SeqBio 2013

Thierry Lecroq, Eric Rivals and Hélène Touzet (Eds.)

Montpellier, France
November 25th and 26th, 2013

GdR BiM gdr iM

Table of Contents

Invited Talks

Regular Submissions

Preface

The pluridisciplinary workshop SeqBio 2013 was held at the CNRS Campus in Montpellier, France on November 2013, 25th and 26th. It gathered computer science and bioinformatic communities working on textual analysis methods and biologists and geneticists interested in sequence bioinformatics.

Thanks to the financial support of GdR (working groups) BIM (BioInformatique Moléculaire) and IM (Informatique Mathématique) of the CNRS and of the project MASTODONS SePhHaDe, the participation was completely free.

The programme includes talks selected on submissions and two invited talks by Robert Giegerich and Paolo Ribeca.

The problems tackled during SeqBio spread from combinatorics on words and text algorithmics to their applications to bioinformatics analysis of biological sequences. This includes, without being restricted to, the following topics:

- text algorithmics;
- indexing data structures;
- combinatorics and statistics on words;
- high performance or parallel algorithmics;
- text mining;
- compression;
- alignment and similarity search;
- pattern or repeat matching, extraction and inference;
- analysys of high throughput sequencing data (genomic, RNA-seq, Chip-seq, ...);
- genome annotation, gene prediction;
- haplotypes and polymorphisms;
- comparative genomics;
- control signals.

This meeting comes after the following previous editions:

- Marne-la-Vallée, November 2012;
- Lille, December 2011;
- Rennes, January 2011;
- Montpellier, January 2010;
- Rouen, September 2008;
- Marne-la-Vallée, September 2007;
- Orsay, November 2005;
- Lille, December 2004;
- Nantes, May 2004;
- Montpellier, November 2003;
- Nancy, January 2003;
- Rouen, June 2002;
- Montpellier, March 2002.

Programme Committee

- Guillaume Blin, LIGM, Univ. Marne-la-Vallée
- Jérémie Bourdon, LINA, Univ. Nantes
- Christine Brun, TAGC, Inserm UMR 1090, Marseille
- Annie Chateau, LIRMM, CNRS Univ. Montpellier 2
- Hélène Chiapello, INRA Toulouse
- Julien Clément, GREYC, Univ. Caen
- Éric Coissac, LECA Univ. Grenoble 1
- Bernard de Massy, CNRS Montpellier
- Thomas Faraut, INRA Toulouse
- Nicolas Galtier, ISEM, CNRS Univ. Montpellier 2
- Thierry Lecroq, LITIS, Univ. Rouen (chair)
- Claire Lemaitre, INRIA Rennes
- Laurent Mouchard, LITIS, Univ. Rouen
- Macha Nikolski, LABRI, Univ. Bordeaux I
- Olivier Panaud, LGDP, Univ. Perpignan
- Fabio Pardi, LIRMM, CNRS Univ. Montpellier 2
- Eric Rivals, LIRMM, CNRS Univ. Montpellier 2 (chair)
- Eric Tannier, LBBE, CNRS Univ. Lyon I
- Hélène Touzet, LIFL, Univ. Lille I (chair)
- Raluca Uricaru, LABRI, Univ. Bordeaux I
- Jean-Stéphane Varré, LIFL, Univ. Lille I

Organizing Committee

The local organization has been realized by:

- the team “Methods and Algorithms for Bioinformatics (MAB)” of the LIRMM (Lab. of Computer Science, Robotics and Microelectronics of Montpellier)
- the “Institut de Biologie Computationnelle (IBC)”
- the French CNRS (Centre National de la Recherche Scientifique)

Members:

- Bastien Cazaux
- Annie Chateau
- Maxime Hébrard
- Vincent Lefort
- Sylvain Milanesi
- Fabio Pardi
- Eric Rivals

I Can Only Read Equations: The ICORE framework for dynamic programming over sequences and trees

Robert Giegerich

Univ. of Bielefeld, Germany, Practical Computer Science lab

Dynamic programming problems are ubiquitous in bioinformatics, mostly dealing with sequences and tree-structured data. A large class of such problems can be cast in a uniform framework based on algebras and term rewrite systems:

A solution of a dynamic programming problem, indicated by an optimal score, can be represented by the formula (term) which computes this score. A simple term rewrite system can specify the transformation of such a formula “backwards” to the input(s) it is derived from. The inverse of this rewrite relation constitutes a declarative problem specification with a high intuitive appeal. Algorithmic ideas and relationships between similar problems come out clearly, and re-use of specification components is high.

