NAME

AtomTypesFingerprints.pl - Generate atom types fingerprints for SD files

SYNOPSIS

AtomTypesFingerprints.pl SDFile(s)...

AtomTypesFingerprints.pl [--AromaticityModel *AromaticityModelType*] [-a, --AtomIdentifierType *AtomicInvariantsAtomTypes* | *DREIDINGAtomTypes* | *EStateAtomTypes* | *MMFF94AtomTypes* | *SLogPAtomTypes* | *SYBYLAtomTypes* | *TPSAAtomTypes* | *UFFAtomTypes*] [--AtomicInvariantsToUse *"AtomicInvariant, AtomicInvariant..."*] [--FunctionalClassesToUse *"FunctionalClass1,FunctionalClass2..."*] [--AtomTypesSetToUse *ArbitrarySize* | *FixedSize*] [--BitsOrder *Ascending* | *Descending*] [-b, --BitStringFormat *BinaryString* | *HexadecimalString*] [--CompoundID *DataFieldName or LabelPrefixString*] [--CompoundIDLabel *text*] [--CompoundIDMode *DataField* | *MolName* | *LabelPrefix* | *MolNameOrLabelPrefix*] [--DataFields *"FieldLabel1,FieldLabel2,..."*] [-d, --DataFieldsMode *All* | *Common* | *Specify* | *CompoundID*] [-f, --Filter *Yes* | *No*] [--FingerprintsLabelMode *FingerprintsLabelOnly* | *FingerprintsLabelWithIDs*] [--FingerprintsLabel *text*] [-h, --help] [-k, --KeepLargestComponent *Yes* | *No*] [-m, --mode *AtomTypesCount* | *AtomTypesBits*] [-i, --IgnoreHydrogens *Yes* | *No*] [--OutDelim *comma* | *tab* | *semicolon*] [--output *SD* |*FP* | *text* | *all*] [-o, --overwrite] [-q, --quote *Yes* | *No*] [-r, --root *RootName*] [-s, --size *number*] [--ValuesPrecision *number*] [-v, --VectorStringFormat *IDsAndValuesString* | *IDsAndValuesPairsString* | *ValuesAndIDsString* | *ValuesAndIDsPairsString*] [-w, --WorkingDir *DirName*]

DESCRIPTION

Generate atom types fingerprints for *SDFile(s)* and create appropriate SD, FP or CSV/TSV text file(s) containing fingerprints bit-vector or vector strings corresponding to molecular fingerprints.

Multiple SDFile names are separated by spaces. The valid file extensions are *.sdf* and *.sd*. All other file names are ignored. All the SD files in a current directory can be specified either by **.sdf* or the current directory name.

The current release of MayaChemTools supports generation of atom types fingerpritns corresponding to following -a, --AtomIdentifierTypes:

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

Based on the values specified for -a, --AtomIdentifierType along with other specified parameters such as --AtomicInvariantsToUse and --FunctionalClassesToUse, initial atom types are assigned to all non-hydrogen atoms or all atoms in a molecule

Using the assigned atom types and specified -m, --Mode, one of the following types of fingerprints are generated:

```
AtomTypesCount - A vector containing count of atom types
AtomTypesBits - A bit vector indicating presence/absence of atom types
```

For *AtomTypesCount* fingerprints, two types of atom types set size are allowed as value of --AtomTypesSetToUse option:

```
ArbitrarySize - Corresponds to only atom types detected in molecule
FixedSize - Corresponds to fixed number of atom types previously defined
```

For *AtomTypesBits* fingerprints, only *FixedSize* atom type set is allowed.

ArbitrarySize corresponds to atom types detected in a molecule where as *FixedSize* implies a fix number of all possible atom types previously defined for a specific -a, --AtomIdentifierType.

