A Novel Data Structure to Support Ultra-fast Taxonomic Classification of Metagenomic Sequences with k-mer Signatures Supplementary materials

1 Proof of Theorem 1

Bipartite graph $G = (U,V,E)$ satisfies $|U| = m_a$, $|V| = m_b$, $E \subset U \times V$. Each edge $(u_i, v_j) \in E$ represents a k-mer s, $h_a(s) = i$ and $h_b(s) = j$. Since h_a and h_b are uniform random hash functions, these edges can be considered as randomly and uniformly chosen from all possible edges in $U \times V$ with probability $\frac{|E|}{|U||V|} = \frac{n}{m_a m_b}$.

Consider a cycle in G, suppose the length of the cycle is $2t$. This cycle is equivalent to a list of $2t$ edges: (u_{i1}, v_{j1}), (v_{j1}, u_{i2}), (u_{i2}, v_{j2}), $\cdots$, (u_{it}, v_{jt}), (v_{jt}, u_{i1}). These $2t$ edges are uniquely decided by a list of $2t$ nodes $u_1, v_1, u_2, v_2, \cdots, u_t, v_t$. The number of such cycles that possibly exist in G is

$$\binom{m_a}{t}\binom{m_b}{t},$$

Here, $\binom{a}{b} = \frac{a!}{b!(a-b)!}$. And each of these cycles exists with probability $(\frac{n}{m_a m_b})^{2t}$

Apply the conclusion presented in (Botelho *et al.*, 2012) (Page 3), we know that when $\frac{n}{\sqrt{m_a m_b}} < 1 - O(\frac{1}{\sqrt{m_a m_b}})$, which is always satisfied because $m_a m_b \geq 1.33n^2$, the number of cycles with length $2t$ converges a Poission distribution with parameter λ_t, and

$$\lambda_t = (\frac{n}{m_a m_b})^{2t}\binom{m_a}{t}\binom{m_b}{t}$$

Let $c = \frac{n}{\sqrt{m_a m_b}}$, note that $t \ll m_a$,$t \ll m_b$, then we have

$$\lim_{t\to\infty} \frac{2t\lambda_t}{c^{2t}} = 1$$

Hence the total number of cycles in G converges a Possion distribution with parameter λ, where

$$\lambda = \sum_{t=1}^{\infty} \lambda_t = -\frac{1}{2}\ln(1-c^2)$$

2 Approaches of tuning p_t

$p(t)$ denotes the probability of an alien query returns t for an l-Othello. Once l-Othello is constructed, the $p(t)$ values can be accordingly computed. In Othello, there are two array of l-bit integers, namely A and B. We are able to modify the values in A and B without affecting any of the query results on the l-Othello. We describe two possible approaches as follows.

- Note that there are some elements in A and B, these elements do not correspond to any k-mers. Hence, we can assign any l-bit integer value to each of them, so that the occurrence frequency of each element are balanced.
- For any connected component of the bipartite graph G, we can execute a `XOR` operation on all of its elements in A and B. That is select any l-bit integer x and replace all $A[i]$ values in the connected component by $A[i] \oplus x$ and replace all $B[j]$ values by $B[j] \oplus x$ simultaneously. As long as for each connected components of G the corresponding values are all simultaneously replaced, this operation does not affect any $\tau(s)$ values.

In practice, we can always tune the values so that $p(t)$ is of the same order of magnitude for all t, and all of them are approximately 2^{-l}.

3 Proof of Theorem 2

We analyze the confidence of a K-mer window as follows. For a window of k-mers, let w be the length of the window. Suppose the query result for these K-mers are $\tau(s_1), \tau(s_2), \cdots, \tau(s_w)$. For a particular level of the taxonomy tree, suppose that these k-mer belongs to taxon t, then $\tau(s_1), \tau(s_2), \cdots, \tau(s_w) \in S_t$, where S_t is the set of the IDs of the nodes in the taxonomy subtree with the root t.

