



# SUBSTRUCTURE SEARCHING FACE-OFF

John May and Roger Sayle  
NextMove Software  
Cambridge, UK



# MOLECULAR IDENTITY

*is foo the same as bar?*

# SUBSTRUCTURE

*is foo part of bar?*

# CHEMICAL SIMILARITY

*how much is foo like bar?*



# SUBSTRUCTURE SEARCH

Some of the **many** uses:

- Fragment-based lead discovery
- Murcko Scaffolds
- Matched pairs
- Functional groups, atom types
- Similarity Keys (MACCS/CACTVS)
- Toxicology - Tim Allen's MIE models

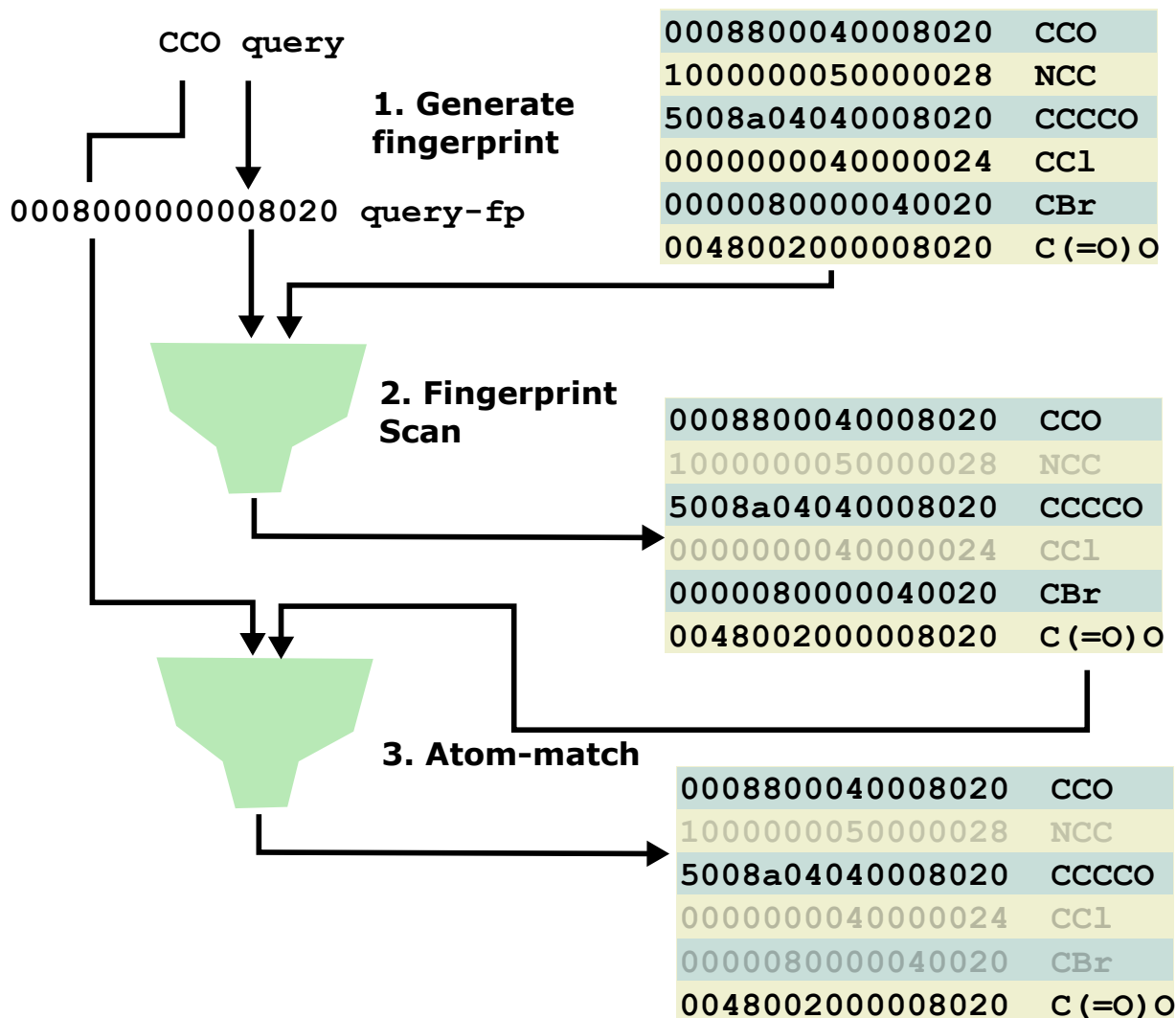
Substructure searches are **infrequent**\*

...they can be **slow** - correlation/causation?

\*~10x more *unique* similarity queries in Structure Query Collection



# ANATOMY OF A SUBSEARCH



## Variables

- fp type
  - none
  - linear
  - sub-linear
- mol format
  - line-notation
  - binary
- atom-match
  - VF2
  - Ullmann
- storage
  - flatfile
  - RDMS

\*CBr is actually filtered with this fingerprint but left to show the FP is generally not perfect (precision<1)



# WHY CAN IT FEEL SLOW?

## Identity and Similarity Search

different queries, similar amount of work

## Substructure Search

different queries, potentially huge difference in work

## Rough est. for ~10 mil compounds

- Identity, index canonical form - b-tree:  $\leq 1$  ms
- Similarity, index fingerprint - linear scan:  $\leq 100$  ms
- Substructure, index fingerprint + structure
  - linear FP scan, atom-based match:  $\geq 100$  ms
  - sub-linear FP scan, atom-based match:  $\geq 5$  ms



# STRATEGIES?

Parallelisation

resource contention from multiple users

Limit query count (first 10)

must decide which ones are first

Limit time (60 s in eMolecules/SureChEMBL)

hits depend on hardware/workload

Cache or filter pathological queries

how to predict?

Limit screen count (fastsearch)

may get no hits?



# EXISTING BENCHMARKS

Several tools publish isolated performance figures and should be commended.

*“what performance can you expect”*

Alas **different** hardware, queries, and target dataset limit meaningful conclusions and comparisons.

Using the **same** hardware, queries, and target dataset we present the **execution efficiency** (how fast). **Retrieval efficiency** (hits) will be touched upon.



OUR BENCHMARK:

COUNT TOTAL  
NUMBER OF HITS



# TARGET DATABASE

Needed moderate size collection

ChEMBL ~1.5 mil, too easy, fits easily in a memory cache

PubChem Compound ~70 mil, too large for many tools

eMolecules 6,996,230 compounds

performed minimal cleanup

eMol IDs will be made available



# QUERIES

## Structure Query Collection (SQC)

3,488 **BindingDB\_substructure** queries

3,342 connected, 121 fragments

User generated

Includes *pathological* queries

Excluded fragments - some consider benzene as bad:

c1ccccc1.c1ccccc1 slow

c1ccccc1.c1ccccc1.c1ccccc1 very slow

c1ccccc1.c1ccccc1.c1ccccc1.c1ccccc1.c1ccccc1 ouch

Andrew Dalke's BitBucket Project - <https://bitbucket.org/dalke/sqc>



# QUERIES

Duplicates due to **aromaticity** or **hydrogens**

[H]C1=CN=CN=C1[H] pyrimidine

[H]C1=CN=CN=C1 pyrimidine

C1=CN=CN=C1 pyrimidine

C1=CC=CC=C1 benzene

c1ccccc1 benzene

3,140/3,342 unique - but used the non-unique set:

- Hydrogens change semantics of match
- Queries run unmodified but standardised\* to **queries-clean** when needed, tools that expect “SMARTS”

\*aromatised, suppressed-H, atom-stereo removed



# TOOLS EVALUATED

KEY	TYPE	CHEMISTRY	STORAGE	FORMAT
<u>arthor</u>	standalone	NextMove	Flatfile	Serialised
bingo-pgsql	cartridge	Indigo	PostgreSQL	Serialised
bingo-orcl	cartridge	Indigo	Oracle	Serialised
bingo-nosql	standalone	Indigo	Flatfile	Serialised
<u>fastsearch</u>	standalone	Open Babel	Flatfile	SMILES <sup>1</sup>
jcart	standalone/cartridge	ChemAxon	Oracle	Serialised
jcart-st (1 thread)	standalone/cartridge	ChemAxon	Oracle	Serialised
mychem	cartridge	Open Babel	MySQL	Serialised
pgchem	cartridge	Open Babel	PostgreSQL	Serialised
orchem	cartridge	CDK	Oracle	Serialised <sup>2</sup>
rdcart	cartridge	RDKit	PostgreSQL	Serialised
rdluc	standalone	RDKit	Lucene	SMILES
<u>rdsmigrep</u>	standalone	RDKit	SMILES	SMILES
tripod-ss	standalone	ChemAxon	Oracle Cache	SMILES <sup>3</sup>

1) FastSearch format depends on input. 2) OrChem serialises as a string rather than binary. 3) Tripod's Search Search uses molfiles by default. Underlined: queries-clean used

# MORE CARTRIDGES

NAME	TYPE	CHEMISTRY	STORAGE
Pinpoint	Cartridge	Dotmatics	Oracle
Direct	Cartridge	BIOVIA/Accelrys/MDL	Oracle
Accord	Cartridge	BIOVIA/Accelrys	Oracle
ABCD	Cartridge	J&J	Oracle
DayCart	Cartridge	Daylight	Oracle/Postgres
Torus	Cartridge	Digital Chem	Oracle
OpenEye Cartridge	Cartridge	OpenEye	Oracle
MolCart	Cartridge	ICM	MySQL
MolSQL	Cartridge	Scilligence	SQL Server
Chord	Cartridge	OpenEye	PostgreSQL
OpenChord	Cartridge	Open Babel / RDKit	PostgreSQL
IC Cartridge	Cartridge	InfoChem	Oracle
AUSPYX	Cartridge	Tripos	Oracle
Oracle Cartridge	Cartridge	PerkinElemer	Oracle

Some not externally available (ABCD) or being retired (Accord)

# MORE STANDALONES

NAME	TYPE	CHEMISTRY	STORAGE
ChemFP	standalone	Multiple	FPS/BFPS in-memory
Similar	standalone	?	MongoDB
Data Warrior	standalone	Acetlion	In-memory
Ambit - OpenTox	standalone	CDK/Ambit	MySQL
MolecularLucene	standalone	CDK	Lucene
Wikipedia CSE	standalone	Acetlion (JS)	In-browser-memory
MOLDB6	standalone	checkmol/matchmol	MySQL
ChemSearch <sup>1</sup>	SQL	SQL	Oracle/PostgreSQL
SQLMOL	SQL	SQL	MS SQL

ChemFP, Similar, MolecularLucene

- currently **similarity** only

MOLDB6

- index time for eMol est. at **22** days... we started last week

1: Golovin A and Henrick K. Chemical Substructure Search in SQL. *JCIM*. 2008. 49(1)

# CHANGES FROM "OUT OF THE BOX"

## MyChem and RDKit Lucene

- fingerprint screen workaround
- OB FP2 and Avalon FP, **no bits  $\neq$  no hits**

## Tripod Search Server

- Load from **SMILES** instead of molfile
- Avoid aromaticity recalculation
- Used larger cache size



# MEASUREMENT IS NOT PROPHECY<sup>1</sup>

Large number of variables

Time taken for this benchmark, with these tool versions, on a machine, with this load. Tools were setup as per user manuals (shared memory etc).

A “*normal*” desktop, £££ less than iPhone 6+.

