RECENT ADVANCES IN CHEMICAL & BIOLOGICAL SEARCH SYSTEMS: EVOLUTION VS. REVOLUTION

Roger Sayle, John Mayfield and Noel O’Boyle

NextMove Software, Cambridge, UK
EVOLUTION VS. REVOLUTION

• Databases and computer power continue to grow at exponential rates.
• A theme in this presentation is the competition between traditional methods, that scale linearly with the size of a problem, and sublinear methods that outperform them.
• At 1M mol/s, searching ChEMBL takes under 2 seconds, searching PubChem takes a minute and a half, and Enamine 2018 takes over 10 minutes.
PART 1:
SMARTS SUBSTRUCTURE SEARCH
Efficient substructure search has a long history in the field of cheminformatics.


The use of a binary fingerprint to pre-screen possible matches improves performance for typical queries.

However, this approach does not affect the worst case and pathological queries require atom-by-atom matching on a significant fraction of the database.
SQC SUBSTRUCTURE BENCHMARKS

https://www.slideshare.net/NextMoveSoftware/substructure-search-faceoff

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
For the SMARTS search “[nH]1ccc2c1ccccc2” of the 6,999,753 compounds in eMolecules 140701, the 120s CPU time is spent on:

- **SMILES Parsing**: 69.8%, 59%
- **Ring Perception**: 32.3%, 27%
- **Aromaticcity**: 11.5%, 10%
- **File I/O**: 3.3%, 3%
- **SMARTS Matching**: 1.1%, 1%

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
PROOF-OF-CONCEPT EXPERIMENT

• Using a large memory server, load the entire database into memory, and achieve/measure SMARTS match only time.
• On the Indole/eMolecules benchmark, this achieved ~6s on a single CPU core (after 128s load time).
• Our C++ molecule footprint required 7.2Gbytes for eMolecules (and 242 Gbytes for Pubchem).
• Perhaps disappointingly still ~2.5s on 8-16 cores
  – Due to iterator allocation contention between threads.
THE "ARTHOR" SEARCH ENGINE

• Implement a substructure search engine uses a compact persistent (pointer-free) binary representation of molecules and a customized SMARTS matcher to operate on it (co-design).

• All 107,404 indole derivatives in 6,653,323 eMolecules structures can be found/counted in 2.9s elapsed time on a single CPU [no FP pre-screen].

• The memory-mapped binary database is about 2Gbytes in size (2,034,444,177 bytes), which averages at 305 bytes per connection table.
SQC SUBSTRUCTURE BENCHMARKS

https://www.slideshare.net/NextMoveSoftware/substructure-search-faceoff

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
SQC SUBSTRUCTURE BENCHMARKS

https://www.slideshare.net/NextMoveSoftware/substructure-search-faceoff

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
SQC SUBSTRUCTURE BENCHMARKS

https://www.slideshare.net/NextMoveSoftware/substructure-search-faceoff

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
SQC SUBSTRUCTURE BENCHMARKS

https://www.slideshare.net/NextMoveSoftware/substructure-search-faceoff

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
SUBSTRUCTURE VIDEO

https://www.youtube.com/watch?v=NmmES_mNF9w
ARTHOR ATDB FUTURE WORK

• Deploy as Oracle/MySQL cartridges, based on the current PostgreSQL cartridge implementation.
• Add support for recursive SMARTS, MDL link atoms, advanced stereochemistry.
• Further optimizations in SMARTS matching.
  – Just-In-Time compilation to x86_64 instructions.
  – More efficient connected components, ring sizes, etc.
PART 2:
TANIMOTO SIMILARITY SEARCH
FP TANIMOTO CALCULATION

- Chemical similarity is traditionally calculated as the Tanimoto coefficient between two binary vectors.
- CUDA code from Olexandr Isayev, UNC

```c
__device__
double similarity(long long *query, long long *target, int data_len) {
    int a = 0, b = 0, c = 0, i;
    for (i = 0; i < data_len; i++) {
        a += __popcll(query[i]);
        b += __popcll(target[i]);
        c += __popcll(query[i] & target[i]);
    }
    return (double) c / (a + b - c);
}
```

https://www.slideshare.net/olexandr1/gpuaccelerated-virtual-screening

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
CHOICE OF FINGERPRINTS

• One of the most significant improvements and differences since Daylight’s era has been the development of circular fingerprints, ECFP4.

