BLINCS: Breadth-first Line Notation for Chemical Structures

Wim Dehaen

RDKit UGM 2025

Prague

Introduction: Line notations

- SMILES
  - **Depth first traversal** of the chemical graph (CG)
    - SMILES is very expressive: not strictly DFS
  - Commonly used, compact way to store a CG
  - Can be "hand-rolled"
  - Works well in Molecular Generators, Chemical Language Models (CLMs)
    - Despite canonicalization issue !
    - Invalid SMILES frequently generated
- SELFIES
  - "Guarantees" validity
    - using token dropping trick and token double meanings
  - Intended for use in ML applications like molecular generators
- Others
  - New: DeepSMILES, FragmentSMILES
  - Older: WLN, SYBYL, InChI, ...
  - IUPAC notation ... ?

"Ordered degree sequence"

- **ODS**: The degree of every vertex in the order of traversal
  - in graph theory, sorted DS is more typical, because it is a graph invariant
  - Several ODS per graph is possible (typical)
- In case of breadth first traversal of a **tree**: ODS allows reconstruction of tree
  - Why? Degree – 1 = nChildren (except for the root node where Degree = nChildren)
- **Graphs** need some more extra info to allow unambiguous reconstruction

Example:

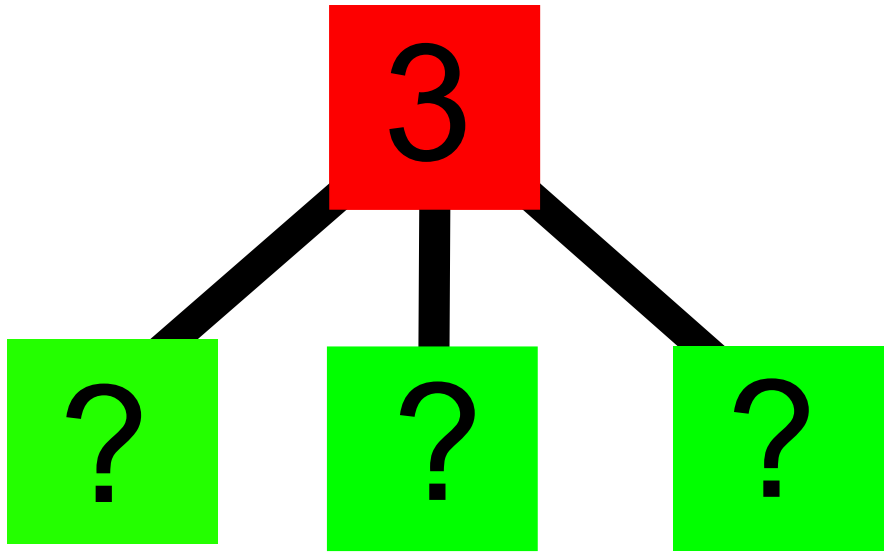
- The following degree sequence is given: {3,2,2,1,3,1,1,1}

{3,

- The first number corresponds to the root node

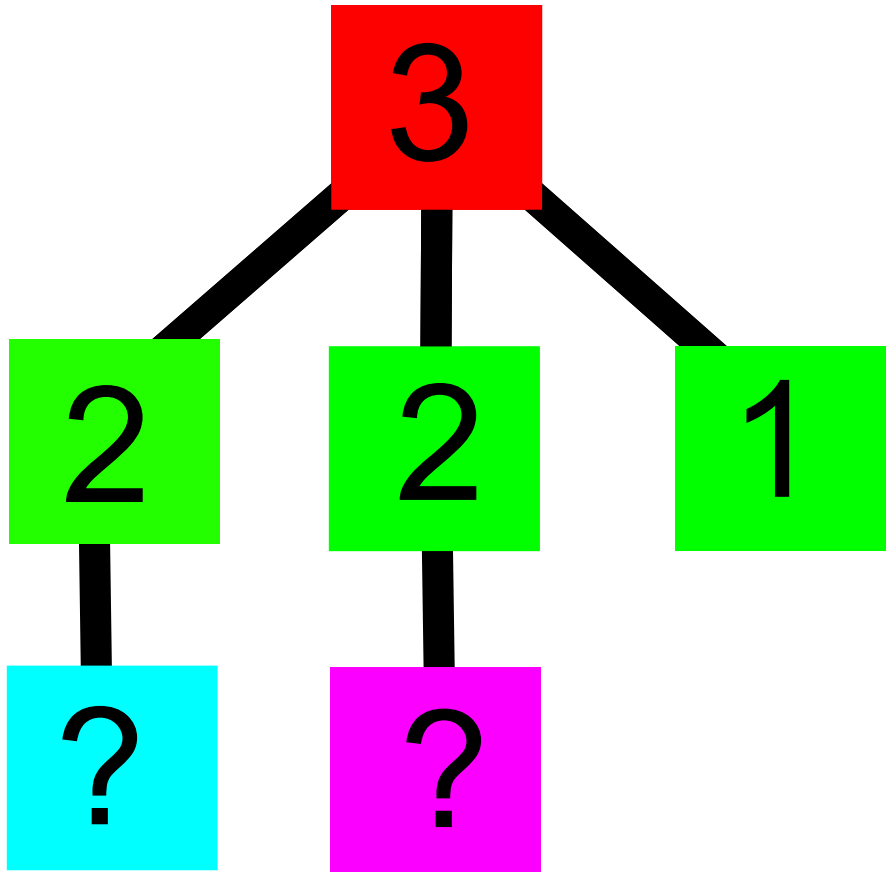
{3,?,?,?