CentOS 7

Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz

16 GB RAM

“*Trust has no place in science*” - NextMove Software will make inputs available for others to test

1: <http://benchmarksgame.alioth.debian.org/>



3342 QUERIES

6996230 TARGETS

$\leq 23,381,400,660$

MATCHES LATER



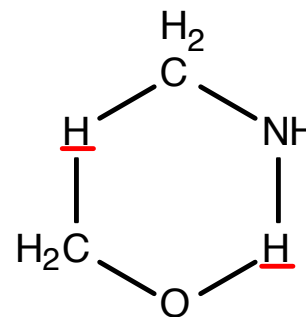
# EXCLUDED QUERIES

**19** queries skipped in one or more tools

3,323/3,342 that were executed by all

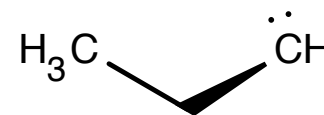
**2** Non-terminal hydrogens

C1N[H]OC[H]1



**1** Invalid stereochemistry

CC[C@@H]



**10** Atom classes

C1C[N:1]CNC1

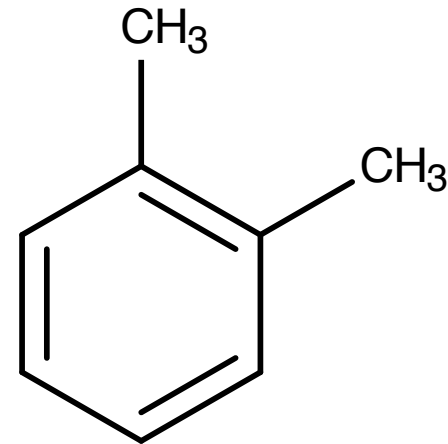


**6** Bond stereo not-implemented (should have cleaned)



# RETRIEVAL DIFFERENCES

*“We demand rigidly defined areas of doubt and uncertainty!”*



o-xylene



# RETRIEVAL DIFFERENCES

*“We demand rigidly defined areas of doubt and uncertainty!”*

Somewhere between **440,901** and **529,386**

rdcart 500,126

rdluc 500,053

bingo (pgsql/orcl) 500,865

bingo (nosql) 498,427

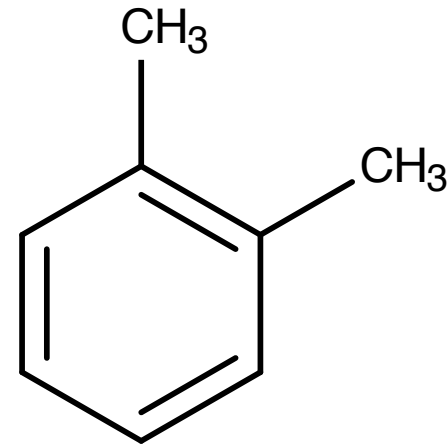
tripod-ss 509,655

jcart 444,541

fastsearch 440,901

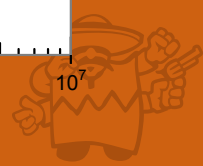
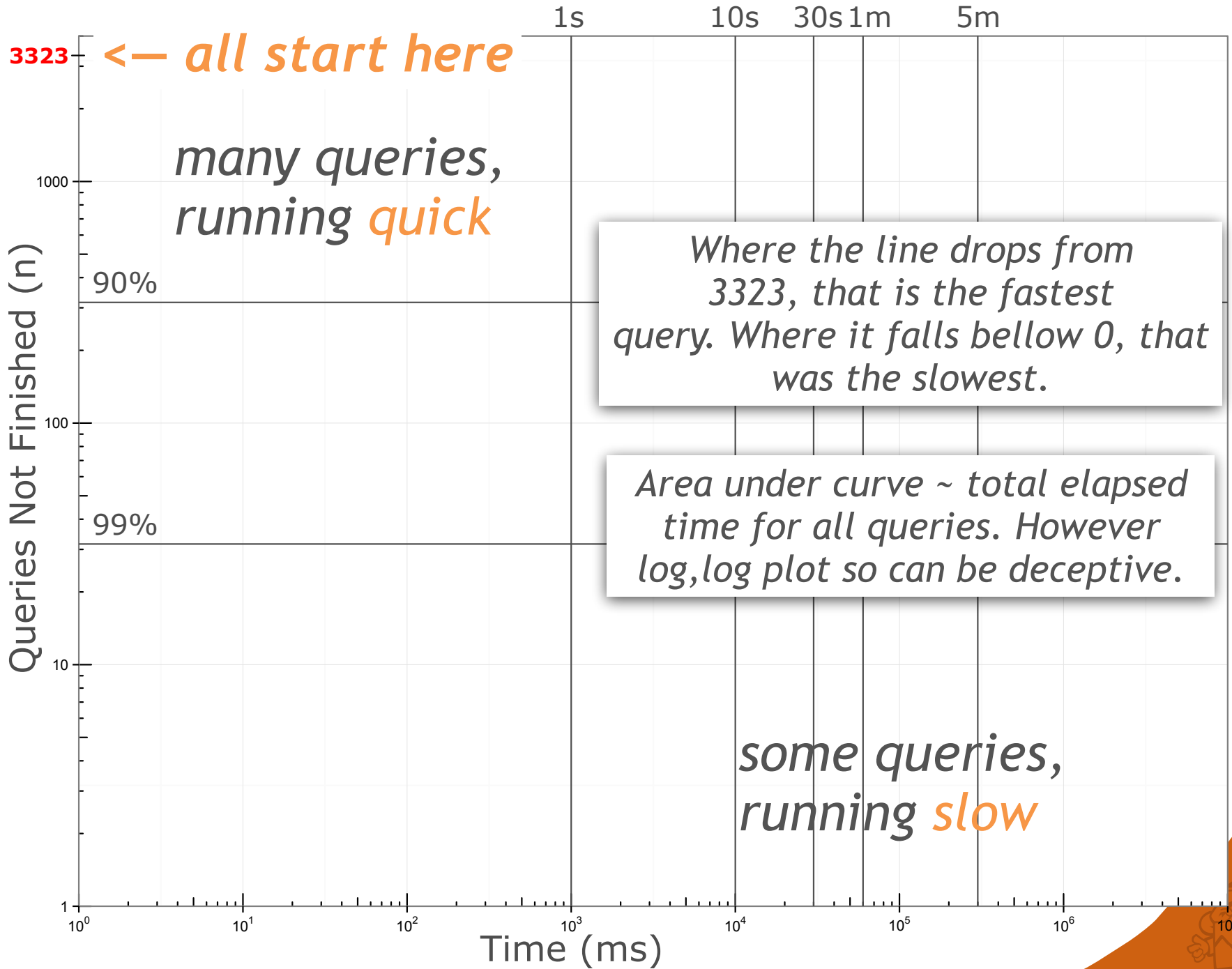
orchem 529,386

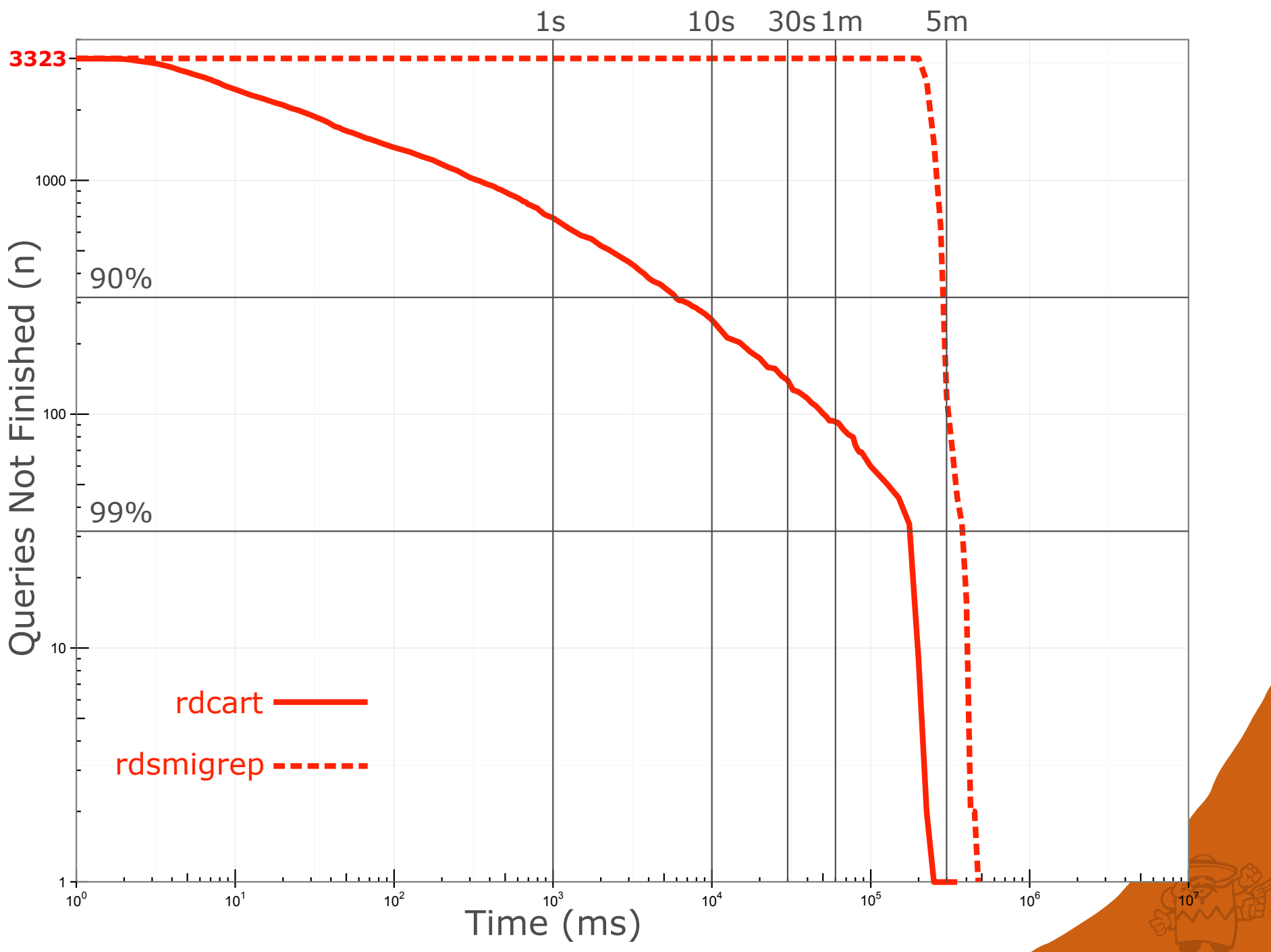
arthor 444,553

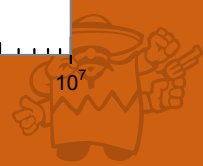
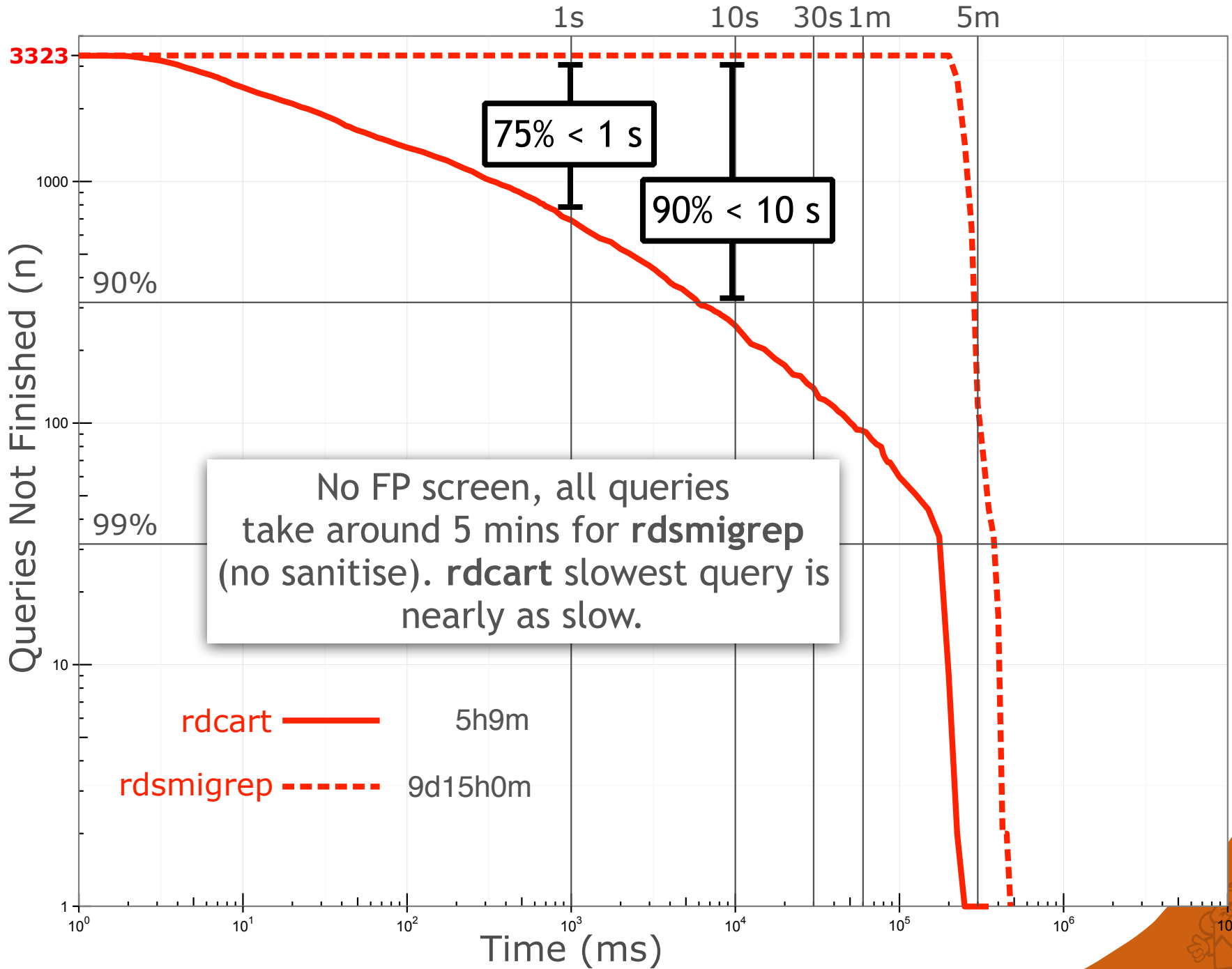


