Modelling haplotypes with respect to reference cohort variation graphs

Yohei Rosen, Jordan Eizenga and Benedict Paten
UC Santa Cruz Genomics Institute,
University of California Santa Cruz, Santa Cruz, CA 95064, USA

January 19, 2017

Abstract

Current statistical models of haplotypes are limited to cohorts of haplotypes which can be represented by arrays of values at linearly ordered bi- or multiallelic loci. These methods cannot model either structural variants or overlapping or nested variants. A variation graph is a mathematical structure can encode arbitrarily complex genetic variation. We present the first model which uses a variation graph representation of haplotypes. We present an algorithm to calculate the likelihood that a haplotype arose from a population through recombinations and demonstrate time complexity linear in haplotype length and sublinear in population size. We demonstrate mathematical extensions to allow modelling of mutations. Our results provide a starting point for haplotype inference on variation graphs. This is an essential step forward for clinical genomics and genetic epidemiology since it is the first haplotype model which can represent all sorts of variation in the population.

1 Background

Statistical modelling of individual haplotypes within population distributions of genetic variation dates back to Kingman's *n-coalescent* [5]. In general, the coalescent and other models describe haplotypes as generated from some structured state space via recombination and mutation events.

Whereas coalescent models are powerful generative tools, their computational complexity is unsuited to inference on observed haplotype data. Therefore, the dominant haplotype likelihood model used for statistical inference is Li and Stephens' 2003 model (LS) [7] and its various modifications. LS closely approximates the more exact coalescent models but admits implementations with rapid runtime.

Orthogonal to statistical models, another important frontier in genomics is the development of variation graphs [10]. This is a structure which encodes the wide variety of variation found in the population, including many types of variation which cannot be represented with conventional models. Variation graphs are a natural structure to represent reference cohorts for haplotype inference since they encode haplotypes in a canonical manner: as node sequences embedded in the graph [8].

In this paper, we present the first statistical model for haplotype modelling with respect to graph-embedded populations. We also describe an efficient algorithm for calculating haplotype likelihoods with respect to large reference panels. The algorithm makes significant use of Novak's graph positional Burrows-Wheeler transform (gPBWT) [8].

2 Encoding the full set of human variation

Haplotypes in the Kingman n-coalescent and Li-Stephens models are represented as sequences of values at linearly ordered, non-overlapping binary loci [5,6,7]. Some authors model multiallelic loci (for

example, single base positions taking on values of *A, C, T, G, or gap*) [6], but all assume that the entirety of genetic variation can be expressed by linearly ordered loci.

However, many types of genetic variation cannot be represented in this manner. Copy number variations, inversions or transpositions of sequence cannot be represented by linearly ordered loci. Large population cohorts such as the *1000 Genomes* project data [1] contain simple insertions, deletions and substitution at a sufficient density that these variants overlap or nest into structures not representable by linearly ordered sites.

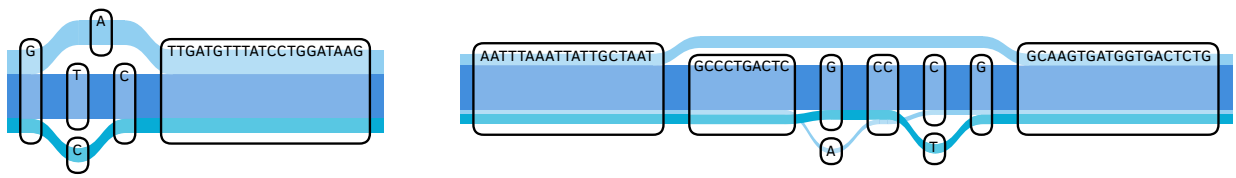

Figure 1: Overlapping, non-linearly orderable loci in a graph of *1000 Genomes* variation data for chromosome 22

In order to represent these more challenging types of variation, we use a *variation graph*. This is a type of *sequence graph*—a mathematical graph in which nodes represent elements of sequence, augmented with $5'$ and $3'$ sides, and edges are drawn between sides if the adjacency of sequence is observed in the population cohort [10]. Haplotypes are embedded as paths through oriented nodes in the graph. We are able to represent novel recombinations, deletions, copy number variations, or other structural events by adding paths with new edges to the graph, and novel inserted sequence by paths through new nodes.

3 Adapting the recombination component of Li and Stephens to graphs

The Li and Stephens model (LS) can be described by an HMM with a state space consisting of haplotypes and observations consisting of the haplotypes' alleles at loci [6,7]. Transitions between states correspond to recombinations and mutations are modelled within the emission probabilities. Since variation graphs encode full nucleic acid sequences rather than lists of sites we extend the model to allow recombinations at base-pair resolution rather than just between loci.

Let G denote a variation graph. Let $\mathcal{S}(G)$ be the set of all possible finite paths visiting oriented nodes of G. A path h in $\mathcal{S}(G)$ encodes a potential *haplotype*. A variation graph posesses an embedded *population reference cohort* H which is a multiset of haplotypes. Given a pair (G, H), we seek the likelihood $P(h|G, H)$ that h arose from haplotypes in H via recombinations.

Recall that every oriented node of G is labelled with a nucleic acid sequence. Therefore, every path $h \in \mathcal{S}$ corresponds to a nucleic acid sequence $seq(h)$ formed by concatenation of its node labels. We represent recombinations between by assembling subsequences of these sequences $seq(h)$ for $h \in H$. We call a concatenation of such subsequences a *recombination mosaic*.

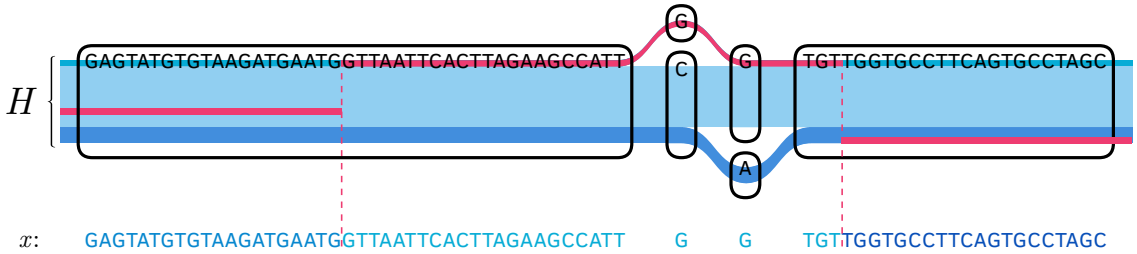

Figure 2: The pink path shows the recombination mosaic x superimposed on the embedded haplotypes H in our *1000 Genomes project* chr 22 graph; below x is mapped onto its nucleic acid sequence.

