The Wavelet Trie: Maintaining an Indexed Sequence of Strings in Compressed Space

CSI 5335 Paper presentation

Roberto Grossi, Giuseppe Ottaviano

presented by: Petr Praus

April 26th, 2012

Use case

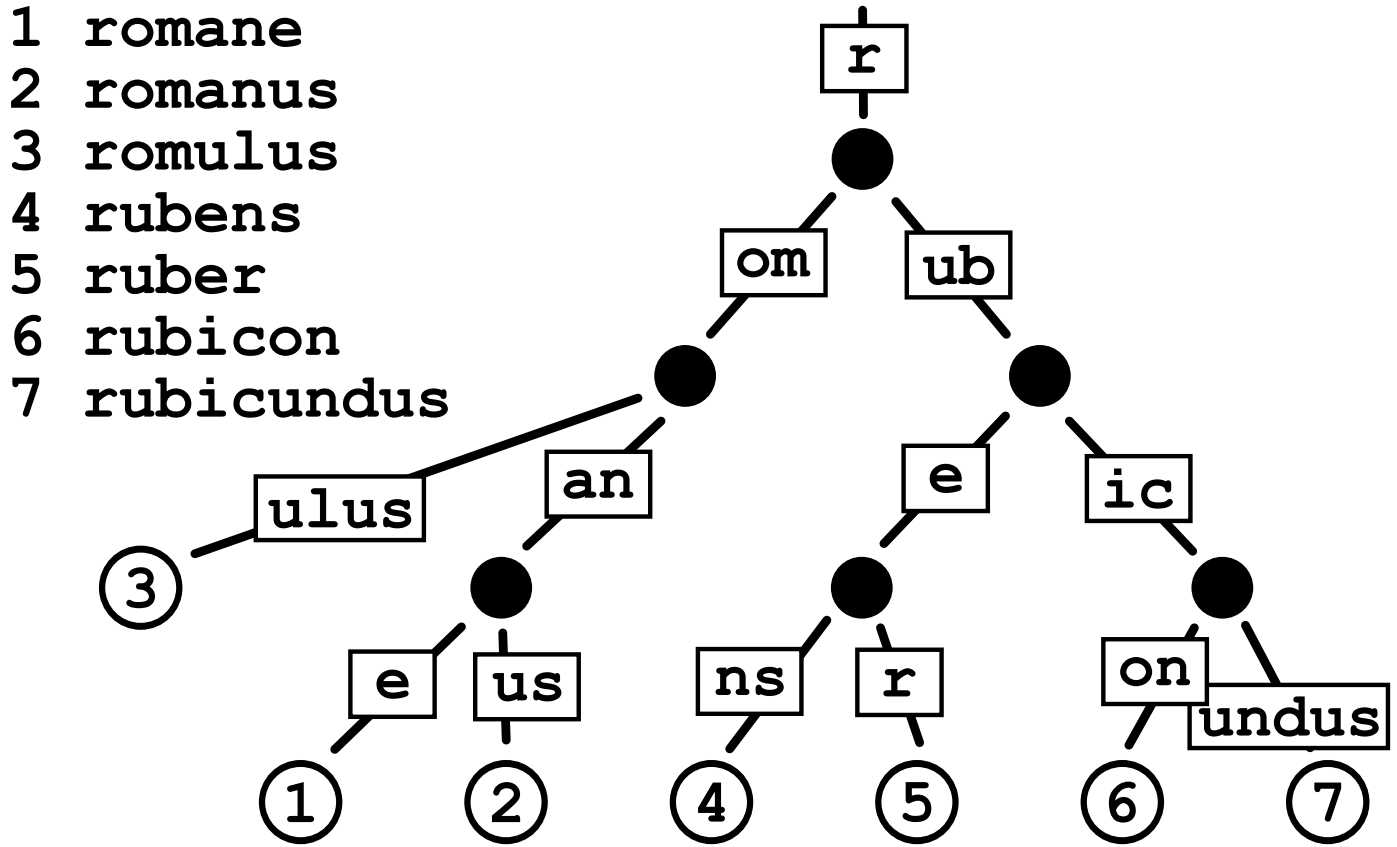
- A lot of things are string sequences.
- Column databases store and index string sequences.
- Great example: access logs

```
188.26.52.117 - - [24/Apr/2013:03:35:48 -0500] "GET /img/welcome/corner.png
188.26.52.117 - - [24/Apr/2013:03:35:49 -0500] "GET /img/welcome/arrowDown.gif
188.26.52.117 - - [24/Apr/2013:03:35:49 -0500] "GET /img/welcome/regionals.jpg
```

- Pretty similar, huh?
- I heard indexes make stuff faster $\rightarrow$ indexed sequence of strings
- Rank query: Number of requests for `/img/welcome/corner.png`?
- Select query: Position of i-th occurrence of `/img/welcome/corner.png`