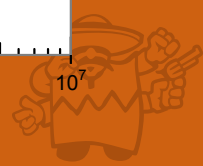
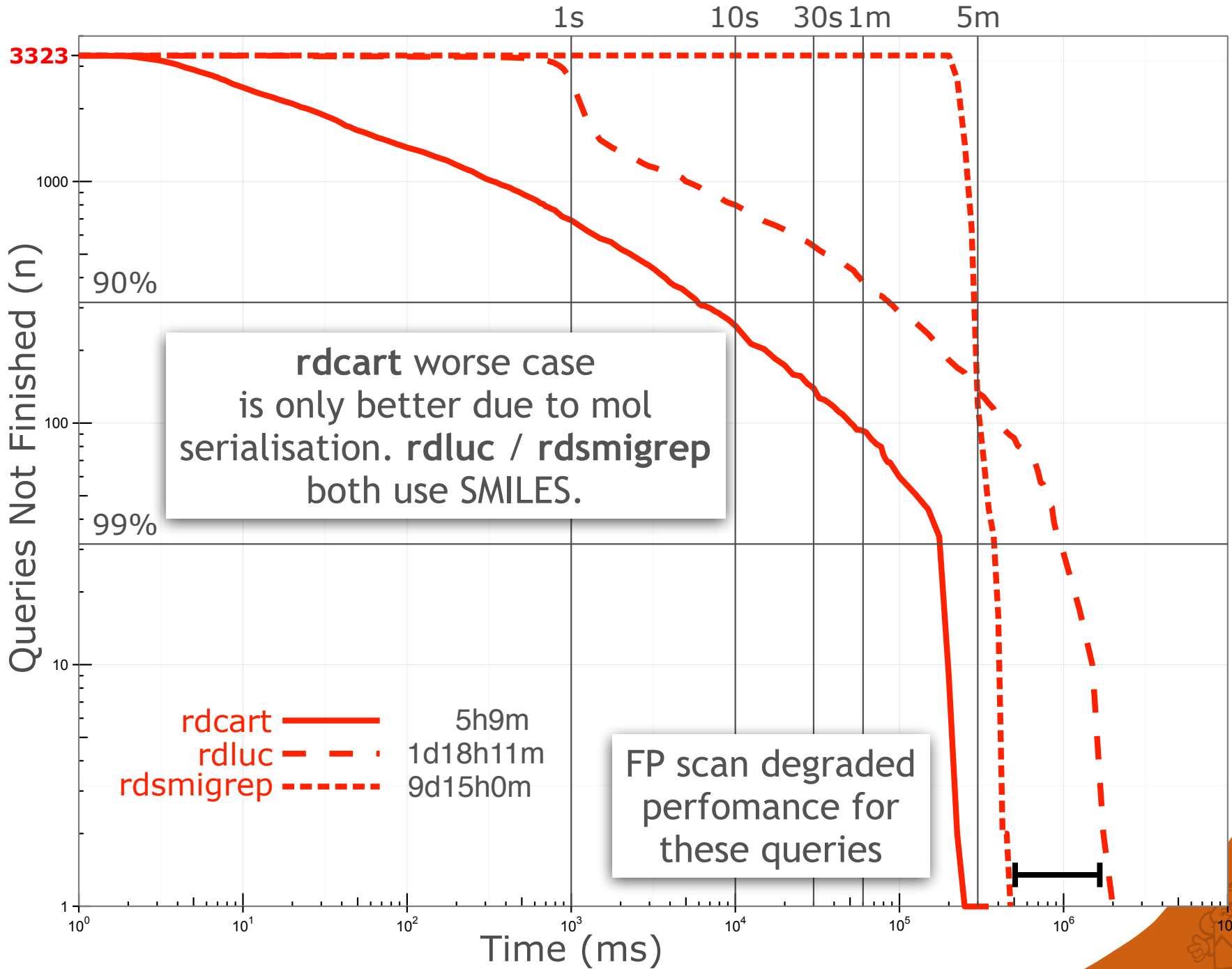
o-xylene

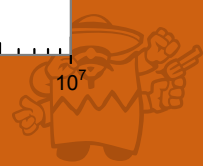
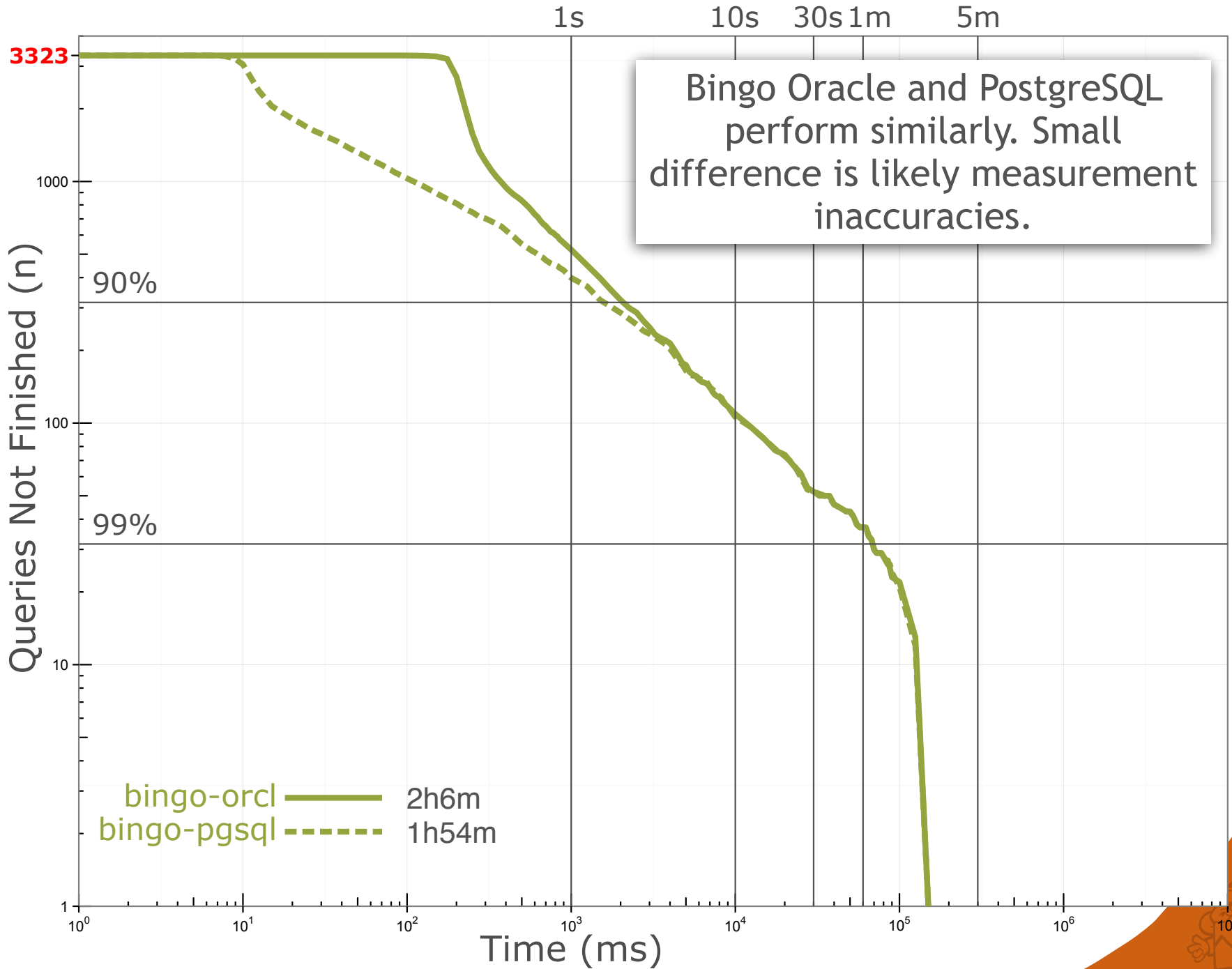


