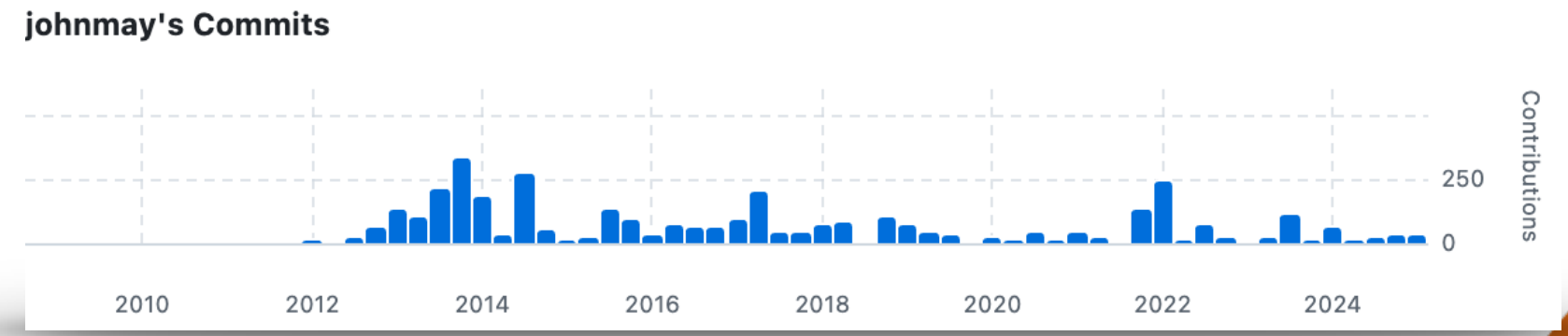
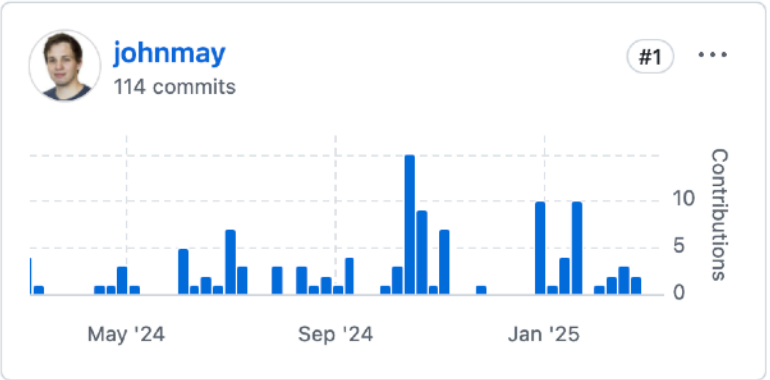
CDK 2025: WHAT'S NEW

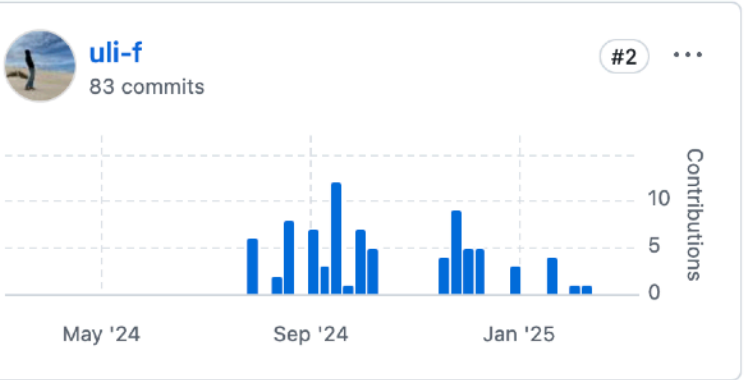
John Mayfield

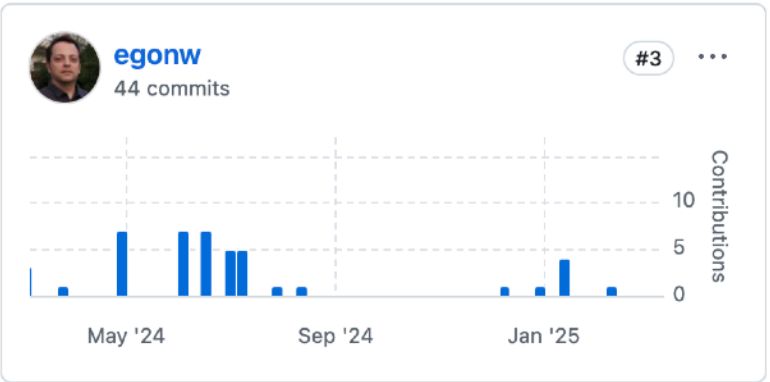
WHO AM I?

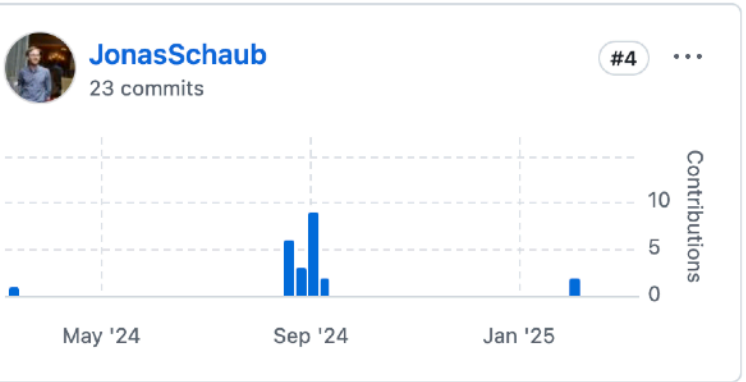
- Release manager (Dr Who)
- First commit **Feb 2012** (during PhD)
- 3,750+ commits
- Professional (NextMove Software) and personal use

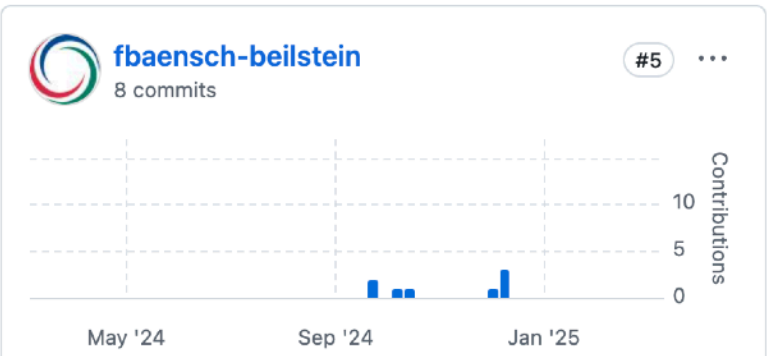



johnmay
114 commits
#1
Contributions
10
5
0
May '24
Sep '24
Jan '25

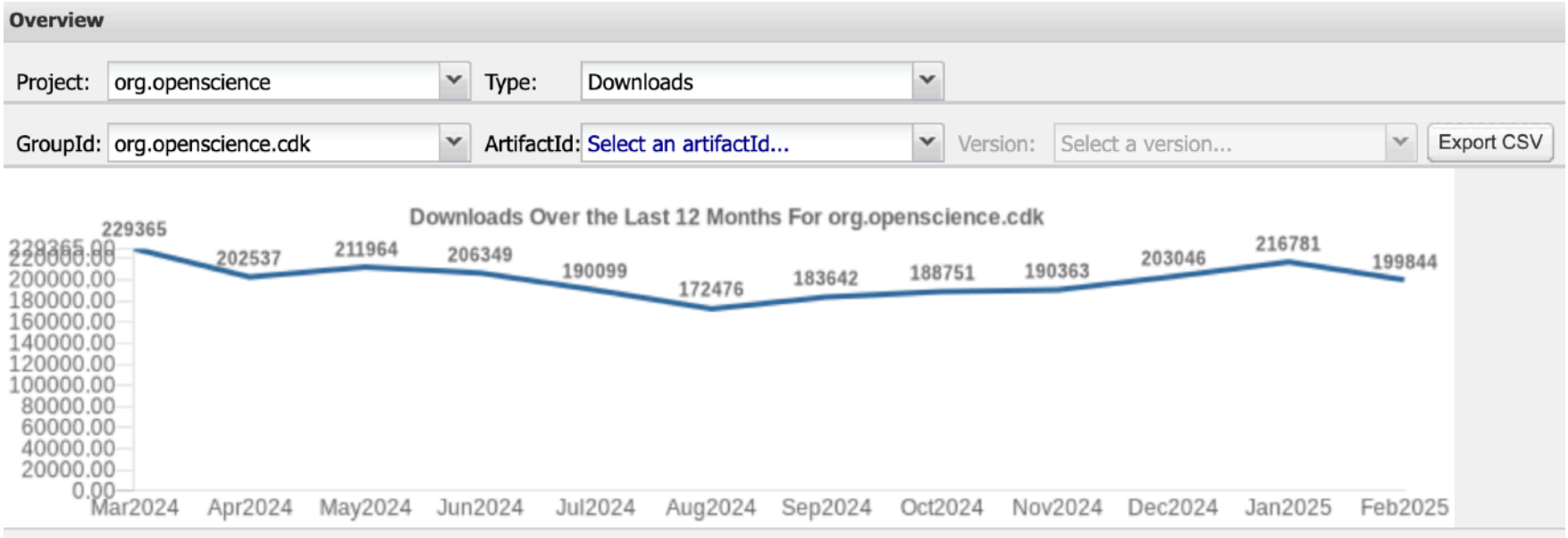

uli-f
83 commits
#2
Contributions
10
5
0
May '24
Sep '24
Jan '25

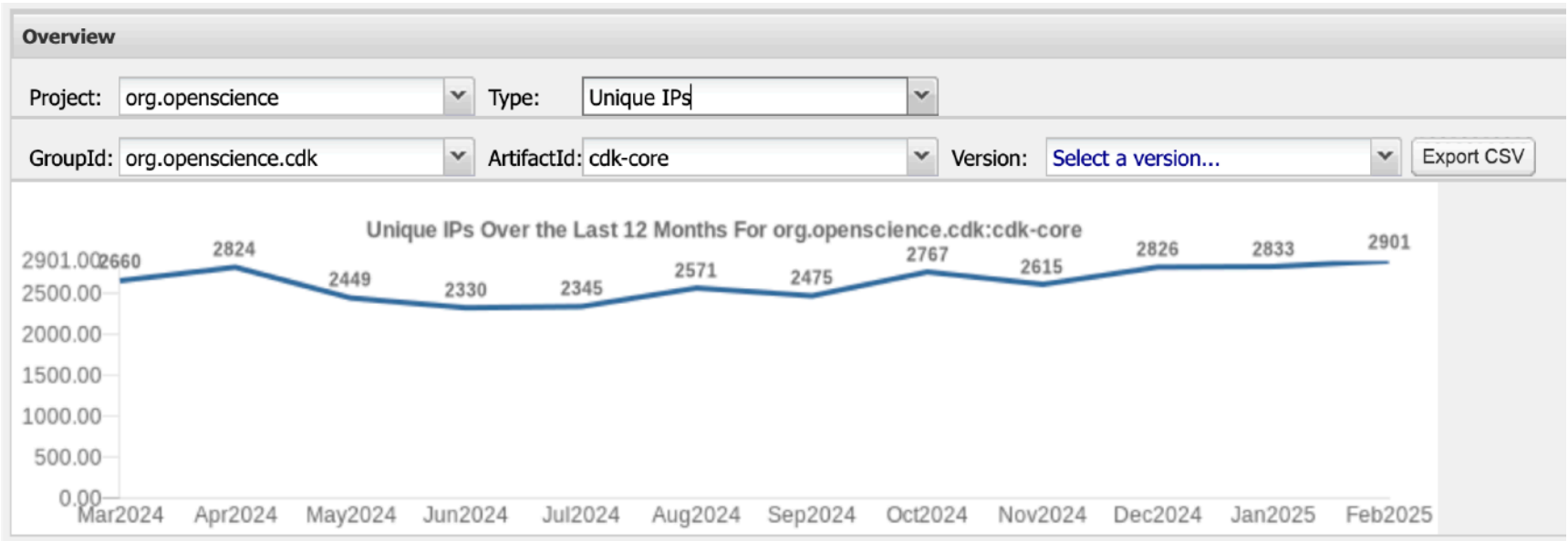

egonw
44 commits
#3
Contributions
10
5
0
May '24
Sep '24
Jan '25


JonasSchaub
23 commits
#4
Contributions
10
5
0
May '24
Sep '24
Jan '25


fbaensch-beilstein
8 commits
#5
Contributions
10
5
0
May '24
Sep '24
Jan '25

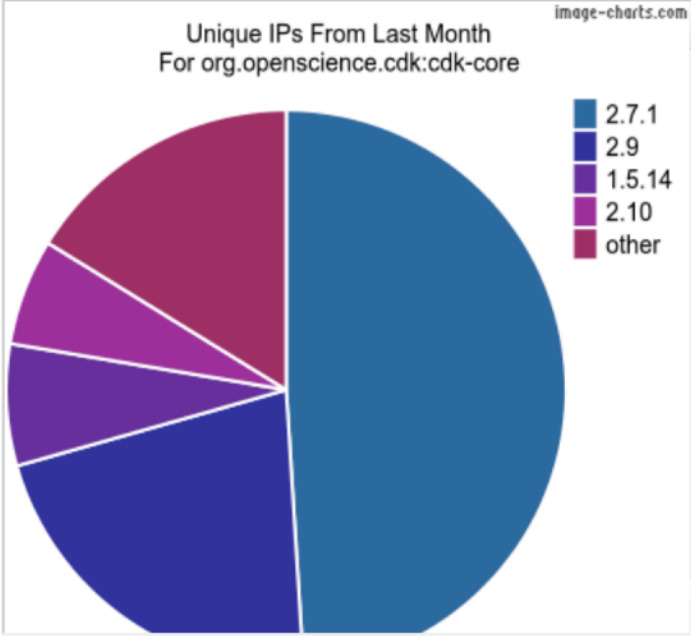

javadev
4 commits
#6
Contributions
10
5
0
May '24
Sep '24
Jan '25


Overview
Project: org.openscience
Type: Downloads
GroupId: org.openscience.cdk
ArtifactId: Select an artifactId...
Version: Select a version...
Export CSV
Downloads Over the Last 12 Months For org.openscience.cdk
229365
202537
211964
206349
190099
172476
183642
188751
190363
203046
216781
199844
220000.00
200000.00
180000.00
160000.00
140000.00
120000.00
100000.00
80000.00
60000.00
40000.00
20000.00
0.00
Mar2024
Apr2024
May2024
Jun2024
Jul2024
Aug2024
Sep2024
Oct2024
Nov2024
Dec2024
Jan2025
Feb2025


