

Read Mapping

Michael Schatz

Sept 25, 2023

Lecture 9: Applied Comparative Genomics



Assignment 3: Genome Assembly

Due Wednesday Sept 27 by 11:59pm

appliedgenomics2023 / assignments / assignment3 / README.md

Preview Code Blame 115 lines (67 loc) · 7.06 KB

Assignment 3: Mapping and WDLs

Assignment Date: Wednesday, September 20, 2023
Due Date: Wednesday, September 27, 2023 @ 11:59pm

Assignment Overview

In this assignment you will explore WDLs as a workflow language to orchestrate a variety of read mapping tasks. You can execute your WDLs using [miniwdl](#) after running `pip install miniwdl`. You may also find [learn-wdl](#) to be very helpful.

If you are using a Mac, make sure to disable [gRPC FUSE for file sharing](#). There is a currently a very weird bug in Docker desktop 4.12.0 (docker engine 20.10.17) on Apple M1 where if you try to disable "Use gRPC FUSE for file sharing" via the GUI it will turn itself back on when you try to activate it. On M1 you will also need to set the DOCKER_HOST environment variable [Notes](#). Docker is aware of the problem and are working on a fix. In the meantime there is a workaround available: [docker-for-mac#6467](#)

The bioinformatics tools you will need for the assignment (bowtie, samtools, etc) are bundled into a [Docker](#) container so you won't need to install other software packages. This will get you ready to run your code in the cloud for future assignments. Instructions for running the container are available here: <https://github.com/mschatz/wga-essentials>. This is known to work on new macs (M1) and older macs (intel chip). It should also run on Linux and Windows but let us know if you have any issues.

As a reminder, any questions about the assignment should be posted to [Piazza](#).

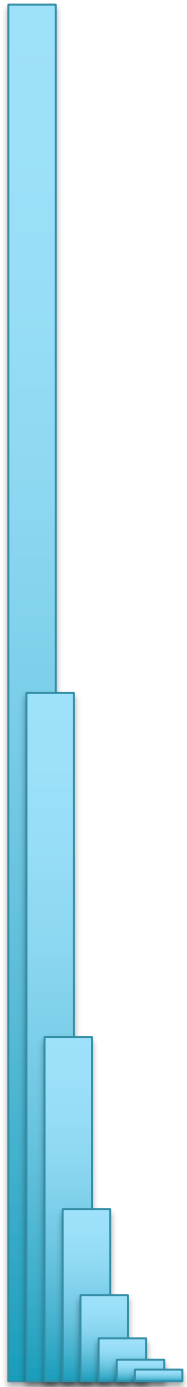
Question 1. de Bruijn Graph construction [10 pts]

- Q1a. Draw (by hand or by code) the de Bruijn graph for the following reads using $k=3$ (assume all reads are from the forward strand, no sequencing errors, complete coverage of the genome). You may find [graphviz](#) to be helpful (see below).

```
ATTCA
ATTGA
CATTG
CTTAT
GATTG
TATTT
TCATT
TCTTA
TGATT
TTATT
```

<https://github.com/schatzlab/appliedgenomics2023/tree/main/assignments/assignment3>

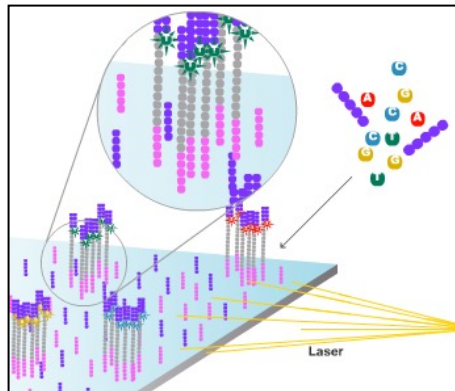
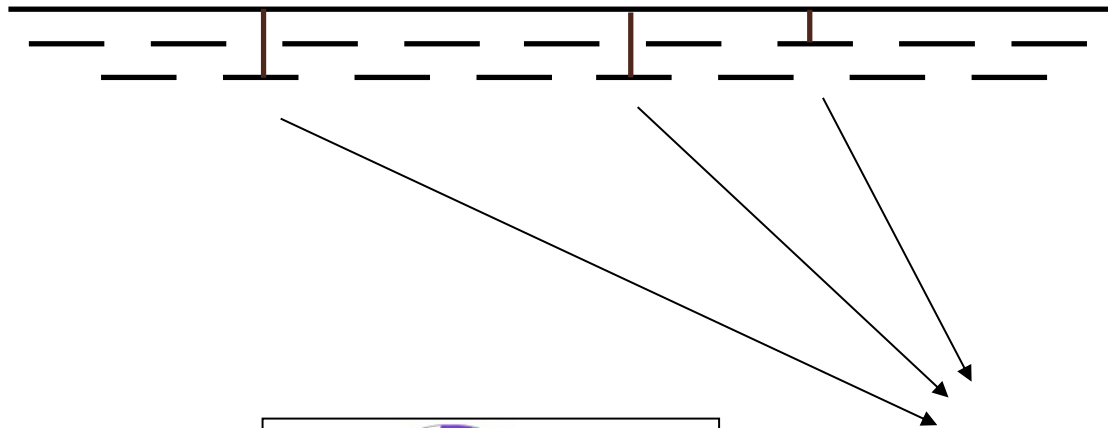
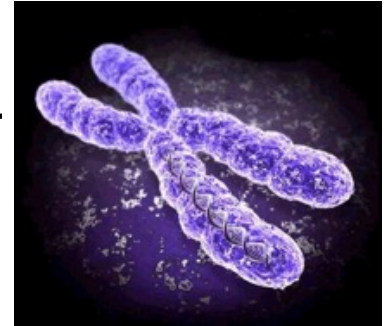
Check Piazza for questions!



Read Mapping

Personal Genomics

How does your genome compare to the reference?