Fix number of all possible atom types for supported *AtomIdentifierTypes* in current release of MayaChemTools are:

AtomIdentifier	Total	TotalWithoutHydrogens
DREIDINGAtomTypes	37	34
EStateAtomTypes	109	87
MMFF94AtomTypes	212	171
SLogPAtomTypes	72	67
SYBYLAtomTypes	45	44
TPSAAtomTypes	47	47
UFFAtomTypes	126	124

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

```
FingerprintsVector;AtomTypesCount:AtomicInvariantsAtomTypes:ArbitraryS
ize;10;NumericalValues;IDsAndValuesString;C.X1.BO1.H3 C.X2.BO2.H2 C.X2
.BO3.H1 C.X3.BO3.H1 C.X3.BO4 F.X1.BO1 N.X2.BO2.H1 N.X3.BO3 O.X1.BO1.H1
O.X1.BO2;2 4 14 3 10 1 1 1 3 2


FingerprintsVector;AtomTypesCount:DREIDINGAtomTypes:ArbitrarySize;8;Nu
```

```
mericalValues;IDsAndValuesString;C_2 C_3 C_R F_ N_3 N_R O_2 O_3;2 9 22
1 1 1 2 3

FingerprintsVector;AtomTypesCount:DREIDINGAtomTypes:FixedSize;34;Order
edNumericalValues;IDsAndValuesString;B_3 B_2 C_3 C_R C_2 C_1 N_3 N_R N
_2 N_1 O_3 O_R O_2 O_1 F_ Al3 Si3 P_3 S_3 Cl Ga3 Ge3 As3 Se3 Br In3 Sn
3 Sb3 Te3 I_ Na Ca Fe Zn;0 0 9 22 2 0 1 1 0 0 3 0 2 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0

FingerprintsBitVector;AtomTypesBits:DREIDINGAtomTypes:FixedSize;34;Bin
aryString;Ascending;00111011001010100000000000000000000000000

FingerprintsVector;AtomTypesCount:EStateAtomTypes:ArbitrarySize;11;Num
ericalValues;IDsAndValuesString;aaCH aasC aasN dO dssC sCH3 sF sOH ssC
H2 ssNH sssCH;14 8 1 2 2 2 1 3 4 1 3

FingerprintsVector;AtomTypesCount:EStateAtomTypes:FixedSize;87;Ordered
NumericalValues;IDsAndValuesString;sLi ssBe ssssBem sBH2 ssBH sssB sss
sBm sCH3 dCH2 ssCH2 tCH dsCH aaCH sssCH ddC tsC dssC aasC aaaC ssssC s
NH3p sNH2 ssNH2p dNH ssNH aaNH tN sssNHp dsN aaN sssN ddsN aasN ss...;
0 0 0 0 0 0 0 2 0 4 0 0 14 3 0 0 2 8 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 3 2 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0...

FingerprintsBitVector;AtomTypesBits:EStateAtomTypes:FixedSize;87;Binar
yString;Ascending;000000010100110011000000100000001011000010000000000
000000000000000000000000000000000000

FingerprintsVector;AtomTypesCount:FunctionalClassAtomTypes:ArbitrarySi
ze;8;NumericalValues;IDsAndValuesString;Ar Ar.HBA HBA HBA.HBD HBD Hal
NI None;22 1 2 3 1 1 1 10

FingerprintsVector;AtomTypesCount:MMFF94AtomTypes:ArbitrarySize;13;Num
ericalValues;IDsAndValuesString;C5A C5B C=ON CB COO CR F N5 NC=O O=CN
O=CO OC=O OR;2 2 1 18 1 9 1 1 1 1 1 1 2

FingerprintsVector;AtomTypesCount:MMFF94AtomTypes:FixedSize;171;Ordere
dNumericalValues;IDsAndValuesString;CR C=C CSP2 C=O C=N CGD C=OR C=ON
CONN COO COON COOO C=OS C=S C=SN CSO2 CS=O CSS C=P CSP =C= OR OC=O OC=
C OC=N OC=S ONO2 ON=O OSO3 OSO2 OSO OS=O -OS OPO3 OPO2 OPO -OP -O-...;
9 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 2 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 18 0 0 0 0 0 0 0 0 ...

FingerprintsBitVector;AtomTypesBits:MMFF94AtomTypes:FixedSize;171;Bina
ryString;Ascending;10000001010000000000011000000000000000001010000001000
01000000000000000000000000000000000000000000010000000000000000000000000
00000000110000000000000000010000000000000000000000000000

