https://vvtesh.sarahah.com/

# Information Retrieval

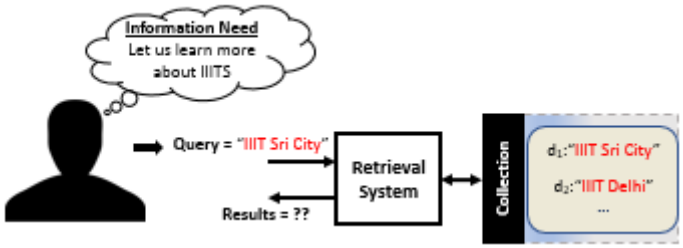
Venkatesh Vinayakarao

Term: Aug – Sep, 2019
Chennai Mathematical Institute

**The study of mathematics, like the Nile, begins in minuteness but ends in magnificence.**
Charles Caleb Colton.

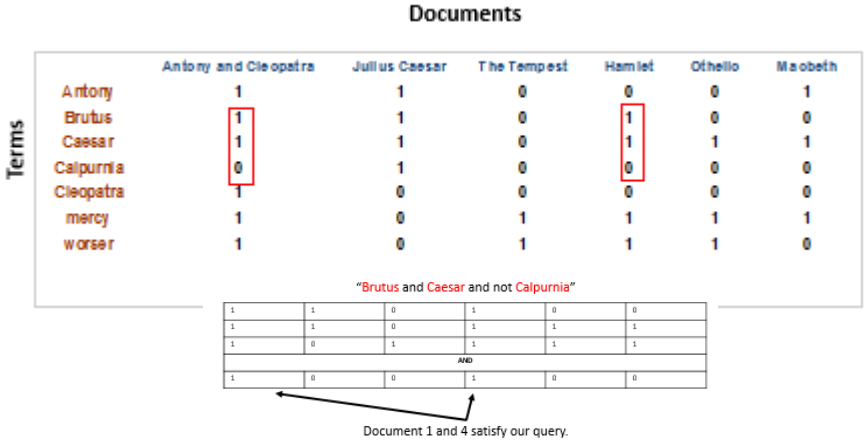
**The endless study of information retrieval, like the Taj Mahal, begins in magnificence and stays in magnificence.**
Venkatesh Vinayakarao.

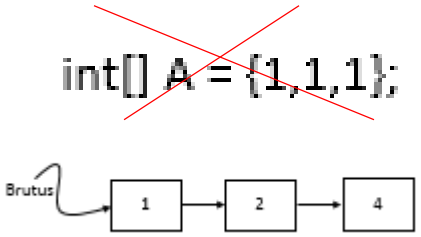
# Review



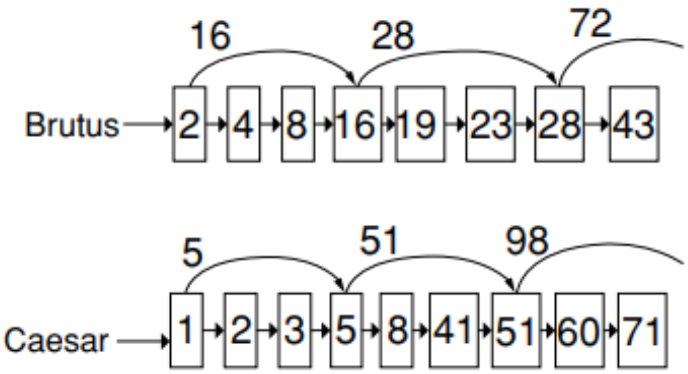

## One (bad) Approach

- First match the **term** IIIT.
  - Filter out documents that contain this term.
- Next match the **term** Sri.
  - Filter out documents that contain this term.
- Next match the **term** City.
  - Filter out documents that contain this term.



