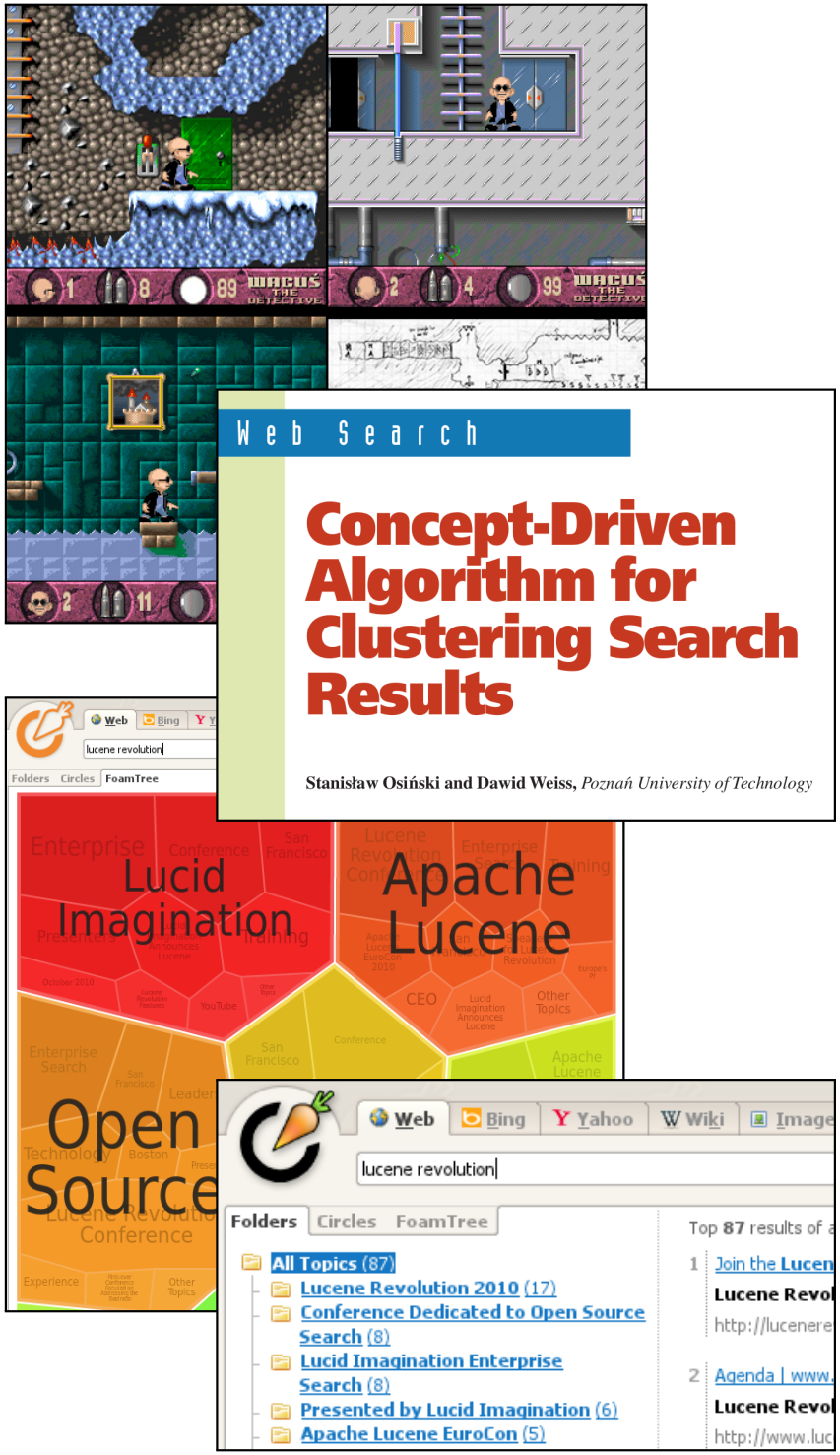
Finite State Automata in Lucene and Solr

Dawid WEISS



Dawid Weiss

20+ years of coding

10 years assembly only

Academia & Research

PhD in Information Retrieval, PUT

Open source

Carrot², HPPC, Lucene, …

Industry & Business

Carrot Search s.c.

Talk outline

State machines (automata)

FSAs, DFAs, FSTs and other XXXs.

