FINGERPRINTING HOT TAKES

(Bits and pieces for better fingerprints)

John Mayfield

WHAT IS A FINGERPRINT?

A **chemical fingerprint** captures selected features of a molecule and can be used for efficient **similarity**, **structure** screening and **reaction** mechanism (difference fingerprint)

Computer science calls it a **bloom filter** but the concepts don't quite map

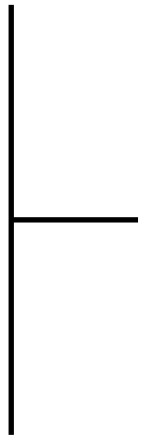
Motivation

- Daylight[1]/Path/RDKit fingerprints are cool
- Fast fingerprints are useful

[1] http://www.dalkescientific.com/writings/diary/archive/2016/12/02/Weiningers_realization.html

SO MANY TO CHOOSE FROM

- Path, Daylight RDKit (unbranched)
- Tree, RDKit (branched), “RDKit OG”
- Circular ECFP Morgan
- Topological Torsion
- Pattern
- AtomPairs
- MACCS
- MHFP



HORSES FOR COURSES

Different fingerprints are good at different tasks

- Any can be used for similarity (*with varied results*)
- Substructure screens can **only** use fingerprints where the bits in a **subgraph** are a **subset**

Depends on **feature type** and **properties** captured:

MACCS

Morgan

AtomPair

MHFP

Topological Torsion

Pattern

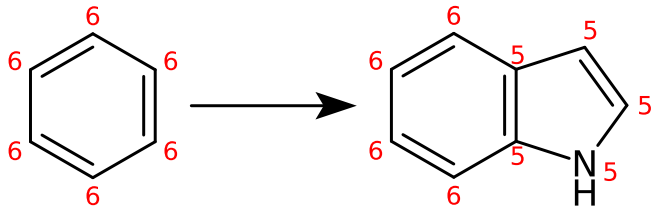
RDKit7 (branched)

RDKit7 (unbranched)

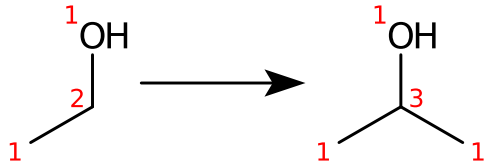
(but depends)

HORSES FOR COURSES

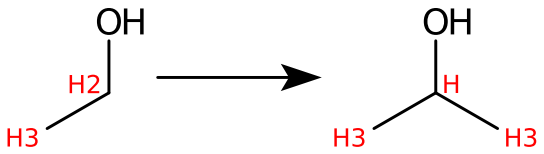
Substructure screening properties must be **invariant**



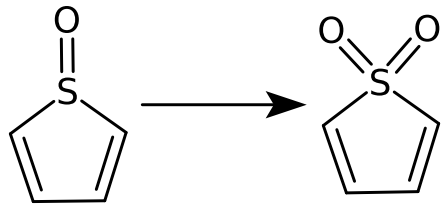
🚫 Smallest Ring



🚫 Degree (*num bonds*)



🚫 Hydrogen Count



🚫 *Aromaticity?*

Aliphaticity, IsInChain, …

https://tinyurl.com/nmblog-fp-2015

BIT DENSITY

How many bits are set determined by:

- number of features
- properties of those features
- hashes per feature

The balancing act

- More bits, more information/space
- Less bits, less information/space

https://greglandrum.github.io/rdkit-blog/fingerprints/reference/2021/07/06/number-of-fp-bits-set.html

paracetamol
Trees
2-7 bonds

7

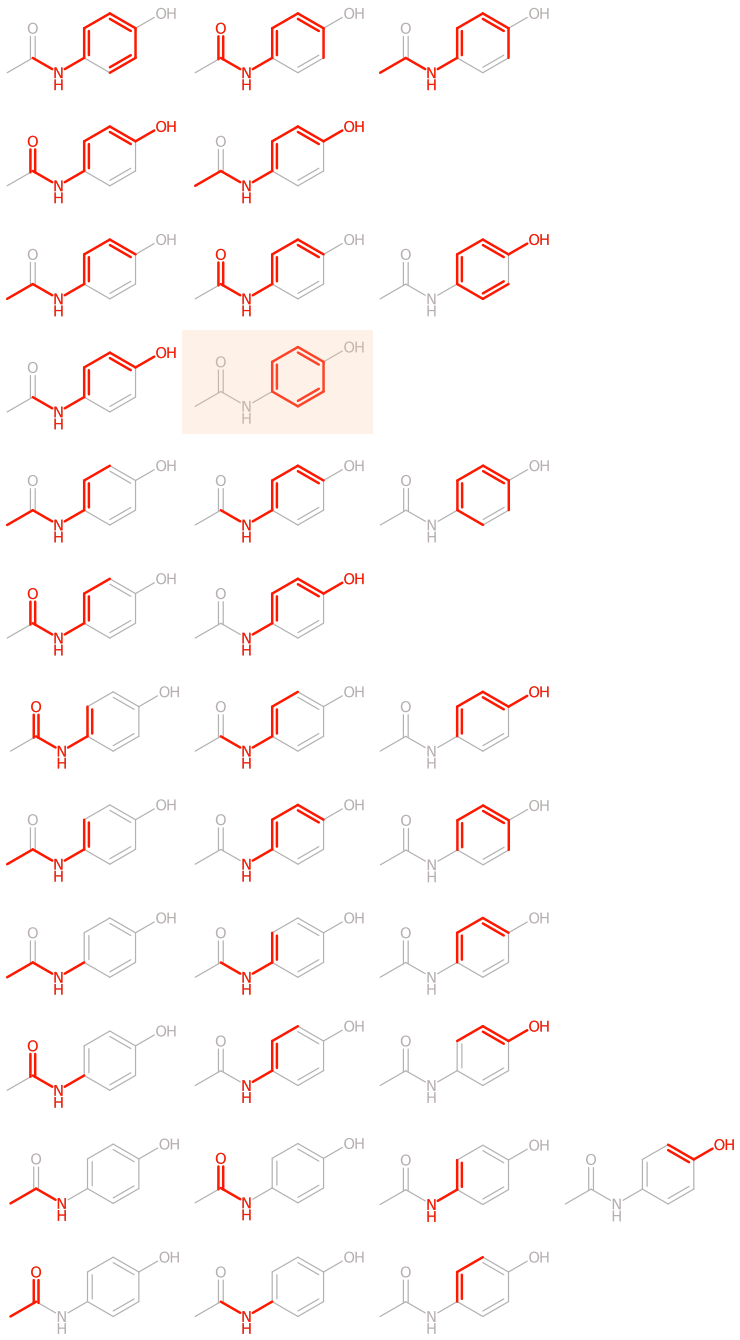
6

5

4

3

2



paracetamol
Paths
2-7 bonds

7

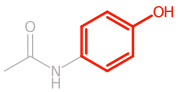
6

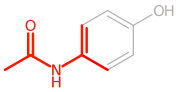
5

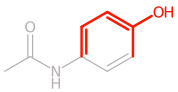
4

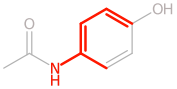
3

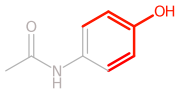
2

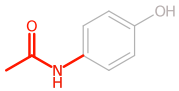


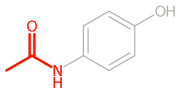


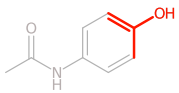


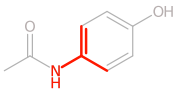


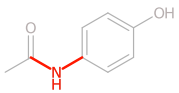


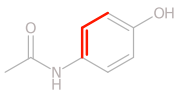






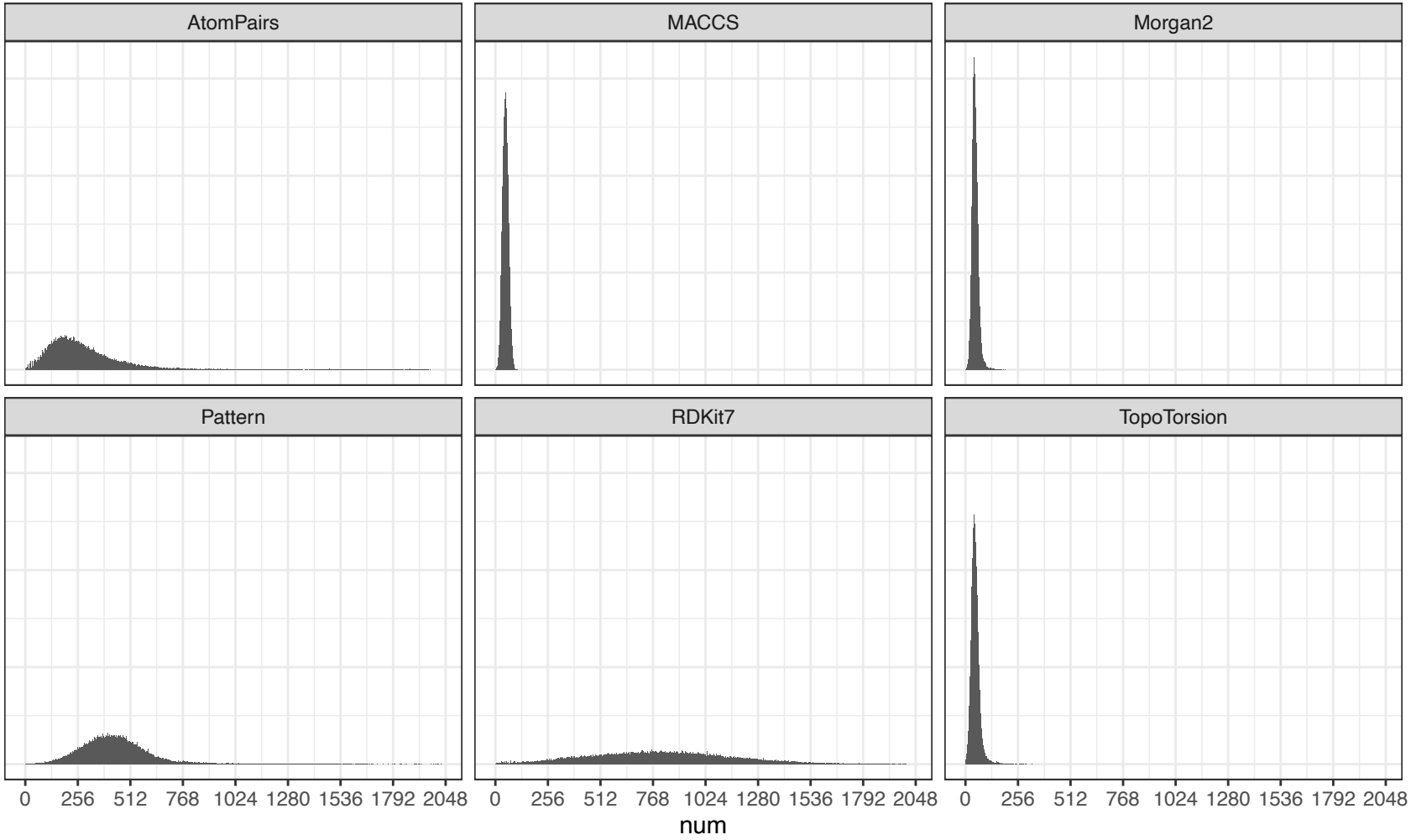






paracetamol
Circular
2-7 bonds

BIT DENSITY



https://greglandrum.github.io/rdkit-blog/fingerprints/reference/2021/07/06/number-of-fp-bits-set.html