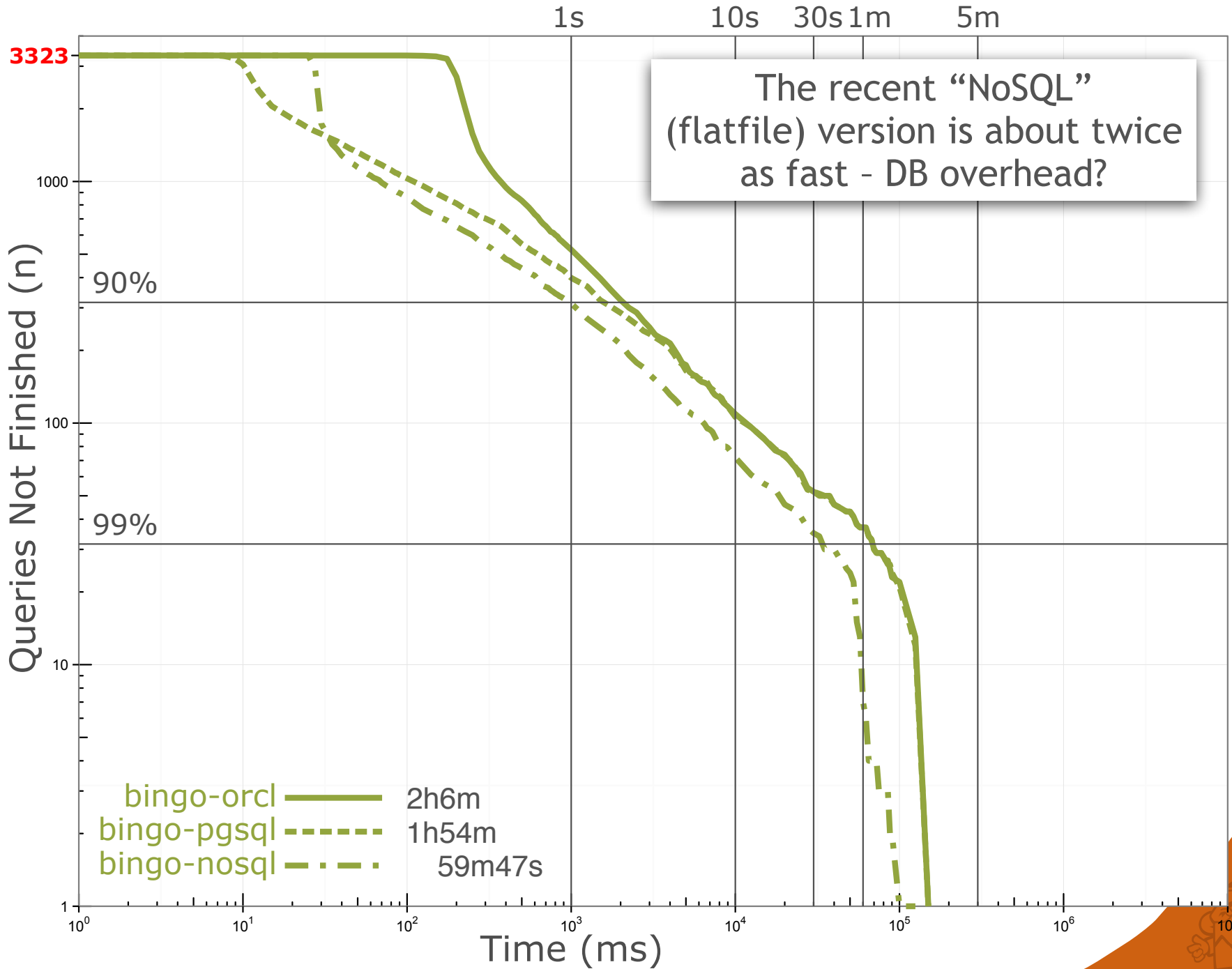






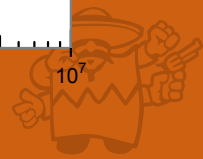


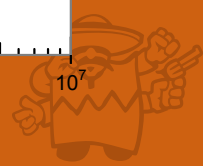
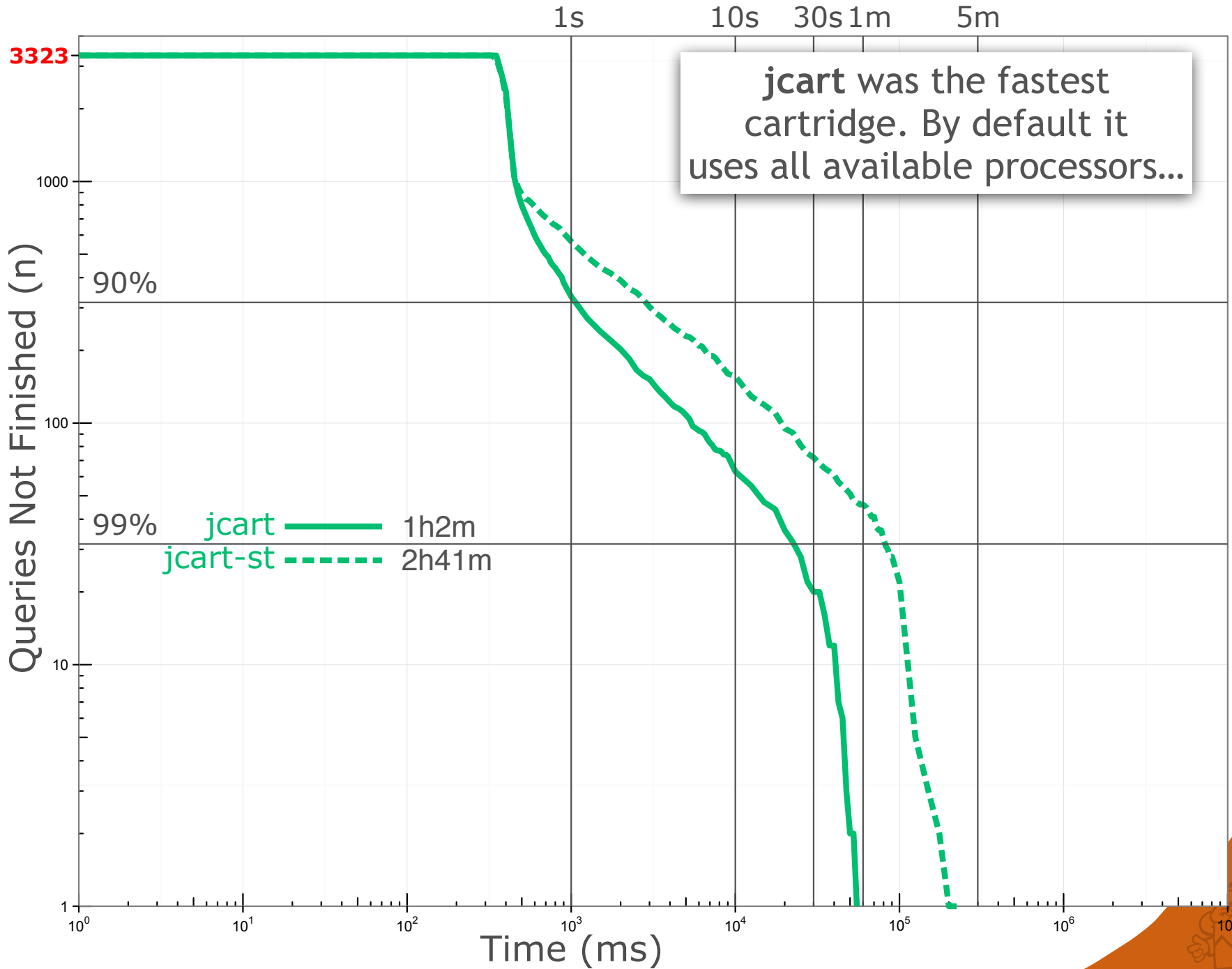


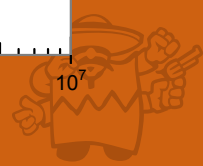
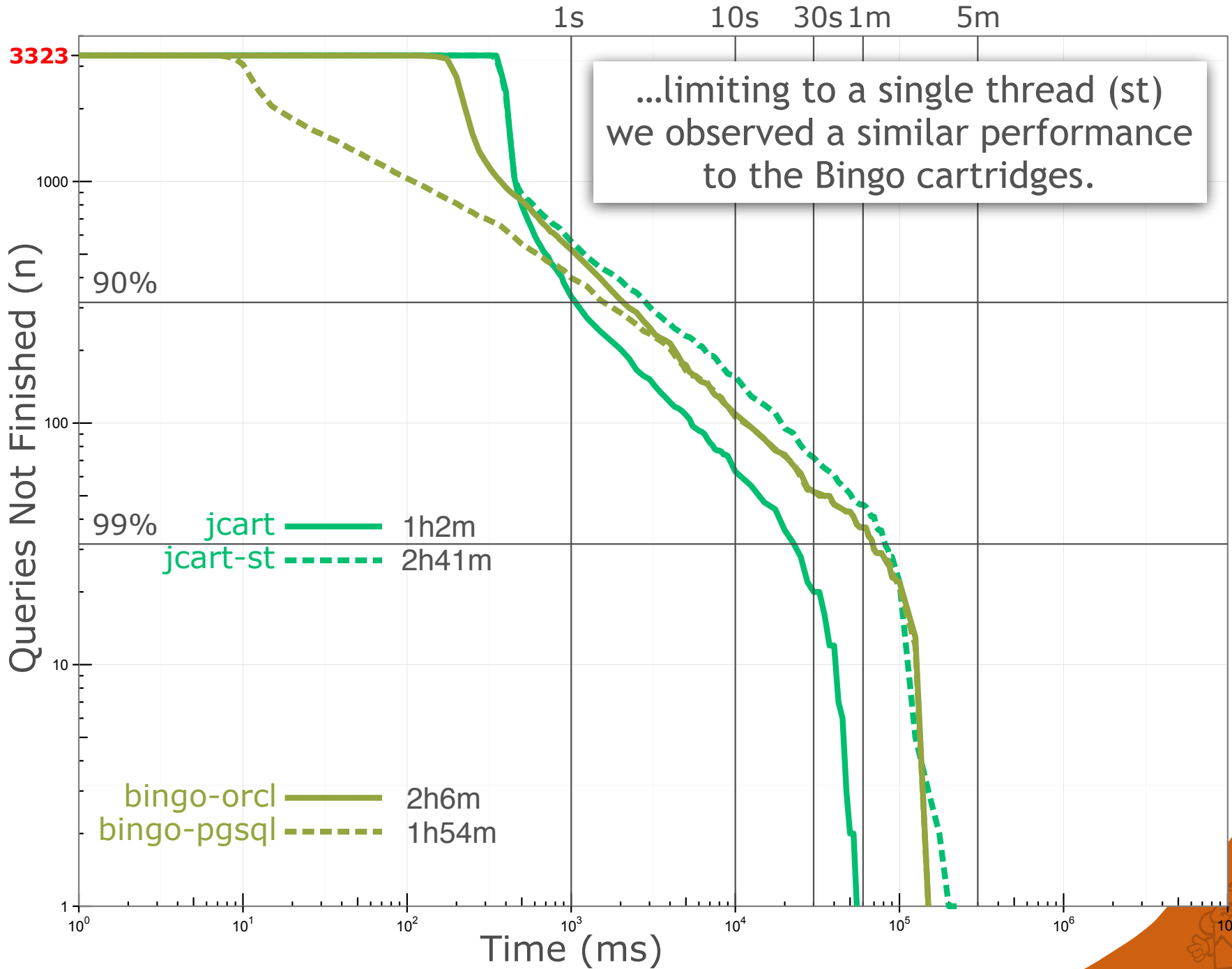


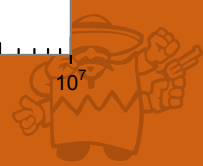
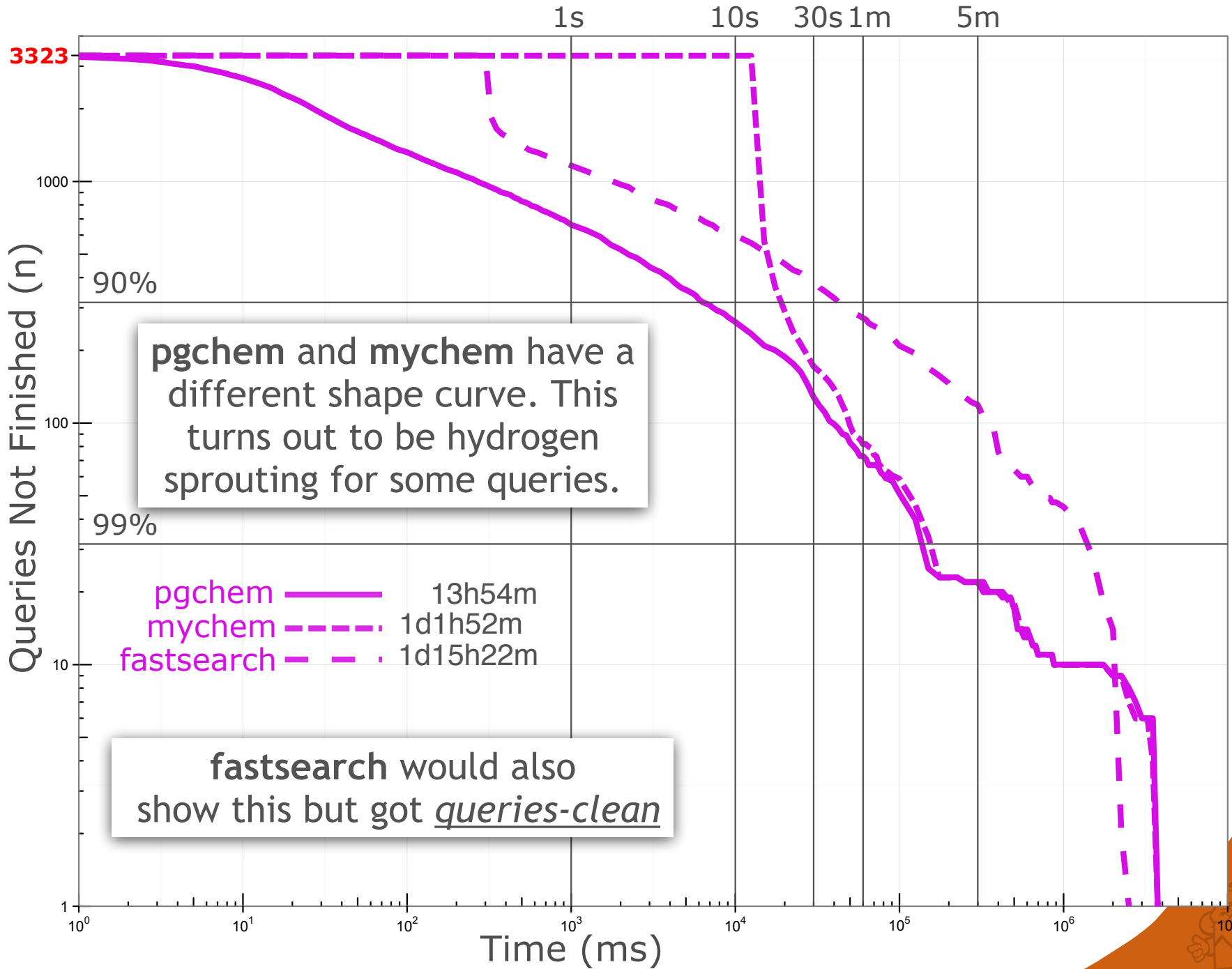
The recent "NoSQL" (flatfile) version is about twice as fast - DB overhead?

