,

Succinct De Bruijn graphs

Implementation report

Brilli Matteo

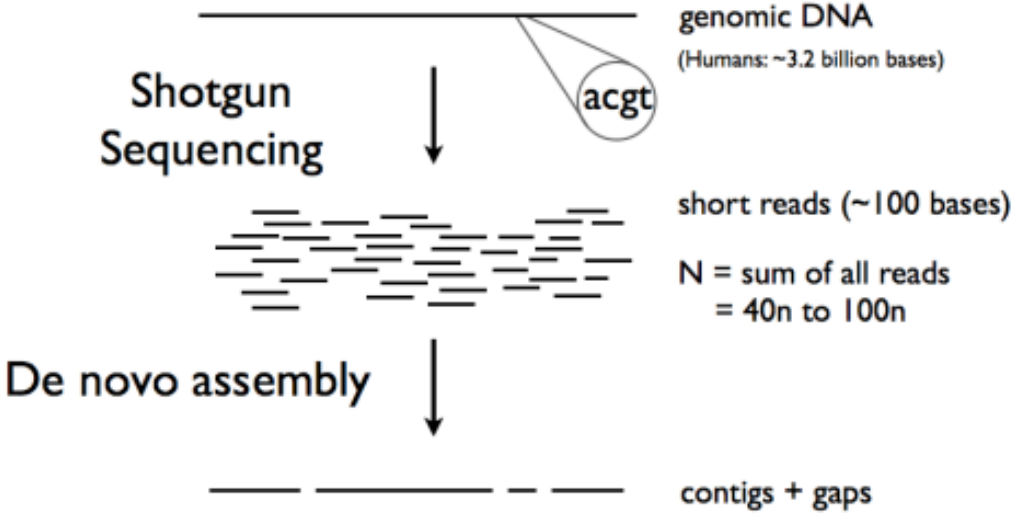
Università Ca' Foscari Venezia

September 9, 2024

"Succint de Bruijn Graphs" by Bowe et al. (2012)

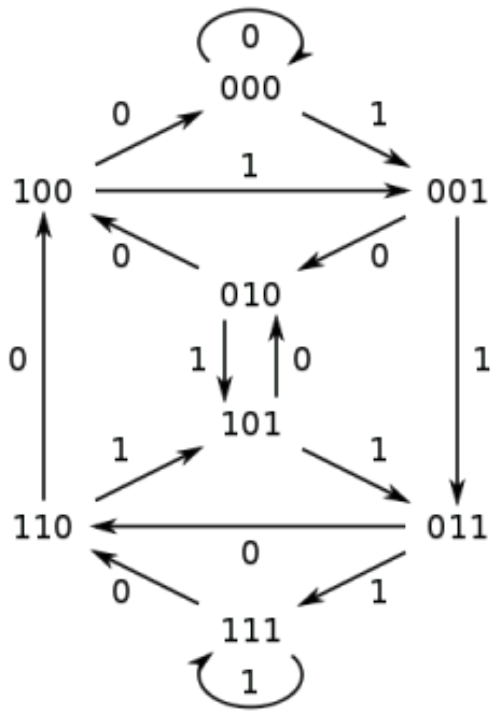
New succinct de Bruijn graph representation for k-mers in a DNA sequence using $(2 + \log \sigma)m + o(m)$ bits (so $4m + o(m)$ for DNA).

- Summary of the main ideas of the paper
- C++ implementation of the BOSS data structure
- Analysis of memory usage and construction time



- $|\Sigma|^k$ distinct nodes, corresponding to all possible sequences.
- Directed edges between overlapping nodes.

- De novo assembly of reads.

