



CREATING A STEP CHANGE IN MEDICINAL CHEMISTRY

SARkush®: Automated Markush-like Structure Generation for SAR Communication

Dr Lauren Reid
lauren.reid@medchemica.com

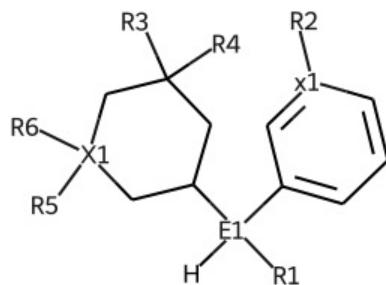
Twitter @MedChemica
www.medchemica.com

Oct 2022

Structure Activity Relationship

Markush

A structural representation designed to communicate SAR



- Variable atoms/ groups:
- x = aromatic atom
 - X = aliphatic ring atom
 - E = linker atom
 - R = side chain

compound id	X1	E1	R1	...	data
1	C	N	None	...	measurement1
2	O	C	[H]	...	measurement2
...

Importance of Markush Structures

- SAR exploration is essential in drug discovery
- Projects can explore several chemical series and generate thousands of data points
- Markush structures and R group tables are an established and effective format for SAR communication
- Markush structures can provide input to Free-Wilson or QSAR approaches
- Challenges:
 - Manual production is time-consuming
 - R-group decomposition algorithms require the user to define the core
 - Chemical space explored in projects is ever-evolving

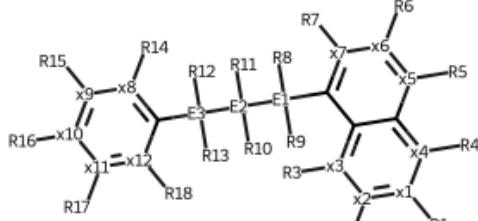
SARkush Tool Brief

- **Automatically** cluster compounds into Markush-like (**SARkush**) structures
- Remove the need for users to **separate** compounds into **chemical series** and **cores**
- Generate SARkush structure **depictions** and compound **decomposition tables**

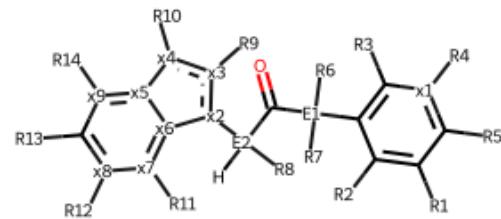
1000s of compounds in

Clustering

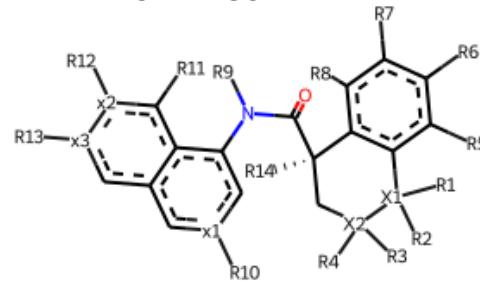
SARkush 1



SARkush 2



SARkush n



A look at the output...

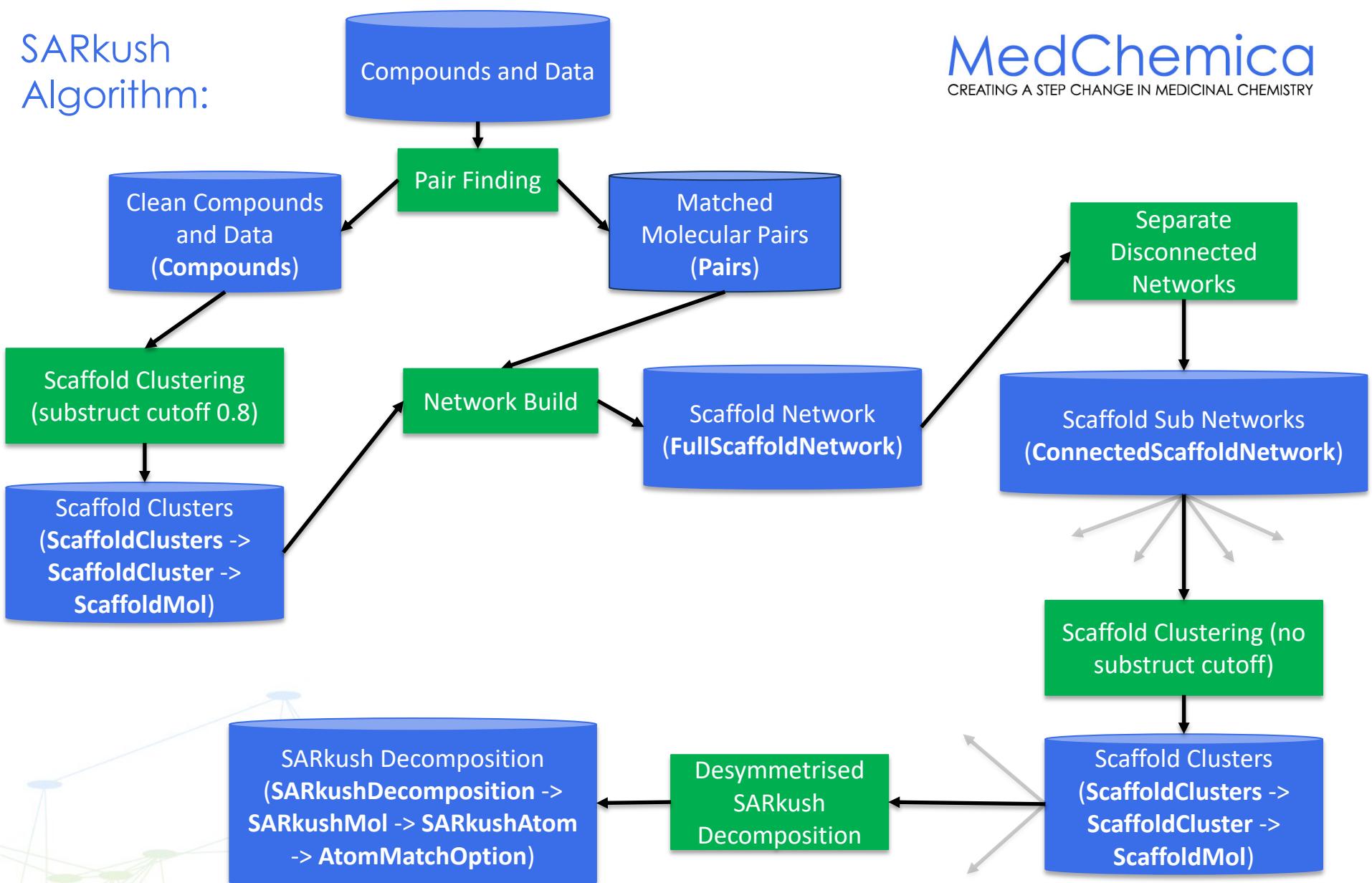
- One excel spreadsheet with SARkush structural depictions:

A	B	C	D	E	F	G	H	I	J	K	
1	network_id	sarkush_id	sarkush	no_of_compounds	no_of_compounds_cumulative	percentage_of_compounds	measurement_min	measurement_max	measurement_median	measurement_mean	measurement_mean_std_dev
2	1	1		17	17	65.38461538	5.455931956	8.301029996	7.853871964	7.310996944	0.965096097
3	1	2		9	26	34.61538462	5.908333042	8.15490196	7.102372909	7.057882674	0.801157503

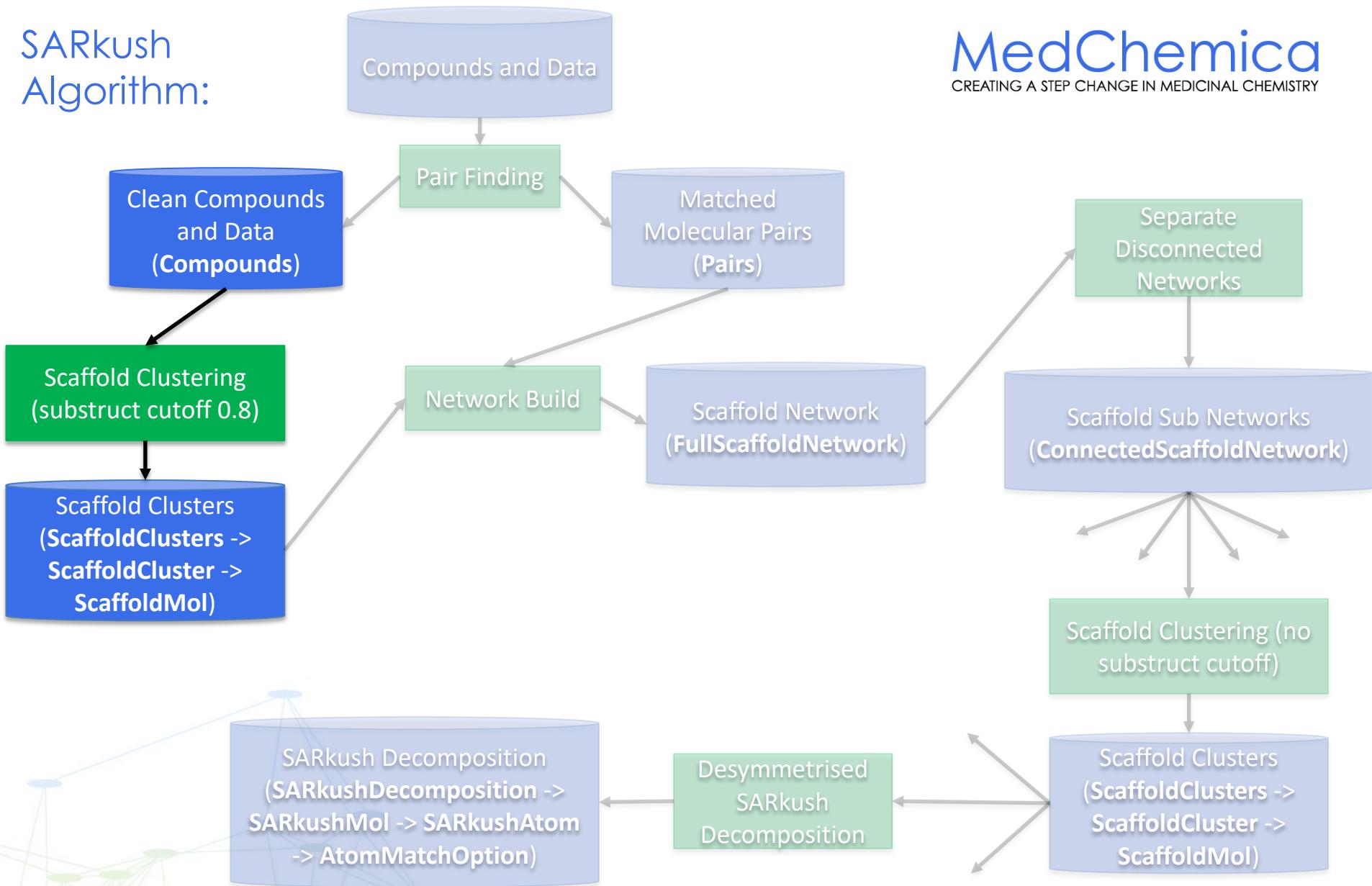
- A txt file per SARkush structure with compound decomposition:

A	C	D	E	F	G	H	I	J	K	L	M	N	
1	compound_name	SARkush_smiles	x1	R1	R2	R3	qualifier	measurement	CLogP	PSA	RMM	HBA	HBD
2	33j	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	c	[R1]Cl	[R2]C([H])([H])C1([H])C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	8.22184875	4.52	78.09	489.37	6	2
3	33i	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	c	[R1]Cl	[R2]C([H])([H])N1C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	8.301029996	3.09	81.33	490.36	7	2
4	33h	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	c	[R1]Cl	[R2]C([H])([H])N1C([H])([H])C([H])([H])N([H])C([H])([H])C1([H])([H])	[R3][H]	=	8.301029996	2.71	90.12	476.33	7	3
5	33g	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	c	[R1]Cl	[R2]N(C([H])([H])C([H])([H])C1([H])C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.602059991	4.6	81.33	504.38	7	2
6	33f	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	c	[R1]Cl	[R2]N1C([H])([H])C([H])([H])N([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.886056648	3.09	90.12	462.3	7	3
7	33k	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	c	[R1]Cl	[R2]OC1([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.853871964	4.34	87.32	491.34	7	2
8	32m	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]C([H])([H])N1C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	8.15490196	2.48	94.22	491.35	8	2
9	32l	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]C([H])([H])N1C([H])([H])C([H])([H])N([H])C([H])([H])C1([H])([H])	[R3][H]	=	8.096910013	2.09	103.01	477.32	8	3
10	32i	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]N1C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.853871964	3.35	103.01	491.35	8	3
11	32g	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]N1C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.431798276	2.61	94.22	477.32	8	2
12	32k	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]N1C([H])([H])C([H])([H])N([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.886056648	2.22	103.01	463.29	8	3
13	32b	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]C([H])([H])	[R3][H]	=	5.920818754	2.56	87.74	358.75	6	2
14	32d	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]N(C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	5.455931956	2.56	90.98	387.8	7	2
15	32f	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]N1C([H])([H])C([H])([H])N(C([H])([H])C([H])([H])C1([H])([H])	[R3][H]	=	7.142667504	1.99	94.22	442.87	8	2
16	32e	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]N1C([H])([H])C([H])([H])OC([H])([H])C1([H])([H])	[R3][H]	=	5.67780705	1.74	100.21	429.83	8	2
17	32c	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]OC([H])([H])	[R3][H]	=	6.102372909	2.31	96.97	374.75	7	2
18	32a	O=C(Nc1ncn([R2])[x1][[R3])c1c1cc(C(=O)c2c(Cl)ccc([R1])c2F)c[nH]1	n	[R1]Cl	[R2]H	[R3][H]	=	6.397940009	2	87.74	344.73	6	2

SARkush Algorithm:



