The FMD and Graph FM Indices

UNIVERSITY OF MARYLAND

The FMD Index

Motivation : The FM-index naturally searches in one direction (from the end of the string to the front)

To find MEMs and SMEMs, it will be useful to extend matches in both directions.

Why not two indices? This could be accomplished with 2 FM-indices, but having the search work within a single index will be more efficient.

First formal description of FMD index

BIOINFORMATICS ORIGINAL PAPER

Vol. 28 no. 14 2012, pages 1838–1844
doi:10.1093/bioinformatics/bts280

Sequence analysis

Advance Access publication May 7, 2012

Exploring single-sample SNP and INDEL calling with whole-genome *de novo* assembly

Heng Li

Medical Population Genetics Program, Broad Institute, 7 Cambridge Center, MA 02142, USA

Associate Editor: Michael Brudno

The FMD Index

Some notation. For pattern P and text T

$$I^l(P) = \min\{k : P \text{ is the prefix of } T_{S(k)}\}$$

$$I^u(P) = \max\{k : P \text{ is the prefix of } T_{S(k)}\}$$

Then $[I^l(P), I^u(P)]$ is the suffix array interval for P

The length of this is given by $I^s(P) = I^u(P) - I^l(P) + 1$

Some notation. For pattern P and text T

The FMD Index

Let $R_0, R_1, \ldots, R_{n-1}$ denote a series of DNA/RNA texts

Define a new text $T=R_0\overline{R}_0R_1\overline{R}_1 \ldots R_{n-1}\overline{R}_{n-1}$

Where $\overline{R}$ is the reverse complement of R

Consider *bi-intervals* of the index of the form $[I^l(P), I^l(P), I^s(P)]$

Also, recall that we can extend a “normal” interval as

$$I^l(aP) = C(a) + O(a, I^l(P) - 1)$$

$$I^u(aP) = C(a) + O(a, I^u(P)) - 1$$

The FMD Index

Assume we have the bi-interval of P, $[I^l(P), I^l(\overline{P}), I^s(P)]$

How do we compute the bi-interval of aP ?

We know that $[I^l(\overline{aP}), I^u(\overline{aP})]$ is a subinterval of $[I^l(\overline{P}), I^u(\overline{P})]$, why?

Because P is a prefix of $\overline{aP} = \overline{P} \circ \overline{a}$

Further, because of the symmetry of T, $I^s(cP) = I^s(\overline{cP}), \forall c$

The FMD Index

Example:
W = AACG
a = G
aW = GAACG
$\overline{\text{Wa}}$ = CGTTC

Consider symmetry of T:

```
#AACG   = #CGTT
#AAACG  = #CGTTT
#CAACG  = #CGTTG
#GAACG  = #CGTTC
#TAACG  = #CGTTA
```

So, given $\overline{\text{W}}$, to extend to $\overline{\text{Wa}}$, we can simply *count!*

Algorithm 2: Backward extension

Input: Bi-interval $[k,l,s]$ of string W and a symbol a
Output: Bi-interval of string aW

Function BACKWARDEXT($[k,l,s],a$) **begin**
  for $b \leftarrow 0$ **to** 5 **do**
    $k_b \leftarrow C(b)+O(b,k-1)$
    $s_b \leftarrow O(b,k+s-1)-O(b,k-1)$
  $l_0 \leftarrow l$;
  $l_4 \leftarrow l_0+s_0$;
  for $b \leftarrow 3$ **to** 1 **do**
    $l_b \leftarrow l_{b+1}+s_{b+1}$
  $l_5 \leftarrow l_1+s_1$;
  return $[k_a,l_a,s_a]$

This is the part that requires some thought

```
$,A,C,G,T,N
0,1,2,3,4,5
```

The FMD Index

Forward extension is simply backward extension in the reverse complement!

Algorithm 3: Forward extension

Input: Bi-interval $[k, l, s]$ of string W and a symbol a
Output: Bi-interval of string Wa

Function FORWARDEXT($[k, l, s], a$) **begin**
    $[l', k', s'] \leftarrow$BACKWARDEXT($[l, k, s], \overline{a}$);
    return $[k', l', s']$

Finding SMEMs with the FMD Index

Algorithm 5: Finding SMEMs

Input: String P and start position i_0; $P[-1]=0$
Output: Set of bi-intervals of SMEMs overlapping i_0

Function SUPERMEM1(P, i_0) **begin**
    Initialize Curr, Prev and Match as empty arrays;
    $[k,l,s] \leftarrow [C(P[i_0]), C(\overline{P[i_0]}), C(P[i_0]+1)-C(P[i_0])]$;
    for $i \leftarrow i_0+1$ **to** $|P|$ **do**
        if $i=|P|$ **then**
            Append $[k,l,s]$ to Curr
        else
            $[k',l',s'] \leftarrow$FORWARDEXT($[k,l,s], P[i]$);
            if $s' \neq s$ **then**
                Append $[k,l,s]$ to Curr
            if $s'=0$ **then**
                break;
            $[k,l,s] \leftarrow [k',l',s']$
    Swap array Curr and Prev;
    $i' \leftarrow |P|$;
    for $i \leftarrow i_0-1$ **to** -1 **do**
        Reset Curr to empty;
        $s'' \leftarrow -1$;
        for $[k,l,s]$ **in** Prev **do**
            $[k',l',s'] \leftarrow$BACKWARDEXT($[k,l,s], P[i]$);
            if $s'=0$ **or** $i=-1$ **then**
                if Curr is empty **and** $i+1<i'+1$ **then**
                    $i' \leftarrow i$;
                    Append $[k,l,s]$ to Match
            if $s' \neq 0$ **and** $s' \neq s''$ **then**
                $s'' \leftarrow s'$;
                Append $[k,l,s]$ to Curr
        if Curr is empty **then**
            break
        Swap Curr and Prev;
    return Match