ACTIVITY BENCHMARK

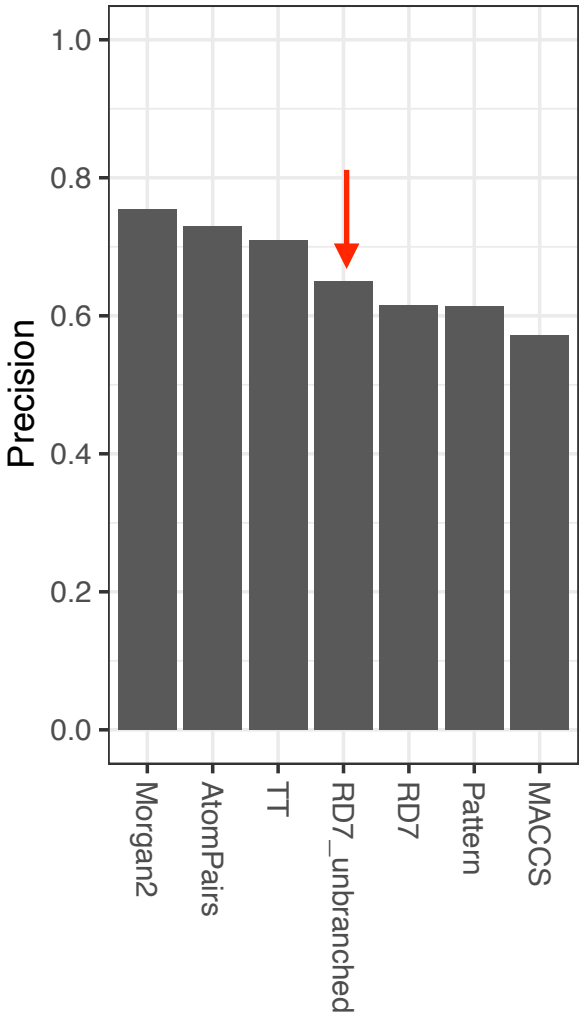
- How ***good*** is a method is for finding **similar** molecules?
- How does it *behave* under different parameters (e.g. folding)

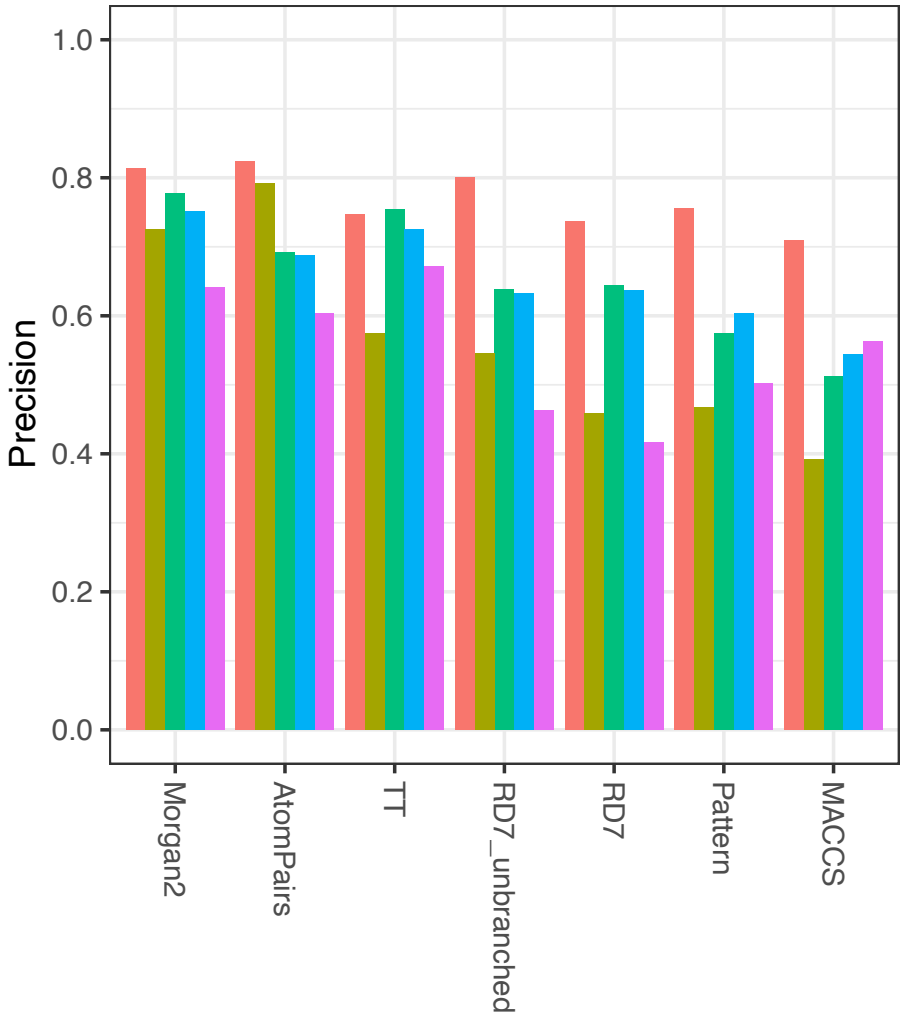
- Framework:
  - Set of **active** and **DECOY** molecules
  - Put them **all** together
  - Find *top-10* closest hits for each **active**
  - Count the number of hits with the same activity class

- Briem/Lessel - (*Lessel and Briem* 2000)
- Landrum/ChEMBL - *"A new lessel and briem like dataset"*

