**如何用Rdkit计算MACCS密钥以及每个指纹位点代表什么**

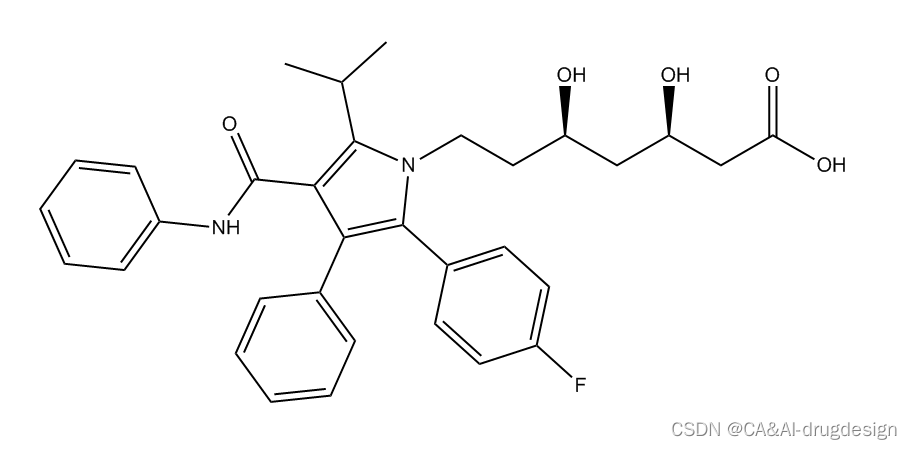
**如何用Rdkit计算MACCS密钥以及每个指纹位点代表什么**

**1.MACCS密钥是什么**

MACCS（分子访问系统）键是最常用的结构键之一，有时也被称为 MDL [密钥](https://so.csdn.net/so/search?q=密钥&spm=1001.2101.3001.7020)，MDL来源于开发它的公司的名称（MDL 信息系统，现为 BIOVIA）。 虽然有两组 MACCS 密钥（一组包含 960 个密钥，另一组包含 166 个密钥的子集），但只有包含 166 个密钥的MACCS可供公众使用。 这 166 个密钥可以通过流行的开源化学信息学软件包（RDKit 、OpenBabel、CDK等）计算。

在结构键中，分子结构被编码为[二进制](https://so.csdn.net/so/search?q=二进制&spm=1001.2101.3001.7020)位串（即 0 和 1 的序列），每个位对应于“预定义的”结构特征（例如，子结构或片段）。 如果分子具有预定义的特征，则对应于该特征的位位置设置为 1 (ON)。 否则，将其设置为 0 (OFF)。 需要注意的是，结构键不能对片段库中未预定义的结构特征进行编码。