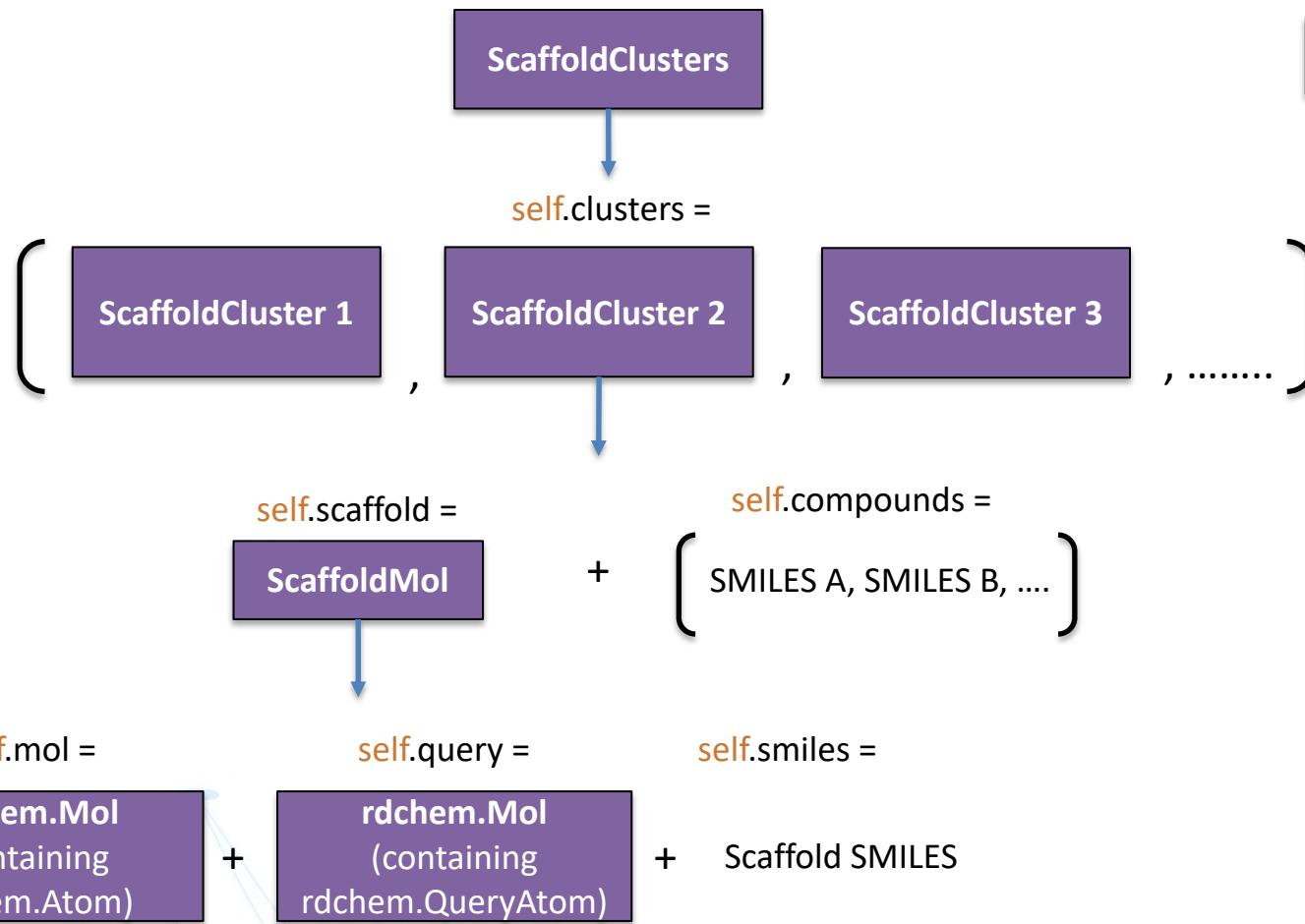
SARkush Algorithm:



Classes used in Scaffold Clustering

...

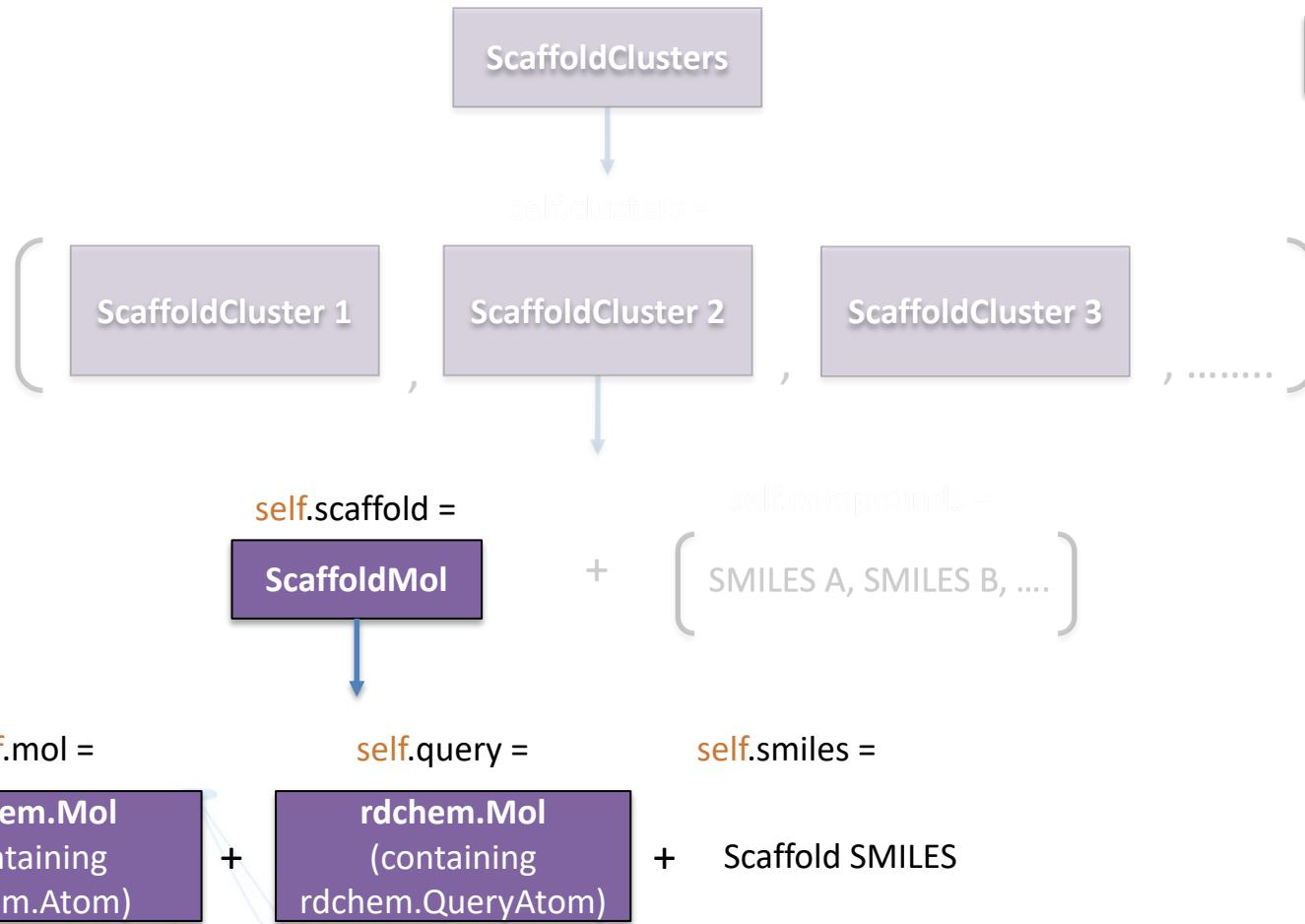
= Class



Classes used in Scaffold Clustering

...

= Class



Scaffold SMILES

- Generic * atoms with isotope labels

```
self._isotope_labels = {  
    "aliphatic_chiral": 1,  
    "carbonyl_heteroatom": 2,  
    "aliphatic_racemic": 3,  
    "E_bond_atom": 4,  
    "Z_bond_atom": 5,  
    "aromatic": 6,  
}
```

- E.g

[2*]=[3*]([3*][6*]1:[6*]:[6*]:[6*]2:[6*]:[6*]:[6*]:[6*]:[6*]:1:2)[3*]1[3*][3*][6*]2:[6*]:[6*]:[6*]:[6*]:[6*]:21

- Contains aromatic/ aliphatic and chiral/ racemic information
- Canonicalisation means identical scaffolds can be found with string matching



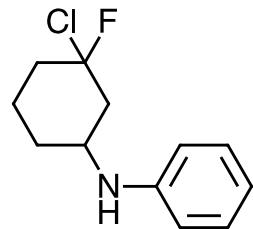
Generic Atom Scaffolds

ScaffoldMol

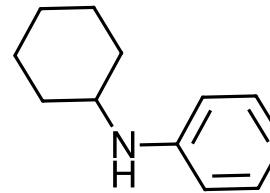
.smiles_to_scaffold_mol(smiles)

c1ccc2c(c1)cncc2NC(=O)C3CCOc4c3cc(cc4)Cl

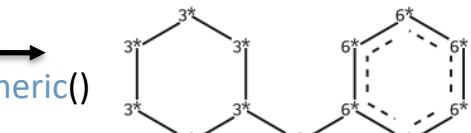
self._create_editable_mol()



self._cut_off_ring_and_branch_substituents()



self._set_atoms_to_generic()



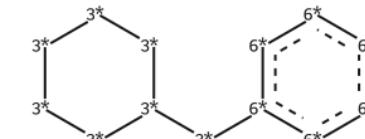
self.get_scaffold_smiles()

self.smiles =
[3*]1[3*][3*][3*](3*[6*]2:[6*]:[6*]:[6*]:2)[3*]1[3*]1

self.mol =

self._create_scaffold_query()

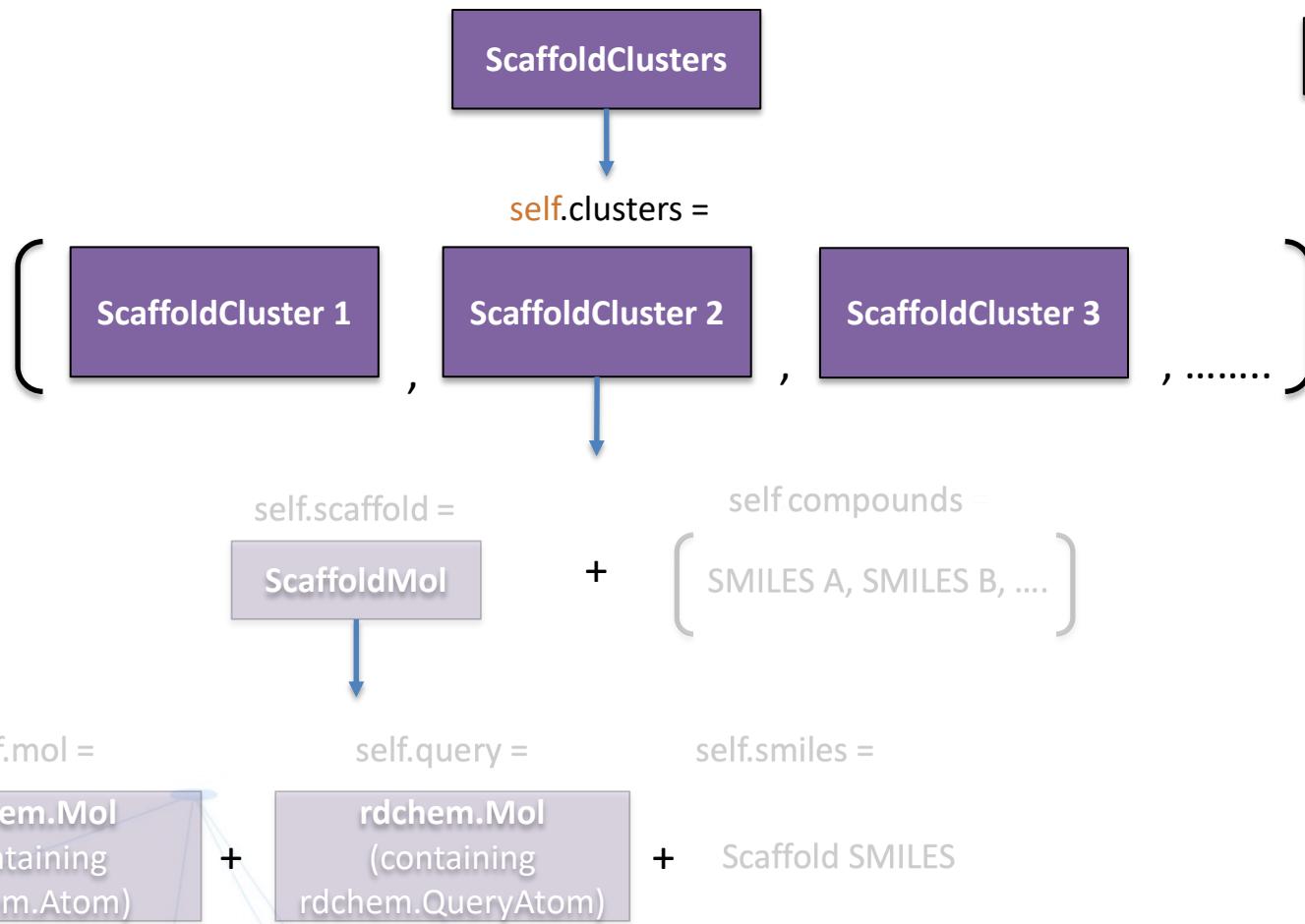
self.query =



Classes used in Scaffold Clustering

...

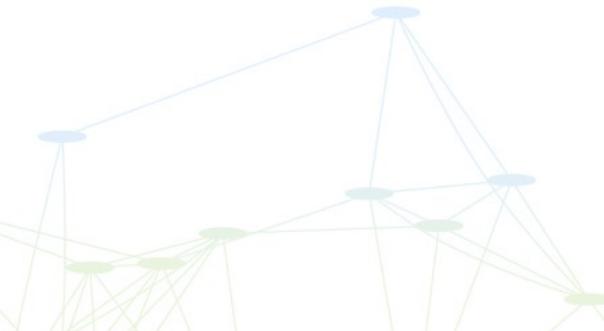
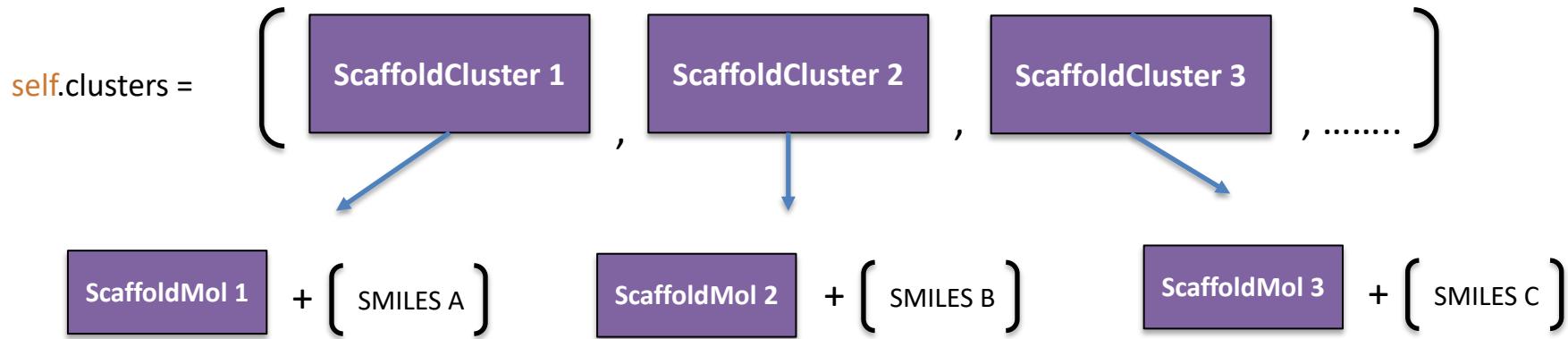
= Class



Generic Atom Scaffold Clustering

ScaffoldClusters

```
.seed_single_compound_clusters(["SMILES A", "SMILES B", "SMILES C"])
```



