Complexity of Mutual Range Intersections across Hierarchical Partitions

Abstract

We analyze the asymptotic complexity of computing all mutual intersections among ranges derived from k hierarchically structured permutations. Each permutation of length n induces $\mathcal{O}(n)$ nested subranges organized as a perfect binary tree. We show that, under a cost model proportional to the smaller of the two range lengths, the total computational cost scales as $\Theta(k^2n^2)$.

1 Problem Formulation

Let $P_1, \ldots, P_k$ be k permutations over the index set $[n] = \{1, 2, \ldots, n\}$. For each permutation P_i, consider a perfect binary decomposition of its range:

$$T_i = \bigcup_{d=0}^{\log_2 n} \mathcal{R}_{i,d},$$

where $\mathcal{R}_{i,d}$ denotes the set of ranges at depth d. Each depth-d range contains exactly $S_d = n/2^d$ elements, and there are $N_d = 2^d$ such ranges.

Thus, for each i,

$$|\mathcal{R}_{i,d}| = 2^d, \qquad \text{and} \qquad \sum_{d=0}^{\log_2 n} |\mathcal{R}_{i,d}| = 2n - 1 = \Theta(n).$$

Let $\mathcal{R} = \bigcup_{i=1}^{k} T_i$ denote the set of all ranges across all trees, with $|\mathcal{R}| = \Theta(kn)$.

Intersection cost. For two ranges $R_a, R_b \in \mathcal{R}$, define

$$I(R_a, R_b) = |R_a \cap R_b|.$$

Assume that computing $I(R_a, R_b)$ incurs cost

$$\text{cost}(R_a, R_b) = \Theta(\min(|R_a|, |R_b|)).$$

We seek the total complexity

$$T = \sum_{R_a, R_b \in \mathcal{R}} \text{cost}(R_a, R_b) = \Theta\left(\sum_{R_a, R_b \in \mathcal{R}} \min(|R_a|, |R_b|) \right).$$

2 Complexity Analysis

2.1 Preliminary Observation

Since all trees share identical hierarchical structure, it suffices to analyze the interaction between two trees T_i and T_j and then multiply appropriately. We denote this pairwise cost by T_{pair} and later extend to k trees.

[Pairwise range cost by depth] For depths $d_1, d_2 \in \{0, \ldots, \log_2 n\}$, the total cost of comparing all ranges from levels d_1 and d_2 satisfies

$$C(d_1, d_2) = 2^{d_1+d_2} \cdot \frac{n}{2^{\max(d_1,d_2)}}.$$

Proof. At depth d_i, there are 2^{d_i} ranges, each of length $n/2^{d_i}$. For every pair (R_a, R_b) with depths d_1 and d_2, the cost is proportional to $\min(S_{d_1}, S_{d_2}) = n/2^{\max(d_1,d_2)}$. Since there are $2^{d_1+d_2}$ such pairs, the stated expression follows. □

2.2 Summation over all depth pairs

Without loss of generality, assume $d_1 \le d_2$. Then $\max(d_1, d_2) = d_2$, and

$$C(d_1, d_2) = n \cdot 2^{d_1}.$$

The total pairwise cost between two trees is

$$T_{\text{pair}} = 2 \sum_{d_2=0}^{\log_2 n} \sum_{d_1=0}^{d_2} C(d_1, d_2) \tag{1}$$

$$= 2n \sum_{d_2=0}^{\log_2 n} \sum_{d_1=0}^{d_2} 2^{d_1}. \tag{2}$$

The inner geometric sum yields

$$\sum_{d_1=0}^{d_2} 2^{d_1} = 2^{d_2+1} - 1,$$

so

$$T_{\text{pair}} = 2n \sum_{d_2=0}^{\log_2 n} (2^{d_2+1} - 1) = 4n \sum_{d_2=0}^{\log_2 n} 2^{d_2} - 2n(\log_2 n + 1).$$