# Review




**Skip Pointers**

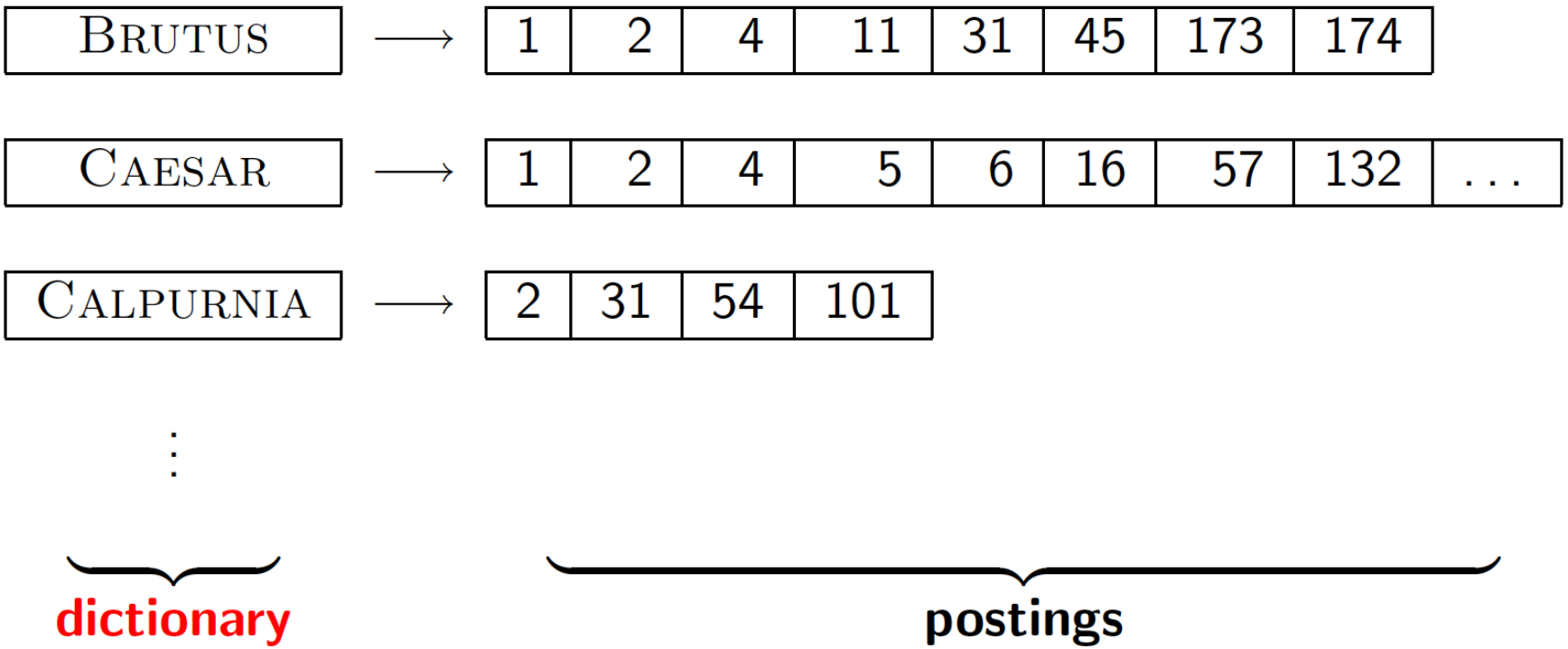

**Content Processing**

$$\text{Precision } P = tp/(tp + fp)$$

$$\text{Recall } \quad R = tp/(tp + fn)$$

**Evaluation**

# How to Store a Dictionary?



# One (bad) Approach

- Store them all in a file.
- Go linearly (one by one) and compare.

Avg. no. of Comparisons $\propto$ No. of Words in Dictionary

# Second (still bad) Approach

- Sort them.
- Do a binary search.

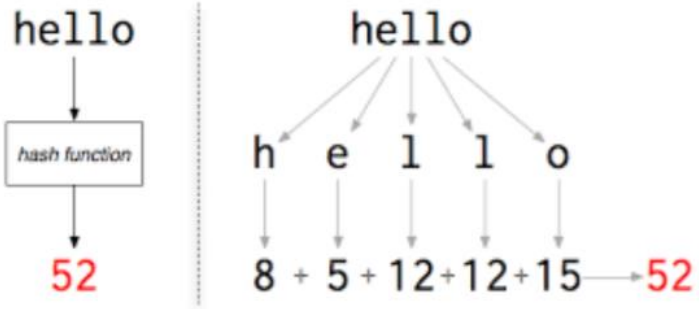
Avg. no. of Comparisons ∝ Log(No. of Words in Dictionary)

Can we do better?