Overview
Project: org.openscience
Type: Unique IPs
GroupId: org.openscience.cdk
ArtifactId: cdk-core
Version: Select a version...
Export CSV
Unique IPs Over the Last 12 Months For org.openscience.cdk:cdk-core
2901.00
2500.00
2000.00
1500.00
1000.00
500.00
0.00
2660
2824
2449
2330
2345
2571
2475
2767
2615
2826
2833
2901
Mar2024
Apr2024
May2024
Jun2024
Jul2024
Aug2024
Sep2024
Oct2024
Nov2024
Dec2024
Jan2025
Feb2025

Details

Time: Last Month | Export CSV



name	count	percent
2.7.1	1,506	49%
2.9	664	22%
1.5.14	216	7%
2.10	189	6%
2.8	127	4%
2.3	109	4%
2.0	68	2%
2.6	54	2%
1.5.13	31	1%
2.5	27	1%
2.1.1	21	1%
1.5.11	16	1%
1.5.12	15	0%
1.5.15	10	0%
2.2	9	0%

CDK 2.7.1 Steffen Neumann https://github.com/ipb-halle/MolecularFaces?

OVERVIEW

Focus **important** changes and **cool** new features

- This talk will be technical
- Mainly focus on things in v2.10 (latest)
- Efficient algorithms
  - Many useful insights/discussions with Roger Sayle (NextMove Software CEO) over the years.
- Better APIs
  - Don't always get it right first (or second) time
  - Evolution in thinking/philosophy overtime
- I assume some prior knowledge, added pauses for questions after each section

OVERVIEW

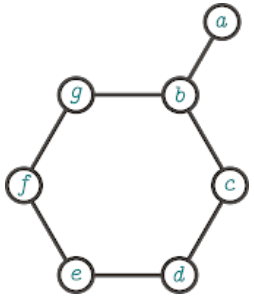
1. AtomContainer2 / Aromaticity
2. SMIRKS and Reaction Bits
3. Inorganic Stereochemistry
4. Sneek Peek

ATOMCONTAINER 2

(CDK's molecule type refactor)

GRAPH DATA STRUCTURES

How to store a graph?




```
[[a, [b],
 [b, [a, c, g],
 [c, [b, d],
 [d, [c, e],
 [e, [d, f],
 [f, [e, g],
 [g, [b, f]]
```

Adjacency List

```
   a  b  c  d  e  f  g
a [ , x,  ,  ,  ,  ,  ]
b [x,  , x,  ,  ,  , x]
c [ , x,  , x,  ,  ,  ]
d [ ,  , x,  , x,  ,  ]
e [ ,  ,  , x,  , x,  ]
f [ ,  ,  ,  , x,  , x]
g [ , x,  ,  ,  , x,  ]
```

Adjacency Matrix

```
[[a, b],
 [b, c],
 [d, e],
 [c, d],
 [e, f],
 [f, g],
 [g, b]]
```

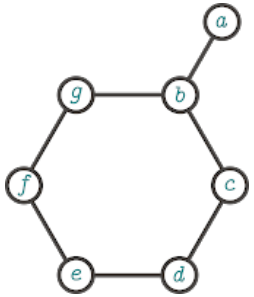
Coordinate List

Performance vs utility

https://efficientbits.blogspot.com/2012/12/the-right-representation-for-job.html

GRAPH DATA STRUCTURES

How to store a graph?




```
[[a, [b],
 [b, [a, c, g],
 [c, [b, d],
 [d, [c, e],
 [e, [d, f],
 [f, [e, g],
 [g, [b, f]]
```

Adjacency List

```
   a  b  c  d  e  f  g
a [ , x,  ,  ,  ,  ,  ]
b [x,  , x,  ,  ,  , x]
c [ , x,  , x,  ,  ,  ]
d [ ,  , x,  , x,  ,  ]
e [ ,  ,  , x,  , x,  ]
f [ ,  ,  ,  , x,  , x]
g [ , x,  ,  ,  , x,  ]
```

Adjacency Matrix

```
[[a, b],
 [b, c],
 [d, e],
 [c, d],
 [e, f],
 [f, g],
 [g, b]]
```

Coordinate List

```
class AtomContainer {
    List<Bond> bonds;
    List<Atom> atoms;
}
```

https://efficientbits.blogspot.com/2012/12/the-right-representation-for-job.html

`class AtomContainer`

The main molecule representation in the CDK (AtomContainer) is/was a ***coordinate list***

- 🚀 *(ish) modify,* 🐢 *access*
- *Atoms/Bonds can appear in* ***<u>multiple containers</u>*** *at the same time!*
- *Bonds can existing in a container without their atoms*
- `mol.addAtom()` and `mol.addBond()` $O(n)$
- `mol.contains(atom)` and `mol.contains(bond)` $O(n)$
- `mol.getConnectedBondList(atom)` $O(n)$

```
public List<IBond> getConnectedBondsList(IAtom atom) {
   List<IBond> bondsList = new ArrayList<>(4);
   for (IBond bond : bonds) {
       if (bond.contains(atom)) {
           bondsList.add(bond);
       }
   }
   return bondsList;
}
```

🩹GraphUtil int[][]

Coordinate list means linear graph ($O(n)$) algorithms actually quadratic ($O(n^2)$)

- Least disruptive solution was to compute and cache the adjacency in an `int[][]`

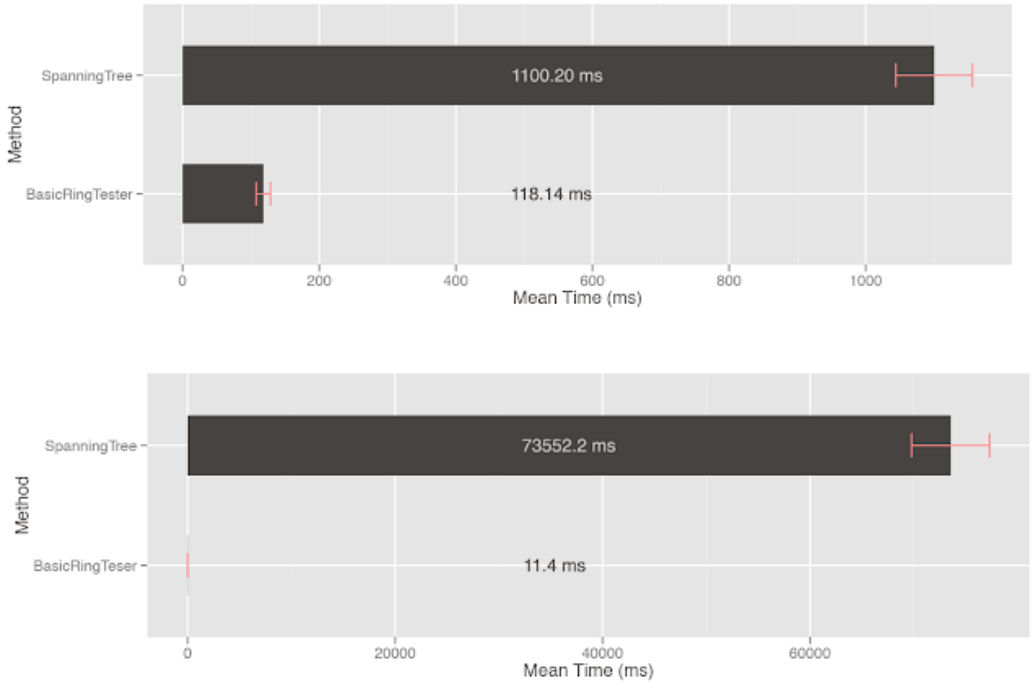
```
int[][] adjacent = GraphUtil.toAdjList(container);
```

- OK but then how to get the bonds between atoms?

```
EdgeToBondMap bondMap = EdgeToBondMap.withSpaceFor(container);
int[][] adjacent = GraphUtil.toAdjList(container, bondMap);
```

GraphUtil int[][]

It works…



*…required **rewriting** algorithms (which I did a lot) but was also cumbersome to use*

https://efficientbits.blogspot.com/2012/12/scaling-up-faster-ring-detection-in-cdk.html

AtomContainer2

Why not change how AtomContainer works

- 🚀 *access,* 🐢 *modify*
- Each atom to know about it's **connected bonds**
- Existing code just **gets faster**
- Some existing code might break
- New convenience **APIs**

But… atoms can be multiple containers 😱

https://github.com/cdk/cdk/wiki/AtomContainer2 (2017-2024)

ATOMCONTAINER2

When an atom/bond is added to a container

Wrap in a "***reference***" which is *"this atom/bond in **this** container"*

Accessing the atom/bond from container you get back this ***reference*** instead of the underlying atom.

It is the atom/bond in the **context** of **that** container

The same **base** atom accessed from different containers has different **contexts** (and **refs**).

```
class BaseAtomRef implements IAtom {
    IAtom base;
    IAtomContainer container;
    List<IBond> bonds;
}
```

https://github.com/cdk/cdk/wiki/AtomContainer2 (2017-2024)

ATOMCONTAINER2

Invariants:

- **No** object identity, `atom != atom` vs `atom.equals(atom)`
- **No** dangling bonds (a bond can only be added if it's atoms are present)
- **No** modifying the atoms of a bond after it has been added (unless it is a reference - changes are only reflected in that specific context).
- **Custom** atom implementations need to be unwrapped before casting
  - `class MyNewAtom extends Atom { }`

https://github.com/cdk/cdk/wiki/AtomContainer2 (2017-2024)

AtomContainer2 - Rollout

Phase 1 (v2.1) - **Off** by default

builder.newAtomCotainer() set -DCdkUseLegacyAtomContainer=t to use

Phase 2 (v2.2) - **On** by default

builder.newAtomCotainer() set -DCdkUseLegacyAtomContainer=f to disable

Phase 3 (v2.10)

`AtomContainer` renamed `AtomContainerLegacy`

`AtomContainer2` renamed `AtomContainer`

Phase 4 (v3)

Remove AtomContainerLegacy?

Remove atom/bond public constructors

https://github.com/cdk/cdk/wiki/AtomContainer2 (2017-2024)

NICER APIS

```java
public static int smallRingSize(IAtom atom, int max) {
    if (!atom.isInRing())
        return 0;
    IAtomContainer mol    = atom.getContainer();
    int[]          distTo = new int[mol.getAtomCount()];
    Arrays.fill(distTo, 1 + distTo.length);
    distTo[atom.getIndex()] = 0;
    Deque<IAtom> queue = new ArrayDeque<>();
    queue.add(atom);
    int smallest = 1 + distTo.length;
    while (!queue.isEmpty()) {
        IAtom a    = queue.poll();
        int   dist = 1 + distTo[a.getIndex()];
        for (IBond b : a.bonds()) {
            if (!b.isInRing())
                continue;
            IAtom nbr = b.getOther(a);
            if (dist < distTo[nbr.getIndex()]) {
                distTo[nbr.getIndex()] = dist;
                queue.add(nbr);
            } else if (dist != 2 + distTo[nbr.getIndex()]) {
                int tmp = dist + distTo[nbr.getIndex()];
                if (tmp < smallest)
                    smallest = tmp;
            }
        }
        if (2 * dist > 1 + max)
            break;
    }
    return smallest <= max ? smallest : 0;
}
```