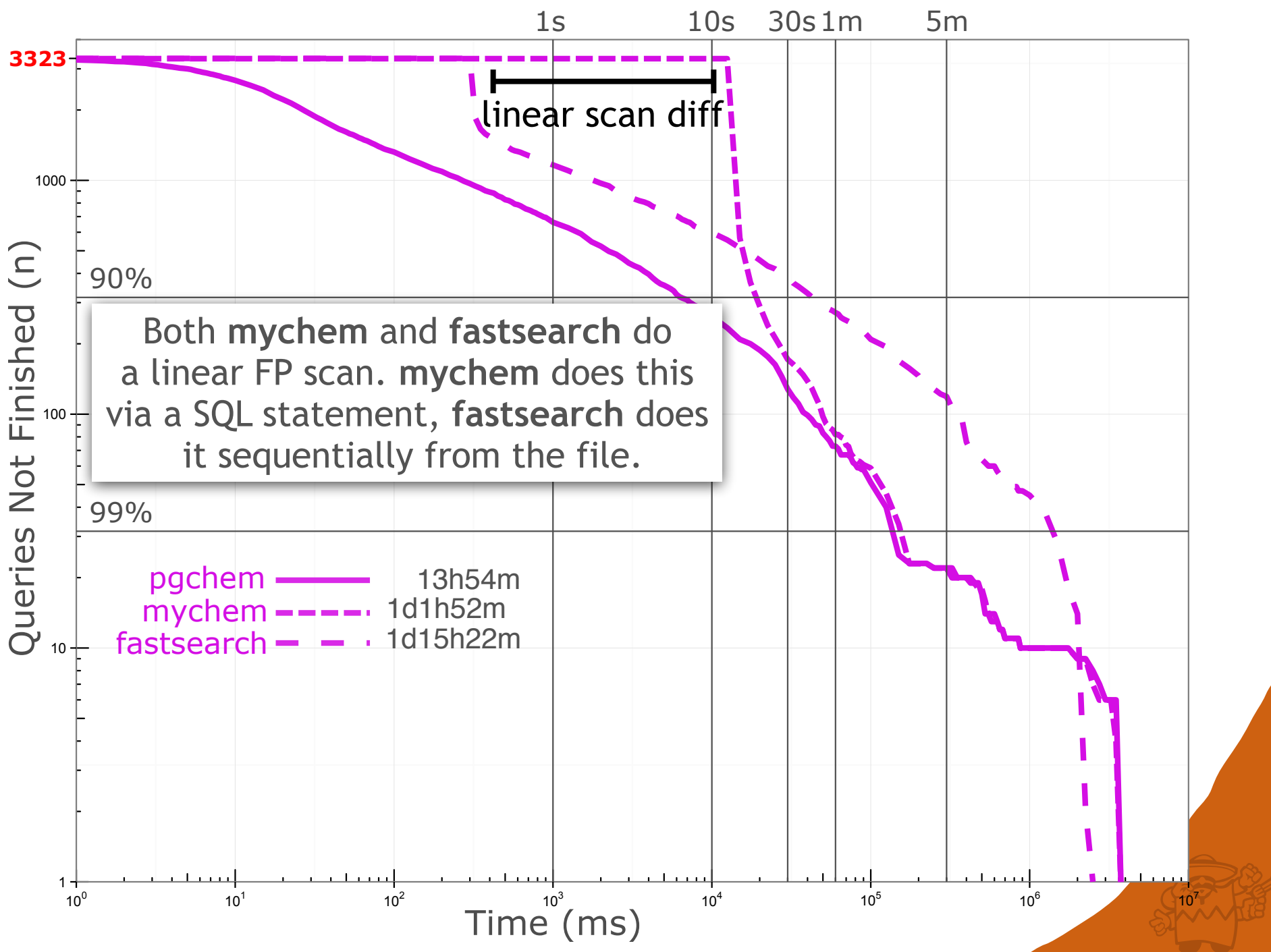
bingo-orcl 2h6m  
 bingo-pgsql 1h54m  
 bingo-nosql 59m47s

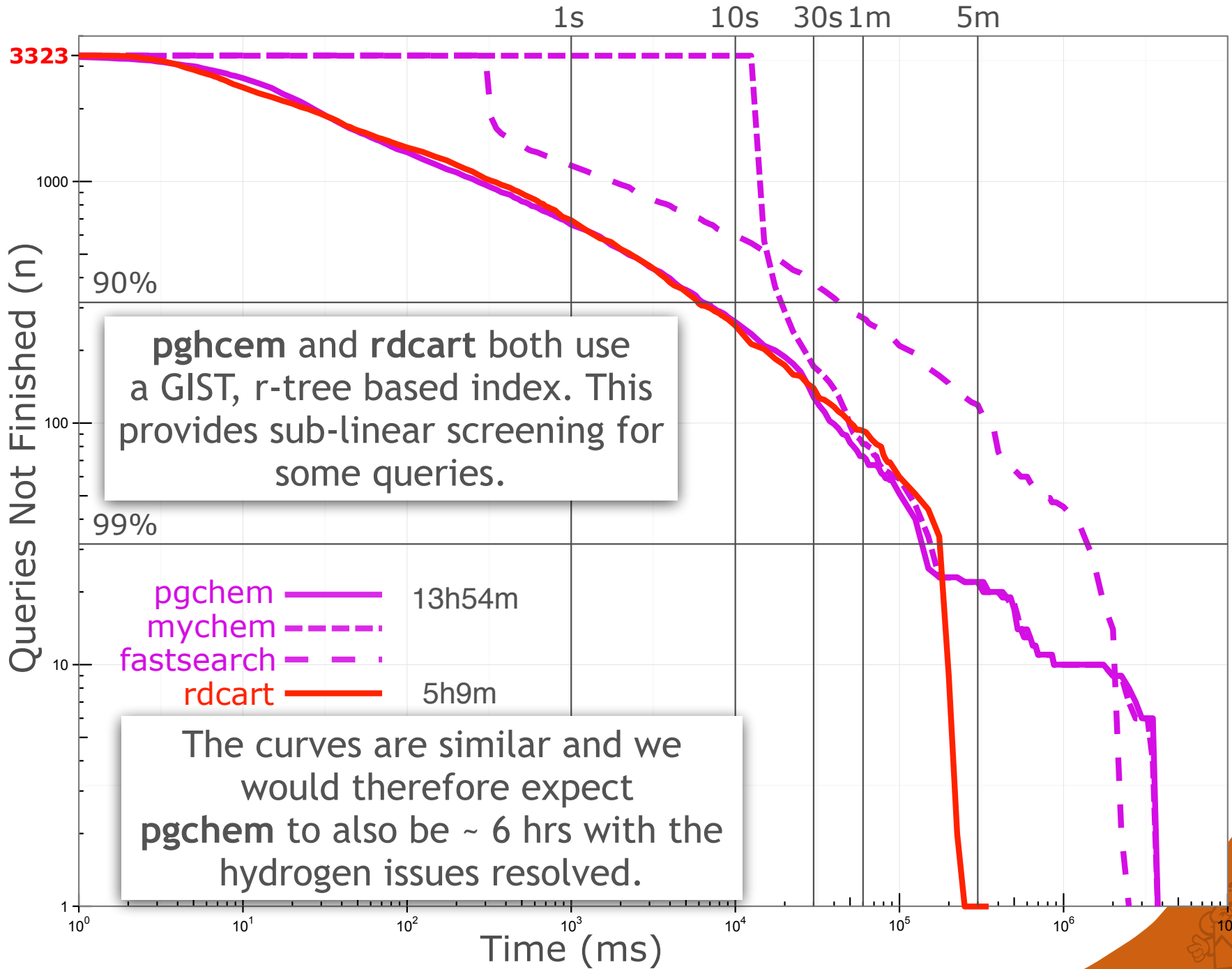






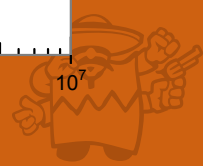


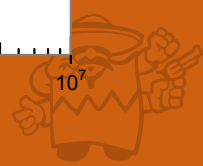
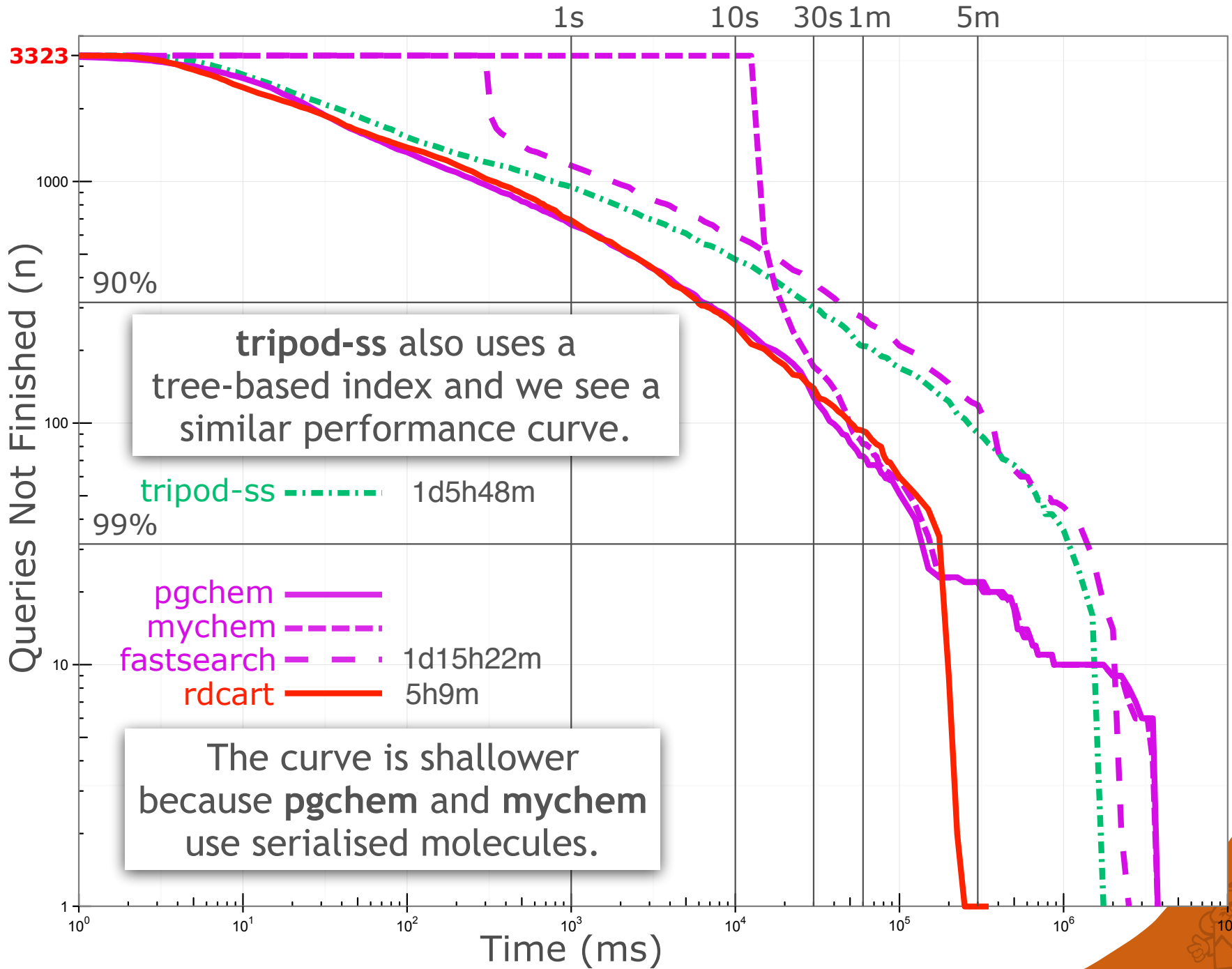