rdkit.blogspot.com/2019/10/a-new-lessel-and-briem-like-dataset.html

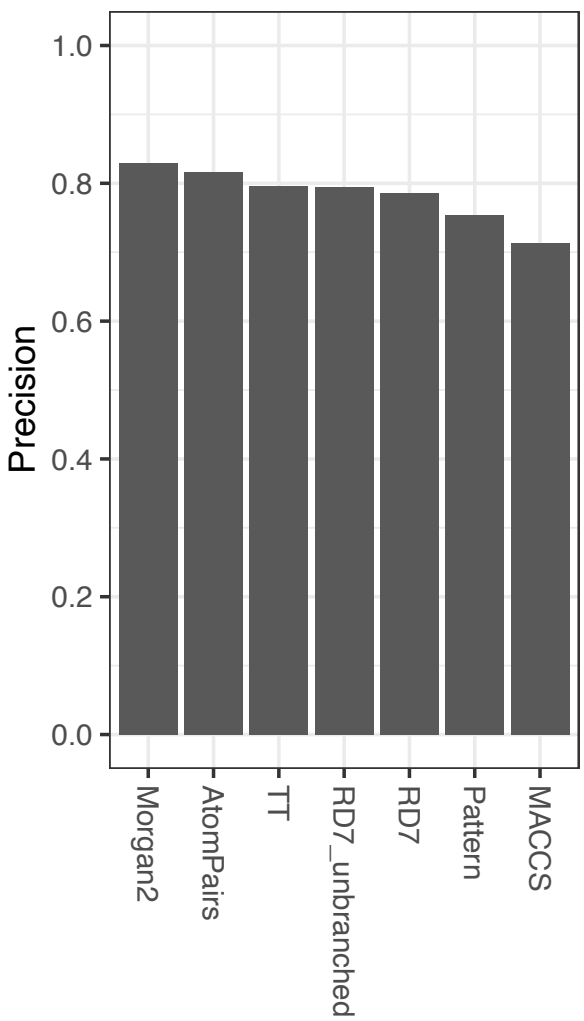
ACTIVITY BENCHMARK II

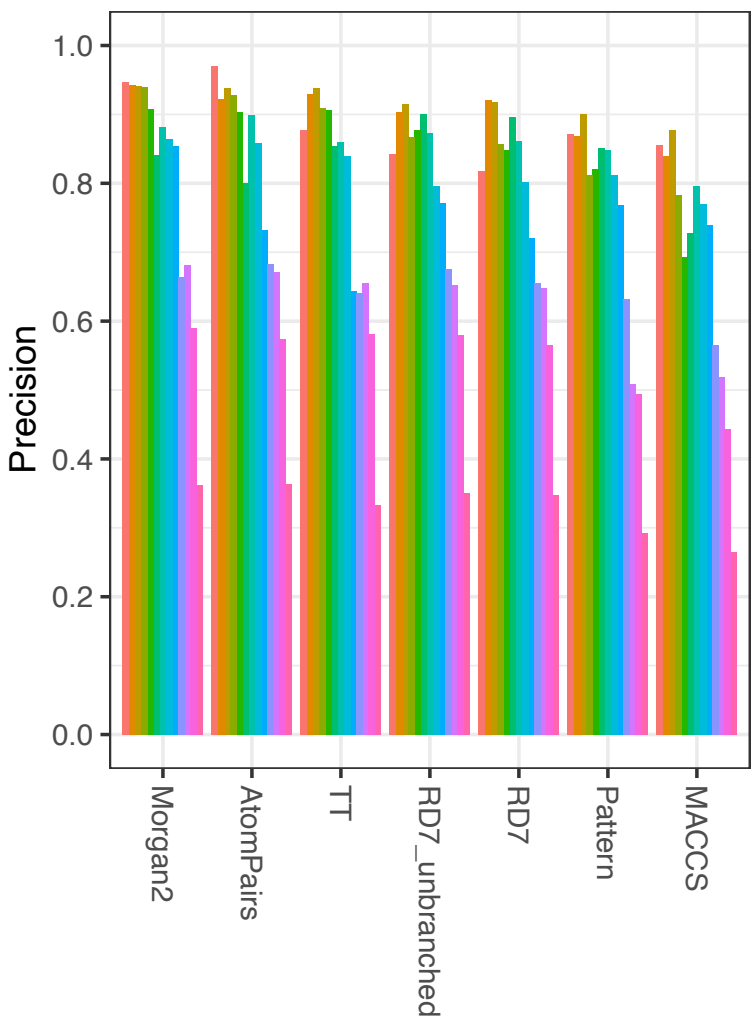


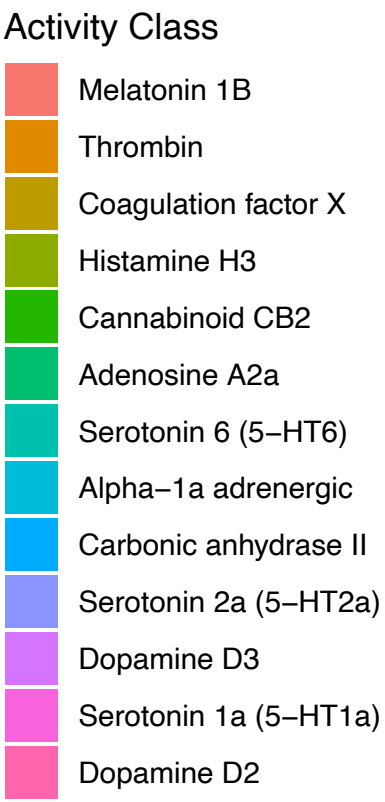


Briem/Lessel

ACTIVITY BENCHMARK III

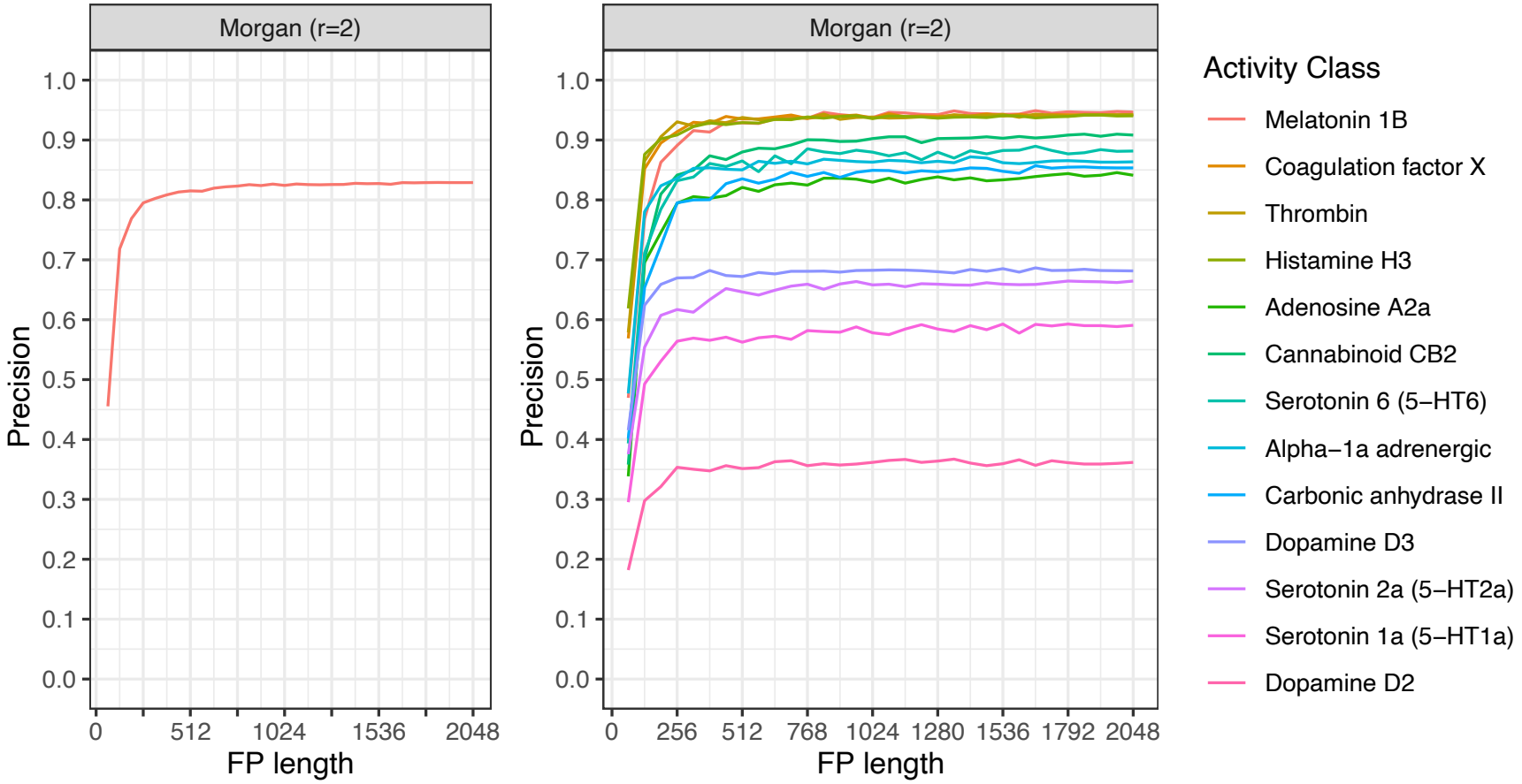






Landrum/ChEMBL

FP FOLDING



Landrum/ChEMBL

THE "RDKIT" FINGERPRINT

ANATOMY OF FINGERPRINT GENERATION

1. Generate some set of subgraph/features
2. Hash properties of each atom and/or bond
    1. Atomic Number?
    2. Bond Type?
    3. Size of feature?
3. Combined these hashes into a single value
4. Set a bit based on the hash value

HASHING/PRNG ALGORITHM

Best but **slower** or good enough and **faster**?

- 🐢 CRC32
- 🐢 MD5
- 🐢 SHA256
- 🚀 DJB2 (Java `String.hashCode()`)
- 🚀 `boost::hash_combine()`

```
static void hash_combine(uint32_t &hash, uint32_t val)
{
  hash ^= val + 0x9e3779b9 +
          (hash << 6) + (hash >> 2);
}
```

- 🐢 Mersenne Twister
- 🚀 XORShift

```
unsigned int xorshift32(unsigned int x)
{
    x ^= x << 13;
    x ^= x >> 17;
    x ^= x << 5;
    return x;
}
```

(depends on needs)

https://en.wikipedia.org/wiki/Xorshift

RDKIT FINGERPRINT OVERVIEW

```
def RDKFingerprintMol(fp, mol):

  atomHashes = hashAtoms(mol.atoms)
  paths = findAllPathsOfLengthsMtoN(mol)

  for path in paths:
    hash = hashBonds(path, atomHashes)
    fp.set_bit(hash % fp.size())
```

```
def hashBonds(path, atomHashes):
  pass

def findAllPathsOfLengthsMtoN(mol):

  adjMatrix = buildAdjMatrix(mol)
  path = pathFinderHelper(adjMatrix)

  # remove duplicates
  seen = [] # vector<dynamic_bitset>
  res  = []
  for path in paths:
    if path not in seen:
      res.append(path)

  return res
```

RDKIT PATH HASHING

NCCO (path)

1. Bond Hashes:
   NC => **0x366a16fa**
   CC => **0x5930a5d3**
   CO => **0x367d2360**

2. Sort Hashes, NumUniqueAtoms:
   [0x366a16fa, 0x367d2360, 0x366a16fa, 0x4]

3. Combined (gboost:hash_combine())
   0x825c8c0f

4. Modulo the number of bits (e.g. % **2048**)
   0x40F (bit 1039)

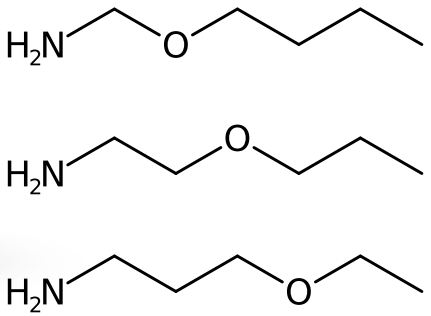
5. PRNG (mersene twister)
   0x825c8c0f => 0x7694520D => **0x20D** (bit 525)

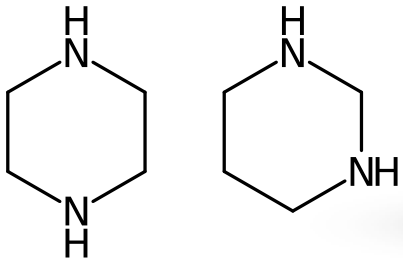
NOT WRONG JUST DIFFERENT

HCl

```
mol = Chem.MolFromSmiles('Cl')
fp = Chem.RDKFingerprint(mol,nBitsPerHash=1,
                         fpSize=65536)
[i for i in range(len(fp)) if fp.GetBit(i) ]
```

```
mol = Chem.MolFromSmiles('NCOCCCC.NCCOCCC.NCCCOCC')
fp = Chem.RDKFingerprint(mol,minPath=6,nBitsPerHash=1,
                         fpSize=65536)
[i for i in range(len(fp)) if fp.GetBit(i) ]
```






```
mol = Chem.MolFromSmiles('N1CCNCC1.N1CNCCC1')
fp = Chem.RDKFingerprint(mol,minPath=6,nBitsPerHash=1,
                         fpSize=65536)
[i for i in range(len(fp)) if fp.GetBit(i) ]
```

NOT WRONG JUST DIFFERENT

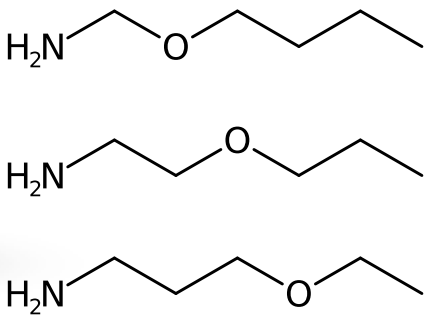
HCl

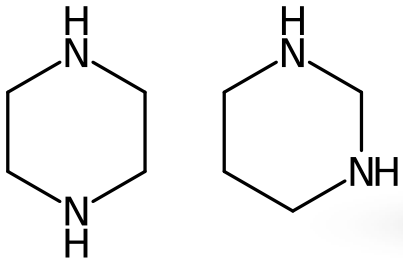
```
mol = Chem.MolFromSmiles('Cl')
fp = Chem.RDKFingerprint(mol,nBitsPerHash=1,
                         fpSize=65536)
[i for i in range(len(fp)) if fp.GetBit(i) ]
```

