Title: A fast, scalable, MinHash-based k-mer tool to assess Sequence Read Archive next generation sequence submissions

Authors:

- Kenneth S. Katz, National Center for Biotechnology Information

National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA, kskatz@nih.gov

- Oleg Shutov, National Center for Biotechnology Information

National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA, shutovo@ncbi.nlm.nih.gov

- Richard Lapoint, National Center for Biotechnology Information

National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA, lapointr@ncbi.nlm.nih.gov

- Michael Kimelman, National Center for Biotechnology Information

National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA, kimelman@ncbi.nlm.nih.gov

- Christopher O'Sullivan, National Center for Biotechnology Information

National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA, osulliva@ncbi.nlm.nih.gov

- J. Rodney Brister, National Center for Biotechnology Information

National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA, jamesbr@ncbi.nlm.nih.gov

Corresponding Author: Kenneth S. Katz kskatz@nih.gov

Abstract

Sequence Read Archive submissions to the National Center for Biotechnology Information, often lack useful metadata, which limits the utility of these submissions.. We describe a scalable k-mer based tool for fast assessment of taxonomic diversity intrinsic to submissions, independent of metadata. We show our MinHash-based k-mer tool is accurate and scalable, offering reliable criteria for efficient selection of data for further analysis by the scientific community, at once validating submissions while also augmenting sample metadata with reliable, searchable, taxonomic terms.

Keywords: metagenomics, MinHash

Background

Established in 2007, the National Center for Biotechnology Information (NCBI) Sequence Read Archive (SRA) accepts raw sequencing data directly from high-throughput sequencing platforms (1). Next generation sequencing (NGS) sets are inherently large, and improved technologies are exquisitely sensitive to contamination. Submissions must be processed before either interpretation, or quality assessment, is possible to provide submitter feedback and submission verification. The growth of data submission is exponential (doubling approximately every 12 months (2)), rendering use of computationally expensive methods, such as *de novo* assembly followed by alignment, impractical due to costs and limits of scale, particularly given the time constraint of submission processing.

We considered that questions about the quality of a given NGS run could reasonably be inferred from the taxonomic distribution of reads within that set, whether based on a single organism or of metagenomic design. This is often enough information to answer basic experimental, or clinical, questions as well as inform decisions about the merit of subsequent resource-intensive assessment methods. Read sets with organismal tags can be used to select data for further analysis. Moreover, binning reads into taxonomic buckets can identify contaminating reads, and those outside of the stated experimental scope. Such identified reads can be filtered from a sample before downstream processing. This proposed taxonomic analysis is independent of metadata and intrinsic to the run,

capable of both validating submissions and augmenting sample metadata with reliable, searchable, taxonomic terms.

Following these principles, we developed a k-mer-based Sequence Taxonomic Analysis Tool (STAT). Based on MinHash (3), and inspired by Mash (4), STAT employs a reference k-mer database built from available sequenced organisms to allow mapping of query reads to the NCBI taxonomic hierarchy (5). We use the MinHash principle to compress the representative taxonomic sequences by orders of magnitude into a k-mer database, followed by a process that yields a set of diagnostic k-mer hashes for each organism. This allows for significant coverage of taxa with a minimal set of diagnostic k-mers. Our results show STAT is a reliable method for examining submitted NGS data in a timely, and scalable, manner.

Results

STAT was developed for quality assessment of SRA submissions to be shared with the submitter, requiring that analyses ideally take no more time than that of existing submission processing, while minimizing resource usage. Our design starts from the MinHash principle that a random selection of the lowest valued constituent blocks in a pool after hashing represents a signature of the parent object. In building k-mer databases from the set of sequences assigned a specific

NCBI taxonomy id (TaxId), we read the 32 base pair (bp) k-mers as 64-bit hashes, selecting the minimum value representative for a window, then iteratively merging k-mers from taxonomic leaves to roots (see Methods, Figures 1, 2).

Initial analysis using only densely populated k-mer databases performed well. However, despite being on average over an order of magnitude smaller than the input sequence database size (see below), we determined that loading the entire densely merged "tree_filter.dbs" into memory for analysis unnecessarily incurred long I/O read time, and large memory costs since most runs required only a fraction of the complete database. Moreover, STAT jobs, like many computational pipelines, are submitted to either a local computer farm cluster scheduler ("grid engine"), or by dispatching cloud-based virtual machines. In both cases job scheduling typically requires explicit needed resource declarations such as CPU and memory. An initial screen capable of evaluating diversity of the sample and necessary resource requirements for detailed analysis minimizes cost and maximizes computational efficiency. For these reasons we pursued a selective two-step analysis, using a sparse filtering database in the first step to identify the presence of any (a) eukaryote if there are more than 100 biological reads of a species, (b) bacteria, or archaea with more than 10 biological reads, and (c) virus if there are 1-2 biological reads. This first pass is neither qualitative, nor exhaustive, but allows us to quickly identify taxa for focus in the second pass (Figure 3).