We can assign a likelihood to a mosaic x by analogy with the recombination model from Li and Stephens. Assume that nucleotide in x has precisely one successor in each $h \in H$ to which it could recombine. Then, between each base pair, we assign a probability π_r of recombining, and therefore a probability $(1 - (|H| - 1)\pi_r)$ of not recombining. Write π_c for $(1 - (|H| - 1)\pi_r)$.

By the same argument underlying the Li and Stephens recombination model, we then we have a probability of a given mosaic having arisen from (G, H) through recombinations:

$$P(x|G, H) = \pi_r^{R(x)} \pi_c^{|x|-R(x)} \tag{1}$$

We will use this to determine the probability $P(h|G, H)$ for a given $h \in \mathcal{S}(G)$, noting that multiple mosaics x can correspond to the same node path $h \in \mathcal{S}(G)$.

Given a haplotype $h \in \mathcal{S}(G)$, let $\chi(h)$ be the set of all mosaics involving the same path through the graph as h. The law of total probability gives

$$P(h|G, H) = \sum_{x \in \chi(h)} P(x|G, H) \tag{2}$$

$$= \sum_{x \in \chi(h)} \pi_c^{|h|-R(x)} \pi_r^{R(x)} = \pi_c^{|h|} \sum_{x \in \chi(h)} \left(\frac{\pi_r}{\pi_c}\right)^{R(x)} \tag{3}$$

Let $\rho := \frac{\pi_r}{\pi_c}$; then $P(h|G, H)$ is proportional to a $\rho^{R(x)}$-weighted enumeration of $x \in \chi(h)$.

We can extend this model by allowing recombination rate $\pi(n)$ and effective population size $|H|_{eff}(n)$ to vary across the genome according to node $n \in G$ in the graph. Varying the effective population size allows the model to remain sensible in regions traversed multiple times by cycle-containing haplotypes. In our basic implementation we will assume that $\pi(n)$ is constant and $|H|_{eff}(n) = |H|$, however varying these parameters does not add to the computational complexity of the model.

4 A linear-time dynamic programming for likelihood calculation

We wish to be able to calculate the sums $\sum_{x \in \chi(h)} \rho^{R(x)}$ efficiently. We will achieve this by traversing the node sequence h left-to-right, computing the sum for all prefixes of h. Write h_b for the prefix of h ending with node b.

Definition 1. A *subinterval* s of a haplotype h is a contiguous subpath of h. Two subintervals s_1, s_2 of haplotypes h_1, h_2 are *consistent* if $s_1 = s_2$ as paths, however we distinguish them as separate objects.

Definition 2. S_b^a is the set of subintervals s' of $h' \in H$ such that

1. there exists a subinterval s of h which begins with a, ends with b and is consistent with s'
2. there exists no such subinterval which begins with $a - 1$, the node before a in h *(left-maximality)*

Definition 3. For a given prefix h_b of h and a subinterval s' of a haplotype $h' \in H$, define the subset $\chi(h)_{s'} \subseteq \chi(h)$ as the set of all mosaics whose rightmost segment corresponds to s'.

The following result is key to being able to efficiently enumerate mosaics:

Claim 1. *If $s_1, s_2 \in S_b^a$ for some a, then there exists a recombination-count preserving bijection between $\chi(h_b)_{s_1}$ and $\chi(h_b)_{s_2}$.*

Proof. See supplement. □

Corollary 1. *If we define*

$$R_b(s_i) := \sum_{x \in \chi(h_b)_{s_i}} \rho^{R(x)} \tag{4}$$

then $R_b(s_1) = R_b(s_2)$ *if* $s_1, s_2 \in S_b^a$ *for some* a*. Called this shared value* $R_b(a)$*.*

Definition 4. A_b is the set of all nodes $a \in G$ such that S_b^a is nonempty.

Using these results, the likelihood $P(h_b|G,H)$ of the prefix h_b can be written as

$$P(h_b|G,H) = \pi_c^{|h_b|} \sum_{s_i} R_b(s_i) = \pi_c^{|h_b|} \sum_{a \in A_b} |S_b^a| R_b(a) \tag{5}$$

Let $b-1$ represent the node preceding b in h; it remains to show that if we know $R_{b-1}(a)$ for all $a \in A_{b-1}$, we can calculate $R_b(a)$ for all $a \in A_b$ in constant time with respect to $|h|$. This can be recognized by inspection of the following linear transformation. Proof that this computes $R_b(\cdot)$ from $R_{b-1}(\cdot)$ is straightforward but lengthy and therefore deferred to the supplement.

$$R_b(a) = \rho f_s(w,\ell)(A+B) + \mathbf{1}_{a \neq b}(1-\rho)\left(f_t(\ell) R_{b-1}(a) + \frac{f_s(w,\ell) + f_t(\ell)}{w} A \right) \tag{6}$$

where $w = \sum_a |S_b^a|$, $f_s(w,\ell) := (1+(w-1)\rho)^{\ell-1}$, $f_t(\ell) := (1-\rho)^{\ell-1}$, and A, B are the $|A_{curr}^{b-1}|$ element sums

$$A := \sum_{a \in A_{b-1}} |S_b^a| R_{b-1}(a), \qquad B := \sum_{a \in A_{b-1}} [|S_{b-1}^a| - |S_b^a|] R_{b-1}(a)$$

Assuming memoization of the polynomials $f_s(h,\ell), f_t(\ell)$, and knowledge of w, ℓ and all $|S_b^a|$'s, all $R_b(a)$'s can be calculated together in two shared $|A_{b-1}|$-element sums (to calculate A and $A+B$) followed by a single sum per $R_{b-1}(a)$. Therefore, by computing increasing prefixes h_b of h, we can compute $P(h|G,H)$ in time complexity which is $\mathcal{O}(nm)$ in $n = |h|$, and $m = max_b |A_b|$. The latter quantity is bounded by $|H|$ in the worst theoretical case; we will show experimentally that runtime is sublinear in $|H|$.