FingerprintsVector;AtomTypesCount:SLogPAtomTypes:ArbitrarySize;16;Nume
ricalValues;IDsAndValuesString;C1 C10 C11 C14 C18 C20 C21 C22 C5 CS F
N11 N4 O10 O2 O9;5 1 1 1 14 4 2 1 2 2 1 1 1 1 3 1

FingerprintsVector;AtomTypesCount:SLogPAtomTypes:FixedSize;67;OrderedN
umericalValues;IDsAndValuesString;C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C
12 C13 C14 C15 C16 C17 C18 C19 C20 C21 C22 C23 C24 C25 C26 C27 CS N1 N
2 N3 N4 N5 N6 N7 N8 N9 N10 N11 N12 N13 N14 NS O1 O2 O3 O4 O5 O6 O7 O8
O9 O10 O11 O12 OS F Cl Br I Hal P S1 S2 S3 Me1 Me2;5 0 0 0 2 0 0 0 0 1
1 0 0 1 0 0 0 14 0 4 2 1 0 0 0 0 0 2 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0...

FingerprintsBitVector;AtomTypesBits:SLogPAtomTypes:FixedSize;67;Binary
String;Ascending;10001000011001000101110000010001000000100000100000011
0001000000000000000

FingerprintsVector;AtomTypesCount:SYBYLAtomTypes:ArbitrarySize;9;Numer
icalValues;IDsAndValuesString;C.2 C.3 C.ar F N.am N.ar O.2 O.3 O.co2;2
9 22 1 1 1 1 2 2
```

```
FingerprintsVector;AtomTypesCount:SYBYLAtomTypes:FixedSize;44;OrderedN
umericalValues;IDsAndValuesString;C.3 C.2 C.1 C.ar C.cat N.3 N.2 N.1 N
.ar N.am N.pl3 N.4 O.3 O.2 O.co2 S.3 S.2 S.o S.o2 P.3 F Cl Br I ANY HA
L HET Li Na Mg Al Si K Ca Cr.th Cr.oh Mn Fe Co.oh Cu Zn Se Mo Sn;9 2 0
22 0 0 0 0 1 1 0 0 2 1 2 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0


FingerprintsBitVector;AtomTypesBits:SYBYLAtomTypes:FixedSize;44;Binary
String;Ascending;11010000110011100000100000000000000000000000000000


FingerprintsVector;AtomTypesCount:TPSAAtomTypes:FixedSize;47;OrderedNu
mericalValues;IDsAndValuesString;N1 N2 N3 N4 N5 N6 N7 N8 N9 N10 N11 N1
2 N13 N14 N15 N16 N17 N18 N19 N20 N21 N22 N23 N24 N25 N26 N O1 O2 O3 O
4 O5 O6 O S1 S2 S3 S4 S5 S6 S7 S P1 P2 P3 P4 P;0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 2 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0


FingerprintsBitVector;AtomTypesBits:TPSAAtomTypes:FixedSize;47;BinaryS
tring;Ascending;00000010000000000000100000000110000000000000000000


FingerprintsVector;AtomTypesCount:UFFAtomTypes:ArbitrarySize;8;Numeric
alValues;IDsAndValuesString;C_2 C_3 C_R F_ N_3 N_R O_2 O_3;2 9 22 1 1
1 2 3


FingerprintsVector;AtomTypesCount:UFFAtomTypes;124;OrderedNumerical
Values;IDsAndValuesString;He4+4 Li Be3+2 B_3 B_2 C_3 C_R C_2 C_1 N_3 N_
R N_2 N_1 O_3 O_3_z O_R O_2 O_1 F_ Ne4+4 Na Mg3+2 Al3 Si3 P_3+3 P_3+5 P
_3+q S_3+2 S_3+4 S_3+6 S_R S_2 Cl Ar4+4 K_ Ca6+2 Sc3+3 Ti3+4 Ti6+4 V_3+
;0 0 0 0 0 12 0 3 0 3 0 1 0 2 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ...


FingerprintsVector;AtomTypesCount:UFFAtomTypes:FixedSize;124;OrderedNu
mericalValues;IDsAndValuesString;He4+4 Li Be3+2 B_3 B_2 C_3 C_R C_2 C_
1 N_3 N_R N_2 N_1 O_3 O_3_z O_R O_2 O_1 F_ Ne4+4 Na Mg3+2 Al3 Si3 P_3+
3 P_3+5 P_3+q S_3+2 S_3+4 S_3+6 S_R S_2 Cl Ar4+4 K_ Ca6+2 Sc3+3 Ti...;
0 0 0 0 0 9 22 2 0 1 1 0 0 3 0 0 2 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0...


FingerprintsBitVector;AtomTypesBits:UFFAtomTypes:FixedSize;124;BinaryS
tring;Ascending;00000111011001001010000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
```