Extend “forward”

i.e. find left-maximal matches (intervals)

Extend “backward”

i.e. find right-maximal for left-maximal intervals

The Graph FM-Index & HISAT2

Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype

Daehwan Kim ✉, Joseph M. Paggi, Chanhee Park, Christopher Bennett & Steven L. Salzberg

Nature Biotechnology **37**, 907–915 (2019) | Download Citation ↓

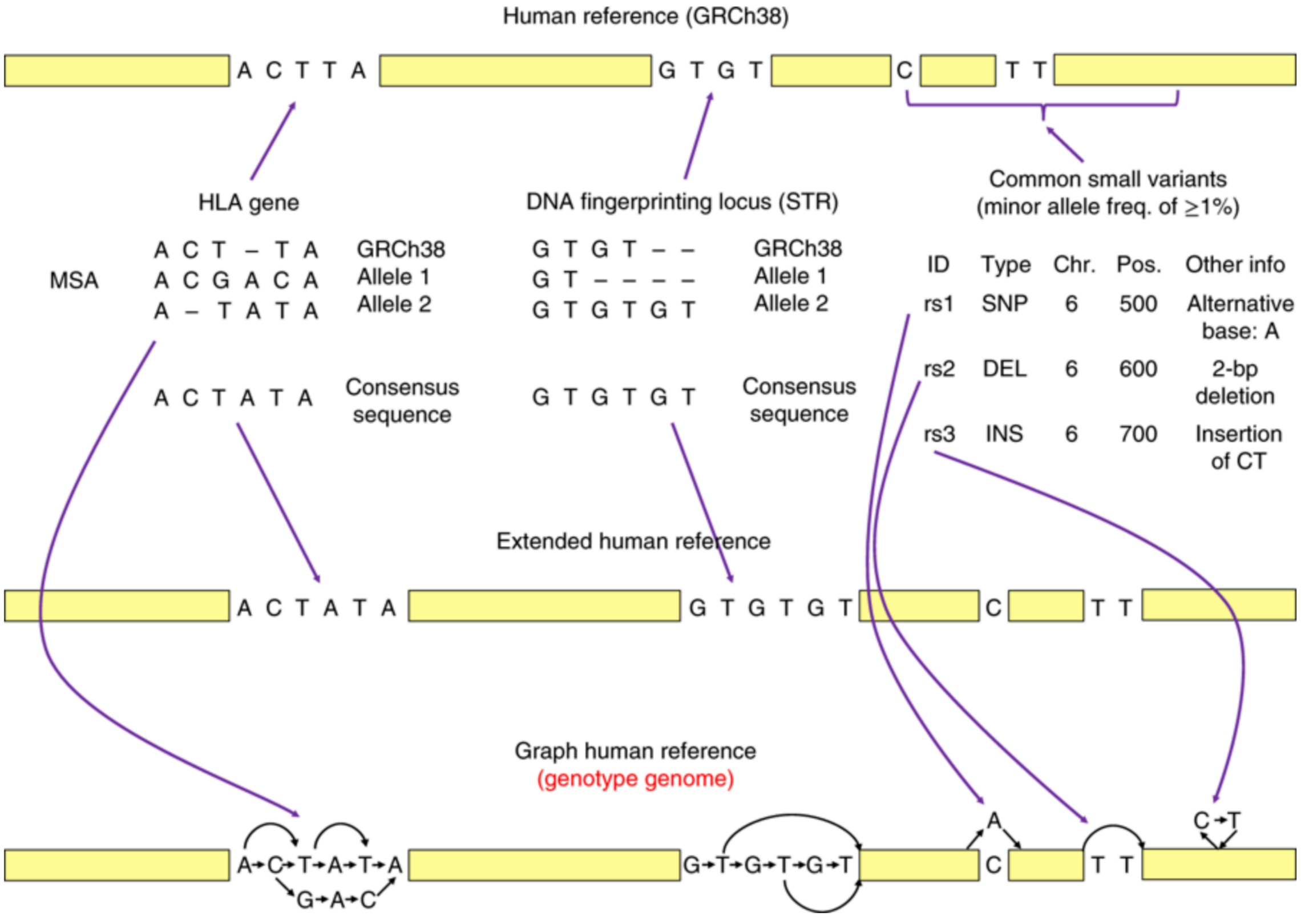
The Graph FM-Index

Idea / motivation : **No sample is the reference**

We have spent a lot of effort characterizing major human variants, yet most aligners simply map against a single human reference genome that doesn't even have the most likely variant at each locus.