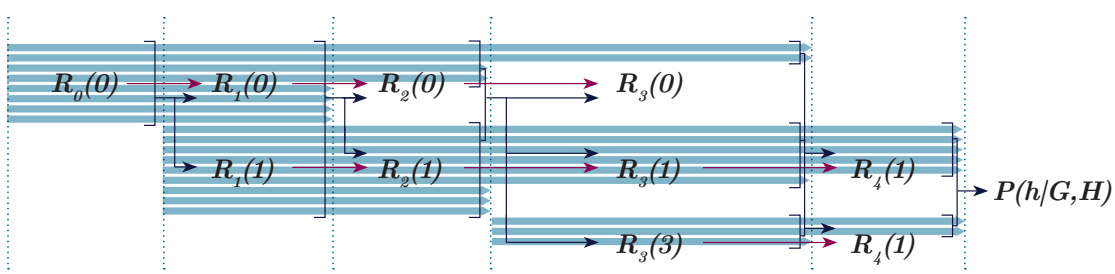

Figure 3: A sketch of the flow of information in the likelihood calculation algorithm described. Blue arrows a represent the *rectangular decomposition*, $R_\cdot(\cdot)$ are prefix likelihoods

5 Using the gPBWT to enumerate equivalence classes in linear time

Novak's graph positional Burrows Wheeler transform index (gPBWT) [8] is a succinct data structure which allows for linear-time subpath search in a variation graph. This is graph analogue of Durbin's positional Burrows Wheeler transform [3] used in Lunter's fast implementation of the Viterbi algorithm in the LS model [6]. Like other Burrows-Wheeler transform variants, its search function returns intervals in a path index, and therefore querying the number of subpaths in a graph matching a path is also linear-time in path length.

We demonstrate that we also need only perform a maximum of $|A_{b-1}|+1$ gPBWT operations and a corresponding amount of arithmetic to calculate all $|S_b^a|$ for a given b, provided that we have all $|S_{b-1}^a|$

Definition 5. $J_b^a :=$ the number of subpaths in H matching h between nodes a and b

This is $\mathcal{O}(n)$ in the gPBWT for $n = b - a$; in particular, it is $\mathcal{O}(1)$ given that we have the search interval to compute J_{b-1}^a.

Definition 6. $I_b^a := J_b^a - J_b^{a-1}$

Claim 2. $I_b^a = |S_b^a|$

Proof. By straightforward manipulation of definitions 2, 5, 6. □

It is then evident that $|A_{b-1}| + 1$ single-node search extension queries[1] are sufficient to determine $|S_b^a|$ for all $a \in A_b$. Determining these values for all $b \in h$ is therefore also $\mathcal{O}(nm)$ in $n = |h|$, and $m = max_b|A_b|$.

In practice, since $J_b^a = \sum_{i \leq a} I_b^a$, we can reduce the number of gPBWT queries even further by employing a recursive partitioning of intervals to avoid querying those whose values we can tell are unchanged.

6 Modelling mutations

We can assign to two haplotypes h, h' the probability $P_m(h|h')$ that h arose from h' through a mutation event. As in Li and Stephens, we can assume conditional independence such that

$$P_{tot}(h|G, H) = \sum_{h' \in seq(G)} P_m(h|h')P_r(h'|G, H)$$

It is reasonable to make the simplifying assumption that $P_{mut}(h|h') = 0$ unless h' differs from h exclusively at simple bubbles since more dramatic mutation events are vanishingly rare. This assumption is implicitly contained in the n-coalescent and Li and Stephens models by their inability to model more complex mutations.

Detection of all simple bubbles in the graph traversed by h can be achieved in linear time with respect to the length of h. The number of such paths remains exponential in the number of simple sites. However, our model allows us to perform branch-and-bound type approaches to exploring these paths. This is possible since we can calculate upper bounds for likelihood knowing either only a prefix, or in cases where we have censored the nodes which a path traverses through an encapsulated region in the middle of the path.

Furthermore, it is evident from our algorithm that if two paths share the same prefix, then we can reuse the calculation over this prefix. Furthermore, if two paths share the same suffix, in general we only need to recompute the $|S_b^a|$ values for a small number of nodes.

[1] the additional one is to compute J_b^b

7 Implementation

To test our methods, we implemented the algorithms describe in C++, using elements of the variation graph toolkit *vg* [4] and the gPBWT index implementation in *xg* [8]. This can be found in the *"haplotypes"* branch at https://github.com/yoheirosen/vg. Since no competing graph-based haplotype models exist, we were not able to provide comparative performance data; absolute performance on a single machine is presented instead.

8 Results

We assessed time complexity of our likelihood algorithm algorithm using the implementation described above on a single thread of an Intel Xeon X7560 running at 2.27 GHz.

We created a 5008 haplotype graph of human chromosome 22, using the 1000 Genomes Phase 3 VCF on the hg19 assembly created using *vg* and data from the *1000 Genomes* project [1]. We calculated likelihoods of randomly selected subhaplotypes of specific lengths.

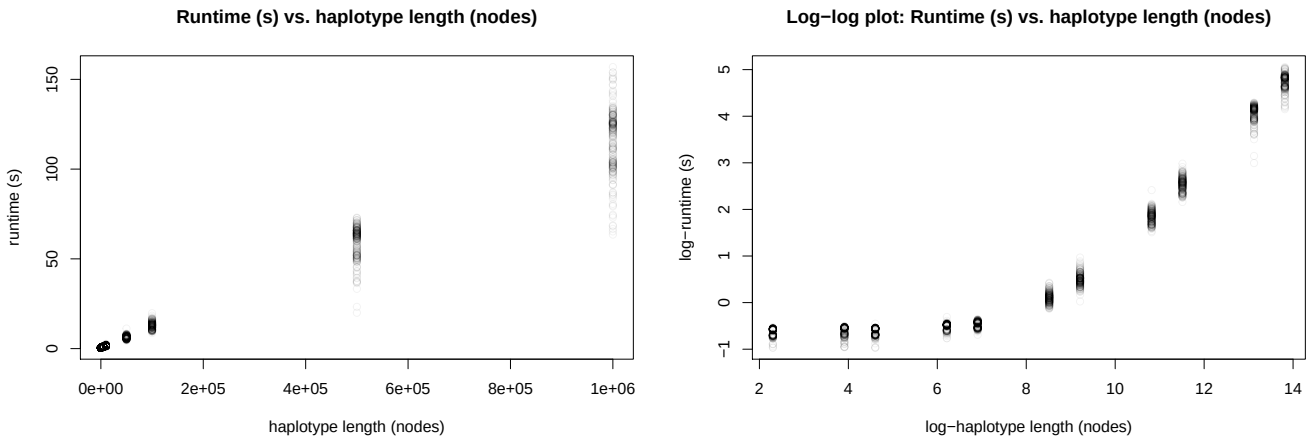

Figure 4: Runtime (s) vs. haplotype length (nodes) for chromosome 22 1000 Genomes data

The observed relationship of runtime to haplotype length is consistent with $\mathcal{O}(n)$ time complexity with respect to $n = |h|$. Average time for per million nodes was 116 seconds; with the node density of a variation graph derived from the 1000 Genomes dataset on chromosome 22, 1 million nodes corresponds to on average 16.0 million base pairs.