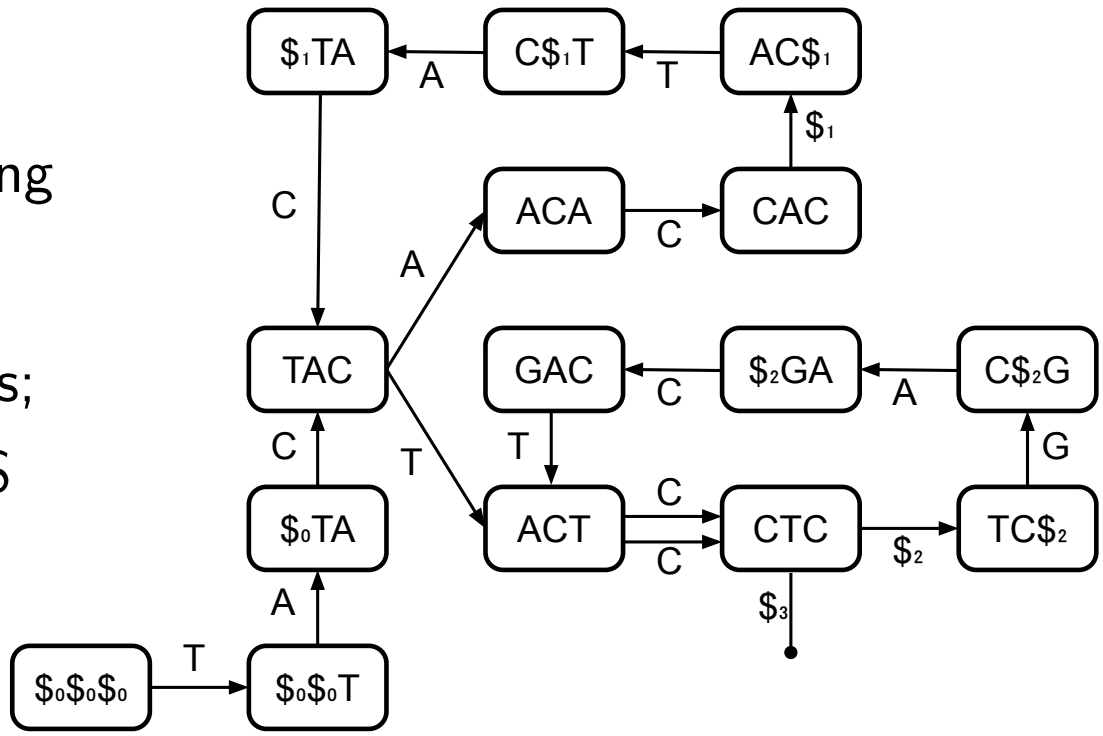


Alex Bowe. *Succinct de Bruijn Graphs*. July 2013

Objective

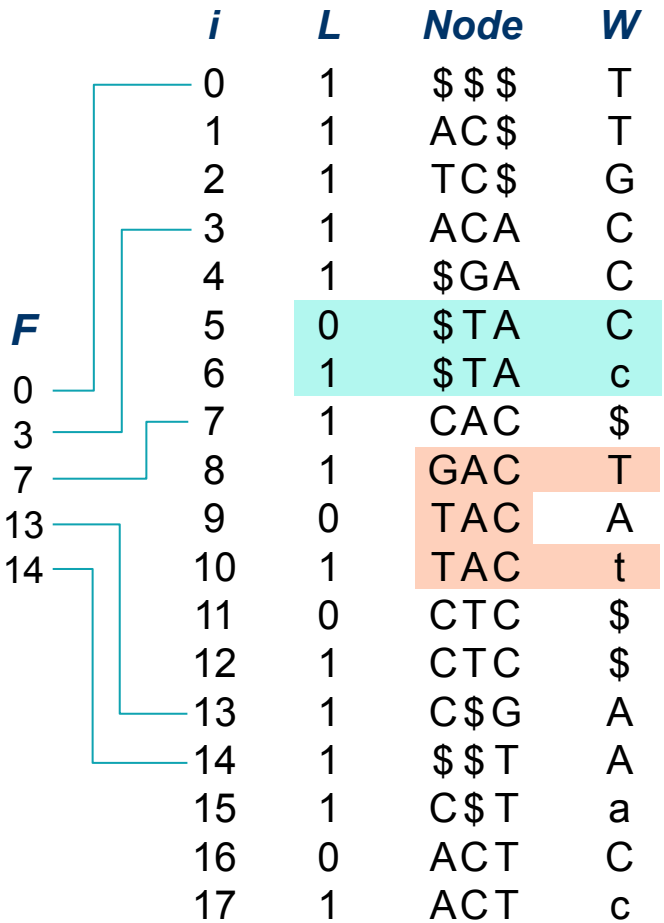
Construction of the k-dimensional dBg of a string T of lenght N:

- At most $N - k + 1$ nodes;
- k terminator characters \$ at the head;
- Different reads are followed by a \$ and concatenated.



$T =$"$\$_0\$_0\$_0$TACAC$\$_1$TACTC$\$_2$GACTC$\$_3$"

Theoretical construction



	i	L	Node	W
	0	1	$$$	T
	1	1	AC$	T
	2	1	TC$	G
	3	1	ACA	C
	4	1	$GA	C
	5	0	$TA	C
	6	1	$TA	c
	7	1	CAC	$
	8	1	GAC	T
	9	0	TAC	A
	10	1	TAC	t
	11	0	CTC	$
	12	1	CTC	$
	13	1	C$G	A
	14	1	$$T	A
	15	1	C$T	a
	16	0	ACT	C
	17	1	ACT	c

F
0
3
7
13
14

First step: taking every <node, edge> pair and sorting them in colex order.

Then, the proposed representation is composed of the following three elements:

- a string W of lenght m over alphabet Σ;
- a bitvector L of lenght m;
- an array F of length $|\Sigma|$.

For a total of $m + m \log(2\sigma + 1) + o(m)$ bits.

- First step: creation of a CSA using the Succinct Data Structure Library (SDSL) $\Longrightarrow$ suffix sorting, label extraction, W, F

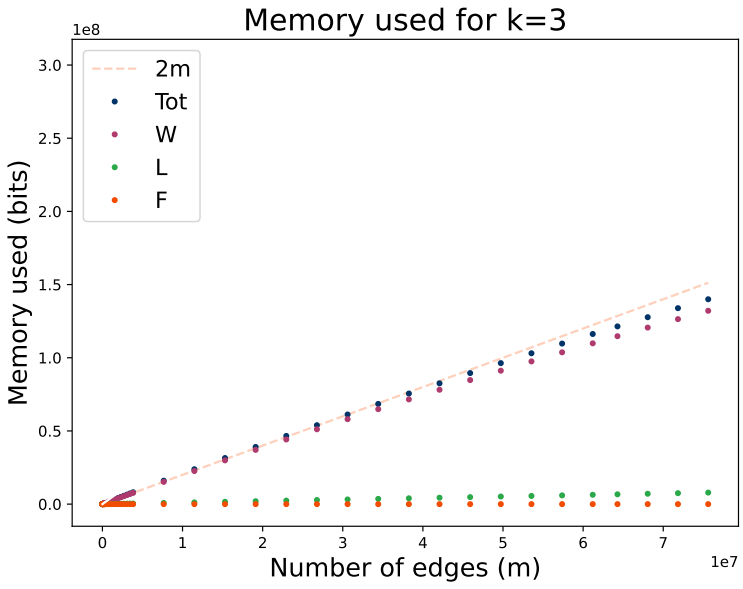
i	SA	ISA	PSI	LF	BWT	T[SA[i]...SA[i]-1]
0	20	14	14	1	$	$CTCAG$CTCAT$CACAT$$$
1	19	19	0	2	$	$$CTCAG$CTCAT$CACAT$$
2	18	11	1	3	$	$$$CTCAG$CTCAT$CACAT$
3	17	7	2	17	T	$$$$CTCAG$CTCAT$CACAT
4	11	16	10	18	T	$CACAT$$$$CTCAG$CTCAT
5	5	5	15	16	G	$CTCAT$CACAT$$$$CTCAG
6	13	15	12	10	C	ACAT$$$$CTCAG$CTCAT$C
⋮	⋮	⋮	⋮	⋮	⋮	⋮