- Degree is 3 so it has 3 children
- These will give the next 3 values in the ODS



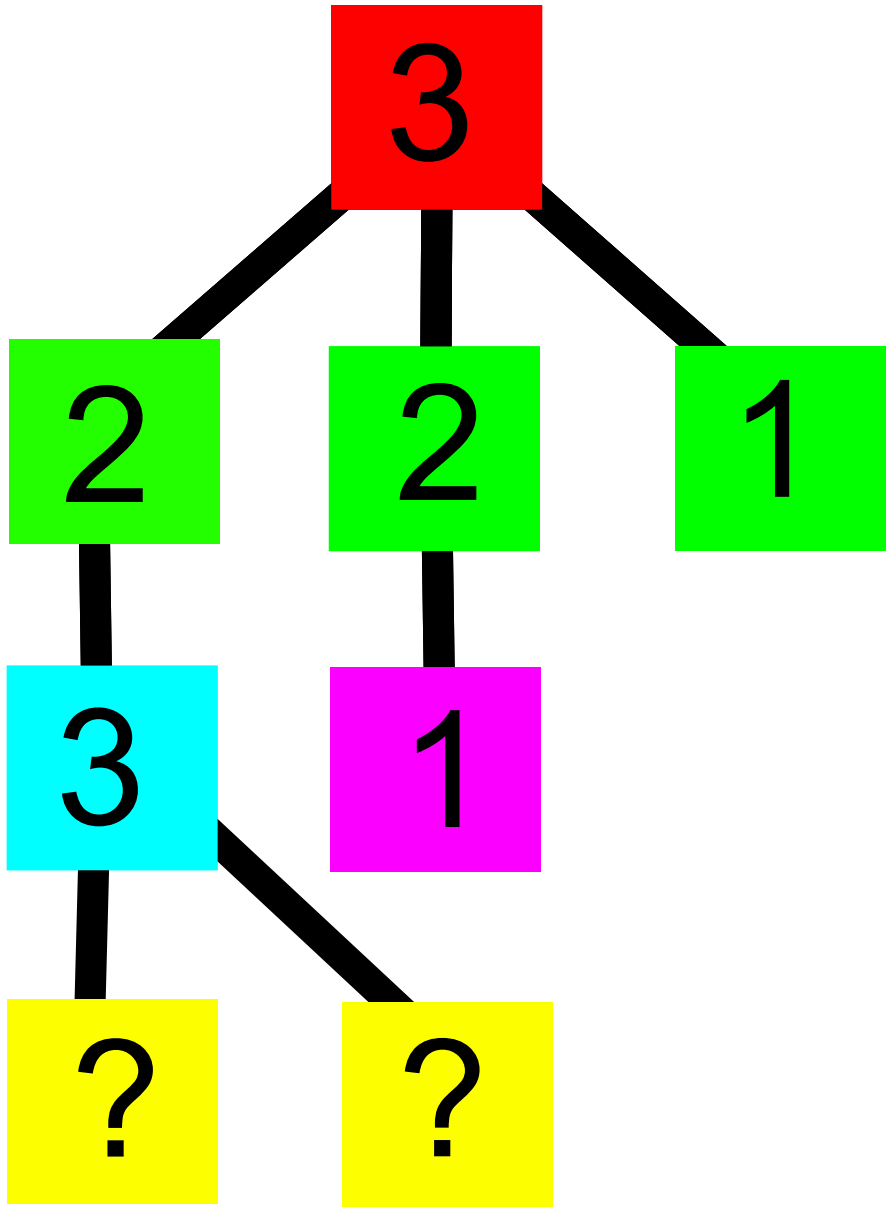
{3,2,2,1,?,?

- Expand from left to right
- Degree of 2: 1 child
- Degree of 1: no child, leaf node



