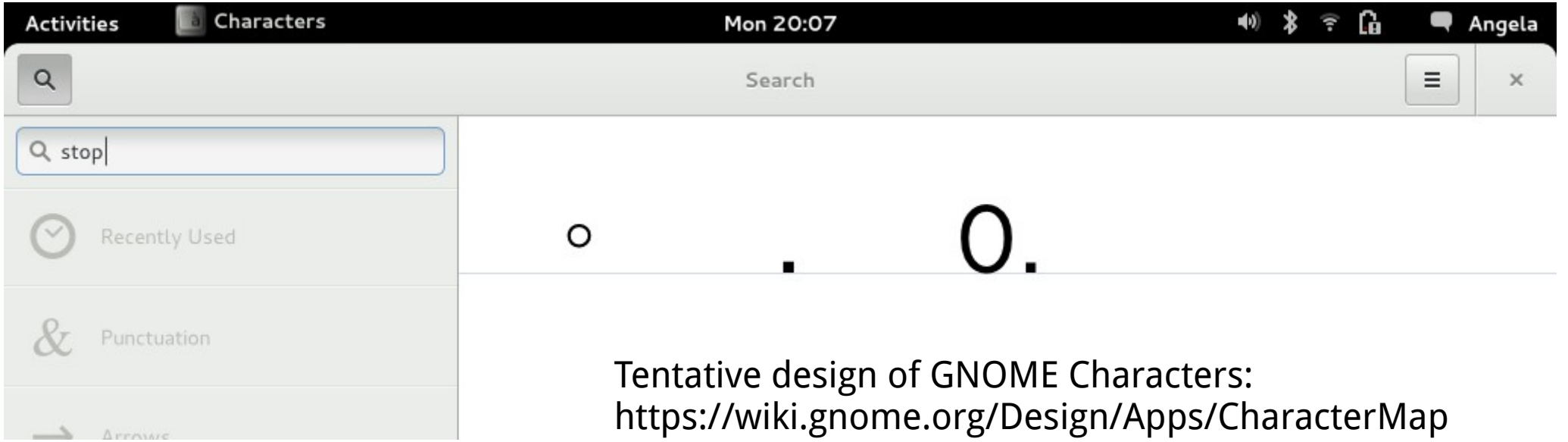
Compact and substring-indexed Unicode character database using suffix array

Daiki Ueno

The problem

- Look up a character from UnicodeData.txt, with a substr of the character name as a key
- Show multiple matches instantly



Tentative design of GNOME Characters:
https://wiki.gnome.org/Design/Apps/CharacterMap

Prior art: gucharmap

- Embeds UnicodeData.txt in the binary
- No compression
  - libgucharmap.so: 4MB
- Linear search / single result at a time

Prior art: GNU libunistring

- Embeds UnicodeData.txt in the binary
- Compression
  - Remove dupes of words
    - e.g. “KATAKANA LETTER SMALL A”
      - “KATAKANA”, “LETTER”, “SMALL”, “A”
      - Assign unique integer value to each word
      - Like Java's constant pool
  - Less than 350KB
- Only supports exact match