To facilitate this two-step process, and further minimize resource requirements, we decreased k-mer database size by 33% by storing only the 8-byte k-mers in a database file, separately storing pairs of TaxId, total TaxId k-mer count for each TaxId respectively in an auxiliary "annotation" file. The k-mer database / k-mer count annotation file pair is designated "dbss", the database sorted by TaxId, with each TaxId set sorted by k-mer. TaxIds identified in the first step against the sparse k-mer database are used in the second step to load into memory only those TaxId k-mer hashes using the counts provided by the annotation file as offsets. MinHash sampling combined with dynamic loading of only necessary dense TaxId database k-mer hashes yields significant benefits for cpu and memory requirements. Further, the selection of TaxIds to load may be augmented by heuristics, such as purposely withholding TaxIds from contamination detected in the prior filtering step.

STAT reports the distribution of biological reads mapping to specific taxonomic nodes as a percentage of total biological reads mapped within the analyzed run. Since results are proportional to the size of sequenced genomes, a mixed sample containing several organisms at equal copy number is expected to find more reads originating from the larger genomes. This means that percentages reported likely reflect sample genome size(s), and must be considered by the user against the genomic complexity of the sequenced sample.

Like all sequence-based classification schemes, STAT reflects and depends upon accurately encapsulating evolutionary paths. The significant achievements of adapting both the NCBI RefSeq data model (6), and international taxonomy to incorporate metagenomic viral sequence (7,8) fundamentally benefit STAT and other similar classification tools.

An important consequence of merging in k-mer database construction is to avert complications caused by biological complexities. For example, most k-mers derived from endogenous retroviruses found in the human input reference genome will likely merge to the root as those k-mers would also be found in the Viruses Super Kingdom.

Further, when analyzing results each level – read, run – requires integration of less than ideal signals. It is common to find multiple TaxIds identified in a single biological read, ideally coherent for a given lineage. Were those Mus musculus, Murinae, Mammal, there is confidence in declaring the read Mus musculus. Should a read map to multiple, related taxonomic nodes, it is reported as originating from the most proximal shared taxonomic node. For example, a read with hits to sibling species may be reported as their common genus, conservatively locating the most proximal common node before ambiguity (Figure 4). Likewise, such conservative heuristics are required when integrating the signals from all biological reads to report the run. If the run subject is a single organism, it is

expected that STAT would identify taxonomic nodes across the lineage, and that the number of reads mapping to higher level nodes will be more than those mapping to terminal nodes.

STAT was designed as a tool for assessing the quality of *any* SRA submission and has matured into a tool that also significantly enhances user comprehension. Many k-mer tools were created for the purpose of metagenomic taxonomic assignment (9) during STAT development. Taxonomic classifiers balance speed, accuracy, and memory requirements. While STAT was neither primarily developed for metagenomic analyses, nor as a tool for distribution, the same concerns apply. Using MinHash to sample and save at most 1 out of every 64 k-mers generated from input sequences yields k-mer databases 1-2 orders of magnitude smaller than the parental reference nucleotide database from which they were derived. For example, currently the Blast refseq_genomes database used is 1.4 terabytes (tb) whereas the representative sparse and dense STAT k-mer databases are approximately 1.5 gigabytes (gb), and 75 gb, respectively.

The STAT k-mer databases contain 248,426 TaxIds before merging. Our complete merged 75 gb dense database ("tree_filter.dbss") represents 130,817 TaxIds after merging (all data reflect the 20200518 build). Compare the Kraken default 70 gb database that only includes "RefSeq complete genomes, of which there are 2,256, while Kraken-GB contains 8,517 genomes" (10). Despite our

sparse index database (“tree_index.dbs”) size of 1.5 gb, it nonetheless contains k-mers from 119,982 TaxIds.

We compare STAT accuracy to Kraken 2 using the strain exclusion test as described by Wood et al. (11). STAT shows the identical accuracy of Kraken 2 for both bacteria and virus (see Figure 5). As expected, STAT sensitivity is notably dampened as we chose to sample the widest taxonomic breadth. Our desire for conservative taxonomic assertions is further reflected by STAT never yielding a false positive in accuracy test results (data not shown).