Since $\sum_{d_2=0}^{\log_2 n} 2^{d_2} = 2n - 1$, it follows that

$$T_{\text{pair}} = 4n(2n - 1) - 2n \log_2 n = \Theta(n^2).$$

□

2.3 Extension to k trees

All trees share identical depth and structure, hence:

$$T = k\,T_{\text{intra}} + \binom{k}{2} T_{\text{pair}}.$$

Since $T_{\text{intra}} = \Theta(n^2)$ and $T_{\text{pair}} = \Theta(n^2)$,

$$T = \Theta(k^2 n^2).$$

2.4 Weight Computation Strategy with Inverse Index Representation

Context and Motivation. Given k permutations of length n, we consider the collection of all bipartitions generated from these permutations across k trees. Each bipartition is defined by two consecutive ranges (i, l_1, r_1, l_2, r_2), where $(A|B)$ corresponds to clusters $A = [l_1, r_1]$ and $B = [l_2, r_2]$, with $l_2 = r_1 + 1$. Let $\mathcal{CB}$ denote the set of candidate bipartitions, and $\mathcal{GB}$ the set of subtree bipartitions extracted from gene trees. Following [**?**], the weight of a candidate bipartition $x = (A_1|B_1) \in \mathcal{CB}$ with respect to the set of gene trees $\mathcal{G}$ is defined as

$$w_{\mathcal{G}}(x) = \sum_{y \in \mathcal{GB}} f_{\mathcal{G}}(y)\, \mathcal{M}(x, y), \tag{3}$$

where $f_{\mathcal{G}}(y)$ denotes the frequency of bipartition $y = (A_2|B_2)$ in $\mathcal{G}$, and $\mathcal{M}(x, y)$ is the number of triplets simultaneously mapped to both x and y:

$$\mathcal{M}(x, y) = \mathcal{NT}(|A_1 \cap A_2|, |B_1 \cap B_2|) + \mathcal{NT}(|A_1 \cap B_2|, |A_2 \cap B_1|), \tag{4}$$

where

$$\mathcal{NT}(n_1, n_2) = \frac{n_1 n_2 (n_1 + n_2 - 2)}{2}. \tag{5}$$

Limitations of Bitset-Based Implementation. Previous GPU implementations encoded each bipartition $(A|B)$ as two bitsets of length n, enabling efficient bitwise intersection via `__popcll` operations. This allowed $\mathcal{O}(n)$ time per $\mathcal{M}(x, y)$ evaluation but required $\mathcal{O}(n|\mathcal{GB}|)$ memory for storing bitsets of all gene tree bipartitions, which becomes prohibitive for large n or k. Furthermore, the memory bandwidth and transfer overhead dominate for large-scale datasets.

Inverse Index Representation. To address these limitations, we represent each permutation not as a dense bitset but by its *inverse index*:

$$\pi_i^{-1}(v) = \text{position of element } v \text{ in permutation } i.$$

Given k permutations $\pi_1, \pi_2, \ldots, \pi_k$, we store an $k \times n$ integer matrix P where $P_{i,v} = \pi_i^{-1}(v)$. This compact representation enables $\mathcal{O}(1)$ membership testing: an element v lies in range $[l, r]$ of permutation i if and only if $l \le P_{i,v} \le r$.

Intersection Counting via Inverse Index. For two bipartitions

$$x = (A_1|B_1) \quad \text{and} \quad y = (A_2|B_2),$$

we require four intersection counts:

$$|A_1 \cap A_2|, \quad |A_1 \cap B_2|, \quad |B_1 \cap A_2|, \quad |B_1 \cap B_2|.$$

Without loss of generality, assume that among two ranges A_p and A_q, the smaller one (in terms of cardinality) is A_p, i.e., $|A_p| \le |A_q|$. Let $A_p = \{\, v_i \mid i \in [l_p, r_p] \,\}$ denote the set of elements spanning indices $[l_p, r_p]$ in permutation π_p. Then, using the precomputed inverse index matrix P where $P_{i,v} = \pi_i^{-1}(v)$ gives the position of element v in permutation π_i, the intersection size between A_p and A_q can be computed as

$$|A_p \cap A_q| = \sum_{v \in A_p} \mathbf{1}_{[\, l_q \le P_{q,v} \le r_q \,]}, \tag{6}$$

where $\mathbf{1}_{[\cdot]}$ is the indicator function returning 1 if the condition inside holds, and 0 otherwise.