The presentation will introduce the novel framework of Inverse COupled REwrite systems (ICOREs), and demonstrate their appeal with a familiar set of bioinformatics problems encoded as ICOREs. It will indicate the sub-class of the framework which we can implement automatically and efficiently today, and sketch the challenges of full ICORE implementation.

ICOREs are joint work with Hélène Touzet

The GEM suite for sequencing data analysis: present and future perspectives

Paolo Ribeca

National Center of Genomic Analyses, Barcelona

In this talk we describe the key features and the future roadmap of the GEM suite of software tools for sequencing data analysis. Built upon a couple of high-performance alignment and processing libraries, as of today several utilities and pipelines are available, mainly a mapper (which has been shown to be significantly faster and more accurate than many other short sequencing read aligners), a tool to compute the mappability of a reference, an RNA-seq alignment pipeline, plus several ancillary tools for easy genome browser track production and visualization. However, new challenges are just round the corner (mainly new sequencing technologies producing longer reads with a higher error rate, and the general need for a higher processing yield), and we discuss how we prepare to tackle them.

Finding the Core-genes of Chloroplast Species

Bassam Alkindy, Jean-François Couchot, Christophe Guyeux and Michel Salomon

FEMTO-ST Institute, UMR 6174 CNRS, University of Franche-Comté, France

Identifying core genes is important to understand evolutionary and functional phylogenies. Therefore, in this work we present two methods to build a genes content evolutionary tree. More precisely, we focus on the following questions considering a collection of 99 chloroplasts annotated from NCBI [1] and Dogma [2] : how can we identify the best core genome and what is the evolutionary scenario of these chloroplasts. Two methods are considered here. The first one is based on NCBI annotation, it is explained below. We start by the following definition.

Definition 1 Let $\mathcal{A} = \{A, T, C, G\}$ be the nucleotides alphabet, and $\mathcal{A}^*$ be the set of finite words on $\mathcal{A}$ (*i.e.*, of DNA sequences). Let $d : \mathcal{A}^* \times \mathcal{A}^* \rightarrow [0, 1]$ be a distance on $\mathcal{A}^*$. Consider a given value $T \in [0, 1]$ called a threshold. For all $x, y \in \mathcal{A}^*$, we will say that $x \sim_{d,T} y$ if $d(x, y) \leqslant T$.

$\sim_{d,T}$ is obviously an equivalence relation. When $d = 1 - \Delta$, where Δ is the similarity scoring function embedded into the emboss package (Needleman-Wunch released by EMBL), we will simply denote $\sim_{d,0.1}$ by $\sim$. The method starts by building an undirected graph based on the similarity rates r_{ij} between sequences g_i and g_j (*i.e.*, $r_{ij} = \Delta(g_i, g_j)$). In this latter, nodes are constituted by all the coding sequences of the set of genomes under consideration, and there is an edge between g_i and g_j if the similarity rate r_{ij} is greater than the given similarity threshold. The Connected Components (CC) of the "similarity" graph are thus computed. This produces an equivalence relation between sequences in the same CC based on Definition 1. Any class for this relation is called "gene" here, where its representatives (DNA sequences) are the "alleles" of this gene. Thus this first method produces for each genome G, which is a set $\{g_1^G, ..., g_{m_G}^G\}$ of m_G DNA coding sequences, the projection of each sequence according to π, where π maps each sequence into its gene (class) according to $\sim$. In other words, G is mapped into $\{\pi(g_1^G), ..., \pi(g_{m_G}^G)\}$. Remark that a projected genome has no duplicated gene, as it is a set. The core genome (resp. the pan genome) of G_1 and G_2 is defined thus as the intersection (resp. as the union) of these projected genomes.