[ ]

```
mol = Chem.MolFromSmiles('NCOCCCC.NCCOCCC.NCCCOCC')
fp = Chem.RDKFingerprint(mol,minPath=6,nBitsPerHash=1,
                         fpSize=65536)
[i for i in range(len(fp)) if fp.GetBit(i) ]
```

[55698]





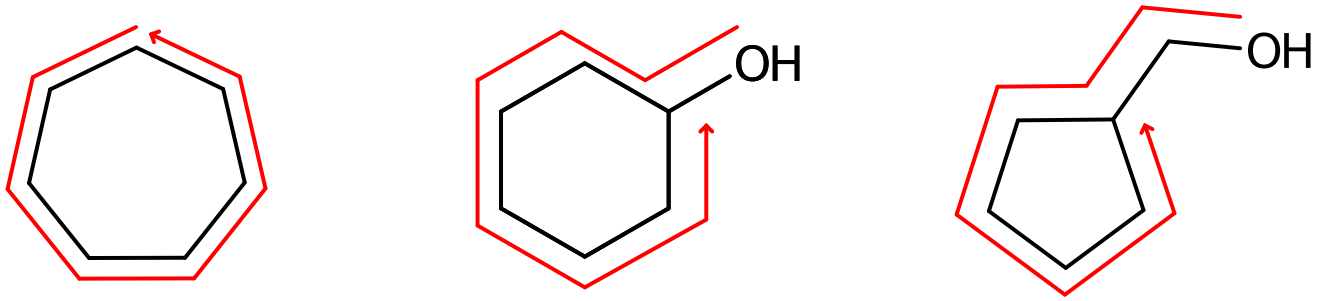

```
mol = Chem.MolFromSmiles('N1CCNCC1.N1CNCCC1')
fp = Chem.RDKFingerprint(mol,minPath=6,nBitsPerHash=1,
                         fpSize=65536)
[i for i in range(len(fp)) if fp.GetBit(i) ]
```

[18034]

NOT WRONG JUST DIFFERENT II

branchedPaths=false encodes cycles/paths with D3 atoms

…but only when **size == maxPath** (default=7)



…smaller rings (size 6 and 5) are not encoded!

FP FOLDING

FP bit set by taking the **hash** modulo the **fpsize**

Size of 2^N is best (and common) choice…

…doesn't just align nicely

```
unsigned int fold_166(unsigned int b) {
    return b % 166;
}

unsigned int fold_1024(unsigned int b) {
    return b % 1024;
}
```

```
fold_166(unsigned int):
        mov     eax, edi
        imul    rax, rax, 827945503
        shr     rax, 37
        imul    edx, eax, 166
        mov     eax, edi
        sub     eax, edx
        ret
fold_1024(unsigned int):
        mov     eax, edi
        and     eax, 1023
        ret
```

FP FOLDING II

FP bit set by taking the **hash** modulo the **fpsize**

Size of 2^N is best (and common) choice

```
unsigned int fold_n(unsigned int h,
                    unsigned int n) {
    return h % n;
}

unsigned int fold_pow2(unsigned int h,
                       unsigned int n) {
    return h & (n-1);
}
```

```
fold_n(unsigned int, unsigned int):
        mov     eax, edi
        xor     edx, edx
        div     esi
        mov     eax, edx
        ret
fold_pow2(unsigned int, unsigned int):
        lea     eax, [rsi-1]
        and     eax, edi
        ret
```

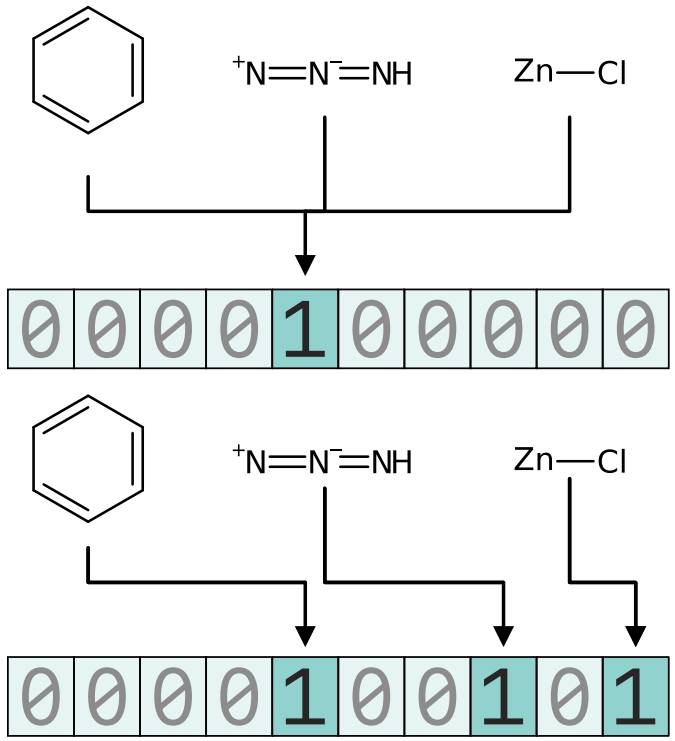
SELECTING BITS

Folding doesn't have to be **random**!

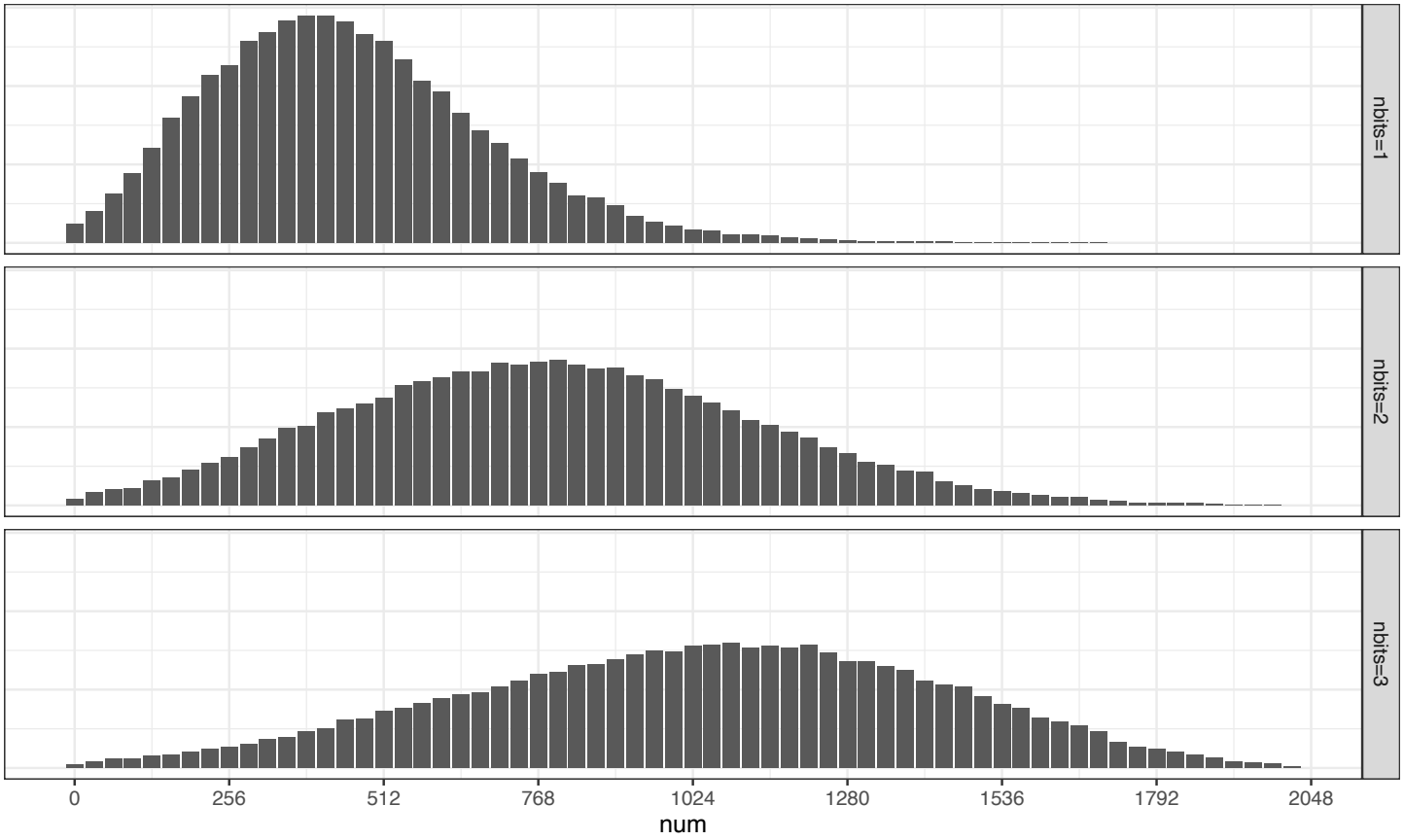
Avoid collisions with common bits

1. Bit frequency on "unfolded" hashes
2. Select bit based on least populated "bucket" (greedy)
3. Store the remapping for reuse

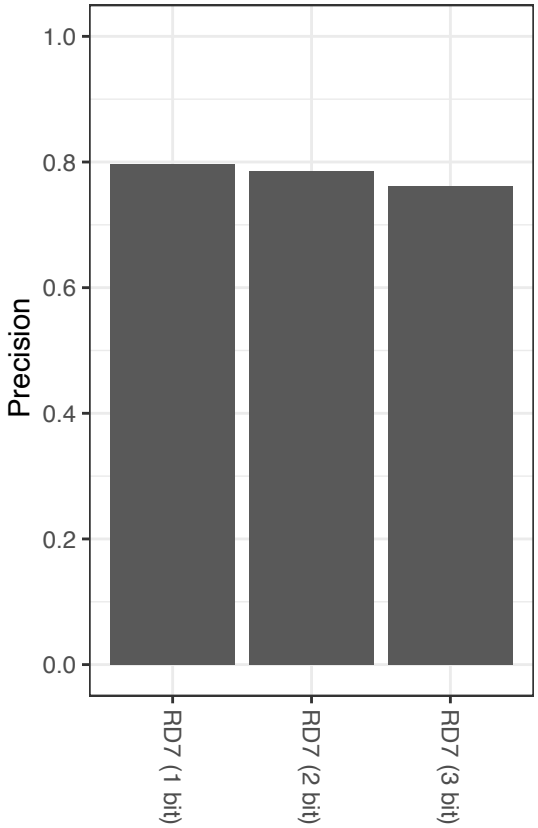
Non-uniform but can be useful!