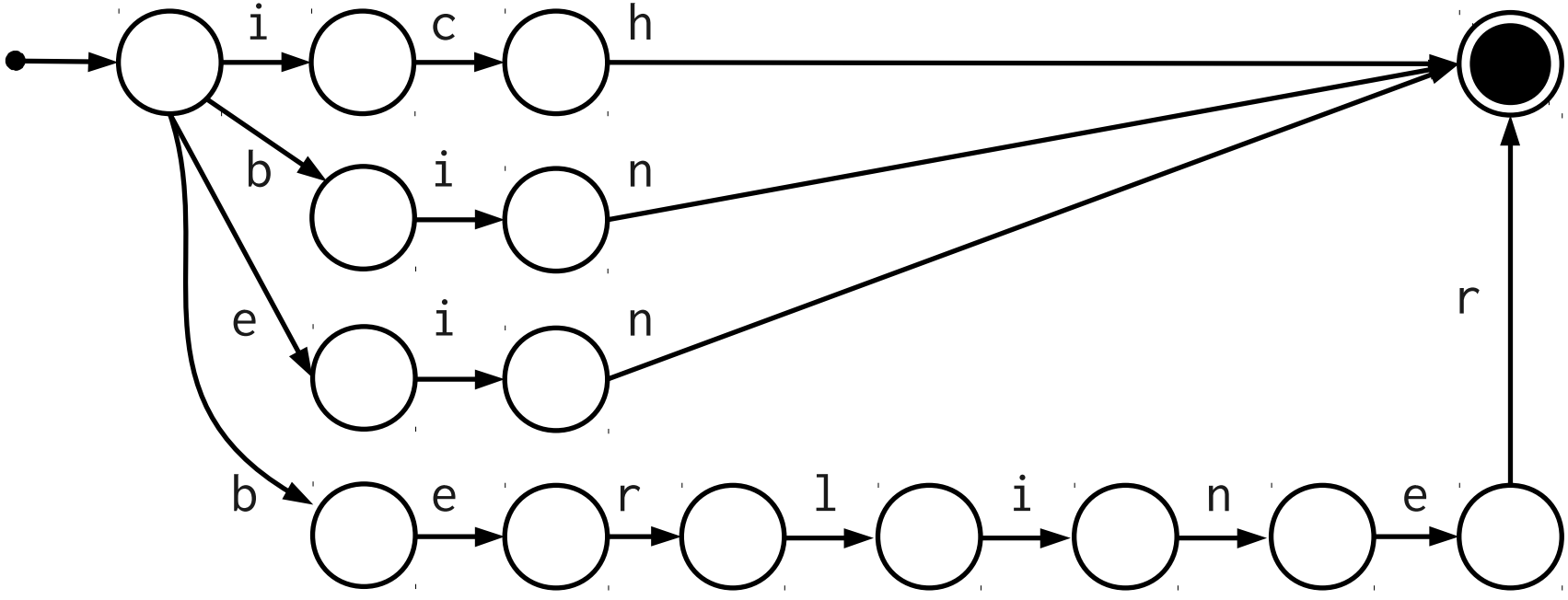
Use cases in Lucene and Solr

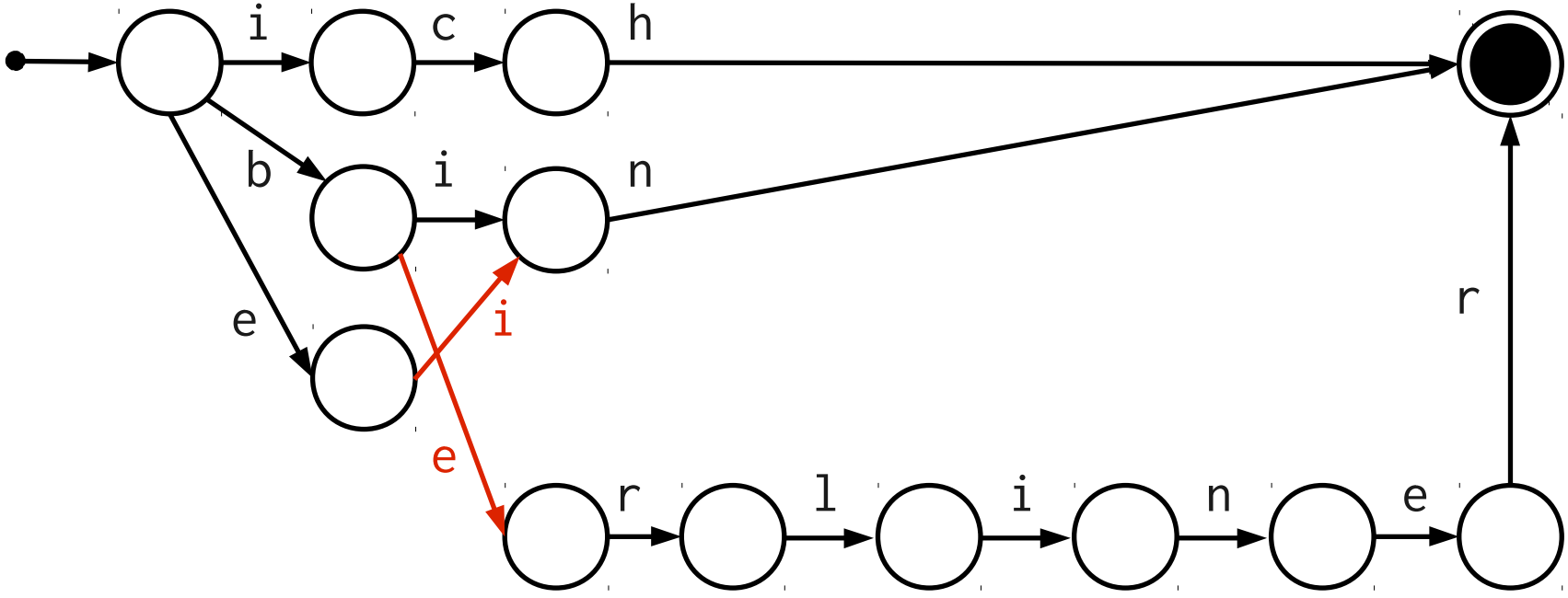
Suggester. FuzzySearch. Index.

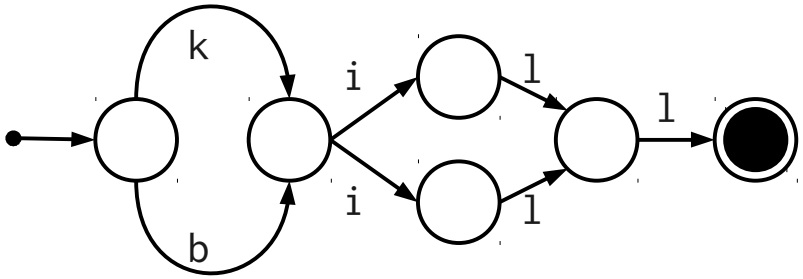
No API details

Still @experimental.

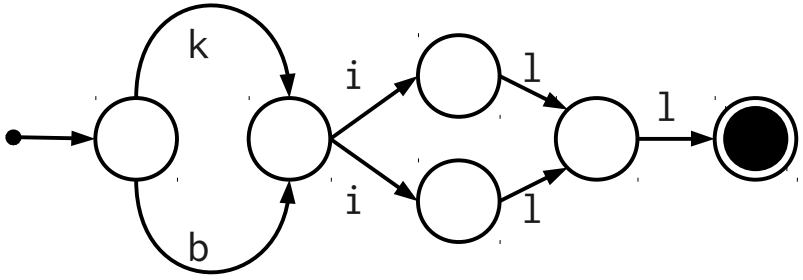
(Non)? Deterministic Finite State (Automata|Machines)