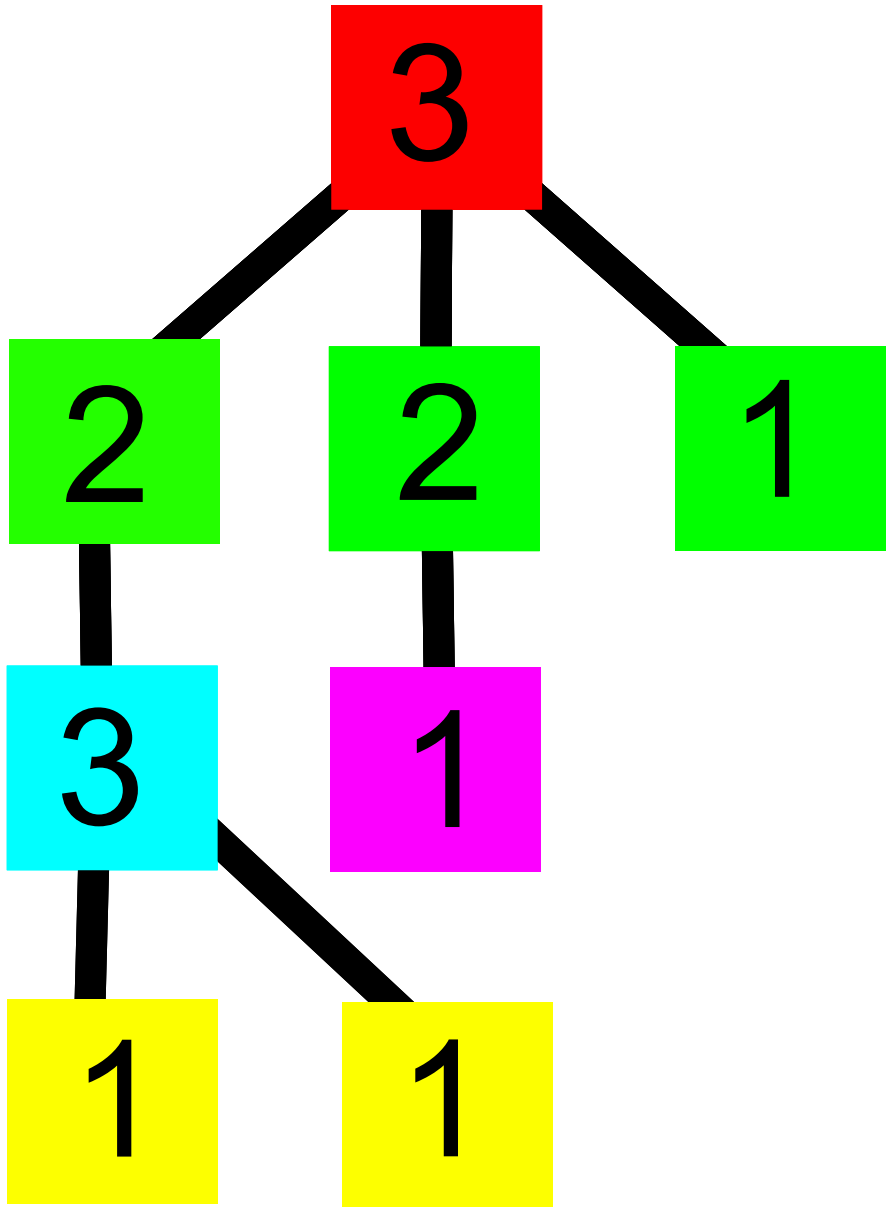
{3,2,2,1,3,1,?,?

- Expand from left to right
- Degree of 3: 2 children
- Degree of 1: no child, leaf node