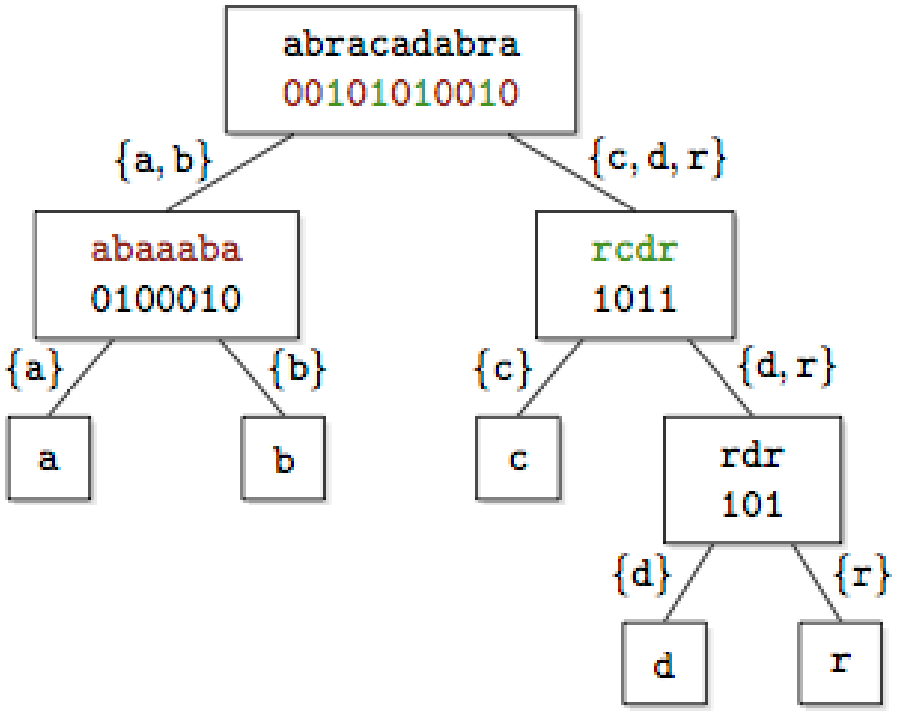
Patricia Trie

- Space-efficient trie.
- Node has always has at least two children.



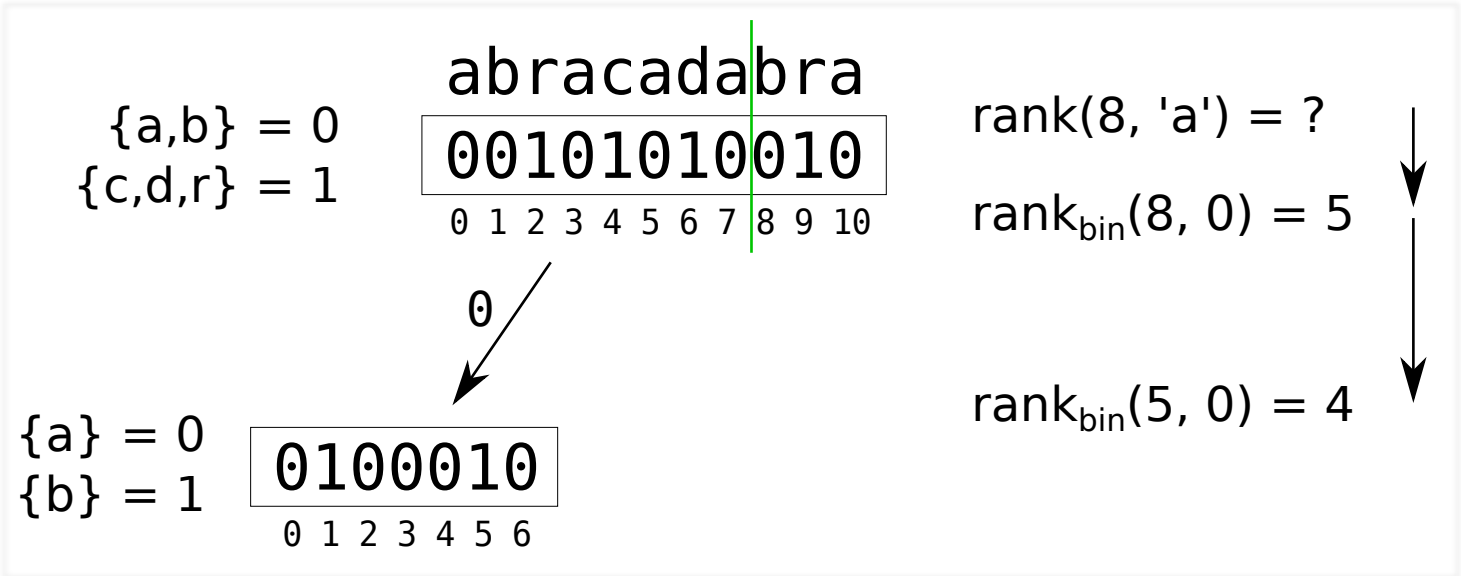
Wavelet Tree

- Organizes a string into a balanced binary tree of bit vectors.
- Alphabet: $\sum = \{a, b, c, d, r\} \rightarrow \{0, 0, 1, 1, 1\}$
- At root, we have *ambiguity*, reducing ambiguity towards leaves



Efficient computation of rank in Wavelet Tree

- $rank(8, a)$ = how many a's before position 8
- $rank_{bin}(pos, s)$ binary rank, # of occurences of s before pos



E.g.: # of requests to `/img/welcome/corner.png` before April 10th.

Summary

Thank you.

Resources

- http://alexbowe.com/wavelet-trees/