For consecutive w k-mers, let G_t be the event that this window of length w is from the taxon with ID t, without any sequence error. Let Q_t be the event that the query results of these k-mers belongs to S_t, namely $\tau(s_1), \tau(s_2), \cdots, \tau(s_w) \in S_t$.

For a particular window of k-mers, let w be the length of the window, (i.e., there are $k+w-1$ bases in this window.

Let G_t be the event that this window is actually from taxon t. We assume there is no sequencing error, hence, when G_t the query results for these w k-mers satisfy $\tau(s_1), \tau(s_2), \cdots, \tau(s_w) \in S_t$. We use notation Q_t to describe the event that $\tau(s_1), \tau(s_2), \cdots, \tau(s_w) \in S_t$.

Now the problem is that if we observe event Q_t, we may indicate two reasons exclusively. (1) Q_t happens as a result of G_t. (2) Note that for alien k-mers τ may return any integer, Q_t happens as a result of the query result of w alien k-mers. We use the

probability $P(G_t|Q_t)$ to describe how confident we are, about that this window is from taxon t.

As described, when G_t happens, Q_t also happens. Hence $P(Q_t|G_t) = 1$.

We estimate the value of $P(G_t|Q_t)$ as follow.

$$P(G_t|Q_t) = \frac{P(Q_t|G_t)P(G_t)}{P(Q_t|G_t)P(G_t) + P(Q_t|\overline{G_t})P(\overline{G_t})} = \frac{P(G_t)}{P(G_t) + P(Q_t|\overline{G_t})P(\overline{G_t})} \quad (1)$$

Let q_t be the abundance of the window from taxon t. i.e., for a particular sample, randomly select one window of length w among all windows in all reads from this sample, the probability that this window is actually from taxon t. Hence $P(G_t) = q_t$.

The value $P(Q_t|\overline{G_t})$ is estimated as follow.

$\overline{G_t}$ means that this window is not from taxon t. $\overline{G_t}$ indicates either one of the following sub-events: (1) C_{other}: In this particular level of taxonomy tree, the window is from one other taxon t', which means the query results $\tau(s_1), \tau(s_2), \cdots, \tau(s_w) \in S_{t'}$ for a $t' \neq t$. Note that $S_{t'} \cap S_t = \emptyset$, this indicates $P(Q_t|C_{\text{other}}) = 0$. (2) C_{alien}: This window is an alien of the taxonomy tree. Let $c_t = P(C_{\text{alien}}|\overline{G_t})$, then $0 < c_t < 1$.

$$P(Q_t|\overline{G_t}) = P(Q_t|C_{\text{other}})P(C_{\text{other}}|\overline{G_t}) + P(Q_t|C_{\text{alien}})P(C_{\text{alien}}|\overline{G_t}) = P(Q_t|C_{\text{alien}})c_t \quad (2)$$

As discussed in Section 2.2.3,

$$P(Q_t|C_{\text{alien}}) = q(t)^w \quad (3)$$

Combine Equation (1) (2) (3), we have

$$P(G_t|Q_t) = \frac{q_t}{q_t + p(t)^w c_t} \quad (4)$$

Note that, $q_t > 0$ and $0 < p(t) \ll 1$. Hence $P(G_t|Q_t) \to 1$ as $t \to \infty$. This is to say when w increases, $P(G_t|Q_t)$ also grows, and we can be more confident that when a query result shows that a window belongs to some taxon t, it reflects the fact that this window is actually from this taxon t. In other words, a longer window is more likely to come from this taxon than a shorter one.

Note that

$$\frac{q_t}{q_t + (p(t))^w c_t} > \frac{q_t}{q_t + (p(t))^w}$$

.