We found it unnecessary to apply the same selection of k-mer hash minimums from query sequences to compose a similarity index (3,4), instead of exact k-mer (hash) matching. We show that accuracy is robust, while still reflecting our conservative bias in taxonomic assignment. Though similar in performance to Kraken 1 input speed (21.6 million reads/minute), and runtime (132.5 seconds) characteristics, STAT (maximum resident set size 830304 kilobytes) required only 8% and 1% of the memory needed by Kraken 1 and Kraken 2, respectively (11). Unsurprisingly, the accuracy test (see Methods) required additional time for extracting the requested TaxId k-mers on demand. Maximum resident set size during the accuracy test was approximately an order of magnitude greater than Kraken 2 (11, data not shown), despite loading a k-mer database 20 times the

“strain_excluded” FASTA file size (3.9 gb) and over 100 times “strain_excluded.dbs” size (545 megabytes (mb)).

Conclusions

STAT has provided a successful framework for our SRA NGS submission pipeline. Sometimes actual sample content may be unknown, and submitted metadata are often incomplete and of poor quality (12,13). Recognizing these limitations stimulated our foremost goal to derive signals able to validate, and accurately describe submitted data for the benefit of our users. Reflecting the National Institutes of Health (NIH) Science and Technology Research Infrastructure for Discovery, Experimentation, and Sustainability (STRIDES) Initiative (14), and ensuring that NIH-funded research data is findable, accessible, interoperable, and reusable (FAIR) (15), results from STAT are available through Amazon Web Services’ Athena and Google Cloud Platform’s BigQuery query services. Both can be searched to identify runs containing specific organismal content (16), allowing efficient selection of data for further analysis by the scientific community. Over approximately five years, we have processed more than 27.9 Peta base pairs from runs averaging 1.1 Giga base pairs in size with average total processing throughput of 3 minutes per run. While roughly 20% of runs analyzed to date are withheld by submitter request until ready for publication,

nearly 10.8 million are publicly-queryable records, now richly annotated by STAT analysis.

Building a STAT database is flexible; it can be tailored to specific needs. For example, we are currently testing a STAT k-mer database designed to identify Antimicrobial Resistance (AMR) in NGS. The AMR_CDS FASTA file containing sequences curated by the NCBI Pathogen group (17) is used as input to generate 32 bp k-mers with a window size = 1; that is, the complete non-redundant k-mer set. For the purpose of removing human reads from clinical pathogen screening samples, we are testing a tool combining STAT *aligns_to* with a human-specific database built using a window size = 32 bp (twice as many minimal k-mers selected compared to our standard 64 bp "dbss" window). As part of recent NIH-wide efforts to combat Sars-Cov-2, we released a detection tool containing *aligns_to* and a Virus "dbs" that allows users to map k-mers found in NGS data to taxa included under Coronaviridae (18).

Our choice to maximize taxonomic coverage while minimizing k-mer count has proved a reasonable, and effective balance. Employing the principle of MinHash in design, we contribute a framework others may find useful, and offer the collection of tools to use freely.

The success we and others have experienced is consistent with the notion of a random model of k-mer occurrence (19). Yet, as keenly shown by Breitwieser et al. (20) *unique* k-mer hits are the most informative. Through serendipity while preparing this manuscript, our colleague John Spouge enlightened us with his method of a non-parametric statistical approach to assess an NGS run using unique hits for confident measurement of taxonomic assignments (21). We are just beginning to explore this implementation in STAT, and look forward to reporting results in the future.

Methods

General Design:

STAT refers to a collection of tools for building k-mer databases, querying those databases, and reporting results of our SRA submission pipeline using the former. Details described below are based on our standard pipeline settings.

k-mer Size:

STAT uses 32 bp k-mers (i.e., k=32) for database generation, and as the unit for comparison. The majority of unaligned SRA data are reads between 60 and 150 bp in length, with systematic error rate of 0.1% (22): such reads can be expected to yield many correct 32 bp k-mers for reliable identification. While reducing from 32 bp k-mers to 16 bp k-mers decreases the size of resulting databases, there is

significant loss of specificity (10 ^9) per k-mer. By comparison, using 64 bp k-mers are extraordinarily more selective, but database size becomes impractical. Finally, with each base encoded in 2 bits, 32 bp k-mers fit fully, and compactly in a 64-bit integer.