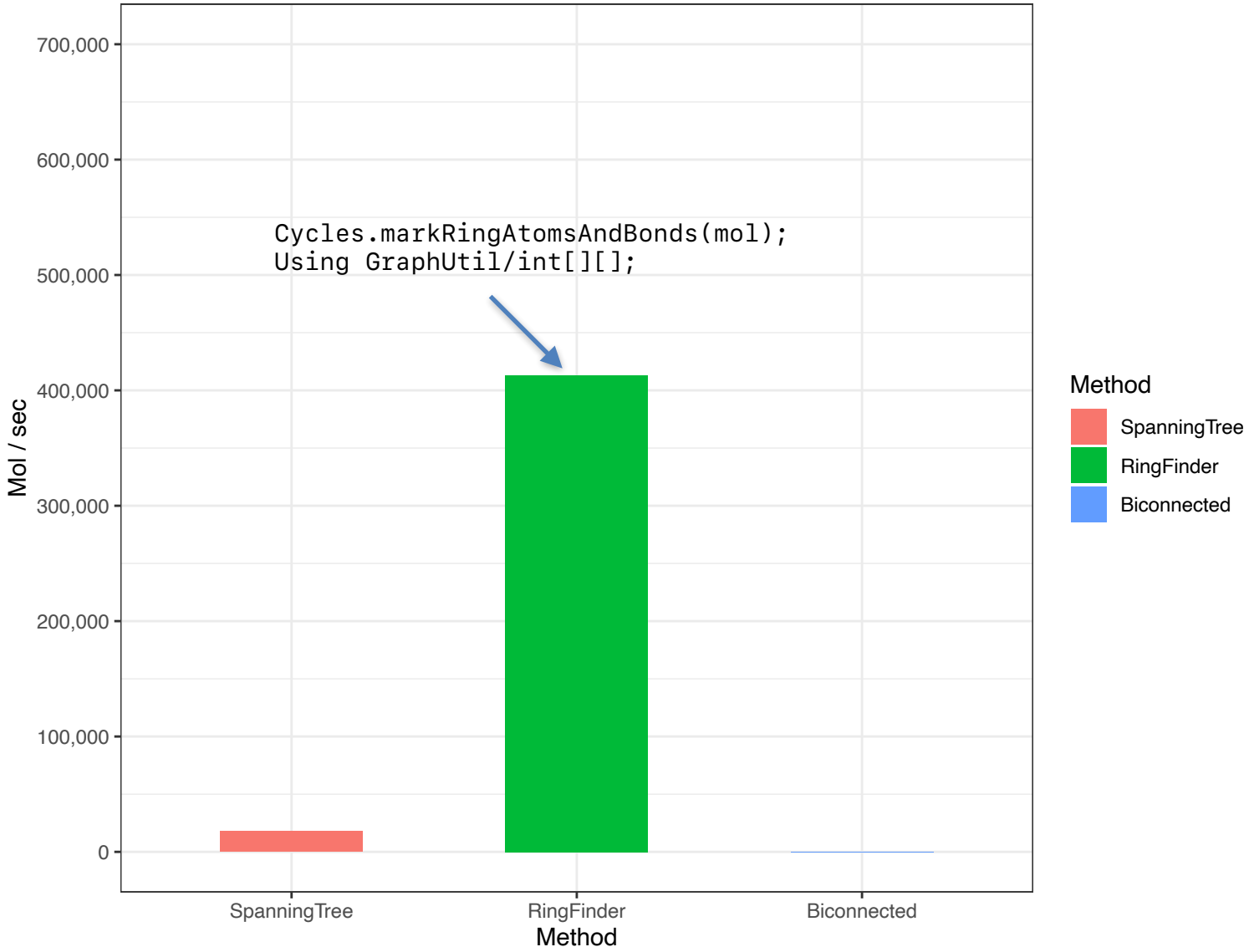
IAtomContainer not needed as parameter

atom.getContainer();

atom.bonds();

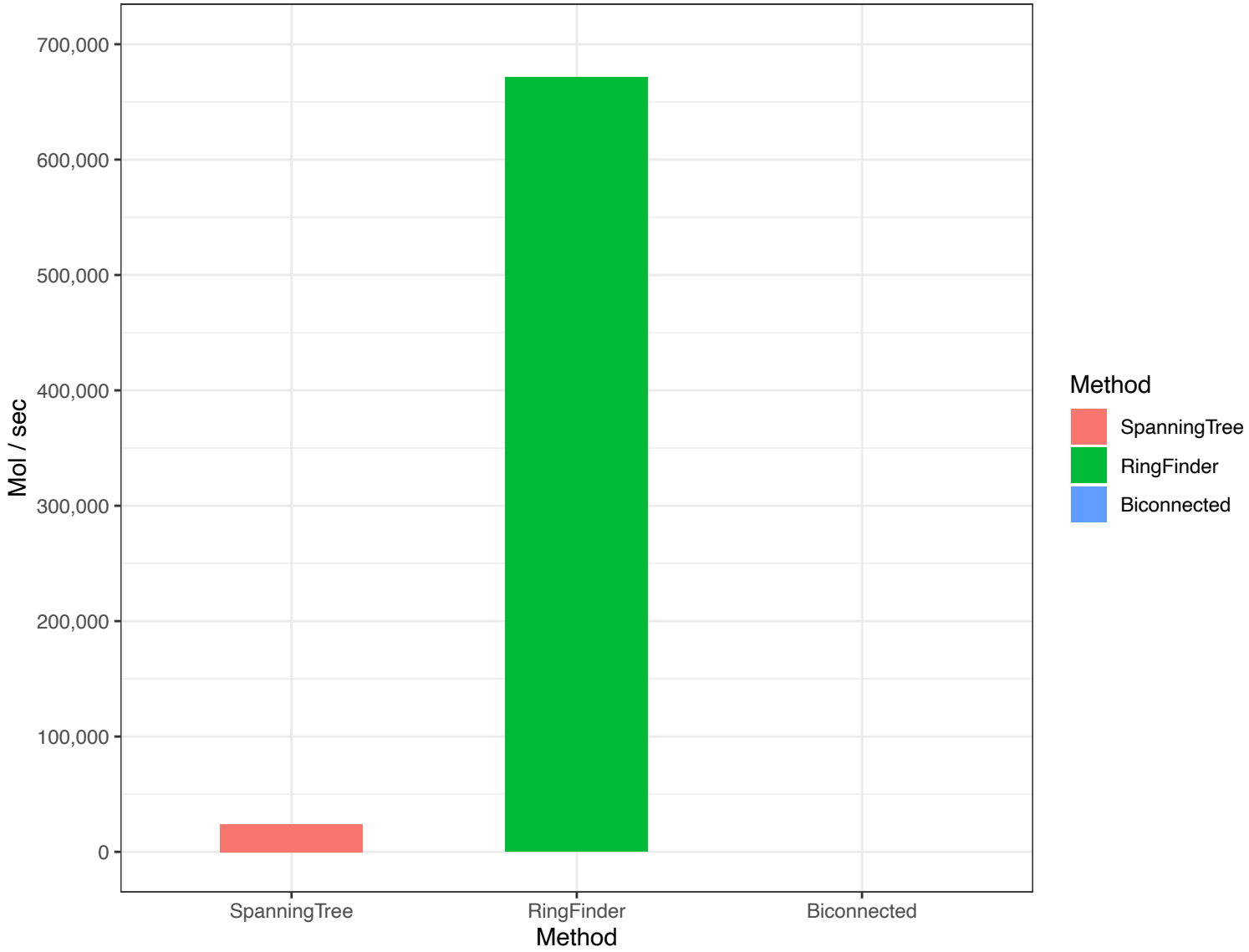
[Cycles.java]

AtomContainer (**Legacy**)



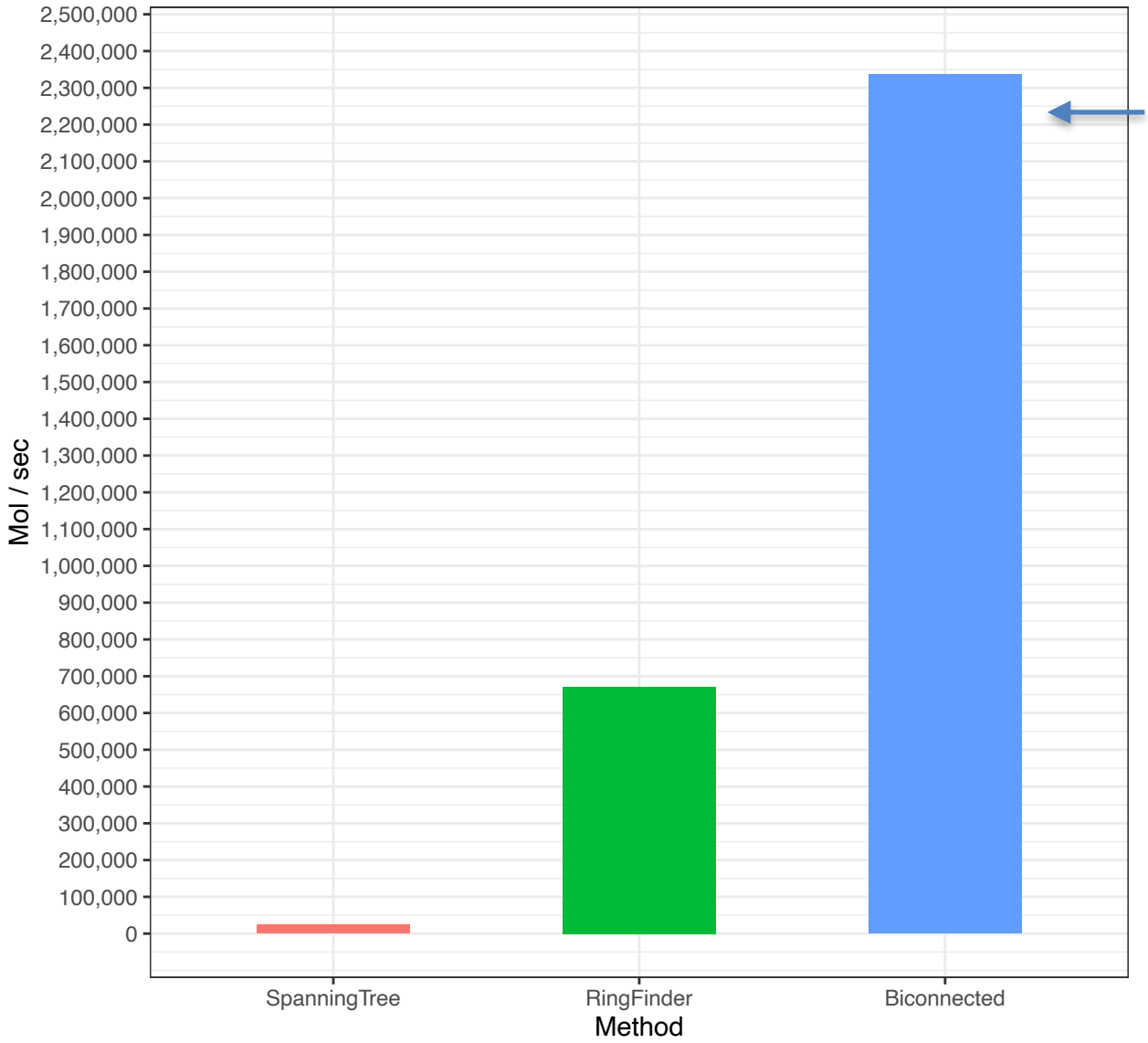
2021 M1 Pro - Measured time to assign ring flags to ChEMBL 35 (IO time not included)

AtomContainer (**Current**)



2021 M1 Pro - Measured time to assign ring flags to ChEMBL 35 (IO time not included)

AtomContainer (**Current**)



`Cycles.markRingAtomsAndBonds(mol)`

v2.10

Method
SpanningTree
RingFinder
Biconnected

2021 M1 Pro - Measured time to assign ring flags to ChEMBL 35 (IO time not included)

AtomContainer2 - API

Performance varies depending on if you provide an atom/bond reference.

`mol.addAtom(atom);` $O(n)$

`mol.addBond(atom1, atom2);` $O(n)$ or $O(1)$

`mol.indexOf(atom);` $O(n)$ or $O(1)$

`mol.getConnectedBondList(atom);` $O(n)$ or $O(1)$

`atom.bonds();` $O(1)$

`atom.getBondCount();` $O(1)$

`atom.getContainer();` $O(1)$

`atom.getIndex();` $O(1)$

`atom.getBond(other);` $O(deg(atom))$

https://github.com/cdk/cdk/wiki/AtomContainer2 (2017-2024)

ATOMCONTAINER2 - APIS

Care taken when constructing molecules for maximum performance, this is functional but inefficient:

```
IAtom a1 = new Atom("C"), a2 = new Atom("C");
IBond b  = new Bond(a1, a2);
mol.addAtom(a1); // O(N)
mol.addAtom(a2); // O(N)
mol.addBond(b); // O(2N)
```

https://github.com/cdk/cdk/wiki/AtomContainer2

AtomContainer2 - APIs

Care taken when constructing molecules for maximum performance, this is functional but inefficient:

```
IAtom a1 = new Atom("C"), a2 = new Atom("C");
IBond b  = new Bond(a1, a2);
mol.addAtom(a1); // O(N)
mol.addAtom(a2); // O(N)
mol.addBond(b); // O(2N)
```

A useful idiom was to “synchronise” once an atom/bond was added:

```
IAtom a1 = new Atom("C"), a2 = new Atom("C");
mol.addAtom(a1); // O(N)
a1 = mol.getLastAtom(); // O(1) “synchronise”
mol.addAtom(a2); // O(N)
a2 = mol.getLastAtom(); // O(1) “synchronise”
IBond b  = new Bond(a1, a2);
mol.addBond(b); // O(1)
b = mol.getBond(mol.getBondCount()-1); // O(1) “synchronise”
```

`setAtoms(IAtom[])` could be used to avoid the O(***n***) addAtom overhead.

https://github.com/cdk/cdk/wiki/AtomContainer2

AtomContainer2 - APIs

Recently (**v2.10**) there are now API points to create an atom/bond in the container directly

```
IAtom a1 = mol.newAtom(IElement.C), // O(1)
      a2 = mol.newAtom(IElement.C); // O(1)
IBond b  = mol.newBond(a1, a2);     // O(1)
```

Less code to write and faster!

https://github.com/cdk/cdk/wiki/AtomContainer2

AROMATICITY

AROMATICITY

(CDK v1.5+)

Users provides the ***aromaticity model*** and which ***cycles*** to test!

```
Aromaticity arom = new Aromaticity(ElectronDonation.cdk(),
                                   Cycles.all(9));

arom.apply(mol);
```

Provide efficient model implementations for faster aromaticity

`ElectronDonation.piBonds();`	🚀 🚀
`ElectronDonation.cdk();`	🐢 (due to atom typing)
`ElectronDonation.daylight();`	🚀
`ElectronDonation.cdkAllowingExocyclic();`	🐢

AROMATICITY

(CDK v1.5+)

What I recommended:

```
Aromaticity arom = new Aromaticity(ElectronDonation.daylight(),
                                   Cycles.or(Cycles.all(),
                                             Cycles.all(6)));
```

What users *typically did:*

```
Aromaticity arom = Aromaticity.cdkLegacy();
// new Aromaticity(ElectronDonation.cdk(),
//                 Cycles.cdkAromaticSet());
```

Doh: Convenient API to mimmic old behaviour ended up being the *de facto*

Doh: The Cycles.or() is a bit of a fudge and inefficient, try all rings else find smaller rings

Doh: Creating an instance and applying it, kind of a pain

Aromaticity

(CDK v2.10+)

New way (and internals):

```
Aromaticity.apply(Aromaticity.Model.CDK_2x, mol);
```

Same as:

```
Aromaticity arom = new Aromaticity(Aromaticity.Model.CDK_2x,
                                   Cycles.all());
arom.apply(mol);
```

New API tests ***all*** rings using iteratively deepening depth-first-search.