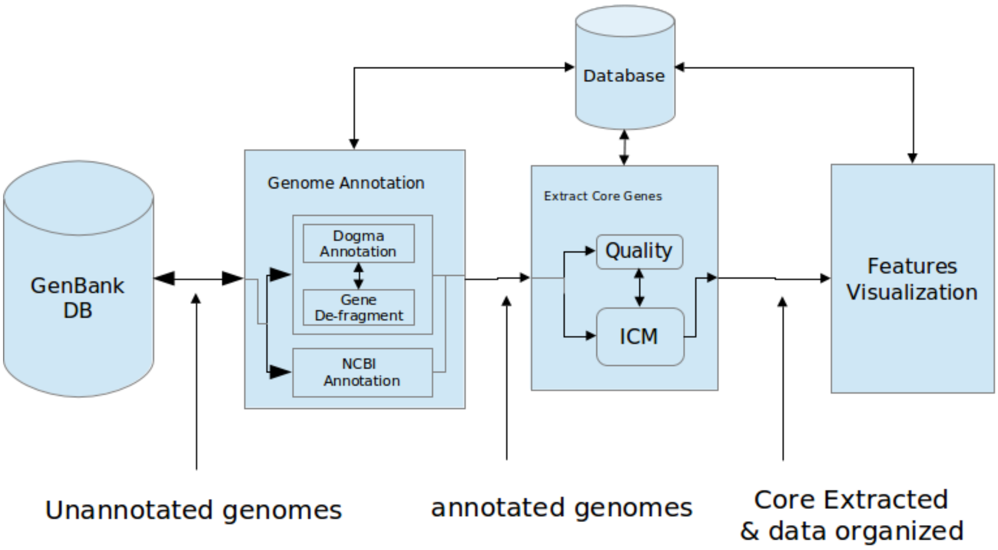


Figure 1: General overview of the system pipeline

We then consider the intersection of all the projected genomes, which is the set of all the genes $\dot{x}$ such that each genome has at least one allele in $\dot{x}$. The pan genome is computed similarly as the union of all the projected genomes. However such approach suffers from producing too small core genomes, for any chosen similarity threshold, compared to what is usually waited by biologists regarding these chloroplasts. We are then left with the following questions: how can we improve the confidence put in the produced core? Can we thus guess the evolution scenario of these genomes?

The second method is based on NCBI and Dogma annotations. In this method, we compute an *Intersection Core Matrix (ICM)*. ICM is a two dimensional symmetric matrix where each row and column represents a genome with its set of genes. Each position in ICM contains the *Intersection Score (IS)* or the cardinality value after intersecting each genome with the other ones. Iteratively, we select the maximum intersection cardinality value in the matrix, according to Equation 1, creating a new core id. Then we remove the two correspondent genomes and add the new established core.

$$Score = \max_{i<j} |x_i \cap x_j| \tag{1}$$

References

[1] E. W. Sayers *et al.* Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research*, 39(suppl 1):D38–D51, 2011.

[2] S. K. Wyman, R. K. Jansen, and J. L. Boore. Automatic annotation of organellar genomes with DOGMA. *Bioinformatics*, 20(172004):3252–3255, 2004.

The HRS-seq: a new method for genome-wide profiling of nuclear compartment-associated sequences

Marie-Odile Baudement, Axel Cournac, Franck Court, Marie Seveno, Hugues Parrinello, Christelle Reynes, Marie-Noëlle Le Lay-Taha, Cathala Guy, Laurent Journot and Thierry Forné

We are interested in how mammalian nuclear organization controls gene expression in physiological and pathological situations. We have suggested that genome organization in gene-rich domains is based on a fundamental level of chromatin compaction: the statistical helix [1]. Functional folding would then be achieved by chromatin looping mediated by locus-specific factors and/or by recruitment to specific nuclear compartments (e.g. rDNA/nucleolus; snRNA genes/Cajal bodies,...). However, most nuclear compartments are difficult to isolate and methods that provide general overviews of such sequences are quite complex to handle. We have developed the HRS-seq method, a novel straightforward genome-wide approach whereby sequences associated with some nuclear compartments are isolated (the HRS fraction) from the rest of genomic DNA (the loop fraction) upon high-salt treatments (HRS=High-salt Recovered Sequences) and subjected to high-throughput sequencing.