i
c
h
b
i
n
e
i
n
r
b
e
r
l
i
n
e

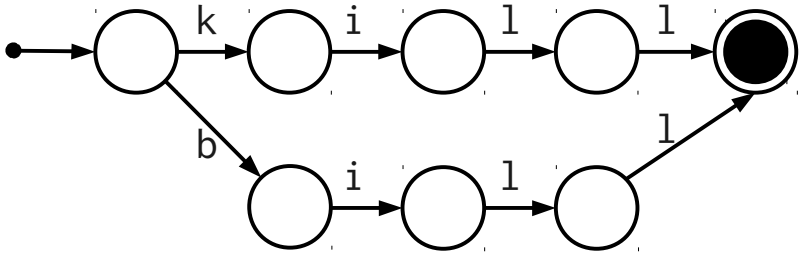

i
c
h
b
i
n
e
i
e
r
r
l
i
n
e



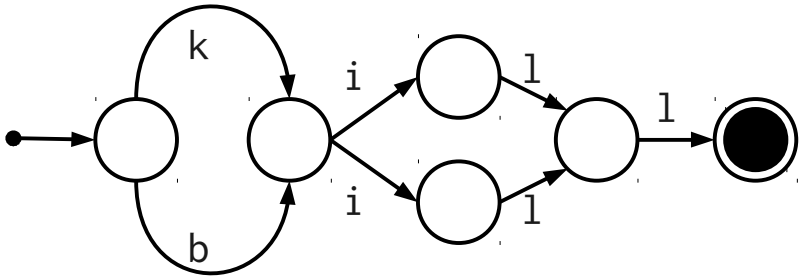
non-deterministic,
non-minimal



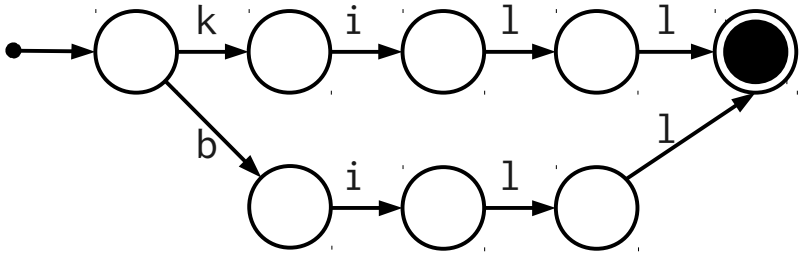
non-deterministic, non-minimal



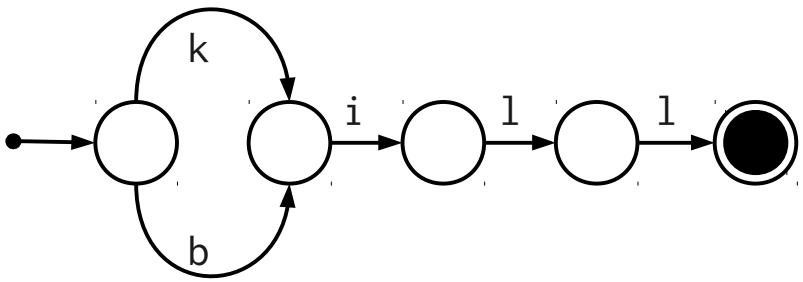
deterministic, non-minimal



non-deterministic, non-minimal



deterministic, non-minimal



deterministic, minimal

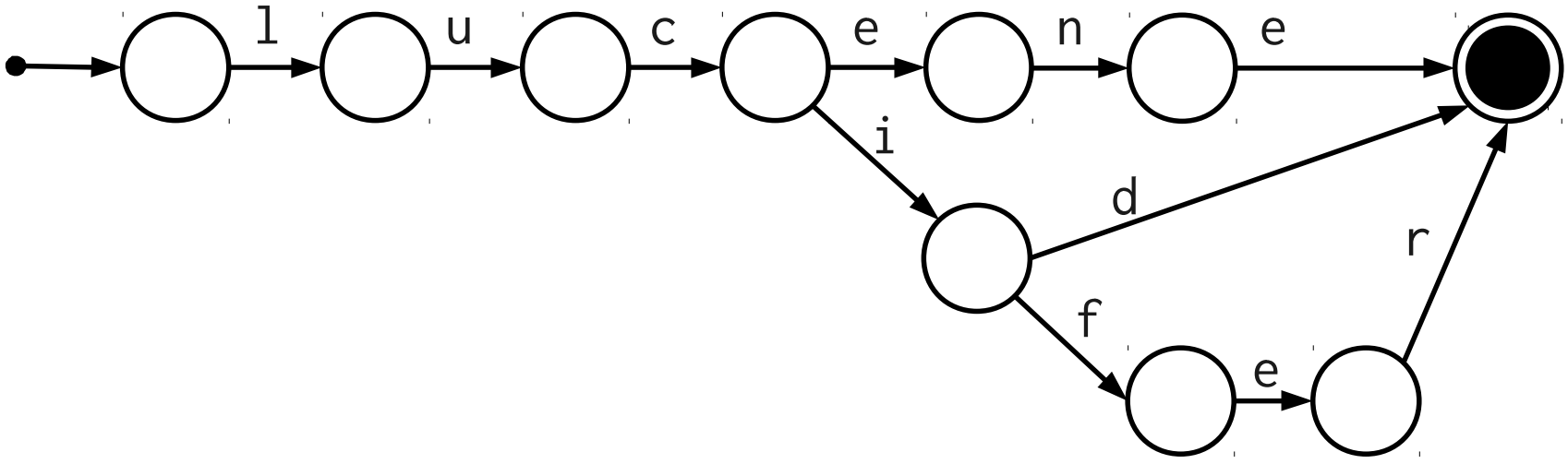
HashSet

hash	→ slot	→ value
0x29384d34		→ lucene
0xde3e3354		→ lucid
0x00000666		→ lucifer

HashSet

hash	→ slot	→ value
0x29384d34		→ lucene
0xde3e3354		→ lucid
0x00000666		→ lucifer

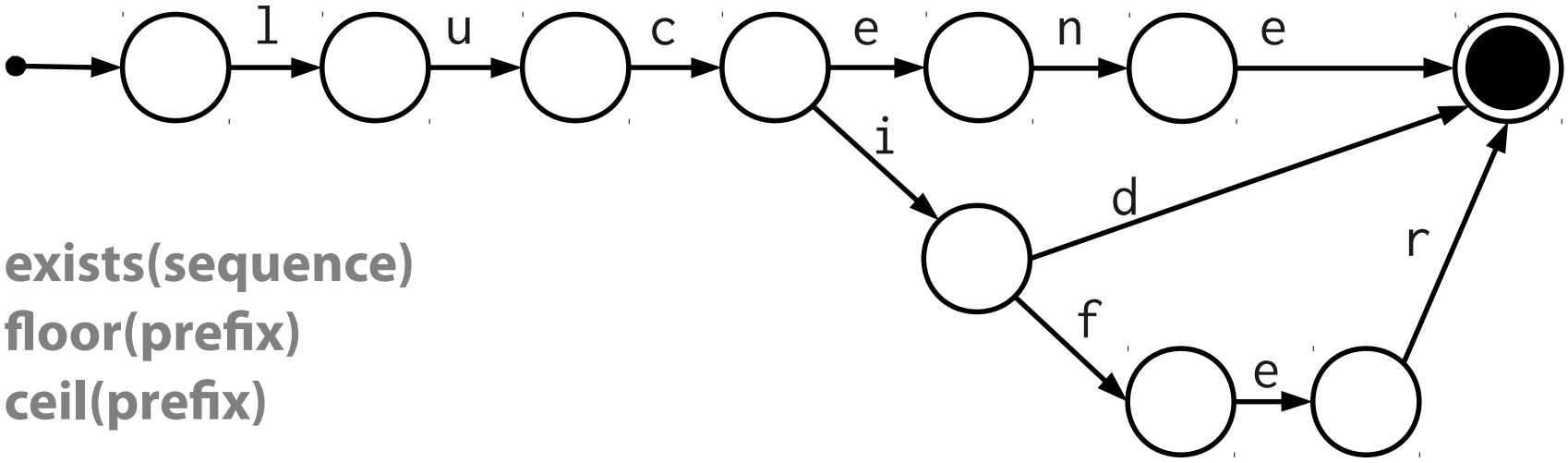
FSA



HashSet

hash	→ slot	→ value
0x29384d34		→ lucene
0xde3e3354		→ lucid
0x00000666		→ lucifer

FSA



exists(sequence)
floor(prefix)
ceil(prefix)