k-mer Databases:

Two primary types of k-mer databases are constructed (as described below): a dense database that selects one k-mer per 64 bp segment (“tree_filter”), of input sequence, while a sparse database (“tree_index”) selects one k-mer per 64 bp (Virus), 8000 bp (Eukaryota), and 2000 bp (Bacteria, and Archaea) segment respectfully, noting that segment size is roughly proportional to genome size.

k-mer Generation:

k-mers are selected using an iterative approach derived from MinHash (3) . To compose STAT databases, for every fixed length segment (“window”) of incoming nucleotide sequence, a list of overlapping k-mers (effectively segment length plus right k-1 bp “wings”) is generated from both strands. The 32 bp k-mers are encoded using 2 bits per base into 64 bits (8 bytes), then a minimum k-mer representing this segment is selected based on the 64-bit encoded k-mer value read as 64-bit integer, effectively a 64-bit hash (see Figure 1) .

Taxonomic k-mer Database generation:

Construction of k-mer databases is guided by the NCBI Taxonomy Database (5), specifically the four root Super Kingdoms: Archaea (722 species, 1330 total nodes), Bacteria (20,259 species, 29,835 total nodes), Eukaryota (455,421 species, 638,336 total nodes), and Viruses (4656 species, 7,583 total nodes) [current as of manuscript date (23)].

From each (Super Kingdom) root, lineage paths traverse nodes where terminal nodes are those containing only child leaves. Input sequences (see below) have an assigned NCBI Taxonomy Id (TaxId), and represent leaves on these trees. These lineage relationships are represented in a two-column file referred to as "parents", wherein each node TaxId (first column) reports its parent node TaxId (second column).

All sequences (see Database Input Sequences) attached to a particular TaxId are input to k-mer database generation using segment ("window") sizes as described. For each input set of sequences assigned a TaxId, the immediate output is a dictionary that contains the set of unique 32 bp k-mers derived as described (we designate this "db" file extension). Each dictionary is further transformed into a binary file that encodes every 32 bp k-mer as an 8-byte (64-bit) integer using 2 bits per base, followed by its TaxId represented in a 4-byte (32-bit) integer. Thus each k-mer record is stored as one 12-byte pair (k-mer, TaxId) in a database file

designated with “dbs” file extension, sorted by k-mer for binary search optimization.

Next, using the taxonomic node relationships (found in the “parents” file), starting from the leaves we recursively merge each binary (“dbs”) file representing a unique set of k-mers derived from a single TaxId to sibling(s), then parent nodes. Each sibling leaf is merged such that k-mers specific to a leaf remain as diagnostic of that TaxId, while those found in neighboring (sibling species) leaves are moved up (“merged”) to the common parent node TaxId (see Figure 2). This process results in a single merged database file (“tree_filter.dbs”) representing all k-mers assigned a TaxId.

While it is difficult to generalize, we note that when the process of merging is complete, approximately 20% of the *Homo sapiens* 32 bp k-mers remain as unique to human; that is, 80% were not diagnostic for the species, and instead merged up the eukaryotic tree.

Database generation can be accomplished using any of the *build_index** tools (see github), and each takes parameters for window size, and k-mer size. The process of merging is accomplished using *merge_db*.

Database Input Sequences:

We use NCBI Blast "refseq_genomes" database, supplemented with viral sequences extracted from the NCBI Blast "nt/nr" database as input source for taxonomy identification in both sparse ("index") and dense ("filter") k-mer databases. Viral records are extracted from "nt/nr" by loading only sequences assigned a TaxId whose lineage root is the Super Kingdom "Viruses".

Querying the Taxonomic k-mer Database (STAT):

To query a k-mer database, input SRA accessions, or FASTA sequence is used to generate the unique set of query 32 bp k-mers read as 64-bit integer hashes for finding identical hashes (and assigned TaxId) from the designated k-mer database using the tool *aligns_to*. Proximal results are counts for each specific taxonomic k-mer hit (see Results, Figure 3). Passed an SRA accession, STAT built with NCBI NGS library support will retrieve query sequences, and *aligns_to* option *-unaligned_only* is available to limit analysis to the unaligned reads found in the SRA object.

Database Filtering:

We determined the need to delete low complexity k-mers composed of >50% homo-polymer or dinucleotide repeats (e.g. AAAAAA or ACACACACACA). This is accomplished using *filter_db.* We have also investigated "dusting" input sequences (24) and found it complementary to filtering, though it is not used at this time in our pipeline.