Generic Atom Scaffold Clustering

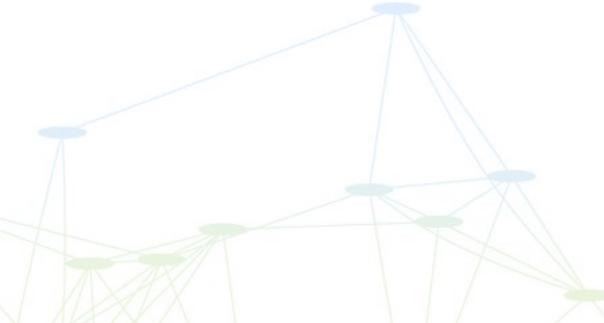
ScaffoldClusters

```
.scaffold_clustering(self, substruct_match_cutoff=0.8):
```

```
self._merge_clusters_on_string_match()
```

```
self._merge_clusters_on_substructure_match(cutoff=substruct_match_cutoff)
```

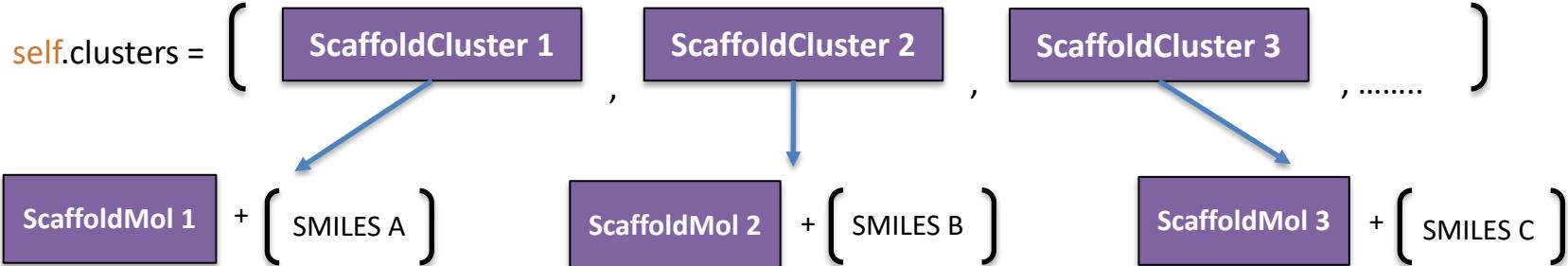
```
self._order_clusters_by_num_compounds()
```



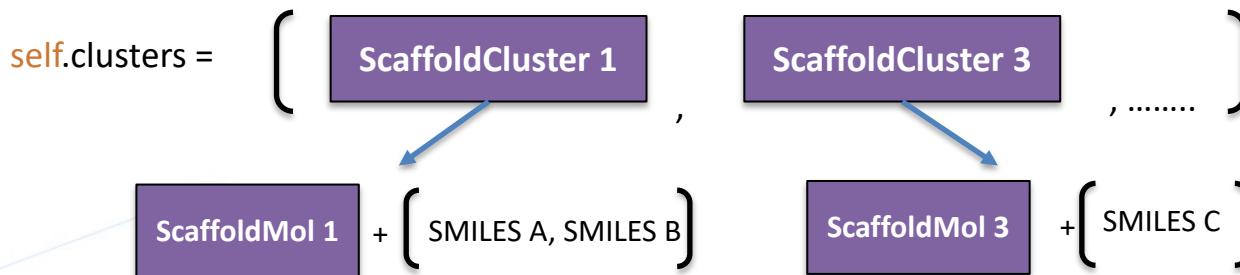
Generic Atom Scaffold Clustering

ScaffoldClusters

.`_merge_clusters_on_string_match()`



TRANSFORMS TO



IF:

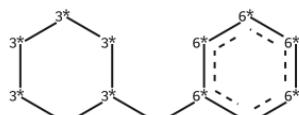
ScaffoldMol 1 .get_scaffold_smiles() == ScaffoldMol 2 .get_scaffold_smiles() != ScaffoldMol 3 .get_scaffold_smiles()