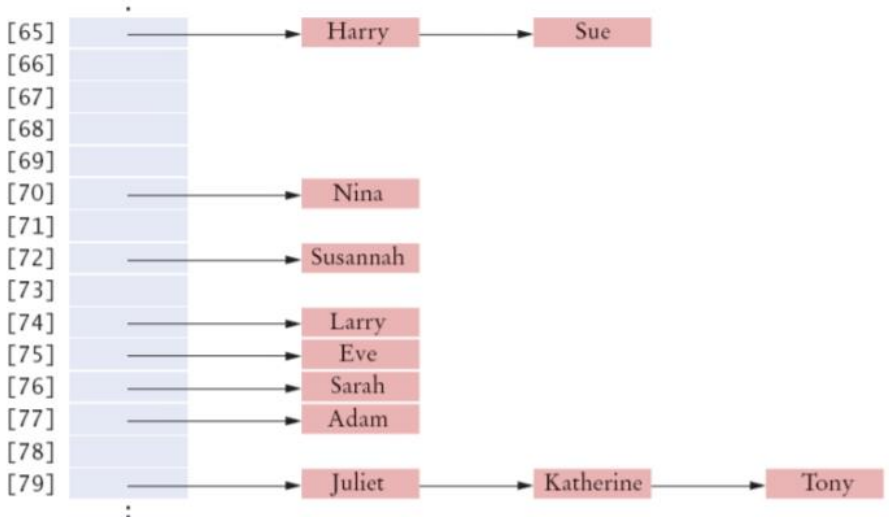
How can we store all dictionary words for a fast look up?