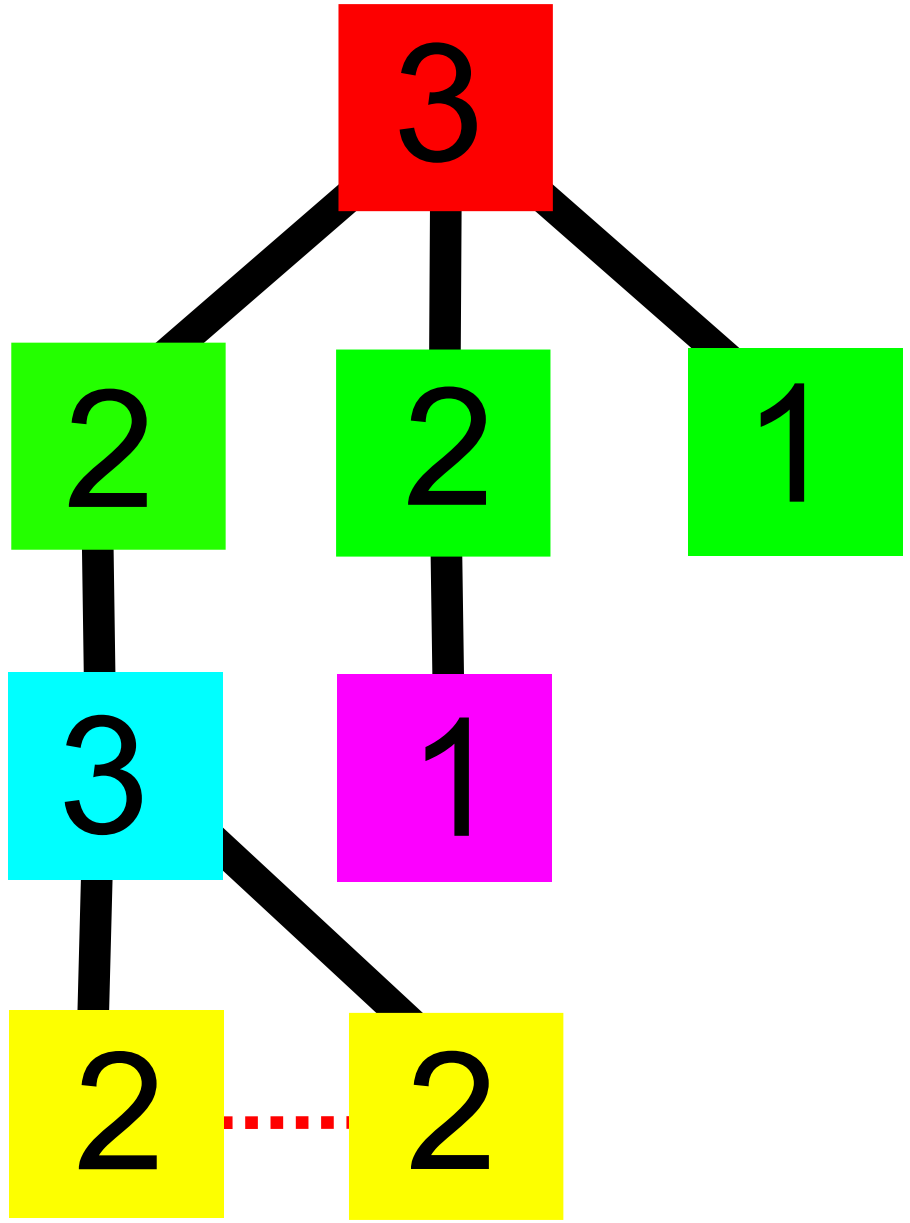
{3,2,2,1,3,1,1,1}

- Expand from left to right
- Degree of 1: no child, leaf node
- All nodes are seen
- All degrees satisfied
- Tree reconstructed
- Top-down, right-to-left



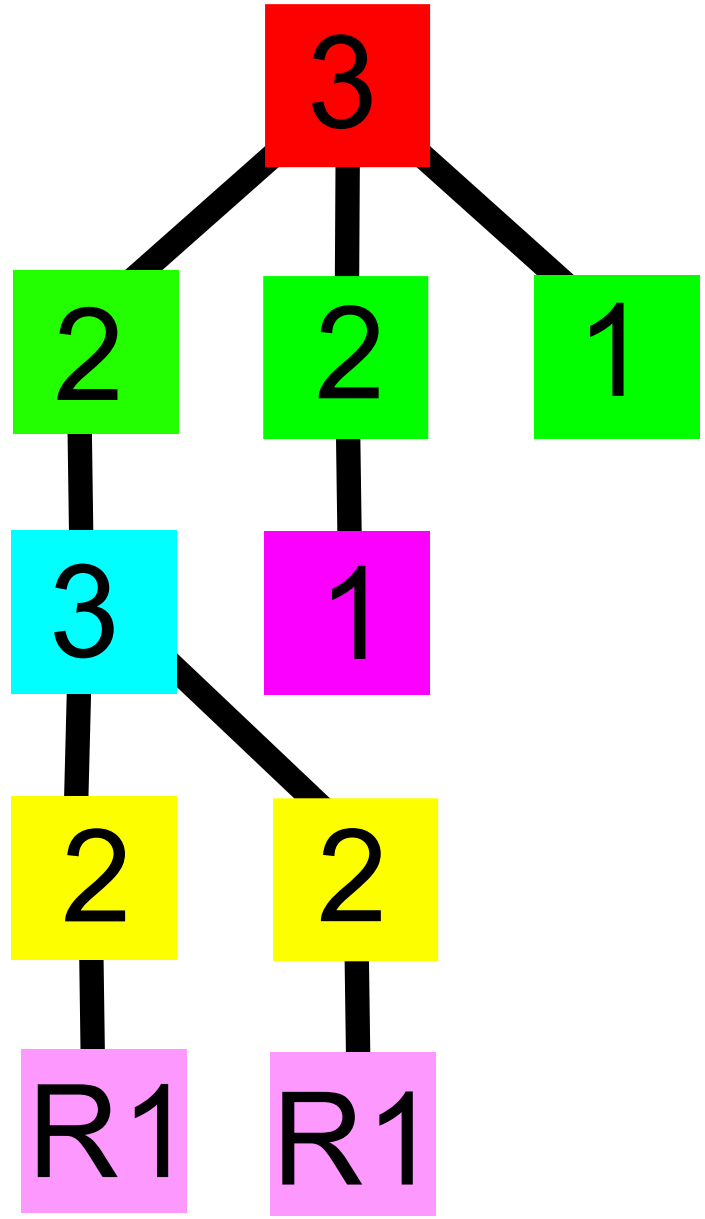
And what about cycles?

- The red dotted bond connected already seen vertices
- Solution needed



Ring closure trick

- Turn ring closures into special leaf nodes!
- SMILES-style linearization trick
- Sequence is now:
- {3,2,2,1,3,1,2,2,R1,R1}



Turning this into a line notation

- For every element in this sequence, add atom information
- Every node except root has a parent it is bonded to
- Per element, giving bond to parent and atom information fully encodes the chemical graph (CG)