Generic Atom Scaffold Clustering

ScaffoldClusters

.`_merge_clusters_on_substructure_match(cutoff=0.8)`

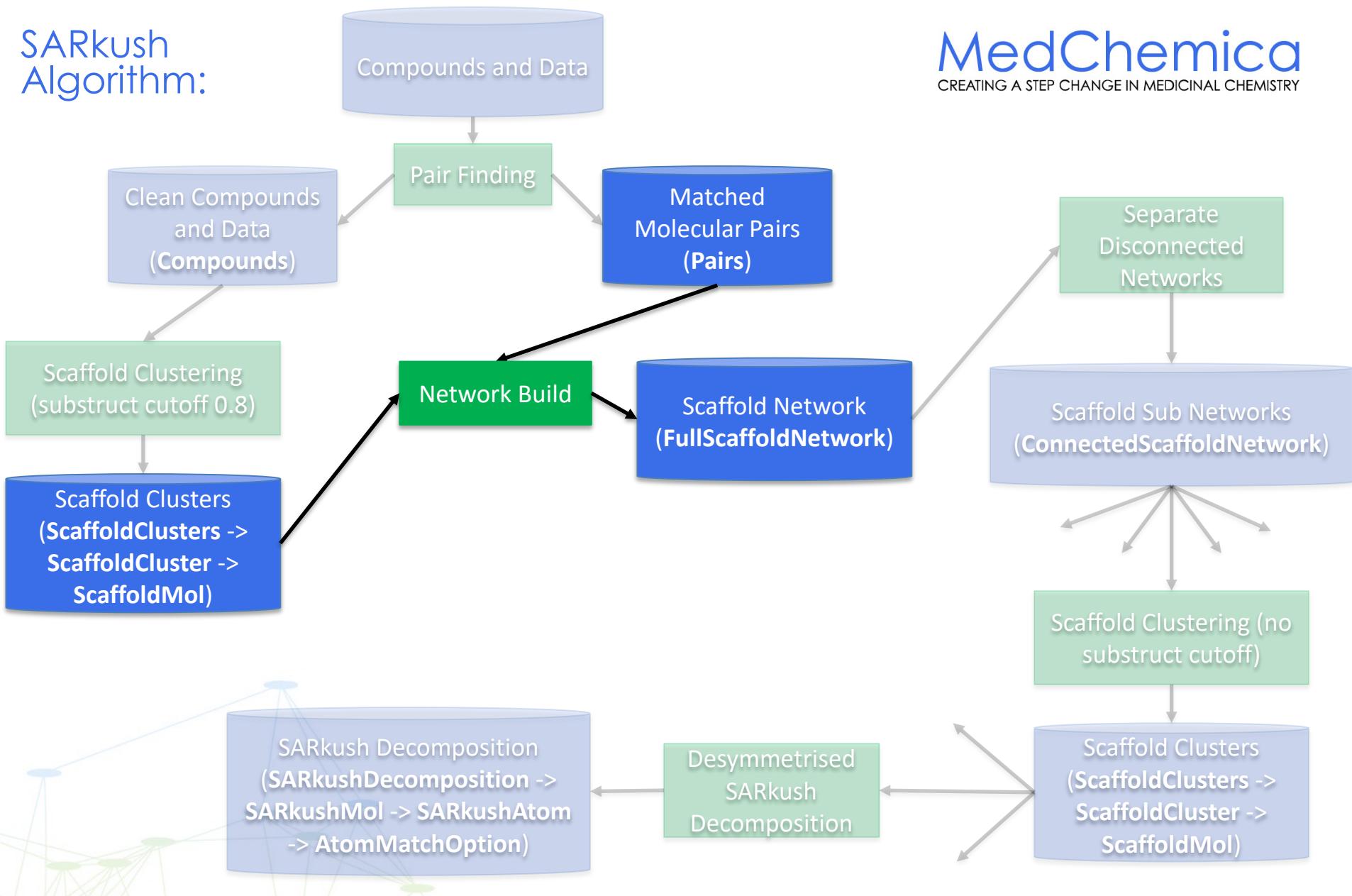
`self.clusters =`



CN1CC[C@H]2[C@@H]3[C@H]1C[C@H]2[C@H]4[C@H]3[C@H]5[C@H]4[C@H]6[C@H]5[C@H]7[C@H]6[C@H]8[C@H]7[C@H]9[C@H]8[C@H]10[C@H]9[C@H]11[C@H]10[C@H]12[C@H]11[C@H]13[C@H]12[C@H]14[C@H]13[C@H]15[C@H]14[C@H]16[C@H]15[C@H]17[C@H]16[C@H]18[C@H]17[C@H]19[C@H]18[C@H]20[C@H]19[C@H]21[C@H]18[C@H]22[C@H]19[C@H]23[C@H]21[C@H]24[C@H]22[C@H]25[C@H]23[C@H]26[C@H]24[C@H]27[C@H]25[C@H]28[C@H]26[C@H]29[C@H]27[C@H]30[C@H]28[C@H]31[C@H]29[C@H]32[C@H]30[C@H]33[C@H]21[C@H]34[C@H]32[C@H]35[C@H]22[C@H]36[C@H]34[C@H]37[C@H]23[C@H]38[C@H]35[C@H]39[C@H]24[C@H]36[C@H]40[C@H]37[C@H]41[C@H]25[C@H]38[C@H]42[C@H]39[C@H]43[C@H]26[C@H]39[C@H]44[C@H]41[C@H]45[C@H]27[C@H]42[C@H]46[C@H]43[C@H]47[C@H]28[C@H]44[C@H]48[C@H]45[C@H]49[C@H]29[C@H]46[C@H]50[C@H]47[C@H]51[C@H]20[C@H]49[C@H]52[C@H]48[C@H]53[C@H]21[C@H]50[C@H]54[C@H]49[C@H]55[C@H]22[C@H]51[C@H]56[C@H]50[C@H]57[C@H]23[C@H]52[C@H]58[C@H]51[C@H]59[C@H]24[C@H]53[C@H]60[C@H]52[C@H]61[C@H]25[C@H]54[C@H]62[C@H]53[C@H]63[C@H]26[C@H]55[C@H]64[C@H]54[C@H]65[C@H]27[C@H]56[C@H]66[C@H]55[C@H]67[C@H]28[C@H]57[C@H]68[C@H]56[C@H]69[C@H]29[C@H]58[C@H]69[C@H]57[C@H]70[C@H]30[C@H]59[C@H]71[C@H]58[C@H]72[C@H]31[C@H]60[C@H]72[C@H]59[C@H]73[C@H]32[C@H]61[C@H]73[C@H]60[C@H]74[C@H]33[C@H]62[C@H]74[C@H]61[C@H]75[C@H]34[C@H]63[C@H]75[C@H]62[C@H]76[C@H]35[C@H]64[C@H]76[C@H]63[C@H]77[C@H]36[C@H]65[C@H]77[C@H]64[C@H]78[C@H]37[C@H]66[C@H]78[C@H]65[C@H]79[C@H]38[C@H]67[C@H]79[C@H]66[C@H]80[C@H]39[C@H]68[C@H]80[C@H]67[C@H]81[C@H]40[C@H]69[C@H]81[C@H]68[C@H]82[C@H]41[C@H]70[C@H]82[C@H]69[C@H]83[C@H]42[C@H]71[C@H]83[C@H]70[C@H]84[C@H]43[C@H]72[C@H]84[C@H]71[C@H]85[C@H]44[C@H]73[C@H]85[C@H]72[C@H]86[C@H]45[C@H]74[C@H]86[C@H]73[C@H]87[C@H]46[C@H]75[C@H]87[C@H]74[C@H]88[C@H]47[C@H]76[C@H]88[C@H]75[C@H]89[C@H]48[C@H]77[C@H]89[C@H]76[C@H]90[C@H]49[C@H]78[C@H]90[C@H]77[C@H]91[C@H]50[C@H]79[C@H]91[C@H]78[C@H]92[C@H]51[C@H]80[C@H]92[C@H]79[C@H]93[C@H]52[C@H]81[C@H]93[C@H]80[C@H]94[C@H]53[C@H]82[C@H]94[C@H]81[C@H]95[C@H]54[C@H]83[C@H]95[C@H]82[C@H]96[C@H]55[C@H]84[C@H]96[C@H]83[C@H]97[C@H]56[C@H]85[C@H]97[C@H]84[C@H]98[C@H]57[C@H]86[C@H]98[C@H]85[C@H]99[C@H]58[C@H]87[C@H]99[C@H]86[C@H]100[C@H]59[C@H]88[C@H]100[C@H]87[C@H]101[C@H]60[C@H]89[C@H]101[C@H]88[C@H]102[C@H]61[C@H]90[C@H]102[C@H]89[C@H]103[C@H]62[C@H]91[C@H]103[C@H]90[C@H]104[C@H]63[C@H]92[C@H]104[C@H]91[C@H]105[C@H]64[C@H]93[C@H]105[C@H]92[C@H]106[C@H]65[C@H]94[C@H]106[C@H]93[C@H]107[C@H]66[C@H]95[C@H]107[C@H]94[C@H]108[C@H]67[C@H]96[C@H]108[C@H]95[C@H]109[C@H]68[C@H]97[C@H]109[C@H]96[C@H]110[C@H]69[C@H]98[C@H]110[C@H]97[C@H]111[C@H]70[C@H]99[C@H]111[C@H]98[C@H]112[C@H]71[C@H]100[C@H]112[C@H]99[C@H]113[C@H]72[C@H]101[C@H]113[C@H]100[C@H]114[C@H]73[C@H]102[C@H]114[C@H]101[C@H]115[C@H]74[C@H]103[C@H]115[C@H]102[C@H]116[C@H]75[C@H]104[C@H]116[C@H]103[C@H]117[C@H]76[C@H]105[C@H]117[C@H]104[C@H]118[C@H]77[C@H]106[C@H]118[C@H]105[C@H]119[C@H]78[C@H]107[C@H]119[C@H]106[C@H]120[C@H]79[C@H]108[C@H]120[C@H]107[C@H]121[C@H]70[C@H]109[C@H]121[C@H]108[C@H]122[C@H]71[C@H]110[C@H]122[C@H]109[C@H]123[C@H]72[C@H]111[C@H]123[C@H]110[C@H]124[C@H]73[C@H]112[C@H]124[C@H]111[C@H]125[C@H]74[C@H]113[C@H]125[C@H]112[C@H]126[C@H]75[C@H]114[C@H]126[C@H]113[C@H]127[C@H]76[C@H]115[C@H]127[C@H]114[C@H]128[C@H]77[C@H]116[C@H]128[C@H]115[C@H]129[C@H]78[C@H]117[C@H]129[C@H]116[C@H]130[C@H]79[C@H]118[C@H]130[C@H]117[C@H]131[C@H]70[C@H]117[C@H]131[C@H]116[C@H]132[C@H]71[C@H]118[C@H]132[C@H]117[C@H]133[C@H]72[C@H]119[C@H]133[C@H]118[C@H]134[C@H]73[C@H]120[C@H]134[C@H]119[C@H]135[C@H]74[C@H]121[C@H]135[C@H]120[C@H]136[C@H]75[C@H]122[C@H]136[C@H]121[C@H]137[C@H]76[C@H]123[C@H]137[C@H]122[C@H]138[C@H]77[C@H]124[C@H]138[C@H]123[C@H]139[C@H]78[C@H]125[C@H]139[C@H]124[C@H]140[C@H]79[C@H]126[C@H]140[C@H]125[C@H]141[C@H]70[C@H]123[C@H]141[C@H]124[C@H]142[C@H]71[C@H]124[C@H]142[C@H]125[C@H]143[C@H]72[C@H]125[C@H]143[C@H]126[C@H]144[C@H]73[C@H]126[C@H]144[C@H]127[C@H]145[C@H]74[C@H]127[C@H]145[C@H]128[C@H]146[C@H]75[C@H]128[C@H]146[C@H]129[C@H]147[C@H]76[C@H]129[C@H]147[C@H]130[C@H]148[C@H]77[C@H]130[C@H]148[C@H]131[C@H]149[C@H]78[C@H]131[C@H]149[C@H]132[C@H]150[C@H]79[C@H]132[C@H]150[C@H]133[C@H]151[C@H]70[C@H]135[C@H]151[C@H]134[C@H]152[C@H]71[C@H]136[C@H]152[C@H]135[C@H]153[C@H]72[C@H]137[C@H]153[C@H]136[C@H]154[C@H]73[C@H]138[C@H]154[C@H]137[C@H]155[C@H]74[C@H]139[C@H]155[C@H]138[C@H]156[C@H]75[C@H]140[C@H]156[C@H]139[C@H]157[C@H]76[C@H]141[C@H]157[C@H]140[C@H]158[C@H]77[C@H]142[C@H]158[C@H]141[C@H]159[C@H]78[C@H]143[C@H]159[C@H]142[C@H]160[C@H]79[C@H]144[C@H]160[C@H]143[C@H]161[C@H]70[C@H]147[C@H]161[C@H]146[C@H]162[C@H]71[C@H]148[C@H]162[C@H]147[C@H]163[C@H]72[C@H]149[C@H]163[C@H]148[C@H]164[C@H]73[C@H]150[C@H]164[C@H]149[C@H]165[C@H]74[C@H]151[C@H]165[C@H]150[C@H]166[C@H]75[C@H]152[C@H]166[C@H]151[C@H]167[C@H]76[C@H]153[C@H]167[C@H]152[C@H]168[C@H]77[C@H]154[C@H]168[C@H]153[C@H]169[C@H]78[C@H]155[C@H]169[C@H]154[C@H]170[C@H]79[C@H]156[C@H]170[C@H]155[C@H]171[C@H]70[C@H]159[C@H]171[C@H]158[C@H]172[C@H]71[C@H]160[C@H]172[C@H]159[C@H]173[C@H]72[C@H]161[C@H]173[C@H]160[C@H]174[C@H]73[C@H]162[C@H]174[C@H]161[C@H]175[C@H]74[C@H]163[C@H]175[C@H]162[C@H]176[C@H]75[C@H]164[C@H]176[C@H]163[C@H]177[C@H]76[C@H]165[C@H]177[C@H]164[C@H]178[C@H]77[C@H]166[C@H]178[C@H]165[C@H]179[C@H]78[C@H]167[C@H]179[C@H]166[C@H]180[C@H]79[C@H]168[C@H]180[C@H]167[C@H]181[C@H]70[C@H]171[C@H]181[C@H]170[C@H]182[C@H]71[C@H]172[C@H]182[C@H]171[C@H]183[C@H]72[C@H]173[C@H]183[C@H]172[C@H]184[C@H]73[C@H]174[C@H]184[C@H]173[C@H]185[C@H]74[C@H]175[C@H]185[C@H]174[C@H]186[C@H]75[C@H]176[C@H]186[C@H]175[C@H]187[C@H]76[C@H]177[C@H]187[C@H]176[C@H]188[C@H]77[C@H]178[C@H]188[C@H]177[C@H]189[C@H]78[C@H]179[C@H]189[C@H]178[C@H]190[C@H]79[C@H]180[C@H]190[C@H]179[C@H]191[C@H]70[C@H]183[C@H]191[C@H]182[C@H]192[C@H]71[C@H]184[C@H]192[C@H]183[C@H]193[C@H]72[C@H]185[C@H]193[C@H]184[C@H]194[C@H]73[C@H]186[C@H]194[C@H]185[C@H]195[C@H]74[C@H]187[C@H]195[C@H]186[C@H]196[C@H]75[C@H]188[C@H]196[C@H]187[C@H]197[C@H]76[C@H]189[C@H]197[C@H]188[C@H]198[C@H]77[C@H]190[C@H]198[C@H]189[C@H]199[C@H]78[C@H]191[C@H]199[C@H]190[C@H]200[C@H]79[C@H]192[C@H]200[C@H]191[C@H]201[C@H]70[C@H]195[C@H]201[C@H]194[C@H]202[C@H]71[C@H]196[C@H]202[C@H]195[C@H]203[C@H]72[C@H]197[C@H]203[C@H]196[C@H]204[C@H]73[C@H]198[C@H]204[C@H]197[C@H]205[C@H]74[C@H]199[C@H]205[C@H]198[C@H]206[C@H]75[C@H]200[C@H]206[C@H]199[C@H]207[C@H]76[C@H]201[C@H]207[C@H]200[C@H]208[C@H]77[C@H]202[C@H]208[C@H]201[C@H]209[C@H]78[C@H]203[C@H]209[C@H]202[C@H]210[C@H]79[C@H]204[C@H]210[C@H]203[C@H]211[C@H]70[C@H]207[C@H]211[C@H]206[C@H]212[C@H]71[C@H]208[C@H]212[C@H]207[C@H]213[C@H]72[C@H]209[C@H]213[C@H]208[C@H]214[C@H]73[C@H]210[C@H]214[C@H]209[C@H]215[C@H]74[C@H]211[C@H]215[C@H]210[C@H]216[C@H]75[C@H]212[C@H]216[C@H]211[C@H]217[C@H]76[C@H]213[C@H]217[C@H]212[C@H]218[C@H]77[C@H]214[C@H]218[C@H]213[C@H]219[C@H]78[C@H]215[C@H]219[C@H]214[C@H]220[C@H]79[C@H]216[C@H]220[C@H]215[C@H]221[C@H]70[C@H]219[C@H]221[C@H]218[C@H]222[C@H]71[C@H]220[C@H]222[C@H]219[C@H]223[C@H]72[C@H]221[C@H]223[C@H]220[C@H]224[C@H]73[C@H]222[C@H]224[C@H]221[C@H]225[C@H]74[C@H]223[C@H]225[C@H]222[C@H]226[C@H]75[C@H]224[C@H]226[C@H]223[C@H]227[C@H]76[C@H]225[C@H]227[C@H]224[C@H]228[C@H]77[C@H]226[C@H]228[C@H]225[C@H]229[C@H]78[C@H]227[C@H]229[C@H]226[C@H]230[C@H]79[C@H]228[C@H]230[C@H]227[C@H]231[C@H]70[C@H]231[C@H]229[C@H]232[C@H]71[C@H]232[C@H]230[C@H]233[C@H]72[C@H]233[C@H]231[C@H]234[C@H]73[C@H]234[C@H]232[C@H]235[C@H]74[C@H]235[C@H]233[C@H]236[C@H]75[C@H]236[C@H]234[C@H]237[C@H]76[C@H]237[C@H]235[C@H]238[C@H]77[C@H]238[C@H]236[C@H]239[C@H]78[C@H]239[C@H]237[C@H]240[C@H]79[C@H]240[C@H]238[C@H]241[C@H]70[C@H]241[C@H]239[C@H]242[C@H]71[C@H]242[C@H]240[C@H]243[C@H]72[C@H]243[C@H]241[C@H]244[C@H]73[C@H]244[C@H]242[C@H]245[C@H]74[C@H]245[C@H]243[C@H]246[C@H]75[C@H]246[C@H]244[C@H]247[C@H]76[C@H]247[C@H]245[C@H]248[C@H]77[C@H]248[C@H]246[C@H]249[C@H]78[C@H]249[C@H]247[C@H]250[C@H]79[C@H]250[C@H]248[C@H]251[C@H]70[C@H]251[C@H]249[C@H]252[C@H]71[C@H]252[C@H]250[C@H]253[C@H]72[C@H]253[C@H]251[C@H]254[C@H]73[C@H]254[C@H]252[C@H]255[C@H]74[C@H]255[C@H]253[C@H]256[C@H]75[C@H]256[C@H]254[C@H]257[C@H]76[C@H]257[C@H]255[C@H]258[C@H]77[C@H]258[C@H]256[C@H]259[C@H]78[C@H]259[C@H]257[C@H]260[C@H]79[C@H]260[C@H]258[C@H]261[C@H]70[C@H]261[C@H]259[C@H]262[C@H]71[C@H]262[C@H]260[C@H]263[C@H]72[C@H]263[C@H]261[C@H]264[C@H]73[C@H]264[C@H]262[C@H]265[C@H]74[C@H]265[C@H]263[C@H]266[C@H]75[C@H]266[C@H]264[C@H]267[C@H]76[C@H]267[C@H]265[C@H]268[C@H]77[C@H]268[C@H]266[C@H]269[C@H]78[C@H]269[C@H]267[C@H]270[C@H]79[C@H]270[C@H]268[C@H]271[C@H]70[C@H]271[C@H]269[C@H]272[C@H]71[C@H]272[C@H]270[C@H]273[C@H]72[C@H]273[C@H]271[C@H]274[C@H]73[C@H]274[C@H]272[C@H]275[C@H]74[C@H]275[C@H]273[C@H]276[C@H]75[C@H]276[C@H]274[C@H]277[C@H]76[C@H]277[C@H]275[C@H]278[C@H]77[C@H]278[C@H]276[C@H]279[C@H]78[C@H]279[C@H]277[C@H]280[C@H]79[C@H]280[C@H]278[C@H]281[C@H]70[C@H]281[C@H]279[C@H]282[C@H]71[C@H]282[C@H]280[C@H]283[C@H]72[C@H]283[C@H]281[C@H]284[C@H]73[C@H]284[C@H]282[C@H]285[C@H]74[C@H]285[C@H]283[C@H]286[C@H]75[C@H]286[C@H]284[C@H]287[C@H]76[C@H]287[C@H]285[C@H]288[C@H]77[C@H]288[C@H]286[C@H]289[C@H]78[C@H]289[C@H]287[C@H]290[C@H]79[C@H]290[C@H]288[C@H]291[C@H]70[C@H]291[C@H]289[C@H]292[C@H]71[C@H]292[C@H]290[C@H]293[C@H]72[C@H]293[C@H]291[C@H]294[C@H]73[C@H]294[C@H]292[C@H]295[C@H]74[C@H]295[C@H]293[C@H]296[C@H]75[C@H]296[C@H]294[C@H]297[C@H]76[C@H]297[C@H]295[C@H]298[C@H]77[C@H]298[C@H]296[C@H]299[C@H]78[C@H]299[C@H]297[C@H]300[C@H]79[C@H]300[C@H]298[C@H]301[C@H]70[C@H]301[C@H]299[C@H]302[C@H]71[C@H]302[C@H]300[C@H]303[C@H]72[C@H]303[C@H]301[C@H]304[C@H]73[C@H]304[C@H]302[C@H]305[C@H]74[C@H]305[C@H]303[C@H]306[C@H]75[C@H]306[C@H]304[C@H]307[C@H]76[C@H]307[C@H]305[C@H]308[C@H]77[C@H]308[C@H]306[C@H]309[C@H]78[C@H]309[C@H]307[C@H]310[C@H]79[C@H]310[C@H]308[C@H]311[C@H]70[C@H]311[C@H]309[C@H]312[C@H]71[C@H]312[C@H]310[C@H]313[C@H]72[C@H]313[C@H]311[C@H]314[C@H]73[C@H]314[C@H]312[C@H]315[C@H]74[C@H]315[C@H]313[C@H]316[C@H]75[C@H]316[C@H]314[C@H]317[C@H]76[C@H]317[C@H]315[C@H]318[C@H]77[C@H]318[C@H]316[C@H]319[C@H]78[C@H]319[C@H]317[C@H]320[C@H]79[C@H]320[C@H]318[C@H]321[C@H]70[C@H]321[C@H]319[C@H]322[C@H]71[C@H]322[C@H]320[C@H]323[C@H]72[C@H]323[C@H]321[C@H]324[C@H]73[C@H]324[C@H]322[C@H]325[C@H]74[C@H]325[C@H]323[C@H]326[C@H]75[C@H]326[C@H]324[C@H]327[C@H]76[C@H]327[C@H]325[C@H]328[C@H]77[C@H]328[C@H]326[C@H]329[C@H]78[C@H]329[C@H]327[C@H]330[C@H]79[C@H]330[C@H]328[C@H]331[C@H]70[C@H]331[C@H]329[C@H]332[C@H]71[C@H]332[C@H]330[C@H]333[C@H]72[C@H]333[C@H]331[C@H]334[C@H]73[C@H]334[C@H]332[C@H]335[C@H]74[C@H]335[C@H]333[C@H]336[C@H]75[C@H]336[C@H]334[C@H]337[C@H]76[C@H]337[C@H]335[C@H]338[C@H]77[C@H]338[C@H]336[C@H]339[C@H]78[C@H]339[C@H]337[C@H]340[C@H]79[C@H]340[C@H]338[C@H]341[C@H]70[C@H]341[C@H]339[C@H]342[C@H]71[C@H]342[C@H]340[C@H]343[C@H]72[C@H]343[C@H]341[C@H]344[C@H]73[C@H]344[C@H]342[C@H]345[C@H]74[C@H]345[C@H]343[C@H]346[C@H]75[C@H]346[C@H]344[C@H]347[C@H]76[C@H]347[C@H]345[C@H]348[C@H]77[C@H]348[C@H]346[C@H]349[C@H]78[C@H]349[C@H]347[C@H]350[C@H]79[C@H]350[C@H]348[C@H]351[C@H]70[C@H]351[C@H]349[C@H]352[C@H]71[C@H]352[C@H]350[C@H]353[C@H]72[C@H]353[C@H]351[C@H]354[C@H]73[C@H]354[C@H]352[C@H]355[C@H]74[C@H]355[C@H]353[C@H]356[C@H]75[C@H]356[C@H]354[C@H]357[C@H]76[C@H]357[C@H]355[C@H]358[C@H]77[C@H]358[C@H]356[C@H]359[C@H]78[C@H]359[C@H]357[C@H]360[C@H]79[C@H]360[C@H]358[C@H]361[C@H]70[C@H]361[C@H]359[C@H]362[C@H]71[C@H]362[C@H]360[C@H]363[C@H]72[C@H]363[C@H]361[C@H]364[C@H]73[C@H]364[C@H]362[C@H]365[C@H]74[C@H]365[C@H]363[C@H]366[C@H]75[C@H]366[C@H]364[C@H]367[C@H]76[C@H]367[C@H]365[C@H]368[C@H]77[C@H]368[C@H]366[C@H]369[C@H]78[C@H]369[C@H]367[C@H]370[C@H]79[C@H]370[C@H]368[C@H]371[C@H]70[C@H]371[C@H]369[C@H]372[C@H]71[C@H]372[C@H]370[C@H]373[C@H]72[C@H]373[C@H]371[C@H]374[C@H]73[C@H]374[C@H]372[C@H]375[C@H]74[C@H]375[C@H]373[C@H]376[C@H]75[C@H]376[C@H]374[C@H]377[C@H]76[C@H]377[C@H]375[C@H]378[C@H]77[C@H]378[C@H]376[C@H]379[C@H]78[C@H]379[C@H]377[C@H]380[C@H]79[C@H]380[C@H]378[C@H]381[C@H]70[C@H]381[C@H]379[C@H]382[C@H]71[C@H]382[C@H]380[C@H]383[C@H]72[C@H]383[C@H]381[C@H]384[C@H]73[C@H]384[C@H]382[C@H]385[C@H]74[C@H]385[C@H]383[C@H]386[C@H]75[C@H]386[C@H]384[C@H]387[C@H]76[C@H]387[C@H]385[C@H]388[C@H]77[C@H]388[C@H]386[C@H]389[C@H]78[C@H]389[C@H]387[C@H]390[C@H]79[C@H]390[C@H]388[C@H]391[C@H]70[C@H]391[C@H]389[C@H]392[C@H]71[C@H]392[C@H]390[C@H]393[C@H]72[C@H]393[C@H]391[C@H]394[C@H]73[C@H]394[C@H]392[C@H]395[C@H]74[C@H]395[C@H]393[C@H]396[C@H]75[C@H]396[C@H]394[C@H]397[C@H]76[C@H]397[C@H]395[C@H]398[C@H]77[C@H]398[C@H]396[C@H]399[C@H]78[C@H]399[C@H]397[C@H]400[C@H]79[C@H]400[C@H]398[C@H]401[C@H]70[C@H]401[C@H]399[C@H]402[C@H]71[C@H]402[C@H]400[C@H]403[C@H]72[C@H]403[C@H]401[C@H]404[C@H]73[C@H]404[C@H]402[C@H]405[C@H]74[C@H]405[C@H]403[C@H]406[C@H]75[C@H]406[C@H]404[C@H]407[C@H]76[C@H]407[C@H]405[C@H]408[C@H]77[C@H]408[C@H]406[C@H]409[C@H]78[C@H]409[C@H]407[C@H]410[C@H]79[C@H]410[C@H]408[C@H]411[C@H]70[C@H]411[C@H]409[C@H]412[C@H]71[C@H]412[C@H]410[C@H]413[C@H]72[C@H]413[C@H]411[C@H]414[C@H]73[C@H]414[C@H]412[C@H]415[C@H]74[C@H]415[C@H]413[C@H]416[C@H]75[C@H]416[C@H]414[C@H]417[C@H]76[C@H]417[C@H]415[C@H]418[C@H]77[C@H]418[C@H]416[C@H]419[C@H]78[C@H]419[C@H]417[C@H]420[C@H]79[C@H]420[C@H]418[C@H]421[C@H]70[C@H]421[C@H]419[C@H]422[C@H]71[C@H]422[C@H]420[C@H]423[C@H]72[C@H]423[C@H]421[C@H]424[C@H]73[C@H]424[C@H]422[C@H]425[C@H]74[C@H]425[C@H]423[C@H]426[C@H]75[C@H]426[C@H]424[C@H]427[C@H]76[C@H]427[C@H]425[C@H]428[C@H]77[C@H]428[C@H]426[C@H]429[C@H]78[C@H]429[C@H]427[C@H]430[C@H]79[C@H]430[C@H]428[C@H]431[C@H]70[C@H]431[C@H]429[C@H]432[C@H]71[C@H]432[C@H]430[C@H]433[C@H]72[C@H]433[C@H]431[C@H]434[C@H]73[C@H]434[C@H]432[C@H]435