• ECFP4 fingerprints perform better on bioactivity benchmarks that path-based fingerprints.

• Alas ECFP4 have different density characteristics to “traditional fingerprints” making a number of classic optimization methods (Baldi bounds) less effective.

• In this work, we consider 1K (and 256 bit) ECFP4 FPs.
TRICK #1: HARDWARE POPCOUNT

• Perhaps the best known approach to achieving high-performance Tanimoto search is use of AMD/Intel’s 32-bit and 64-bit popcount instructions.
• These are provided by the \texttt{__builtin\_popcount} and \texttt{__builtin\_popcountll} builtins in the GNU compilers.
• Historically, there has been a technical interest in using SSE2 and SSE3 instruction sequences, but the widespread availability of hardware popcount makes such approaches unnecessary.
TRICK #2: SORT FPS BY POPCOUNT

- A technique employed by high-performance FP search systems is to sort FPs by their popcount.
- This is traditionally done to enable “Baldi bounds” pruning to achieve “sub-linear” searching.
- The same approach is used by Arthor, but purely as a data storage strategy, allowing the “popcount” for each FP in the database to stored implicitly.
- Arthor can work with unsorted FP files, but search performance is typically several fold slower.
TRICK #3: RECIPROCAL MULTIPLICATION

• Traditionally, calculating a Tanimoto co-efficient requires a (double-precision) floating point division.

• Arthor replaces this with an integer multiplication by using a table of reciprocals.

• Before

  return (double) c / (a + b - c);

• After

  return c * recip_table[c];
TRICK #4: THE SORTING BOTTLENECK

• Analysis of current FP search systems reveals that typically sorting, not searching, is the bottleneck.
• The search phase is $O(N)$, but sorting the results is typically $O(N \cdot \log N)$ for non-trivial numbers of hits.
• ChemSpace hit lists are 200 to 2000 compounds.
• Arthor uses an efficient $O(N)$ two-pass counting sort.
TRICK #5: JUST-IN-TIME COMPILATION

• A powerful optimization based on Just-in-Time compilation techniques is called code specialization.

• Using this technique, searches can take advantage of properties of a chemical similarity query that are not known ahead of time.

• The search engine acts a compiler generating the machine code required to perform the database search and then executes it.
TRICK #5A: SKIP EMPTY WORDS

• The biggest win of specialization is from zero words.

\[
c = \_\_\_\text{popcll}(\text{target}[0] & \text{query}[0]) + \_\_\_\text{popcll}(\text{target}[1] & \text{query}[1]) \\
+ \_\_\_\text{popcll}(\text{target}[2] & \text{query}[2]) + \_\_\_\text{popcll}(\text{target}[3] & \text{query}[3]) \\
+ \_\_\_\text{popcll}(\text{target}[4] & \text{query}[4]) + \_\_\_\text{popcll}(\text{target}[5] & \text{query}[5]) \\
+ \_\_\_\text{popcll}(\text{target}[6] & \text{query}[6]) + \_\_\_\text{popcll}(\text{target}[7] & \text{query}[7]) \\
+ \_\_\_\text{popcll}(\text{target}[8] & \text{query}[8]) + \_\_\_\text{popcll}(\text{target}[9] & \text{query}[9]) \\
+ \_\_\_\text{popcll}(\text{target}[10] & \text{query}[10]) + \_\_\_\text{popcll}(\text{target}[11] & \text{query}[11]) \\
+ \_\_\_\text{popcll}(\text{target}[12] & \text{query}[12]) + \_\_\_\text{popcll}(\text{target}[13] & \text{query}[13]) \\
+ \_\_\_\text{popcll}(\text{target}[14] & \text{query}[14]) + \_\_\_\text{popcll}(\text{target}[15] & \text{query}[15])
\]

• Benzene only has two non-zero query words

\[
c = \_\_\_\text{popcll}(\text{target}[0] & \text{query}[0]) + \_\_\_\text{popcll}(\text{target}[15] & \text{query}[15])
\]

\[
c = \_\_\_\text{popcll}(\text{target}[0] & 272) + \_\_\_\text{popcll}(\text{target}[15] & 1024)
\]
TRICK #5B: WORDS WITH A SINGLE BIT