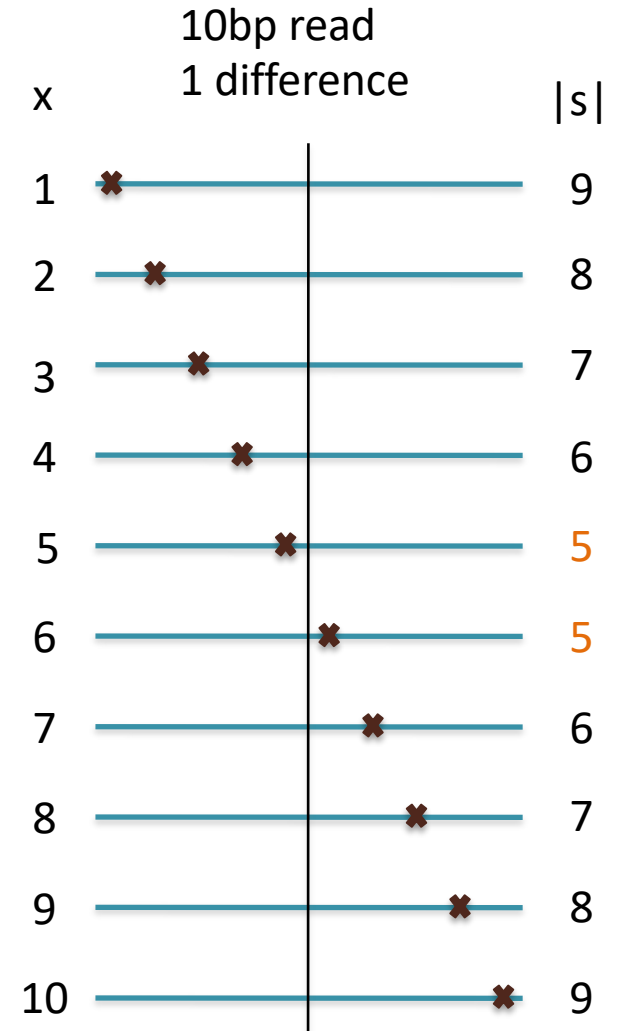
Heart Disease
Cancer
Presidential smile



Seed-and-Extend Alignment

Theorem: An alignment of a sequence of length m with at most k differences **must** contain an exact match at least $s = m / (k + 1)$ bp long
(Baeza-Yates and Perleberg, 1996)

- Proof: Pigeonhole principle
 - 1 pigeon can't fill 2 holes
- Seed-and-extend search
 - Use an index to rapidly find short exact alignments to seed longer in-exact alignments
 - BLAST, MUMmer, Bowtie, BWA, SOAP, ...
 - Specificity of the depends on seed length
 - Guaranteed sensitivity for k differences
 - Also finds some (but not all) lower quality alignments <- heuristic



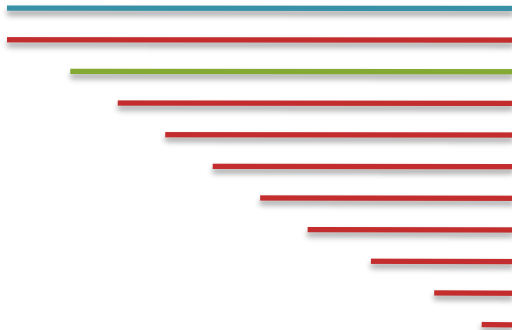
Brute Force Analysis



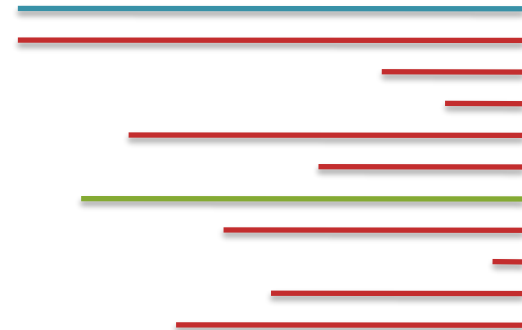
- Brute Force:
 - At every possible offset in the genome:
 - Do all of the characters of the query match?
- Analysis
 - Simple, easy to understand
 - Genome length = n [3B]
 - Query length = m [7]
 - Comparisons: $(n-m+1) * m$ [21B]
- Overall runtime: $O(nm)$
 - [How long would it take if we double the genome size, read length?]
 - [How long would it take if we double both?]

Suffix Arrays: Searching the Phone Book

- What if we need to check many queries?
 - We don't need to check every page of the phone book to find 'Schatz'
 - Sorting alphabetically lets us immediately skip 96% (25/26) of the book *without any loss in accuracy*
- Sorting the genome: Suffix Array (Manber & Myers, 1991)
 - Sort every suffix of the genome



Split into n suffixes



Sort suffixes alphabetically

[Challenge Question: How else could we split the genome?]

Binary Search Analysis

- Binary Search

Initialize search range to entire list

$mid = (hi+lo)/2$; $middle = suffix[mid]$

if query matches middle: done

else if query < middle: pick low range

else if query > middle: pick hi range

Repeat until done or empty range

[WHEN?]

- Analysis

- More complicated method

- How many times do we repeat?

- How many times can it cut the range in half?

- Find smallest x such that: $n/(2^x) \leq 1$; $x = \lg_2(n)$

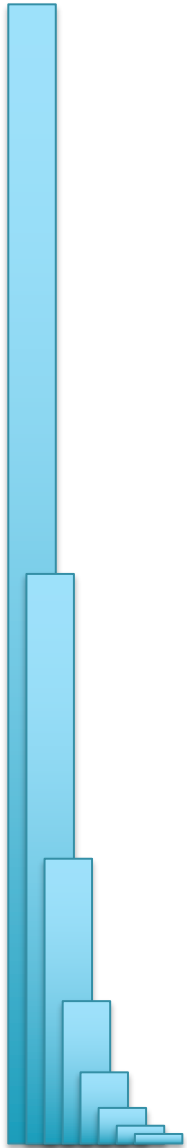
[32]