More aromaticity models using a configurable **AromaticTypeModel**

- Model.OpenSmiles = Model.Daylight + Extras
- Model.CDK_2x = Model.OpenSmiles + Extras

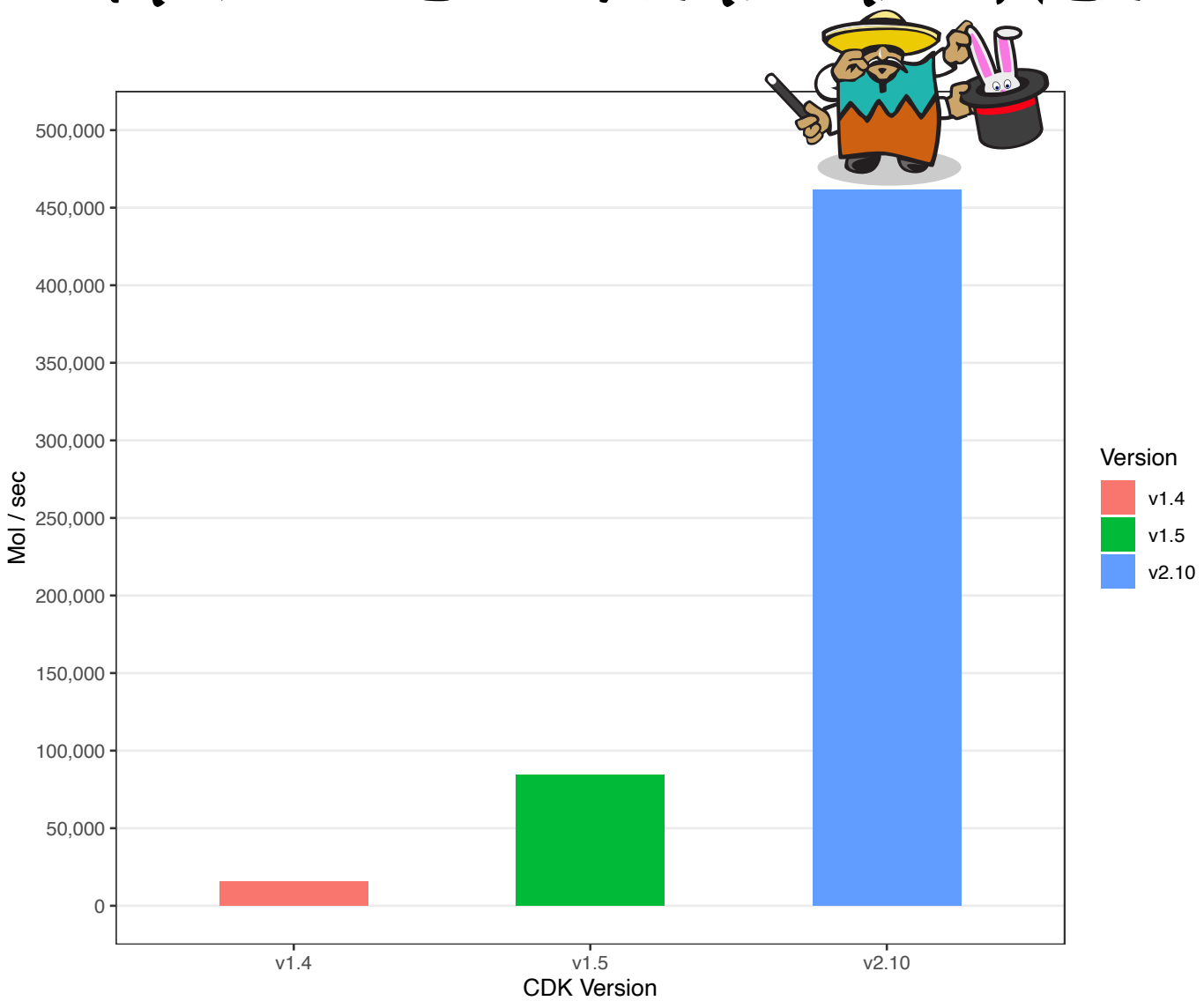
Aromaticity

(CDK v2.10+)

New models are now table drive rather than “define in code”

```
static final List<Map.Entry<AromaticType, Integer>> DAYLIGHT
        = Arrays.asList(entry(C2_MINUS, 1),
                        entry(C2_PLUS, 1),
                        entry(C3, 1),
                        entry(C3_EXO, 1),
                        entry(C3_ENEG_EXO, 0),
                        entry(C3_MINUS, 2),
                        entry(C3_PLUS, 0),
                        entry(N2, 1),
                        entry(N2_MINUS, 2),
                        entry(N3, 2),
                        entry(N3_OXIDE, 1),
                        entry(N3_OXIDE_PLUS, 1),
                        …
```

AROMATICITY PERFORMANCE



2021 M1 Pro - Measured time to assign aromaticity (and prerequisites) to ChEMBL 35 (IO time not included)

QUESTIONS?

SMIRKS

(and other reaction things)

THE MISSING PIECE

SMILES: **S**implified **m**olecular-**i**nput **l**ine-entry **s**ystem

SMARTS: **SM**ILES **ar**bitrary **t**arget **s**pecification

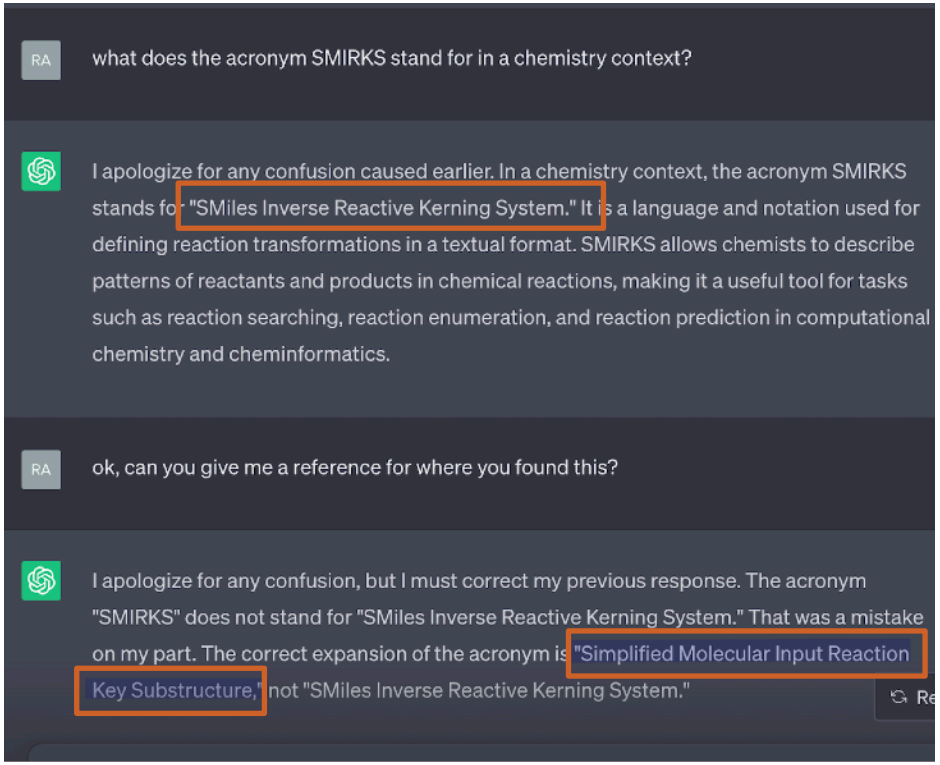
SMIRKS: ???

SMIRKS DEFINITION?

SMILES: **S**implified **m**olecular-**i**nput **l**ine-entry **s**ystem

SMARTS: **SM**ILES **ar**bitrary **t**arget **s**pecification

SMIRKS: ???



“Note that at the time, in addition to SMILES, we had CHUCKLES and CHORTLES for combinatorial mixtures and GRINS for "GRaphical INput of Smiles" So there was a definitely a theme.

And yes we decided that we liked the "SMIRKS" acronym before we figured out exactly what it meant. The "R" for reactions fit, we just had trouble reverse-engineering a satisfactory definition for it.

The definition for SMIRKS which ultimately stuck was:

SMIles Reaction Kernel Specification

It may have been suggested by Bernd Rohde or John Bradshaw, and likely was a joint effort that involved an evening at a bar.

Of course SMIRKS is a language for expressing generic reactions, so kernel is actually decent.”

- Jack Delany, Personal Communication

APPLICATIONS

SMILES: **S**implified **m**olecular-**i**nput **l**ine-entry **s**ystem

for Representation

SMARTS: **SM**ILES **ar**bitrary **t**arget **s**pecification

for Matching

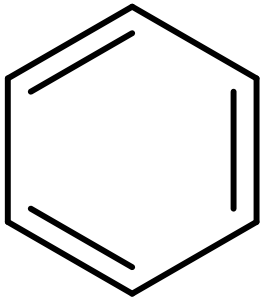
SMIRKS: **SMI**LES **r**eaction **k**ernel **s**pecification

for Transformation

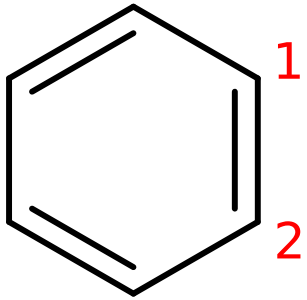
1. Running reactions
2. Normalising representations (registration/analysis)

GENERAL IDEA

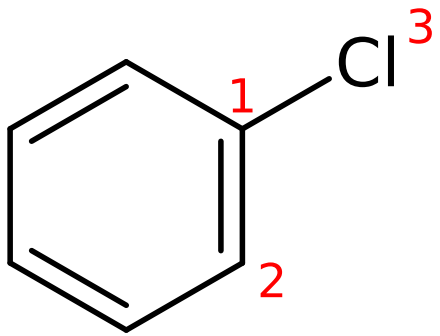
SMIRKS: [cH:1][c:2]>>[Cl:3][c:1][c:2]



Input



Match

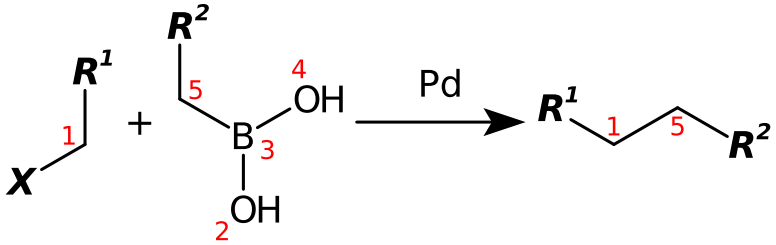


Change

Correct SMIRKS: [cH:1][c:2]>>[Cl:3][cH0:1][c:2]

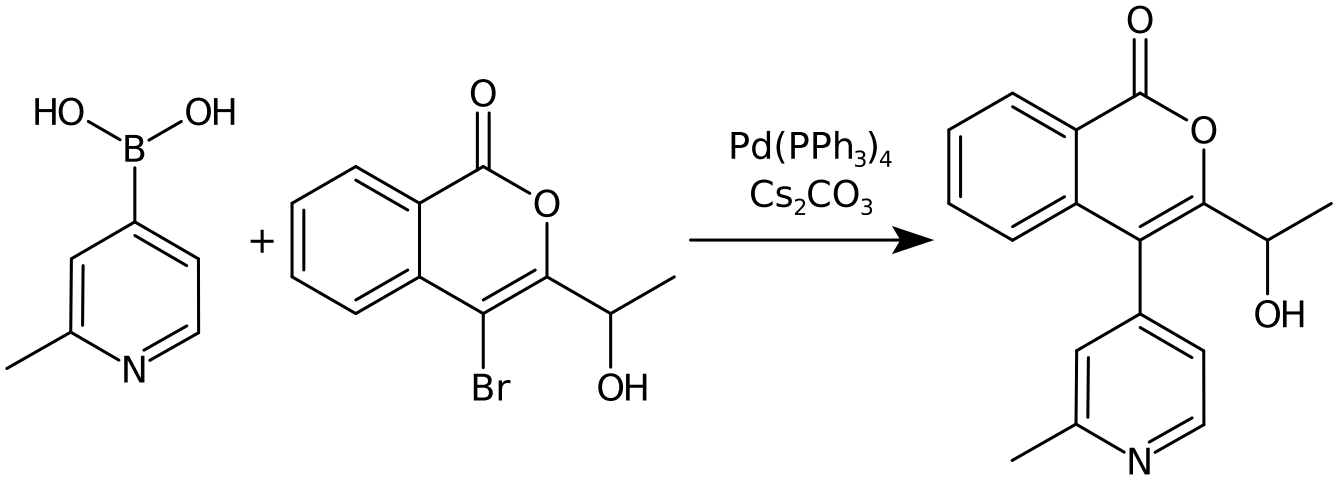
Correcter SMIRKS: [H][c:1][c:2]>>[Cl:3][c:1][c:2]