HISAT2 is one of a new breed of "graph" aligners, that views the genome as a graph rather than a simple string. This framework allows encoding variants as alternative "paths" through the genome.

The Graph FM-Index



Graph FM-Index

Siren, J., Valimaki, N. & Makinen, V. Indexing graphs for path queries with applications in genome research. *IEEE-ACM Trans. Comput. Biol. Bioinform.* **11**, 375–388 (2014).

Construction of graph FM index relies on creation of prefix-range-sorted automata

- Key property needed for backward search:

  - For list (u,v) of outgoing edges, sorted by pairs (p(u), p(v)), l(u)p(v) must be sorted by sequences

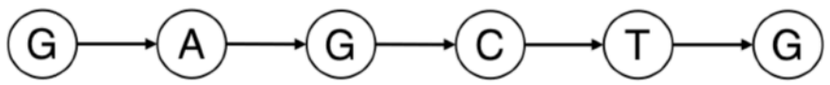
  - For any c, all outgoing edges from nodes labeled with c are lexicographically adjacent and are sorted by the prefix p(v) of the destination node

  - All occurrences of c in BWT encode an incoming edge from a node with label c, and thus are sorted by prefix p(v) of the destination

  - Hence, incoming edge labeled by nth occurrence of c is the same as the outgoing edge for rank C[c]+j

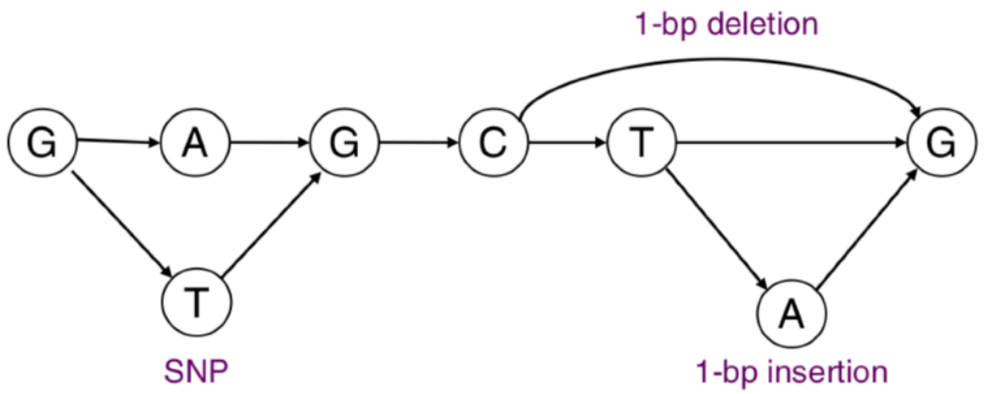
The Graph FM-Index

1. Reference sequence (6 bp long)



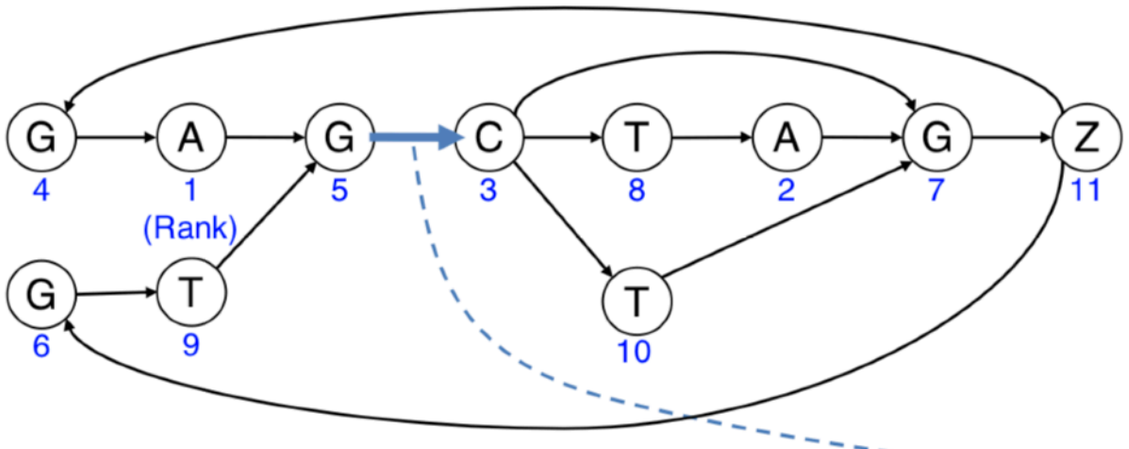
single-nucleotide variant (A/T), a 1-bp deletion (T) and a 1-bp insertion (A)