We built graphs of subpopulations of specific sizes to asses the effect of reference cohort size on runtime. Random subsets of the *1000 Genomes* data were made using *vcftools* [2] and our graph-building process was repeated. Five replicate subset graphs were made per population size.

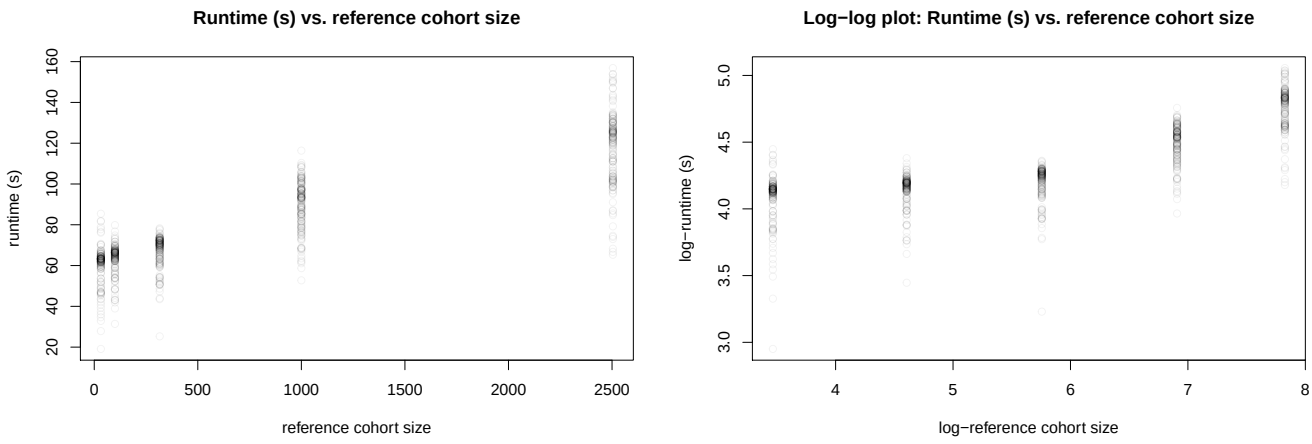

Figure 5: Runtime (s) vs. reference cohort size (diploid individuals) for chromosome 22 1000 Genomes data

We observe an asymptotically sublinear relationship between runtime and reference cohort size.

9 Conclusions

We have introduced a method of representing the shared sequence relationship of an individual haplotype to a reference cohort represented in a variation graph. We have extended this into an efficient algorithm for haplotype inference based on Novak's gPBWT [8]. We applied this method to a full-chromosome graph consisting of 5008 haplotypes from the 1000 Genomes data set to show that this algorithm can efficiently model recombination with respect to both long sequences and large reference cohorts. This is an important proof of concept for translating haplotype inference to the richer set of variation representable on graphs.

Our implementation does not model mutation. This depends on being able to efficiently calculate likelihoods for similar haplotypes. There exist mathematical generalizations of our algorithm which allow us to reuse information to compute the likelihood of haplotypes which are similar with respect to their embedding in the graph. Methods to most efficiently explore this space of haplotypes are necessary to allow extension to a mutation model for graphs. This is a subject of our continuing research.

10 Acknowledgements

Y.R. is supported by a Howard Hughes Medical Institute Medical Research Fellowship. This work was also supported by the National Human Genome Research Institute of the National Institutes of Health under Award Number 5U54HG007990 and grants from the W.M. Keck foundation and the Simons Foundation. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

[1] 1000 Genomes Project Consortium, et al.: A global reference for human genetic variation. Nature 526(7571), 68-74 (2015)

[2] Danecek, P., Auton, A., Abecasis, G., Albers, C.A., Banks, E., DePristo, M.A., Handsaker, R.E., Lunter, G., Marth, G.T., Sherry, S.T., McVean, G.: (2011). The variant call format and VCFtools. Bioinformatics, 27(15), 2156-2158 (2011)

[3] Durbin, R.: Efficient haplotype matching and storage using the positional Burrows-Wheeler transform (PBWT). Bioinformatics 30(9), 1266-1272 (2014)

[4] Garrison, E.: vg: the variation graph toolkit (2016), https://github.com/vgteam/vg/blob/80e823f5d241796f10b7a

[5] Kingman, J.F.C.: On the genealogy of large populations. Journal of Applied Probability, 19, 27-43 (1982)

[6] Lunter, G.: Fast haplotype matching in very large cohorts using the Li and Stephens model. bioRxiv (2016), http://biorxiv.org/content/early/2016/04/12/048280

[7] Li, N., Stephens, M.: Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. Genetics, 165(4), 2213-2233 (2013)

[8] Novak, A., Garrison, E., Paten, B.: A Graph Extension of the Positional Burrows-Wheeler Transform and Its Applications. bioRxiv (2016) http://biorxiv.org/content/early/2016/05/02/051409

[9] Paten, B., Novak, A., Garrison, E., Hickey, G.: Superbubbles, ultrabubbles, and cacti. In preparation (2016)

[10] Paten, B., Novak, A., Haussler, D.: Mapping to a reference genome structure. ArXiv e-prints (Apr 2014), http://arxiv.org/abs/1404.5010

11 Appendix A: An $\mathcal{O}(n \cdot m)$ implementation of the rectangular decomposition construction

Suppose that we wish to find subhaplotypes embedded in the graph which are consistent with a query sequence h of nodes. In brief, in the gPBWT, indexing information for haplotypes is stored in such a way that this can be achieved by calling a function StartSearchAtNode($Node$) on the first node of h, which returns a search interval $gPBWTInt$ of a form analogous to the search interval of a Burrows–Wheeler Transform based index of sequences. This search interval is extended by calling an operation Extend($gPBWTInt, Node$) to extend this search with each additional node in h. Finally, this search interval can be converted into a count of matching subhaplotypes using a function Count(gPBWTInt). It is shown in [8] that StartSearchAtNode, Extend and Count all admit $\mathcal{O}(1)$ implementations.