OPTIONS

--AromaticityModel *MDLAromaticityModel* | *TriposAromaticityModel* | *MMFFAromaticityModel* | *ChemAxonBasicAromaticityModel* | *ChemAxonGeneralAromaticityModel* | *DaylightAromaticityModel* | *MayaChemToolsAromaticityModel*

Specify aromaticity model to use during detection of aromaticity. Possible values in the current release are: *MDLAromaticityModel, TriposAromaticityModel, MMFFAromaticityModel, ChemAxonBasicAromaticityModel, ChemAxonGeneralAromaticityModel, DaylightAromaticityModel or MayaChemToolsAromaticityModel*. Default value: *MayaChemToolsAromaticityModel*.

The supported aromaticity model names along with model specific control parameters are defined in AromaticityModelsData.csv, which is distributed with the current release and is available under lib/data directory. Molecule.pm module retrieves data from this file during class instantiation and makes it available to method DetectAromaticity for detecting aromaticity corresponding to a specific model.

-a, --AtomIdentifierType *AtomicInvariantsAtomTypes* | *DREIDINGAtomTypes* | *EStateAtomTypes* | *FunctionalClassAtomTypes* | *MMFF94AtomTypes* | *SLogPAtomTypes* | *SYBYLAtomTypes* | *TPSAAtomTypes* | *UFFAtomTypes*

Specify atom identifier type to use for assignment of atom types to hydrogen and/or non-hydrogen atoms during calculation of atom types fingerprints. Possible values in the current release are: *AtomicInvariantsAtomTypes, DREIDINGAtomTypes, EStateAtomTypes, FunctionalClassAtomTypes, MMFF94AtomTypes, SLogPAtomTypes, SYBYLAtomTypes, TPSAAtomTypes, UFFAtomTypes*. Default value: *AtomicInvariantsAtomTypes*.

--AtomicInvariantsToUse *"AtomicInvariant,AtomicInvariant..."*

This value is used during *AtomicInvariantsAtomTypes* value of a, --AtomIdentifierType option. It's a list of comma separated valid atomic invariant atom types.

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.

--FunctionalClassesToUse *"FunctionalClass1,FunctionalClass2..."*

This value is used during *FunctionalClassAtomTypes* value of a, --AtomIdentifierType option. It's a list of comma separated valid functional classes.

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
```

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
```

--AtomTypesSetToUse *ArbitrarySize* | *FixedSize*

Atom types set size to use during generation of atom types fingerprints.

Possible values for *AtomTypesCount* values of -m, --mode option: *ArbitrarySize* | *FixedSize*; Default value: *ArbitrarySize*.

Possible values for *AtomTypesBits* value of -m, --mode option: *FixedSize*; Default value: *FixedSize*.

FixedSize value is not supported for *AtomicInvariantsAtomTypes* value of -a, --AtomIdentifierType option.

ArbitrarySize corresponds to only atom types detected in molecule; *FixedSize* corresponds to fixed number of previously defined atom types for specified -a, --AtomIdentifierType.

--BitsOrder *Ascending* | *Descending*

Bits order to use during generation of fingerprints bit-vector string for *AtomTypesBits* value of =item --BitsOrder *Ascending* | *Descending*

Bits order to use during generation of fingerprints bit-vector string for *AtomTypesBits* value of -m, --mode option. Possible values: *Ascending, Descending*. Default: *Ascending*.

Ascending bit order which corresponds to first bit in each byte as the lowest bit as opposed to the highest bit.

Internally, bits are stored in *Ascending* order using Perl vec function. Regardless of machine order, big-endian or little-endian, vec function always considers first string byte as the lowest byte and first bit within each byte as the lowest bit.