After bioinformatic filters applied by the MGX platform of Montpellier, we apply two other filters based on the location of the start of tags with the restriction site used to separate the wo different types of sequences, and based on the length of tags due to our strategy of sequencing. Tags not fulfilling the two conditions are deleted and the sum of tags for each restriction enzyme fragment are calculated, independently in the two fractions (R software).

After the application of a statistical analysis (collaboration with Robert Sabatier), we determine which restriction enzyme fragments are significantly enriched in the HRS fraction versus the loop fraction. With the help of this analysis, we obtained a sub-list of fragments that are called "HRS fragments".

Using mouse liver cells, we showed that these HRS fragments are highly clustered in the genome by the help of differential analysis versus a randomization and that two categories of HRS can be distinguished: AT-rich HRS and GC-rich HRS. Thanks to UCSC genes database, we identify bioinformatically Transcription Start Sites (TSS) inside and close to HRS fragments : remarkably, GC-rich HRS are seen to map close to TSS, including TSS of histone genes (DAVID Ontology analysis). With a collaboration with Axel Cournac, we have decided to try to identify if HRS fragments cluster together in 3 dimension. For this, we have used contact map identified by Hi-C. Firstly, we calculated the mean interaction score obtained by all possible contacts between all HRS fragments, isolated from liver nuclei, and secondly we confront this result versus a randomization. We found a strong difference ; so; our HRS fragments clustered together preferentially in 3D space of the nucleus. We tried to see if there are typical family of repeats over-represented in our data ; we found transfert RNA genes, which are already known to cluster in 3D within the nucleus in yeast [2]. This last finding confirms that a significant part of the GC-rich HRS represents sequences associated with specific nuclear compartments.

To better understand more the specificity of the AT-rich population of HRS, we have decided to cross our data with data from Bas van Steensel lab [3], which identify Lamina-Associated Domains (LADs). These LADs are described as AT-rich sequences but also as associated to the laminB1. We have selected DNA microarrays that are triple positive in three different cell types, and we have compared its positioning with the location of HRS, with the help of the "IRanges" and "GenomicRanges" package. We obtained a strong interaction between AT-rich HRS and LADS. This result is concordant with the knowledge of the so-called MAR sequences that are also AT-rich.

We are now applying this method to cellular models in which specific types of nuclear compartments are perturbed, but also in undifferentiated and differentiated cells. Global profiling of HRS should help us to better understand how genome organization impacts on its functions and so, if some diseases are due to an alteration of the recruitment of specific HRS in these nuclear bodies.

References

[1] Court et al. *Genome Biol.*, 12:R42, 2011.

[2] Haeusler et al. *Genes Dev.*, 22:2204–2214, 2008.

[3] Peric-Hupkes et al. *Molecular Cell*, 38:603–613, 2010.

Greedy algorithm for the shortest superstring and shortest cyclic cover of linear strings

Bastien Cazaux and Eric Rivals

L.I.R.M.M. & Institut Biologie Computationnelle, University of Montpellier II, CNRS U.M.R. 5506, 161 rue Ada, F-34392 Montpellier Cedex 5, France

Abstract: In bioinformatics, the assembly of reads to produce a complete genome currently is a major bottleneck in genomics. This problem has been modeled by the Shortest Superstring problem, where given a set of input words, one asks for a shortest string such that each input word is substring of this *superstring*. This well studied problem is known to be NP-hard [2] and difficult to approximate [1]. The optimisation can measure either the length of the obtained superstring, or the amount of compression it realizes (i.e., the cumulated length of the words minus that of the superstring). Numerous approximation algorithms, which achieve a constant approximation ratio have been described in the literature; see for instance [6]. Many of these use the question of finding a set of cyclic superstrings of minimal length for the input words, also known as *Shortest Cyclic Cover*, as a procedure. We study the greedy algorithm, which iteratively agglomerate the two words having the maximal largest overlap, and has been shown to reach 1/2 compression ratio [7]. Using hereditary systems [5], we provide a simple proof for this 1/2 approximation ratio. By extending the reasoning, we obtain that the greedy algorithm is optimal for the Shortest Cyclic Cover (when each word is allowed to overlap itself). The simplicity of the greedy algorithm confers it practical advantages compared to other complex approximation algorithms in terms of coding for instance.