2. Graphical representation (original graph)



Prefix doubling and pruning

3. Prefix-sorted graph



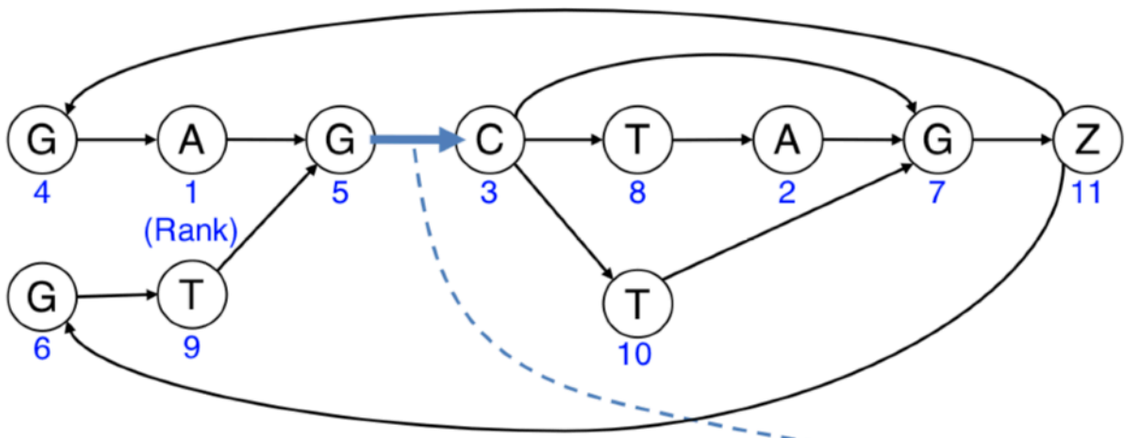
4. Tabular representation of the prefix-sorted graph

Outgoing edge(s)			Incoming edge(s)	
Node rank	First		Last	Node rank
1	A		G	1
2	A		T	2
3	C C C		G	3
			Z	4
			A	5
4	G		T	
5	G		Z	6
6	G		A C T	7
7	G			
8	T			
9	T		C	8
10	T		G	9
11	Z Z		C	10
			G	11

Siren, J., Valimaki, N. & Makinen, V. Indexing graphs for path queries with applications in genome research. *IEEE-ACM Trans. Comput. Biol. Bioinform.* **11**, 375–388 (2014).

(the transformation from 2 -> 3 is crucial to allow indexing)

The Graph FM-Index



Outgoing edge(s)			Incoming edge(s)	
Node rank	First		Last	Node rank
1	A		G	1
2	A		T	2
3	C C C		G	3
			Z	4
			A	5
4	G		T	
5	G		Z	6
6	G		A	7
7	G		C	
8	T		T	
9	T		C	8
10	T		G	9
11	Z		C	10
	Z		G	11

The Graph FM-Index

Note: We have an LF mapping here, just like a normal BWT

This 1-to-1 correspondence isn't possible without the graph transformation.

Hint: try and search for the pattern "GTG"

Outgoing edge(s)			Incoming edge(s)	
Node rank	First		Last	Node rank
1	A_0		G_0	1
2	A_1		T_0	2
3	C_0 C_1 C_2		G_1	3
			Z_0	4
4	G_0		A_0 T_1	5
5	G_1		Z_1	6
6	G_2		A_1 C_0 T_2	7
7	G_3			
8	T_0			
9	T_1		C_1	8
10	T_2		G_2	9
11	Z_0 Z_1		C_2	10
			G_3	11