C 0 1 6 10 16 17 21

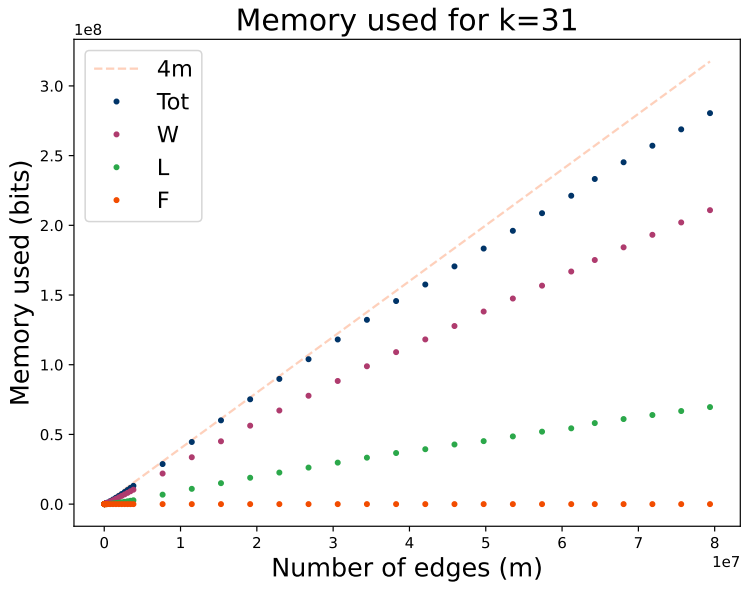
- Second step: for loop $\Longrightarrow$ label extraction, creation of L, flags for the characters of W

- Third step: compression of data structures
  - W $\Longrightarrow$ Wavelet Tree which occupies $mH_0(W) + 2\sigma \log m$.
  - L $\Longrightarrow$ RRR bitvector which occupies $mH_0(L) + o(m)$ bits.
  - F $\Longrightarrow$ compressed with the function $sdsl :: util :: bit\_compress(F)$

Operation	Description	Time
outdegree(v)	Returns number of outgoing edges from node V.	$\mathcal{O}(1)$
outgoing(v,c)	From node V, follows the edge labeled by symbol c.	$\mathcal{O}(1)$
label(v)	Returns the label of node v.	$\mathcal{O}(k)$
index(s)	Returns the index of k-mer s if present.	$\mathcal{O}(k)$
indegree(v)	Returns the number of incoming edges to node v	$\mathcal{O}(1)$
incoming(v,c)	Returns predecessor node starting with c that has an edge to node v	$\mathcal{O}(k \log \sigma)$