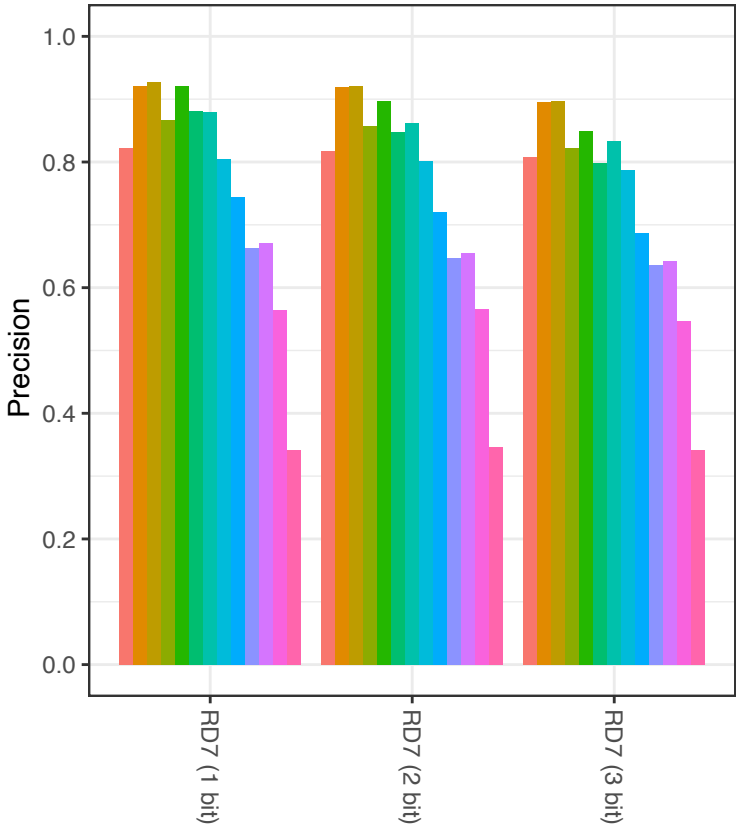


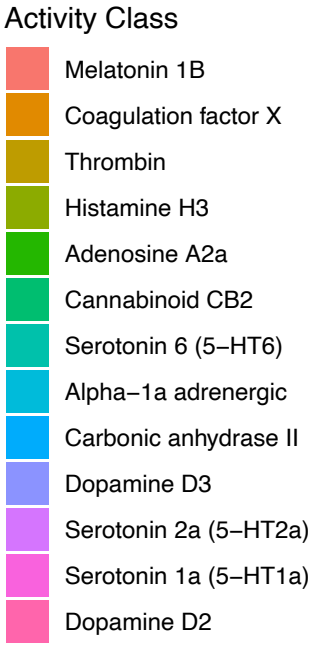
RDKIT7 BITS PER HASH



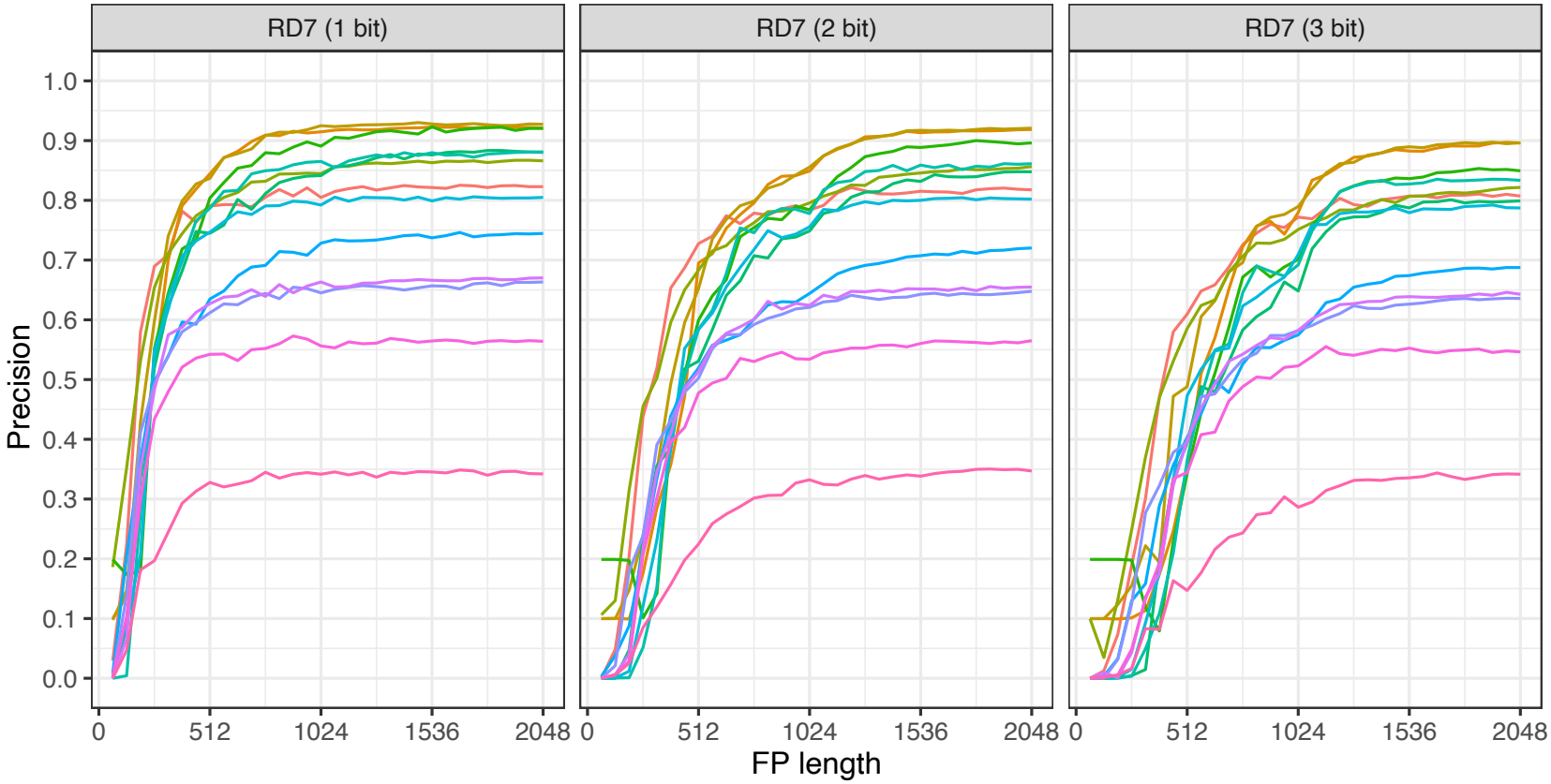
FP FOLDING RDKIT7





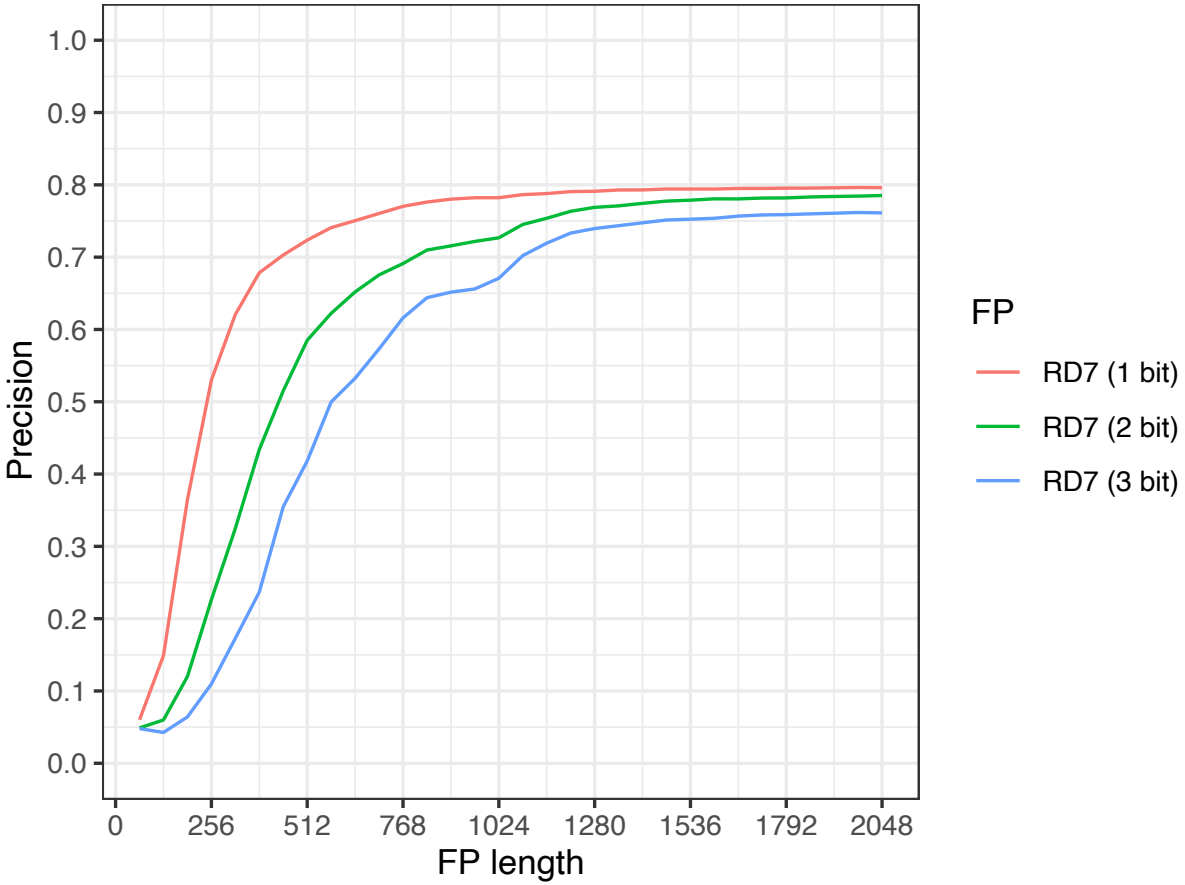


FP FOLDING RDKIT7



Landrum/ChEMBL

FP FOLDING RDKIT7



HOW BIG A FINGERPRINT DO YOU NEED?