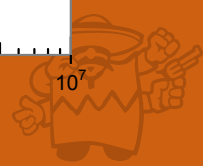
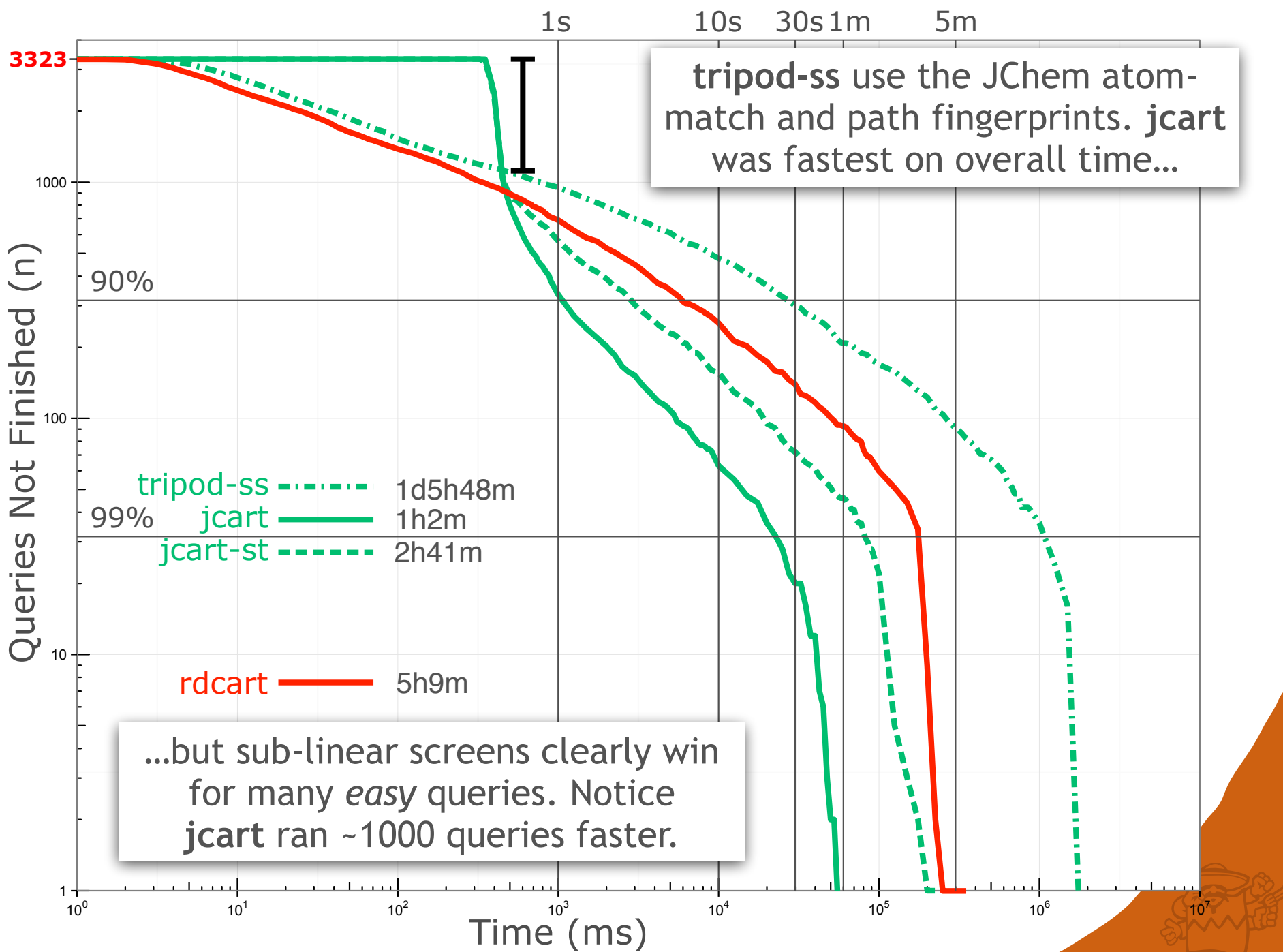


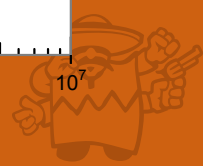
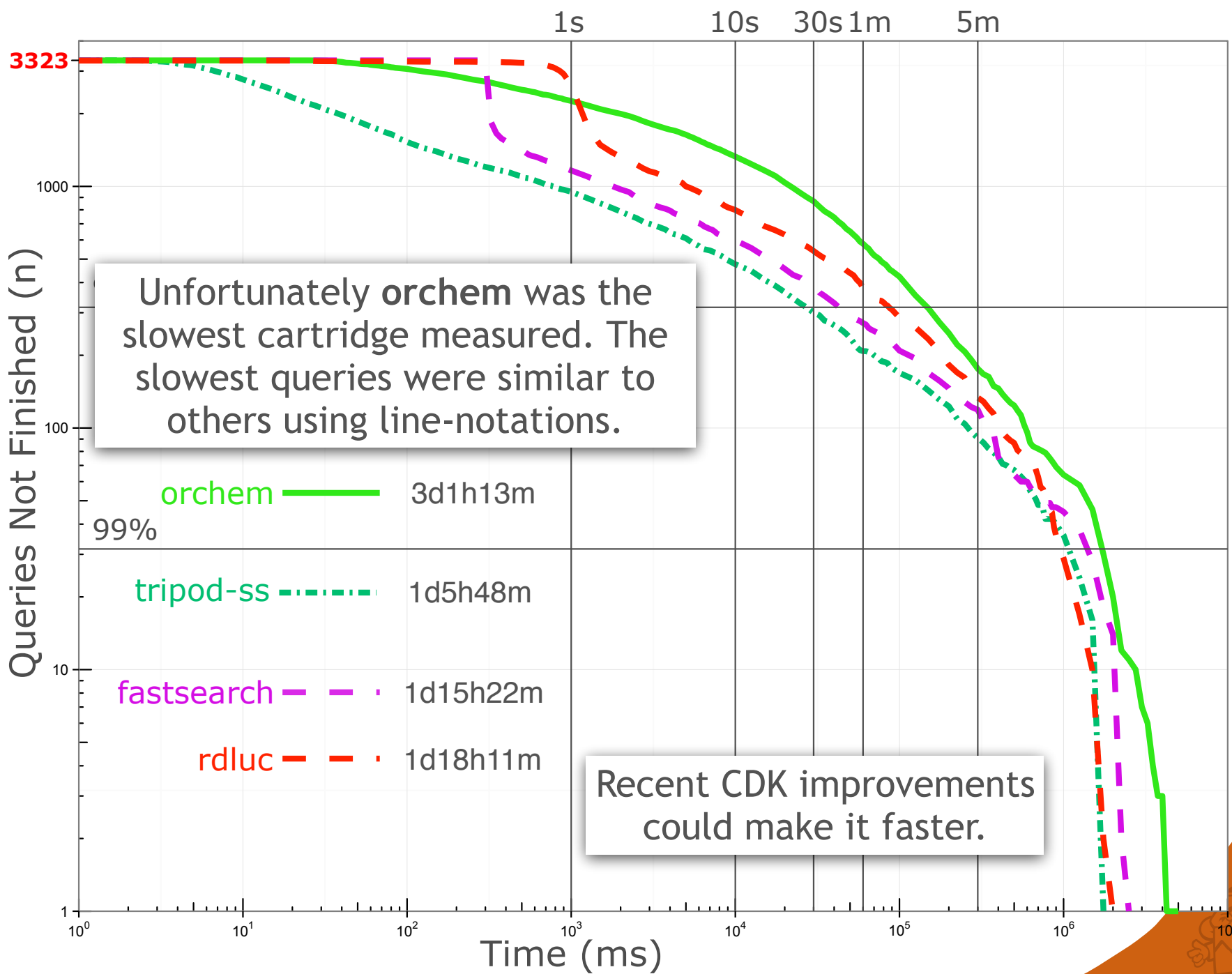
pgchem and rdcart both use a GIST, r-tree based index. This provides sub-linear screening for some queries.

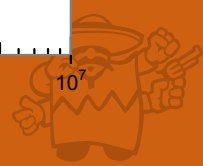
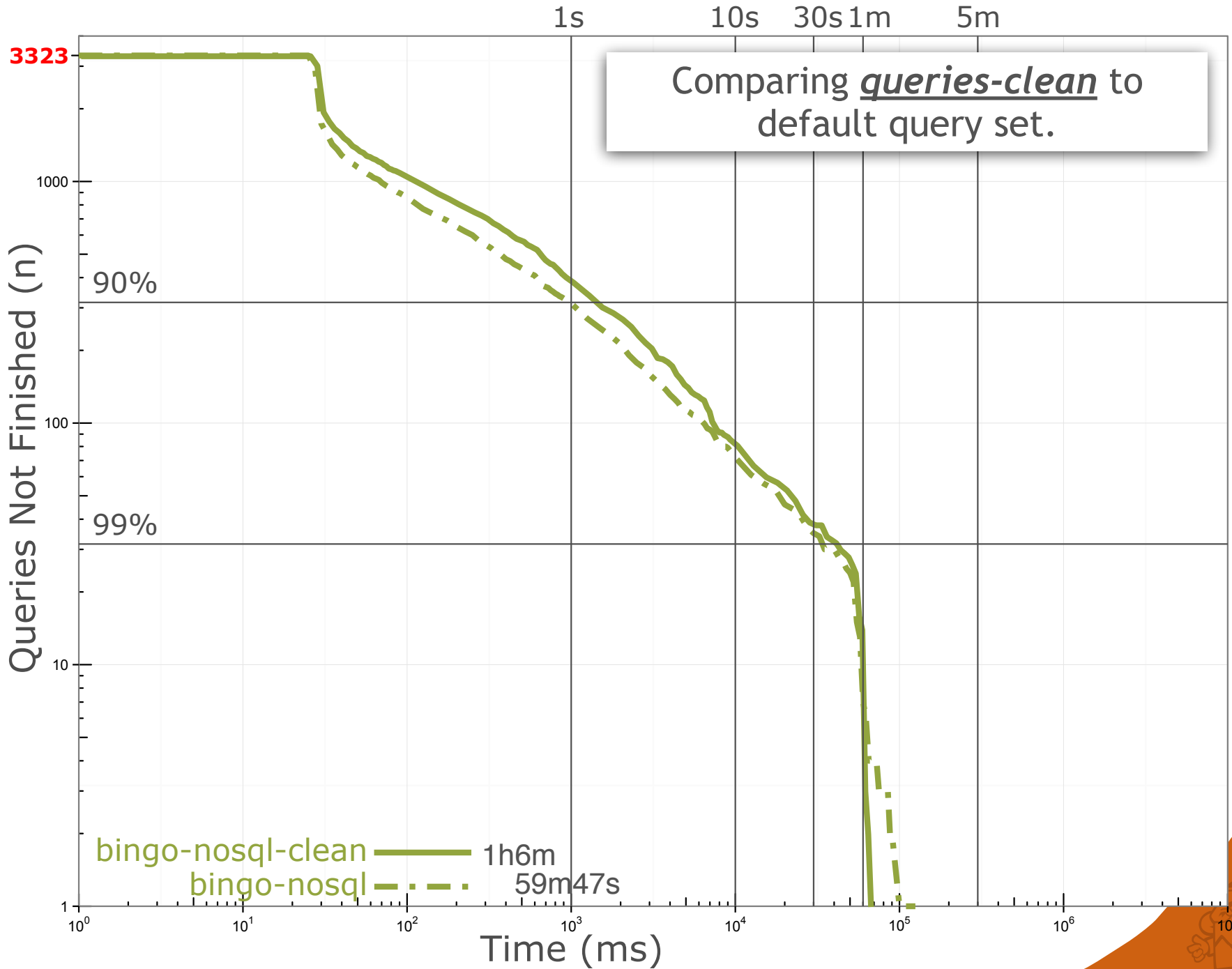
The curves are similar and we would therefore expect pgchem to also be ~ 6 hrs with the hydrogen issues resolved.