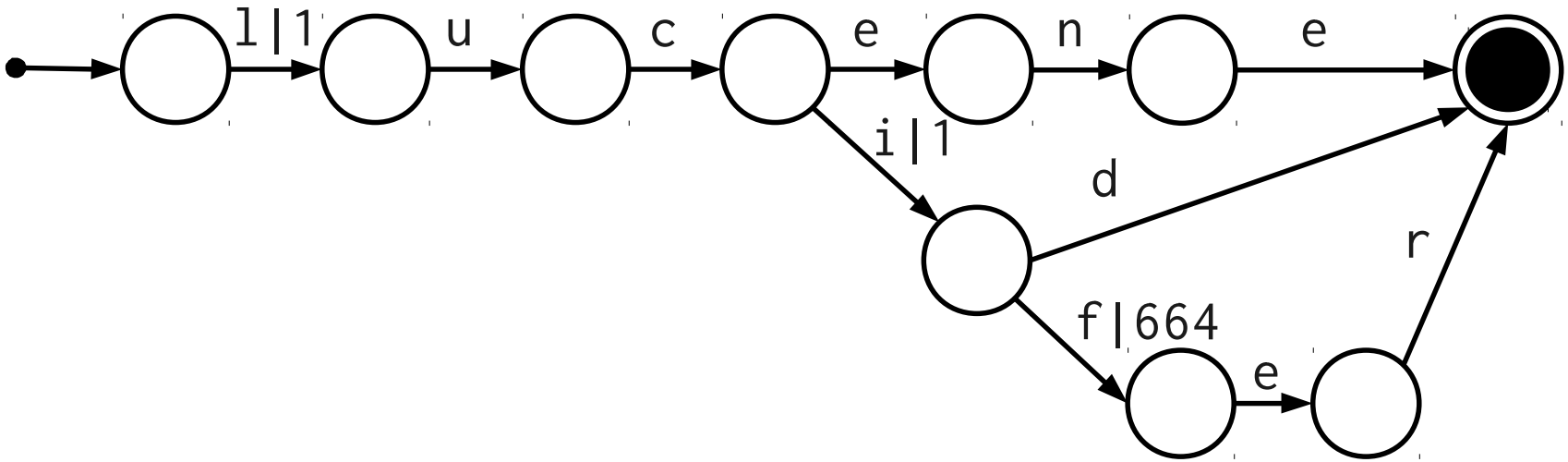
(Sorted)Map

```
lucene  → 1
lucid   → 2
lucifer → 666
```

(Sorted)Map

```
lucene   → 1
lucid    → 2
lucifer  → 666
```

FST (transducer)



Linear-time, minimal, deterministic

FSA construction

Linear algorithm from sorted input

by Daciuk, Mihov, et al.

Active path

states that still can change

States dictionary

nodes that will never change

FS(A|T)s in (Lucene|Solr)

Automata in

Lucene|Solr

org.apache.lucene.util.automaton.*

partial port of brics, FuzzyQuery, AutomatonTermsEnum

org.apache.lucene.util.fst.*

FSA and FSTs from sorted data, suggester, indexes

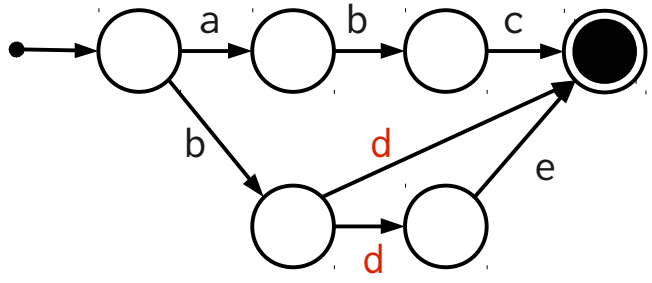
org.apache.lucene.util.fst.*

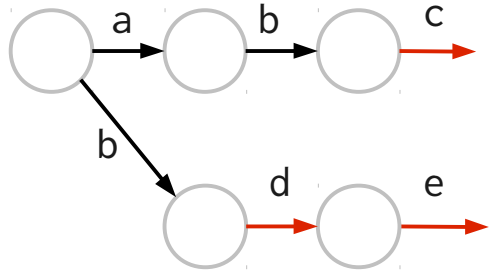
FSA representation

Arc-based, not state-based

Moore vs. Mealy. Compact vs. intuitive

Input: abc, bd, bde.





org.apache.lucene.util.fst.*

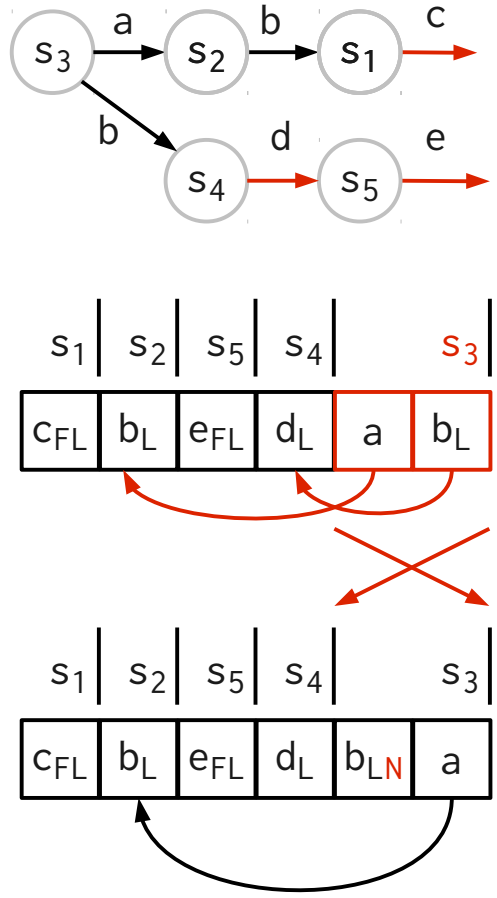
FSA representation

Arc-based, not state-based

Moore vs. Mealy. Compact vs. intuitive

Next-state chaining

requires unusual tricks during construction



org.apache.lucene.util.fst.*

FSA representation

Arc-based, not state-based

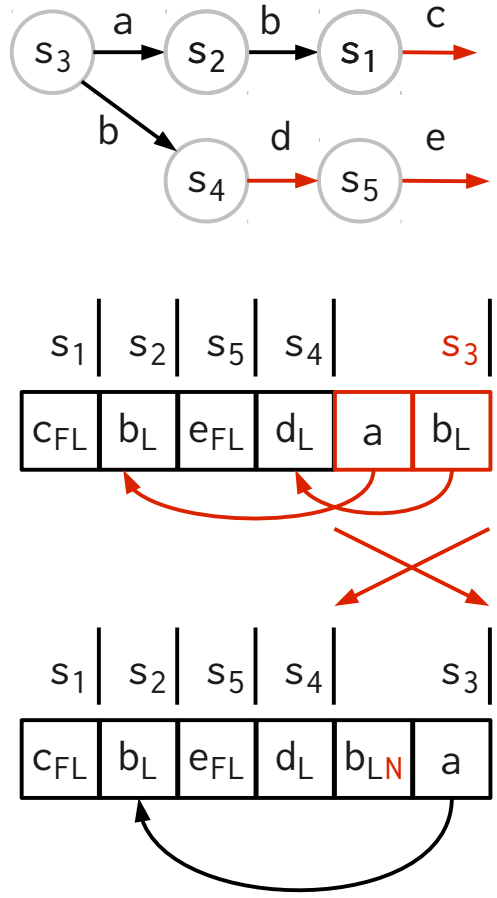
Moore vs. Mealy. Compact vs. intuitive

Next-state chaining

requires unusual tricks during construction

Everything in a byte[]

traversals-ready, memory-efficient



org.apache.lucene.util.fst.*

FSA representation

Arc-based, not state-based

Moore vs. Mealy. Compact vs. intuitive

Next-state chaining

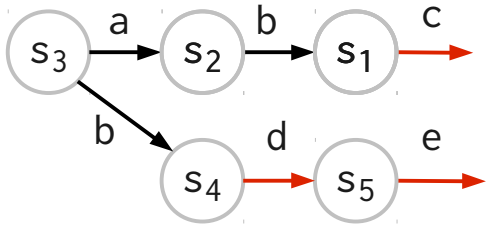
requires unusual tricks during construction