Performance measurement:

STAT performance metrics were gathered as described in Wood et al. (see "Execution of strain exclusion experiments", and "Evaluation of accuracy in strain exclusion experiments" in Methods, 11). A "dense" k-mer database was created using the excluded taxa sequences for input (11). Briefly, we used Mason 2 (25) to generate 500,000 simulated Illumina 100 bp paired reads for each excluded strain TaxId, and collected cpu, and memory using ram-disk storage of the simulated reads employing 16 threads (16 Intel® Xeon® 2.8 GHz CPUs 64 GB RAM). Accuracy was measured using *aligns_to* against "tree_filter.dbss" (see Results) with a list of all TaxIds excluding the 50 strains tested (130,769 TaxIds total).

References

1. Shumway M, Cochrane G, Sugawara H. Archiving next generation sequencing data. Nucleic Acids Res. 2010 Jan;38 Database issue:D870-1.

2. Kodama Y, Shumway M, Leinonen R; International Nucleotide Sequence Database Collaboration. The Sequence Read Archive: explosive growth of sequencing data. Nucleic Acids Res. 2012 Jan;40 Database issue:D54-6.
3. Broder AZ. Identifying and filtering near-duplicate documents. In: COM ’00 Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching. London: Springer; 2000. p.1–10
4. Ondov, B.D., Treangen, T.J., Melsted, P. et al. Mash: fast genome and metagenome distance estimation using MinHash. Genome Biol. 2016;17:132. Available from: https://doi.org/10.1186/s13059-016-0997-x
5. NCBI Taxonomy Browser [internet].Taxonomy [Internet]. Available from: https://www.ncbi.nlm.nih.gov/taxonomy/
6. Brister JR, Ako-Adjei D, Bao Y, Blinkova O. NCBI viral genomes resource. Nucleic Acids Res. 2015 Jan;43(Database issue):D571-7.
7. Simmonds P, Adams MJ, Benkő M, Breitbart M, Brister JR, Carstens EB, et al. Consensus statement: Virus taxonomy in the age of metagenomics. Nat Rev Microbiol. 2017 Mar;15(3):161-168.
8. A sea change for virology. Nat Rev Microbiol. 2017 Feb 13;15(3):129.
9. Breitwieser FP, Lu J, Salzberg SL. A review of methods and databases for metagenomic classification and assembly. Brief Bioinform. 2019;20:1125–1136.
10. Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biol. 2014;15:R46.

11. Wood DE, Lu J, Langmead B. Improved metagenomic analysis with Kraken 2. Genome Biol. 2019 Nov 28;20(1):257.
12. Bernstein MN, Doan A, Dewey CN. MetaSRA: normalized human sample-specific metadata for the Sequence Read Archive. Bioinformatics. 2017 Sep 15;33(18):2914-2923.
13. Bernstein MN, Gladstein A, Latt KZ, Clough E, Busby B, Dillman A. Jupyter notebook-based tools for building structured datasets from the Sequence Read Archive. F1000Res. 2020 May 19;9:376. Available from: https://doi.org/10.12688/f1000research.23180.2.
14. NIH Office of Data Science Strategy [internet]. STRIDES. Available from: https://datascience.nih.gov/strides
15. Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, et al.The FAIR Guiding Principles for scientific data management and stewardship. Sci Data. 2016 Mar 15;3:160018.
16. NCBI Sequence Read Archive (SRA) [internet]. SRA in the cloud. Available from: https://www.ncbi.nlm.nih.gov/sra/docs/sra-cloud-based-examples/
17. NCBI National database of antibiotic resistant organisms (NDARO). AMR cds fasta. Available from: https://ftp.ncbi.nlm.nih.gov/pathogen/Antimicrobial_resistance/AMRFinderPlus/data/latest/AMR_CDS

