GigaScience, 2017, 1–9

doi: xx.xxxx/xxxx
Manuscript in Preparation
Technical note

TECHNICAL NOTE

Smash++: finding rearrangements

Morteza Hosseini[1,*], Diogo Pratas[2,*] and Armando J. Pinho[2,*]

[1,2]IEETA/DETI, University of Aveiro, Portugal

[*]seyedmorteza@ua.pt; pratas@ua.pt; ap@ua.pt

Abstract

The Abstract (250 words maximum) should be structured to include the following details: **Background**, the context and purpose of the study; **Results**, the main findings; **Conclusions**, brief summary and potential implications. Please minimize the use of abbreviations and do not cite references in the abstract.

Key words: Keyword1; keyword 2; keyword 3 (Three to ten keywords representing the main content of the article)

Introduction to this Template

This is the epigraph text, should you like to add one. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

—Epigraph source name

This is the LaTeX template for GigaScience journal manuscript submissions. **Please note that whilst this template provides a preview of the typeset manuscript for submission, it will not necessarily be the final publication layout.**

There are important commands in the preamble that you will need to modify for your own manuscript. If you are using this template on Overleaf, please switch the editor to Source code mode to view them; or if you prefer to stay in the Rich Text view, click on the title in the Rich Text view to display the preamble code.

Use the `\journal{...}` command in the preamble so that the correct journal name, logo and colours are loaded automatically. **Only certain journals and options are supported at this time;** check with your journal's editorial office if your journal is supported.

Alternatively you can re-define `\jname`, `\jlogo` and the `jcolour` explicitly, though check with your journal's editorial office to confirm that this is appropriate.

Specify your manuscript's category with the `\papercat{...}` command in the preamble.

See the sample code in the preamble for a sample of how author and affiliation information can be specified.

Use later sections starting with 'Background' on page 2 to write your manuscript. The remainder of this current section will provide some sample LaTeX code for various elements you may want to include in your manuscript.

Sectional Headings

You can use `\section{...}`, `\subsection{...}` commands to add more sections and subsections to your manuscript. Further sectional levels are provided by `\subsubsection`, `\paragraph` and `\subparagraph`.

Citations and References

Use the `num-refs` document class option for numerical citations, and `alph-refs` option for author-year citations. Use the `\citep` command for parenthetical citations, and `\citet` command for text citations (when using `alpha-refs`). This is a citation: [1] and here are two more: [2, 3].

- This is a bullet list.
- Another point.
- A third point.

This[1] is a footnote. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et

1 This is the footnote text. This is the footnote text. This is the footnote text. This is the footnote text. This is the footnote text.

Compiled on: May 28, 2019.
Draft manuscript prepared by the author.

Key Points

- This is the first point
- This is the second point
- One last point.

dolore magna aliqua.

- This is a numbered list.
- Another point.
- A third point.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

This is a 3rd level heading

Use `\subsubsection` to get a 3rd level heading. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

This is a 4th level heading. Use `\paragraph` to get a 4th level heading. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

THIS IS A 5TH LEVEL HEADING. Use `\subparagraph` to get a 5th level heading. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Figures and Tables

Figures and tables can be added with the usual `figure` and `table` environments, e.g. Figure ?? and Table ??. Use `figure*` and `table*` if you need a two-column wide figure or table, as in Figure ?? and Table ??.

If you have a very wide table or figure, you can use `sidewaystable` or `sidewaysfigure`, as in Table ??: this will be rotated sideways and occupy a *single column* on its own.

If your table or figure is both wide and tall (so it wouldn't fit well in a single column with `sidewaystable` or `figure`), you can use `table` or `figure` inside a `landscape` environment for a full-page landscaped alternative. A page break will be inserted *immediately before and after* the `landscape` environment (Table ??), so you'll need to carefully position it in a suitable location in your manuscript.

Some Mathematics Sample

Let $X_1, X_2, \ldots, X_n$ be a sequence of independent and identically distributed random variables with $\text{E}[X_i] = \mu$ and $\text{Var}[X_i] = \sigma^2 < \infty$, and let

$$S_n = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n}\sum_{i}^{n} X_i \tag{1}$$

denote their mean. Then as n approaches infinity, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $\mathcal{N}(0, \sigma^2)$.

Background

The background section should be written in a way that is accessible to researchers without specialist knowledge in that area and must clearly state—and, if helpful, illustrate—the background to the research and its aims. The section should end with a brief statement of what is being reported in the article.

Data Description

A statement providing background and purpose for collection of these data should be presented for readers without specialist knowledge in that area. A brief description of the protocol for data collection, data curation and quality control, as well as potential uses should be included, as well as outlining how the data can be accessed if it is not deposited in our repository.

Analyses

This section should provide details of all of the experiments and analyses that are required to support the conclusions of the paper. The authors should make clear the goal of each analysis and state the basic findings.

Discussion

The discussion should spell out the major conclusions and interpretations of the work, including some explanation on the importance and relevance of the dataset and analysis. It should not be restatement of the analyses done and their basic conclusions. The discussion section can end with a concluding paragraph that clearly states the main conclusions of the research along with directions for future work. Summary illustrations can be included.

Potential implications

Authors should provide some additional comments about potential, more broad-ranging implications of their work that are not directly related to the current focus of their manuscript. This section is meant to promote discussion on possible ways the findings or data presented might be used in or have a relationship with other areas of research that may not be directly apparent in the work. It is not meant to provide 'proof of importance' of the work. Only to engender expansion of use to other areas.