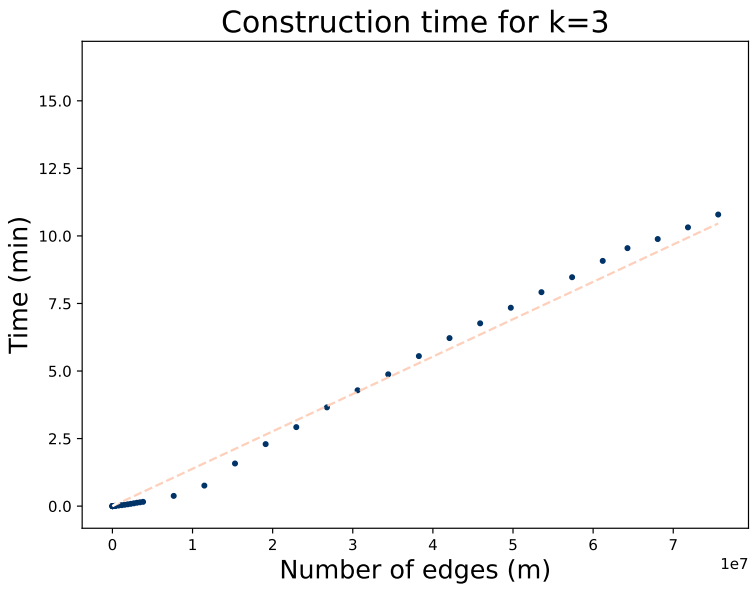


Number of bits for $k = 3$

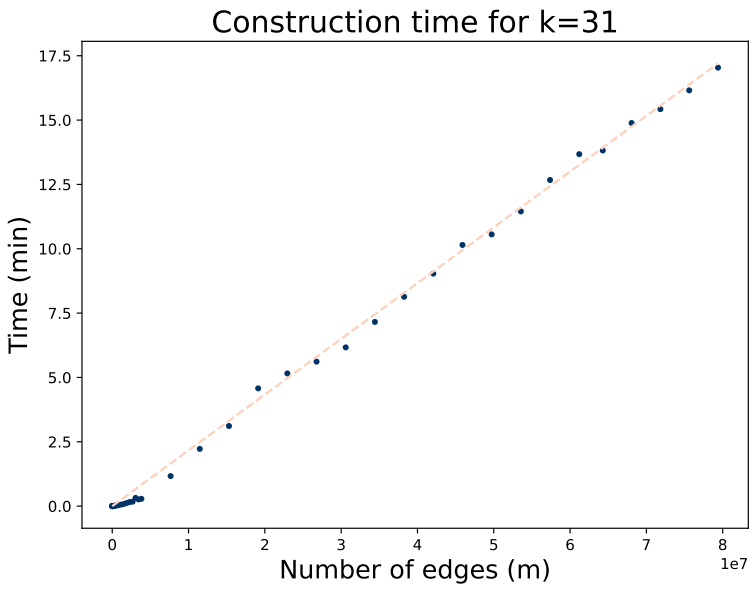


Number of bits for $k = 31$

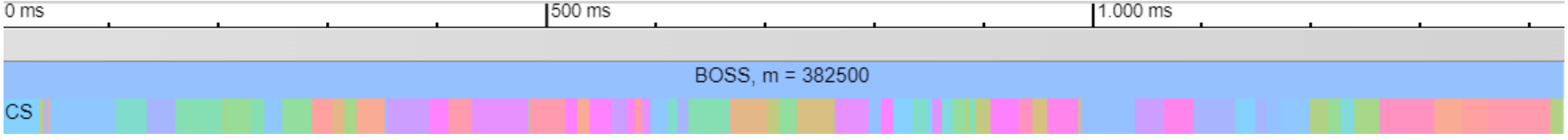
- The number of bits used depends on $H_0(L)$ and on $H_0(W)$ and thus it changes depending on the value of k and the input string.



Construction time for $k = 3$



Construction time for $k = 31$



Breakdown of timing for $k = 3$, $m = 382500$

Results

- We have described a novel approach to representing de Bruijn graphs efficiently, while alsosupporting a full suite of navigation operations quickly.
- The total space is a theoretical $4m$ bits for DNA. Using specially modified indexes we can further lower this value.

Further work

- The data structure can be constructed in $\mathcal{O}(Nk)$ time.
- We can think of reducing the construction time by parallelizing the for loop, which ends up being the most time consuming part of the algorithm.

[1] Alex Bowe. *Succinct de Bruijn Graphs*. July 2013. URL: https://www.alexbowe.com/succinct-debruijn-graphs/ (visited on 09/01/2024).

[2] Alex Bowe et al. "Succinct de Bruijn Graphs". In: *Algorithms in Bioinformatics, Volume 7534 of Lecture Notes in Computer Science* (Sept. 2012), pp. 225–235. DOI: 10.1007/978-3-642-33122-0_18.

[3] Rayan Chikhi. "A Tale of Optimizing the Space Taken by de Bruijn Graphs". In: *Connecting with Computability*. Cham: Springer International Publishing, 2021, pp. 120–134.

[4] Simon Gog et al. "From Theory to Practice: Plug and Play with Succinct Data Structures". In: *13th International Symposium on Experimental Algorithms, (SEA 2014)*. 2014, pp. 326–337.