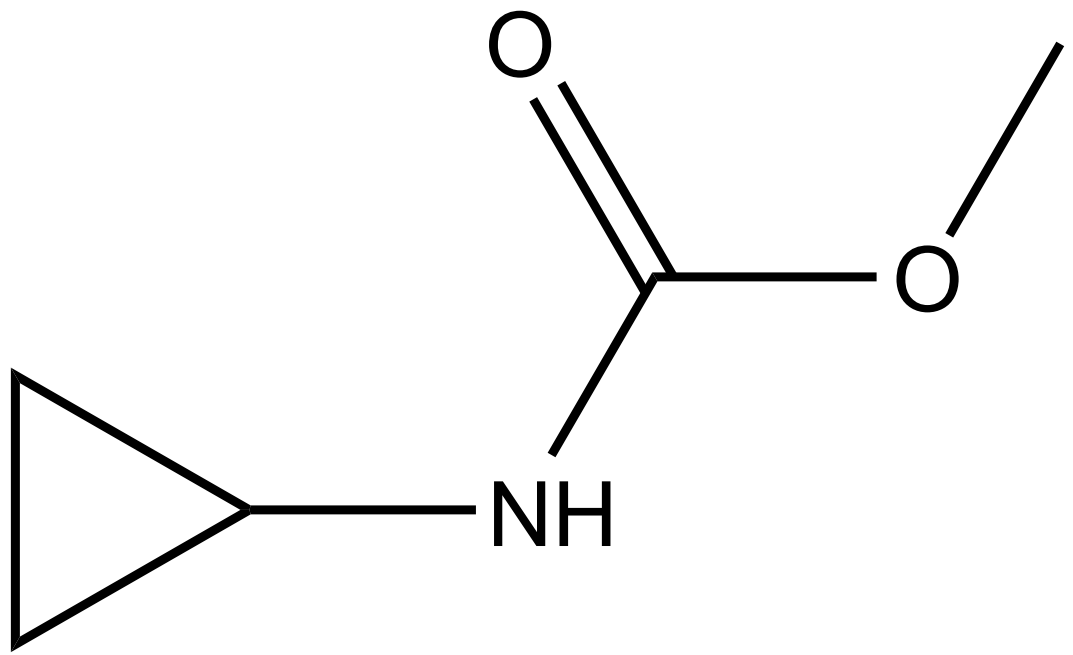
The structure of BLINCS token (SMILESlike)

-[13CH1+1]2 "secondary 13C carbocation"

- Bond between atom and its parent
- Isotope
- Atomic Symbol
- Hydrogen Count
- Formal Charge
- Degree (heavy atom degree)
  - Can be left implicit if 2

Not shown but also impementedl::
Stereo incl R/S E/Z

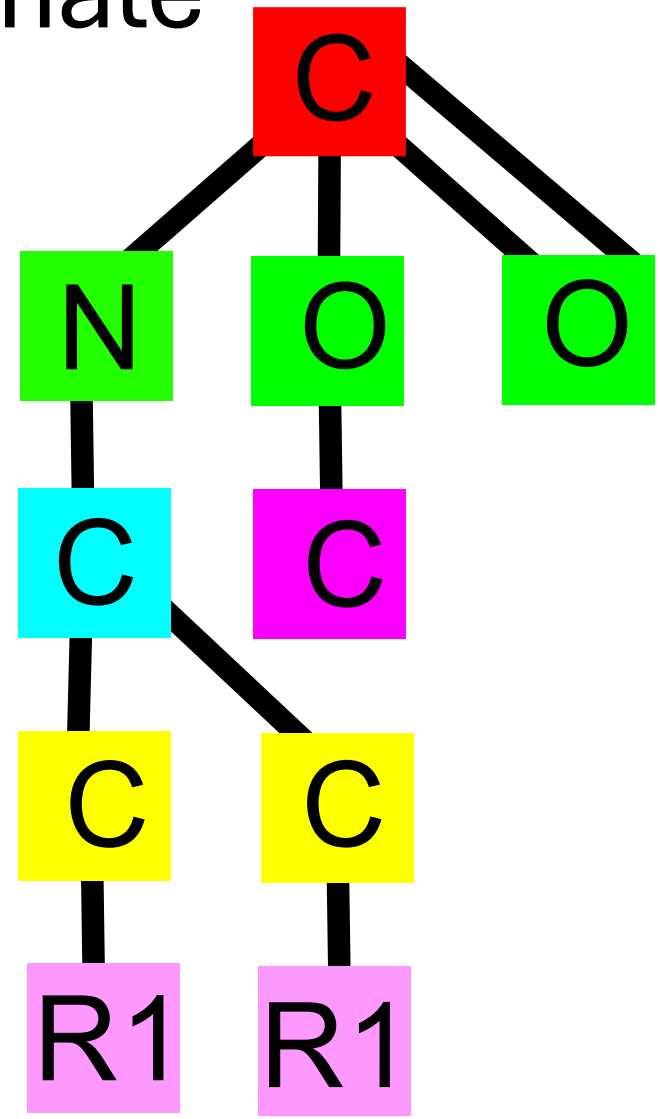
Example: methyl cyclopropylcarbamate



methyl cyclopropylcarbamate

Example: methyl cyclopropylcarbamate

- Degrees and rings:
  {3,2,2,1,3,1,2,2,R1,R1}
- Atoms:
  {C,N,O,=O,C,C,C,C,R1,R1}
- BLINCS:
  C3NO=O1C3C1CC%1%1
- (SMILES: O=C(OC)NC1CC1)



BLINCS implementation

- RDKit (python) based implementation
- Freely available
- Ready to use
- github.com/dehaenw/blincs

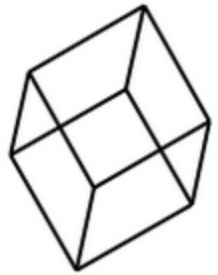
magic BLINCS regex:
r"[=#:-
]?(?:[EZ])?(?:%\d+|\[.*?\]\d*|(?:Cl|Br|[A-Z])\d*|[cnbosp]\d*)"

```python
import blincs

blincs.blincs_to_mol("C1CCCC1")
```

```python
from rdkit import Chem
from rdkit.Chem import rdDepictor
rdDepictor.SetPreferCoordGen(True)
cubane = Chem.MolFromSmiles("C12C3C4C1C5C4C3C25")
print(blincs.mol_to_blincs(cubane))
display(blincs.blincs_to_mol(blincs.mol_to_blincs(cubane)))
```

```
C3C3C3C3C3C3%1C3%2%3%1C3%4%3%5%2%5%4
```