It is evident that this search process yields a function CountHaplotypeMatches(h) which is $\mathcal{O}(n)$ in the length $|h|$ of h in nodes. Let $h_1 h_2 h_3 \ldots h_{|h|-1} h_{|h|}$ denote the node sequence of h. Using CountHaplotypeMatches we can identify the set A of nodes in h such that either $J_a^{a-1} \neq J_{a-1}^{a-1}$ or $I_a^a \neq 0$ in $\mathcal{O}(n)$ independent length-2 subhaplotype count queries:

Algorithm 1 Identifying A, the set of "relevant" nodes

```
function BuildA(h, B[])
    A ← [1]
    ht_prev ← |B[h_1]|
    for i = 2, ..., |h| do
        ht_new ← |B[h_i]|
        J_i^{i-1} ← CountHaplotypeMatches(h_{i-1} h_i)
        if J_i^{i-1} < ht_new or ht_prev > ht_new then
            Append(A, i)
        ht_prev ← ht_new
```

Given that we have constructed A, we can determine the rest of the rectangular decomposition and all of the J-values according to the following algorithm:

Algorithm 2 Building the J's and A_{curr}'s

```
function BuildJs(h, B[])
    J_1^1 ← |B[h_1]|
    A_curr^1 ← 1
    for i ∈ A do
        A_curr^i ← []
        if |B[h_i]| > J_i^i then
            Append(A_curr^i, i)
            S_i ← StartSearch(h_i)
        for j ∈ A_curr^{i-1} do
            S_j ← Extend(S_j, h_i)
            if Count(S_j) ≠ 0 then
                J_i^j ← Count(S_j)
                Append(A_curr^i, j)
            else
                break
```

12 Appendix B: Arithmetic for derivation of Equation ??

Here we lay out the arithmetic to derive Equation (6), which is used in our iterative computation of likelihood of a haplotype h with respect to a population reference cohort H embedded in a variation graph G. The reasoning is straighforward but involves many subcases which require care.

12.1 Notation

Definition 7. A haplotype is a sequence of nodes $n_1 \to \cdots \to n_{|h|}$ in a variation graph. The base sequence of a haplotype is the sequence of DNA bases spelled by its node labels. A haplotype subinterval is a contiguous subsequence of a haplotype. A haplotype base sequence subinterval is analogously defined. Denote by $|h|$ the length of a haplotype base sequence in base pairs.

Definition 8. Haplotypes h, h' are consistent if $|h| = |h'|$ and $n_i = n'_i \, \forall i$.

Definition 9. A mosaic of haplotypes x consistent with h is a vector $\langle x_{(i)} \rangle$ of subintervals of base sequences of haplotypes in H whose concatenation is consistent with the base sequence of h. The

recombination count $R(x)$ is one less than the number of elements in $\langle x_{(i)} \rangle$. NB: defining these in terms of base sequence rather than node subintervals permits recombination within nodes. Recall Figure 3 from the main text.

Definition 10. $\chi(h)$ is the set of all mosaics x consistent with h. $\chi(h)^R$ is the subset with $R(x) = R$. $\chi(h)[,g]$ is the subset whose final subinterval is a subinterval of g. $\chi(h)[g,]$ is that with initial subinterval a subinterval of g. $|\chi(h)|$ is the number of elements in $\chi(h)$.

12.2 Arithmetic shortcuts

Lemma 1. *There exists a partition of h into subintervals $h_1, h_2, \ldots, h_n$ such that if a haplotype $g \in H$ has a subinterval consistent with a subinterval of h_i then it has a subinterval consistent with all of h_i.*

Proof. It is straightforward to verify that the intervals between successive nodes in the set A described in the main text produce such a partition of h. □

This is important because we will show that it is simple to calculate $|\chi(h_i)|$ within any interval with this property.

The following is a more notationally precise statement of Lemma 1 from the main text:

Lemma 2. *For any $b \in A, a \leq b$, given that f and g are members of the same equivalence class S_b^a of haplotypes, the haplotype mosaics $\chi^R(h_{[0,b]})[,f]$ and $\chi^R(h_{[0,b]})[,g]$ consistent with the subinterval $h_{[0,b]}$ and ending with subintervals of f and g are in bijective correspondence.*

Proof. We assume that $g \neq f$ else this is trivial. Consider any mosaic x in $\chi^R(h_{[0,b]})[,f]$. Given $x = \langle x_1, x_2, \ldots, x_{R+1} \rangle$, let $j = \max\{i \in 1, \ldots, R | x_i$ is not a subinterval of g or $f\}$. We will construct a mosaic $y = \langle y_1, y_2, \ldots, y_{R+1} \rangle$ such that for all $i \leq j$, $y_i = x_i$, and for all $i > j$, y_i is the subinterval of the same length as x_i but derived from the opposite haplotype of the pair f, g.

The concatenation $y_1 y_2 \cdots y_{R+1}$ is consistent with $h_{[0,b]}$ since given that both $f, g \in S_b^a$, the first node of y_{j+1} must be at or after a. Therefore clearly $y_i \in \chi^R(h_{[0,b]})[,g]$ since its final subinterval corresponds to g. The inherent invertibility of this transformation proves that it is a bijection. □

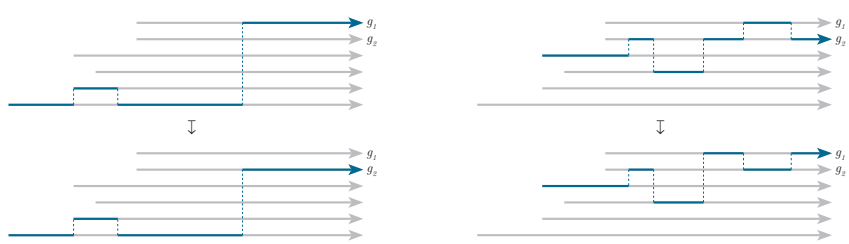

Figure 6: Visual proof of the above lemma by explicit construction of the bijection involved

Lemma 3. *Suppose that h_i is a subinterval of h such that if a haplotype $g \in H$ has a subinterval consistent with a subinterval of h_i then it has a subinterval consistent with all of h_i. Then suppose that $f_1, f_2, g \in H$, and all have subintervals consistent with all of h_i. Then for all $R < |h_i|$ there is a bijection between $\chi(h_i)[f_1, g]$ and $\chi(h_i)[f_2, g]$.*

Proof. The proof imitates that of the previous lemma. □

12.3 The case of a single simple interval h_i

Suppose that h_i is an interval of the form in Lemma 2, ℓ base pairs in in length, and has subintervals of ht haplotypes of H consistent with it. Consider $f, g \in H$ such that both have subintervals consistent with h_i. Suppose that we wish to calculate, for some $R < \ell$, the number $|\chi^R(h)[,g]|$ of mosaics consistent with h_i having R recombinations and ending with haplotype g. To calculate $|\chi(h_i)|$ within an interval of the form above, we need only calculate

1. $|\chi^R(h_i)[g,g]|$, the number of paths both beginning on and ending on g and
2. $|\chi^R(h_i)[f,g]|$, the number of paths beginning on $f \neq g$ and ending on g, which, by lemma 4, is the same for all such f.