SARkush Algorithm:

MedChemica
CREATING A STEP CHANGE IN MEDICINAL CHEMISTRY



Scaffold Network

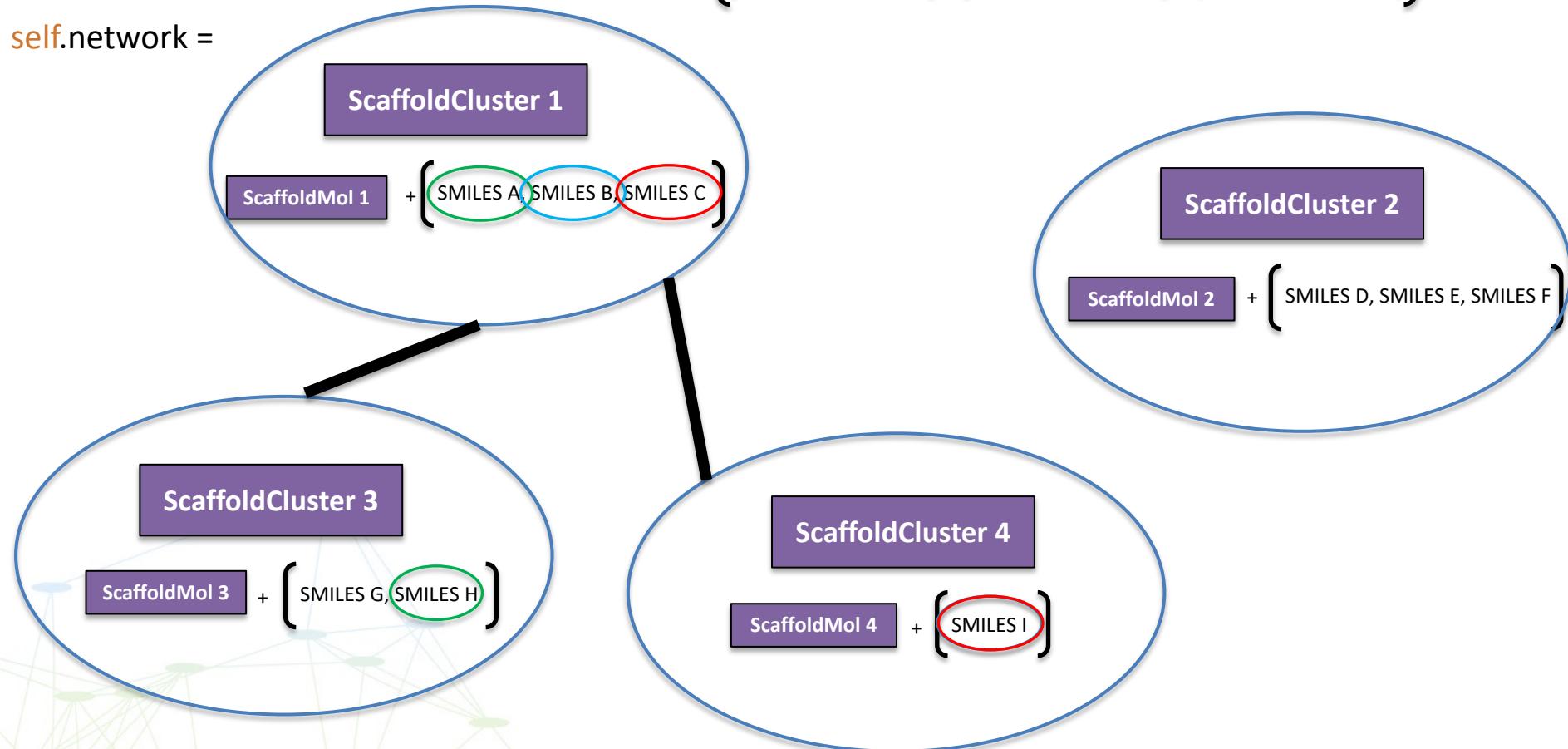
MedChemica
CREATING A STEP CHANGE IN MEDICINAL CHEMISTRY

FullScaffoldNetwork

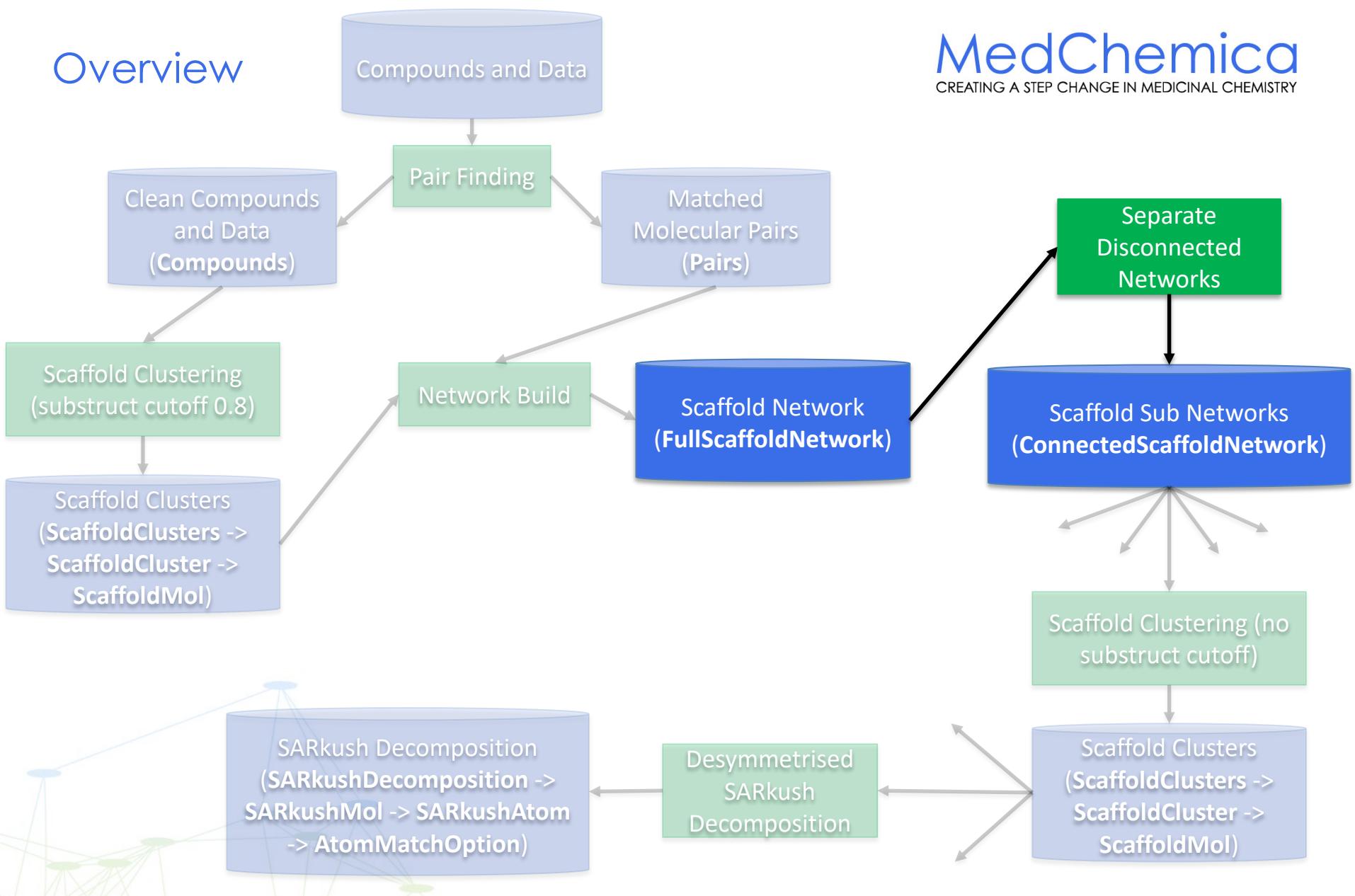
.build_full_network(ScaffoldClusters , Pairs)

{ SMILES A, SMILES H }, { SMILES B, SMILES I }, { SMILES C, SMILES I }

self.network =



Overview



Scaffold Network

FullScaffoldNetwork

.separate_networks()

self.sub_networks =

ConnectedScaffoldNetwork 1

ConnectedScaffoldNetwork 2

ScaffoldCluster 1

ScaffoldMol 1

+ { SMILES A, SMILES B, SMILES C }

ScaffoldCluster 2

ScaffoldMol 2

+ { SMILES D, SMILES E, SMILES F }

ScaffoldCluster 3

ScaffoldMol 3

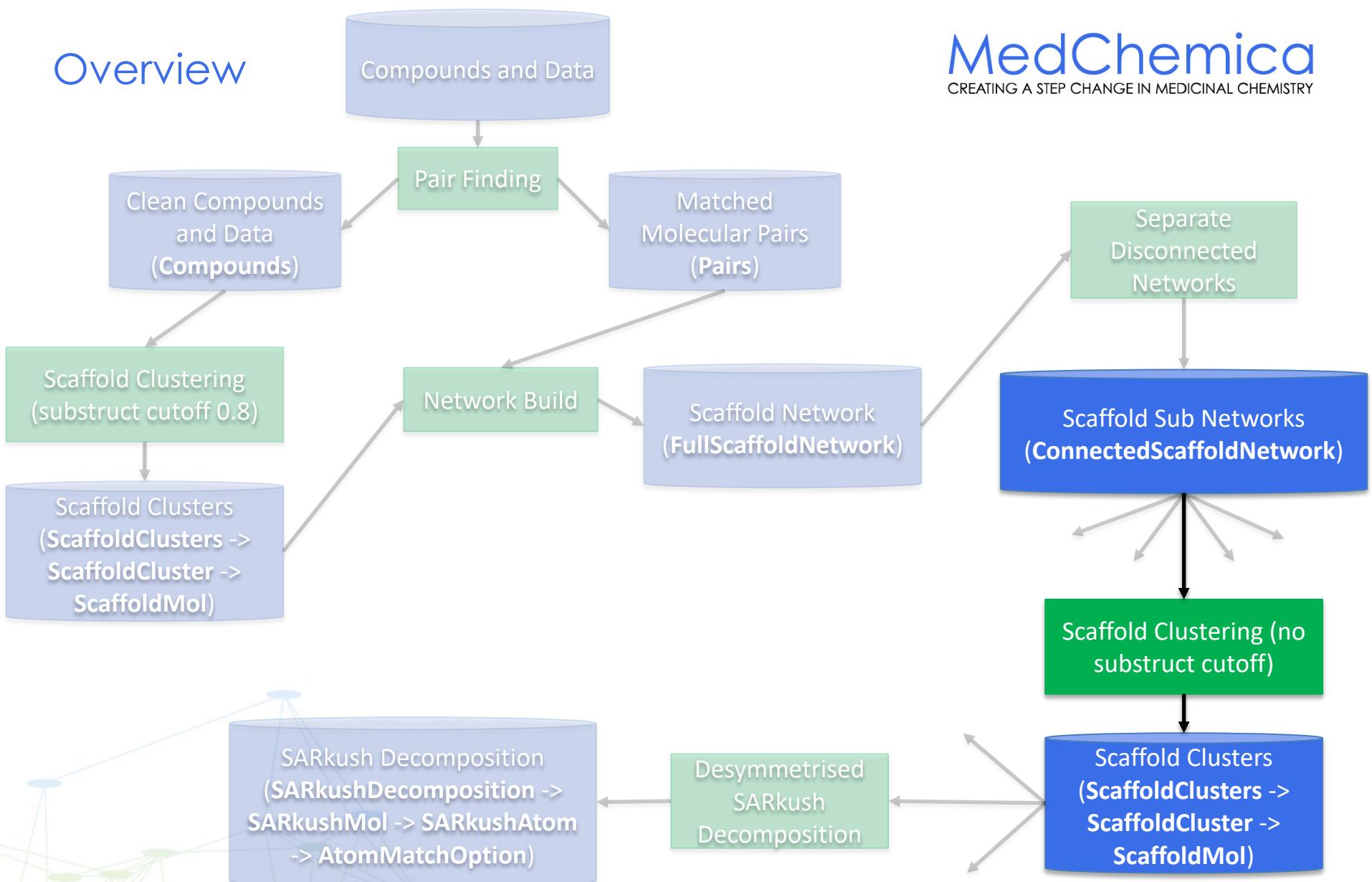
+ { SMILES G, SMILES H }

ScaffoldCluster 4

ScaffoldMol 4

+ { SMILES I }

Overview



Generic Atom Scaffold Clustering

ConnectedScaffoldNetwork 1

ConnectedScaffoldNetwork 2



ScaffoldClusters 1

ScaffoldClusters 2

.scaffold_clustering(substruct_match_cutoff=0.0):