**2. 如何计算MACCS密钥**



以该化合物结构为例计算该化合物的MACCS密钥

|  |
| --- |
| Python  from rdkit import Chem  mol = Chem.MolFromSmiles('CC(C)C1=C(C(=C(N1CC[C@H](C[C@H](CC(=O)O)O)O)C2=CC=C(C=C2)F)C3=CC=CC=C3)C(=O)NC4=CC=CC=C4')  from rdkit.Chem import MACCSkeys  fp = MACCSkeys.GenMACCSKeys(mol)  print(type(fp)) #<class 'rdkit.DataStructs.cDataStructs.ExplicitBitVect'>  for i in range(len(fp)):  print(fp[i], end='')  #将其转换为位串  #00000000000000000000000000000000000000000010000000000100000000100100000000110000100101010111100011001000100110110000011001110100110111111101101011111111111111111111110  fp.ToBitString()  #将其转换为位串的另一种更简单的方法  #打印位串形式的MACCS密钥00000000000000000000000000000000000000000010000000000100000000100100000000110000100101010111100011001000100110110000011001110100110111111101101011111111111111111111110  len(fp)  #167  #请注意，MACCS 密钥是 166 位长的，但 RDKit 会生成一个 167 位长的指纹。 这是因为许多编程语言（包括 python）中列表/向量的索引从 0 开始。为了使用 MACCS 键的原始编号（1-166）（而不是 0-165），MACCS 键被实现为 长度为 167 位，位 0 始终为零。 因为所有化合物的位 0 都设置为 OFF，所以它不会影响分子相似性的评估。  fp\_bits = tuple(fp.GetOnBits())  print(fp\_bits)  #含有1的密钥位点（42,53,62,65,74,75,80,83,85,87,89,90,91,92,96,97,104，107,108,110,111,117,118,121,122,123,125,128,129，，，，165）  print(fp.GetNumBits())  #167  print(fp.GetNumOffBits())  # 105  #位点为0的位数  print(fp.GetNumOnBits())  #62  #位点为1的位数  print(fp.ToBinary())  #b'\xe0\xff\xff\xff\xa7\x00\x00\x00>\x00\x00\x00T\x14\x10\x04\x10\x00\x08\x04\x02\x02\x02\x00\x00\x00\x06\x00\x04\x06\x04\x00\x02\x00  \x00\x04\x00\x00\x02\x04\x00\x02\x00\x00\x00\x00\x00\x00\x02\x00\x02\x02\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02' |

**3.MACCS的每个密钥位点代表什么**

MACCS 166 个密钥的片段定义如参考链接2所示，这些密钥是基于SMARTS 编辑的。要想了解SMARTS，我们先要了解一下SMILES。

SMILES是简化分子线性输入的方法（Simplified molecular input line entry specification），用字符串来描述分子结构，\*\*一个SMILES代表一个唯一的化学结构。\*\*详细介绍可以参考https://www.jianshu.com/p/8c915de5ad4d

SMARTS（Smiles Arbitrary Target Specification，Smiles任意目标规范）是另一种描述分子结构的语言，是在SMILES基础上的改进版，SMARTS编码允许使用符号表示原子和化学键，**一个SMARTS代表一类化学结构。**

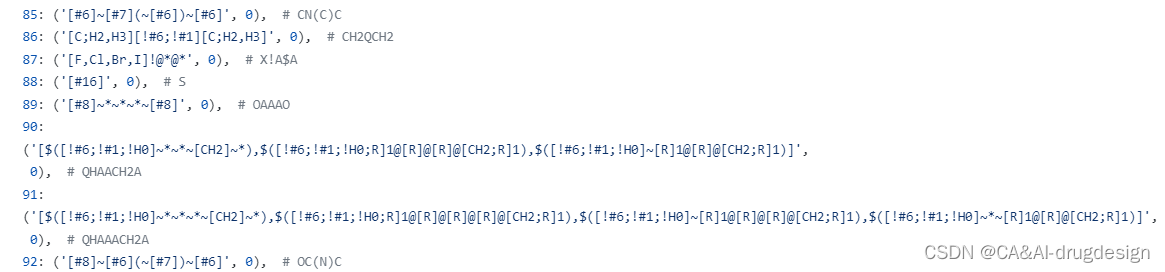
SMARTS的具体语法，其中的原子属性、键属性详见如下博客：

https://blog.csdn.net/dreadlesss/article/details/105739826

另外，我们可以基于rdkit将SMILES转化为SMARTS,具体教程见如下链接：

https://blog.csdn.net/recher\_He1107/article/details/115213129

这里以化合物**85**这个位点为例，解释每个位点代表了什么结构。



**85位点：**

(‘[#6]~[#7](~[#6])~[#6]’)

‘#6’代表’C’

‘#7’代表’N’

[#6]~[#7]代表C和N由任意键相连

~代表表示通配键

因此，85位点代表的结构为CN( C)C

引用：

1. <https://blog.csdn.net/u012325865/article/details/101697880>

2.https://github.com/rdkit/rdkit/blob/master/rdkit/Chem/MACCSkeys.py

3.https://blog.csdn.net/u012325865/article/details/101697880