The idea

- Based on the libunistring approach
- Use suffix array to allow substr match

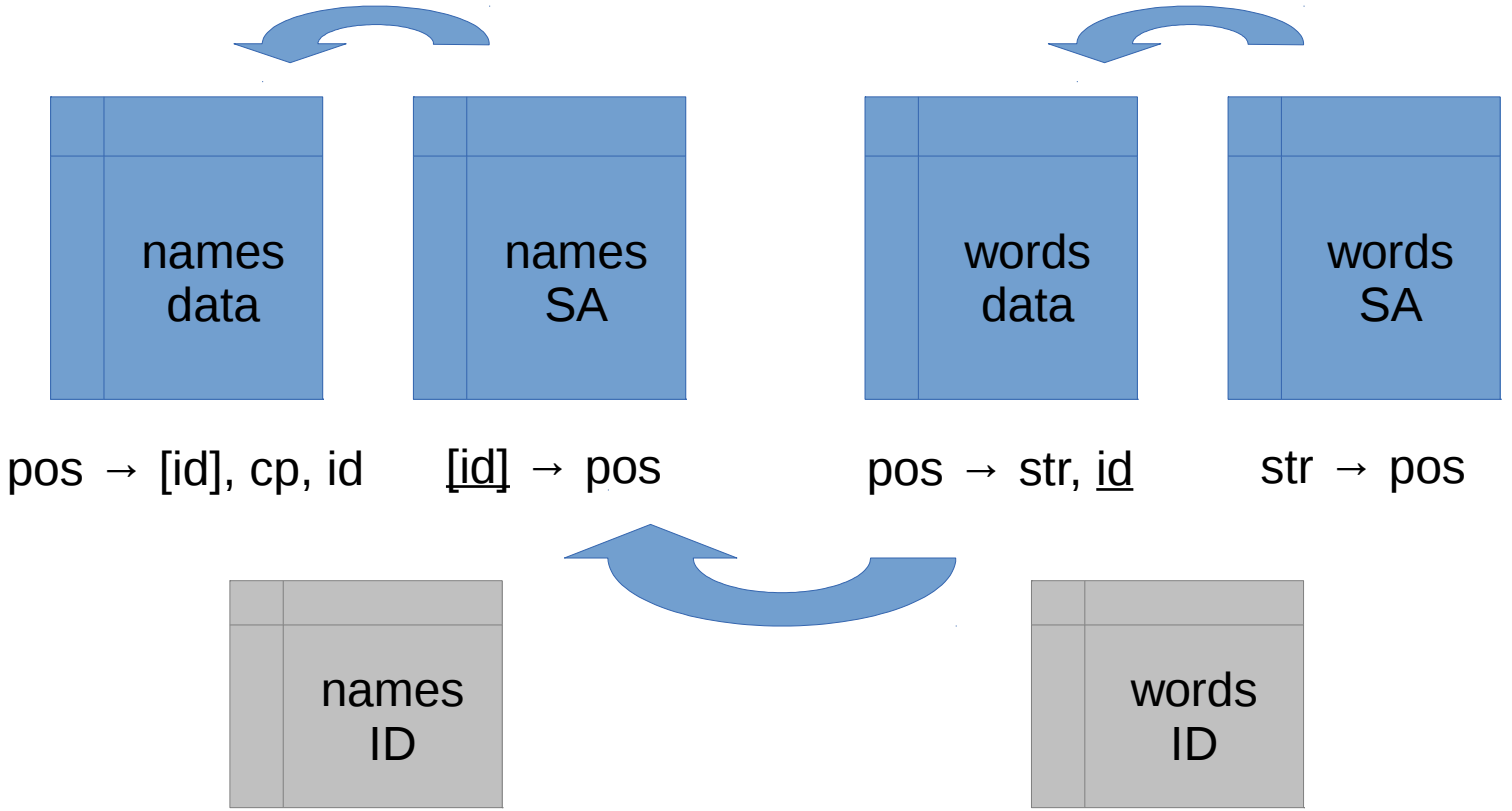
Suffix array

- Simple data structure which allows substr match
- e.g. “abracadabra$”
  - $ = terminal symbol
  - Sort each possible suffixes
  - Remember starting indices

index	SA	suffix
0	11	$
1	10	a$
2	7	abra$
3	0	abracadabra$
4	3	acadabra$
5	5	adabra$
6	8	bra$
7	1	bracadabra$
8	4	cadabra$
9	6	dabra$
10	9	ra$

Two suffix arrays

- For words and names
- First look up in words and then look up in names



Prototype in Python

- Unicode 6.3.0
  - 1.4MB in text, 10,446 characters
  - 24,345 named characters out of 24,434
  - http://www.unicode.org/Public/6.3.0/ucd/UnicodeData.txt
- Sizes

	data	SA	ID	total
words	100K	204K	32K	336K
names	208K	284K	72K	564K
total	308K	488K	104K	900K

Conclusion

- Not too bad
  - 2.5x larger than data + ID
  - but probably I'm missing something
    - e.g. special treatment for Hangul syllables
- I don't have time to implement that actually...
  - Feel free to take the task if interested...