18. NCBI Sequence Read Archive (SRA) [internet]. SRA detection tool. Available from: https://www.ncbi.nlm.nih.gov/sra/docs/sra-detection-tool
19. Fofanov Y, Luo Y, Katili C, Wang J, Belosludtsev Y, Powdrill T, Belapurkar C, Fofanov V, Li TB, Chumakov S, Pettitt BM. How independent are the appearances of n-mers in different genomes? Bioinformatics. 2004 Oct 12;20(15):2421-8.
20. Breitwieser FP, Baker DN, Salzberg SL. KrakenUniq: confident and fast metagenomics classification using unique k-mer counts. Genome Biol. 2018;19:198.
21. Spouge J, personal communication.
22. Pfeiffer F, Gröber C, Blank M, Händler K, Beyer M, Schultze JL, Mayer G. Systematic evaluation of error rates and causes in short samples in next-generation sequencing. Sci Rep. 2018 Jul 19;8(1):10950.
23. NCBI Taxonomy Browser.Taxonomy Statistics [Internet]. Taxonomy Nodes (all dates). Available from: https://www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html/index.cgi?chapter=statistics&uncultured=hide&unspecified=hide
24. Morgulis A, Gertz EM, Schäffer AA, Agarwala R. A fast and symmetric DUST implementation to mask low-complexity DNA sequences. J Comput Biol. 2006 Jun;13(5):1028-40.
25. Holtgrewe, M. Mason - a read simulator for second generation sequencing data. Technical Report 2010 TR–B–10–06.

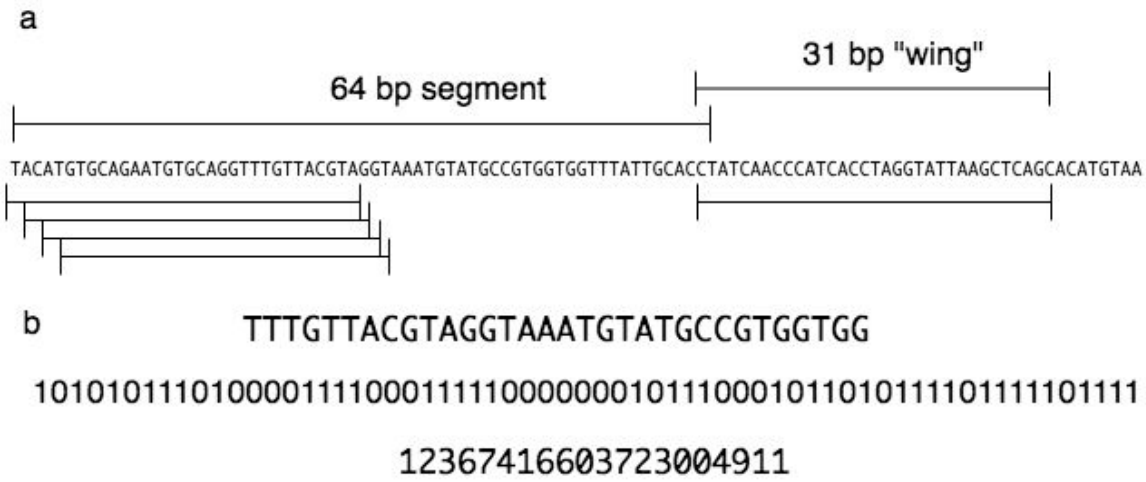

Figure 1. *Finding a minimum representative 32 base pair k-mer.* **a** From a selected 64 base pair segment, the series of 64 possible 32 base-pairs k-mers is defined by sequentially shifting the 32-base window by one base. The first four, and last of the possible k-mers is shown schematically. **b** An example selected k-mer sequence is shown followed by: first the two-bit encoding of that same k-mer sequence; finally the 64-bit decimal value of the encoded k-mer. The lowest 64-bit decimal value is selected as the representative k-mer for this 64 base segment.