# OTHER CONSIDERATIONS

What are the hits? *o*-xylene example

How do they scale?

- Index, tree-based indexes will have many more internal nodes
- Memory usage
  - binary < SMILES < molfile
  - fingerprint length

PubChem Compound currently ~10x eMolecules

# BENCHMARK SUMMARY

Many tools have reasonable performance for most queries but are hindered by their **worst cases**.

Minimal difference between Oracle vs PostgreSQL

OrChem vs JCart (slowest and fastest cartridges)

MyChem and pgchem will benefit from improved query preparation (hydrogen handling)

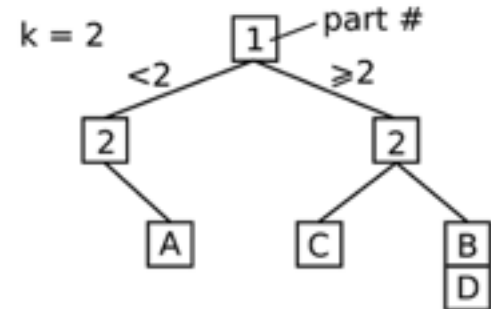
**Domain specific indexes**

preferable to table selects.

Efficient IO

## kD Binary Search Tree

median popcount part 1: 2  
median popcount part 2: 2

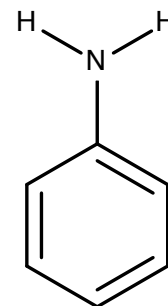


# FINGERPRINT PERFORMANCE

Much work has been done on optimising fingerprints

Screen performance measured for several of the tools

- only partially explains observed time differences
- occasional v. bad precision for some queries
- hydrogens are difficult to encode<sup>1</sup>
- **Maybe False Negatives! - but they're relative**



TOOL	TYPE	SCREENOUT	PRECISION
bingo	Hashed Trees/Rings	98.63%	80.9%
rdcart	Hashed Patterns	97.95%	59.9%
jcart, tripod-ss	Hashed Paths	98.15%	59.7%
pgchem, mychem, fastsearch	Hashed Paths/Rings	98.2%	57.3%

1. <http://nextmovesoftware.com/blog/2015/02/16/>



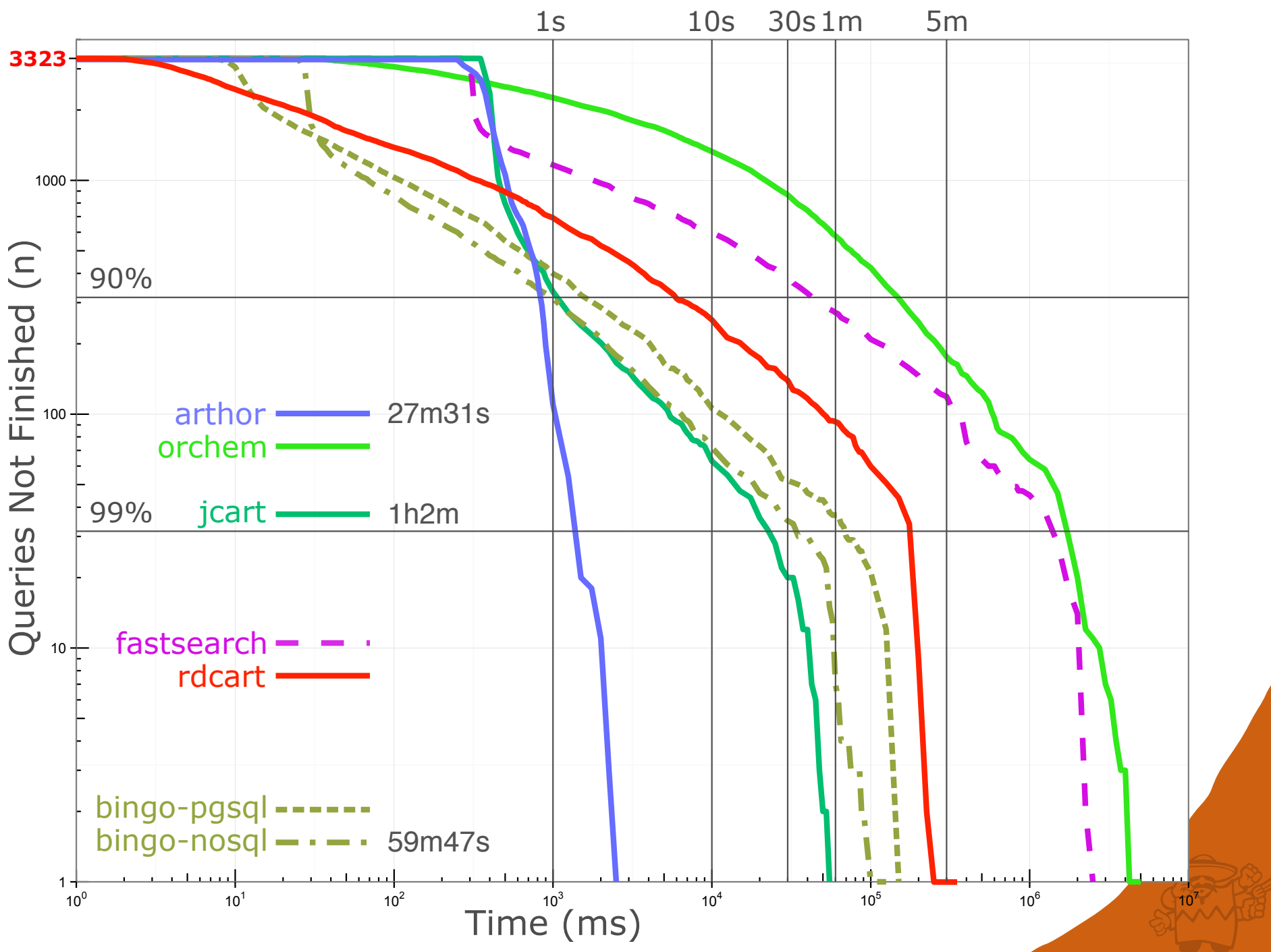
# AN EFFICIENT ATOM-MATCH

- Query optimisation
- Database preprocessing
- Efficient binary representation
  - eMolecules, 1.2GiB
- Matching done on representation

# ARTHOR

(CODENAME)





# FINGERPRINTING MATERS

Arthor - 0% screen-out

23,290,449,670 atom-matches

Bingo - 98.63% screen-out

319,079,160 atom-matches

Can test how much different fingerprints mater...

Provide **arthor** a pre-computed list of screen hits

Times therefore presume instant **bit screen**

False negatives are relative to 0% screen-out.

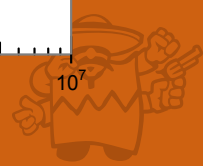
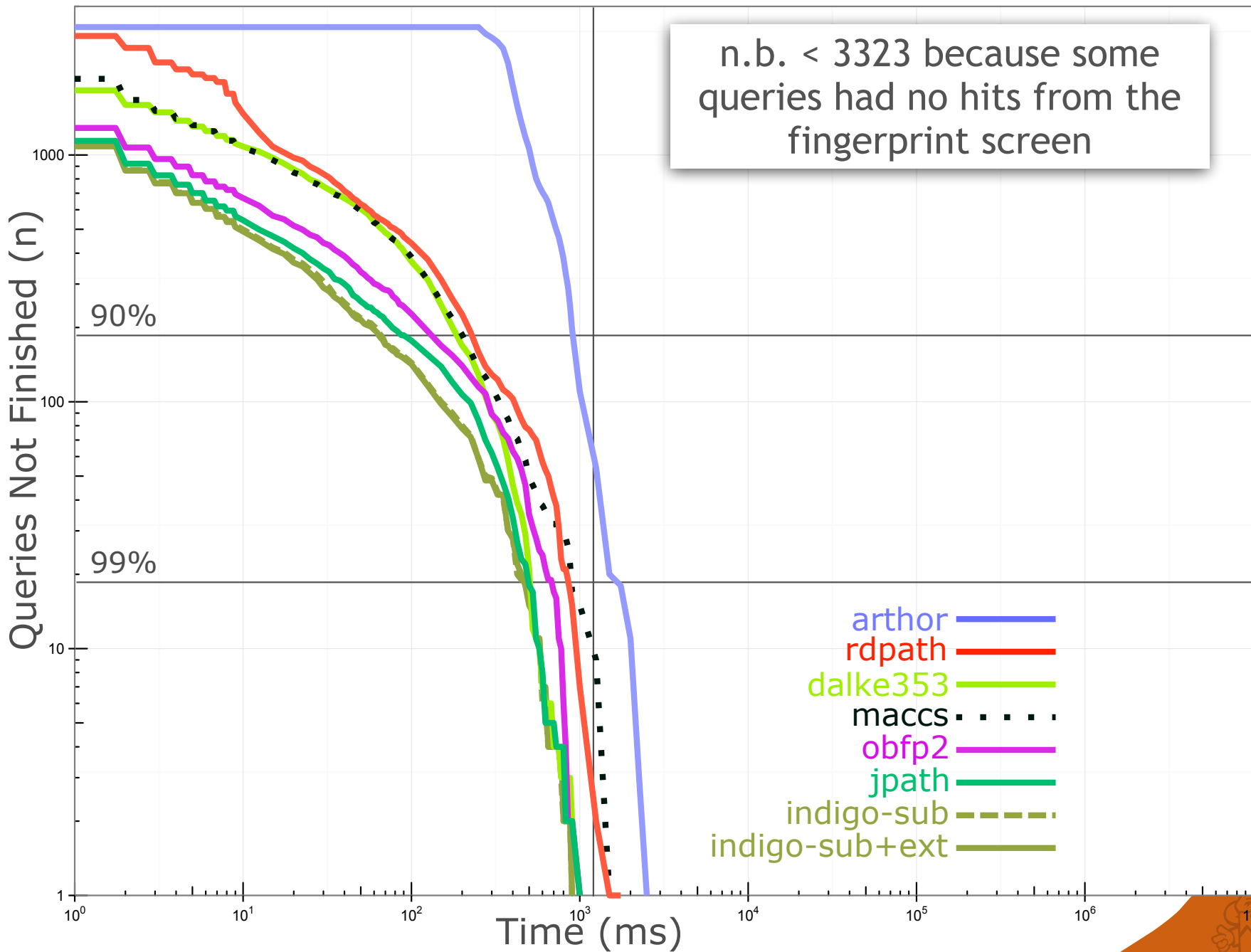


# FINGERPRINTING MATERS

FINGERPRINT	LENGTH	DENSITY	TIME	FALSE NEGATIVES
<b>arthur</b>	-	-	27m35s	-
<b>+rdpath</b>	2048	61.90%	2m52s (x9.5)	61,915 (0.02%)
<b>+maccs</b>	166	29.08%	2m26s (x11.2)	140,968,649 (57.63%)
<b>+dalke353</b>	353	6.59%	2m2s (x13.5)	0
<b>+obfp2</b>	1021	12.55%	1m32s (x17.9)	641,712 (0.26%)
<b>+jpath</b>	1024	23.36%	1m7s (x24.5)	41,889 (0.01)
<b>+indigo-sub</b>	1600	23.09%	55s (x29.6)	128,288 (0.05%)
<b>+indigo-sub+ext</b>	1624	22.48%	54s (x30.1)	128,288 (0.05%)

1s

n.b. < 3323 because some queries had no hits from the fingerprint screen



# CONCLUSIONS

Fast searching depends on many factors

- index data-structure
- fingerprint type
- molecule representation
- atom-match algorithm

Fingerprint optimisation does matter but is **not** the performance bottleneck and can't always help:

Generic patterns, a-a

Hydrogens, [H]N([H])C



# ACKNOWLEDGEMENTS

Daniel Lowe and Noel O'Boyle, NextMove Software

Andrew Dalke

- assembling Structure Query Collection (SQC)

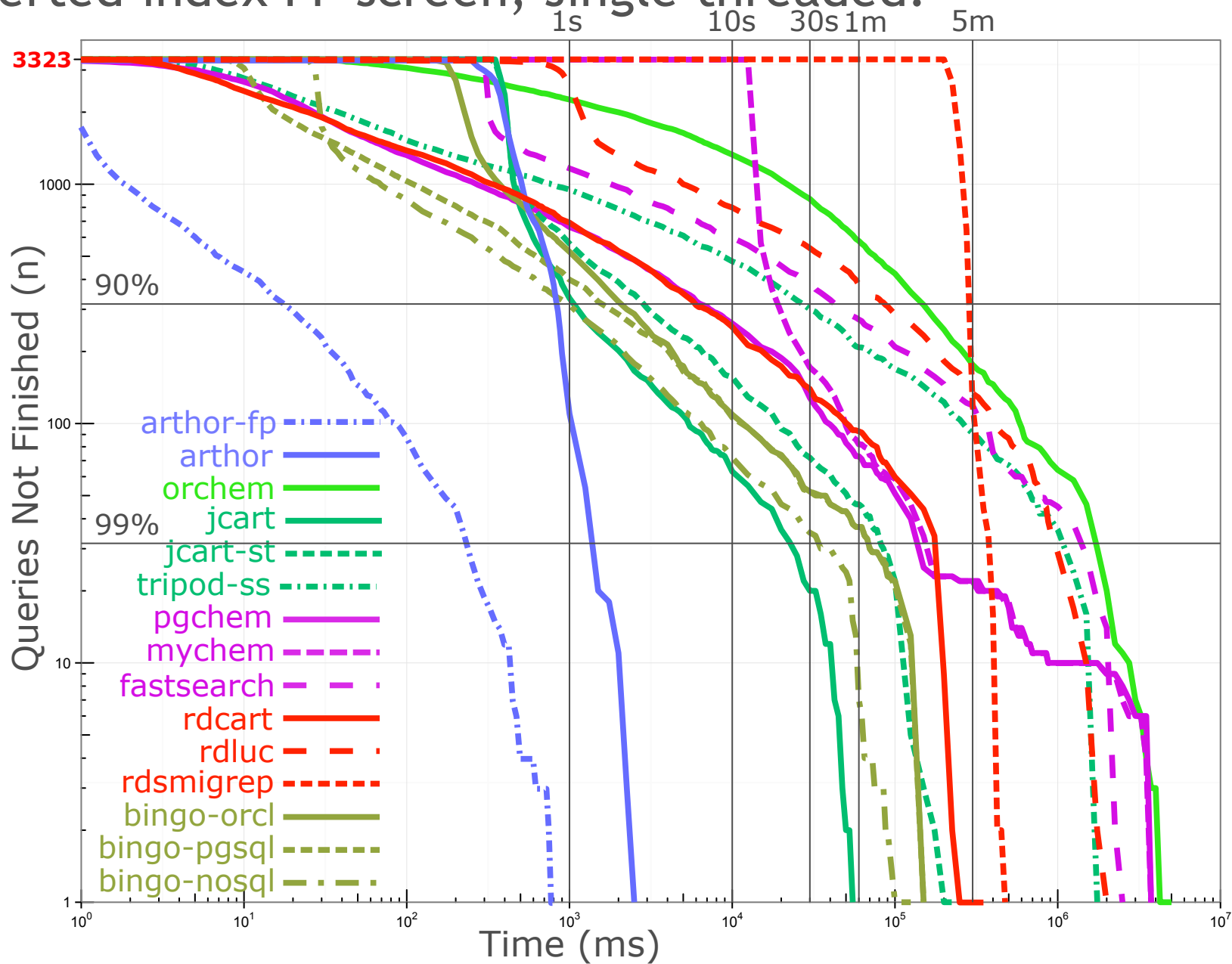
Michael Gilson and Tiqing Liu

- providing BindingDB (<http://www.bindingdb.org/>) queries for SQC

Tool and cartridge developers

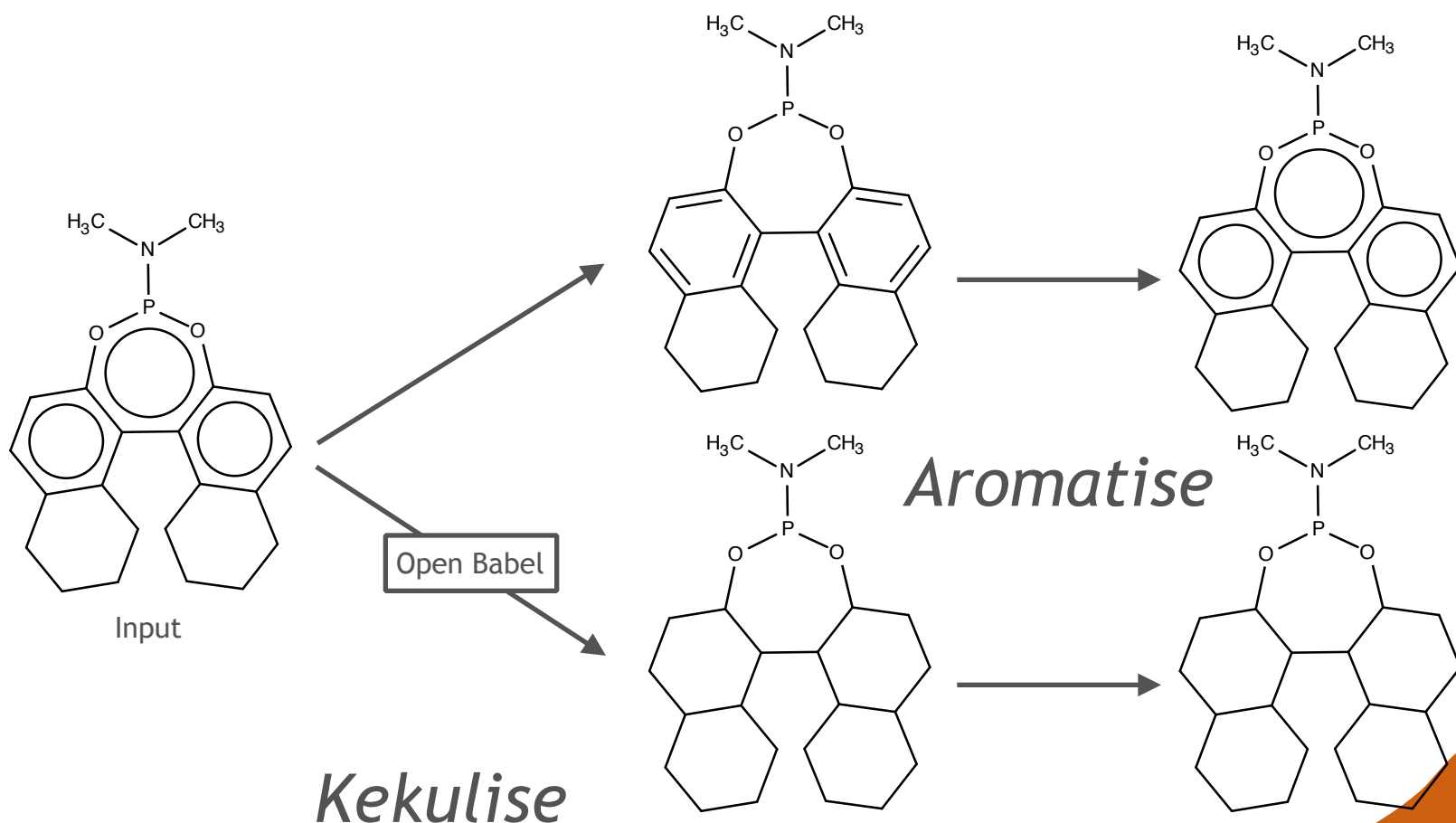


Further optimisation and keeping file open: 36 s including inverted index FP screen, single threaded:

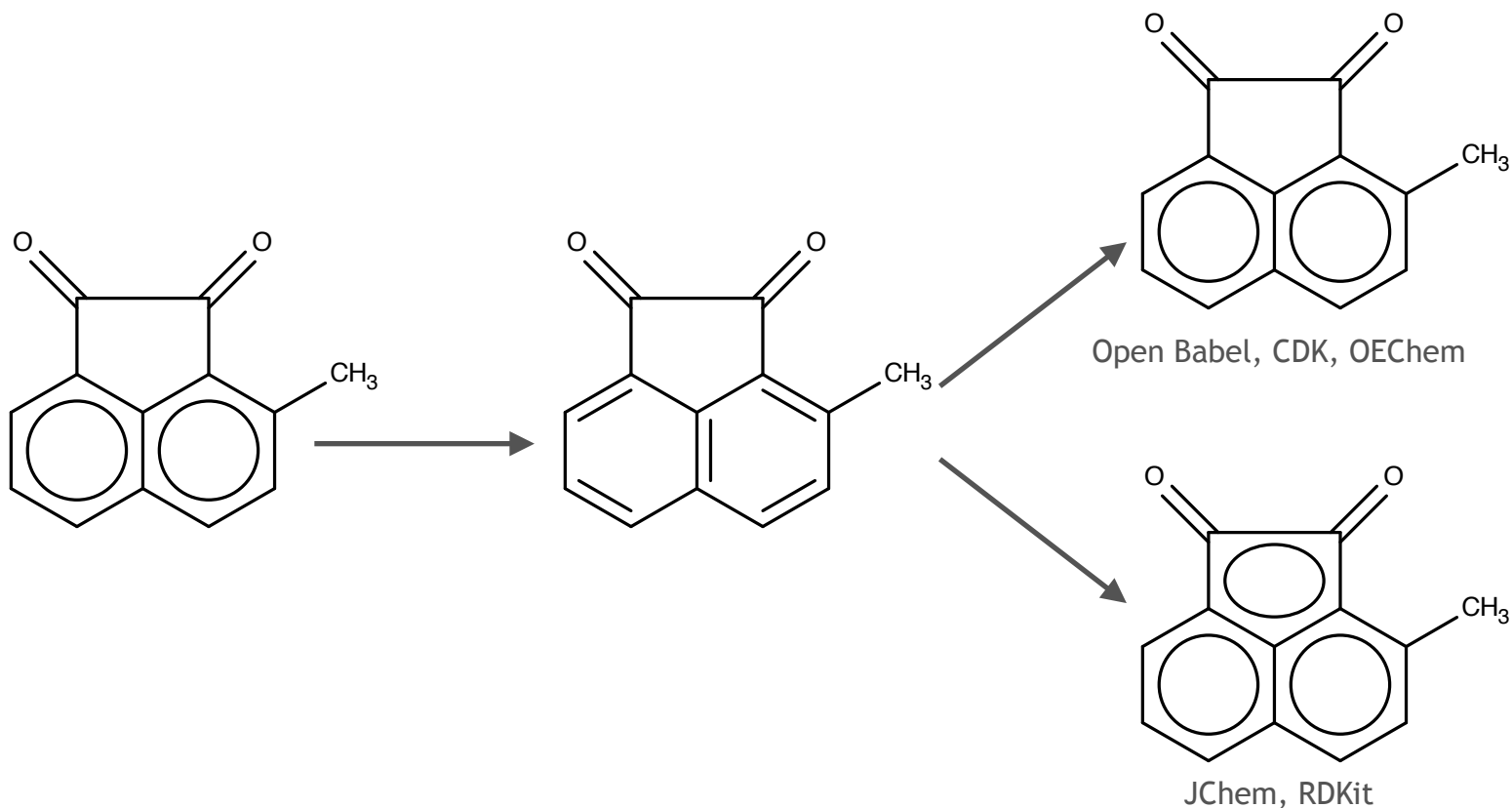


# SOME FALSE NEGATIVES

Most false negatives are due to Kekulisation/Aromaticity. Typically when reading SMILES a tool will assign a Kekulé form and then re-aromatise.



# SOME FALSE NEGATIVES



*Kekulise*

*Aromatise*

