String-searching Algorithms

Suffix Array

Sergio Peignier sergio.peignier@insa-lyon.fr

Associate Professor
INSA Lyon
Biosciences department

Naive Suffix Array Construction

Given a string $S \in \mathcal{S}$

- Extract all the suffixes of S
- Sort them
- Return a **suffix array**, i.e., list of suffixes (and their indexes)

Complexity: $\sim \mathcal{O}(|S|^2 \ \log |S|)$

DC3 algorithm | Overview

DC3: **D**ifference **C**over size **3**

- **Recursive** algorithm of **suffix array** construction
- **Linear in time**
- Relies on **radix sort** (efficient for lists of numbers)
- String **characters** should be **converted in numbers**
- Main steps:
  1. **Divide** suffixes into **two groups**
     Sort first group
  2. **Use first** group to **sort second** one
  3. **Merge groups** to find suffix array.

DC3 algorithm | Step 0

- Convert the string into a **numeric sequence** with alphabet $\mathcal{A} = \{0, 1, \ldots, A-1\}$, with $A = |\mathcal{A}|$
  we assume that $|\mathcal{A}|$ is small.

- Example of mapping: ASCII code
- Append **three sentinel numbers** at the end (here 0)

$S = abcabcacab \quad \rightarrow$

$T = (97, 98, 99, 97, 98, 99, 97, 99, 97, 98, 0, 0, 0)$

DC3 algorithm | Step 1

Given $T = (t[i] \ldots, \forall i \in \{0, \ldots, |T| - 1\} \mid t_i \in \mathcal{A})$

$P_k = (i \in \{0, \ldots, |T| - 1\} \quad | \quad i\%3 = k \textbf{ and } i + 2 < |T|)$:

- P_0 : Positions multiples of 3.
- P_1 : Positions following those in P_0
- P_2 : Positions following those in P_1
- $P_{1,2} = P_1 + P_2$ (concatenation of P_1 and P_2)

Goal: Sort **recursively** suffixes in indexes $P_{1,2}$.

$P_{1,2} = (1, 4, 7, 10, 2, 5, 8)$

DC3 algorithm | Step 1

Sequences of three numbers staring from indexes $P_{1,2}$:

$R_{1,2} = ((T[p], T[p+1], T[p+2]) , \dots , \forall p \in P_{1,2})$

$R_{1,2}$ = ((98, 99, 97), (98, 99, 97), (99, 97, 98), (0, 0, 0), (99, 97, 98), (99, 97, 99), (97, 98, 0))

Apply Radix sort to $R_{1,2}$:

$R_{1,2}^{sorted}$ = ((0, 0, 0), (97, 98, 0), (98, 99, 97) , (98, 99, 97), (99, 97, 98) , (99, 97, 98) , (99, 97, 99)

Order = (1, 2, 3, 3, 4, 4, 5)

Index = (10, 8, 1, 4, 7, 2, 5)

DC3 algorithm | Step 1

Make new sequence T' by replacing each triplet of $R_{1,2}$ by its corresponding order.

$R_{1,2}$ = ((98, 99, 97), (98, 99, 97), (99, 97, 98), (0, 0, 0), (99, 97, 98), (99, 97, 99), (97, 98, 0))

$P_{1,2} = (1, 4, 7, 10, 2, 5, 8)$

$Order_{1,2} = (1, 2, 3, 3, 4, 4, 5)$

$Index_{1,2} = (10, 8, 1, 4, 7, 2, 5)$

$T' = (3, 3, 4, 1, 4, 5, 2)$

Finding **suffix array** of T' is **equivalent** of finding $R_{1,2}$ **order**.

Apply **recursively** same operations to T', T'', ... until **no repeated characters** are in T^n

DC3 algorithm | Step 1

Apply **recursively** same operations to T', T'', ... until **no repeated characters** are in T^n

$T' = (3, 3, 4, 1, 4, 5, 2)$

$P'_{1,2} = (1, 4, 7, 2, 5)$

$R'_{1,2} = ((3,4,1), (4,5,2),(0,0,0),(4,1,4),(5,2,0))$

$R'^{sorted}_{1,2} = ((0,0,0),(3,4,1),(4,1,4),(4,5,2),(5,2,0))$

$Order'_{1,2} = (1, 2, 3, 4, 5)$

$Index'_{1,2} = (7, 1, 2, 4, 5)$

DC3 algorithm | Step 2

Find **order** associated to R'_0 given order of $R'_{1,2}$

$T' = (3, 3, 4, 1, 4, 5, 2)$, $P'_0 = (0, 3, 6)$, $Index'_{1,2} = (7, 1, 2, 4, 5)$