Explicit personal opinions by the authors are permitted, but they should be made clear as such. References or related information to support the propositions should be included. These section should focus on work that can be done within the foreseeable future and specifically using the information within the

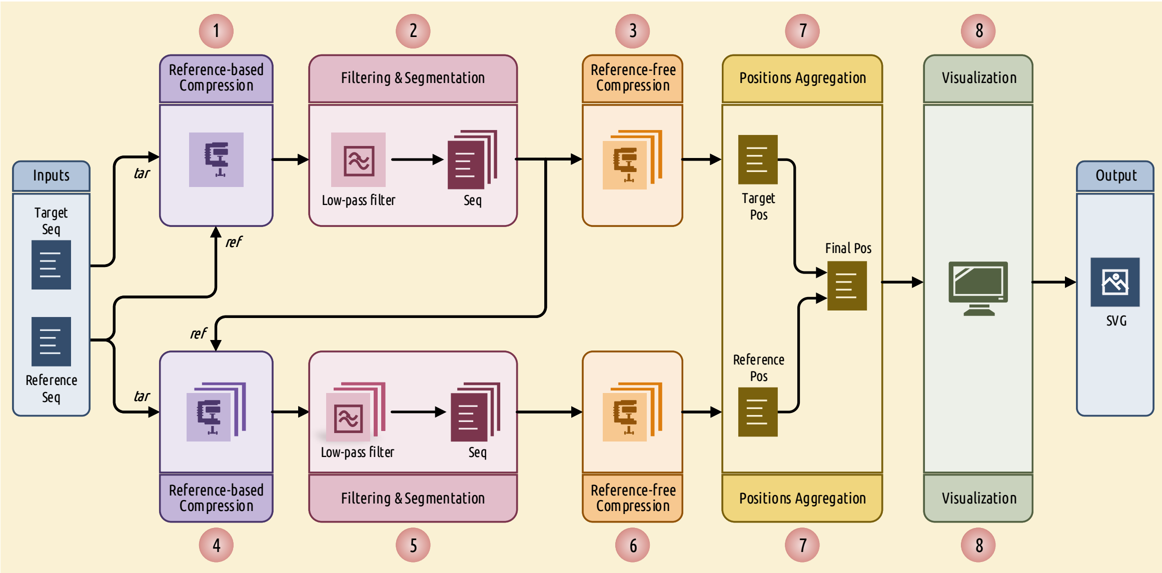

Figure 1. The schema of Smash++. The process of finding similar regions in reference and target sequences and also, computing redundancy in each region includes eight stages. Finally, Smash++ outputs a *.pos file that includes the positions of the similar regions, and can be then visualized, resulting in an SVG image.

manuscript, not provide speculation on how it will relate to far-reaching goals of the research area.

Methods

The schema of the proposed method is illustrated in Figure 1. Smash++ takes as inputs a reference and a target file and produces as output a position file, which is then fed to the Smash++ visualizer to produce an SVG image. This process has eight major stages: (1) compression of the original target file, based on the model of original reference file, (2) filtering and segmentation of the compressed file, (3) reference-free compression of the segmented files, obtained by the previous stage, (4) compression of the original reference file, based on the model of segmented files obtained by stage 2, (5) filtering and segmentation of the compressed files, (6) reference-free compression of the segmented files, that are obtained by the stage 5, (7) aggregating positions, generated by stages 3 and 6, and (8) visualizing the positions. The following sections describe the process in detail.

Data modeling

Smash++ works on the basis of cooperation between finite-context models (FCMs) and substitutional tolerant Markov models (STMMs). Applying these models on various contexts provides probability and weight values, illustrated in Figure 2a, which are then mixed (by multiplication and addition, shown in Figure 2b) to provide the final probability (P) of occurring an input symbol. The following subsections describe FCMs and STMMs in detail.

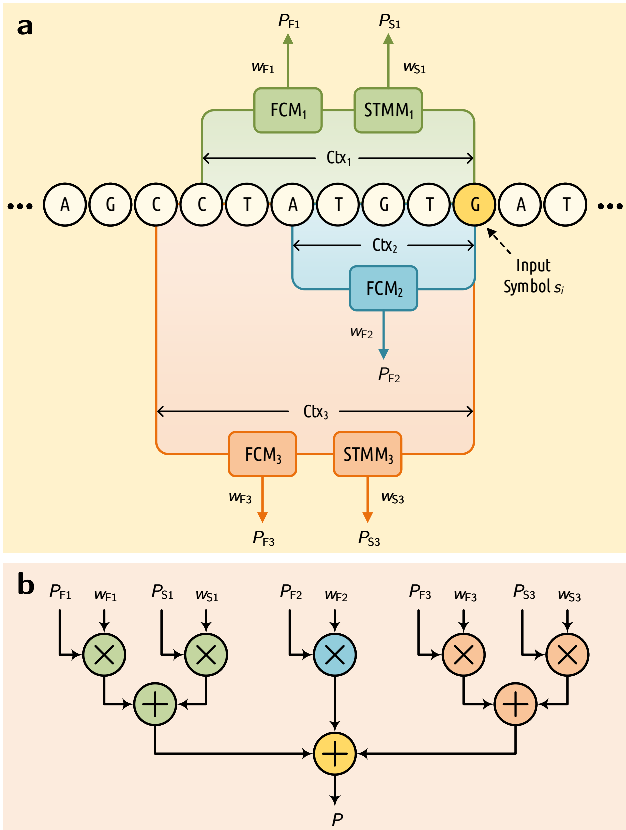

Figure 2. Data modelling by Smash++. (a) cooperation between finite-context models (FCMs) and substitutional-tolerant Markov models (STMMs). Note that each STMM needs to be associated with an FCM. (b) probability of an input symbol is estimated by employing the probability and weight values that have been obtained from processing previous symbols.