Consider $R = \ell - 1$. It is clear that

$$\sum_j |\chi^R(h_i)[j,g]| = (ht-1)^R \tag{7}$$

Lemma 4 tells us that all haplotypes $f \neq g$ are equivalent for the purposes of enumeration, therefore we write $\neg g$ to denote any arbitrary representative $f \neq g$. There are $ht - 1$ such haplotypes.

$$|\chi^R(h_i)[g,g]| + (ht-1)|\chi^R(h_i)[\neg g,g]| = (ht-1)^R \tag{8}$$

We begin by calculating $|\chi^R(h_i)[\neg g,g]|$. Consider first $\ell = R+1 = 1$, for which, given the lack of possible recombinations, $|\chi^R(h_i)[\neg g,g]| = 0$. For $\ell = R+1 = 2$, any $x \in \chi^R(h_i)[\neg g,g]$ must at its second node visit a haplotype which is neither g nor the $\neg g$ under consideration, therefore $|\chi^R(h_i)[\neg g,g]| = (ht-2)$. Suppose now that, for arbitrary $\ell = R$, we know $|\chi^R(h_i)[\neg g,g]|$. Then, counting the $(ht-1)$ possible haplotypes before finally recombining to g shows us that

$$|\chi^{R+1}(h_i)[g,g]| = (ht-1)|\chi^R(h_i)[\neg g,g]| \tag{9}$$

By (8), we know that

$$|\chi^{R+1}(h_i)[\neg g,g]| = \frac{(ht-1)^R - |\chi^{R+1}(h_i)[g,g]|}{(ht-1)}$$

Which by (9) implies

$$|\chi^{R+1}(h_i)[\neg g,g]| = \frac{(ht-1)^R - (ht-1)|\chi^R(h_i)[\neg g,g]|}{(ht-1)}$$
$$\implies |\chi^{R+1}(h_i)[\neg g,g]| = (ht-1)^{R-1} - |\chi^R(h_i)[\neg g,g]| \tag{10}$$

Using (10) as the induction step with base case $\ell = R+1 = 2$ we find that $\forall \ell = R+1 \geq 2$

$$|\chi^R(h_i)[\neg g,g]| = \frac{(ht-1)^{R-1} + (-1)^R}{ht}$$

We now relax the restriction that $R = \ell$. For given $R < \ell$ each subset of nodes at which recombinations happen will define an additional set of possible recombinations. Counting all possible such subsets

$$|\chi^R(h_i)[\neg g,g]| = \binom{\ell-1}{R}\frac{(ht-1)^{R-1} + (-1)^R}{ht}$$

and

$$|\chi^R(h_i)[g,g]| = (ht-1)|\chi^R(h_i)[\neg g,g]|$$

.

12.4 Extending a computation for a prefix by a simple subinterval h_i

To extend our ability to calculate $|\chi(h)|$ beyond the single interval h_i, suppose we have a partition $\{h_1, h_2, \ldots, h_n\}$ of h into subintervals of the form in Lemma 2. Let $b \in A$ such that b is a node on the boundary of such an interval, let $h_{[0,b-1]}$ be the prefix of h formed by concatenation of the subintervals preceding node b, and let $h_{[b-1,b]}$ be the subinterval beginning with node b.

Suppose now that we have calculated each $|\chi^R(h_{[0,b-1]})[,f]|$ and now wish to calculate these values up to b, the node in A succeeding $b-1$. By Lemma 2, the intervening sequence $h_{[b-1,b]}$ is of the form for which we have just calculated $|\chi^R(h)[g,g]|$ and $|\chi^R(h)[\neg g,g]|$. We divide this into cases.

Case 1: Suppose that f has no subinterval consistent with $h_{[b,b+1]}$, that is, $f \in S^a_{b-1}$ for some a but $f \notin S^a_b$. Then any mosaic extending any mosaic in $\chi^R(h_{[0,b]})[,f]$ must recombine. Since $f \notin S^a_b$, there are $ht := J^b_b$ possible haplotypes to which this recombination at $b-1 \to b$ may occur. Let ℓ be the length (in base pairs) of the interval $b-1$ to b, then $\forall R' < \ell(b)$ we have previously calculated in (8) that

$$|\chi^{R'}(h_{[b-1,b]})[,g]| = \binom{\ell-1}{R'}(ht-1)^{R'-1}$$

and therefore, where we write $\chi^R(h_{[0,b-1]})[,f] \oslash \chi^{R'}(h_{[b-1,b]})[,g]$ for the set of mosaics formed by continuing mosaics in $\chi^R(h_{[0,b-1]})[,f]$ such that they recombine between $h_{[0,b-1]}$ and $h_{[b-1,b]}$ and end with a subinterval of g,

$$|\chi^R(h_{[0,b-1]})[,f] \oslash \chi^{R'}(h_{[b-1,b]})[,g]| = |\chi^R(h_{[0,b-1]})[,f]|\binom{\ell-1}{R'}(ht-1)^{R'-1} \tag{11}$$

Case 2: Suppose now that we know $|\chi^R(h_{[0,b-1]})[,f]|$, and $f \in S^a_b$ for some a, that is, f does have a subinterval consistent with $h_{[b-1,b]}$