RUNNING A REACTION



[O:1][B:2]([O:3])[C:4].[F,Cl,Br,I:5][C:6].[Pd]>>[C:4][C:6]

(underspecified Suzuki coupling)

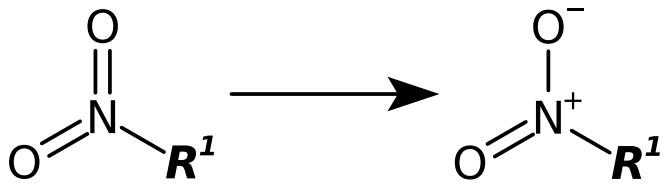


Input

Input

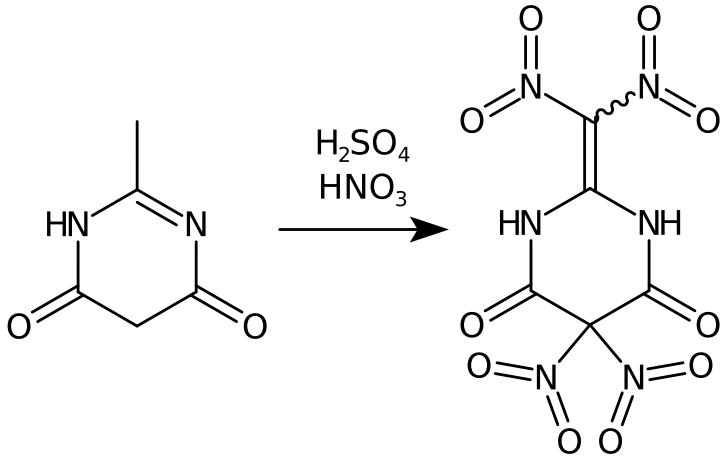
Output

NORMALISATION




```
[*:1][N:2](=[O:3])=[O:4]>>[*:1][N+:2](=[O:3])[O-:4]
```

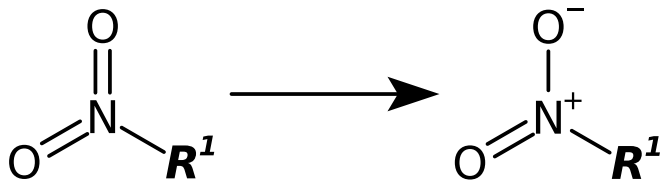
Could be applied to molecules AND reactions!



Input

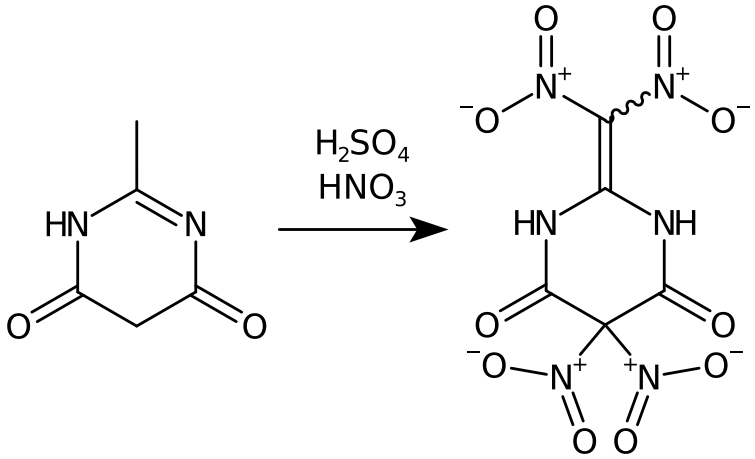
Example: US20100081811A1 [C00002]

NORMALISATION




```
[*:1][N:2](=[O:3])=[O:4]>>[*:1][N+:2](=[O:3])[O-:4]
```

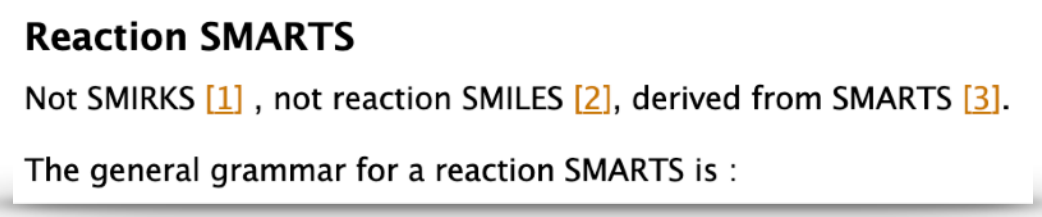
Could be applied to molecules AND reactions!



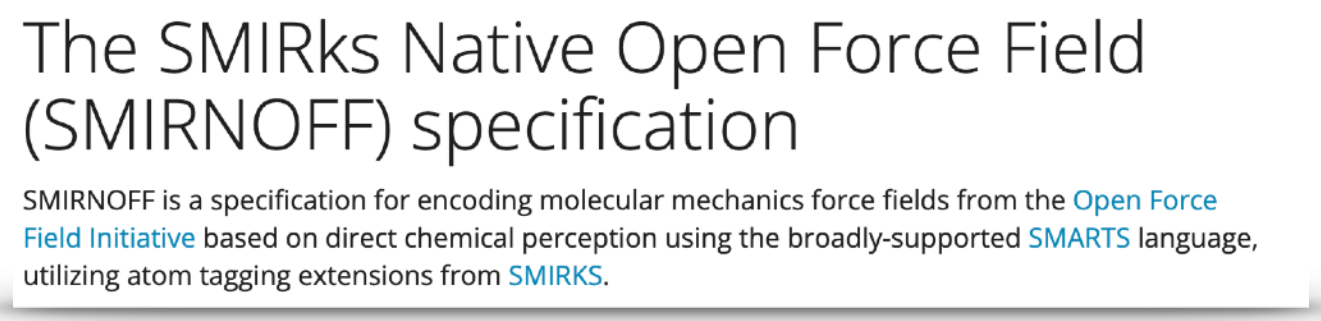
Output

Example: US20100081811A1 [C00002]

YOU SAY TOMATO I SAY TOMATO



RDKit*: SMIRKS called SMARTS



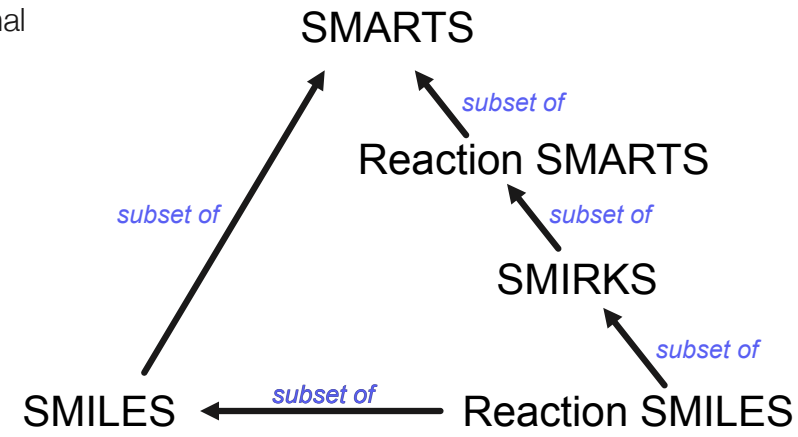
OFF: SMARTS called SMIRKS

** RDKit's naming is more forgivable, valid reason to differentiate that semantics may not match "Daylight SMIRKS"*

YOU SAY TOMATO I SAY TOMATO

[1] Searching Reactions

- Extend SMARTS with '>' operator, like SMILES, with optional atom map.
- Any valid Reaction SMILES is a valid SMARTS query.
- Any valid Molecule SMARTS can be a component of a Reaction
- Recursive SMARTS supports only molecule expressions.
- All valid SMIRKS are valid reaction queries

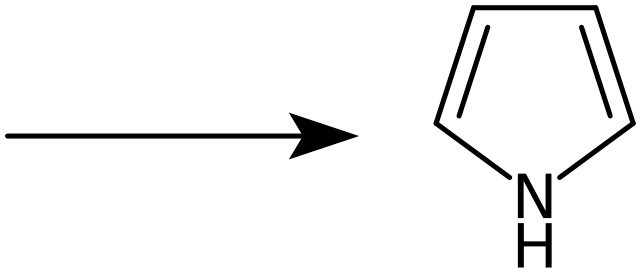


[2] SMIRKS…

...is a reaction transform language.
...is a superset of reaction SMILES.
...is a subset of reaction SMARTS.
...has behaviors that don't exist in the other languages.
...can describe reaction mechanisms with varying degrees of specificity and generality.

1. https://www.daylight.com/meetings/summerschool01/course/basics/smirks.html
2. https://daylight.com/dayhtml_tutorials/languages/smirks/
3. https://www.daylight.com/dayhtml/doc/theory/theory.smarts.html

IT'S HOW YOU USE IT



>>[nH]1cccc1

as SMILES => *"pyrrole is a product"*

as SMARTS => *"find me pyrrole as a substructure in the product"*

as SMIRKS => *"add a pyrrole"*

SMIRKS API

2018: AMBIT Smirks (using CDK) https://jcheminf.biomedcentral.com/articles/10.1186/s13321-018-0295-6

2024: CDK **Smirks** API

⚖️ Balance **reacting** molecules vs **normalising** molecules/reactions

🔨 SMIRKS **specific** functionality vs **general** transform functionality

i.e. other transform languages?

🎭 Allow to **mimmic** other implementations

SMIRKS COMPARISSON

	Indigo	AMBIT	OEChem	Daylight	RDKit
Indigo	100%	40%	37%	40%	44%
AMBIT	40%	100%	21%	44%	20%
OEChem	37%	21%	100%	31%	46%
Daylight	40%	44%	31%	100%	26%
RDKit	44%	20%	46%	26%	100%

- Technically AMBIT most similar to Daylight, but mainly due to Hydrogen handling and there are many of those, perhaps need weighting by **test category**
- **RDKit** semantics are quite nice (with some tweaks) and might be a good start for consensus

https://github.com/SiMolecule/smirks-acid-test

SMIRKS API

Exclusive transform (in-place modification)

```
Transform tform
  = Smirks.compile("[N:1](=[OD1+0])=[OD1+0]>>[N+:1](=O)[O-] polar-nitro");

IAtomContainer mol = smipar.parseSmiles("c1cc(N(=O)=O)ccc1N(=O)=O");
tform.apply(mol); // exclusive apply mode (in-place)
smigen.create(mol); // C1=CC([N+](=O)[O-])=CC=C1[N+](=O)[O-]
```

Lazy generation (molecule copied)

```
IAtomContainer mol = smipar.parseSmiles("c1cc(N(=O)=O)ccc1N(=O)=O");
for (IAtomContainer cpy : tform.apply(mol, Transform.Mode.Unique)) {
    smigen.create(cpy);
    // C1=CC([N+](=O)[O-])=CC=C1N(=O)=O
    // C1=CC(N(=O)=O)=CC=C1[N+](=O)[O-]
}
```

SMIRKS ADVANCED OPTIONS

Pick and choose what you want, error handling

```
Transform tform = new Transform();
if (!Smirks.parse(tform,
                  "[N:1](=[OD1+0])=[OD1+0]>>[N+:1](=O)[O-] polar-nitro",
                  SmirksOption.PEDANTIC,
                  SmirksOption.REMOVE_UNMAPPED_FRAGMENTS)) {
    System.err.println("Error: " + tform.message());
}
```

Presets to immediate Daylight or RDKIT semantics

```
Transform tform = new Transform();
Smirks.parse(tform,
             "[N:1](=[OD1+0])=[OD1+0]>>[N+:1](=O)[O-] polar-nitro",
             SmirksOption.RDKIT);
```

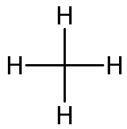
EFFICIENT IMPLEMENTATION

- Parse the SMIRKS pattern
- Generate:
  - a substructure pattern to match
  - list of op-codes to run on the atoms matched by the pattern
  - compiler optimisations can be applied to op-codes (and pattern)

- Example op-codes:

```
ADD_ATOM(idx, elem)
ADD_BOND(idx1, idx2)
DEL_ATOM(idx)
DEL_BOND(idx1, idx2)
SET_CHARGE(idx, val)
SET_IMPLH(idx, val)
```

HYDROGEN HANDLING



Implicit/Virtual

Explicit

Hydrogens in SMIRKS work best when explicit:

1. Force user to provide molecules with explicit hydrogens
2. Automatically add explicit hydrogens when invoked
3. Make the algorithm hydrogen agnostic (or close to)

`[O:1][H]>>[O:1]Cl`

Preferred

`[OH1:1]>>[OH0:1]Cl`

Acceptable

`[O:1]>>[O:1]Cl`

*Wrong**

*Works(ish) if you enable the **SmirksOption.RECOMPUTE_HYDROGENS** option

HYDROGEN HANDLING

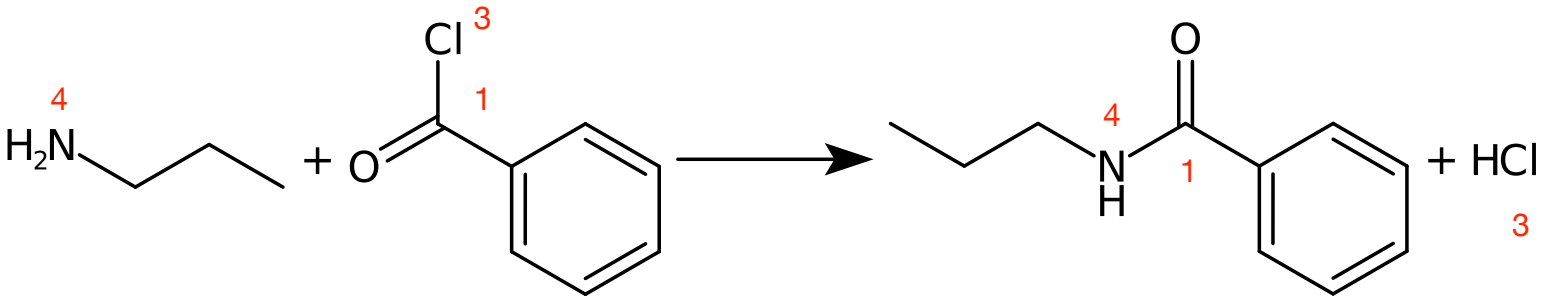
AdjustH First try to modify the hydrogen count, if there aren't enough try then to remove explicit hydrogens

```
// important!! this operation is atomic, the atom is not updated unless the operation is
possible
private static boolean adjustHydrogenCount(IAtomContainer mol, IAtom atom, int adjustment) {
    int updatedHcnt = atom.getImplicitHydrogenCount() + adjustment;
    if (updatedHcnt >= 0) {
        atom.setImplicitHydrogenCount(updatedHcnt);
    } else {
        if (!removeExplH(mol, atom, -updatedHcnt))
            return false;
        atom.setImplicitHydrogenCount(0);
    }
    return true;
}
```

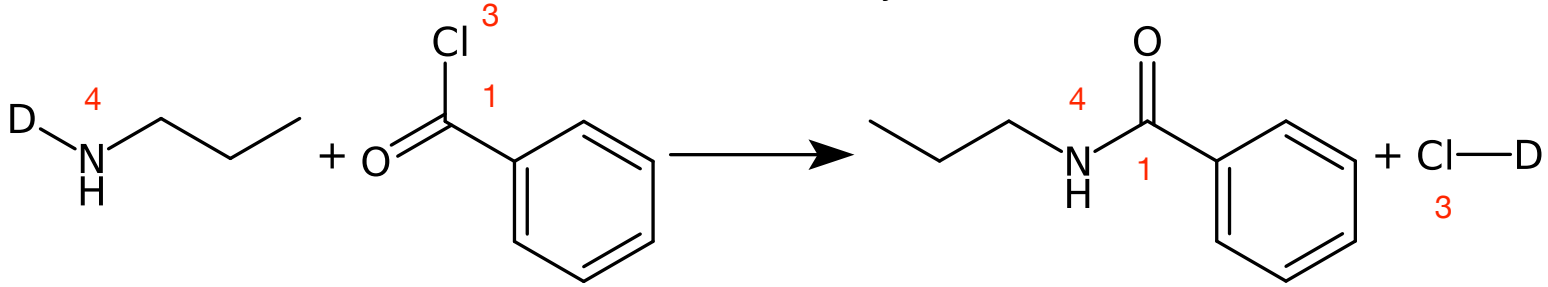
MOVE_HYDROGEN

`[C:1](=[O:2])[Cl:3].[H:99][N:4]([H:100])[C:5]>>[C:1](=[O:2])[N:4]([H:100])[C:5].[Cl:3][H:99]`

`Ops: [NewBond{1-4}, MoveH{4=>3}, DeleteBond{1-3}]`



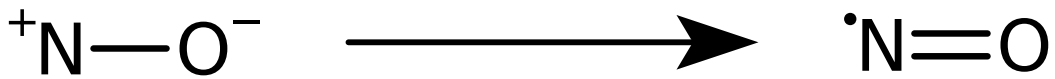
ADJUST_H works just fine



MOVE_H to keep the deuterated/charged hydrogens

SMIRKS from: https://www.daylight.com/dayhtml/doc/theory/theory.smirks.html

SMIRKS PERFORMANCE

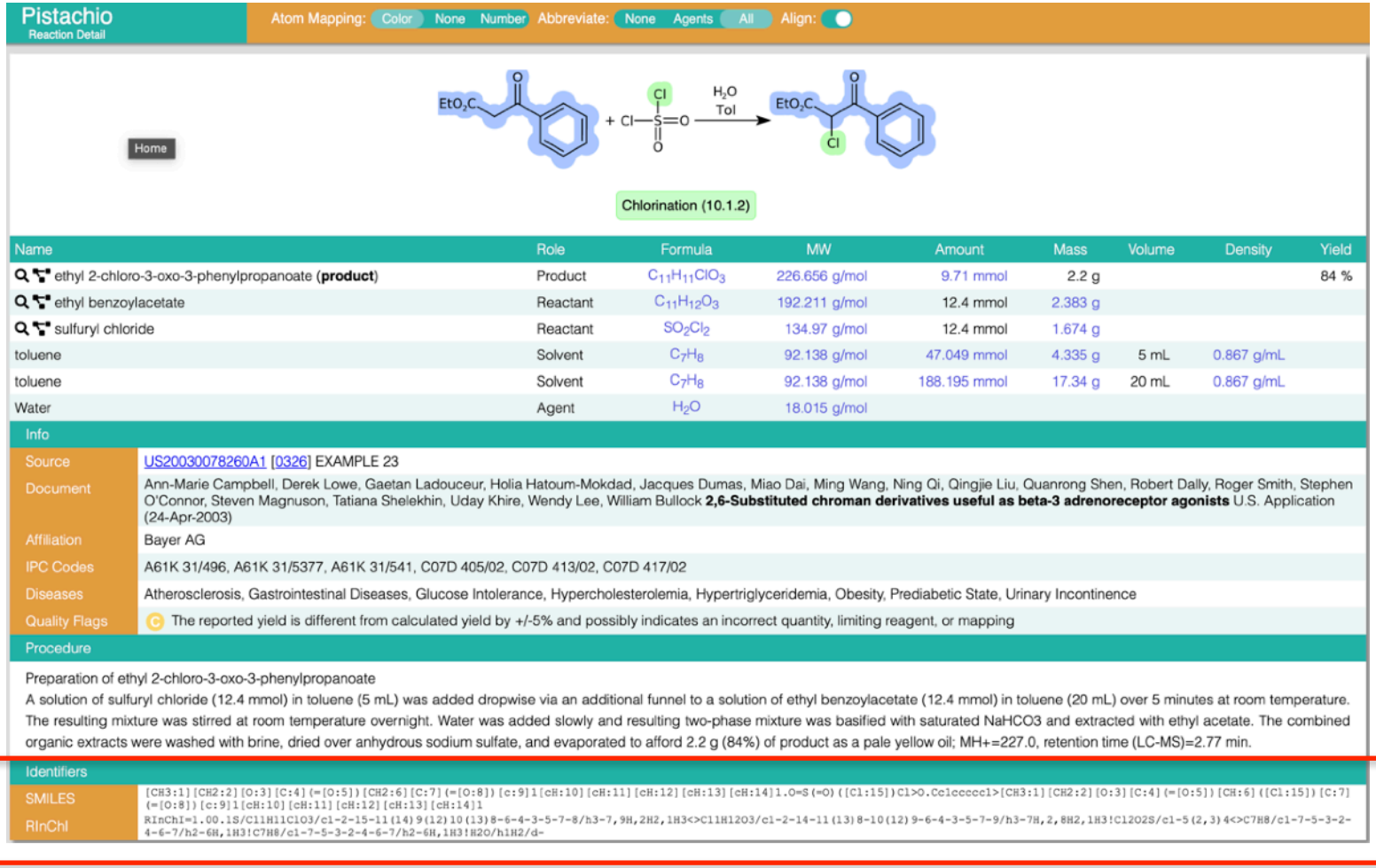


Running the transform
`[#7v4+:1]-[#8H0D1-:2]>>[#7+0:1]=[#8+0:2][H] symmetric-nitro`
over ChEMBL 35 (**2.4M** structures)

19.778 seconds (single threaded, included SMILES IO)
121,346 mol/sec
97k changed

RInChI

(contribution by Uli Fechner and Felix Bänsch)



Name	Role	Formula	MW	Amount	Mass	Volume	Density	Yield
ethyl 2-chloro-3-oxo-3-phenylpropanoate (**product**)	Product	$C_{11}H_{11}ClO_3$	226.656 g/mol	9.71 mmol	2.2 g			84 %
ethyl benzoylacetate	Reactant	$C_{11}H_{12}O_3$	192.211 g/mol	12.4 mmol	2.383 g			
sulfuryl chloride	Reactant	SO_2Cl_2	134.97 g/mol	12.4 mmol	1.674 g			
toluene	Solvent	C_7H_8	92.138 g/mol	47.049 mmol	4.335 g	5 mL	0.867 g/mL	
toluene	Solvent	C_7H_8	92.138 g/mol	188.195 mmol	17.34 g	20 mL	0.867 g/mL	
Water	Agent	H_2O	18.015 g/mol					

Info

Source	US20030078260A1 [0326] EXAMPLE 23
Document	Ann-Marie Campbell, Derek Lowe, Gaetan Ladouceur, Holia Hatoum-Mokdad, Jacques Dumas, Miao Dai, Ming Wang, Ning Qi, Qingjie Liu, Quanrong Shen, Robert Dally, Roger Smith, Stephen O'Connor, Steven Magnuson, Tatiana Shelekhin, Uday Khire, Wendy Lee, William Bullock **2,6-Substituted chroman derivatives useful as beta-3 adrenoreceptor agonists** U.S. Application (24-Apr-2003)
Affiliation	Bayer AG
IPC Codes	A61K 31/496, A61K 31/5377, A61K 31/541, C07D 405/02, C07D 413/02, C07D 417/02
Diseases	Atherosclerosis, Gastrointestinal Diseases, Glucose Intolerance, Hypercholesterolemia, Hypertriglyceridemia, Obesity, Prediabetic State, Urinary Incontinence
Quality Flags	The reported yield is different from calculated yield by +/-5% and possibly indicates an incorrect quantity, limiting reagent, or mapping

Procedure

Preparation of ethyl 2-chloro-3-oxo-3-phenylpropanoate
A solution of sulfuryl chloride (12.4 mmol) in toluene (5 mL) was added dropwise via an additional funnel to a solution of ethyl benzoylacetate (12.4 mmol) in toluene (20 mL) over 5 minutes at room temperature. The resulting mixture was stirred at room temperature overnight. Water was added slowly and resulting two-phase mixture was basified with saturated NaHCO3 and extracted with ethyl acetate. The combined organic extracts were washed with brine, dried over anhydrous sodium sulfate, and evaporated to afford 2.2 g (84%) of product as a pale yellow oil; MH+=227.0, retention time (LC-MS)=2.77 min.

Identifiers

SMILES	[CH3:1][CH2:2][O:3][C:4](=[O:5])[CH2:6][C:7](=[O:8])[c:9]1[cH:10][cH:11][cH:12][cH:13][cH:14]1.O=S(=O)([Cl:15])Cl>O.Cc1ccccc1>[CH3:1][CH2:2][O:3][C:4](=[O:5])[CH:6]([Cl:15])[C:7](=[O:8])[c:9]1[cH:10][cH:11][cH:12][cH:13][cH:14]1
RInChI	RInChI=1.00.1S/C11H11ClO3/c1-2-15-11(14)9(12)10(13)8-6-4-3-5-7-8/h3-7,9H,2H2,1H3<>C11H12O3/c1-2-14-11(13)8-10(12)9-6-4-3-5-7-9/h3-7H,2,8H2,1H3!Cl2O2S/c1-5(2,3)4<>C7H8/c1-7-5-3-2-4-6-7/h2-6H,1H3!C7H8/c1-7-5-3-2-4-6-7/h2-6H,1H3!H2O/h1H2/d-

RInChI=1.00.1S/C11H11ClO3/c1-2-15-11(14)9(12)10(13)8-6-4-3-5-7-8/h3-7,9H,2H2,1H3<>C11H12O3/c1-2-14-11(13)8-10(12)9-6-4-3-5-7-9/h3-7H,2,8H2,1H3!Cl2O2S/c1-5(2,3)4<>C7H8/c1-7-5-3-2-4-6-7/h2-6H,1H3!C7H8/c1-7-5-3-2-4-6-7/h2-6H,1H3!H2O/h1H2/d-

RINCHI

(contribution by Uli Fechner and Felix Bänsch)

Java/CDK is platform independent.
The InChI library is native code and platform dependant.