- Although hardware popcount is very fast (3 cycles on x86_64), it is sometimes possible to do better.
- When $P$ is a constant containing a single bit, i.e. $P = (1 << C)$,
  \[ \text{popcount}(x \& P) = (x >> C) \& 1 \]
  This replaces a popcount with a right shift.
  Additionally $C$ and 1 are smaller constants than $P$.

  \[
c = \_	ext{popcll}(target[0] \& 272) + \_	ext{popcll}(target[15] \& 1024);
\]

  \[
c = \_	ext{popcll}(target[0] \& 272) + ((target[15] >> 10) \& 1);
\]
TRICK #5C: COALESCE MEMORY READS

• Fingerprint data is usually read from memory as aligned 64-bit “unsigned long longs”.
• When only the top or bottom 32-bits are required, these can be read/processed as “unsigned int”.
• On some architectures, consecutive 32-bit words can also be processed as “unaligned” 64-bit data.
• Deciding the set of memory reads and size of each can be optimized via (Viterbi) dynamic programming.
• On GPUs, interleaving of fingerprints is faster still.
TRICK #5D: POPCOUNT COMBINING

• This transformation allows us to reduce the total number of popcounts we need to perform.

• \( \text{popcount}(x \& P) + \text{popcount}(y \& Q) = \text{popcount}((x \& P) + (y \& Q)) \)
  
  — when \((P \& Q) = 0\)
TRICK #5E: GRAPH COLORING

0: 4000400000101110
1: 0002000000082000
2: 0000000010000000
3: 0010001010046000
4: 0000000000010000
5: 9000002001000080
6: 1000000810000940
7: 0000000004800900
8: 0000100400000008
9: 004804000040400
10: 0000000010000000
11: 002000000240020
12: 00000000500a0000
13: 0000800400080002
14: 0000000001002004
TRICK #5E: GRAPH COLORING

- CUDA code for similarity to Aripiprazole

```c
int c = __popcll((target[0] & 0x4000400000101110) +
                  (target[3] & 0x0010001010046000) +
                  (target[5] & 0x9000002001000080) +
                  (target[13] & 0x0000800400080002))
       + __popcll((target[1] & 0x0002000000082000) +
                  (target[6] & 0x1000000810000940) +
                  (target[8] & 0x0000100400000008) +
                  (target[9] & 0x0040804000040400))
       + __popcll((target[7] & 0x0000000004800900) +
                  (target[11] & 0x00200000000240020) +
                  (target[12] & 0x000000000500a000) +
                  (target[14] & 0x0000000001002004))
       + ((target[2] >> 24) & 1)
       + ((target[4] >> 12) & 1)
       + ((target[10] >> 28) & 1);
```

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
TRICK #5E: GRAPH COLORING RESULTS

Graph coloring attempts to combine optimally.

<table>
<thead>
<tr>
<th>Before:</th>
<th>After:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Plan has 1 popcount</td>
<td>1 Plan has 1 popcount</td>
</tr>
<tr>
<td>12 Plan has 2 popcounts</td>
<td>55 Plan has 2 popcounts</td>
</tr>
<tr>
<td>55 Plan has 3 popcounts</td>
<td>363 Plan has 3 popcounts</td>
</tr>
<tr>
<td>154 Plan has 4 popcounts</td>
<td>399 Plan has 4 popcounts</td>
</tr>
<tr>
<td>231 Plan has 5 popcounts</td>
<td>104 Plan has 5 popcounts</td>
</tr>
<tr>
<td>205 Plan has 6 popcounts</td>
<td>28 Plan has 6 popcounts</td>
</tr>
<tr>
<td>161 Plan has 7 popcounts</td>
<td>3 Plan has 7 popcounts</td>
</tr>
<tr>
<td>73 Plan has 8 popcounts</td>
<td></td>
</tr>
<tr>
<td>41 Plan has 9 popcounts</td>
<td></td>
</tr>
<tr>
<td>16 Plan has 10 popcounts</td>
<td></td>
</tr>
<tr>
<td>2 Plan has 11 popcounts</td>
<td></td>
</tr>
<tr>
<td>2 Plan has 12 popcounts</td>
<td></td>
</tr>
</tbody>
</table>