SMILES < BLINCS < SELFIES

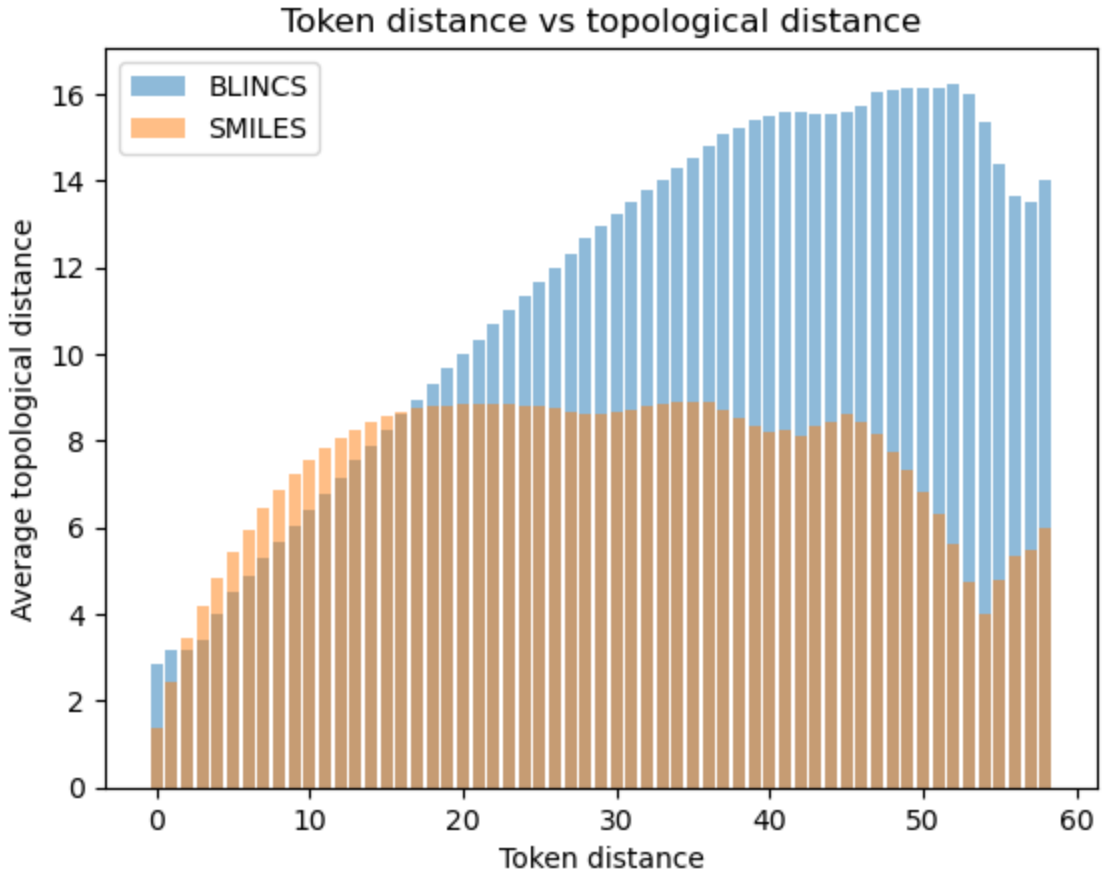
Name	Size (bytes)
SMILES (100K ChEMBL)	5277039
BLINCS (100K ChEMBL)	6380649
SELFIES (100K ChEMBL)	22776193
SMILES (100K ChEMBL) GZIPPED	1482989
BLINCS (100K ChEMBL) GZIPPED	1916519
SELFIES (100K ChEMBL) GZIPPED	2601599

Topological vs Token distance

- Tokens in a SMILES string can be distant when the corresponding atoms are topologically close.
  - Example: n1ccccn1 pyridazine nitrogens are adjacent but 6 chars apart
- BFT should make close atoms be close in the string too
  - Though if there is a lot of branching and the traversal is already deep, that doesn't hold

Topological vs Token distance is better in BLINCS

- But not perfect, due to distant branching

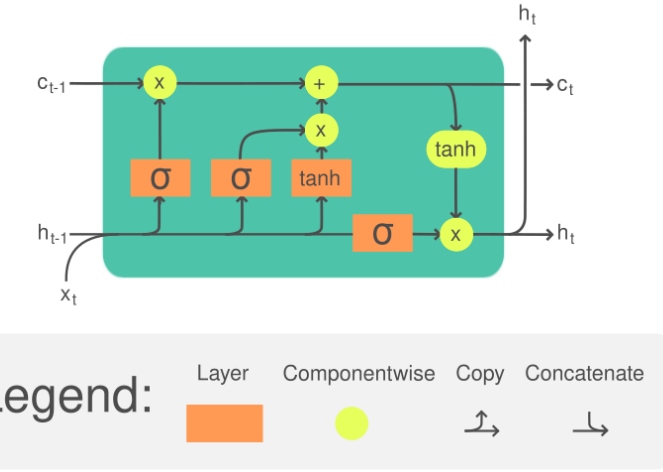


Chemical Language Models

- Just like protein language models (PLMs), a model trained on a large bulk of chemical "sequences" can implicitly learn quite a bit of chemistry
- Despite many-to-one sequence-to-mol problem!
- Typically based on RNNs including LSTM or Transformer