-b, --BitStringFormat *BinaryString* | *HexadecimalString*

Format of fingerprints bit-vector string data in output SD, FP or CSV/TSV text file(s) specified by --output used during *AtomTypesBits* value of -m, --mode option. Possible values: *BinaryString, HexadecimalString*. Default value: *BinaryString*.

BinaryString corresponds to an ASCII string containing 1s and 0s. *HexadecimalString* contains bit values in ASCII hexadecimal format.

Examples:

```
FingerprintsBitVector;AtomTypesBits:DREIDINGAtomTypes;34;BinaryString;
Ascending;00101010101010000000000000000000000000
```

```
FingerprintsBitVector;AtomTypesBits:MMFF94AtomTypes;171;BinaryString;
Ascending;10000101010000000000011000000000000000010100001010000000000
00000000000000000000000000000000000001000000000000000000000000000000
00000000000000000000000000000000000000
```

--CompoundID *DataFieldName or LabelPrefixString*

This value is --CompoundIDMode specific and indicates how compound ID is generated.

For *DataField* value of --CompoundIDMode option, it corresponds to datafield label name whose value is used as compound ID; otherwise, it's a prefix string used for generating compound IDs like LabelPrefixString<Number>. Default value, *Cmpd*, generates compound IDs which look like Cmpd<Number>.

Examples for *DataField* value of --CompoundIDMode:

```
MolID
ExtReg
```

Examples for *LabelPrefix* or *MolNameOrLabelPrefix* value of --CompoundIDMode:

```
Compound
```

The value specified above generates compound IDs which correspond to Compound<Number> instead of default value of Cmpd<Number>.

--CompoundIDLabel *text*

Specify compound ID column label for FP or CSV/TSV text file(s) used during *CompoundID* value of --DataFieldsMode option. Default: *CompoundID*.

--CompoundIDMode *DataField* | *MolName* | *LabelPrefix* | *MolNameOrLabelPrefix*

Specify how to generate compound IDs and write to FP or CSV/TSV text file(s) along with generated fingerprints for *FP* | *text* | *all* values of --output option: use a *SDFile(s)* datafield value; use molname line from *SDFile(s)*; generate a sequential ID with specific prefix; use combination of both MolName and LabelPrefix with usage of LabelPrefix values for empty molname lines.

Possible values: *DataField* | *MolName* | *LabelPrefix* | *MolNameOrLabelPrefix*. Default: *LabelPrefix*.

For *MolNameAndLabelPrefix* value of --CompoundIDMode, molname line in *SDFile(s)* takes precedence over sequential compound IDs generated using *LabelPrefix* and only empty molname values are replaced with sequential compound IDs.

This is only used for *CompoundID* value of --DataFieldsMode option.

--DataFields *"FieldLabel1,FieldLabel2,..."*

Comma delimited list of *SDFiles(s)* data fields to extract and write to CSV/TSV text file(s) along with generated fingerprints for *text* | *all* values of --output option.

This is only used for *Specify* value of --DataFieldsMode option.

Examples:

```
Extreg
MolID,CompoundName
```

-d, --DataFieldsMode *All* | *Common* | *Specify* | *CompoundID*

Specify how data fields in *SDFile(s)* are transferred to output CSV/TSV text file(s) along with generated fingerprints for *text* | *all* values of --output option: transfer all SD data field; transfer SD data files common to all compounds; extract specified data fields; generate a compound ID using molname line, a compound prefix, or a combination of both. Possible values: *All* | *Common* | *specify* | *CompoundID*. Default value: *CompoundID*.

-f, --Filter *Yes* | *No*

Specify whether to check and filter compound data in SDFile(s). Possible values: *Yes or No*. Default value: *Yes*.

By default, compound data is checked before calculating fingerprints and compounds containing atom data corresponding to non-element symbols or no atom data are ignored.

--FingerprintsLabelMode *FingerprintsLabelOnly* | *FingerprintsLabelWithIDs*

Specify how fingerprints label is generated in conjunction with --FingerprintsLabel option value: use fingerprints label generated only by --FingerprintsLabel option value or append atom type value IDs to --FingerprintsLabel option value.