Total: 5477 popcounts  
Total: 3505 popcounts
## INFLUENCE OF ATFP OPTIMIZATIONS

<table>
<thead>
<tr>
<th>Implementation</th>
<th>1 thread M mol/s</th>
<th>4 threads M mol/s</th>
<th>6 threads M mol/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Traditional (transpose)</td>
<td>72</td>
<td>157</td>
<td>158</td>
</tr>
<tr>
<td>CPU Traditional</td>
<td>75</td>
<td>178</td>
<td>192</td>
</tr>
<tr>
<td>CPU Implicit Popcount</td>
<td>97</td>
<td>191</td>
<td>200</td>
</tr>
<tr>
<td>CPU Implicit Popcount (transpose)</td>
<td>100</td>
<td>154</td>
<td>175</td>
</tr>
<tr>
<td>CPU JIT Compilation</td>
<td>121</td>
<td>191</td>
<td>197</td>
</tr>
<tr>
<td>CPU JIT Compilation (transpose)</td>
<td>133</td>
<td>173</td>
<td>180</td>
</tr>
<tr>
<td>GPU JIT</td>
<td>203</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU Traditional</td>
<td>221</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU JIT (transpose)</td>
<td>230</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU Traditional (transpose)</td>
<td>267</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

953 queries over ChEMBL23 database on my Dell Laptop
COMPARISON TO PREVIOUS WORK

Split bars indicate single thread vs. 16 threads for CPU.
FP SIMILARITY VIDEO

https://www.youtube.com/watch?v=NmmES_mNF9w
ARThOR ATFP JIT BACKENDS

• NVidia PTX Assembly Language

```assembly
ld.global.u64    %rd27, [%rd10+56];
and.b64          %rd28, %rd27, 4947953319952;
popc.b64         %r19, %rd28;
add.s32          %r20, %r18, %r19;
ld.global.u32    %r21, [%rd10+68];
shr.u32          %r22, %r21, 3;
and.b32          %r23, %r22, 1;
add.s32          %r24, %r20, %r23;
```

• ARM v6 Assembly Language

```assembly
ldrd    r0, [fp]
lsl     r0, r0, #11
lsr     r7, r0, #31
lsl     r0, r0, #8
add     r7, r7, r0, lsr #31
lsl     r0, r0, #4
add     r7, r7, r0, lsr #31
lsl     r1, r1, #1
add     r7, r7, r1, lsr #31
lsl     r1, r1, #16
add     r7, r7, r1, lsr #31
```
ARThor ATFP Future Work

• Support for multiple GPU cards [federated search].
• Direct generation of NVidia SASL via cubin binaries for Volta, Pascal, Maxwell and Kepler architectures.
• Improved statistical significance scoring & Tversky.
• Optimizations incorporated in the GNU compilers:

  2018-05-24  Roger Sayle  <roger@nextmovesoftware.com>

  * fold-const.c (tree_nonzero_bits): New function.
  * fold-const.h (tree_nonzero_bits): Likewise.
  * match.pd (POPCOUNT): New patterns to fold BUILTIN_POPCOUNT and friends.  POPCOUNT(x&1) => x&1, POPCOUNT(x)==0 => x==0, etc.
PART 3:
PROTEIN SEQUENCE SEARCH
PROTEIN SEQUENCE NAMING

• In cheminformatics, InChI and canonical SMILES can be used to semantically link database/tables/graphs.
• Traditionally in bioinformatics, accession numbers (such as SwissProt) have been used for proteins.
• Mature proteins however require derived names.
  – PDB 1CRN (crambin) is [L25I]P01542 CRAM_CRAAB
  – PDB 4ZAU is Gly-Ala-Met-P00533 (696-1022) EGFR_HUMAN
  – PDB 1UA2 is des-(32-43,145)-P50613 (13-311) CDK7_HUMAN
  – PDB 5NN9 is [A187D]P03472 (83-470) NRAM_I75A5
  – PDB 1HXB is P04585 (489-587) POL_HV1H2
  – PDB 1JTE is [Y181C]P04585 (588-1147) POL_HV1H2
Algorithm: Longest Common Prefix

• Traditionally, longest common subsequence search uses a linear scaling algorithm (e.g. blast, fasta, FSM).