The **number of features**

131,636	ChEMBL	~2M
1,274,747	PubChem	~100M
264,046	Zinc	~1,700M
68,952	Enamine	~5,500M

Feature types in **Zinc**

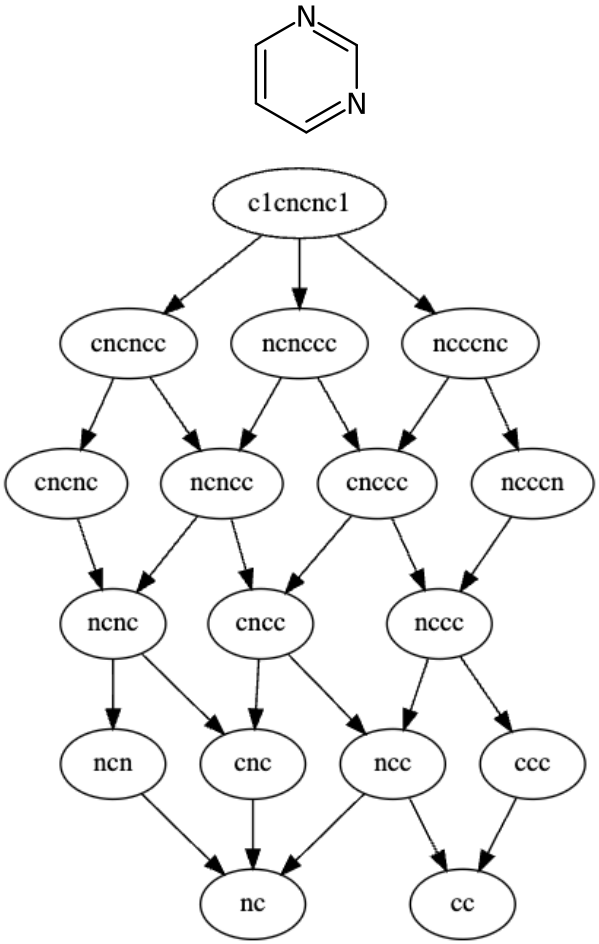
12	path0	57	ring3
171	path1	141	ring4
1,176	path2	795	ring5
5,206	path3	1,361	ring6
19,320	path4	1,156	ring7
61,178	path5	2,143	ring8
175,546	path6	5,629	ring9

NOT ALL DATA LOOKS THE SAME

	ChEMBL		PubChem		Enamine	
1	2,168,022	cc	97,855,732	CC	5,473,307,675	CC
2	2,142,576	ccc	93,589,722	cc	5,461,345,035	CN
3	2,084,597	cccc	91,622,660	ccc	5,372,138,787	CCN
4	2,039,442	ccccc	88,204,447	cccc	5,366,510,414	C=O
5	1,997,600	CC	84,964,389	ccccc	5,316,940,253	CNC
6	1,958,733	cccccc	83,050,894	CN	5,297,069,369	CCC
→ 7	1,931,176	c1ccccc1	82,691,428	Cc	5,283,242,702	NC=O
8	1,897,639	Cc	80,315,502	CCC	5,269,463,921	CCNC
9	1,850,151	Ccc	80,051,318	Ccc	5,143,691,918	CNC=O
10	1,787,641	Cccc	80,010,917	cccccc	5,140,099,867	CCCN
			11	78,969,230 c1ccccc1	66	2,494,976,722 c1ccccc1

Is **benzene** worth encoding?

BIT DEPENDENCY



There is **correlation** between bits

Possible to track this correlation as the fingerprint is generated

implicitly allows signal recover from bit collisions

ARTHOR CHALLENGES

Arthor (*currently*) uses a **1024-bit** path-based fingerprint…

essentially the same as "unbranched paths"

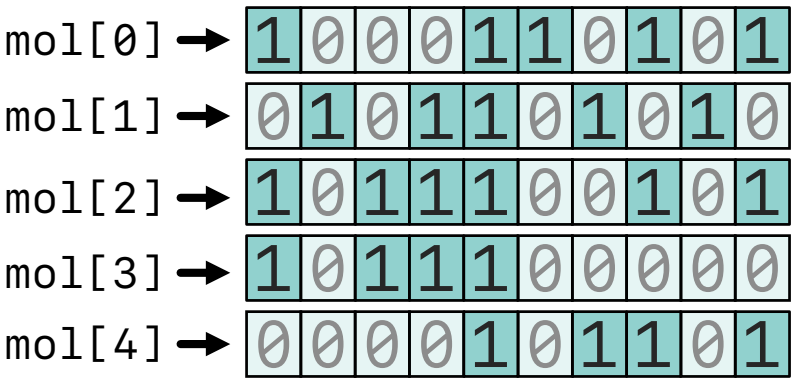
- Path 0 .. 6 bonds
- Rings 3 .. 10
- Different hashing scheme

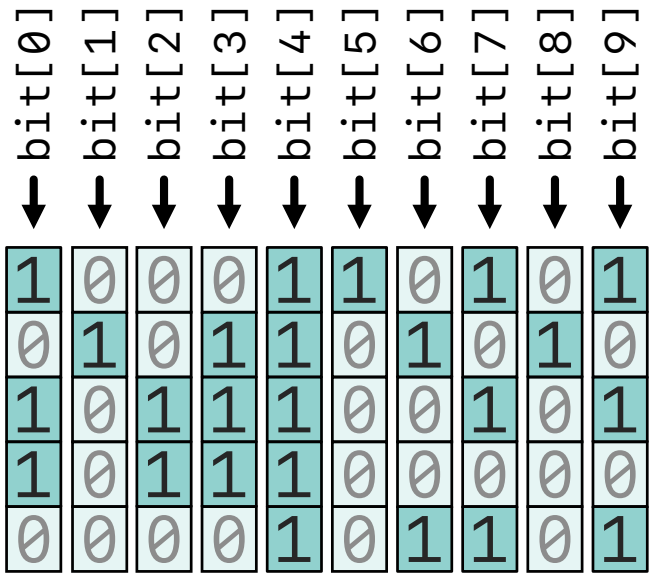
Arthor's connection tables are (*currently*) **111 bytes**…

… so fingerprint is bigger than the molecule

But! still useful for pruning, just need to be efficient

INVERTED INDEX II





1 billion molecules,

1024-bit FP each,

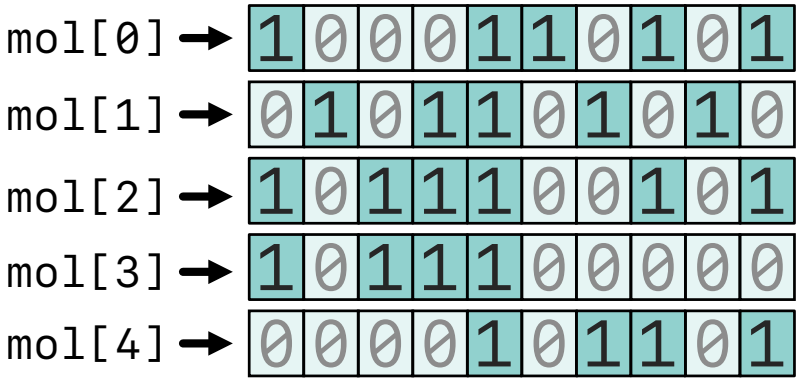
128GB!

(~21 mins on a HDD @ ~100MBps)

Using an inverted bitmap means we only need to read the bits present in the query **128MB per bit**

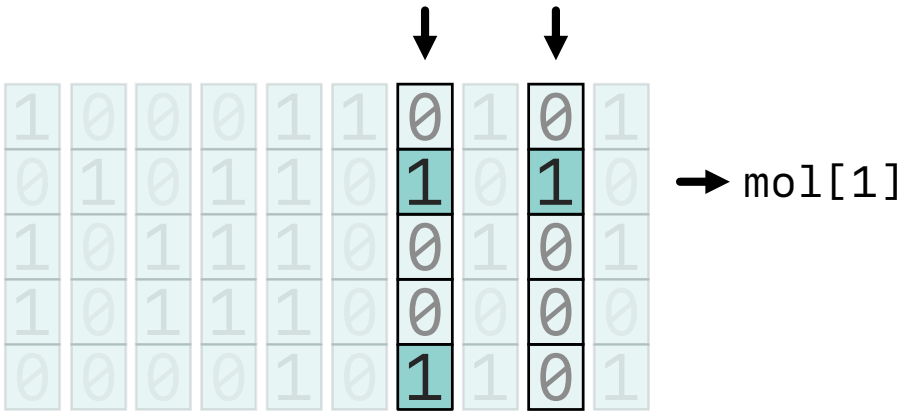
(for 1 billion mols)

INVERTED INDEX II



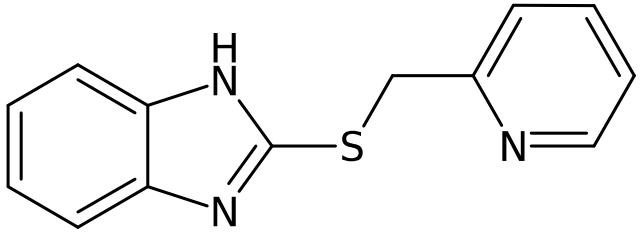
Storing the frequency each bit is set allows us to choose which bits to use.

Use the **rarest** *n* bits!