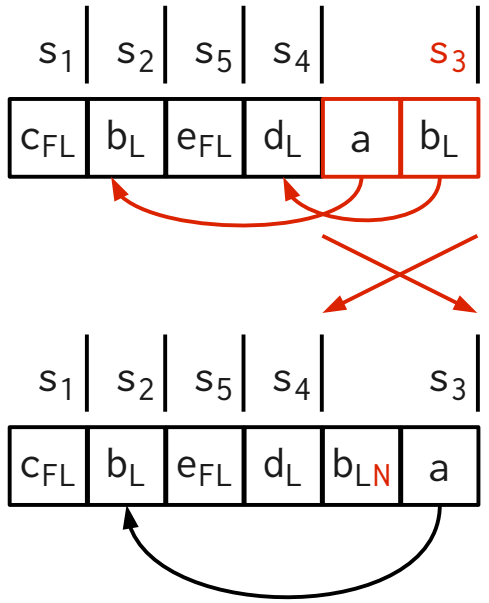
Everything in a byte[]

traversals-ready, memory-efficient

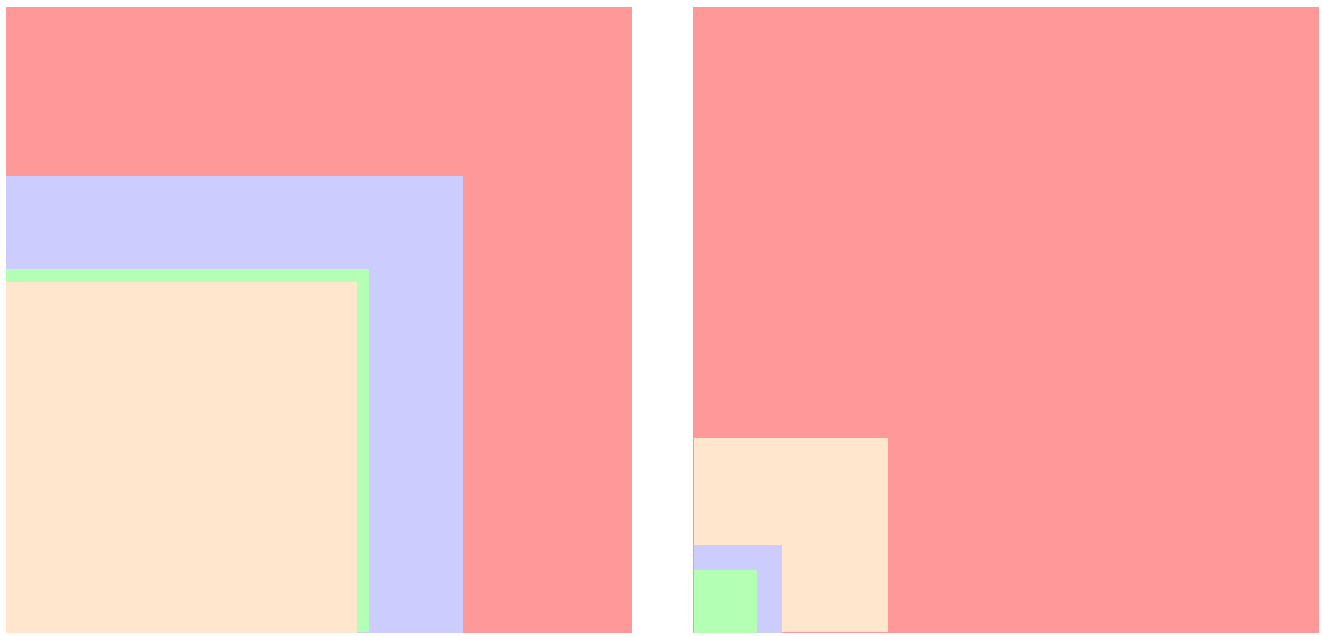
Dual transition storage format

lookup: bsearch or linear scan





	Input size		Compressed size (MB)		
Input	MB	Terms	Lucene	morf.	gzip
Wikipedia t.index	481	38 092 045	258	164	149
Polish infl.	162	3 672 200	3.1	1.7	15.4



Use Cases:
Solr's Autocomplete

Solr's

Suggesters

Design choices

sort order (alpha, score), prefix vs. spelling, boost exact matches?

Weights

term→weight, lookup(term, onlyMorePopular)

org.apache.solr.spelling.suggest.Lookup

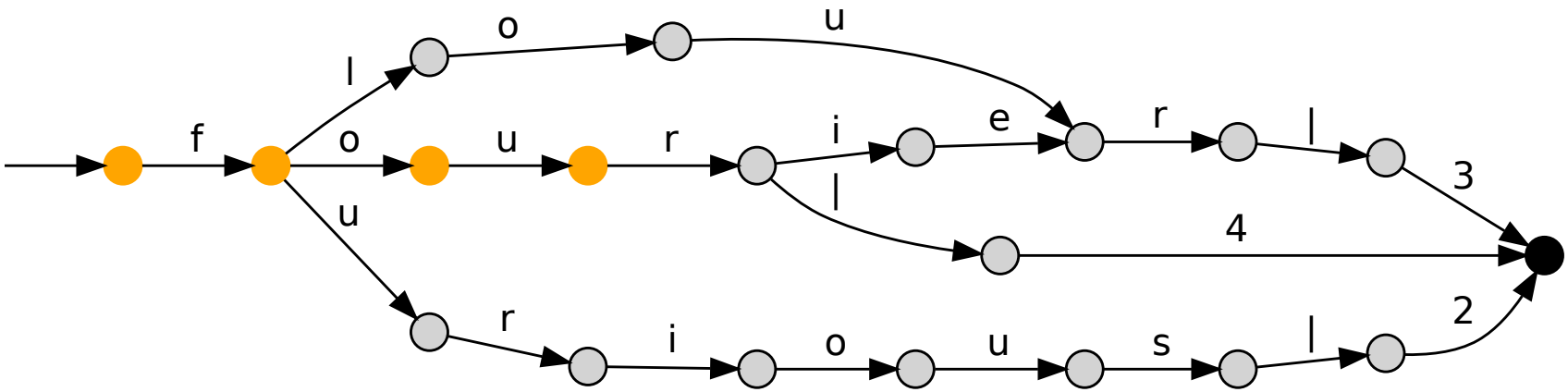
JaspellLookup, TSTLookup, FSTLookup

```
flour|3
four|4
fourier|3
furious|2
```