• Sequence identify and longest common prefix can be solved by binary search of an alphabetically sorted sequence database.
  – APPLE
  – BANANA
  – PEAR
ALGORITHM: LONGEST COMMON SUBSTRING

- Suffix arrays efficiently index every substring:
  - A 6@BANANA
  - ANA 4@BANANA
  - ANANA 2@BANANA
  - APPLE 1@APPLE
  - AR 3@PEAR
  - BANANA 1@BANANA
  - E 5@APPLE
  - EAR 2@PEAR
  - LE 4@APPLE
  - NA 5@BANANA
  - NANA 3@BANANA
  - PEAR 1@PEAR
  - PLE 3@APPLE
  - PPLE 2@APPLE
  - R 4@PEAR
Part 4:
Graph Edit Distance Search
FIGHTING BIG DATA WITH BIGGER DATA

• The same technique used to speed up longest common subsequence and string edit distance search in bioinformatics can also be applied to maximum common substructure and graph edit distance search in cheminformatics.

• Here we describe the use of a sublinear-scaling search method over a database that is approximately constant (perhaps 1K-1M) times larger.

• As data set sizes increase, these approaches make traditional methods increasingly uncompetitive.
SMALLWORLD CHEMICAL SPACE

Graph search (GED) of 68 billion subgraphs vs. 340 million molecules.
## Counting Molecular Subgraphs

<table>
<thead>
<tr>
<th>Name</th>
<th>Atoms</th>
<th>MW</th>
<th>Subgraphs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benzene</td>
<td>6</td>
<td>78</td>
<td>7</td>
</tr>
<tr>
<td>Cubane</td>
<td>8</td>
<td>104</td>
<td>64</td>
</tr>
<tr>
<td>Ferrocene</td>
<td>11</td>
<td>186</td>
<td>3,154</td>
</tr>
<tr>
<td>Aspirin</td>
<td>13</td>
<td>180</td>
<td>127</td>
</tr>
<tr>
<td>Dodecahedrane</td>
<td>20</td>
<td>260</td>
<td>440,473</td>
</tr>
<tr>
<td>Ranitidine</td>
<td>21</td>
<td>314</td>
<td>436</td>
</tr>
<tr>
<td>Clopidrogel</td>
<td>21</td>
<td>322</td>
<td>10,071</td>
</tr>
<tr>
<td>Morphine</td>
<td>21</td>
<td>285</td>
<td>176,541</td>
</tr>
<tr>
<td>Amlodipine</td>
<td>28</td>
<td>409</td>
<td>58,139</td>
</tr>
<tr>
<td>Lisinopril</td>
<td>29</td>
<td>405</td>
<td>24,619</td>
</tr>
<tr>
<td>Gefitinib</td>
<td>31</td>
<td>447</td>
<td>190,901</td>
</tr>
<tr>
<td>Atorvastatin</td>
<td>41</td>
<td>559</td>
<td>3,638,523</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>≤ Bond Count</th>
<th>%PubChem</th>
</tr>
</thead>
<tbody>
<tr>
<td>≤ 20 bonds</td>
<td>14%</td>
</tr>
<tr>
<td>≤ 25 bonds</td>
<td>30%</td>
</tr>
<tr>
<td>≤ 30 bonds</td>
<td>55%</td>
</tr>
<tr>
<td>≤ 35 bonds</td>
<td>77%</td>
</tr>
<tr>
<td>≤ 40 bonds</td>
<td>89%</td>
</tr>
<tr>
<td>≤ 45 bonds</td>
<td>93%</td>
</tr>
<tr>
<td>≤ 50 bonds</td>
<td>95%</td>
</tr>
<tr>
<td>≤ 55 bonds</td>
<td>97%</td>
</tr>
<tr>
<td>≤ 60 bonds</td>
<td>98%</td>
</tr>
<tr>
<td>≤ 65 bonds</td>
<td>98%</td>
</tr>
<tr>
<td>≤ 70 bonds</td>
<td>99%</td>
</tr>
</tbody>
</table>

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
SMALLWORLD CHEMICAL SPACE

Graph search (GED) of 68 billion subgraphs vs. 340 million molecules.
Graph Edit Distance (GED) is the minimum number of edit operations required to transform one graph into another.