- Total Runtime: $O(m \lg n)$

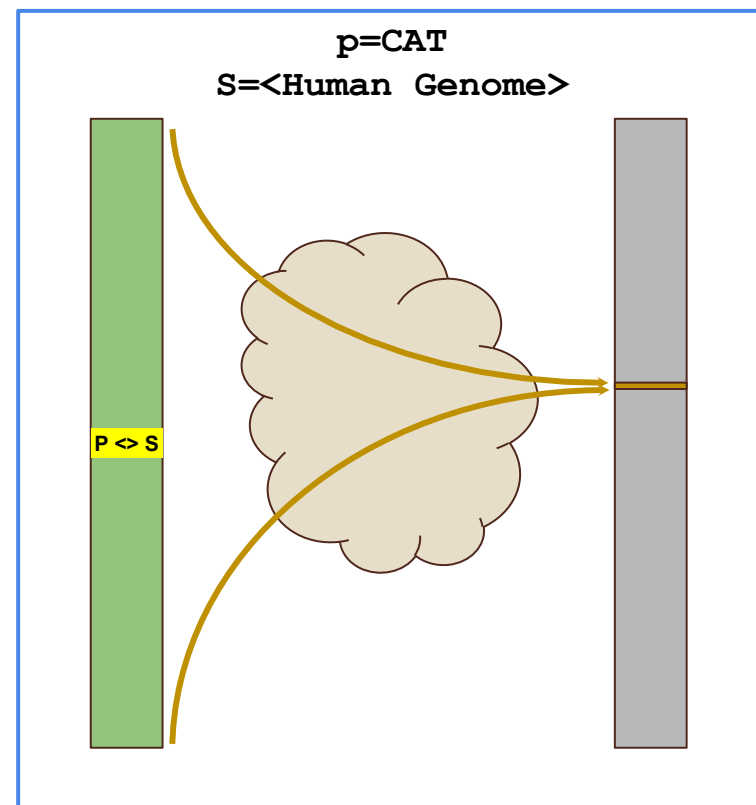
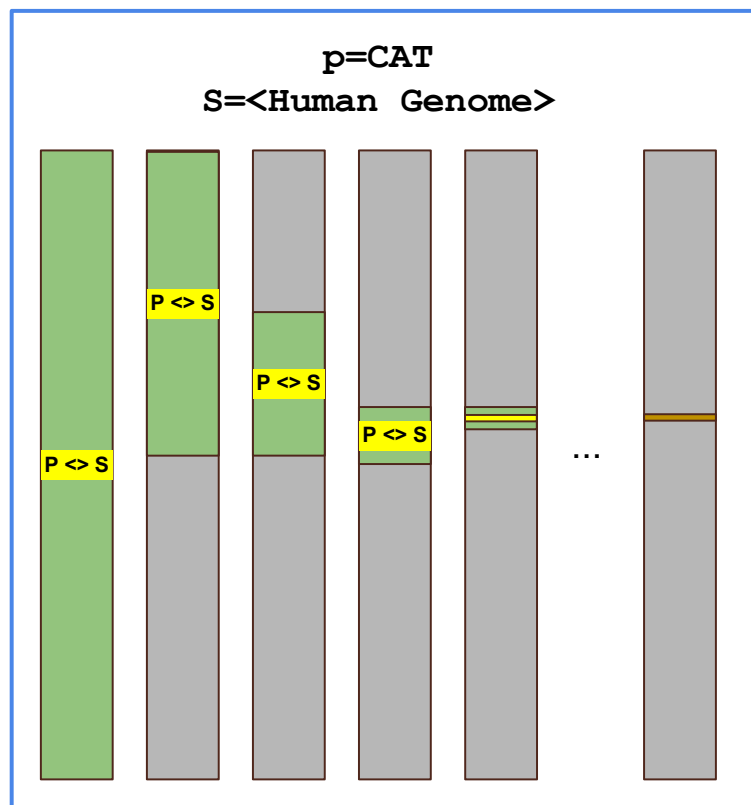
- More complicated, but **much** faster!

- Looking up a query loops 32 times instead of 3B

Can be reduced to $O(m + \lg n)$
using an auxiliary data structure called the LCP array



Sapling: Accelerating Suffix Array Queries with Learned Data Models



What if instead of a slow algorithmic approach to find the correct rows, we could somehow quickly guess/predict the correct rows?

Kirsche, M, Das, A, Schatz, MC (2020) Bioinformatics
doi: <https://doi.org/10.1093/bioinformatics/btaa911>

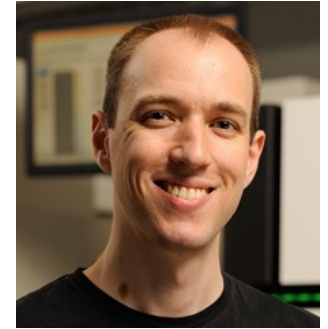


Part 2: Burrows Wheeler Transform

Algorithmic challenge

How can we combine the speed of a suffix array $O(m + \lg(n))$ (or even $O(m)$) with the size of a brute force analysis (n bytes)?

What would such an index look like?



Bowtie: Ultrafast and memory efficient alignment of short DNA sequences to the human genome

Slides Courtesy of Ben Langmead

Algorithm Overview

1. Split read into segments

Read

Read (reverse complement)

CCAGTAGCTCTCAGCCTTATTTTACCCAGGCCTGTA TACAGGCCTGGGTAAAATAAGGCTGAGAGCTACTGG

Policy: extract 16 nt seed every 10 nt

Seeds

+ , 0: CCAGTAGCTCTCAGCC - , 0: TACAGGCCTGGGTAAA

+ , 10: TCAGCCTTATTTTACC - , 10: GGTAAAATAAGGCTGA

+ , 20: TTTACCCAGGCCTGTA - , 20: GGCTGAGAGCTACTGG

2. Lookup each segment and prioritize

Seeds

- + , 0: CCAGTAGCTCTCAGCC
- + , 10: TCAGCCTTATTTTACC
- + , 20: TTTACCCAGGCCTGTA
- , 0: TACAGGCCTGGGTAAA
- , 10: GGTAAATAAGGCTGA
- , 20: GGCTGAGAGCTACTGG

Ungapped alignment with FM Index

aaac

\$acacg\$acacg\$acg\$acac

g\$acac

Seed alignments (as B ranges)

- { [211, 212], [212, 214] }
- { [653, 654], [651, 653] }
- { [684, 685] }
- { }
- { }
- { [624, 625] }

3. Evaluate end-to-end match

Extension candidates

SA:684, chr12:1955
SA:624, chr2:462
SA:211, chr4:762
SA:213, chr12:1935
SA:652, chr12:1945