RARE BIT SCREENING

Feature	*n* Bits	Checked
–	0	1.7 billion
ncSCc	1	5.5 million
cSCcncc	2	560,246
cSCcnc	3	460,173
cccncS	4	252,462
cncSC	5	251,730
cSCcc	6	110,821
Scnccn	7	18,797
ccncncc	8	13,010
–	ALL	7,532



esomeprazole scaffold

(2,499 hits in Zinc 2022)

RARE BIT SCREENING

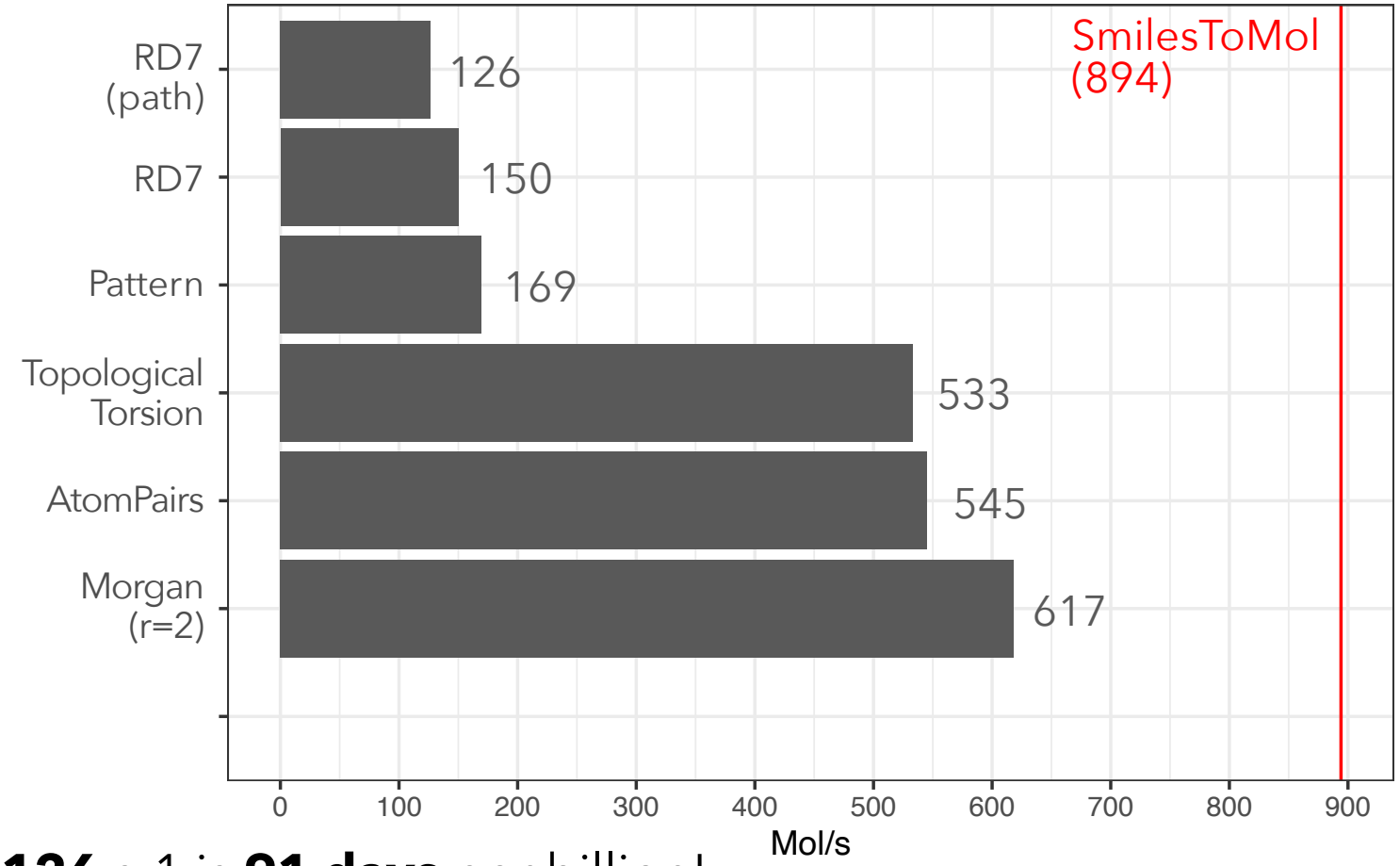
Feature	*n* Bits	Checked	1 Thread	16 Threads	64 Threads
–	0	1.7 billion	51s 74ms	2s 971ms	1s 360ms
ncSCc	1	5.5 million	1s 2ms	82ms	46ms
cSCcncc	2	560,246	277ms	41ms	35ms
cSCcnc	3	460,173	254ms	37ms	36ms
cccncS	4	252,462	212ms	37ms	35ms
cncSC	5	251,730	212ms	36ms	34ms
cSCcc	6	110,821	207ms	36ms	32ms
Scnccn	7	18,797	169ms	33ms	32ms
ccncncc	8	13,010	165ms	33ms	34ms
–	ALL	7,532	186ms	36ms	34ms

RARE BIT SCREENING

Feature	*n* Bits	Checked	1 Thread	16 Threads	64 Threads
–	0	1.7 billion	51s 74ms	2s 971ms	1s 360ms
ncSCc	1	5.5 million	1s 2ms	82ms	46ms
cSCcncc	2	560,246	277ms	41ms	35ms
cSCcnc	3	460,173	254ms	37ms	36ms
cccncS	4	252,462	212ms	37ms	35ms
cncSC	5	251,730	212ms	36ms	34ms
cSCcc	6	110,821	207ms	36ms	32ms
Scnccn	7	18,797	169ms	33ms	32ms
ccncncc	8	13,010	165ms	33ms	34ms
–	ALL	7,532	186ms	36ms	34ms

SPEED BENCHMARK

How fast are the fingerprint methods (**Molecules per second**):



126 s-1 is **91 days** per billion!

(Times recorded on 100,000 ChEMBL molecules on M1 Pro)

RDKIT FAST PATH IMPLEMENTATION

```
struct RdPathFp {

  RDKit::Bond *bonds[MAX_PATH_CAP];
  unsigned int len = 0;

  unsigned char degree[...];
  unsigned char avisit[…];

  void encode() {
    ...
  }

  bool push(RDKit::Bond *bond) {
    bonds[len++] = bond;
    degree[bond->getBeginAtomIdx()]++;
    degree[bond->getEndAtomIdx()]++;
    encode();
    return len < max_len;
  }

  void pop() {
    RDKit::Bond *bond = bonds[--len];
    degree[bond->getBeginAtomIdx()]--;
    degree[bond->getEndAtomIdx()]--;
  }

  …
}
```

Hash the paths as they are discovered

Use a **stack** of bonds

Encode hash whenever a new bond is pushed

johnmay/rdpath-ugm22

RDKIT FAST PATH IMPLEMENTATION

```
void traverse(RDKit::Atom *atom)
{
  if (push(prev)) {
    avisit[atom->getIdx()] = 1;

    for (RDKit::Bond *bond = mol->atomBonds(atom)) {
      RDKit::Atom *nbor = bond->getOtherAtom(atom);
      if (!avisit[nbor->getIdx()])
        traverse(nbor, bond);
      else if (len+1 == max_len)
        encode_ring(bond);
    }

    avisit[atom->getIdx()] = 0;
  }
  pop();
}
```

back-edges iff *maxSize*

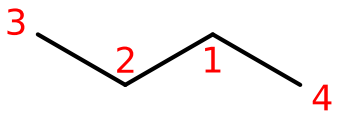
```
void generate(RDKit::ROMol *mol)
{
  memset(avisit,0,mol->getNumAtoms());

  for (RDKit::Atom *atom : mol->atoms()) {
    avisit[atom->getIdx()] = 2;
    for (RDKit::Bond *bond = mol->atomBonds(atom))
      traverse(bond->getOtherAtom(atom), bond);
    avisit[aptr->getIdx()] = 0;
  }
}
```

avisit[] = 2 for root atom

LEXICOGRAPHIC ORDERING

Each path gets visited twice…

… not a problem but quick to deduplicate

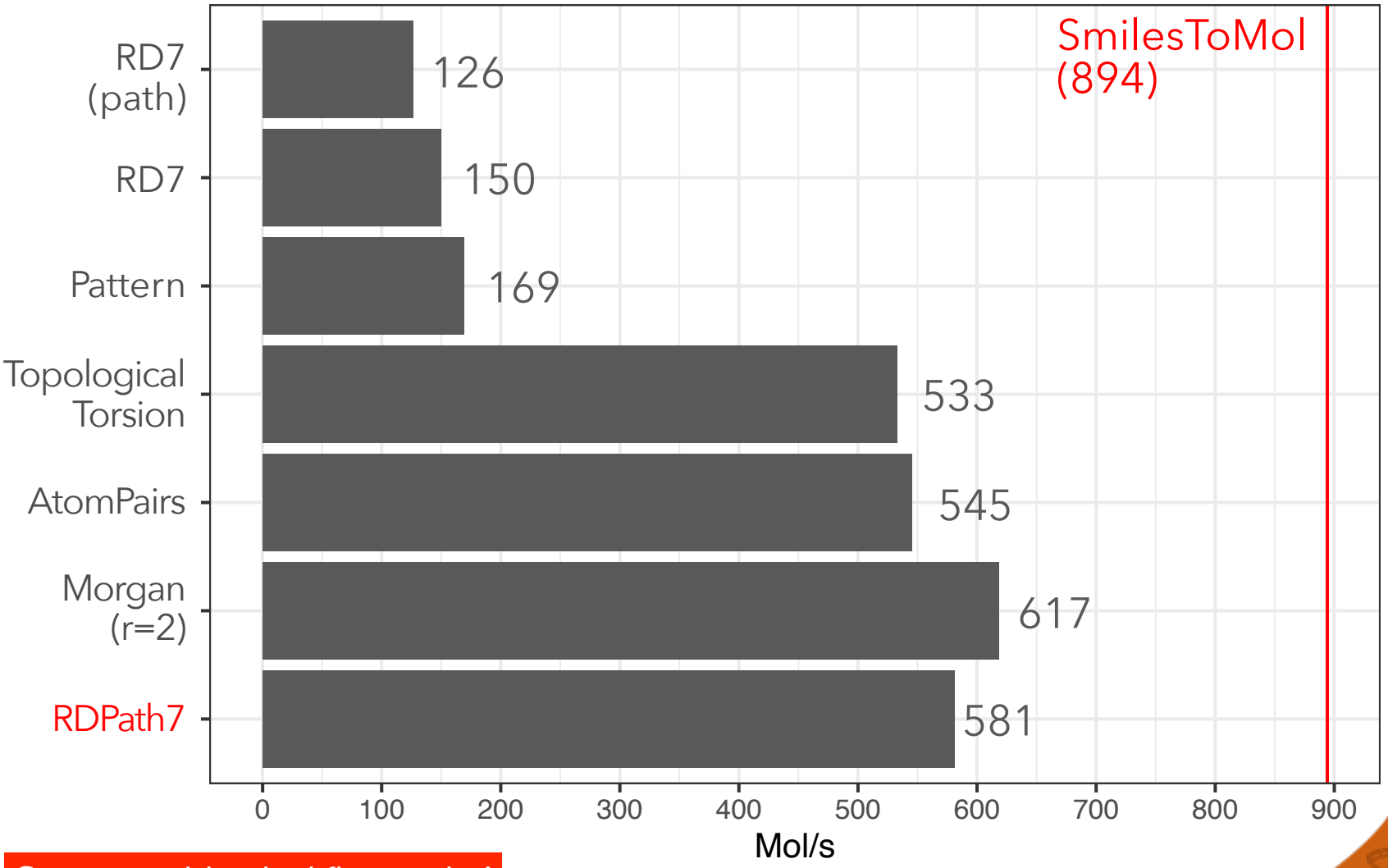



```
if (path[0] < path[len-1])
  do_encode(path);
```

1,2 => encode
1,2,3 => encode
1,4 => encode
2,1
2,1,4 => encode
2,3 => encode
3,2
3,2,1
3,2,1,4 => encode
4,1
4,1,2
4,1,2,3

…similar idea for rings

RDKIT FAST PATH IMPLEMENTATION



Generates identical fingerprint!