Possible values: *FingerprintsLabelOnly* | *FingerprintsLabelWithIDs*. Default value: *FingerprintsLabelOnly*.

This option is only used for *FixedSize* value of -e, --AtomTypesSetToUse option during generation of *AtomTypesCount* fingerprints and ignored for *AtomTypesBits*.

Atom type IDs appended to --FingerprintsLabel value during *FingerprintsLabelWithIDs* values of --FingerprintsLabelMode correspond to fixed number of previously defined atom types.

--FingerprintsLabel *text*

SD data label or text file column label to use for fingerprints string in output SD or CSV/TSV text file(s) specified by --output. Default value: *AtomTypesFingerprints*.

-h, --help

Print this help message.

-i, --IgnoreHydrogens *Yes* | *No*

Ignore hydrogens during fingerprints generation. Possible values: *Yes or No*. Default value: *Yes*.

For *yes* value of -i, --IgnoreHydrogens, any explicit hydrogens are also used for generation of atom type fingerprints; implicit hydrogens are still ignored.

-k, --KeepLargestComponent *Yes* | *No*

Generate fingerprints for only the largest component in molecule. Possible values: *Yes or No*. Default value: *Yes*.

For molecules containing multiple connected components, fingerprints can be generated in two different ways: use all connected components or just the largest connected component. By default, all atoms except for the largest connected component are deleted before generation of fingerprints.

-m, --mode *AtomTypesCount* | *AtomTypesBits*

Specify type of atom types fingerprints to generate for molecules in *SDFile(s)*. Possible values: *AtomTypesCount or AtomTypesBits*. Default value: *AtomTypesCount*.

For *AtomTypesCount* values of -m, --mode option, a fingerprint vector string is generated. The vector string corresponding to *AtomTypesCount* contains count of atom types.

For *AtomTypesBits* value of -m, --mode option, a fingerprint bit-vector string containing zeros and ones indicating presence or absence of atom types is generated.

For *AtomTypesCount* atom types fingerprints, two types of atom types set size can be specified using -a, --AtomTypesSetToUse option: *ArbitrarySize or FixedSize*. *ArbitrarySize* corrresponds to only atom types detected in molecule; *FixedSize* corresponds to fixed number of atom types previously defined.

For *AtomTypesBits* atom types fingeprints, only *FixedSize* is allowed.

Combination of -m, --Mode and --AtomTypesSetToUse along with -a, --AtomtomIdentifierType allows generation of following different atom types fingerprints:

```
Mode                AtomIdentifierType          AtomTypesSetToUse

AtomTypesCount      AtomicInvariantsAtomTypes   ArbitrarySize [ Default ]

AtomTypesCount      DREIDINGAtomTypes           ArbitrarySize
AtomTypesCount      DREIDINGAtomTypes           FixedSize
AtomTypesBits       DREIDINGAtomTypes           FixedSize

AtomTypesCount      EStateAtomTypes             ArbitrarySize
AtomTypesCount      EStateAtomTypes             FixedSize
AtomTypesBits       EStateAtomTypes             FixedSize

AtomTypesCount      FunctionalClassAtomTypes    ArbitrarySize

AtomTypesCount      MMFF94AtomTypes             ArbitrarySize
```

```
AtomTypesCount        MMFF94AtomTypes             FixedSize
AtomTypesBits         MMFF94AtomTypes             FixedSize

AtomTypesCount        SLogPAtomTypes              ArbitrarySize
AtomTypesCount        SLogPAtomTypes              FixedSize
AtomTypesBits         SLogPAtomTypes              FixedSize

AtomTypesCount        SYBYLAtomTypes              ArbitrarySize
AtomTypesCount        SYBYLAtomTypes              FixedSize
AtomTypesBits         SYBYLAtomTypes              FixedSize

AtomTypesCount        TPSAAtomTypes                FixedSize
AtomTypesBits         TPSAAtomTypes                FixedSize

AtomTypesCount        UFFAtomTypes                ArbitrarySize
AtomTypesCount        UFFAtomTypes                FixedSize
AtomTypesBits         UFFAtomTypes                FixedSize
```

The default is to generate *AtomicInvariantAtomTypes* fingeprints corresponding to *ArbitrarySize* as value of --AtomTypesSetToUse option.