→

SIMD dynamic programming aligner

→

SAM alignments

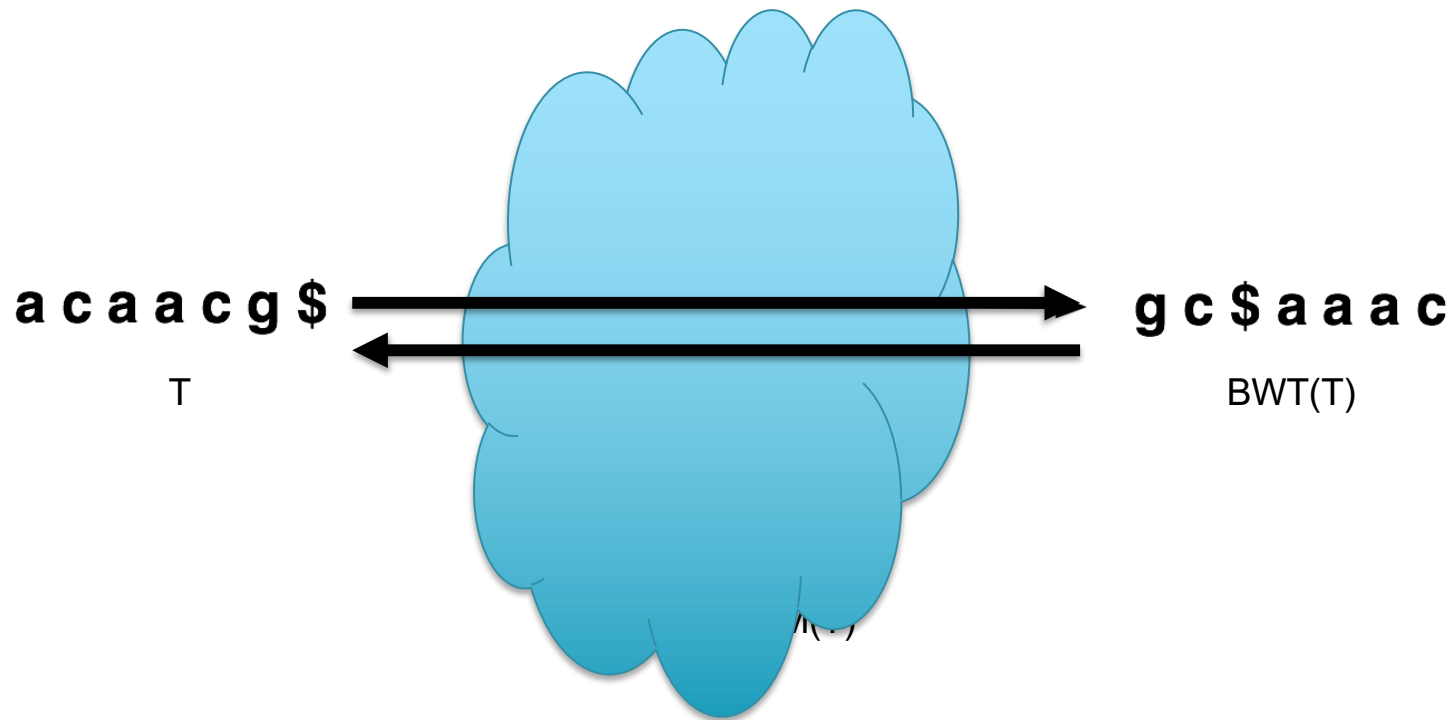
```
r1    0   chr12   1936     0
      36M *      0
      CCAGTAGCTCTCAGCCTTATTTTACCCAGGCCTGTA
      IXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
AS:i:0   XS:i:-2  XN:i:0
XM:i:0   XO:i:0   XG:i:0
NM:i:0   MD:Z:36  YT:Z:UU
YM:i:0
```

(Langmead & S)

(Langmead & Salzberg, 2012)

Burrows-Wheeler Transform

- Reversible permutation of the characters in a text



A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Permutation of the characters in a text

a c a a c g \$ →
T

A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Permutation of the characters in a text

a c a a c g \$ \longrightarrow

T

a c a a c g \$
c a a c g \$ a
a a c g \$ a c
a c g \$ a c a
c g \$ a c a a
g \$ a c a a c
\$ a c a a c g

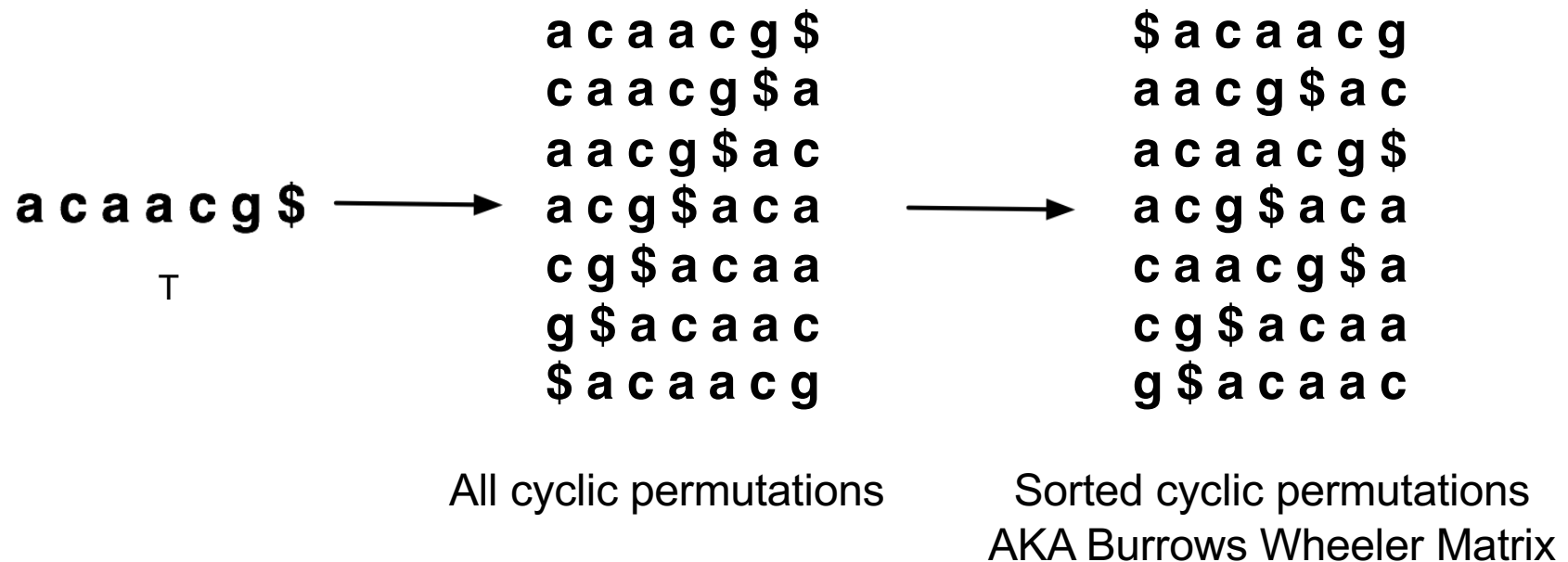
All cyclic permutations

A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Permutation of the characters in a text