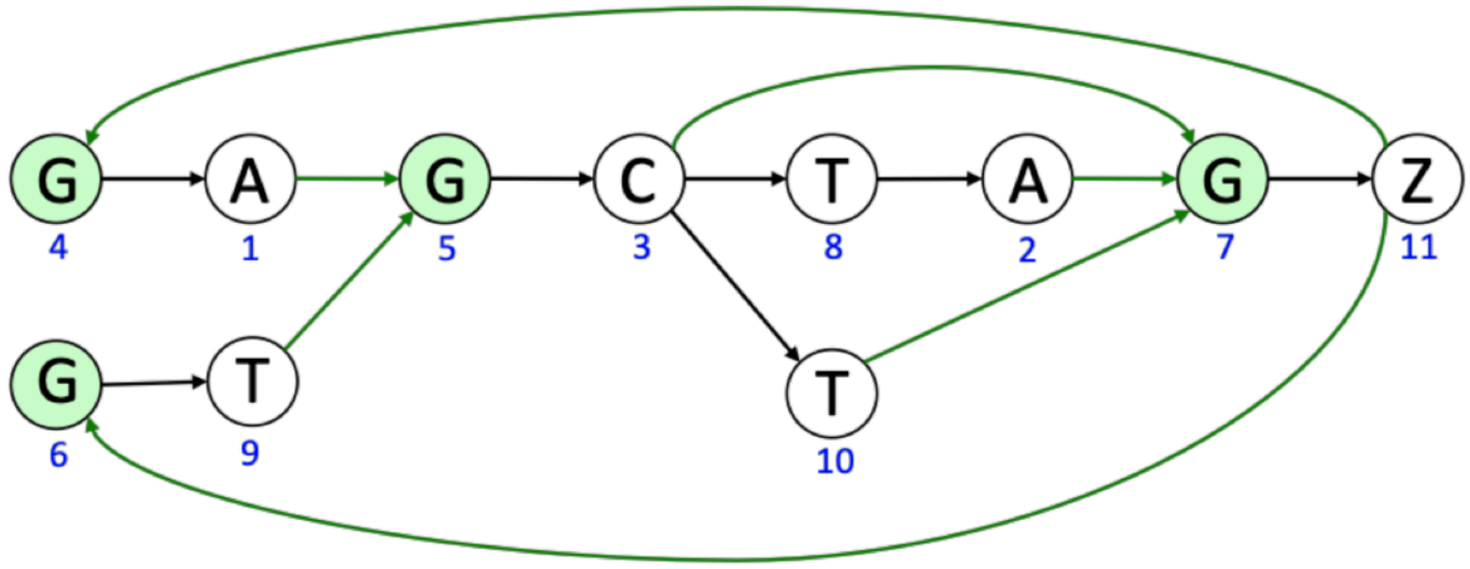
The Graph FM-Index

Outgoing edge(s)			Incoming edge(s)	
Node ID	First		Last	Node ID
1	A		G	1
2	A		T	2
3	C C C		G Z A T	3 4 5
4	G			
5	G		Z	6
6	G		A C T	7
7	G			
8	T			
9	T		C	8
10	T		G	9
11	Z Z		C G	10 11

Searching the Graph FM-Index

Outgoing edge(s)			Incoming edge(s)	
Node ID	First		Last	Node ID
1	A		G	1
2	A		T	2
3	C C C		G	3
			Z	4
4	G	1	A T	5
5	G		Z	6
6	G		A C T	7
7	G			
8	T			
9	T		C	8
10	T		G	9
11	Z Z		C	10
			G	11

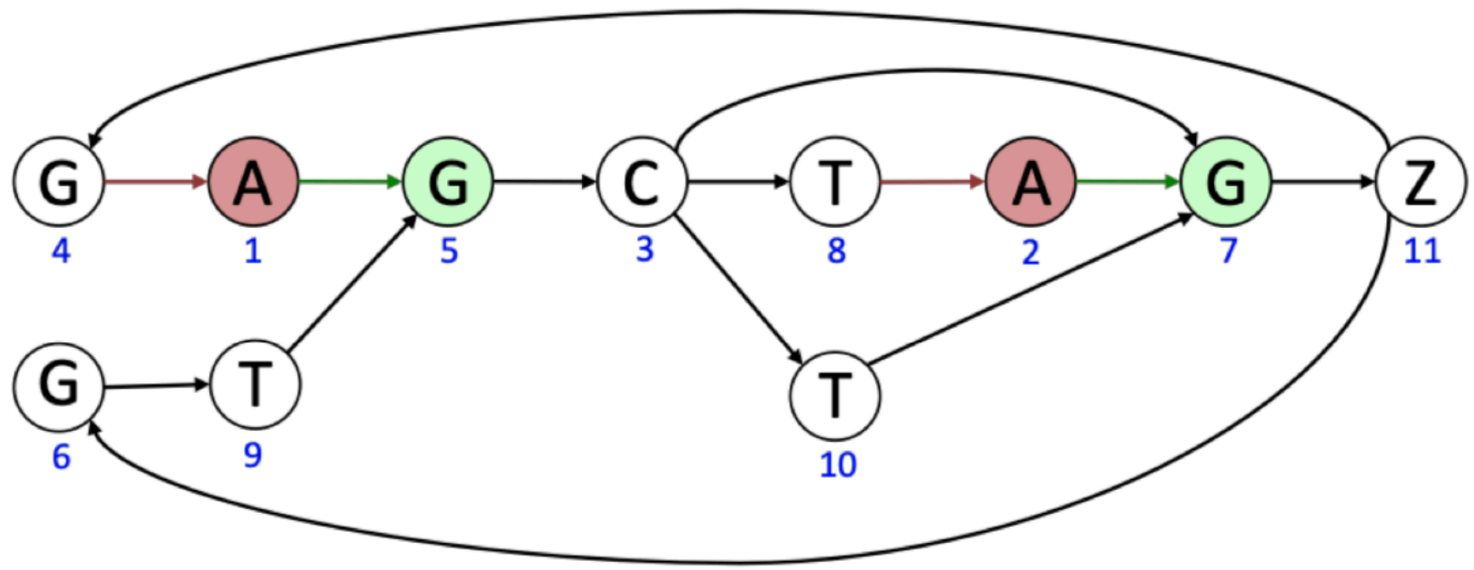
Query : TAG



Searching the Graph FM-Index

Outgoing edge(s)			Incoming edge(s)	
Node ID	First		Last	Node ID
1	A		G	1
2	A		T	2
3	C C C	2	G	3
			Z	4
4	G	1	A T	5
5	G		Z	6
6	G		A C T	7
7	G			
8	T			
9	T		C	8
10	T		G	9
11	Z Z		C	10
			G	11