--OutDelim *comma* | *tab* | *semicolon*

Delimiter for output CSV/TSV text file(s). Possible values: *comma, tab, or semicolon* Default value: *comma*.

--output *SD* | *FP* | *text* | *all*

Type of output files to generate. Possible values: *SD, FP, text, or all*. Default value: *text*.

-o, --overwrite

Overwrite existing files.

-q, --quote *Yes* | *No*

Put quote around column values in output CSV/TSV text file(s). Possible values: *Yes or No*. Default value: *Yes*.

-r, --root *RootName*

New file name is generated using the root: <Root>.<Ext>. Default for new file names: <SDFileName><AtomTypesFP>.<Ext>. The file type determines <Ext> value. The sdf, fpf, csv, and tsv <Ext> values are used for SD, FP, comma/semicolon, and tab delimited text files, respectively.This option is ignored for multiple input files.

-v, --VectorStringFormat *ValuesString* | *IDsAndValuesString* | *IDsAndValuesPairsString* | *ValuesAndIDsString* | *ValuesAndIDsPairsString*

Format of fingerprints vector string data in output SD, FP or CSV/TSV text file(s) specified by --output used during <AtomTypesCount> value of -m, --mode option. Possible values: *ValuesString, IDsAndValuesString* | *IDsAndValuesPairsString* | *ValuesAndIDsString* | *ValuesAndIDsPairsString*.

Default value during *ArbitrarySize* value of -e, --AtomTypesSetToUse option: *IDsAndValuesString*. Default value during *FixedSize* value of -e, --AtomTypesSetToUse option: *ValuesString*.

Example of *SD* file containing atom types fingerprints string data:

```
... ...
... ...
$$$$
... ...
... ...
... ...
41 44  0  0  0  0  0  0  0  0999 V2000
 -3.3652    1.4499    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
... ...
2  3  1  0  0  0  0
... ...
M  END
>  <CmpdID>
Cmpd1

>  <AtomTypesFingerprints>
FingerprintsVector;AtomTypesCount:AtomicInvariantsAtomTypes:ArbitrarySi
ze;10;NumericalValues;IDsAndValuesString;C.X1.BO1.H3 C.X2.BO2.H2 C.X2.B
O3.H1 C.X3.BO3.H1 C.X3.BO4 F.X1.BO1 N.X2.BO2.H1 N.X3.BO3 O.X1.BO1.H1 O.
X1.BO2;2 4 14 3 10 1 1 1 3 2

$$$$
... ...
... ...
```

Example of *FP* file containing atom types fingerprints string data:

```
#
# Package = MayaChemTools 7.4
# Release Date = Oct 21, 2010
#
# TimeStamp = Fri Mar 11 14:28:07 2011
#
# FingerprintsStringType = FingerprintsVector
#
# Description = AtomTypesCount:AtomicInvariantsAtomTypes:ArbitrarySize
# VectorStringFormat = IDsAndValuesString
# VectorValuesType = NumericalValues
#
Cmpd1 10;C.X1.BO1.H3 C.X2.BO2.H2 C.X2.BO3.H1 C.X3.BO3.H1 C.X3.BO4 F...
Cmpd2 9;C.X1.BO1.H3 C.X2.BO2.H2 C.X3.BO3.H1 C.X3.BO4 N.X1.BO1.H2 N....
... ...
... ..
```

Example of CSV *Text* file atom types containing fingerprints string data:

```
"CompoundID","AtomTypesFingerprints"
"Cmpd1","FingerprintsVector;AtomTypesCount:AtomicInvariantsAtomTypes:Ar
bitrarySize;10;NumericalValues;IDsAndValuesString;C.X1.BO1.H3 C.X2.BO2.
H2 C.X2.BO3.H1 C.X3.BO3.H1 C.X3.BO4 F.X1.BO1 N.X2.BO2.H1 N.X3.BO3 O.X1.
BO1.H1 O.X1.BO2;2 4 14 3 10 1 1 1 3 2"
O.X1.BO2;3 3 6 3 1 1 2 2 2"
... ...
... ...
```