Finite-context model (FCM)

A finite-context model considers Markov property to estimate the probability of the next symbol in an information source, based on the past k symbols (a context of size k) [? 4?]. Denoting the context as $c_{k,i} = s_{i-k}s_{i-k+1}\dots s_{i-2}s_{i-1}$, the probability of the next symbol s_i in an information source S, which is posed at i, can be estimated as

$$P_m(s_i|c_{k,i}) = \frac{N(s_i|c_{k,i}) + \alpha}{N(c_{k,i}) + \alpha|\Theta|}, \tag{2}$$

in which m stands for model (FCM in this case), $N(s_i|c_{k,i})$ shows the number of times that the information source has generated symbol s_i in the past, $|\Theta|$ denotes size of the alphabet Θ, $N(c_{k,i}) = \sum_{b\in\Theta} N(b|c_{k,i})$ represents the total number of events occurred for the context $c_{k,i}$ and α allows to keep a balance between the maximum likelihood estimator and the uniform distribution. Eq. 2 turns to the Laplace estimator, for $\alpha = 1$, and also behaves as a maximum likelihood estimator, for large number of events i [5].

Substitutional tolerant Markov model (STMM)

A substitutional tolerant Markov model [6] is a probabilistic-algorithmic model that assumes at each position, the next symbol in the information source is the symbol which has had the highest probability of occurrence in the past. This way, an STMM ignores the real next symbol in the source. Denoting the past k symbols as $c_{k,i} = s_{i-k}s_{i-k+1}\ldots s_{i-2}s_{i-1}$, the probability of the next symbol s_i, can be estimated as

$$P_m(s_i|c'_{k,i}) = \frac{N(s_i|c'_{k,i}) + \alpha}{N(c'_{k,i}) + \alpha|\Theta|}, \quad (3)$$

where N represents the number of occurrences of symbols, that is saved in memory, and $c'_{k,i}$ is a copy of the context $c_{k,i}$ which is modified as

$$c'_{k,i} = \underset{\forall b\in\Theta}{\arg\max}\, P_m(b|c'_{k,i}). \quad (4)$$

STMMs can be used along with FCMs to modify the behavior of Smash++ in confronting with nucleotide substitutions in genomic sequences. These models have the potential to be disabled, to reduce the number of mathematical calculations and consequently, increase the performance of the proposed method. Such operation is automatically performed using an array of size k (the context size), named history, which preserves the past k hits/misses. Seeing a symbol in the information source, the memory is checked for the symbol with the highest number of occurrences. If they are equal, a hit is saved in the history array; otherwise, a miss is inserted into the array. Before getting to store a hit/miss in the array, it is checked for the number of misses and in the case they are more than a predefined threshold t, the STMM will be disabled and also the history array will be reset. This process is performed for each symbol in the sequence.

This example shows the distinction between a finite-context model and a substitutional tolerant Markov model. Assume, the current context at position i is $c_{11,i}$ = GGCTAACGTAC, and the number of occurrences of symbols saved in memory is A = 10, C = 12, G = 13 and T = 11. Also, the symbol to appear in the sequence is T. An FCM would consider the next context as $c_{11,i+1}$ = GCTAACGTACT, while an STMM would consider it as $c'_{11,i+1}$ = GCTAACGTACG, since the base G is the most probable symbol, based on the number of occurrences stored in memory.

Cooperation of FCMs and STMMs

When FCMs and STMMs are in cooperation, the probability of the next symbol s_i in an information source S, at position i, can be estimated as

$$P(s_i) = \sum_{m\in M_F} P_m(s_i|c_{k,i})\, w_{m,i} + \sum_{m\in M_S} P_m(s_i|c'_{k,i})\, w'_{m,i},$$
$$\forall s_i \in S,\ 1 \le i \le |S|,\ 1 \le k \le i-1, \quad (5)$$

in which M_F and M_S denote sets of FCMs and STMMs, respectively, $P_m(s_i|c_{k,i})$ shows the probability of the next symbol estimated by the FCM, $P_m(s_i|c'_{k,i})$ represents this probability estimated by the STMM, and $w_{m\,i}$ and $w'_{m,i}$ are weights assigned to each model based on its performance. We have

$$\forall m \in M_F: \quad w_{m,i} \propto (w_{m,i-1})^{\gamma_m} P_m(s_i|c_{k+1,i-1}),$$
$$\forall m \in M_S: \quad w'_{m,i} \propto (w'_{m,i-1})^{\gamma'_m} P_m(s_i|c'_{k+1,i-1}), \quad (6)$$

where γ_m and $\gamma'_m \in [0,1)$ are forgetting factors predefined for each model. Also,

$$\sum_{m\in M_F} w_{m,i} + \sum_{m\in M_S} w'_{m,i} = 1. \quad (7)$$

By experimenting different forgetting factors for models, we have found that higher factors should be assigned to models that have higher context-order sizes (less complexity) and vice versa. As an example, when the context size $k = 6$, γ_m or $\gamma'_m \simeq 0.9$ and when $k = 18$, γ_m or $\gamma'_m \simeq 0.95$ would be appropriate choices. These values show that forgetting factor and complexity of a model are inversely related.