.scaffold_clustering(substruct_match_cutoff=0.0):

self.clusters =

ScaffoldCluster 1

ScaffoldCluster 2

ScaffoldCluster 3

,

self.clusters =

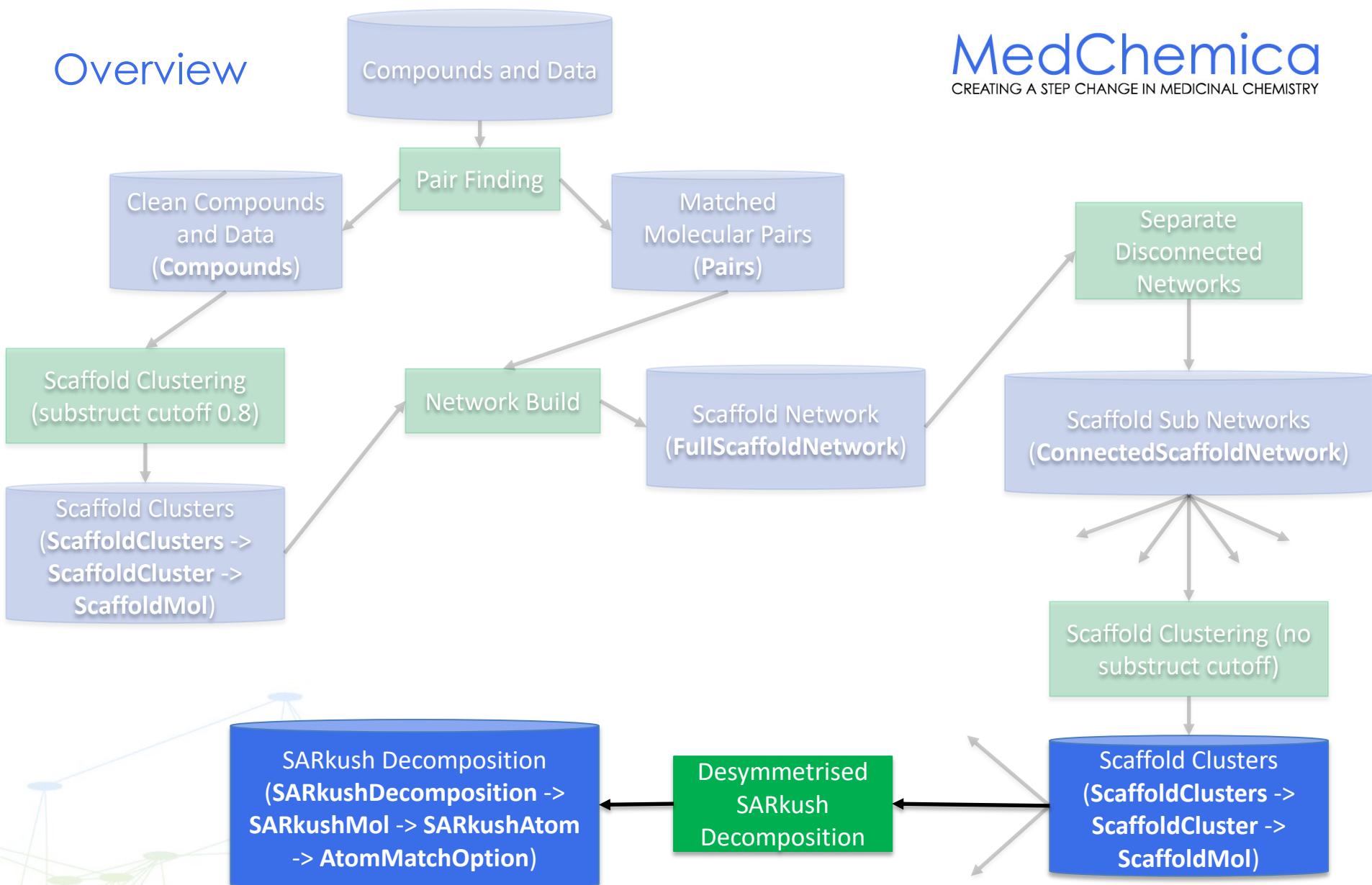
ScaffoldCluster 4

ScaffoldCluster 5

,

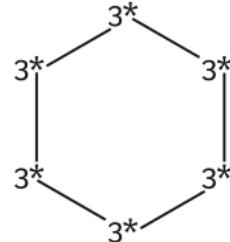
- A ScaffoldClusters object is created from each ConnectedScaffoldNetwork, and scaffold clustering is performed with no substructure match cutoff
- We're left with multiple ScaffoldCluster objects, originating from one of multiple ConnectedScaffoldNetworks

Overview

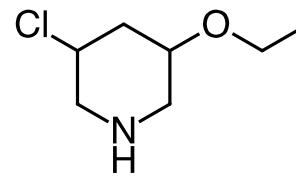


What do we mean by “desymmetrised decomposition”?

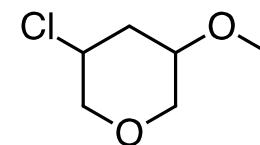
Scaffold



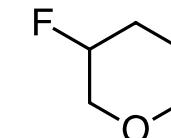
Compounds



1

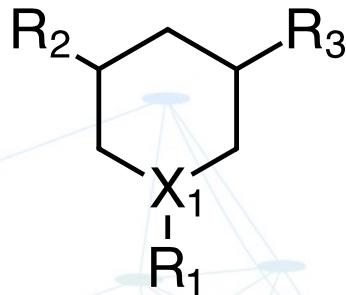


2



3

SARkush Decomposition:

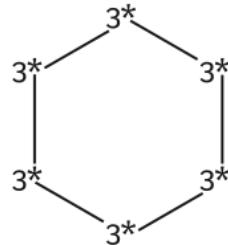


compound	X1	R1	R2	R3
1	N	[R1][H]	[R2]Cl	[R3]OCC
2	O	None	[R2]Cl	[R3]OC
3	O	None	[R2]F	[R3][H]

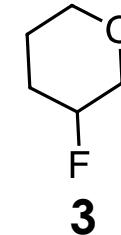
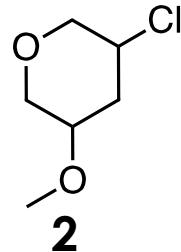
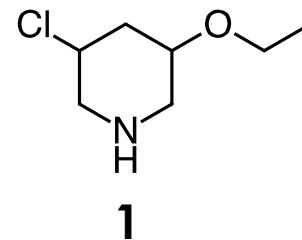


What do we mean by “desymmetrised decomposition”?

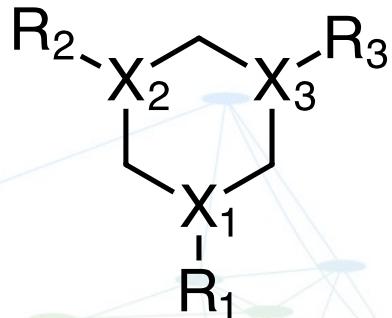
Scaffold



Compounds



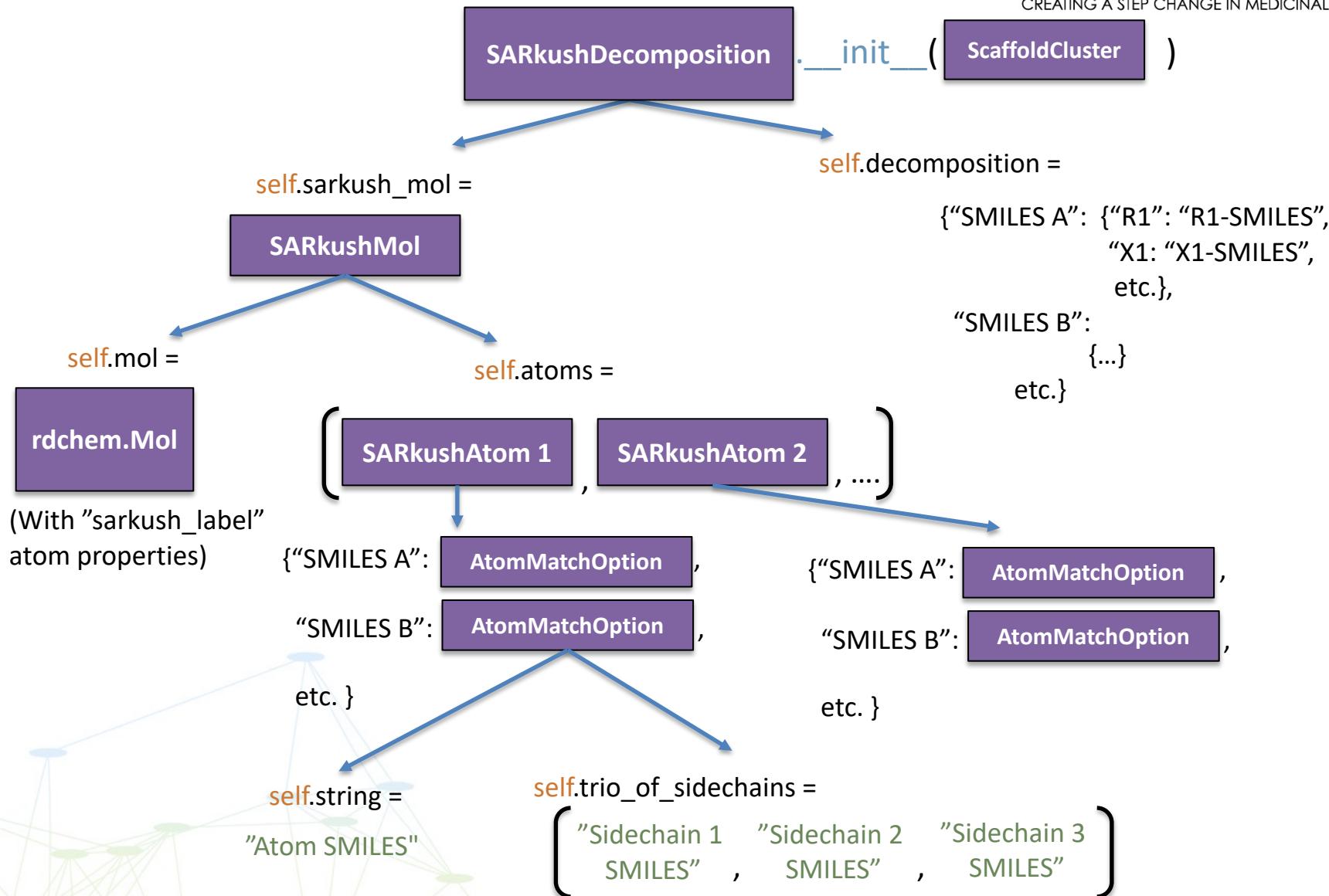
SARKush Decomposition:



compound	X1	X2	X3	R1	R2	R3
1	N	C	C	[R1]H	[R2]Cl	[R3]OCC
2	C	O	C	[R1]OC	None	[R3]Cl
3	C	C	O	[R1]F	[R2][H]	None

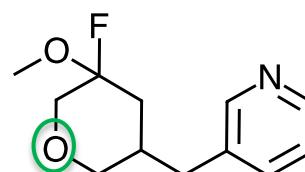
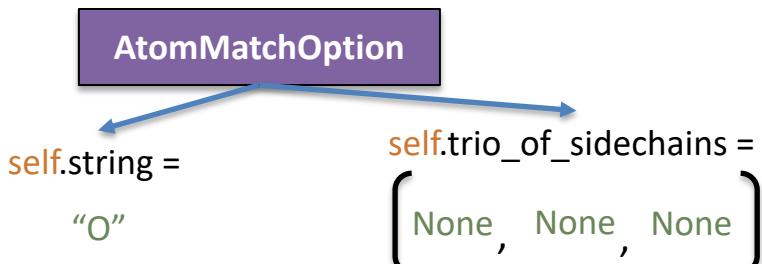
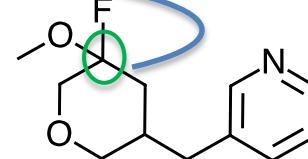
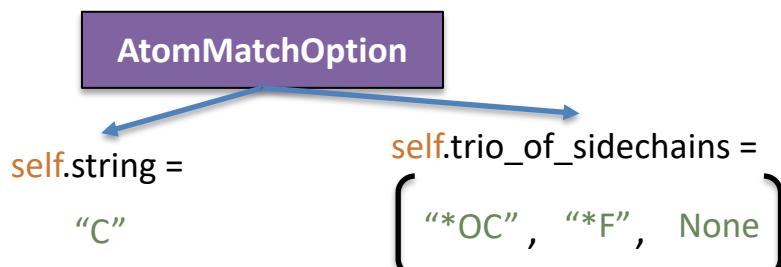
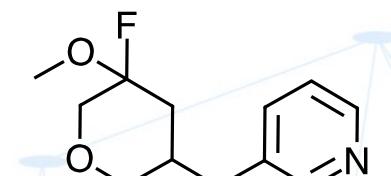
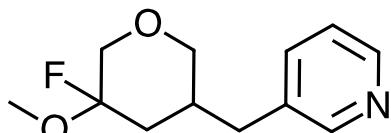
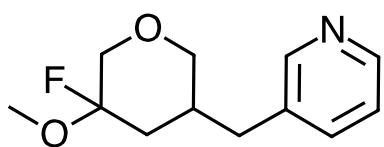
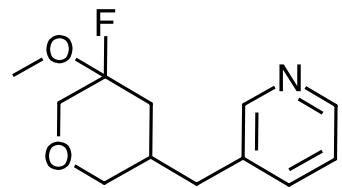
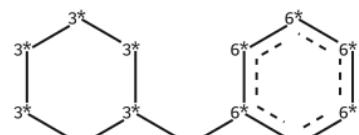