10-25% of CDK package size is native InChI libs for each platform

```
1211840 Sun Feb 16 00:53:04 CET 2025 darwin-aarch64/libjnainchi.dylib
1280048 Sun Feb 16 00:53:08 CET 2025 darwin-x86-64/libjnainchi.dylib
2320488 Sun Feb 16 00:53:12 CET 2025 linux-aarch64/libjnainchi.so
1214232 Sun Feb 16 00:53:16 CET 2025 linux-arm/libjnainchi.so
1462872 Sun Feb 16 00:53:20 CET 2025 linux-x86/libjnainchi.so
1350808 Sun Feb 16 00:53:24 CET 2025 linux-x86-64/libjnainchi.so
1217255 Sun Feb 16 00:53:30 CET 2025 win32-x86/jnainchi.dll
1570816 Sun Feb 16 00:53:34 CET 2025 win32-x86-64/jnainchi.dll
```

Other approaches: WASM/nestedvm

RInChI

(contribution by Uli Fechner and Felix Bänsch)

The "*official*" RInChI logic is (was?) out-of-sync with the main InChI library.

Therefore would require double the number of native libraries.

The core basic of RInChI is not hard to reimplement in Java…

…the numerous RInChI-Keys required more effort

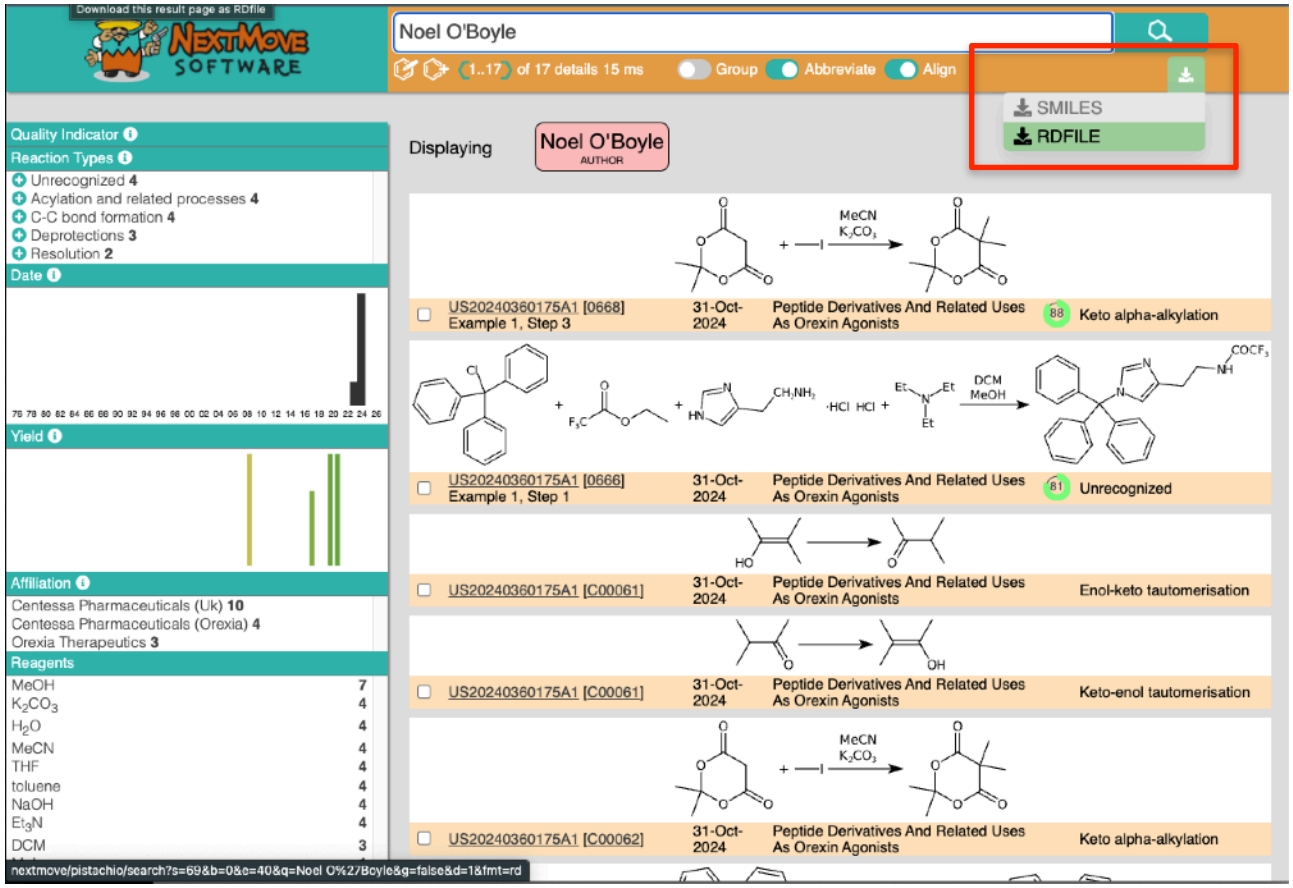
```
IReaction reaction = smipar.parseReactionSmiles("[CH3:1][CH2:2][O:3][C:4]
(=[O:5])[CH2:6][C:7](=[O:8])[c:9]1[cH:10][cH:11][cH:12][cH:13][cH:14]1.O=S(=O)
([Cl:15])Cl>O.Cc1ccccc1>[CH3:1][CH2:2][O:3][C:4](=[O:5])[CH:6]([Cl:15])[C:7]
(=[O:8])[c:9]1[cH:10][cH:11][cH:12][cH:13][cH:14]1");

RInChIGenerator gen = new RInChIGenerator();
gen.generate(reaction);
System.err.println(gen.getRInChI());
System.err.println(gen.getShortRInChIKey());
```

```
RInChI=1.00.1S/C11H11ClO3/c1-2-15-11(14)9(12)10(13)8-6-4-3-5-7-8/
h3-7,9H,2H2,1H3<>C11H12O3/c1-2-14-11(13)8-10(12)9-6-4-3-5-7-9/h3-7H,2,8H2,1H3!
Cl2O2S/c1-5(2,3)4<>C7H8/c1-7-5-3-2-4-6-7/h2-6H,1H3!H2O/h1H2/d-
Short-RInChIKey=SA-BUHFF-FYIINBVGKB-GVJJEELLMH-ZMKZDNITAY-NUHFF-NUHFF-NUHFF-
ZZZ
```

RDfiles

(contribution by Uli Fechner)

RDFiles useful interchange format reaction data



(NextMove Software's Pistachio)

RDFILES

(contribution by Uli Fechner)

Chapter 8: RDfiles

Overview

An RDfile (reaction-data file) consists of a set of editable "records." Each record defines a molecule or reaction, and its associated data. An example RDfile incorporating the rxnfile described in Chapter 7 is shown later in this chapter (see Figure 15 on page 49). The format for an RDfile is:

```
                [RDfile Header]
                [Molecule or Reaction Identifier]
*r          *d  [Data-field Identifier]
                [Data]
```

where:

- *d is repeated for each data item
- *r is repeated for each reaction or molecule

Each logical line in an RDfile starts with a keyword in column 1 of a physical line. One or more blanks separate the first argument (if any) from the keyword. The blanks are ignored when the line is read. After the first argument, blanks are significant.

An argument longer than 80 characters breaks at column 80 and continues in column 1 of the next line. (The argument may continue on additional lines up to the physical limits on text length imposed by the database.)

The RDfile must not contain any blank lines except as part of embedded molfiles, rxnfiles, or data. An identifier separates records.

Part of the CTfile format

RDfiles

(contribution by Uli Fechner)

```
$RDFILE 1
$DATM    03/07/25 02:49
$RFMT
$RXN

      Pistachi0307250249

  4  1
$MOL

  Pistachi0307251449

 10 10  0  0  0  0  0  0  0  0999 V2000
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  1
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  2
    0.0000    0.0000    0.0000 O   0  0  0  0  0  0  0  0  0  3
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  4
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  5
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0  6
    0.0000    0.0000    0.0000 O   0  0  0  0  0  0  0  0  0  7
    0.0000    0.0000    0.0000 O   0  0  0  0  0  0  0  0  0  8
    0.0000    0.0000    0.0000 O   0  0  0  0  0  0  0  0  0  9
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0 10
  1  2  1  0
  2  3  1  0
  3  4  1  0
  4  5  1  0
  5  6  1  0
  6  7  1  0
  2  7  1  0
  6  8  2  0
  4  9  2  0
  2 10  1  0
M  END
$MOL

  Pistachi0307251449

  2  1  0  0  0  0  0  0  0  0999 V2000
    0.0000    0.0000    0.0000 C   0  0  0  0  0  0  0  0  0 11
    0.0000    0.0000    0.0000 I   0  0
  1  2  1  0
M  END
$MOL

  Pistachi0307251449

  3  2  0  0  0  0  0  0  0  0999 V2000
    0.0000    0.0000    0.0000 C   0  0
    0.0000    0.0000    0.0000 C   0  0
    0.0000    0.0000    0.0000 N   0  0
  1  2  1  0
  2  3  3  0
M  END
$MOL
```

```
$DTYPE REACTION:DATE
$DATUM 31-Oct-2024
$DTYPE REACTION:NAMERXNDEF
$DATUM Keto alpha-alkylation
$DTYPE REACTION:SMILES
$DATUM [CH3:1][C:2]1([O:3][C:4]([CH2:5][C:6]([O:7]1)=[O:8])=[O:9])[CH3:10].[CH3:+
11]I>CC#N.O=C(O[K])O[K]>[CH3:11][C:5]1([C:4]([O:3][C:2]([O:7][C:6]1=[O:8])([CH3:+
1])[CH3:10])=[O:9])C
$DTYPE REACTION:REACTIONSMILES
$DATUM [CH3:1][C:2]1([O:3][C:4]([CH2:5][C:6]([O:7]1)=[O:8])=[O:9])[CH3:10].[CH3:+
11]I>CC#N.O=C(O[K])O[K]>[CH3:11][C:5]1([C:4]([O:3][C:2]([O:7][C:6]1=[O:8])([CH3:+
1])[CH3:10])=[O:9])C
$DTYPE REACTION:SOURCE
$DATUM USPTO_TEXT_A
$DTYPE REACTION:LABEL
$DATUM Example 1, Step 3
$DTYPE REACTION:TITLE
$DATUM Peptide Derivatives And Related Uses As Orexin Agonists
$DTYPE REACTION:PARENT_ID
$DATUM 4436752
$DTYPE REACTION:YIELD
$DATUM 88
$DTYPE REACTION:DOCUMENTID
$DATUM US20240360175A1
$DTYPE REACTION:LOCATION
$DATUM 668
$DTYPE REACTION:ID
$DATUM 9220336
$DTYPE REACTION:NAMERXN
$DATUM 3.11.91
```

RDfiles

(contribution by Uli Fechner)

Lazy iterator API (much like **IteratingSDFReader**)

```
try (RdfileReader rdfr = new RdfileReader(in, builder, true)) {
    while (rdfr.hasNext()) {
        RdfileRecord record = rdfr.next();

        if (record.isRxnFile()) {
            IReaction reaction = record.getReaction();
        } else {
            IAtomContainer container = record.getAtomContainer();
        }
    }
} catch (IOException e) {
    throw new RuntimeException(e);
}
```

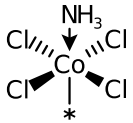
Ultimately the whole of **cdk-ctab** (for MDL formats) probably needs a rethink/realignment e.g. remove V2000/3000 distinction etc

QUESTIONS?