Storing models in memory

The FCMs and STMMs include, in fact, count values which need to be saved in memory. For this purpose, four different data structures have been employed considering the context-order size k, as follows:

- table of 64 bit counters, for $k \in [1, 11]$,
- table of 32 bit counters, for $k = \{12, 13\}$,
- table of 8 bit approximate counters, for $k = 14$, and
- Count-Min-Log sketch of 4 bit counters, for $k \ge 15$.

The table of 64 bit counters, that is shown in Figure 3a, simply saves number of events for each context. The table of 32 bit counters saves in each position the number of times that the associated context is observed. When a counter reaches to the maximum value $2^{32} - 1 = 4294967295$, all the counts will be renormalized by dividing by two, as shown in Figure 3b.

The approximate counting is a method that employs probabilistic techniques to count large number of events, while using small amount of memory [7]. Figure 4 shows the algorithm for two major functions associated with this method, Update and Query. In order to update the counter, a pseudo-random number generator (PRNG) is used the number of times of the counter's current value to simulate flipping a coin. If it comes up 0/Heads each time or 1/Tails each time, the counter will be incremented. Figure 3c shows the difference between arithmetic and approximate counting, and also the values which are actually stored in memory. Note that since an approximate counter represents the actual count by an order of magnitude estimate, one only needs to save the exponent. For example, if the actual count is 8, we store it in memory as $\log_2 8 = 3$.

The Count-Min-Log Sketch (CMLS) is a probabilistic data structure to save frequency of events in a table by means of a family of independent hash functions [8]. The algorithm for updating and querying the counter is shown in Figure 5. In order to update the counter, its current value is hashed with d independent hash functions. Then, a coin is flipped the number of times of the counter's current value, employing a pseudo-random number generator. If it comes up 0/Heads each time or 1/Tails each time, the minimum hashed values (out of d values) will be updated, as shown in Figure 3d.

The CMLS requires a family of pairwise independent hash functions $H = \{h : U \to [m]\}$, in which each function h maps some universe U to m bins. To have this family, we use universal hashing by randomly selecting a hash function from a universal family in which $\forall x, y \in U,\ x \neq y: \ P_{h\in H}[h(x) = h(y)] \le \frac{1}{m}$.