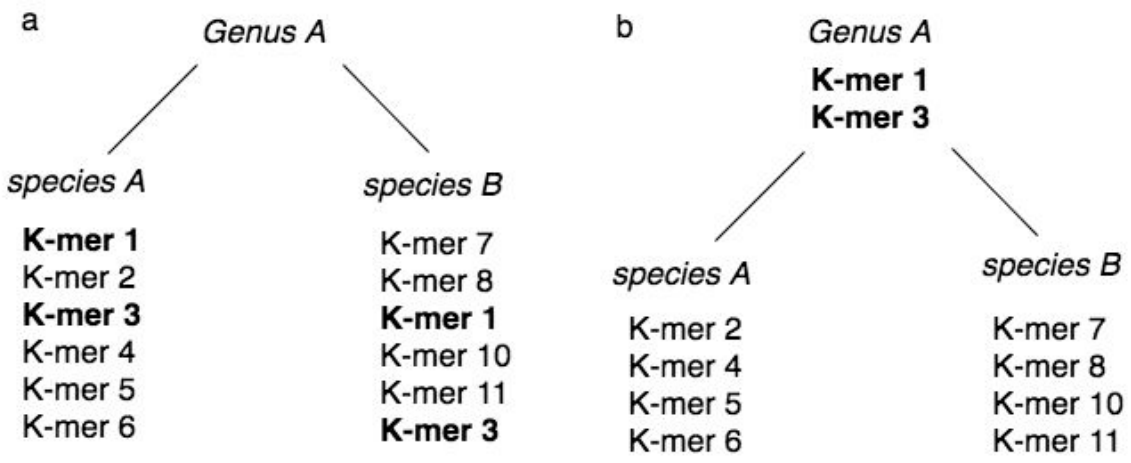

Figure 2. *K-mer taxonomic merging*. See Methods for details. **a** Before merging two sibling species are depicted containing both unique and shared k-mers (indicated in bold).. **b** After merging the two shared k-mers "merge" up to the *genus* level, while those unique to each remain diagnostic for the *species*.

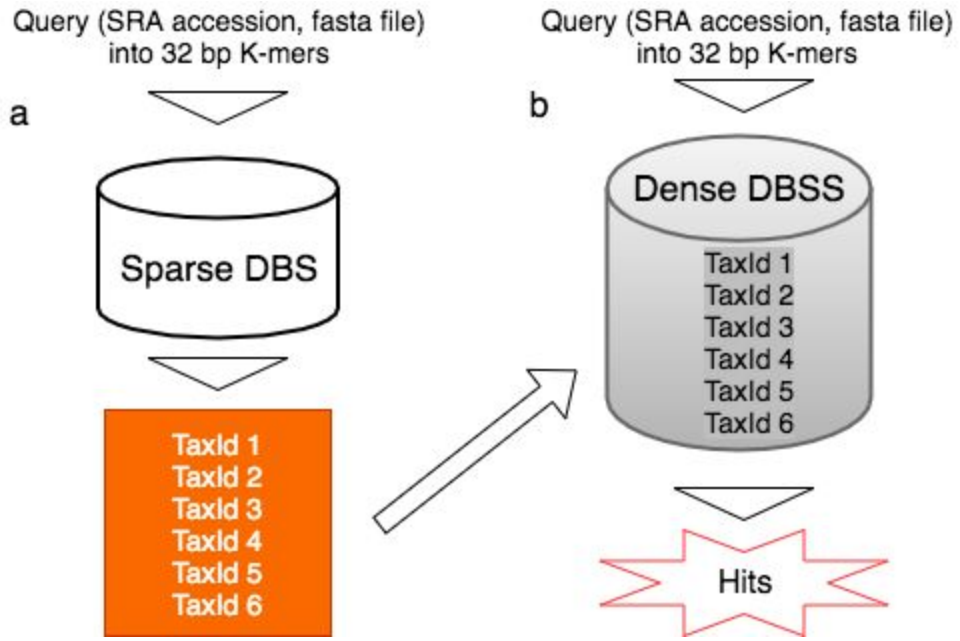

Figure 3. *STAT two phase query*. **a** In the first qualitative phase the input query (an SRA accession, or fasta file) is sequentially rendered into 32 bp k-mers, and matches to the decimal hash values found in the sparse database identifying taxa for deeper analysis. **b** The TaxIds identified are used to compose the dense database using the same query in the second quantitative pass.