A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Permutation of the characters in a text

$a\ c\ a\ a\ c\ g\ \$$ \longrightarrow

$\begin{matrix} \$a\ c\ a\ a\ c\ g \\ a\ a\ c\ g\ \$\ a\ c \\ a\ c\ a\ a\ c\ g\ \$ \\ a\ c\ g\ \$\ a\ c\ a \\ c\ a\ a\ c\ g\ \$\ a \\ c\ g\ \$\ a\ c\ a\ a \\ g\ \$\ a\ c\ a\ a\ c \end{matrix}$

Sorted cyclic permutations
AKA Burrows Wheeler Matrix

A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Permutation of the characters in a text

$a\ c\ a\ a\ c\ g\ \$$ \longrightarrow

T

\$	a	c	a	a	c	g
a	a	c	g	\$	a	c
a	c	a	a	c	g	\$
a	c	g	\$	a	c	a
c	a	a	c	g	\$	a
c	g	\$	a	c	a	a
g	\$	a	c	a	a	c

Sorted cyclic permutations
AKA Burrows Wheeler Matrix

Last Column = Burrows Wheeler Transform

A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Permutation of the characters in a text



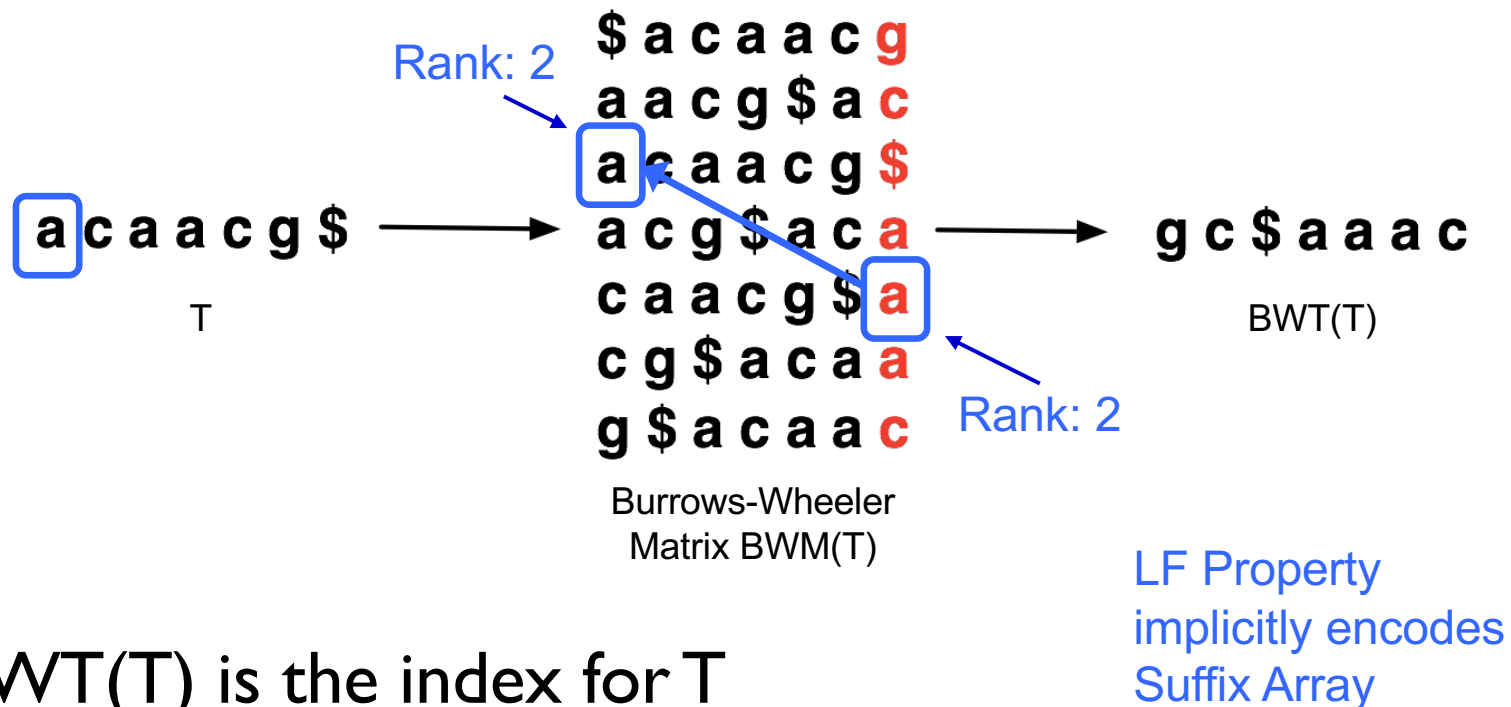
BWT(T) is the index for T

A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

Burrows-Wheeler Transform

- Reversible permutation of the characters in a text



- $BWT(T)$ is the index for T

A block sorting lossless data compression algorithm.

Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation*. Technical Report 124