Classes used in SARkush Decomposition



AtomMatchOption

MedChemica
CREATING A STEP CHANGE IN MEDICINAL CHEMISTRY



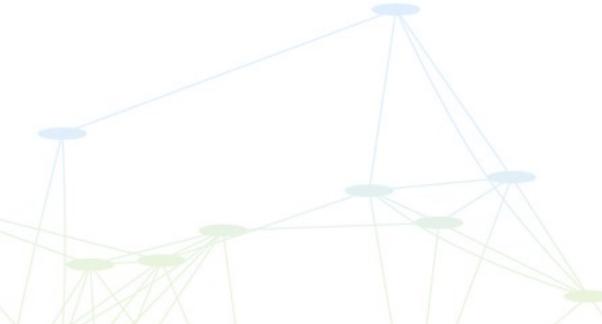
SARkush Decomposition

```
SARkushDecomposition . __init__( ScaffoldCluster , report_groups_implicitly=False)
```

```
self._perform_decomposition( ScaffoldCluster .compounds, ScaffoldCluster .scaffold)
```

```
self._desymmetrised_atom_assignment(compounds, ScaffoldMol )
```

```
self._atom_assignments_to_sarkush_decomposition()
```



```
self._desymmetrised_atom_assignment(compounds, ScaffoldMol )
```

```
self.sarkush_mol = SARkushMol .__init__( ScaffoldMol )
```

Finds ANY atom match between Scaffold and Mol

- Iterate over compounds in order of descending number of sidechains:

- For first compound:

```
match_atom_ids = ScaffoldMol .get_substructure_match_for_mol( rdchem.Mol )
```

```
mol_match = [ AtomMatchOption for atom_id in match_atom_ids]
```

Finds ALL atom matches between Scaffold and Mol

```
self. SARkushMol .add_mol_match(mol_match, compound_SMILES)
```

- For following compounds:

```
mol_match_options = ScaffoldMol .get_substructure_matches_for_mol( rdchem.Mol )
```

- Iterate over mol_match_options:

```
mol_match = [ AtomMatchOption for atom_id in match_atom_ids]
```

```
match_score = self. SARkushMol .score_new_mol_match_option(mol_match)
```

- For highest scoring mol_match:

```
self. SARkushMol .add_mol_match(best_mol_match, compound_SMILES)
```

SARkushMol

self.atoms =

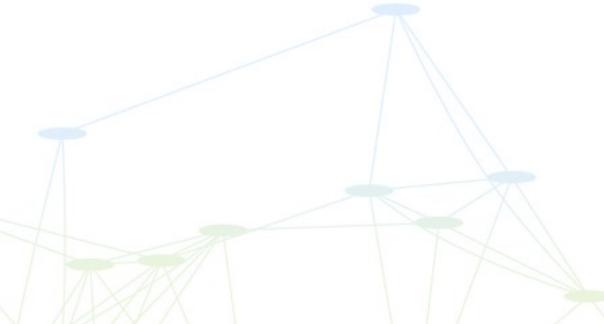
$\left[\begin{array}{c} \text{SARkushAtom 1} \\ , \\ \text{SARkushAtom 2} \\ , \\ \dots \end{array} \right]$

self.score_new_mol_match_option($\left[\begin{array}{c} \text{AtomMatchOption 1} \\ , \\ \text{AtomMatchOption 2} \\ , \\ \dots \end{array} \right]$)

- **Iterate over SARkushAtoms self.atoms :**

atom_score = $\text{SARkushAtom } i \cdot \text{score_new_atom_match_option(AtomMatchOption } i \text{)}$

- **Return cumulative atom score**



SARkushAtom

{"SMILES A": AtomMatchOption A,
 "SMILES B": AtomMatchOption B,
 etc. }

String
 comparisons
 using own scoring
 function

```
self.possible_sidechain_matches =  

[ [0, 1, 2], [0, 2, 1],  

[1, 0, 2], [1, 2, 0],  

[2, 0, 1], [2, 1, 0] ]
```

self.score_new_atom_match_option(AtomMatchOption C)

- Iterate over compound AtomMatchOptions in self :

cumulative_atom_score += AtomMatchOption C .atom_similarity(AtomMatchOption)

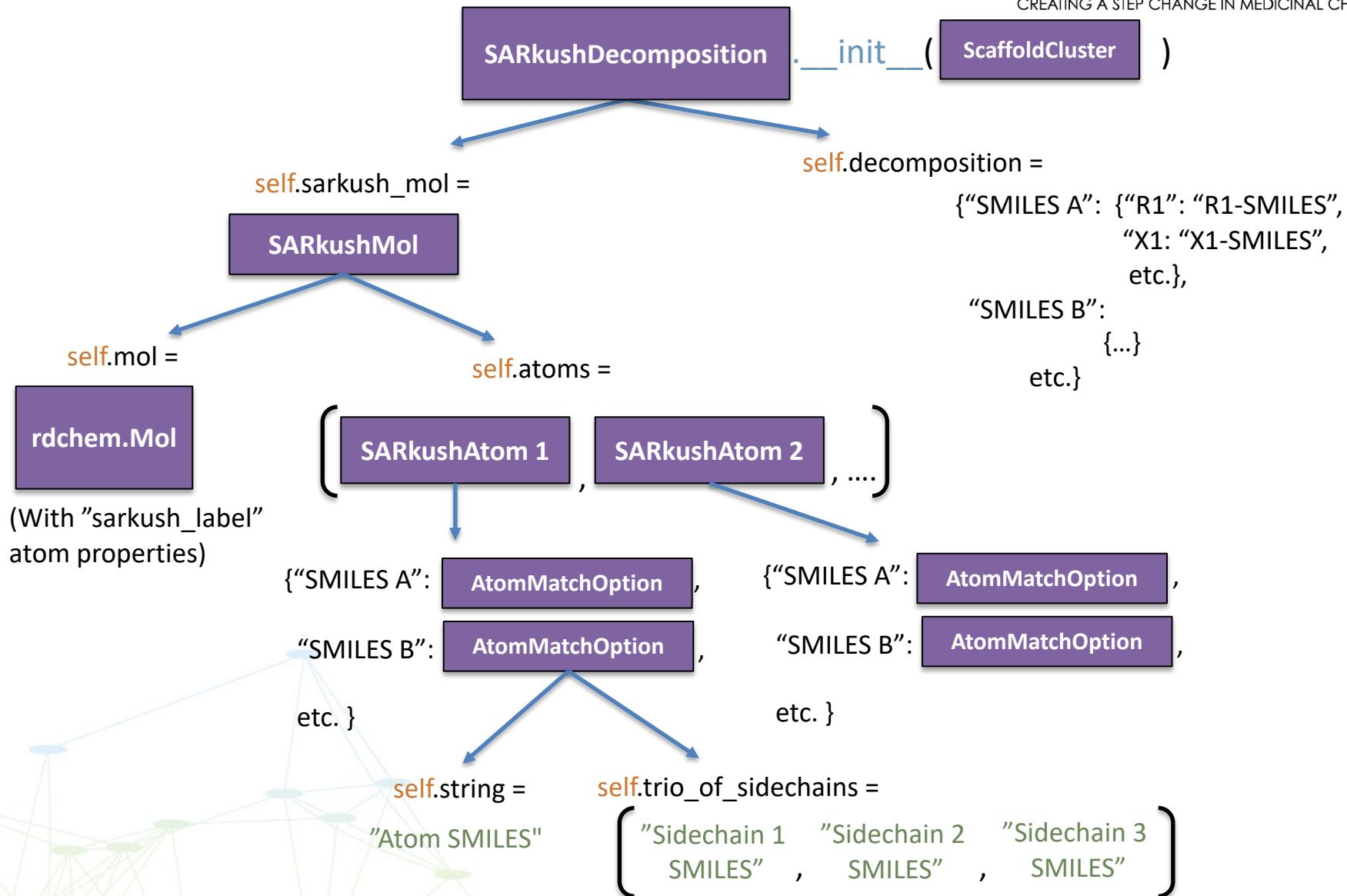
- Iterate over self.possible_sidechain_matches:
 - Iterate over sidechain ids in sidechain match option:

sidechain_match_score += AtomMatchOption C .sidechain_similarity(new_sidechain_id,

AtomMatchOption ,
 sidechain_id)

- Store best scoring sidechain match option
- Return sum of cumulative atom score and cumulative best sidechain match score

Classes used in SARkush Decomposition



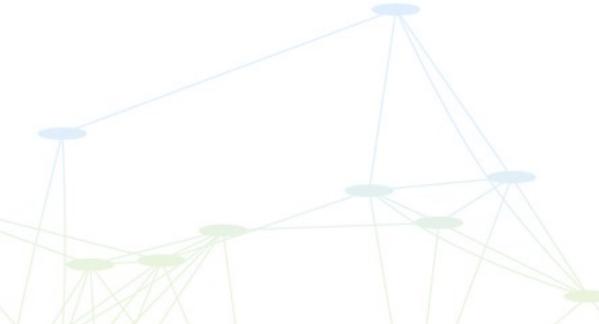
SARkush Decomposition

```
SARkushDecomposition . __init__( ScaffoldCluster , report_groups_implicitly=False)
```

```
self._perform_decomposition( ScaffoldCluster .compounds, ScaffoldCluster .scaffold)
```

```
self._desymmetrised_atom_assignment(compounds, ScaffoldMol )
```

```
self._atom_assignments_to_sarkush_decomposition()
```



SARKushDecomposition

self.decomposition = {}

self.sarkush_mol =

SARKushMol

self.mol =

rdchem.Mol

self.atoms =

SARKushAtom 1

SARKushAtom 2

, ...,

self._atom_assignments_to_sarkush_decomposition()

- Iterate over SARKushAtoms in self. SARKushMol .atoms:

```
if SARKushAtom .is_constant():
```

```
    self. SARKushMol .replace_variable_atom_with_constant_atom(sarkush_atom_id, SARKushAtom .return_atom_i_string(0))
```

```
    self._add_sidechains_to_sarkush_mol(sarkush_atom_id, SARKushAtom )
```

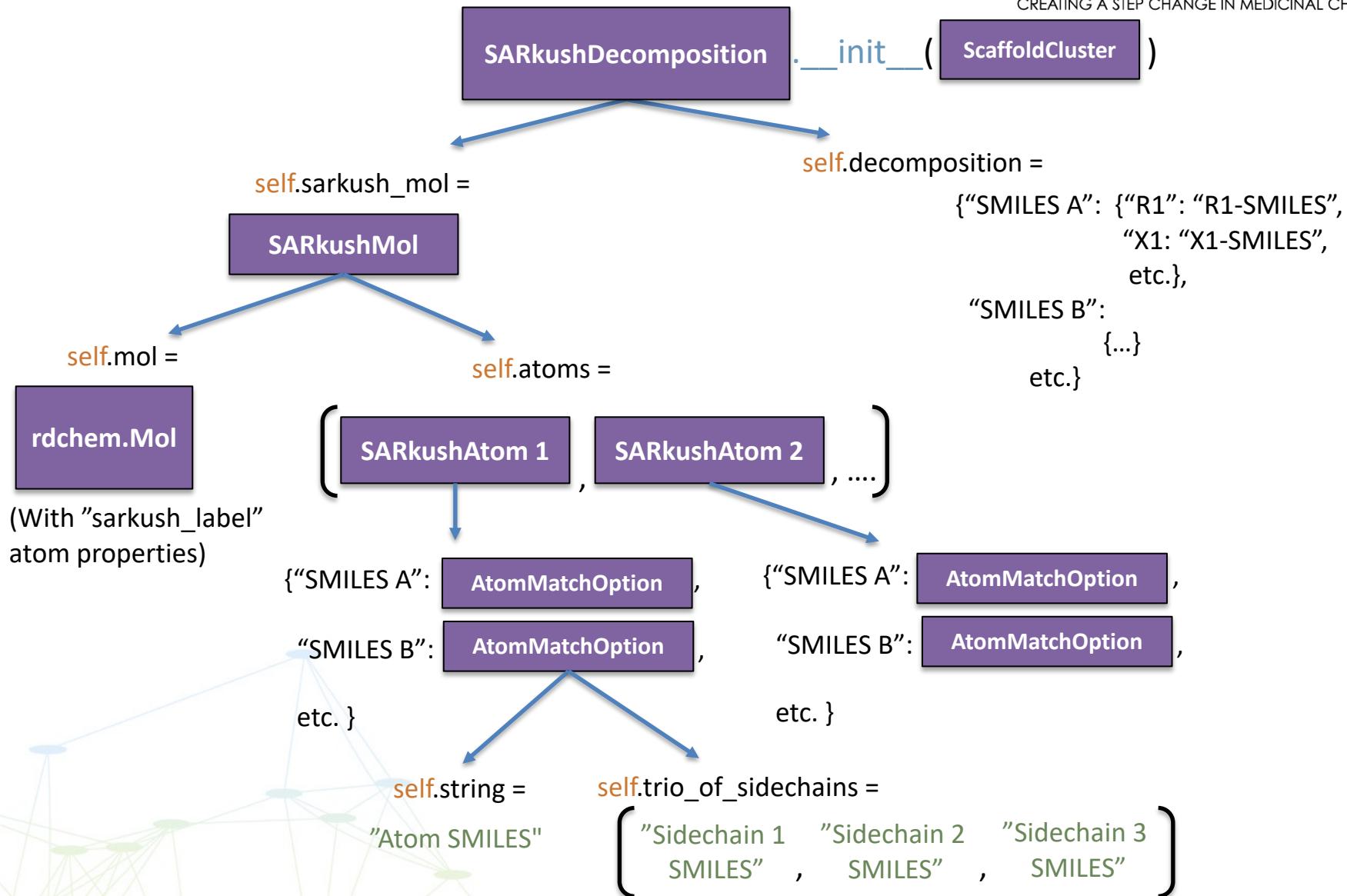
else:

```
    sarkush_label = self. SARKushMol .set_atom_sarkush_label(sarkush_atom_id)
```