- https://en.wikipedia.org/wiki/Graph_edit_distance

Edit operations consist of insertions, deletions and substitutions of nodes and edges (atoms and bonds).
**TOPOLOGICAL EDIT/EDGE TYPES**

- **tup**: add a terminal bond
- **tdn**: remove a terminal bond
- **rup**: form a ring bond
- **rdn**: break a ring bond
- **lup**: insert a (degree 2) linker node
- **ldn**: remove a (degree 2) linker node

---

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
SmallWorld lattice: Bold circles denote indexed molecules, thin circles represent virtual subgraphs.
The solid circle denotes a query structure which may be either an indexed molecule or a virtual subgraph.
The first iteration of the search adds the neighbors of the query to the "search wavefront".
SMALLWORLD SEARCH

Each subsequent iteration propagates the wavefront by considering the unvisited neighbors of the wavefront.
At each iteration, “hits” are reported as the set of indexed molecules that are members of the wavefront.
SMALLWORLD SEARCH

The search terminates once sufficient indexed neighbors have been found (or a suitable iteration limit is reached).
SMALLWORLD SEARCH
SMALLWORLD SEARCH
SMALLWORLD

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018

https://www.youtube.com/watch?v=hZ4QyQSeSWg
EXAMPLE EDIT OPERATIONS

Ticlodipine

Penicillin G

Clopidogrel

Amoxicillin

11th ICCS, Noordwijkerhout, The Netherlands, Wednesday 30th May 2018
EXAMPLE EDIT OPERATIONS

Sildenafil (Viagra)
Vardenafil (Levitra)
Sumatriptan (Imitrex)
Zolmitriptan (Zomig)
CURRENT DATABASE STATISTICS

• As of October 2017, the SmallWorld index has
• 68,921,678,269 nodes (~69B or \(2^{36}\) nodes)
• 258,787,077,793 edges (~259B or \(2^{38}\) edges)
  – 128,762,041,180 ring edges.
  – 95,709,763,280 terminal edges
  – 34,315,273,333 linker edges.
• Average degree (fan-out) of node: ~7.5
• 8.22B acyclic nodes, 7.12B have a single ring.
• Runtime index requires 5TB of disk space.
Total Distance: 8   Total Topological Distance: 6
CLASSIC EX SCIENTIA EXAMPLE
SUMMARY

• Algorithmic improvements and Moore’s law advances in hardware should allow traditional cheminformatics and bioinformatics search techniques to be applied for the time being, but ultimately next generation approaches will be required to handle multi-billion compound databases.
ACKNOWLEDGEMENTS

• Andrew Dalke, Dalke Scientific Software.
• Yurii Moroz, Enamine and ChemSpace.
• Evan Bolton, PubChem Group, NCBI.
• Pat Walters, Relay Therapeutics.
• Andrew Grant, AstraZeneca.
• Darren Green, GSK.
ADVANTAGES OVER FINGERPRINTS

• FP similarity based on “local” substructures.
• FP saturation of features/Chemical Space.
  – Many peptides/proteins/nucleic acids have identical FPs.
  – For alkanes, C16 should be more similar to C18 than C20.
  – Identical FPs in Chemistry Toolkit Rosetta benchmark.
  – PubChem “similar compounds” uses 90% threshold.
• FPs make no distinction atom type changes.
  – Chlorine to Bromine more conservative than HBD to HBA.
  – Tautomers/protonation states often have low similarity.
  – FPs are more sensitive to Normalization/Standardization.
• Stereochemistry is poorly handled by FPs.
  – Either not represented or isomers have low similarity.
Graph Database Fabrication

- The “raw” source representation of SmallWorld is 28.7 TB of data, one ASCII line (of two SMILES) for each edge, i.e. 259 billion text lines.
- Hypothetically, these 259B triples could be loaded into a database such as Oracle, Virtuoso or Neo4j.
- Instead, we “compile” this graph database down to a 5TB form that is very efficiently searched at run-time.
- This 5TB can be delivered to customers on a £150 external USB disk (like a subscription service).
DATABASE PARTITIONING

• Instead of treating the database as a single monolithic entity, the nodes are partitioned by their atom, bond and ring counts.

• This results in 2406 partitions, named $B_x R_y$ where $x$ is the number of bonds, $y$ is the number of rings.

• Each edge links vertices in neighboring partitions.
  – A tdn edge from $B_x R_y$ leads to $B_{x-1} R_y$, tup to $B_{x+1} R_y$.
  – A rdn edge from $B_x R_y$ leads to $B_{x-1} R_{y-1}$, rup to $B_{x+1} R_{y+1}$.
  – A ldn edge from $B_x R_y$ leads to $B_{x-1} R_y$, lup to $B_{x+1} R_y$. 