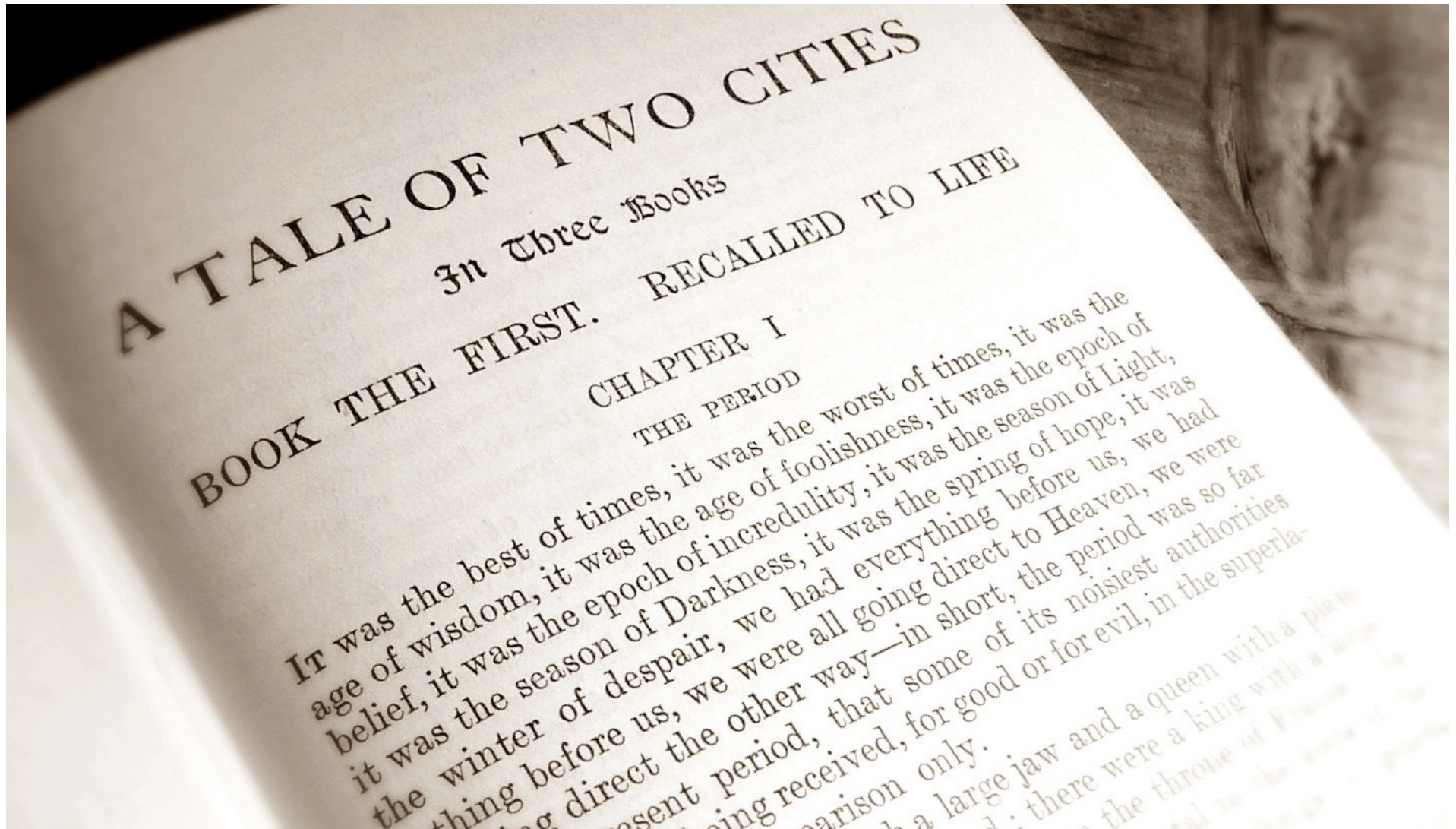
Burrows-Wheeler Transform

- Recreating T from BWT(T)
 - Start in the first row and apply **LF** repeatedly, accumulating predecessors along the way



[Decode this BWT string: ACTGA\$TTA]

Run Length Encoding



Run Length Encoding

ref[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_foolishness,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,_it_wa_s_the_season_of_Darkness,_it_was_the_spring_of_hope,_it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_all_going_direct_to_Heaven,_we_were_all_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_that_some_of_its_noisiest_authorities_insisted_on_its_being_received,_for_good_or_for_evil,_in_the_superlative_degree_of_comparison_only.\$

Run Length Encoding:

- Replace a “run” of a character X with a single X followed by the length of the run
- GAAAAAAAAATTACA => GA8T2ACA (reverse is also easy to implement)
- If your text contains numbers, then you will need to use a (slightly) more sophisticated encoding

Run Length Encoding

ref[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_foolishness,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,_it_wa_s_the_season_of_Darkness,_it_was_the_spring_of_hope,_it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_all_going_direct_to_Heaven,_we_were_all_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_that_some_of_its_noisiest_authorities_insisted_on_its_being_received,_for_good_or_for_evil,_in_the_superlative_degree_of_comparison_only.\$

rle(ref)[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_fo2lishnes2,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,_it_wa_s_the_season_of_Darknes2,_it_was_the_spring_of_hope,_it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_al2_going_direct_to_Heaven,_we_were_al2_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_that_some_of_its_noisiest_authorities_insisted_on_its_being_received,_for_go2d_or_for_evil,_in_the_superlative_degre2_of_comparison_only.\$

Run Length Encoding

ref[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_foolishness,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,_it_wa_s_the_season_of_Darkness,_it_was_the_spring_of_hope,_it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_all_going_direct_to_Heaven,_we_were_all_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_that_some_of_its_noisiest_authorities_insisted_on_its_being_received,_for_good_or_for_evil,_in_the_superlative_degree_of_comparison_only.\$

bwt[614]:

.dlmssfty sesdtrsns_y_\$_yfofeeeetggsfefefggeedrofr,llreef-,fs,,,,,,,,, ,nfrsdnnhereghettedndeteegenstee,sssst,esssnssffteedtttttttttttr,, ,eeefehh__p__fpDwwwwwwwwweehl_ew_____eoo_neeeoaaeoo_____sephhrrhvh hwwegmghhhhhhhkrrwwhhssHrrrvtrribbdbcbvs__thwwpppvmmirdnnib__eoosooo oooooo_____eennnnnnaai__ecc__ttttttttttttttttttts_tsgltsLlvtt__hhoor e_wrraddwlors_____r__lteirillre_ouaanooiioeoosoiikhiiiiioo__iei tsppioi_____ggnodsc_sss_gfhf_fffhwh_nsmo__uee_sioooaeeeeoo_ii cgppeeaoaeooesseuutetataaaaaaaaaai__ei_in__aaie_eereei_hrsssnacciiIi iiiiiisn_____oyoui__a_iids__aiaae_____tlar

Run Length Encoding

ref[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_foolishness,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,_it_wa_s_the_season_of_Darkness,_it_was_the_spring_of_hope,_it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_all_going_direct_to_Heaven,_we_were_all_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_ that_some_of_its_noisiest_authorities_insisted_on_its_being_received ,_for_good_or_for_evil,_in_the_superlative_degree_of_comparison_only.\$

bwt[614]:

.dlmssfty sesdtrsns_y_\$_yfofeeeetggsfefefggeedrofr,llreef-,fs,,,,,,,,, ,nfrsdnnhereghettedndeteegenstee,sssst,esssnssffteedtttttttttttr,, ,eeefehh__p__fpDwwwwwwwwweehl_ew_____eoo_neeeoaaeoo_____sephhrrhvh hwwegmghhhhhhhkrrwwhhssHrrrvtrribbdbcbvs__thwwpppvmmirdnnib__eoooooo oooooo_____eennnnnnaai__ecc__ttttttttttttttttttts_tsgltsLlvtt__hhoor e_wrraddwlors_____r__lteirillre_ouaanooiioeooooiihkiiiiiiio__iei tsppioi_____ggnodsc_sss_gfhf_fffhwh_nsmo__uee_sioooaeeeeoo_ii cgppeeaoaeooeesseuutetaaaaaaaaaaaai__ei_in__aaie_eereei_hrsssnacciiIi iiiiiisn_____ Why does the BWT tend to make runs in english text? _____tlar

Run Length Encoding

bwt[614]:

```
.dlmssftysesdtrsns_y__$_yfofeeeeetggsfefefggeedrofr,llreef-,fs,,,,,,,,,
,,nfrsdnnhereghettedndeteegenstee,sssst,esssnssffteedttttttttttr,,
,,eeefehh__p__fpDwwwwwwwwweehl_ew_____eoo_neeeoaaeoo_____sephhrrhvh
hwwegmghhhhhhhkrrwhhssHrrrvtrribbdbcbvs__thwwpppvmmirdnnib__eoooooo
oooooo_____eennnnnnaai__ecc__ttttttttttttttttts_tsgltsLlvt__hhoor
e_wrraddwlors_____r__lteirillre_ouaanooioeooooiihkiiiiio__iei
tsppioi_____ggnodsc_sss_gfhf_fffhwh_nsmo__uee_sioooaeeeeoo__ii
cgppeaaoaeooeesseuutetaaaaaaaaaaai__ei_in__aaie_eere_i_hrssnacciiIi
iiiiisn_____oyoui__a_iids__aiaae_____tlar
```

rle(bwt)[464]:

```
.dlms2ftysesdtrsns_y_2$_yfofe4tg2sfefefg2e2drofr,l2re2f-,fs,9nfrsdn2
hereghet2edndete2ge2nste2,s5t,es3ns2f2te2dt10r,4e3feh2_2p_2fpDw11e2h
l_ew_5eo2_ne3oa2eo2_4seph2r2hvh2w2egmgh7kr2w2h2s2Hr3vtr2ib2dbcbvs_2t
hw2p3vm2irdn2ib_2eo12_4e2n6a2i_3ec2_2t18s_tsgltsLlvt2_3h2o2re_wr2ad2
wlors_9r_2lteiril2re_oua2no2i2oeo4i3hki6o_2ieitsp2ioi_12g2nodsc_s3_g
fhf_f3hwh_nsmo_2ue2_sio3ae4o2_i2cgp2e2aoaeo2e2s2eu2teta11i_2ei_in_2a
2ie_e3rei_hrs3nac2i2Ii7sn_15oyoui_2a_i3ds_2ai2ae2_21tlar
```

Run Length Encoding

bwt[614]:

```
.dlmssftysesdtrsns_y__$yfofeeeeetggsfefefggedrofr,llreef-,fs,,,,,,,,,
,,nfrsdnnherghettedndeteegenstee,sssst,esssnssffteedtttttttttr,,
,,eeefehh__p__fpDwwwwwwwwweehl_ew_____eoo_neeeoaaeoo_____sephhrrhvh
hwwegmghhhhhhhkrrwwhhssHrrrvtrribbdbcbvs__thwwpppvmmirdnnib__eooooo
ooooo_____eennnnnnaai__ecc__ttttttttttttttts_tsgltsLlvt__hhoor
e_wrraddwlors_____r__lteirillre_ouaanooioeooooiihkiiiiio__iei
tsppioi_____ggnodsc_sss_gfhf_fffhwh_nsmo__uee_sioooaeeeeoo__ii
cgppeaaoaeooesseuutetaaaaaaaaai__ei_in__aaie_eere_i_hrssnacciiIi
iiiiisn_____oyoui__a_iids__aiaae_____tlar
```

rle(bwt)[464]:

```
.dlms2ftysesdtrsns_y_2$yfofe4tg2sfefefg2e2drofr,l2re2f-,fs,9nfrsdn2
herghet2edndete2ge2nste2,s5t,es3ns2f2te2dt10r,4e3feh2_2p_2fpDw11e2h
l_ew_5eo2_ne3oa2eo2_4seph2r2hvh2w2egmgh7kr2w2h2s2Hr3vtr2ib2dbcbvs_2t
hw2p3vm2irdn2ib_2eo12_4e2n6a2i_3ec2_2t18s_tsgltsLlvt2_3h2o2re_wr2ad2
wlors_9r_2lteiril2re_oua2no2i2oeo4i3hki6o_2ieitsp2ioi_12g2nodsc_s3_g
fhf_f3hwh_nsmo_2ue2_sio3ae4o2_i2cgp2e2aoaeo2e2s2eu2tetall1i_2ei_in_2a
2ie_e3rei_hrs3nac2i2Ii7sn_15oyoui_2a_i3ds_2ai2ae2_21tlar
```

Run Length Encoding

ref[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_foolishness,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,it_wa_s_the_season_of_Darkness,it_was_the_spring_of_hope,it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_all_going_direct_to_Heaven,_we_were_all_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_that_some_of_its_noisiest_authorities_insisted_on_its_being_received,_for_good_or_for_evil,in_the_superlative_degree_of_comparison_only.\$

rle(bwt)[464]:

.dlms2ftysesdtrsns_y_2\$_yfofe4tg2sfefefg2e2drofr,l2re2f-,fs,9nfrsdn2 hereghet2edndete2ge2nste2,s5t,es3ns2f2te2dt10r,4e3feh2_2p_2fpDw11e2h l_ew_5eo2_ne3oa2eo2_4seph2r2hvh2w2egmgh7kr2w2h2s2Hr3vtr2ib2dbcbvs_2t hw2p3vm2irdn2ib_2eo12_4e2n6a2i_3ec2_2t18s_tsgltsLlvt2_3h2o2re_wr2ad2 wlors_9r_2lteiril2re_oua2no2i2oeo4i3hki6o_2ieitasp2ioi_12g2nodsc_s3_g fhf_f3hwh_nsmo_2ue2_sio3ae4o2_i2cgp2e2aoaeo2e2s2eu2tetall1i_2ei_in_2a 2ie_e3rei_hrs3nac2i2Ii7sn_15oyoui_2a_i3ds_2ai2ae2_21tlar