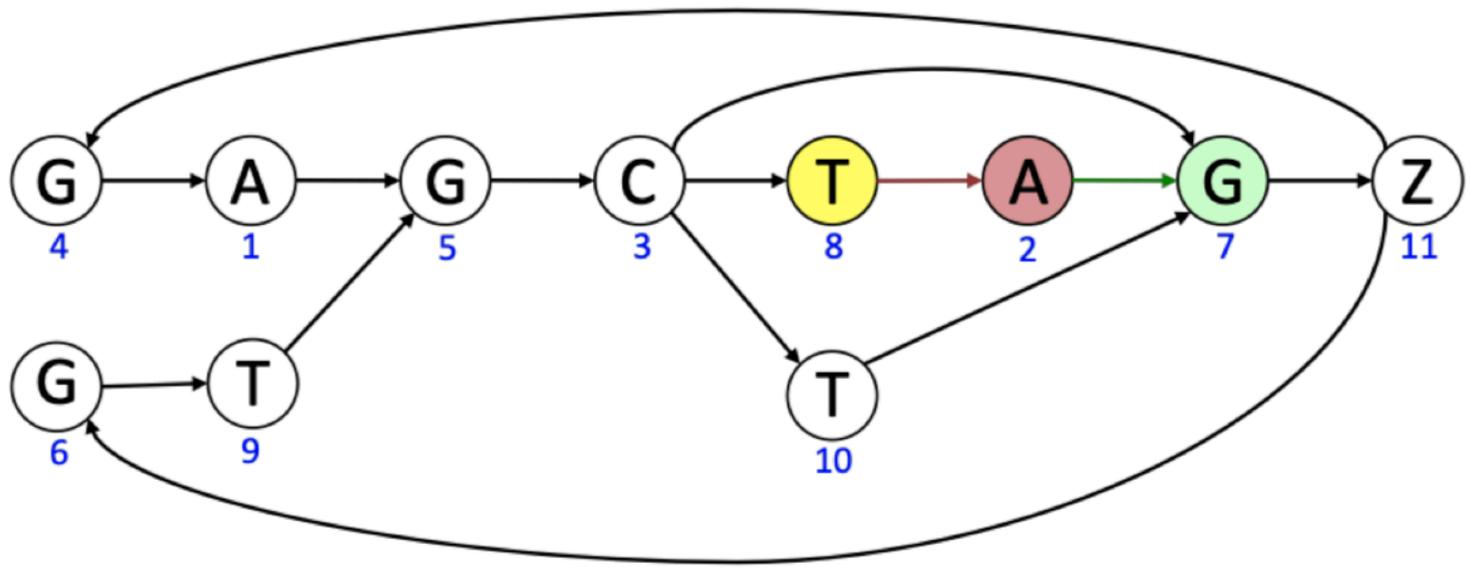
Query : TAG



Searching the Graph FM-Index

Outgoing edge(s)			Incoming edge(s)	
Node ID	First		Last	Node ID
1	A	3 →	G	1
2	A		T	2
3	C C C	2 ↖	G	3
			Z	4
			A T	5
4	G	1 ↗	Z	6
5	G		A C T	7
6	G			
7	G			
8	T			
9	T		C	8
10	T		G	9
11	Z Z		C	10
			G	11

Query : TAG



Searching the Graph FM-Index

Outgoing edge(s)			Incoming edge(s)	
Node ID	First		Last	Node ID
1	A		G	1
2	A		T	2
3	C		G	3
	C		Z	4
	C		A	5
4	G		T	
5	G		Z	6
6	G		A	7
7	G		C	
8	T		T	
9	T		C	8
10	T		G	9
11	Z		C	10
	Z		G	11

