String sorting problem

Sort a set $R = \{s_1, s_2, \ldots, s_n\}$ of n (non-empty) strings into the lexicographic order.

Size of input

- $N =$ total length of strings
- $D =$ total length of distinguishing prefixes

Some Notation:

- $s = s[0] \ldots s[|s|-1]$
- $\forall c \in \Sigma : s[|s|] > c$ (special sentinel character)

Distinguishing prefix

The distinguishing prefix of string s in R is

- shortest prefix of s that is not a prefix of another string (or s if s is a prefix of another string)
- shortest prefix of s that determines the rank of s in R

```
alignment
all
allocate
alphabet
alternate
alternative
```

Distinguishing prefix

The distinguishing prefix of string s in R is

- shortest prefix of s that is not a prefix of another string (or s if s is a prefix of another string)
- shortest prefix of s that determines the rank of s in R

A sorting algorithm needs to access

- every character in the distinguishing prefixes
- no character outside the distinguishing prefixes

```
alignment
all
allocate
alphabet
alternate
alternative
```

Alphabet model

Ordered alphabet

- only comparisons of characters allowed

Constant alphabet

- ordered alphabet of constant size
- multiset of characters can be sorted in linear time

Integer alphabet

- alphabet is $\{1,\ldots,\sigma\}$ for integer $\sigma \geq 2$
- multiset of k characters can be sorted in $O(k+\sigma)$ time

Lower bounds

alphabet	lower bound
ordered	$\Omega(D + n \log n)$
constant	$\Omega(D)$
integer	$\Omega(D)$

Standard sorting algorithm

- $\Theta(n \log n)$ string comparisons

Let $s_i = \alpha\beta_i$, where $|\alpha| = |\beta_i| = \log n$

- $D = \Theta(n \log n)$
- lower bound:
  $\Omega(D + n \log n) = \Omega(n \log n)$
- standard sorting:
  $\Theta(n \log n) \cdot \Theta(\log n) = \Theta(n \log^2 n)$

```
aaaaaa  aaaaab
aaaaba  aaaabb
aaabaa  aaabab
aaabba  aaabbb
```

Multikey quicksort *[Bentley & Sedgewick ’97]*

- ternary partition
- on one character at a time

<	=	>

al	p	habet
al	i	gnment
al	l	ocate
al	g	orithm
al	t	ernative
al	i	as
al	t	ernate
al	l	

$\Longrightarrow$

al	i	gnment
al	g	orithm
al	i	as
al	l	ocate
al	l	
al	p	habet
al	t	ernative
al	t	ernate

Multikey quicksort ***[Bentley & Sedgewick ’97]***

Multikey-quicksort(R, ℓ) // $R =$ set of strings with
// common prefix of length ℓ

if $|R| \leq 1$ then return R

choose pivot $p \in R$

$R_< := \{s \in R \mid s[\ell+1] < p[\ell+1]\}$
$R_= := \{s \in R \mid s[\ell+1] = p[\ell+1]\}$
$R_> := \{s \in R \mid s[\ell+1] > p[\ell+1]\}$

Multikey-quicksort($R_<$, ℓ)
Multikey-quicksort($R_=$, $\ell+1$)
Multikey-quicksort($R_>$, ℓ)

return $R_<R_=R_>$

Multikey quicksort: Analysis

- comparisons in partitioning step dominate runtime

if $|R| \leq 1$ then return R

choose pivot $p \in R$

$R_< := \{s \in R \mid s[\ell+1] < p[\ell+1]\}$
$R_= := \{s \in R \mid s[\ell+1] = p[\ell+1]\}$
$R_> := \{s \in R \mid s[\ell+1] > p[\ell+1]\}$

Multikey-quicksort($R_<$, ℓ)
Multikey-quicksort($R_=$, $\ell+1$)
Multikey-quicksort($R_>$, ℓ)

return $R_<R_=R_>$

Multikey quicksort: Analysis

- If $s[\ell+1] \neq p[\ell+1]$, charge the comparison on s
  - assume perfect choice of pivot
  - size of the set containing s is halved
  - total charge on s is $\leq \log n$
  - total number of $\neq$-comparisons is $\leq n \log n$

3 $R_< := \{s \in R \mid s[\ell+1] < p[\ell+1]\}$
$R_= := \{s \in R \mid s[\ell+1] = p[\ell+1]\}$
$R_> := \{s \in R \mid s[\ell+1] > p[\ell+1]\}$

Multikey quicksort: Analysis

- If $s[\ell+1] = p[\ell+1]$, charge the comparison on $s[\ell+1]$
  - $s[\ell+1]$ becomes part of common prefix
  - total charge on $s[\ell+1]$ is ≤ 1
  - total number of $=$-comparisons is $\leq D$

3 $R_< := \{s \in R \mid s[\ell+1] < p[\ell+1]\}$
$R_= := \{s \in R \mid s[\ell+1] = p[\ell+1]\}$
$R_> := \{s \in R \mid s[\ell+1] > p[\ell+1]\}$
4 Multikey-quicksort($R_<$, ℓ)
5 Multikey-quicksort($R_=$, $\ell+1$)

Multikey quicksort: Analysis

- comparisons in partitioning step dominate runtime
- If $s[\ell+1] \neq p[\ell+1]$, charge the comparison on s
  - assume perfect choice of pivot
  - size of the set containing s is halved
  - total charge on s is $\leq \log n$
  - total number of $\neq$-comparisons is $\leq n \log n$
- If $s[\ell+1] = p[\ell+1]$, charge the comparison on $s[\ell+1]$
  - $s[\ell+1]$ becomes part of common prefix
  - total charge on $s[\ell+1]$ is ≤ 1
  - total number of $=$-comparisons is $\leq D$
- $O(D + n \log n)$ time