We use a threshold value λ, when $P(G_t|Q_t) > 1 - \lambda$, we accept, which is equivalent to:

$$w > \log_{p(t)} \frac{\lambda q_t}{(1-\lambda)} \sim \log_{p(t)} \lambda q_t$$

Here, the value of q_t can not be directly measured. However, for any actually detected taxon, we are sure that $q_t \geq \frac{1}{M}$, where M is the total number of reads in the dataset. Hence we use the following threshold to decide the length of accepted windows.

$$w > \log_{p(t)} \frac{\lambda}{(1-\lambda)M} \sim \log_{p(t)} \frac{\lambda}{M}$$

Note that we can always use the l-Othello to compute the value of $p(t)$. Thus, given λ ($\lambda = 0.001$ by default), for each taxon, we can pre-compute the minimum size threshold for K-mer window. Only the K-mer windows which are not shorter than its associated minimum window size will be accepted for final assignment determination.

4 Implementation of MetaOthello

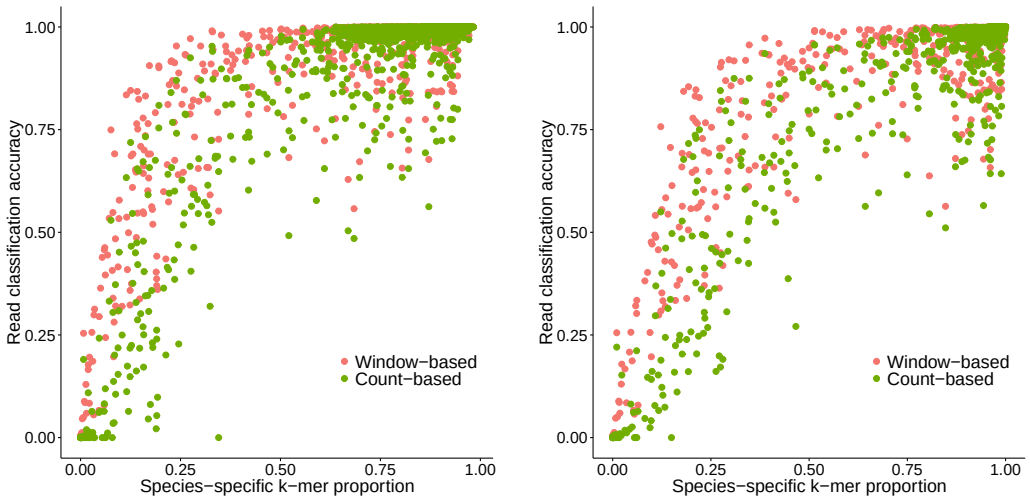
Jellyfish is used to collect all distinct k-mers from the designated reference genome database. For each K-mer, we counted its frequencies among all taxa at each taxonomic rank and also stores the first taxon it appears in. And a K-mer will be assigned to a taxon-specific K-mer set if it exists and only exists in that taxon for that taxonomic level, and its frequency is larger than 1 at the next level.

l-Othello will be built given the set of k-mers and their associated taxon IDs.

During the classification of the sequencing reads, l-Othello will be loaded into the memory first. Read will be classified one at a time sequentially. In the case of paired-end reads, information from both ends will be combined as one score when selecting the best assignment.

5 Results of count-based MetaOthello

To investigate how the window-based approach helps MetaOthello in sequencing read classification, we implemented and ran a count-based version MetaOthello on the sequencing datasets used in section 3.1 and 3.2. Figure 1 shows the correlation of species-specific k-mer signatures with classification accuracy for both window-based MetaOthello and count-based MetaOthello. Clearly, using both 20-mers and 31-mers, the window-based implementation exhibits higher accuracy. Table 1 presents the results (read assignment precision, sensitivity, and F-score) of count-based MetaOthello. Compared with the results of the default window-based MetaOthello in section 3.2, significant decreases on precision can be easily found for count-based MetaOthello.



(a) 20mer (b) 31mer

Figure 1: Correlation of species-specific k-mer proportion with classification accuracy for window-based MetaOthello and count-based MetaOthello when k=20 (A) and k=30 (B).