That is, for each element v in the smaller range A_p, we check whether its position in permutation q (given by $P_{q,v}$) falls within the interval $[l_q, r_q]$. Since each membership test is $\mathcal{O}(1)$ via the inverse index, the total complexity of computing $|A_p \cap A_q|$ is

$$\mathcal{O}(\min(|A_p|, |A_q|)).$$

The total time for computing all four intersection counts between two bipartitions is

$$\mathcal{O}\big(\min(|A_1|, |A_2|) + \min(|A_1|, |B_2|) + \min(|B_1|, |A_2|) + \min(|B_1|, |B_2|)\big),$$

which is typically much smaller than $\mathcal{O}(n)$ since bipartition clusters are localized contiguous segments in the permutations.

GPU-Parallel Weight Computation. This inverse-index formulation is inherently parallelizable across bipartition pairs. Each GPU thread can process a candidate bipartition x and iterate through all gene-tree bipartitions $y \in \mathcal{GB}$, computing $\mathcal{M}(x, y)$ using Equation (4). The procedure is as follows:

1. Precompute the inverse index matrix P ($k \times n$ integers).
2. For each bipartition $(A|B)$, store the boundary pairs (l_A, r_A) and (l_B, r_B).
3. On the GPU, assign each thread to a candidate $x = (A_1|B_1)$.
4. For each gene-tree bipartition $y = (A_2|B_2)$:
   (a) Compute the four intersection counts using inverse indexing:
   $$\begin{aligned} n_{AA} &= |A_1 \cap A_2|, \\ n_{AB} &= |A_1 \cap B_2|, \\ n_{BA} &= |B_1 \cap A_2|, \\ n_{BB} &= |B_1 \cap B_2|. \end{aligned}$$
   (b) Compute $\mathcal{M}(x, y)$ via Eq. (4).
   (c) Accumulate $f_{\mathcal{G}}(y)\mathcal{M}(x, y)$ into the total weight.

This scheme replaces the memory-heavy bitset representation with a lightweight positional representation, achieving comparable or better time complexity and superior memory efficiency. Each intersection computation leverages the contiguous range property of bipartitions, allowing early termination and minimal branching on GPU cores.

Complexity Summary. Let $m = |\mathcal{CB}|$ and $t = |\mathcal{GB}|$. Then the total cost of precomputing all weights becomes

$$\mathcal{O}\left(\sum_{x \in \mathcal{CB}} \sum_{y \in \mathcal{GB}} \big(\min(|A_1|, |A_2|) + \min(|A_1|, |B_2|) + \min(|B_1|, |A_2|) + \min(|B_1|, |B_2|)\big)\right),$$

which is sublinear in n per pair when clusters are small, and trivially parallelizable across GPU threads. This offers a practical and scalable replacement for bitset-based weight computation in large-scale gene tree analyses.

3 Conclusion

We have established the following result.

Let $P_1, \ldots, P_k$ be k permutations of length n, each decomposed into $\Theta(n)$ hierarchical ranges forming a perfect binary tree. Under the cost model $\mathrm{cost}(R_a, R_b) = \Theta(\min(|R_a|, |R_b|))$, the total computational cost of evaluating all pairwise range intersections satisfies

$$T = \Theta(k^2 n^2).$$

This analysis provides a theoretical upper bound for full mutual range intersection evaluation. In practice, the bound can be significantly reduced by exploiting disjointness and structural sparsity of ranges, or by distributing the pairwise computations in parallel across GPU threads.