# Hashing

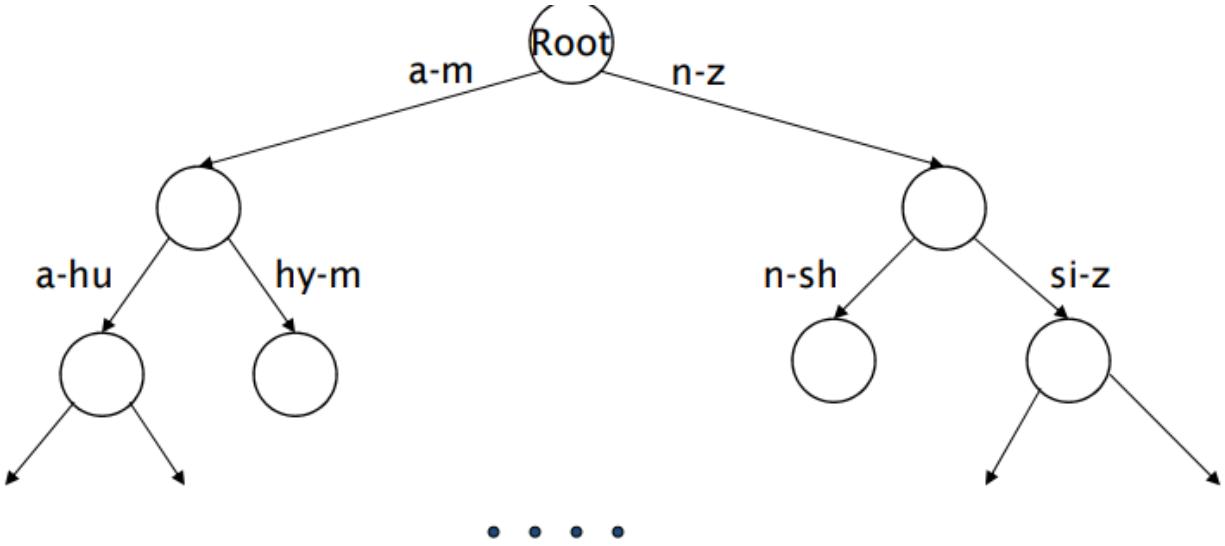
Simple hash function




Simple hash table



# Binary Search Tree

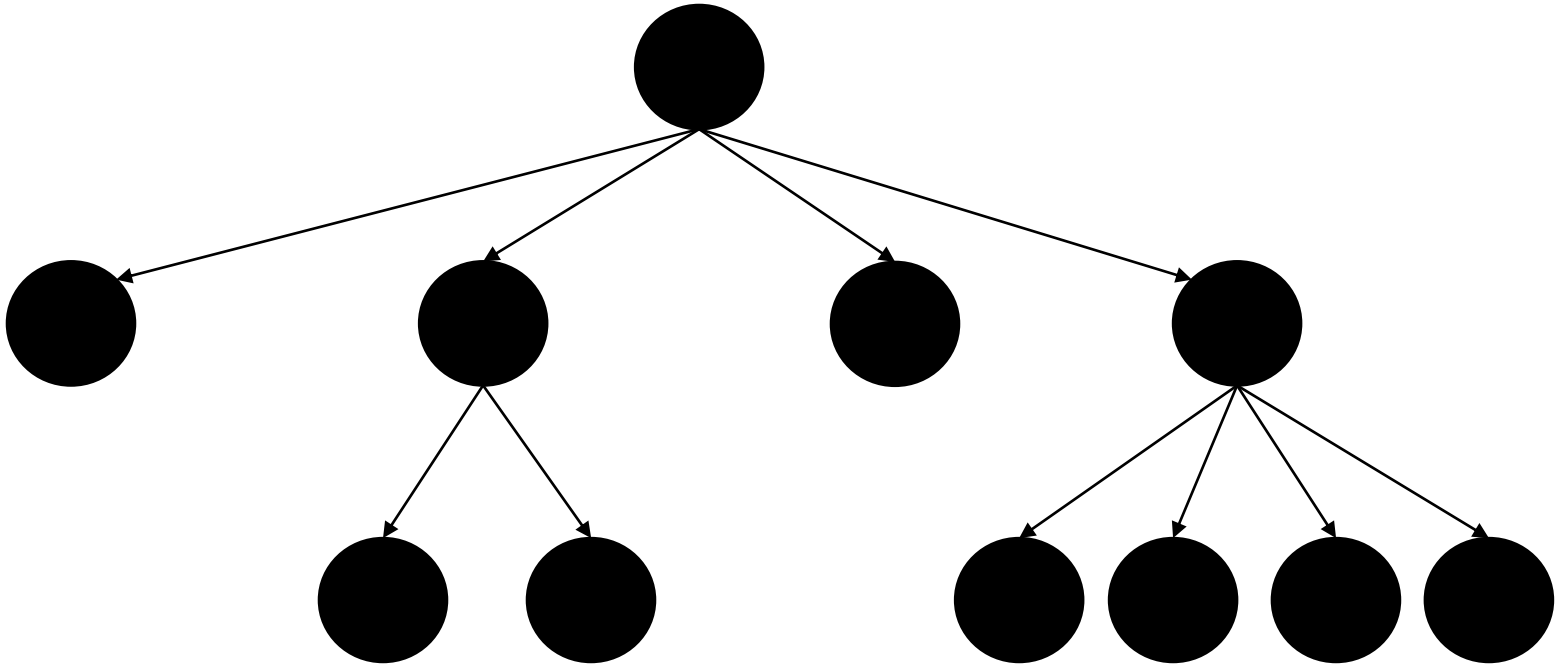



Dictionary terms are stored in the leaf nodes.