Take 1

```
flour|3
four|4
fourier|3
furious|2
```

→fou*



Find prefix.
Depth-in traversal for completions.
PQ on score|alpha

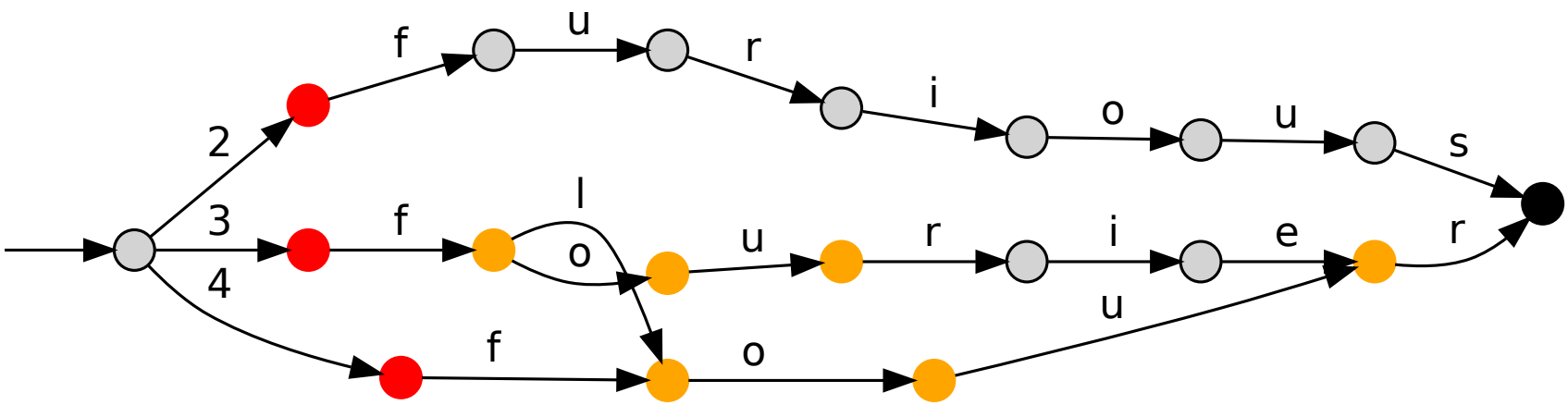
Take 1

2furious
3flour
3fourier
4four

Take 2

```
2furious
3flour
3fourier
4four
```

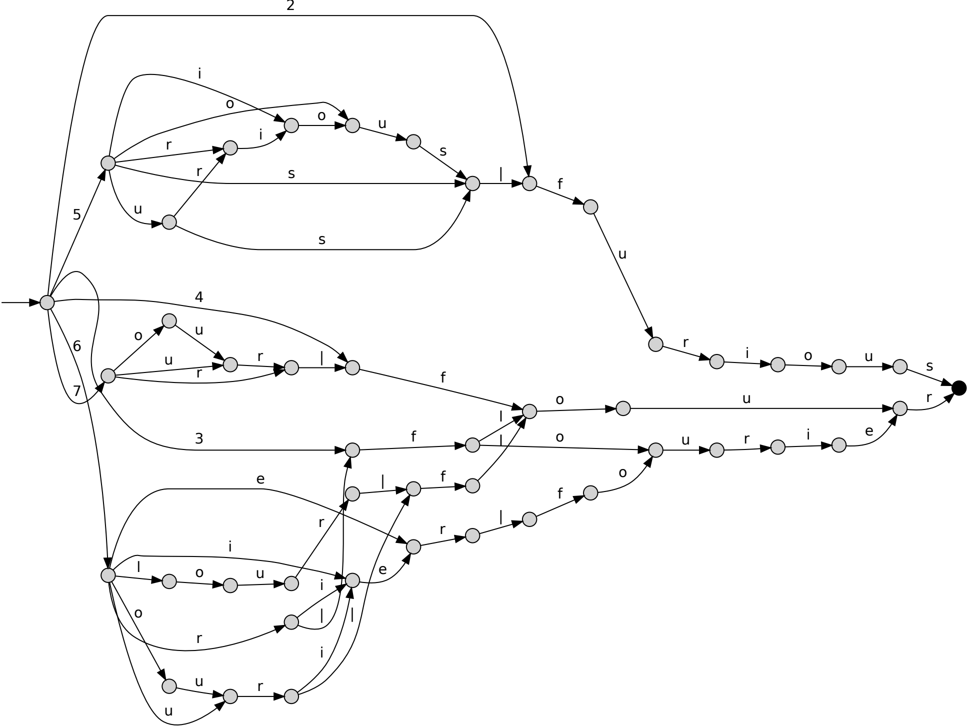
→fou*



From score roots, until N collected.
Find prefix.
Depth-in traversal for completions, stop if N collected.
Find/boost exact match.

Take 2

```
2furious
5urious|furious
5rious|furious
5ious|furious
5ous|furious
5us|furious
5s|furious
3flour
…
```


2
i
o
o
u
r
i
s
r
s
|
f
u
5
s
4
u
o
u
r
|
6
u
r
i
o
u
s
r
7
f
o
u
r
|
3
f
o
|
u
r
i
e
e
|
f
r
|
f
o
i
l
o
u
r
e
i
o
|
r
|
i
u
r
u

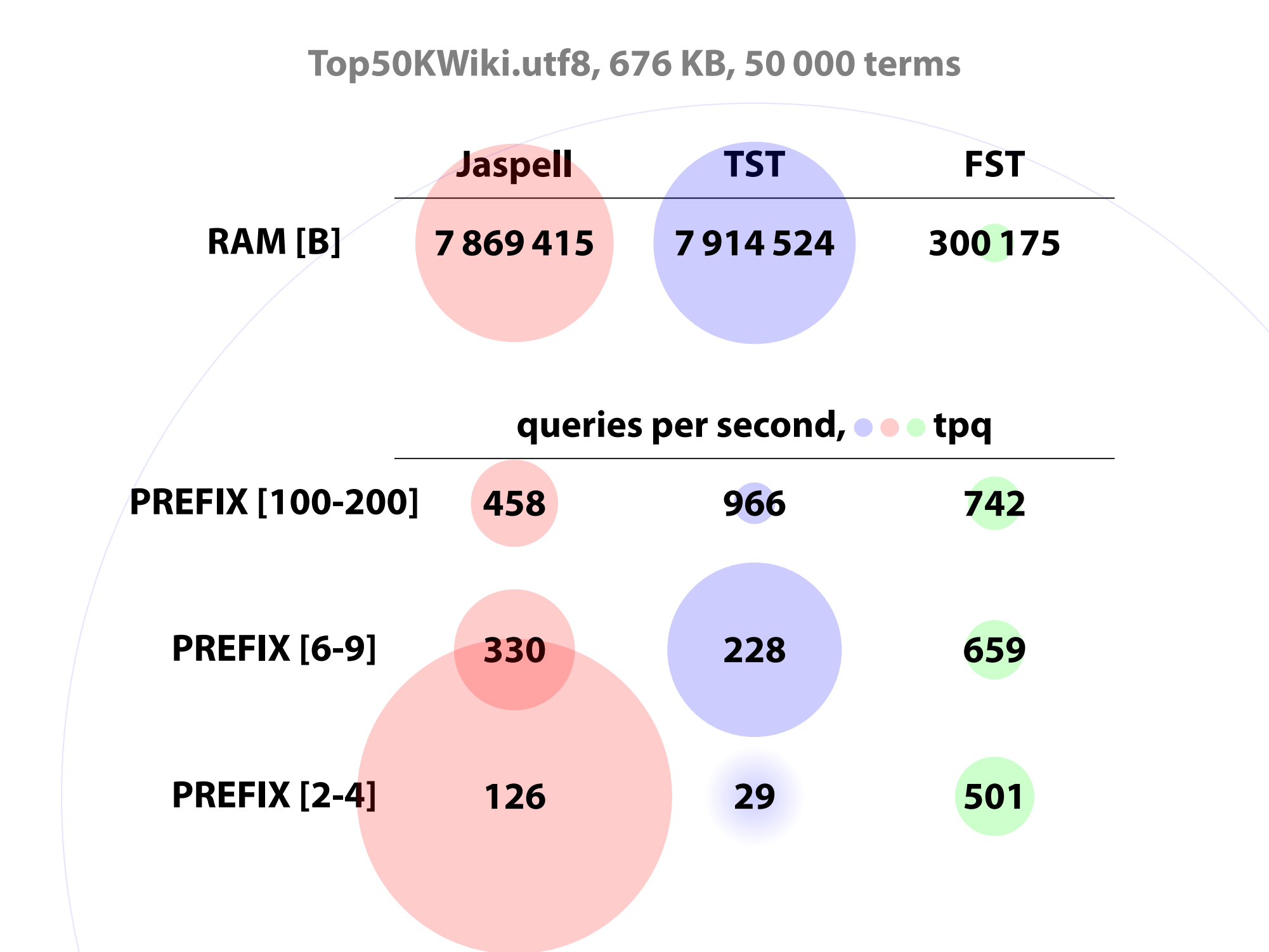
Constant time lookups!