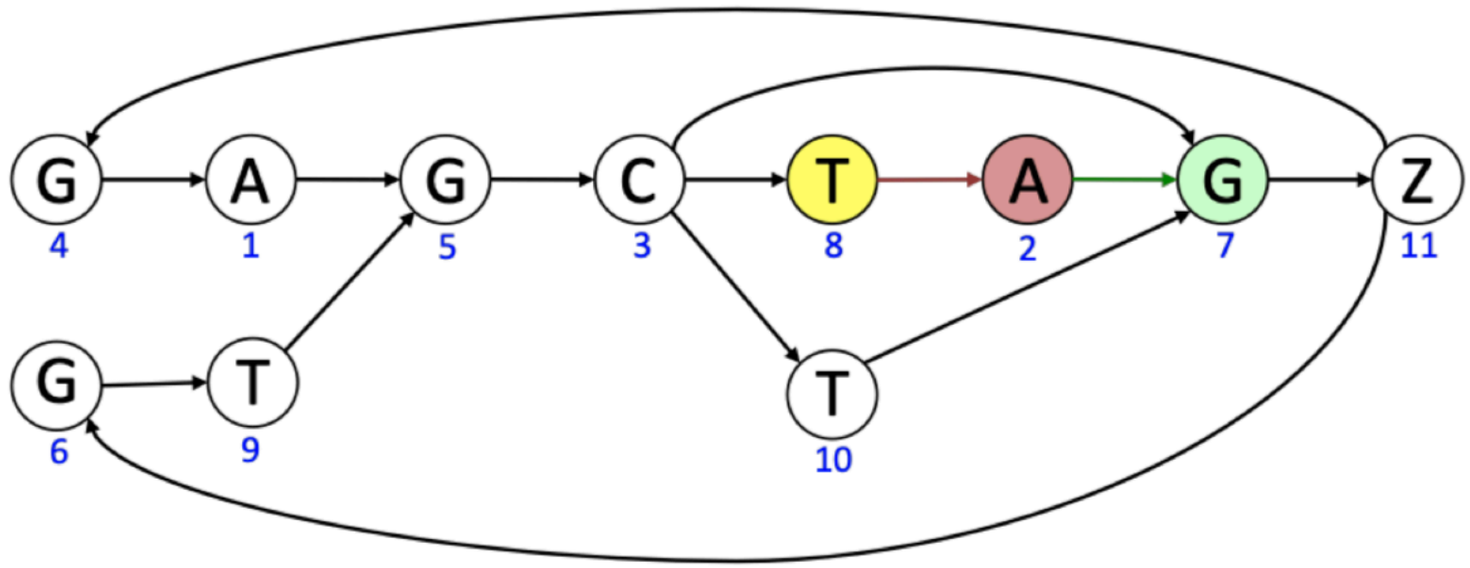
3

2

1

4

Query : TAG



The last step, to row 8, gives us the **ID** of the node corresponding to the prefix.

How to store the GFM efficiently

Outgoing edge(s)	
Node rank	First
1	A
2	A
3	C C
4	G
5	G
6	G
7	G
8	T
9	T
10	T
11	Z Z

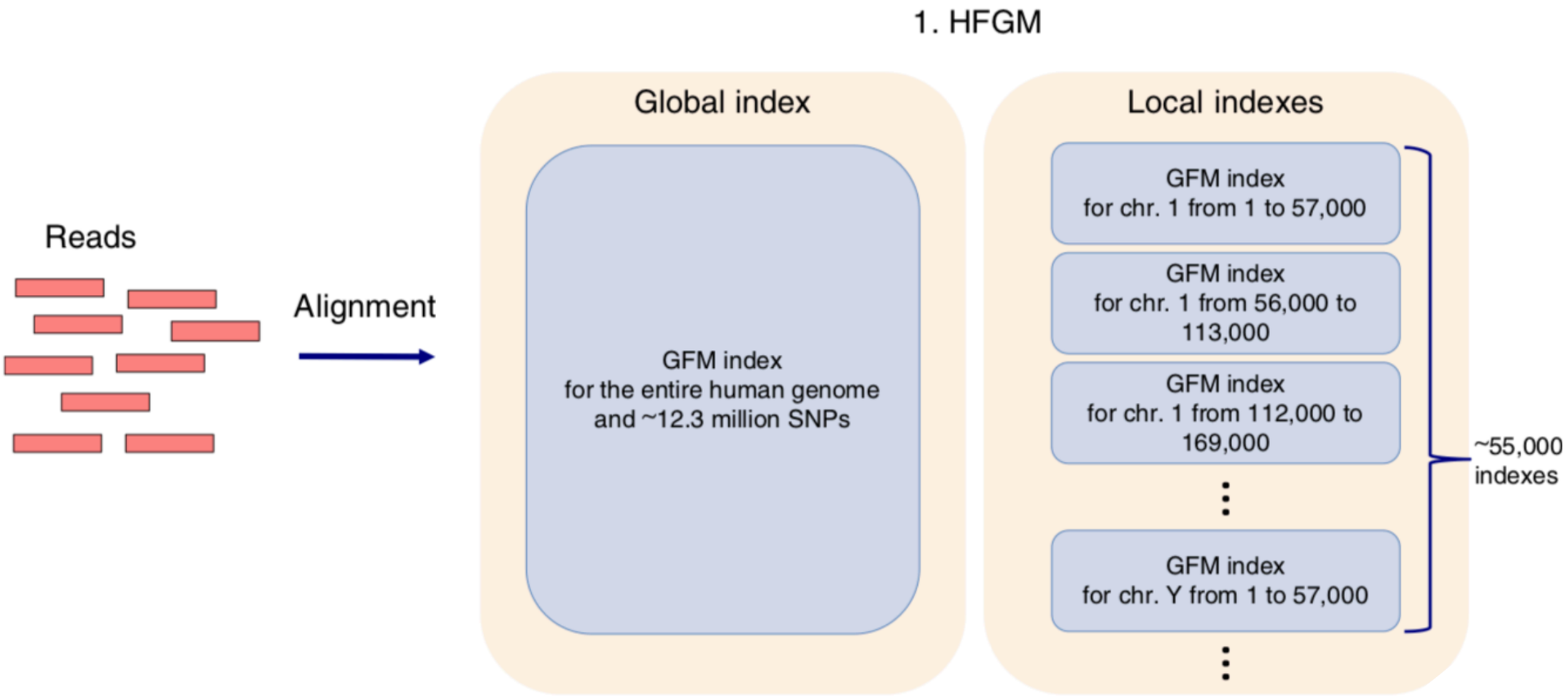
Incoming edge(s)	
Last	Node rank
G	1
T	2
G	3
Z	4
A T	5
Z	6
A C T	7
C	8
G	9
C	10
G	11

Outgoing edge(s)
Node rank
1
1
1
0 0
1
1
1
1
1
1
1
1
0

Incoming edge(s)	
Last	Node rank
10	1
11	1
10	1
00	1
00 11	1 0
00	1
00 01 11	1 0 0
01	1
10	1
01	1
10	1