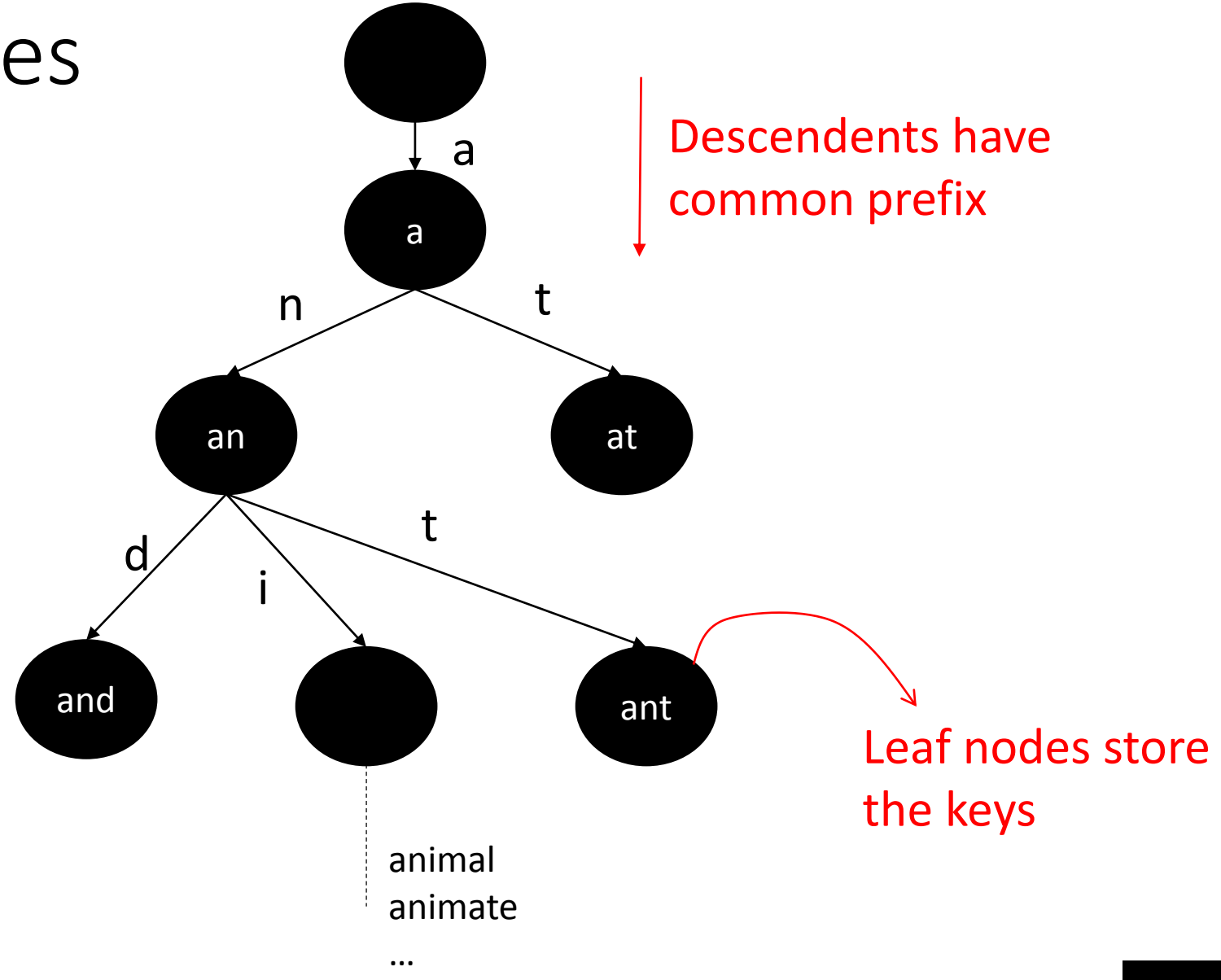
**Balancing this tree is a concern. Can we do better?**

# B-Trees

- Nodes may have any number of children in the interval [a,b]



# Tries


a
a
n
t
an
at
d
i
t
and
ant
animal
animate
…
Descendents have common prefix
Leaf nodes store the keys

# Wild-card queries: *

- ***mon*:*** retrieve all words in range: ***mon ≤ w < moo***
- ****mon:*** find words ending in "mon": harder
  - Maintain an additional B-tree for terms *backwards.*