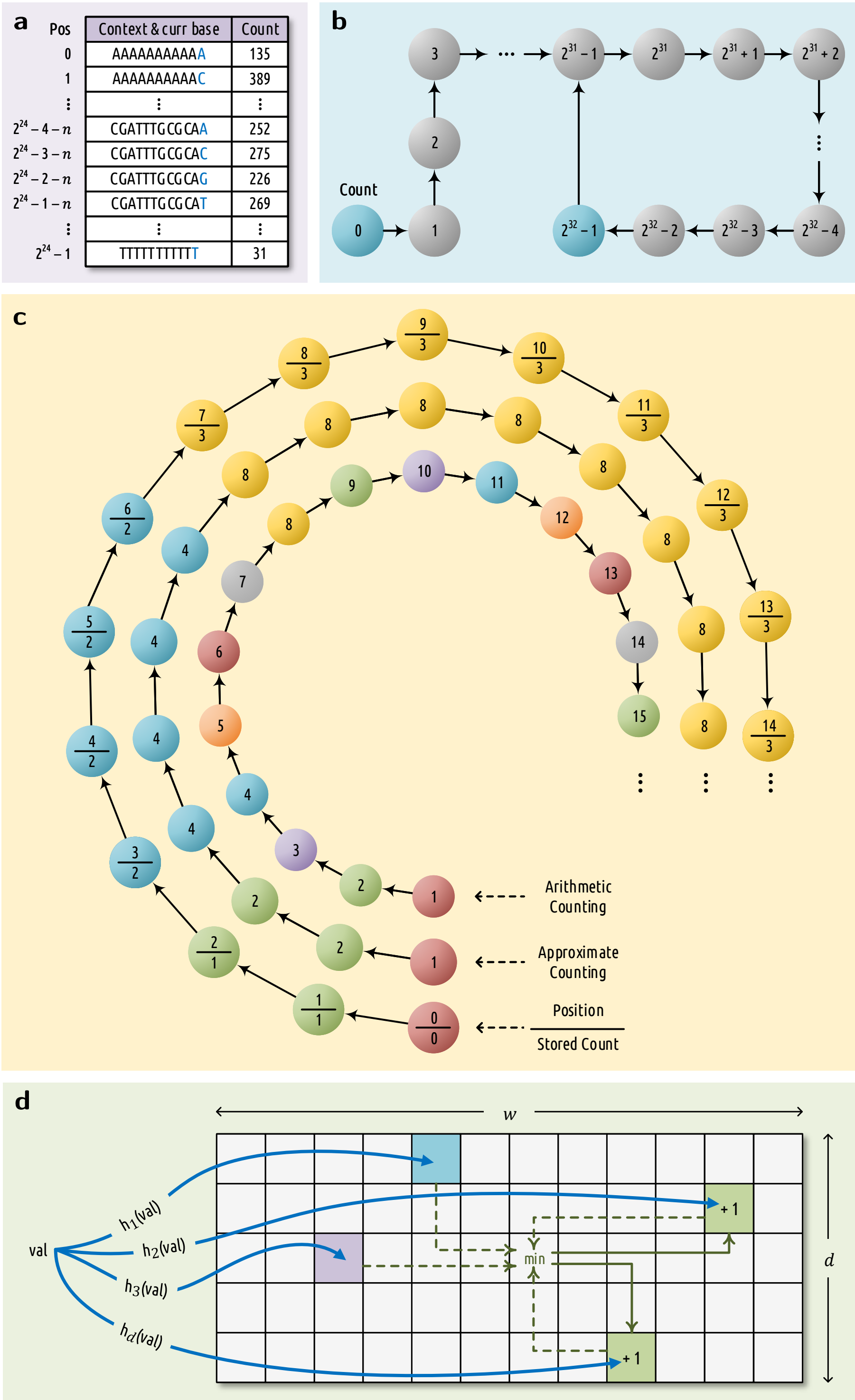

Figure 3. The data structures used by Smash++ to store the models in memory. (a) table of 64 bit counters that uses up to 128 MB of memory, (b) table of 32 bit counters that consumes at most 960 MB of memory, (c) table of 8 bit approximate counters with memory usage of up to 1 GB and (d) Count-Min-Log sketch of 4 bit counters which consumes up to $\frac{1}{2}w \times d$ B of memory, e.g., if $w = 2^{30}$ and $d = 4$, it uses 2 GB of memory.

The hash function can be obtained by

$$h_{a,b}(x) = ((ax + b) \mod p) \mod m, \tag{8}$$

where $p \geq m$ is a prime number and a and b are randomly chosen integers modulo p with $a \neq 0$.

Finding similar regions

To find similar regions in reference and target files, a quantity is required for measuring the similarity. We use "per symbol information content" for this purpose, which can be calculated as

$$I(s_i) = -\log_2 P(s_i) \quad \text{bpb}, \quad \forall s_i \in S,\ 1 \leq i \leq |S| \tag{9}$$

```
function IncreaseDecision(x)
    return True with probability 1/2^x, else False
end function

function Update(x)
    c ← table[x]
    if IncreaseDecision(c) = True then
        table[x] ← c + 1
    end if
end function

function Query(x)
    c ← table[x]
    return 2^c − 1
end function
```

Figure 4. Approximate counting update and query.

```
Input: sketch width w, sketch depth d, m bins, prime
    p ≥ m, randomly chosen integers a_{1..d} and b_{1..d}
    modulo p with a ≠ 0

function Hash(k, x)                    ▷ Universal hash family
    return ((a_k x + b_k) mod p)  mod m
end function

function MinCount(x)
    minimum ← 15                       ▷ Biggest 4 bit number
    for k ← 1 to d do
        h ← Hash(k, x)
        if sketch[k][h] < minimum then
            minimum ← sketch[k][h]
        end if
    end for
    return minimum
end function

function IncreaseDecision(x)
    return True with probability 1/2^x, else False
end function

function Update(x)
    c ← MinCount(x)
    if IncreaseDecision(c) = True then
        for k ← 1 to d do
            h ← Hash(k, x)
            if sketch[k][h] = c then
                sketch[k][h] ← c + 1
            end if
        end for
    end if
end function

function Query(x)
    c ← MinCount(x)
    return 2^c − 1
end function
```

Figure 5. Count-Min-Log Sketch update and query.

where $P(s_i)$ denotes the probability of the next symbol s_i in the information source S, obtained by Equation 5, and also "bpb" stands for bit per base.

The information content is the amount of information required to represent a symbol in the target sequence, based on the model of the reference sequence. The less the value of this measure is for two regions, the more amount of information is shared between them, and therefore, the more similar are the two regions. Note that a version of this measure has been introduced in [5], which employs a single FCM to calculate the probabilities. In this paper, however, we exploit a cooperation between multiple FCMs and STMMs for highly accurate calculation of such probabilities.

The procedure of finding similar regions in a reference and a target sequence, illustrated in Figure 6, is as follows: after creating the model of the reference, the target is compressed based on that model and the information content is calculated for each symbol in the target. Then, the content of the whole target sequence is smoothed by Hann window [9], which is a discrete window function given by $w[n] = 0.5 - 0.5 \cos\left(\frac{2\pi n}{N}\right)$, where $0 \leq n \leq N$ and length of the window is $N + 1$. Next, the smoothed information content is segmented considering a predefined threshold, meaning that the regions with the content greater than the threshold are filtered out. This is carried out for both regular and inverted repeat homologies and at the end, the result would be the regions in the target sequence that are similar to the reference sequence (Figure 6a). The described phase repeats for all of the target regions found, in the way that after creating the model for each region, the whole reference sequence is compressed to find those regions in the reference that are similar to each of the target regions (Figure 6b). The final result would have the form of Figure 6c.

Computing complexity

After finding the similar regions in reference and target sequences, we evaluate redundancy in each region, knowing that it is inversely related to Kolmogorov complexity, i.e., the more complex a sequence is, the less redundant it will be [10]. The Kolmogorov complexity, K, of a binary string s, of finite length, is the length of the smallest binary program p that computes s in a universal Turing machine and halts. In other words, $K(s) = |p|$ is the minimum number of bits required to computationally retrieve the string s [11, 12].