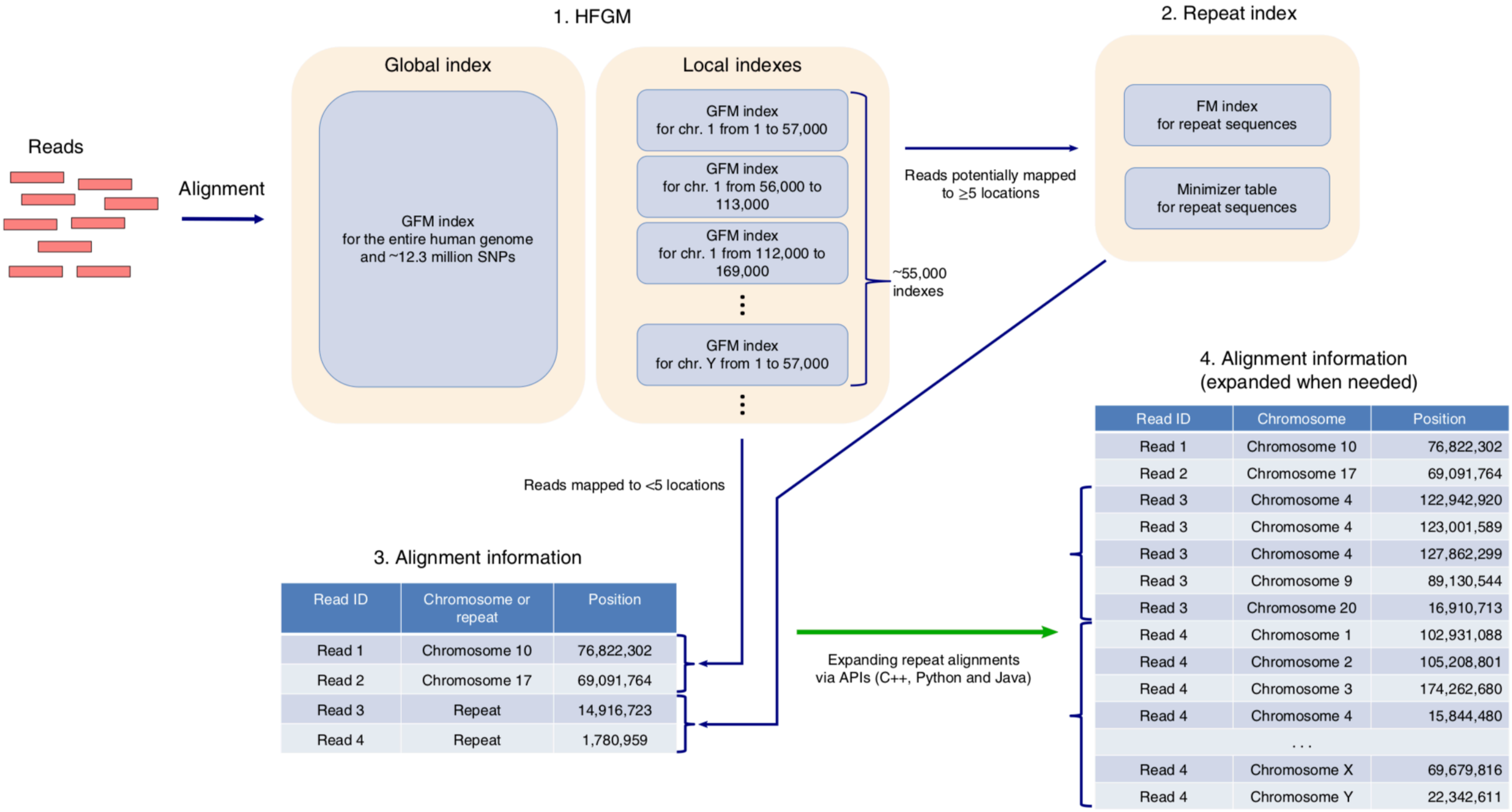
First	
A	2
C	3
G	4
T	3
Z	2

Uses same idea as HISAT to make GFM
Cache-efficient



Uses same idea as HISAT to make GFM
Cache-efficient

Special handling of repetitive sequences



Read ID	Chromosome or repeat	Position
Read 1	Chromosome 10	76,822,302
Read 2	Chromosome 17	69,091,764
Read 3	Repeat	14,916,723
Read 4	Repeat	1,780,959

Read ID	Chromosome	Position
Read 1	Chromosome 10	76,822,302
Read 2	Chromosome 17	69,091,764
Read 3	Chromosome 4	122,942,920
Read 3	Chromosome 4	123,001,589
Read 3	Chromosome 4	127,862,299
Read 3	Chromosome 9	89,130,544
Read 3	Chromosome 20	16,910,713
Read 4	Chromosome 1	102,931,088
Read 4	Chromosome 2	105,208,801
Read 4	Chromosome 3	174,262,680
Read 4	Chromosome 4	15,844,480
	. . .	
Read 4	Chromosome X	69,679,816
Read 4	Chromosome Y	22,342,611