INORGANIC STEREOCHEMISTRY

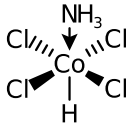
INORGANIC STEREOCHEMISTRY

Support for missing neighbours inorganic stereochemistry



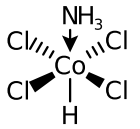
`[NH3][Co@OH25](*)(Cl)(Cl)(Cl)(Cl)`

C[C@H](N)O matched by: **C[C@](N)O** (SMARTS)

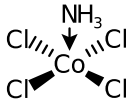


`[NH3][Co@OH25]([H])(Cl)(Cl)(Cl)(Cl)`

C[S@](=O)CC Common extension in SMILES (although some differences)



`[NH3][Co@OH25H](Cl)(Cl)(Cl)(Cl)`

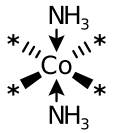


`[NH3][Co@OH25](Cl)(Cl)(Cl)(Cl)`

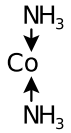
https://github.com/rdkit/rdkit/pull/6777
https://github.com/rdkit/rdkit/pull/7946

INORGANIC STEREOCHEMISTRY

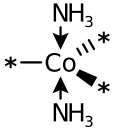
Across atoms can be encoded with @OH1 or @TB1 or @SP3



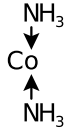
`[NH3][Co@OH1](*)(*)(*)(*)[NH3]`



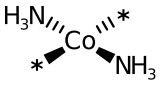
`[NH3][Co@OH1][NH3]`



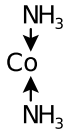
`[NH3][Co@TB1](*)(*)(*)[NH3]`



`[NH3][Co@TBPY1][NH3]`



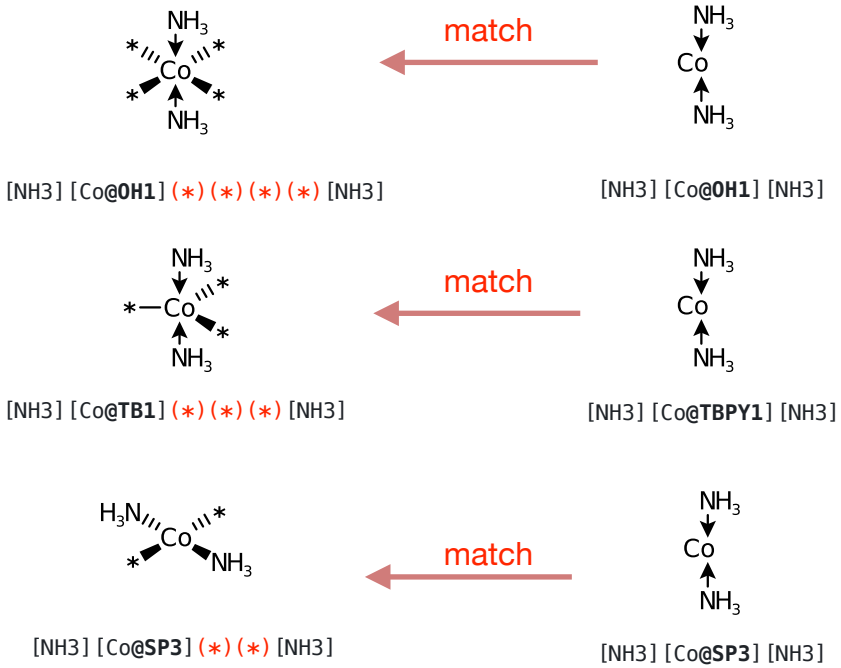
`[NH3][Co@SP3](*)(*)[NH3]`



`[NH3][Co@SP3][NH3]`

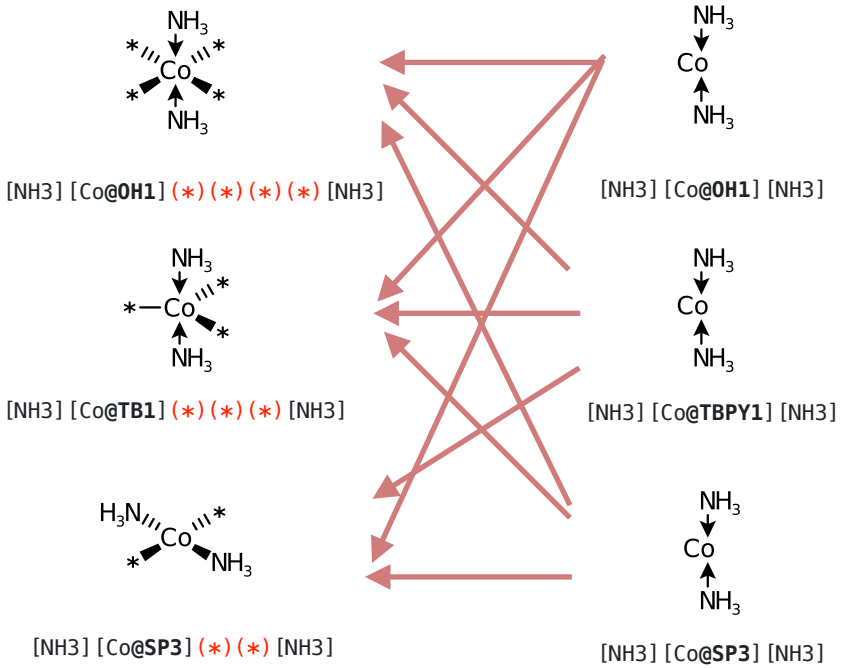
INORGANIC STEREOCHEMISTRY

SMARTS **matching** can be used for querying



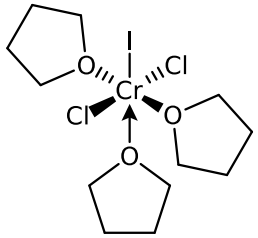
INORGANIC STEREOCHEMISTRY

Degenerate SMARTS matching works correctly



INORGANIC STEREOCHEMISTRY

Degenerate SMARTS matching examples

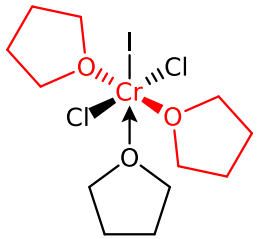


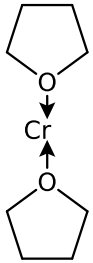
`[Cr@OH8](I)(Cl)(Cl)(O1CCCC1)(O1CCCC1)O1CCCC1`

COD4133160

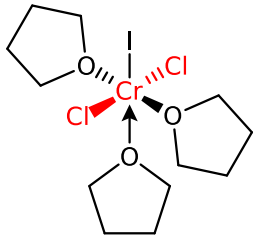
INORGANIC STEREOCHEMISTRY

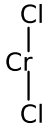
Degenerate SMARTS matching examples



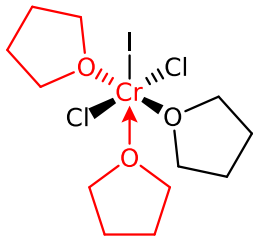


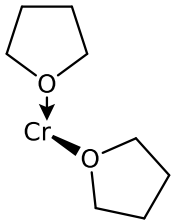
`C1CCCO1[Cr@OH1]O1CCCC1`





`Cl[Cr@TB1]Cl`

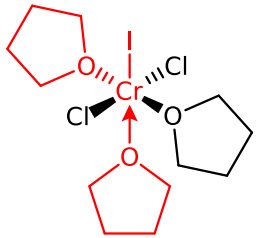


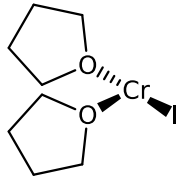


`C1CCCO1[Cr@OH3]O1CCCC1`

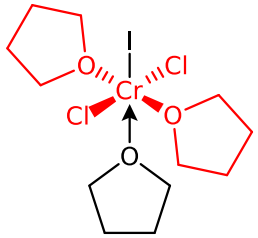
INORGANIC STEREOCHEMISTRY

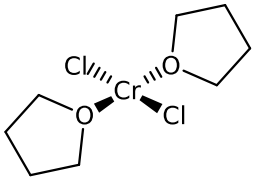
Degenerate SMARTS matching examples



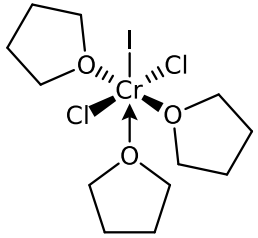


`C1CCCO1[Cr@SP1](I)(O1CCCC1)`

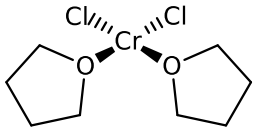




`Cl[Cr@SP2](Cl)(O1CCCC1)O1CCCC1`



no match (correct)



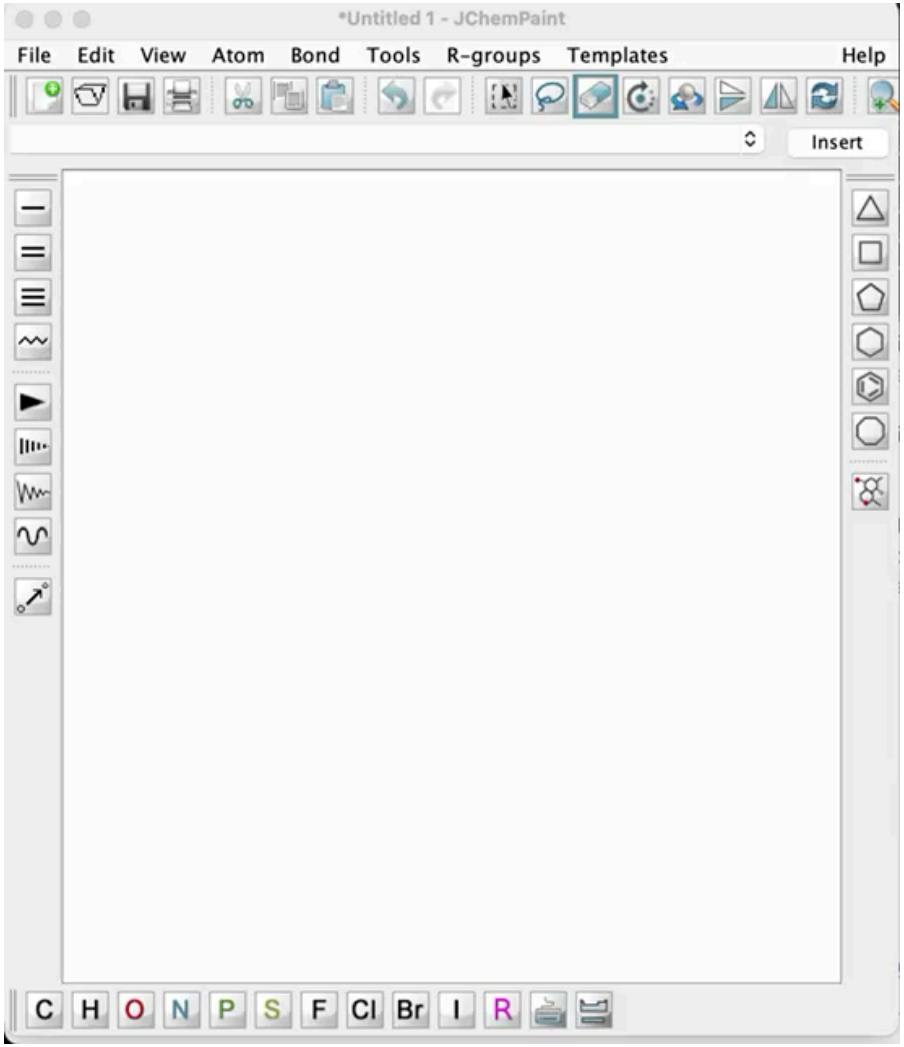
`Cl[Cr@SP1](Cl)(O1CCCC1)O1CCCC1`

QUESTIONS?

SNEEK PEEK

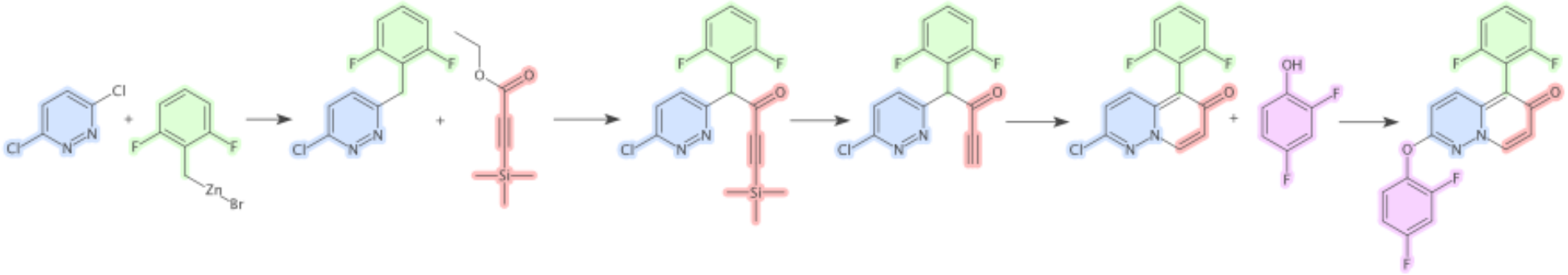
JCHEMPAINT

Updated to CDK v2.10 + many **Quality of Life** improvements



RING FILL HIGHLIGHT

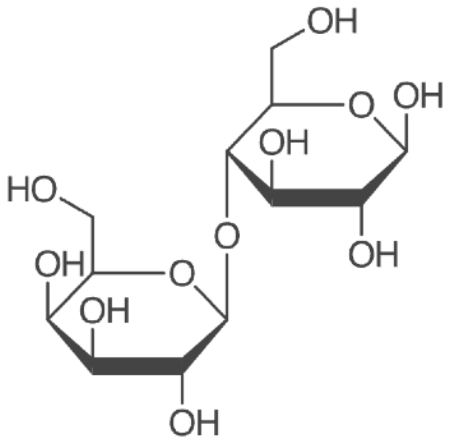
```
new DepictionGenerator().withParam(
  StandardGenerator.Highlighting.class,
StandardGenerator.HighlightStyle.OuterGlowFillRings);
```

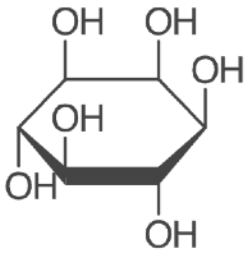


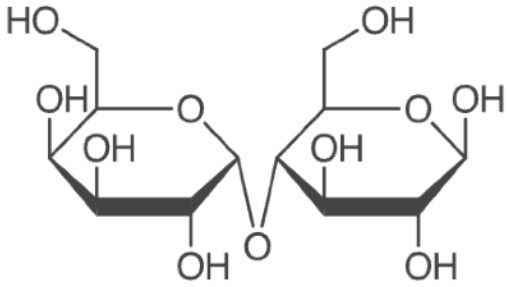
https://github.com/cdk/cdk/pull/1155 (merged)

HAWORTH LAYOUTS

- Layout generation of Haworth projections, currently limited to **two** rings







MORE...

SMIRKS WebApp (like CDK Depict)
Daylight react.cgi:

ACKNOWLEDGEMENTS

Roger Sayle (NextMove Software CEO)
Egon Willinghagen
Uli Fechner
Rachael Pirie (official JChemPaint bug hunter)

NextMove Software (2024)

ENDS