TWEAKS

Find rings 3, 4, .., *maxSize*.

```
if (!avisit[nbor->getIdx()])
  traverse(root, nbor, bond);
else if (avisit[nbor->getIdx()] == 2)
  encode_ring(bond);
```

Hash bonds once and differently

```
uint32_t hash_bond(const RDKit::Bond *bond)
{
  uint32_t hash = 13;
  const RDKit::Atom *beg = bond->getBeginAtom();
  const RDKit::Atom *end = bond->getEndAtom();
  uint32_t bhash = 2*beg->getAtomicNum() + beg->getIsAromatic();
  uint32_t ehash = 2*end->getAtomicNum() + end->getIsAromatic();
  gboost::hash_combine(hash, bhash * ehash); // bidirectional
  if (bond->getIsAromatic())
    gboost::hash_combine(hash, RDKit::Bond::AROMATIC);
  else
    gboost::hash_combine(hash, bond->getBondType());
  return hash;
}
```

DIFFERENT HASHING

```
bool push(RDKit::Bond *bond) {
  bonds[len] = bond;
  bhash[len] = bondHashes[bond];
  len++;
  encode();
  return len < max_len;
}
```

Push/pop hashes

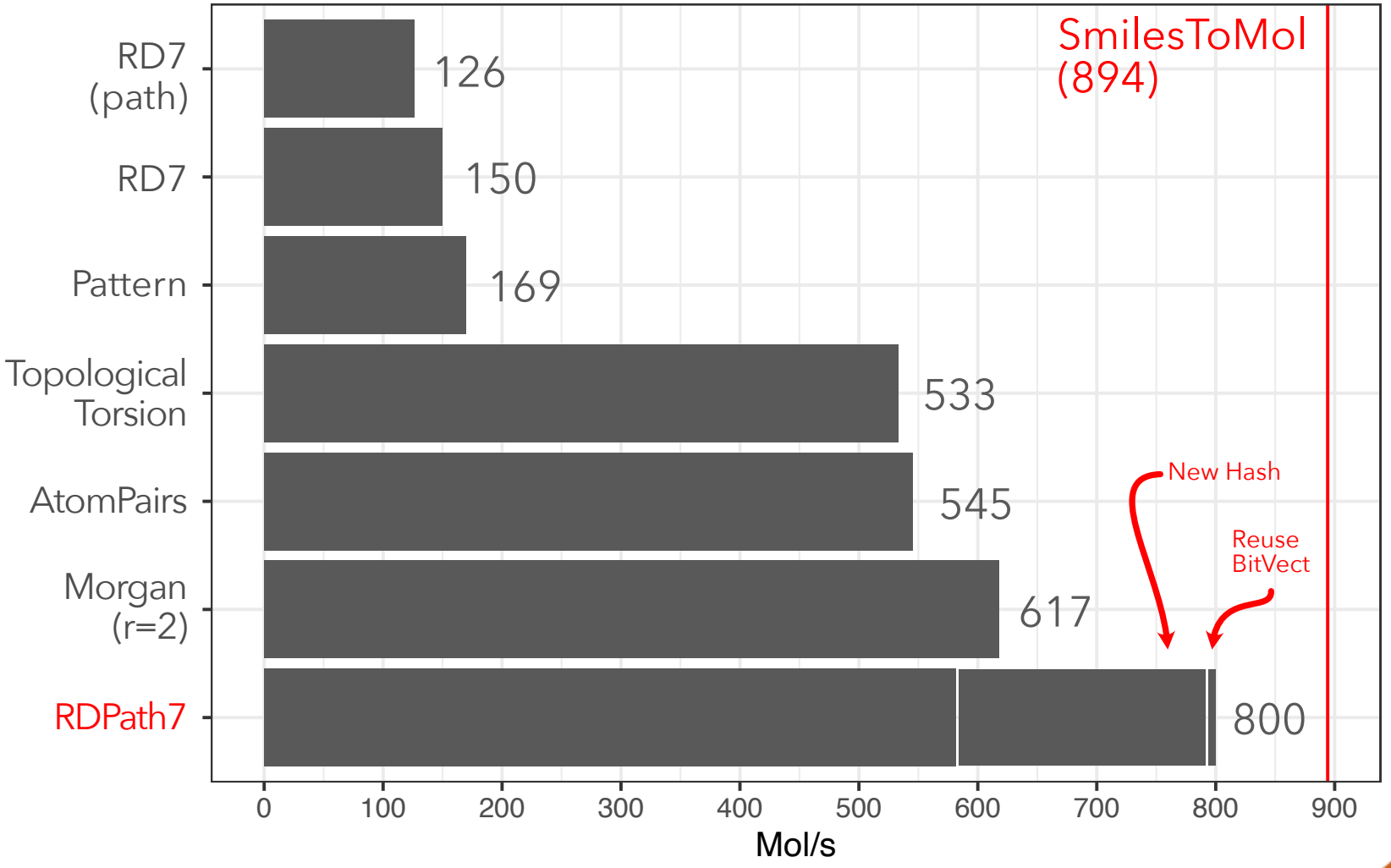
hash("CCO") == hash("OCC")

```
void encode_path() {
  unsigned int hash;
  if (is_reversed(bhash, len) > 0)
    hash = hash_fwd(bhash, len);
  else
    hash = hash_rev(bhash, len);
  bit_vect->setBit(hash & fp_mask);
}
```

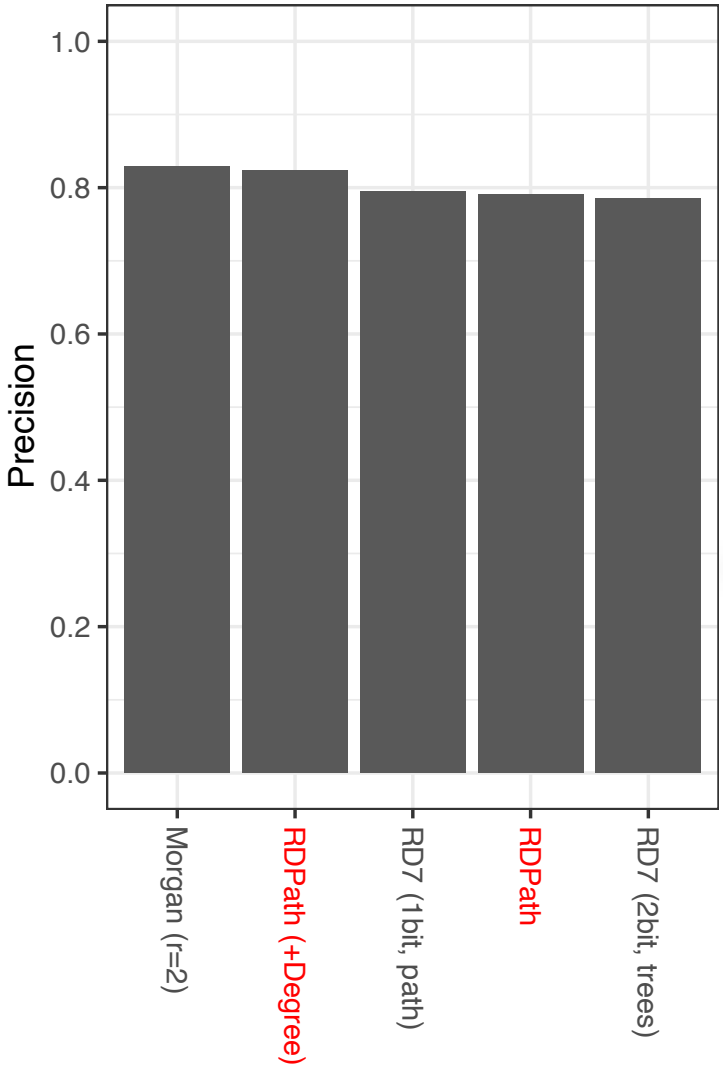
Hash forward or reverse

(see **Topological Torsion**!)

RDKIT FAST PATH IMPLEMENTATION



ACTIVITY BENCHMARK




```
// for similarity ONLY!
uint32_t bhash = 2*beg->getAtomicNum() +
                   beg->getIsAromatic() +
                   (beg->getDegree() << 8);
uint32_t ehash = 2*end->getAtomicNum() +
                   end->getIsAromatic() +
                   (beg->getDegree() << 8);
```

SUMMARY/TAKEHOME

- Overview multiple hashes, folding, benchmarks, inverted indexes
- Described a faster **path** based fingerprint
  - Identical fingerprint (under certain options)
  - More distinct hashing, better for sub-screens
  - The **Pattern** fingerprint looks for paths **matcher**, can use a path fp (len=3)

Skipped:

  - Daylight set multiple bits per featured, but depends on size e.g. 3, 2, 1 for path lengths 1, 2, ≥3
  - Inverted indexes (posting lists) can be compressed allows sparser fingerprints in less space (time/space tradeoff)
  - Same algorithm can produce variable size hashes (CANSMI)

MY OPINIONS

- Separate fingerprints for
  - **path** (unbranched path)
  - **tree** (branched paths)
- Nicer API to pass in the **ExplicitBitVect** then create a new
- **ExplicitBitVect** API improvements (iterators, `toBytes()`)
- Fingerprint namespaces are inconsistent

I understand changing/breaking things is difficult, useful to have a deprecation mechanism

ACKNOWLEDGEMENTS

NextMove Software

Roger Sayle

Delia Sayle

Ingvar Lagerstedt

Micheal Blakey

Alumni

Richard Gowers

RDKit

Greg Landrum

Boran Adas (GSOC)

other RDKit contributors