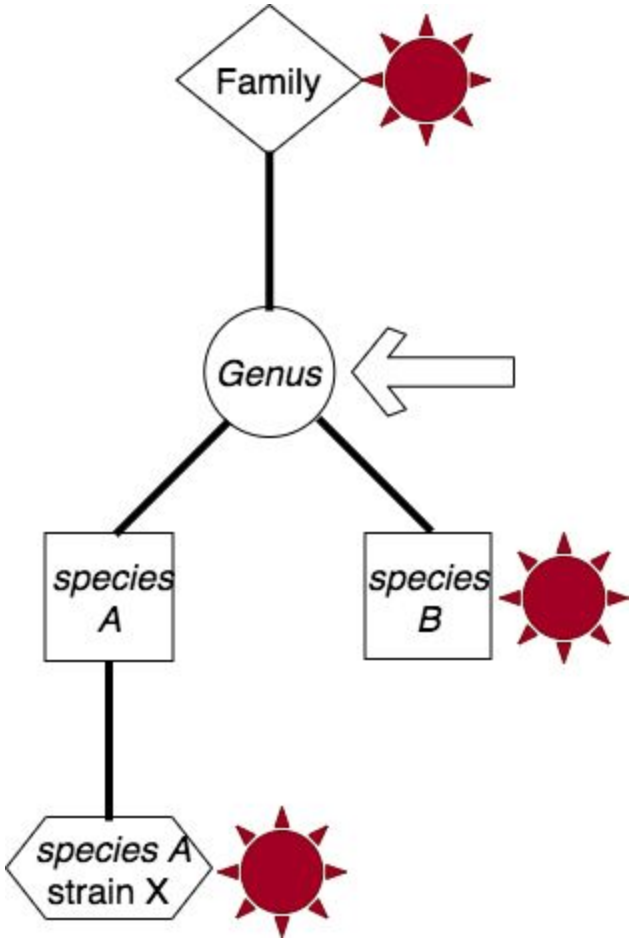

Figure 4. *Resolution of taxonomic assignment.* Depicted are three distinct k-mer hits within a single read to multiple branch taxa. The arrow indicates conservatively the most proximal unambiguous common node.

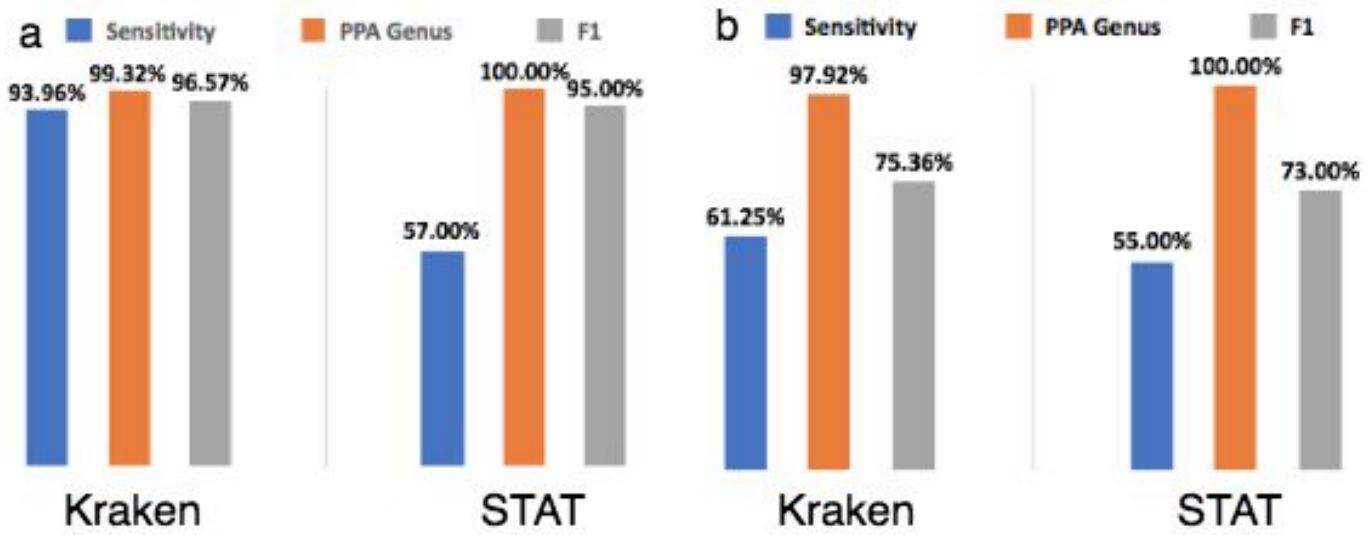

Figure 5. *STAT Accuracy and Sensitivity.* Comparison of STAT and *Kraken2* accuracy, sensitivity, and F1 measure using positive predictive value at the *genus* level for **a** Bacteria. **b** Virus.

Declarations

- *Ethics approval and consent to participate*
  - Not applicable
- *Consent for publication*
  - Not applicable
- *Availability of data and materials*
  - https://github.com/ncbi/ngs-tools/tree/tax/tools/tax/src
- *Competing interests*
  - Not applicable
- *Funding*
  - This work is supported by the Intramural Research Program of the National Library of Medicine, National Institutes of Health.
- *Authors' contributions*
  - KSK prepared the manuscript including figures, and executed performance and accuracy analyses
  - OS authored most of the C++ / python STAT tools
  - JRB, CO contributed to discussions of design, and implementation
  - RL tested early versions of the tools

- *Acknowledgements*

- Vadim Zalunin, Alex Efremov, and Andrey Kochergin for STAT pipeline and post-processing support.
- Ryan Connor for stimulating conversation.
- David Lipman for ideas, support, and vigorous discussion.
- Benjamin Langmead for supplying strain exclusion fasta files.
- Susan Roberts and Lydia Fleischmann for excellent editing assistance.