Building an LSTM CLM

- Model built based on ChEMBL35 Molecules (~2.5mil after cleanup)
- SELFIES vs SMILES vs BLINCS (3 models)
- Standard settings:
  - Stacked Vanilla LSTM (3 layers) for Torch
  - Hidden_dimensions = 1024
  - Embedding_size = 128
  - Trained using Adam optimizer, lr 0.001 and cross entropy prediction criterion
  - Early stopping when validation loss doesn't increase for 5 epochs
  - Takes about 1h to train on a gaming GPU



https://commons.wikimedia.org/wiki/File:LSTM_Cell.svg

Generating 500K molecules

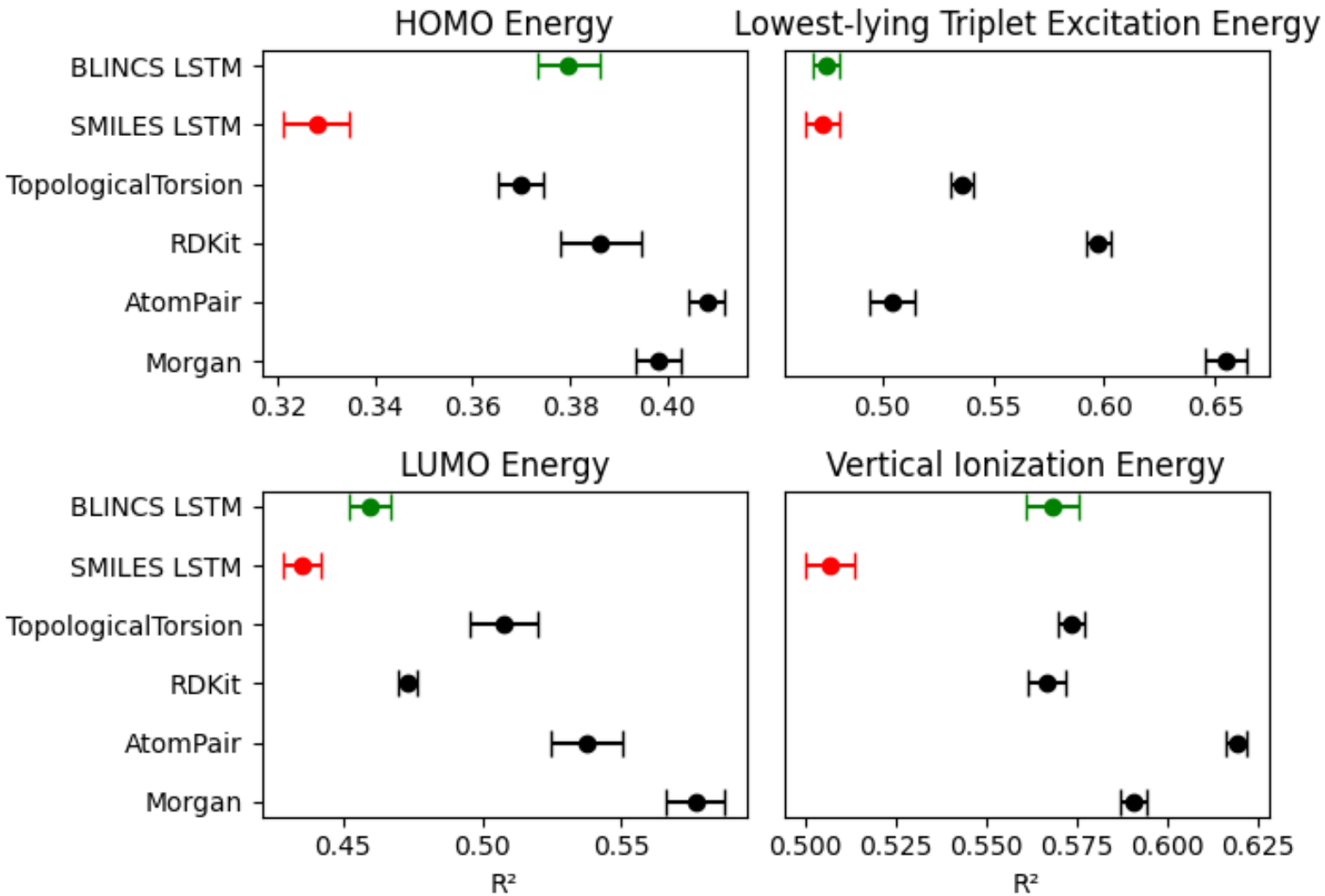
CLM used	Validity (%)	Uniqueness (%)	Frechet Chemnet Distance
SMILES LSTM	91.7	99.2	0.676 +- 0.0262
BLINCS LSTM	90.7	99.1	0.608 +- 0.0238
SELFIES LSTM	89.9*	99.6	1.70 +- 0.0161