The Kolmogorov complexity is not computable, hence, an alternative is required to compute it approximately. It has been shown in the literature that a compression algorithm can be employed for this purpose [13, 14, 15]. In this paper, we employ a reference-free compressor to approximate the complexity and consequently, the redundancy of the founded similar regions in the reference and the target sequences. This compressor works based on cooperation of FCMs and STMMs, which has been previously described in detail. Note that the difference between reference-based and reference-free version of such compressor is that in the former mode, a model is first created for the reference sequence and then, the target sequence is compressed based on that model, while in the latter mode, the model is progressively created at the time of compressing the target sequence.

Experiments setup

Datasets

Results

Smash++ and several other methods have been carried out on a collection of synthetic and real sequences. The machine used for the tests had an 8-core 3.40 GHz Intel® Core™ i7-6700 CPU and 32 GB of RAM.

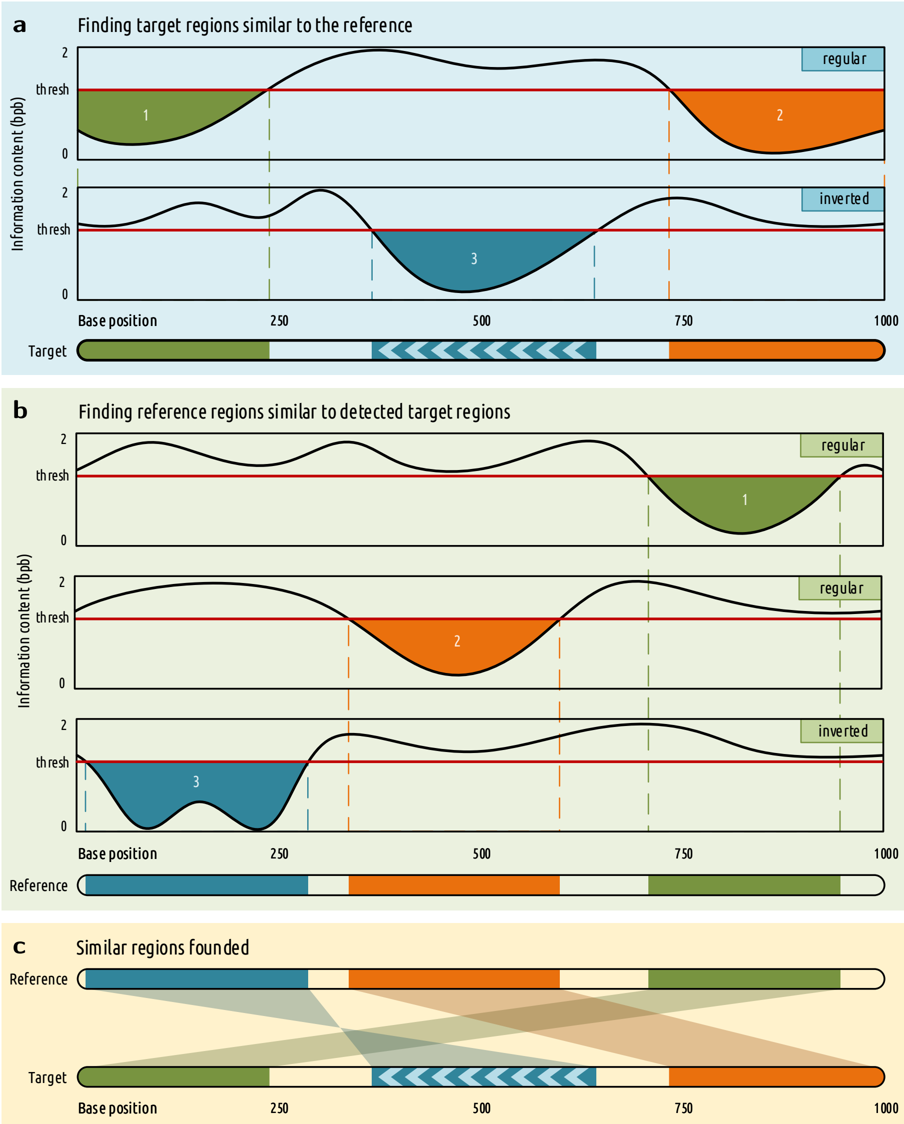

Figure 6. Finding similar regions in reference and target sequences. Smash++ finds, first, the regions in the target that are similar to the reference, and then, finds the regions in the reference that are similar to the detected target regions. This procedure is performance for both regular and inverted homologies.

Availability of source code and requirements (optional, if code is present)

Lists the following:

- Project name: e.g. My bioinformatics project
- Project home page: e.g. `http://sourceforge.net/projects/mged`
- Operating system(s): e.g. Platform independent
- Programming language: e.g. Java
- Other requirements: e.g. Java 1.3.1 or higher, Tomcat 4.0 or higher
- License: e.g. GNU GPL, FreeBSD etc. Any restrictions to use by non-academics: e.g. licence needed

This needs to be under an Open Source Initiative approved license where practicable compiled running software is made available. If the code is not hosted in a repository the *GigaScience* GitHub repository is also available for this purpose.

Availability of supporting data and materials

GigaScience requires authors to deposit the data set(s) supporting the results reported in submitted manuscripts in a publicly-accessible data repository such as *Giga*DB (see *Giga*DB database terms of use for complete details). This section should be included when supporting data are available and must include the name of the repository and the permanent identifier or accession number and persistent hyperlinks for the data sets (if appropriate). The following format is recommended:

"The data set(s) supporting the results of this article is(are) available in the [repository name] repository, [cite unique persistent identifier]."