Regardless of the terms dictionary size.
Regardless of prefix length.

Constant time lookups!

Regardless of the terms dictionary size.
Regardless of prefix length.

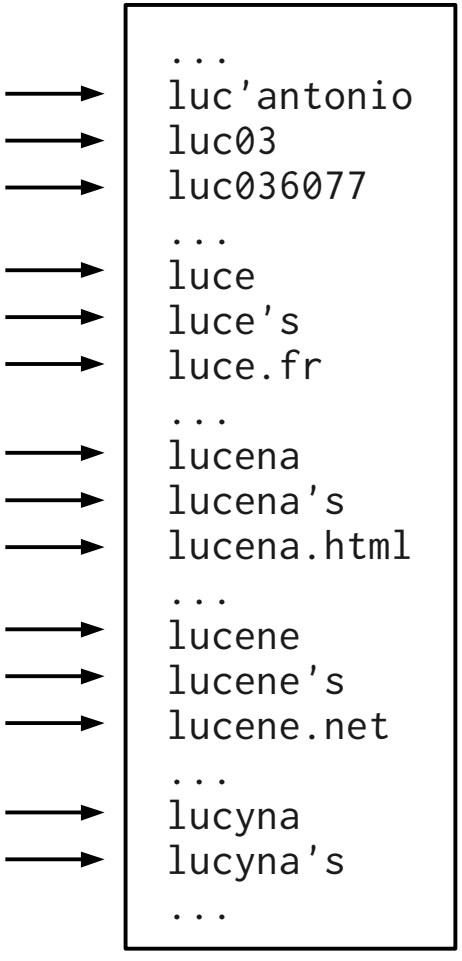
Exact matches only.
Static snapshot (not incremental).
Discretized weights.


Top50KWiki.utf8, 676 KB, 50 000 terms
Jaspell
TST
FST
RAM [B]
7 869 415
7 914 524
300 175
queries per second,
tpq
PREFIX [100-200]
458
966
742
PREFIX [6-9]
330
228
659
PREFIX [2-4]
126
29
501

Use Cases:
Automaton Queries

Fuzzy Matches

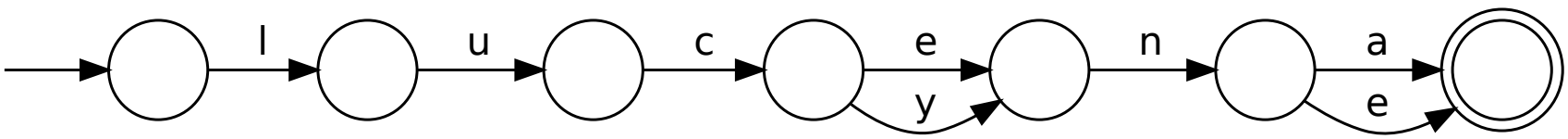
In the Past



All index terms scanned

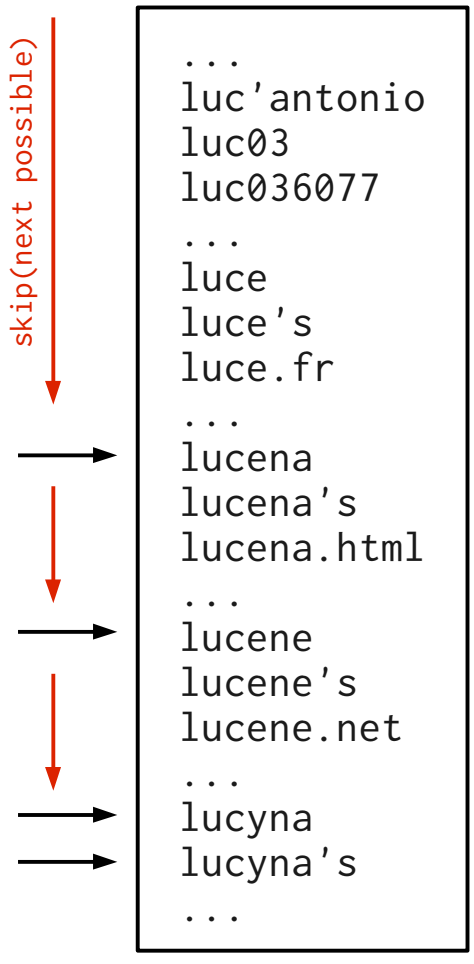
Cost grows with terms dict

luc([ey])n([ea])



Fuzzy Matches

Now



Skip to possible FSA paths

Unless impossible (loops)

AutomatonTermsEnum

(Automaton|Regexp|Wildcard)Query

Fuzzy Matches with

Levenshtein Edit Distance


Chuck Norris

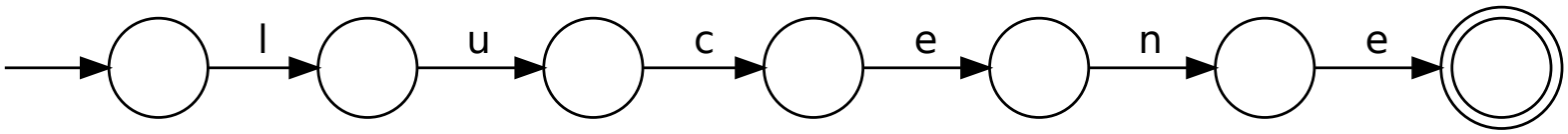
Fuzzy Matches with

Levenshtein Edit Distance

Fuzzy Matches with

Levenshtein Edit Distance

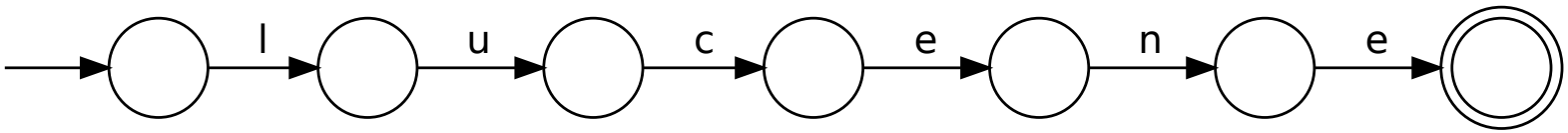
L_0(lucene):



