_2C_R 1 C_2N_3 1 C_2O_3 1 C_
3C_3 7 C_3C_R 1 C_3N_R 1 C_3O_3 2 C_R:C_R 21 C_R:N_R 2 C_RC_R 2 C_RF_
1 C_RN_3 1 C_2C_3C_3 1 C_2C_R:C_R 2 C_2N_3C_R 1 C_3C_2=O_2 1 C_3C_2O_3
1 C_3C_3C_3 5 C_3C_3C_R 2 C_3C_3N_R 1 C_3C_3O_3 4 C_3C_R:C_R 1 C_3...
```

METHODS

new

```
$NewPathLengthFingerprints = new PathLengthFingerprints(
                                                %NamesAndValues);
```

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

```
Molecule = '';
Type = ''
Size = 1024
MinSize = 32
MaxSize = 2**32
NumOfBitsToSetPerPath = 1
MinLength = 1
MaxLength = 8
AllowSharedBonds = 1
AllowRings = 1
UseBondSymbols = 1
UseUniquePaths = ''
AtomIdentifierType = ''
SetAtomicInvariantsToUse = ['AS']
FunctionalClassesToUse = ['HBD', 'HBA', 'PI', 'NI', 'Ar', 'Hal']
```

Examples:

```
$PathLengthFingerprints = new PathLengthFingerprints(
                              'Molecule' => $Molecule,
```

```
                                    'Type' => 'PathLengthBits',
                                    'AtomIdentifierType' =
                                                'AtomicInvariantsAtomTypes');

        $PathLengthFingerprints = new PathLengthFingerprints(
                                    'Molecule' => $Molecule,
                                    'Type' => 'PathLengthBits',
                                    'Size' => 1024,
                                    'MinLength' => 1,
                                    'MaxLength' => 8,
                                    'AllowRings' => 1,
                                    'AllowSharedBonds' => 1,
                                    'UseBondSymbols' => 1,
                                    'UseUniquePaths' => 1,
                                    'AtomIdentifierType' =
                                                'AtomicInvariantsAtomTypes',
                                    'AtomicInvariantsToUse' => ['AS']);

        $PathLengthFingerprints = new PathLengthFingerprints(
                                    'Molecule' => $Molecule,
                                    'Type' => 'PathLengthCount',
                                    'MinLength' => 1,
                                    'MaxLength' => 8,
                                    'AllowRings' => 1,
                                    'AllowSharedBonds' => 1,
                                    'UseBondSymbols' => 1,
                                    'UseUniquePaths' => 1,
                                    'AtomIdentifierType' =>
                                                'AtomicInvariantsAtomTypes',
                                    'AtomicInvariantsToUse' => ['AS']);

        $PathLengthFingerprints = new PathLengthFingerprints(
                                   'Molecule' => $Molecule,
                                    'Type' => 'PathLengthBits',
                                    'AtomIdentifierType' =
                                                'SLogPAtomTypes');

        $PathLengthFingerprints = new PathLengthFingerprints(
                                   'Molecule' => $Molecule,
                                    'Type' => 'PathLengthCount',
                                    'AtomIdentifierType' =
                                                'SYBYLAtomTypes');

        $PathLengthFingerprints = new PathLengthFingerprints(
                                    'Molecule' => $Molecule,
                                    'Type' => 'PathLengthBits',
                                    'AtomIdentifierType' =
                                                'FunctionalClassAtomTypes',
                                    'FunctionalClassesToUse' => ['HBD', 'HBA', 'Ar']);

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

GetDescription

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

Returns a string containing description of path length fingerprints.

GenerateFingerprints

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

Generates path length fingerprints and returns *PathLengthFingerprints*.

SetMaxLength

```
        $PathLengthFingerprints->SetMaxLength($Length);
```

Sets maximum value of atom path length to be used during atom path length fingerprints generation

and returns *PathLengthFingerprints*

SetAtomIdentifierType

```
$PathLengthFingerprints->SetAtomIdentifierType();
```

Sets atom *IdentifierType* to use during path length fingerprints generation and returns *PathLengthFingerprints*.

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

SetAtomicInvariantsToUse

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

Sets atomic invariants to use during *AtomicInvariantsAtomTypes* value of *AtomIdentifierType* for path length fingerprints generation and returns *PathLengthFingerprints*.

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

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

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

Sets functional classes invariants to use during *FunctionalClassAtomTypes* value of *AtomIdentifierType* for path length fingerprints generation and returns *PathLengthFingerprints*.

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
```

SetMinLength

```
$PathLengthFingerprints->SetMinLength($Length);
```

Sets minimum value of atom path length to be used during atom path length fingerprints generation and returns *PathLengthFingerprints*.

SetMaxLength

```
$PathLengthFingerprints->SetMaxLength($Length);
```

Sets maximum value of atom path length to be used during atom path length fingerprints generation and returns *PathLengthFingerprints*.

SetNumOfBitsToSetPerPath

```
$PathLengthFingerprints->SetNumOfBitsToSetPerPath($NumOfBits);
```

Sets number of bits to set for each path during *PathLengthBits* Type during path length fingerprints generation and returns *PathLengthFingerprints*.

SetType

```
$PathLengthFingerprints->SetType($Type);
```

Sets type of path length fingerprints and returns *PathLengthFingerprints*. Possible values: *PathLengthBits or PathLengthCount*.

StringifyPathLengthFingerprints

```
$String = $PathLengthFingerprints->StringifyPathLengthFingerprints();
```

Returns a string containing information about *PathLengthFingerprints* object.

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

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

COPYRIGHT