Following the Joint Declaration of Data Citation Principles, where appropriate we ask that the data sets be cited where it is first mentioned in the manuscript, and included in the reference list. If a DOI has been issued to a dataset please always cite it using the DOI rather than the less stable URL the DOI resolves to (e.g. `http://dx.doi.org/10.5524/100044` rather than `http://gigadb.org/dataset/100044`). For more see:

Data Citation Synthesis Group: Joint Declaration of Data Citation Principles. Martone M. (ed.) San Diego CA: FORCE11; 2014 [`https://www.force11.org/datacitation`]

Table 1. Datasets used in the experiments.

Reference	Length (base)	Target	Length (base)	Description
Synthetic data – generated by GOOSE ??				
RefS	1,000	TarS	1,000	RefS has four segments: I, II, III and IV. To build TarS, I and IV are inversely repeated, II is mutated 1% and III is duplicated. Read lines are 50 base long.
RefM	100,000	TarM	100,000	For building TarM, segments I and III of RefM (out of total four) are duplicated, segment II is inversely repeated and segment IV is mutated 1%. The length of read lines is 100.
RefL	5,000,000	TarL	5,000,000	
RefXL	100,000,000	TarXL	100,000,000	
RefMut	60,000	TarMut	60,000	
Real data				

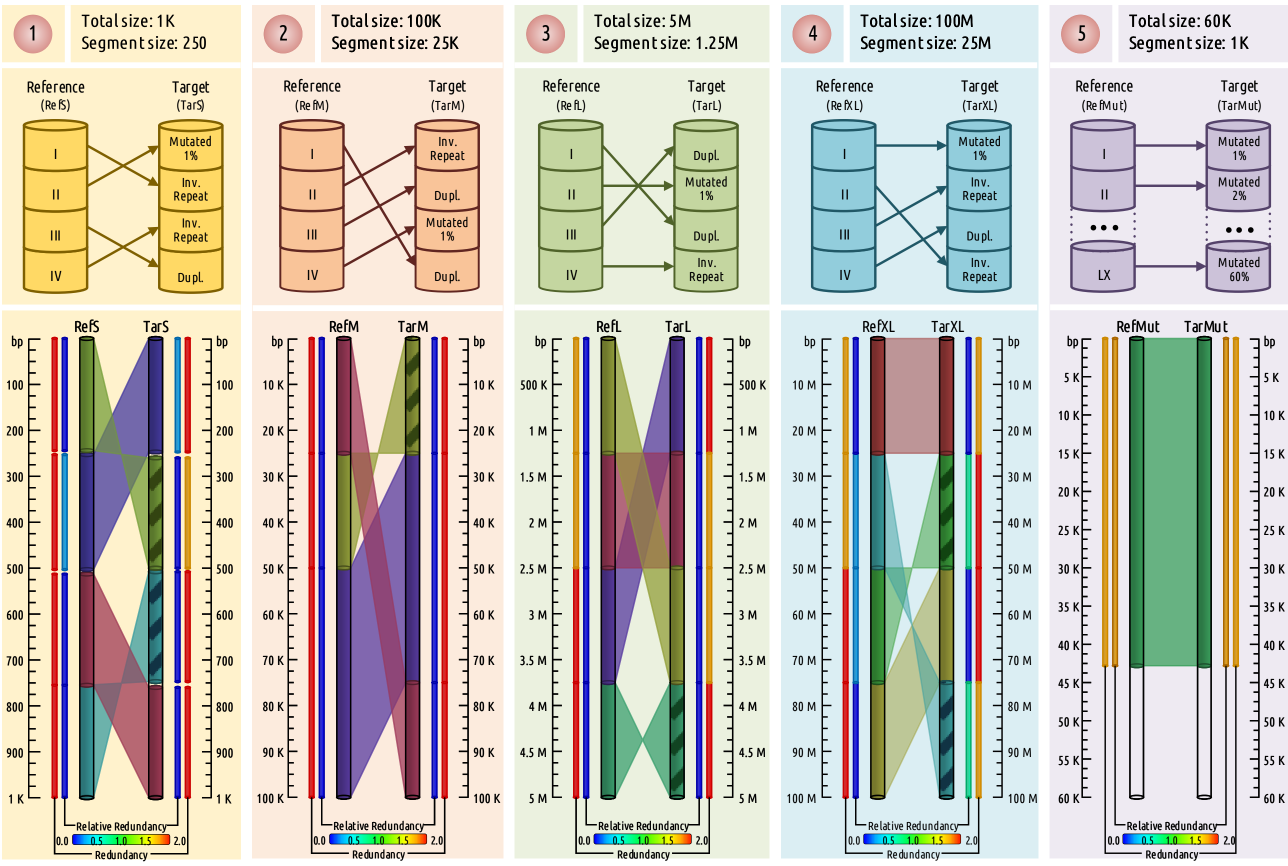

Figure 7. synthetic.

A list of available scientific research data repositories can be found in res3data and BioSharing.

Declarations

List of abbreviations

CMLS: Count-Min-Log Sketch; CPU: central processing unit; FCM: finite-context model; GB: gigabyte; GHz: gigahertz; KB: kilobyte; MB: megabyte; RAM: random access memory; PRNG: pseudo-random number generator; STMM: substitutional tolerant Markov model.