There are two subcases: either there is, or there is not a recombination between the last base in $h_{[0,b-1]}$ and the subsequent base at the beginning of $h_{[b-1,b]}$. Suppose that there is not. In this case, where we write $\chi^R(h_{[0,b-1]})[,f] \ominus \chi^{R'}(h_{[b-1,b]})[,g]$ for the set of mosaics formed by continuing mosaics in $\chi^R(h_{[0,b-1]})[,f]$ such that they do not recombine between $h_{[0,b-1]}$ and $h_{[b-1,b]}$ and such that they do end with a subinterval of g,

$$|\chi^R(h_{[0,b-1]})[,f] \ominus \chi^{R'}(h_{[b-1,b]})[,g]| = |\chi^R(h_{[0,b-1]})[,f]||\chi^{R'}(h_{[b-1,b]})[f,g]| \tag{12}$$

such that if $f \neq g$

$$|\chi^R(h_{[0,b-1]})[,f] \ominus \chi^{R'}(h_{[b-1,b]})[,g]| = |\chi^R(h_{[0,b-1]})[,f]||\chi^{R'}(h_{[b-1,b]})[\neg g,g]| \tag{13}$$

else

$$|\chi^R(h_{[0,b-1]})[,f] \ominus \chi^{R'}(h_{[b-1,b]})[,g]| = |\chi^R(h_{[0,b-1]})[,f]||\chi^{R'}(h_{[b-1,b]})[g,g]| \tag{14}$$

The other subcase is that there is a recombination between the last base in $h_{[0,b-1]}$ and the subsequent base at the beginning of $h_{[b-1,b]}$. In this case if $f \neq g$,

$$|\chi^R(h_{[0,b-1]})[,f] \oslash \chi^{R'}(h_{[b-1,b]})[,g]| = |\chi^R(h_{[0,b-1]})[,f] \oslash \chi^{R'}(h_{[b-1,b]})[g,g]| \quad + \sum_{f' \neq f,g} |\chi^R(h_{[0,b-1]})[,f] \oslash \chi^{R'}(h_{[b-1,b]})[f,g]| \tag{15}$$

$$= |\chi^R(h_{[0,b-1]})[,f]||\chi^{R'}(h_{[b-1,b]})[g,g]|| \quad + \underbrace{(ht-2)|\chi^R(h_{[0,b-1]})[,f]||\chi^{R'}(h_{[b-1,b]})[\neg g,g]|}_{\text{by Lemma 4}} \tag{16}$$

else

$$|\chi^R(h_{[0,b-1]})[,f] \oslash \chi^{R'}(h_{[b-1,b]})[,g]| = (ht-1)|\chi^R(h_{[0,b-1]})[,f]||\chi^{R'}(h_{[b-1,b]})[\neg g,g]| \tag{17}$$

12.5 Deriving the Formula for $P(h|G,H)$

Suppose that we have calculated $|\chi(h_{[0,b-1]})[,f]|$ for all f and now wish to calculate $|\chi(h_{[0,b]})[,g]|$ for some $g \in S_b^a$, for some $a \leq b$.

Note that as defined in the main text, $R_{b-1}(a) = |\chi(h_{[0,b-1]})[,f]|$ for the a such that $f \in S_b^a$. This means this calculatiuon will in fact give us the formula with which to calculate $\overrightarrow{R_b}$ given $\overrightarrow{R_{b-1}}$. Let us write $R_b(f)$ for $R_b(a)$ such that $f \in S_b^a$.

Accounting for all prefixes in $\chi(h_{[0,b-1]})$ which can produce mosaics in $\chi(h_{[0,b]})[,g]$, then

$$|\chi(h_{[0,b]})[,g]| = \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2)}|\chi^{R_1}(h_{[0,b-1]}) \ominus \chi^{R_2}(h_{[b-1,b]})[,g]| \; + \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2+1)}|\chi^{R_1}(h_{[0,b-1]}) \oslash \chi^{R_2}(h_{[b-1,b]})[,g]|$$

$$= \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \Big(\rho^{(R_1+R_2)}|\chi^{R_1}(h_{[0,b-1]}) \ominus \chi^{R_2}(h_{[b-1,b]})[,g]| \; + \rho^{(R_1+R_2+1)}|\chi^{R_1}(h_{[0,b-1]}) \oslash \chi^{R_2}(h_{[b-1,b]})[,g]| \Big)$$

$$\begin{aligned} = \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2)} \Bigg(& \sum_{a<b} \sum_{\substack{f \in S_b^a \\ f \neq g}} |\chi^{R_1}(h_{[0,b-1]})[,f] \ominus \chi^{R_2}(h_{[b-1,b]})[f,g]| \\ & + |\chi^{R_1}(h_{[0,b-1]})[,g] \ominus \chi^{R_2}(h_{[b-1,b]})[g,g]| \\ & + \rho \Big(|\chi^{R_1}(h_{[0,b-1]})[,g] \oslash \chi^{R_2}(h_{[b-1,b]})[,g]| \\ & \quad + \sum_{a<b} \sum_{\substack{f \in S_b^a \\ f \neq g}} |\chi^{R_1}(h_{[0,b-1]})[,f] \oslash \chi^{R_2}(h_{[b-1,b]})[,g]| \\ & \quad + \sum_{a<b} \sum_{f \notin S_b^a} |\chi^{R_1}(h_{[0,b-1]})[,f] \oslash \chi^{R_2}(h_{[b-1,b]})[,g]| \Big) \Bigg) \end{aligned}$$

$$
\begin{aligned}
= \sum_{\substack{R_1<|h_{[0,b-1]}|\\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2)}\Bigg(&\sum_{a<b}\sum_{\substack{f\in S_b^a\\ f\neq g}} \underbrace{|\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|}_{\text{by (13)}}\\
&+\underbrace{|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[g,g]|}_{\text{by (14)}}\\
&+\rho\Bigg(\underbrace{|\chi^{R_1}(h_{[0,b-1]})[,g]|(ht-1)|\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|}_{by(17)}\\
&\quad+\sum_{a<b}\sum_{\substack{f\in S_b^a\\ f\neq g}} \underbrace{|\chi^{R_1}(h_{[0,b-1]})[,f]|\big(|\chi^{R_2}(h_{[b-1,b]})[g,g]|+(ht-2)|\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|\big)}_{\text{by (16)}}\\
&\quad+\sum_{a<b}\sum_{f\notin S_b^a}|\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[,g]|\Bigg)\Bigg)
\end{aligned}
$$