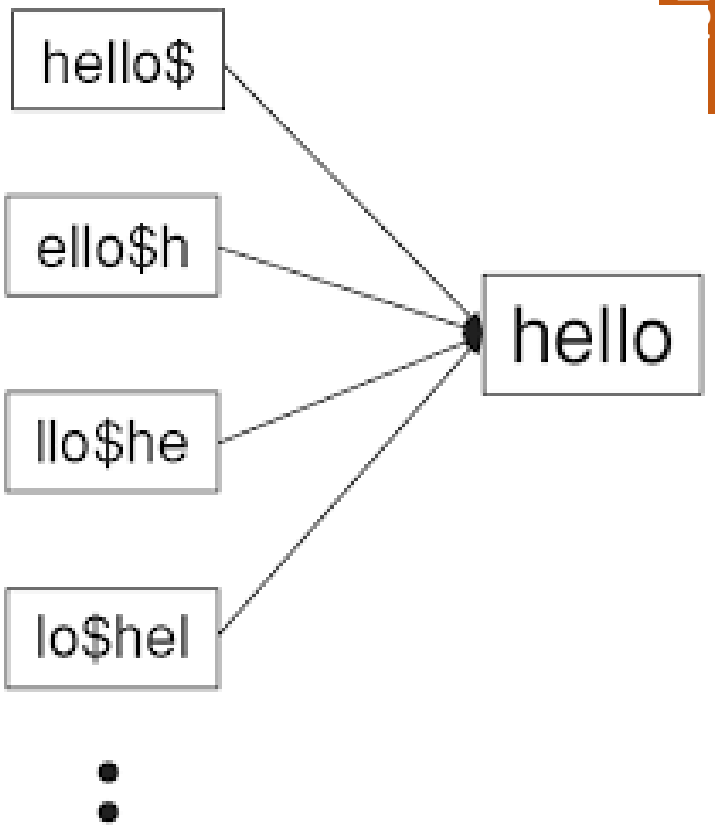
  Can retrieve all words in range: ***nom ≤ w < non****.*

# B-trees handle *'s at the end of a query term

- How can we handle *'s in the middle of query term?
  - ***co*tion***
- We could look up ***co**** AND ****tion*** in a B-tree and intersect the two term sets
  - Expensive
- The solution: transform wild-card queries so that the *'s occur at the end
- This gives rise to the **Permuterm** Index.

# How to Match he*lo?

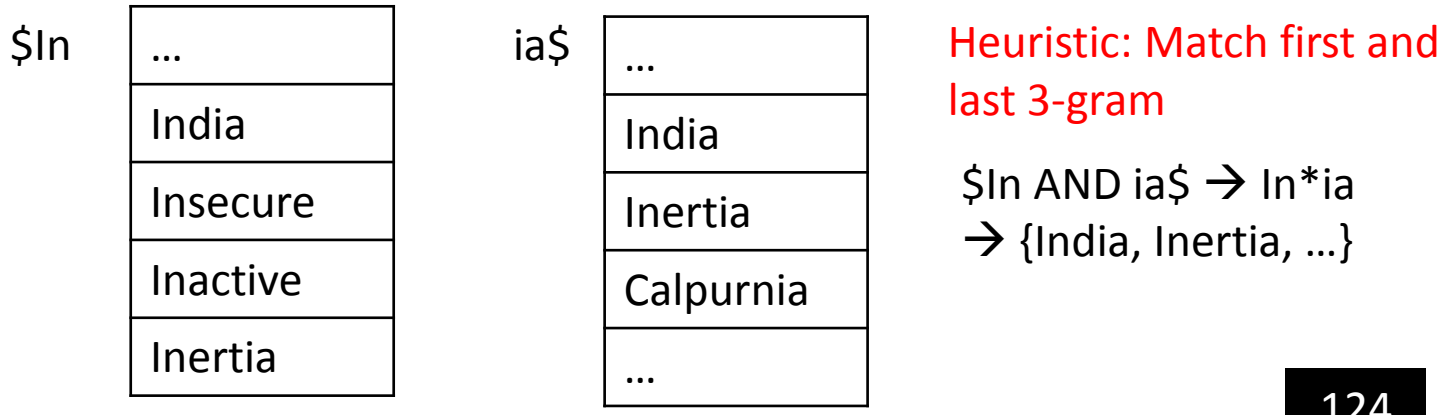
- Rotate he*lo → he*lo$ → $he*lo → o$he*l → lo$he*
- Till * is at the end.
- Exercise: How will you match h*l*o?




Clue: Enumerate all matches for o$h*. Check if they contain l.

# k-gram Index for Wildcard Queries

- k-gram is a sequence of k characters.
- 3-grams in “India” are Ind, ndi, dia.
- We add a $ to mark beginning and ending of the term.
- Therefore, “India” → {“$In”, “ndi”, “dia”, “ia$”}



# Test Your Understanding

- Can we define a 3-gram conjunctive query for red*?
  - $re AND red does not match red*.
  - We need a postfiltering step to check the results for red*.