Ethical Approval (optional)

Manuscripts reporting studies involving human participants, human data or human tissue must:

- include a statement on ethics approval and consent (even where the need for approval was waived)
- include the name of the ethics committee that approved the study and the committee's reference number if appropriate

Studies involving animals must include a statement on ethics approval and have been treated in a humane manner in line with the ARRIVE guidelines.

See our editorial policies for more information.

If your manuscript does not report on or involve the use of any animal or human data or tissue, this section is not applicable to your submission. Please state "Not applicable" in this section.

Consent for publication

If your manuscript contains any individual person's data in any form, consent to publish must be obtained from that person, or in the case of children, their parent or legal guardian. All presentations of case reports must have consent to publish. You can use your institutional consent form. You should not send the form to us on submission, but we may request to see a copy at any stage (including after publication). Please also confirm you have followed national guidelines on data collection and release in the place the research was carried out, for example confirming you have Ministry of Science and Technology (MOST) approval in China.

If your manuscript does not contain any individual person's data, please state "Not applicable" in this section.

Competing Interests

The authors declare that they have no competing interests.

Funding

This work was supported by Programa Operacional Factores de Competitividade – COMPETE (FEDER); and by national funds through the Foundation for Science and Technology (FCT), in the context of the projects [UID/CEC/00127/2013, PTCD/EEI-SII/6608/2014] and the grant [PD/BD/113969/2015].

Author's Contributions

The individual contributions of authors to the manuscript should be specified in this section. Guidance and criteria for authorship can be found in our editorial policies. We would recommend you follow some kind of standardised taxonomy like the CASRAI CRediT (Contributor Roles Taxonomy).

Acknowledgements

Please acknowledge anyone who contributed towards the article who does not meet the criteria for authorship including anyone who provided professional writing services or materials.

Authors should obtain permission to acknowledge from all those mentioned in the Acknowledgements section. If you do not have anyone to acknowledge, please write "Not applicable" in this section.

See our editorial policies for a full explanation of acknowledgements and authorship criteria.

Group authorship: if you would like the names of the individual members of a collaboration group to be searchable through their individual PubMed records, please ensure that the title of the collaboration group is included on the title page and in the submission system and also include collaborating author names as the last paragraph of the "Acknowledgements" section. Please add authors in the format First Name, Middle initial(s) (optional), Last Name. You can add institution or country information for each author if you wish, but this should be consistent across all authors.

Please note that individual names may not be present in the PubMed record at the time a published article is initially included in PubMed as it takes PubMed additional time to code this information.

Authors' information (optional)

You may choose to use this section to include any relevant information about the author(s) that may aid the reader's interpretation of the article, and understand the standpoint of the author(s). This may include details about the authors' qualifications, current positions they hold at institutions or societies, or any other relevant background information. Please refer to authors using their initials. Note this section should not be used to describe any competing interests.

References

1. Fan J, Peng H. Nonconcave penalized likelihood with a diverging number of parameters. Ann Statist 2004;32:928–61.
2. Cox DR. Regression models and life tables (with Discussion). J R Statist Soc B 1972;34:187–220.
3. Heard NA, Holmes CC, Stephens DA. A Quantitative Study of Gene Regulation Involved in the Immune Response of Anopheline Mosquitoes: An Application of Bayesian Hierarchical Clustering of Curves. J Am Statist Assoc 2006;101:18–29.
4. Hosseini M, Pratas D, Pinho AJ. AC: A Compression Tool for Amino Acid Sequences. Interdisciplinary Sciences: Computational Life Sciences 2019;11(1):68–76.
5. Pratas D, Silva RM, Pinho AJ, Ferreira PJ. An alignment-free method to find and visualise rearrangements between pairs of DNA sequences. Scientific reports 2015;5:10203.
6. Pratas D, Hosseini M, Pinho AJ. Substitutional tolerant Markov models for relative compression of DNA sequences. In: International Conference on Practical Applications of Computational Biology & Bioinformatics (PACBB) Springer; 2017. p. 265–272.
7. Morris R. Counting Large Numbers of Events in Small Registers. Commun ACM 1978;21(10):840–842.
8. Pitel G, Fouquier G. Count-min-log sketch: Approximately counting with approximate counters. In: International Symposium on Web AlGorithms Deauville, France; Jun 2015. .
9. Blackman R, Tukey J. Particular pairs of windows. The measurement of power spectra, from the point of view of communications engineering 1959;p. 95–101.
10. Hosseini M, Pratas D, Pinho AJ. Cryfa: a secure encryption tool for genomic data. Bioinformatics 2018;35(1):146–148.
11. Turing A. On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society 1936;42(2):230–265.
12. Li M, Vitányi P. An introduction to Kolmogorov complexity and its applications. 3rd ed. Springer; 2009.
13. Zenil H, Soler-Toscano F, Delahaye JP, Gauvrit N. Two-dimensional Kolmogorov complexity and an empirical validation of the Coding theorem method by compressibility. PeerJ Computer Science 2015 Sep;1:e23.
14. Antão R, Mota A, Machado JAT. Kolmogorov complexity as a data similarity metric: application in mitochondrial DNA. Nonlinear Dynamics 2018 Aug;93(3):1059–1071.
15. Faloutsos C, Megalooikonomou V. On data mining, compression, and Kolmogorov complexity. Data Mining and Knowledge Discovery 2007 Aug;15(1):3–20.