$$
\begin{aligned}
= \sum_{\substack{R_1<|h_{[0,b-1]}|\\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2)}\Bigg(&\sum_{a<b}\sum_{f\in S_b^a}|\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|-|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|\\
&+|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[g,g]|\\
&+\rho\Bigg(|\chi^{R_1}(h_{[0,b-1]})[,g]|\big(|\chi^{R_2}(h_{[b-1,b]})[,g]|-|\chi^{R_2}(h_{[b-1,b]})[g,g]|\big)\\
&\quad+\sum_{a<b}\sum_{\substack{f\in S_b^a\\ f\neq g}}|\chi^{R_1}(h_{[0,b-1]})[,f]|\big(|\chi^{R_2}(h_{[b-1,b]})[,g]|-|\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|\big)\\
&\quad+\sum_{a<b}\sum_{f\notin S_b^a}|\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[,g]|\Bigg)\Bigg)
\end{aligned}
$$

$$
\begin{aligned}
= \sum_{\substack{R_1<|h_{[0,b-1]}|\\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2)}\Bigg(&\sum_{a<b}\sum_{f\in S_b^a}|\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|-|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|\\
&+|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[g,g]|\\
&+\rho\Bigg(\sum_{a<b}\sum_{f\in S_b^a}|\chi^{R_1}(h_{[0,b-1]})[,f]|\big(|\chi^{R_2}(h_{[b-1,b]})[,g]|-|\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|\big)\\
&\quad+|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|-|\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[g,g]|\\
&\quad+\sum_{a<b}\sum_{f\notin S_b^a}|\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[,g]|\Bigg)\Bigg)
\end{aligned}
$$

$$= \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2)} \Bigg(\Big(1-\rho\Big) \Bigg(\sum_{a<b} \sum_{f\in S_b^a} |\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|$$

$$- |\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[\neg g,g]| + |\chi^{R_1}(h_{[0,b-1]})[,g]||\chi^{R_2}(h_{[b-1,b]})[g,g]| \Bigg)$$

$$+ \rho \sum_{a<b} \sum_{f\in S_{b-1}^a} |\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[,g]| \Bigg)$$

Letting

$$RRSame = \sum_{R_2<|h_{[b-1,b]}|} \rho^{R_2} |\chi^{R_2}(h_{[b-1,b]})[g,g]|,$$

$$RRDiff = \sum_{R_2<|h_{[b-1,b]}|} \rho^{R_2} |\chi^{R_2}(h_{[b-1,b]})[\neg g,g]|$$

(And we note that $RRSame$ and $RRDiff$ do not actually depend on choice of g)

$$= \sum_{R_1<|h_{[0,b-1]}|} \rho^{R_1} \Bigg(\Big(1-\rho\Big) \Bigg(\sum_{a<b} \sum_{f\in S_b^a} |\chi^{R_1}(h_{[0,b-1]})[,f]| RRDiff$$

$$- |\chi^{R_1}(h_{[0,b-1]})[,g]| RRDiff + |\chi^{R_1}(h_{[0,b-1]})[,g]| RRSame \Bigg)$$

$$+ \sum_{R_2<|h_{[b-1,b]}|} \rho^{(R_2+1)} \sum_{a<b} \sum_{f\in S_{b-1}^a} |\chi^{R_1}(h_{[0,b-1]})[,f]| \underbrace{\binom{|h_{[b-1,b]}|-1}{R_2} (ht-1)^{R_2}}_{\text{by (7)}} \Bigg)$$

Noting that

$$\sum_{R_1<|h_{[0,b-1]}|} \rho^{R_1} |\chi^{R_1}(h_{[0,b-1]})[,f]| = R_{b-1}(f)$$

Letting:

$$S_1 := \sum_{a<b} \sum_{f\in S_b^a} R_{b-1}(f)$$

$$S_2 := \sum_{a<b} \sum_{f\notin S_b^a} R_{b-1}(f)$$

then the above is equal to

$$\Big(1-\rho\Big)\Big(S_1 RRDiff - R_b(g)\big(RRDiff - RRSame\big)\Big)$$

$$+ \big(S_1+S_2\big) \sum_{R_2<|h_{[b-1,b]}|} \rho^{(R_2+1)} \binom{\ell(b)-1}{R_2} (ht-1)^{R_2}$$

For $g \in S_b^b$, the calculation is similar:

$$
\begin{aligned}
|\chi(h_{[0,b]})[,g]| &= \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2+1)} |\chi^{R_1}(h_{[0,b-1]}) \oslash \chi^{R_2}(h_{[b-1,b]})[,g]| \\
&= \sum_{\substack{R_1<|h_{[0,b-1]}| \\ R_2<|h_{[b-1,b]}|}} \rho^{(R_1+R_2+1)} \Bigg(\sum_{f \in S_b^a} |\chi^{R_1}(h_{[0,b-1]})[,f]||\chi^{R_2}(h_{[b-1,b]})[,g]| \Bigg) \\
&= \big(S_1+S_2\big) \sum_{R_2<|h_{[b-1,b]}|} \rho^{(R_2+1)} \Bigg(\binom{|h_{[b-1,b]}|-1}{R_2} (ht-1)^{R_2} \Bigg)
\end{aligned}
$$

We can simplify the sums above by writing

$$
\begin{aligned}
RRS(ht,\ell) &:= \sum_{R_2<\ell} \rho^{R_2} \Bigg(\binom{\ell-1}{R_2} (ht-1)^{R_2} \Bigg) \\
&= \Bigg(1+(ht-1)\rho\Bigg)^{\ell-1}
\end{aligned} \tag{18}
$$

Given a second definition

$$
RRT(\ell) := (1-\rho)^{\ell-1} \tag{19}
$$

we can actually write

$$
\begin{aligned}
RRSame - RRDiff &= RRT(|h_{[b-1,b]}|) \\
RRDiff &= \frac{RRS(ht,|h_{[b-1,b]}|) - RRT(|h_{[b-1,b]}|)}{ht}
\end{aligned}
$$

and so finally, we can write our formula for $R_b(g)$ in a compact form as

$$
\boxed{R_b(g) = \begin{cases} (1-\rho)\Bigg(S_1 \frac{RRS(ht,|h_{[b-1,b]}|)-RRT(|h_{[b-1,b]}|)}{ht} + R_{b-1}(g) RRT(|h_{[b-1,b]}|)\Bigg) & \\ \qquad\qquad + \rho\big(S_1+S_2\big) RRS(ht,|h_{[b-1,b]}|) & \text{if } g \notin S_b^b \\ \rho\big(S_1+S_2\big) RRS(ht,|h_{[b-1,b]}|) & \text{if } g \in S_b^b \end{cases}}
$$