		Phylum	Genus	Species
		Prec / Sens / F-score	Prec / Sens / F-score	Prec / Sens / F-score
HiSeq	20mer	95.7 / 95.1 / .954	96.8 / 91.4 / .940	81.5 / 68.4 / .744
	25mer	94.8 / 93.7 / .943	97.8 / 89.9 / .937	83.3 / 68.1 / .749
	31mer	93.7 / 91.8 / .927	97.7 / 86.8 / .919	84.7 / 67.0 / .748
MiSeq	20mer	98.3 / 97.8 / .980	95.6 / 91.2 / .933	91.2 / 77.6 / .838
	25mer	98.1 / 95.3 / .967	96.3 / 90.3 / .932	92.0 / 77.3 / .840
	31mer	97.9 / 92.6 / .951	96.8 / 88.6 / .925	92.8 / 76.2 / .837
SimBA5	20mer	98.6 / 98.6 / .986	98.7 / 94.7 / .967	98.3 / 83.1 / .901
	25mer	97.5 / 97.4 / .976	98.7 / 93.0 / .958	98.6 / 81.7 / .893
	31mer	93.9 / 93.0 / .935	98.3 / 86.4 / .920	98.5 / 75.7 / .856

Table 1: Count-based MetaOthello read assignment precision, sensitivity, and F-score.

6 Results of Kaiju using the indices built on the two other options of source databases

In section 3.2, to conduct the comparative studies in a fair manner, we ran Kaiju using the same source database as the other three tools (MetaOthello, Kraken, and Clark). We notice that in Kaiju's manual (`https://github.com/bioinformatics-centre/kaiju/blob/master/README.md`), there two additional recommended databases (*nr* and proGenomes). To investigate how the choice of source database affects its performance, we further ran Kaiju using both of the two databases. As reported in Table 2, though some improvements are achieved (sensitivities at the phylum/genus level on HiSeq/MiSeq data), its performance is still far behind that of MetaOthello, Kraken, and Clark.

		Phylum	Genus	Species
		Prec / Sens / F-score	Prec / Sens / F-score	Prec / Sens / F-score
HiSeq	Kaiju nr	99.5 / 86.4 / .925	98.9 / 77.9 / .872	89.4 / 16.3 / .275
	Kaiju proGenomes	99.3 / 83.6 / .908	97.5 / 77.0 / .861	81.1 / 41.7 / .551
MiSeq	Kaiju nr	99.4 / 91.6 / .953	97.5 / 65.2 / .781	89.8 / 21.5 / .346
	Kaiju proGenomes	98.2 / 87.8 / .927	93.2 / 71.5 / .809	85.1 / 53.8 / .660
SimBA5	Kaiju nr	99.1 / 79.5 / .882	96.8 / 62.7 / .761	92.6 / 36.2 / .520
	Kaiju proGenomes	99.2 / 78.3 / .875	96.0 / 65.5 / .778	86.5 / 46.1 / .601

Table 2: Kaiju read assignment precision, sensitivity, and F-score using the indices built on the two other options of source databases.

References

Botelho, F. C., Wormald, N., and Ziviani, N. (2012). Cores of random r-partite hypergraphs. *Information Processing Letters*, **112**(8-9), 314–319.

Cunha, M. S., Esposito, D. L. A., *et al.* (2016). First Complete Genome Sequence of Zika Virus (Flaviviridae, Flavivirus) from an Autochthonous Transmission in Brazil. *Genome announcements*, **4**(2), 2015–2016.

EMBL-EBI webserver (2017). EMBL-EBI webserver. `https://www.ebi.ac.uk/Tools/sss/ncbiblast/nucleotide.html`. Accessed:2017-01-31.

Sardi, S. I., Somasekar, S., Naccache, S. N., *et al.* (2016). Coinfections of zika and chikungunya viruses in bahia, Brazil, identified by metagenomic next-generation sequencing. *Journal of Clinical Microbiology*, **54**(9), 2348–2353.

Wikipedia (2016). 2015-16 Zika virus epidemic. https://en.wikipedia.org/wiki/2015-16_Zika_virus_epidemic.