- R'_0 is built as pairs of the elements at index $P'_0 = (0, 3, 6)$ and the rank of the next item
  $R'_0 = ((3, 2), (1, 4), (2, 1))$
- Sort R'_0 pairs using **Radix**:
  $R'^{sorted}_0 = ((1, 4), (2, 1), (3, 2))$, $Index'_0$ = (3,6,0)

DC3 algorithm | Step 3

Merge $Index'_{1,2}$ and $Index'_0$ indexes

For $a \in Index'_0$ and $b \in Index'_{1,2}$:

- If $T'[a] < T'[b]$ or $T'[b] < T'[a]$: **Add smaller**
- If $T'[a] = T'[b]$: Compare $T'[a+1]$ and $T'[b+1]$ (knowing that $(a+1)\%3 = 1 \Rightarrow a+1 \in Index'_{1,2}$)
  - If $b\%3 = 1 \Rightarrow (b+1)\%3 = 2 \Rightarrow (b+1) \in Index'_{1,2}$: **Order known**
  - If $b\%3 = 2$: $(b+1)\%3 = 0$: **Compare**:
    If $T'[a+1] = T'[b+1]$: $(b+2)\%3 = 1$ and $(a+2)\%3 = 2$ thus $(b+1) \in Index'_{1,2}$ and $(a+2) \in Index'_{1,2}$: **Order known**

DC3 algorithm | Step 3

Merge $Index'_{1,2}$ and $Index'_0$ indexes

Example:

$T' = (3, 3, 4, 1, 4, 5, 2)$

$Index'_{1,2} = (7, 1, 2, 4, 5)$, $Index'_0 = (3, 6, 0)$

$\rightarrow Index'_{0,1,2} = (7, 3, 6, 0, 1, 2, 4, 5)$

Remove index corresponding to **centinel**

$\rightarrow Index'_{0,1,2} = (3, 6, 0, 1, 2, 4, 5)$

DC3 algorithm | Resume to higher order

Map indexes ($Index^{n}_{0,1,2} \rightarrow Index^{n-1}_{1,2}$)

$P_{1,2} = (1, 4, 7, 10, 2, 5, 8)$

$Index'_{0,1,2} = (3, 6, 0, 1, 2, 4, 5)$

The indexes in T' correspond to the locations of the indexes stored in $P_{1,2}$

$Index_{1,2} = (10, 8, 1, 4, 7, 2, 5)$

(Then run steps 2 and 3 and map to higher order ...)

DC3 algorithm | Step 2

$P_0 = (0, 3, 6, 9)$, $R_0 = ((97, 3), (97, 4), (97, 5), (98, 1))$

$R_0^{sorted} = ((97, 3), (97, 4), (97, 5), (98, 1))$, $Index_0 = (0, 3, 6, 9)$

DC3 algorithm | Step 3

Merge $Index_0 = (0, 3, 6, 9)$ and $Index_{1,2} = (10, 8, 1, 4, 7, 2, 5)$

$Index_{0,1,2} = (10, 8, 0, 3, 6, 9, 1, 4, 7, 2, 5)$

Remove index for sentinel triplet:

$Index_{0,1,2} = (8, 0, 3, 6, 9, 1, 4, 7, 2, 5) \leftarrow$ suffix array

DC3 algorithm | Summary

DC3 (input: T)

- Append sentinels to T
- Compute $R_{1,2}$ and $P_{1,2}$
- Sort $R_{1,2}$: find $Index_{1,2}$ and $Order_{1,2}$
- If $Order_{1,2}$ has repeated elements:
  - $T' \leftarrow$ Replace triplets by their order
  - $Index'_{0,1,2} \leftarrow DC3(T')$
  - $Index_{1,2} \leftarrow$ Resume $Index'_{0,1,2}$ to higher order
- Compute R_0 and P_0
- Sort R_0 : find $Index_0$ and $Order_0$
- $Index_{0,1,2} \leftarrow$ Merge Indexes $Index_0$ and $Index_{1,2}$
- Remove index associated to sentinel
- Return $Index_{0,1,2}$

Bibliography

- *https://www.cs.cmu.edu/afs/cs/academic/class/15210-s12/www/lectures/lecture25.pdf*
- *https://web.stanford.edu/class/archive/cs/cs166/cs166.1166/lectures/04/Small04.pdf*
- *https://people.csail.mit.edu/jshun/6886-s19/lectures/lecture9-1.pdf*
- *http://algo2.iti.kit.edu/documents/jacm05-revised.pdf*
- *https://github.com/vikasawadhiya/DC3-Algorithm/raw/main/DC3AlgorithmTutorial.pdf*