Run Length Encoding

ref[614]:

It_was_the_best_of_times,_it_was_the_worst_of_times,_it_was_the_age_of_wisdom,_it_was_the_age_of_foolishness,_it_was_the_epoch_of_belief,_it_was_the_epoch_of_incredulity,_it_was_the_season_of_Light,_it_wa_s_the_season_of_Darkness,_it_was_the_spring_of_hope,_it_was_the_wint_er_of_despair,_we_had_everything_before_us,_we_had_nothing_before_us,_we_were_all_going_direct_to_Heaven,_we_were_all_going_direct_the_o ther_way_-_in_short,_the_period_was_so_far_like_the_present_period,_that_some_of_its_noisiest_authorities_insisted_on_its_being_received,_for_good_or_for_evil,_in_the_superlative_degree_of_comparison_only.\$

rle(bwt)[464]:

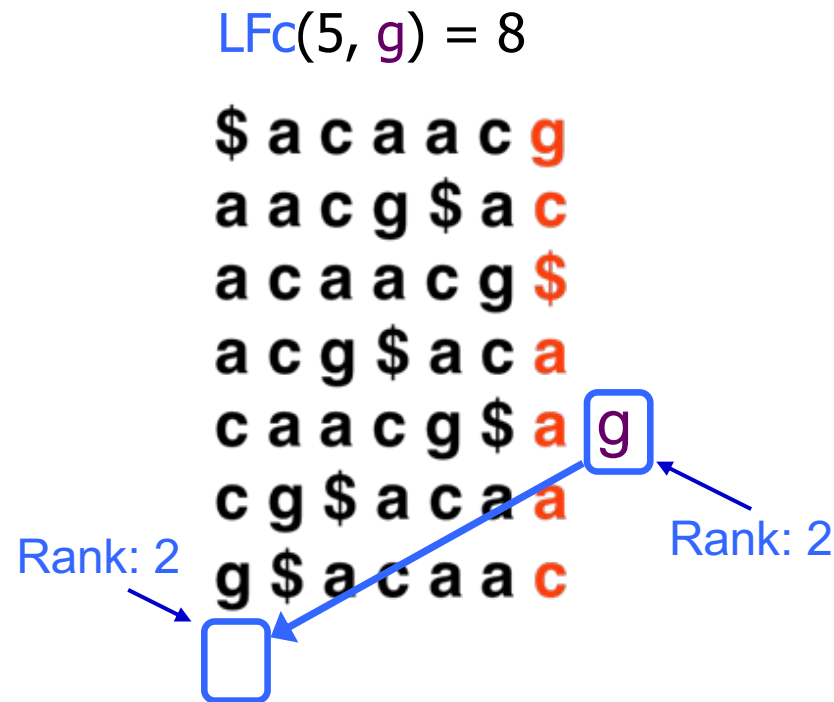
.dlms2ftysestrsns_y_2\$_yfofe4tg2sfefefg2e2drofr,l2re2f-,fs,9nfrsdn2 hereghet2edndete2ge2nste2,s5t,es3ns2f2te2dt10r,4e3feh2_2p_2fpDw11e2h l_ew_5eo2_ne3oa2eo2_4seph2r2hvh2w2egmgh7kr2w2h2s2Hr3vtr2ib2dbcbvs_2t hw2p3vm2irdn2ib_2eo12_4e2n6a2i_3ec2_2t18s_tsgltsLlvt2_3h2o2re_wr2ad2 wlors_9r_2lteiril2re_oua2no2i2oeo4i3hki6o_2ieitasp2ioi_12g2nodsc_s3_g fhf_f3hwh_nsmo_2ue2_sio3ae4o2_i2cgp2e2aoaeo2e2s2eu2tet11i_2ei_in_2a 2ie_e3rei

Saved 614-464 = 150 bytes (24%) with zero loss of information!

Common to save 50% to 90% on real world files with bzip2

BWT Exact Matching

- **LFc**(r, c) does the same thing as **LF**(r) but it ignores r's actual final character and “pretends” it's c:



BWT Exact Matching

- Start with a range, (**top**, **bot**) encompassing all rows and repeatedly apply **LFc**:

$$\text{top} = \text{LFc}(\text{top}, \text{qc}); \text{bot} = \text{LFc}(\text{bot}, \text{qc})$$

qc = the next character to the left in the query



Ferragina P, Manzini G: Opportunistic data structures with applications. *FOCS. IEEE Computer Society; 2000.*

[Search for TTA this BWT string: ACTGA\$TTA]

Exact Matching Review & Overview

Where is GATTACA in the human genome?

Brute Force
(3 GB)

BANANA
BAN
ANA
NAN
ANA

$O(m * n)$

Slow & Easy

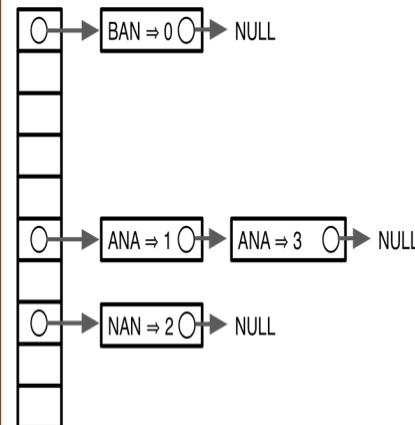
Suffix Array
(>15 GB)

6	\$
5	A\$
3	ANA\$
1	ANANA\$
0	BANANA\$
4	NA\$
2	NANA\$

$O(m + \lg n)$

Full-text index

Hash Table
(>15 GB)



$O(1)$

Fixed-length lookup

BWT
(3 GB)

BANANA\$
=>
\$BANANA
A\$BANAN
ANA\$BAN
ANANA\$B
BANANA\$
NA\$BAN
NANA\$BA
=>
ANNB\$AA

$O(m)$

Full-text and concise

*** These are general techniques applicable to any text search problem ***