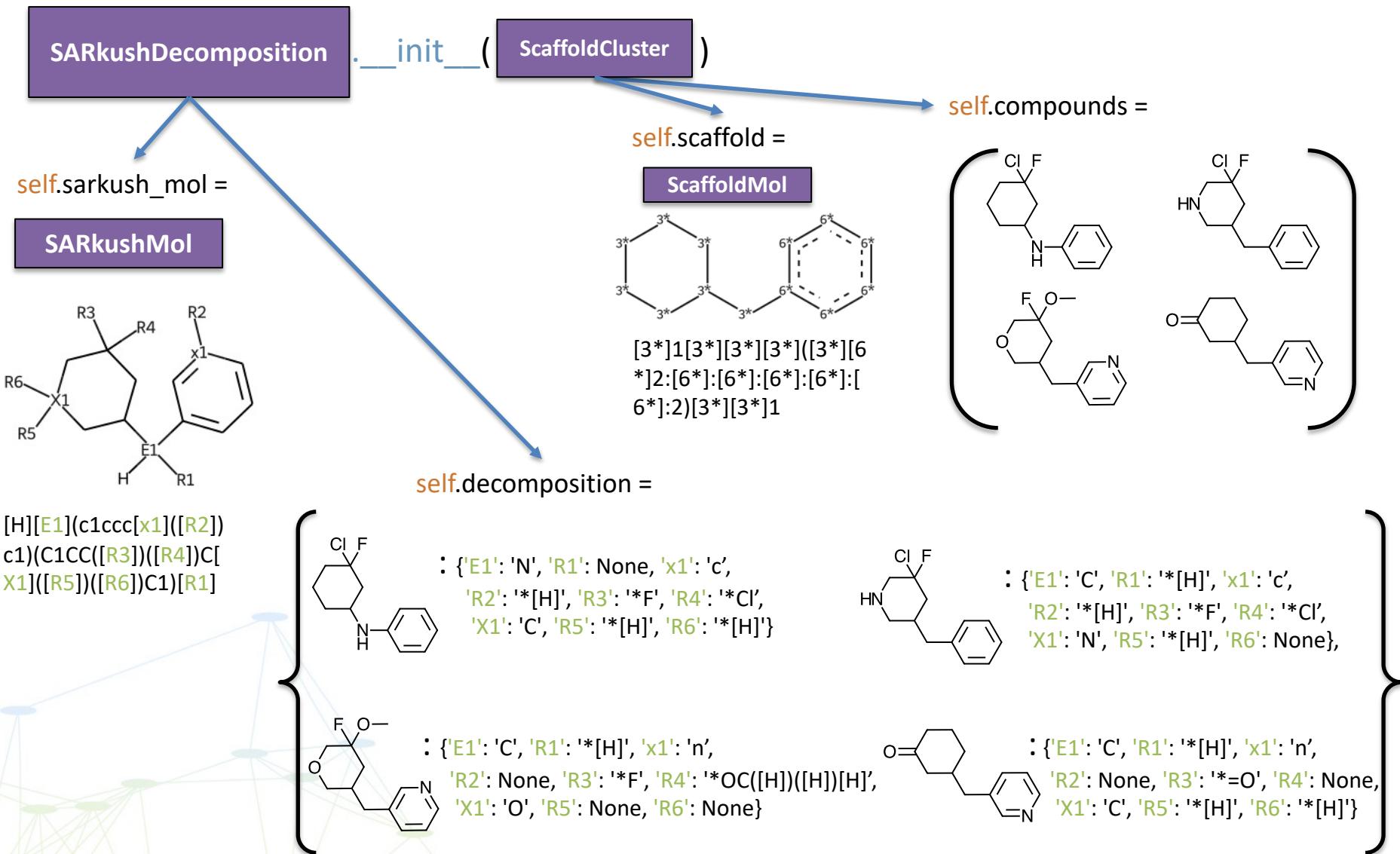
```
    self._add_sarkush_atom_to_decomposition_dict(sarkush_label, SARKushAtom )
```

```
    self._add_sidechains_to_sarkush_mol(sarkush_atom_id, SARKushAtom )
```

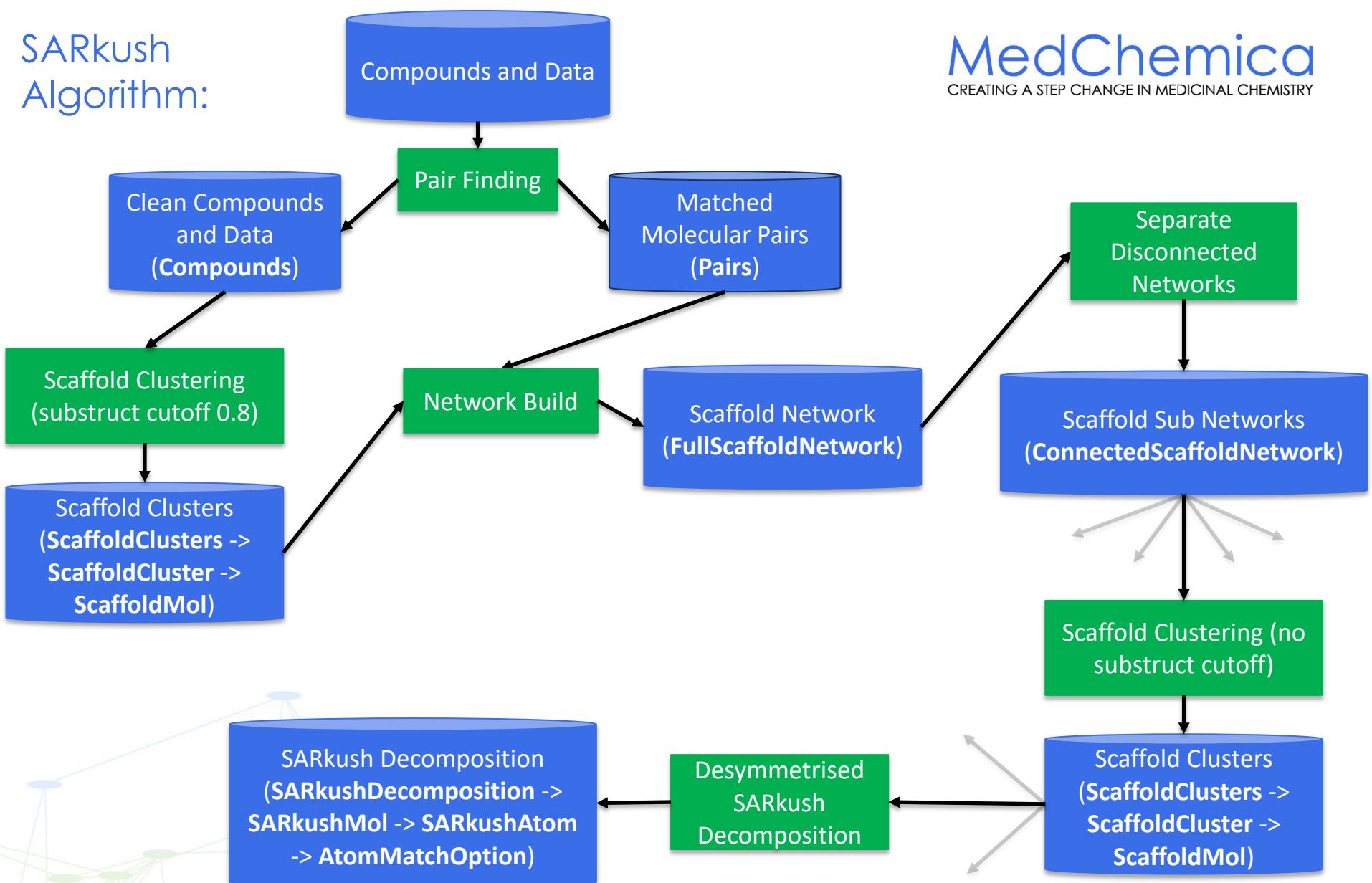
Classes used in SARkush Decomposition



Example



SARkush Algorithm:



Thank you!

Cheminformatics Expectations:

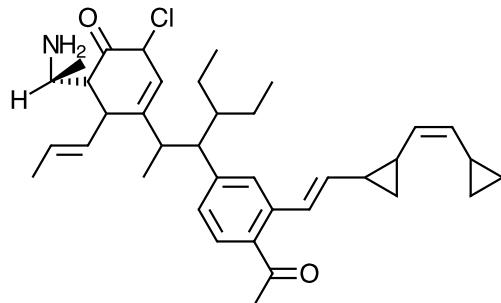


Cheminformatics Reality:



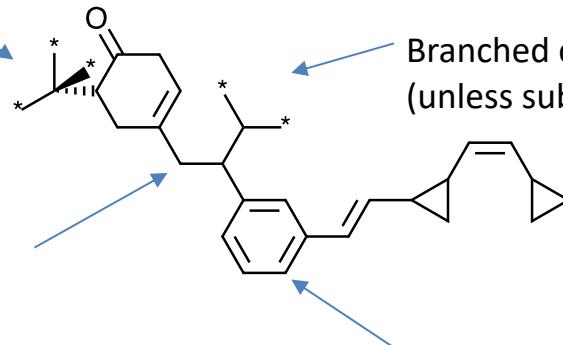
Complicated Test Molecule

ScaffoldMol



`self._cut_off_ring_and_branch_substituents()`

Chiral centres are retained using dummy atoms



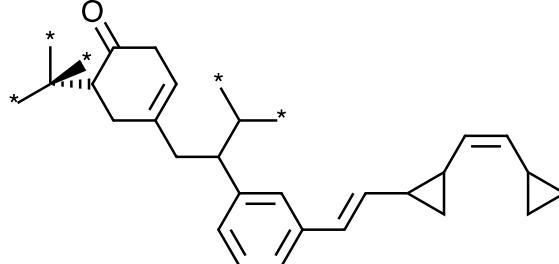
Branched centres are retained using dummy atoms (unless substituents are terminal atoms)

If a branched centre leads to a terminal atom, the terminal atom is removed

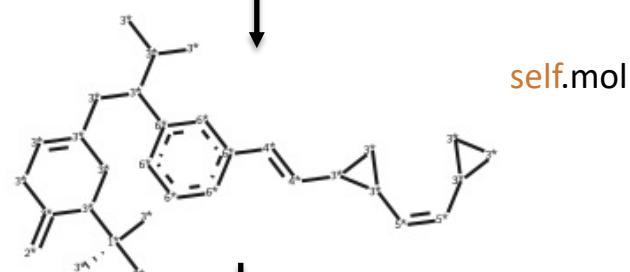
Ring substituents are removed

Complicated Test Molecule

ScaffoldMol



`self._set_atoms_to_generic()`



`self.mol`

Isotope 2 = carbonyl or thiocarbonyl heteroatom

$[2^*]=[3^*]1[3^*][3^*]=[3^*]([3^*][3^*]([3^*]([3^*])[3^*])[6^*]2:[6^*]:[6^*]:[6^*]:[6^*]([/4^*]=[4^*]/$
 $[3^*]3[3^*][3^*]3/[5^*]=[5^*]\backslash[3^*]3[3^*][3^*]3:[6^*]:2)[3^*][3^*]1[1^*]([3^*])([3^*])[3^*]$

`self.get_scaffold_smiles()`

`self.smiles`

Isotope 4 = atom belonging to E double bond

Isotope 6 = aromatic atom

Isotope 3 = aliphatic racemic atom

Isotope 1 = aliphatic chiral atom

- the "@" is missing from the SMILES because the asymmetry has been removed, meaning rdkit no longer recognises the chirality
- using an isotope label to encode the presence of chirality allows us to separate these scaffolds from the racemic ones

Isotope 5 = atom belonging to Z double bond

AtomMatchOption

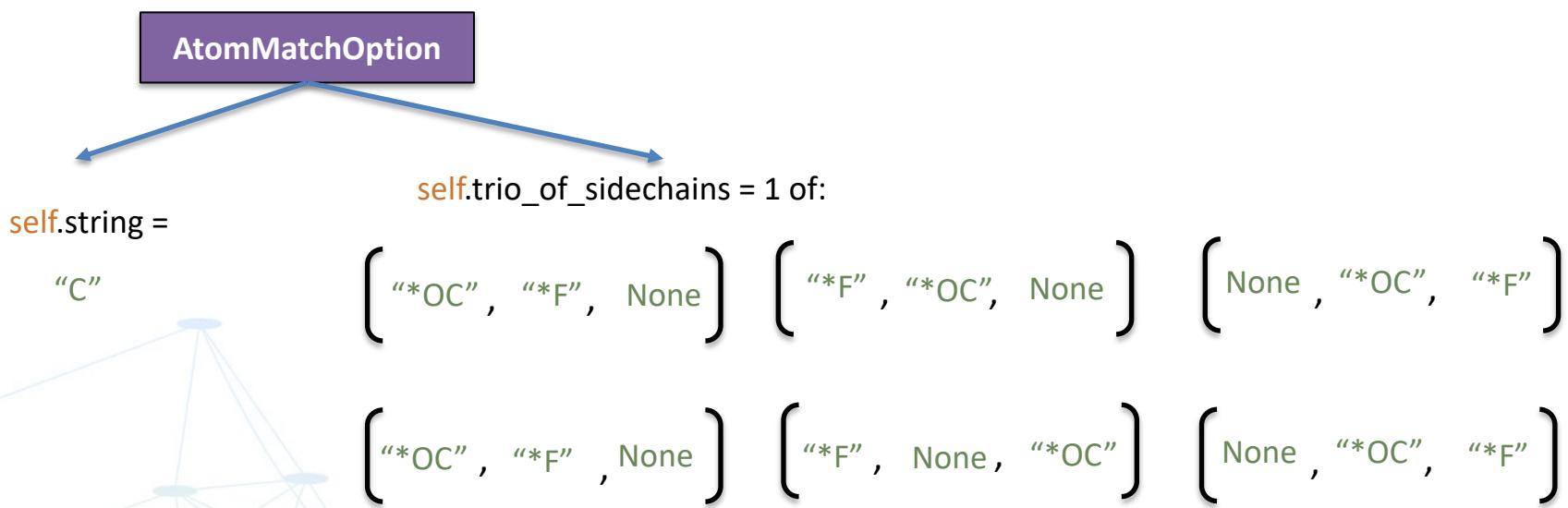
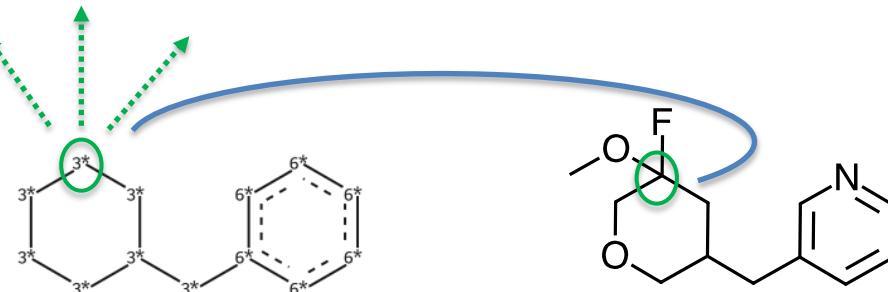
```
self._carbon = ["C", "[13C]", "c"]  
self._halogens = ["Cl", "F", "Br", "I"]  
self._heteroatoms = ["O", "S", "N", "P", "Si", "Se", "o", "s", "n", "p",  
                     "si", "se", "[n+]", "[S+]", "[N+]", "[s+]", "B", "b"]  
self._hydrogen = "[H]"
```

```
self._exact_atom_match_score = 1.0  
self._same_atom_different_impl_groups = 0.5  
self._no_match_score = 0.0  
self._both_heteroatoms_score = 0.5  
self._both_halogen_score = 0.5  
self._one_heteroatom_one_halogen = 0.33
```

self.atom_similarity(AtomMatchOption)

- **Compares strings and scores based on above definitions**

Desymmetrisation also important for sidechains



SARKushDecomposition

`self.atoms =`

`{ SARKushAtom 1 , SARKushAtom 2 , }`

`self.sarkush_mol =`

`SARKushMol`

`self._add_sidechains_to_sarkush_mol(sarkush_atom_id, SARKushAtom)`

- **Iterate over sidechain ids:**

`if SARKushAtom .sidechain_i_is_constant(sidechain_id):`

`constant_sidechain = SARKushAtom .return_sidechain_i(0, sidechain_id)`

`self. SARKushMol .add_sidechain(sarkush_atom_id, constant_sidechain)`

`else:`

`sarkush_label = self. SARKushMol .add_R_group(sarkush_atom_id)`

`self._add_R_group_to_decomposition_dict(sarkush_label, SARKushAtom , sidechain_position=sidechain_id)`