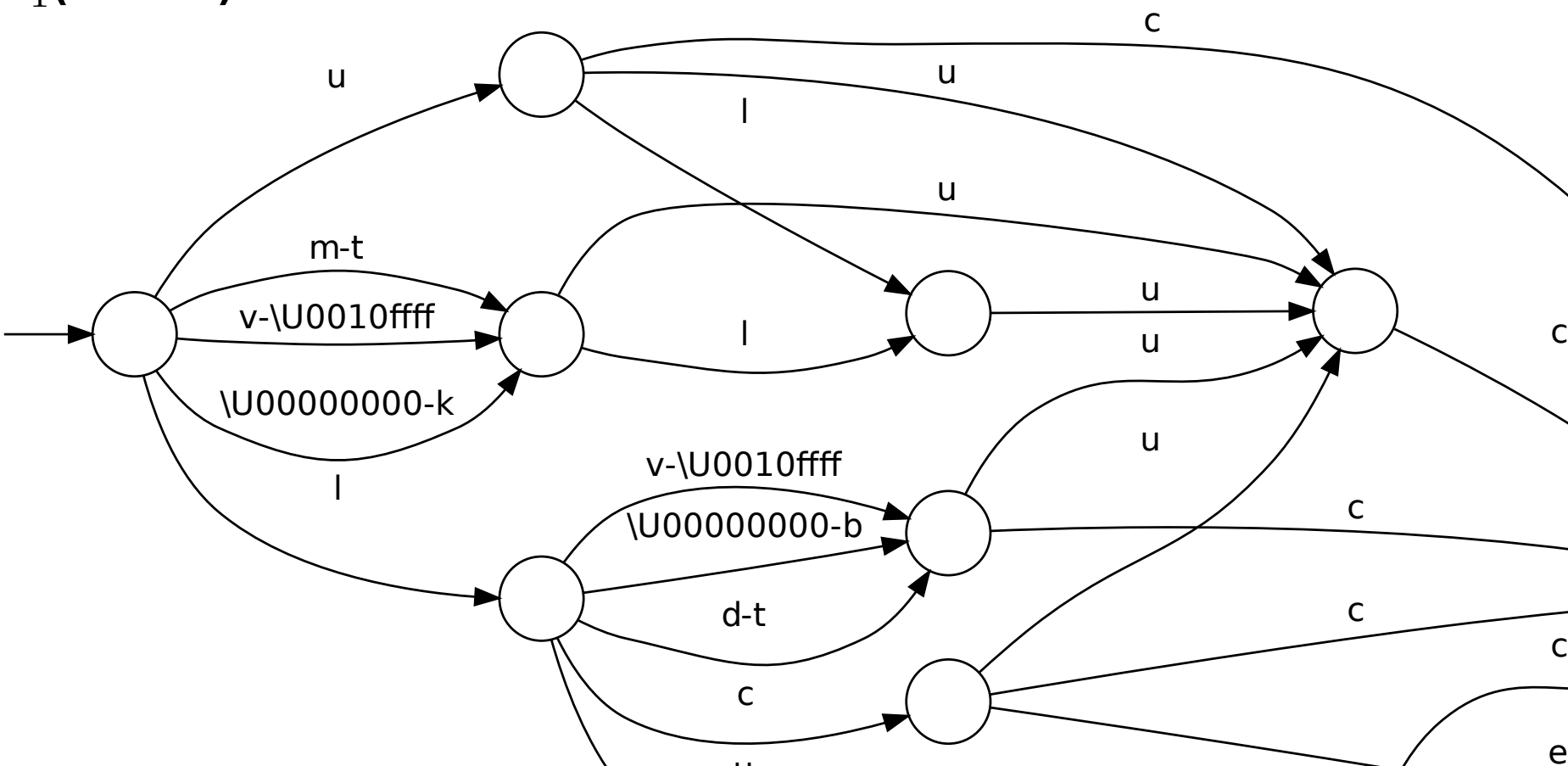
Fuzzy Matches with

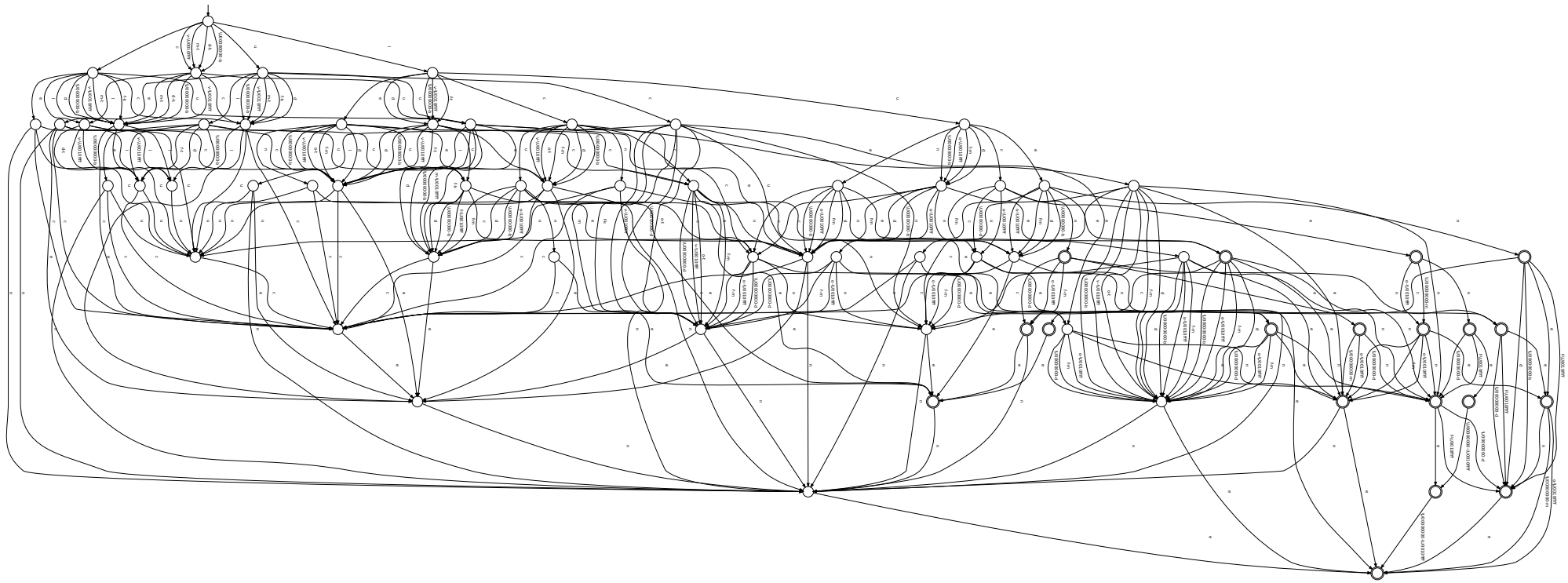
Levenshtein Edit Distance

L_0(lucene):



L_1(lucene):





L_2(lucene)

Automaton and Fuzzy Matches

Future

Intersection

Linear wrt automaton size

Prefix FSA for terms index

LUCENE-3030 → Mike McCandless

Summary

Summary and Conclusions

Automata

compact, powerful, efficient data structure

Lucene/Solr benefits

behind the scenes, but spreading: index, queries, suggesters

API in Lucene

…is shaped right now, still @experimental

Acknowledgement

Michael McCandless

Robert Muir

committer: .+

dawid.weiss@carrotsearch.com