Examples:

```
FingerprintsVector;AtomTypesCount:EStateAtomTypes:ArbitrarySize;11;Num
ericalValues;IDsAndValuesString;aaCH aasC aasN dO dssC sCH3 sF sOH ssC
H2 ssNH sssCH;14 8 1 2 2 2 1 3 4 1 3

FingerprintsVector;AtomTypesCount:SYBYLAtomTypes:ArbitrarySize;9;Numer
icalValues;IDsAndValuesString;C.2 C.3 C.ar F N.am N.ar O.2 O.3 O.co2;2
9 22 1 1 1 1 2 2

FingerprintsVector;AtomTypesCount:SYBYLAtomTypes:FixedSize;44;OrderedN
umericalValues;IDsAndValuesString;C.3 C.2 C.1 C.ar C.cat N.3 N.2 N.1 N
.ar N.am N.pl3 N.4 O.3 O.2 O.co2 S.3 S.2 S.o S.o2 P.3 F Cl Br I ANY HA
L HET Li Na Mg Al Si K Ca Cr.th Cr.oh Mn Fe Co.oh Cu Zn Se Mo Sn;9 2 0
22 0 0 0 0 1 1 0 0 2 1 2 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0
```

-w, --WorkingDir *DirName*

Location of working directory. Default: current directory.

EXAMPLES

To generate atomic invariants atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -r SampleATFP -o Sample.sdf
```

To generate functional class atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a FunctionalClassAtomTypes
  -r SampleATFP -o Sample.sdf
```

To generate E-state atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a EStateAtomTypes
  --AtomTypesSetToUse ArbitrarySize -r SampleATFP -o Sample.sdf
```

To generate E-state atom types count fingerprints of fixed size in vector string with IDsAndValues format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a EStateAtomTypes
  --AtomTypesSetToUse FixedSize -v IDsAndValuesString
```

```
  -r SampleATFP -o Sample.sdf
```

To generate E-state atom types bits fingerprints of fixed size in bit-vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesBits -a EStateAtomTypes
  --AtomTypesSetToUse FixedSize -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --AtomTypesSetToUse ArbitrarySize -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of fixed size in vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --AtomTypesSetToUse FixedSize -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of fixed size in vector string with IDsAndValues format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --AtomTypesSetToUse FixedSize -v IDsAndValuesString
  -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types bits fingerprints of fixed size in bit-vector string format and create a SampleATFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesBits -a MMFF94AtomTypes
  --AtomTypesSetToUse FixedSize -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing compound ID from molecule name line along with fingerprints vector strings data, type

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --DataFieldsMode CompoundID --CompoundIDMode MolName
  -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing compound IDs using specified data field along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --DataFieldsMode CompoundID --CompoundIDMode DataField --CompoundID
  Mol_ID -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing compound ID using combination of molecule name line and an explicit compound prefix along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --DataFieldsMode CompoundID --CompoundIDMode MolnameOrLabelPrefix
  --CompoundID Cmpd --CompoundIDLabel MolID -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing specific data fields columns along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
  --DataFieldsMode Specify --DataFields Mol_ID -r SampleATFP
  -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create a SampleATFP.csv file containing common data fields columns along with fingerprints vector strings data, type:

```
% AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
 --DataFieldsMode Common -r SampleATFP -o Sample.sdf
```

To generate MMFF94 atom types count fingerprints of arbitrary size in vector string format and create SampleATFP.sdf, SampleATFP.fpf and SampleATFP.csv files containing all data fields columns in CSV file along with fingerprints vector strings data,

```
type:    % AtomTypesFingerprints.pl -m AtomTypesCount -a MMFF94AtomTypes
     --DataFieldsMode All --output all -r SampleATFP -o Sample.sdf
```

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

InfoFingerprintsFiles.pl, SimilarityMatricesFingerprints.pl, AtomNeighborhoodsFingerprints.pl, ExtendedConnectivityFingerprints.pl, MACCSKeysFingeprints.pl, PathLengthFingerprints.pl, TopologicalAtomPairsFingerprints.pl, TopologicalAtomTorsionsFingerprints.pl, TopologicalPharmacophoreAtomPairsFingerprints.pl, TopologicalPharmacophoreAtomTripletsFingerprints.pl

COPYRIGHT