*Min one dropped token during parsing regarded as failure

LSTM embeddings for feature prediction

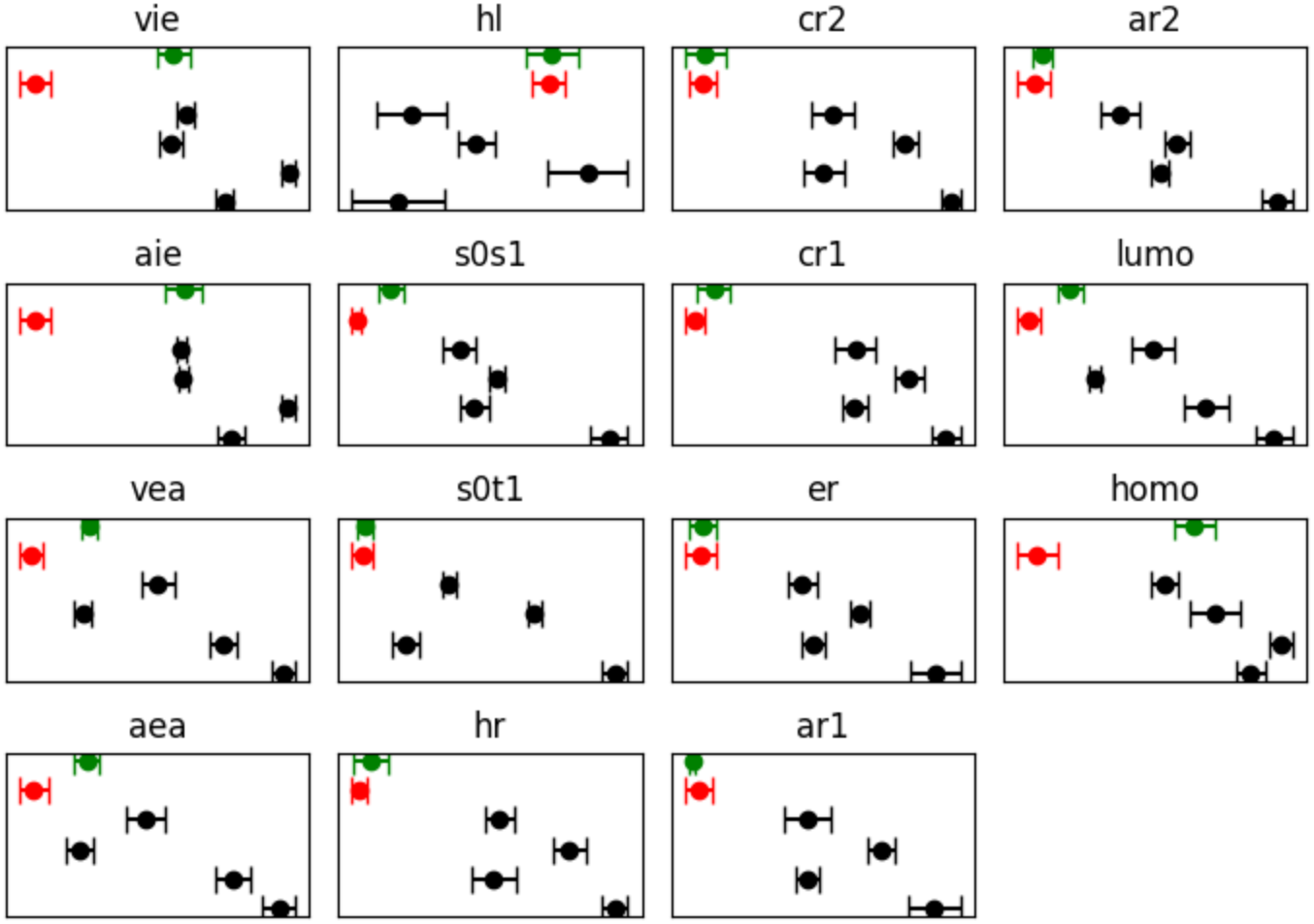
- LSTM embeddings:
  - Cell state from last layer
- Used set: OcelotML_Chromophore_v1
  - QM properties of 25K pi conjugated small molecules
  - Considered a realistic, challenging, consistent dataset
  - Considered less problematic than moleculeNet (by some)
- Captures generalization potential of CLM
  - Very different molecules/properties than ChEMBL
- Compare with standard fingerprints
  - Morgan, AtomPair, TopologicalTorsion, RDKit
- Used estimator: scikitlearn RandomForestRegressor

BLINCS LSTM is on par with SMILES LSTM



On all OcelotML tasks:

- BLINCS LSTM tends to outperform SMILES LSTM
- However, **Morgan remains king**
  - (typical result for learned embeddings)



Conclusion

- BLINCS is a compact line notation comparable with SMILES but with BF graph traversal
- Token distance vs topological distance corresponds better
- BLINCS CLM achieves good performance in distribution learning
- Application in CLM shows comparable to better performance than SMILES in OcelotML Chromophore tasks
  - though no better than baseline FPs
- Just like SMILES, the grammar is easily learnable by ML models
- Prototype Python implementation available and ready to use

Acknowledgement

W.D. was supported by the Ministry of Education, Youth and Sports of the Czech Republic – National Infrastructure for Chemical Biology (CZ-OPENSCREEN, LM2023052) and the project “New Technologies for Translational Research in Pharmaceutical Sciences/NETPHARM”, project ID CZ.02.01.01/00/22_008/0004607, cofunded by the European Union.