Résumé : En biologie moléculaire, les méthodes de séquençage d'ADN produisent les séquences de petites portions de la molécule séquencée. Ensuite, la phase d'assemblage consiste à déterminer la séquence complète de la molécule à partir des chevauchements entre les séquences produites. Les molécules d'ADN ou d'ARN peuvent être linéaires ou circulaires. Ainsi, l'assemblage peut être modélisé par la recherche d'une Plus Courte Superchaîne, dont chacune des séquences en entrée est par définition une sous-chaîne. Il s'agit d'un problème NP-difficile [2] et difficile à approximer [1] pour lequel de nombreux algorithmes à ratio constant ont été décrits (par exemple voir [6]). Si l'on recherche non plus une superchaîne linéaire, mais un ensemble de superchaînes circulaires, aussi appelé Couverture Cyclique, le problème s'appelle alors Plus Courte Couverture Cyclique (de chaînes linéaires). Pour ces deux problèmes, nous étudions les performances de l'algorithme glouton qui consiste à itérativement agglomérer deux séquences en entrée ayant le plus grand chevauchement maximal. En utilisant les systèmes héréditaires [5], nous donnons une preuve simple que l'algorithme glouton donne un ratio d'approximation de compression de 1/2 pour la question de la Plus Courte Superchaîne (PCS linéaire - [7]). En étendant le raisonnement, nous obtenons que ce même algorithme répond exactement au problème de la Plus Courte Couverture Cyclique (PCCC). Ce problème, pour un ensemble de mots donnés en entrée, demande un ensemble de chaînes circulaires, telles que leurs tailles cumulées soient minimale et que chaque mot en entrée soit au moins sous-chaîne d'une chaîne cyclique. Une chaîne cyclique est en fait une chaîne linéaire disposée sur un cercle et bouclant sur elle-même. L'algorithme glouton peut être mis en œuvre en temps linéaire par rapport à la somme des longueurs des chaînes en entrée grâce à des structures d'indexation telles que l'arbre des suffixes généralisé. Notre résultat acquiert aussi une portée pratique puisque la résolution de PCCC est utilisée dans divers algorithmes d'approximation de PCS [4, 3].

References

[1] A. Blum, T. Jiang, M. Li, J. Tromp, and M. Yannakakis. Linear approximation of shortest superstrings. In *ACM Symposium on the Theory of Computing*, pages 328–336, 1991.

[2] J. Gallant, D. Maier, and J. A. Storer. On finding minimal length superstrings. *J. Comput. Syst. Sci.*, 20 :50–58, 1980.

[3] D. Gusfield. *Algorithms on Strings, Trees and Sequences.* Cambridge University Press, 1997.

[4] H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko. Approximation algorithms for asymmetric tsp by decomposing directed regular multigraphs. *J. ACM*, 52(4) :602–626, July 2005.

[5] J. Mestre. Greedy in Approximation Algorithms. In *Proceedings of 14th Annual European Symposium on Algorithms (ESA)*, volume 4168 of *Lecture Notes in Computer Science*, pages 528–539. Springer, 2006.

[6] M. Mucha. Lyndon words and short superstrings. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 958–972, 2013.

[7] J. Tarhio and E. Ukkonen. A greedy approximation algorithm for constructing shortest common superstrings. *Theor. Comput. Sci.*, 57 :131–145, 1988.

On the number of prefix and border tables

Julien Clément and Laura Giambruno

GREYC, CNRS-UMR 6072, Université de Caen, 14032 Caen, France

Julien Clément: julien.clement@unicaen.fr
Laura Giambruno: laura.giambruno@unicaen.fr

The prefix table of a string w reports for each position i the length of the longest substring of w that begins at i and matches a prefix of w. This table stores the same information as the border table of the string, which memorises for each position the maximal length of prefixes of the string w ending at that position. Indeed two strings have the same border table if and only if they have the same prefix table.

Both tables are useful in several algorithms on strings. They are used to design efficient string-matching algorithms and are essential for this type of applications (see for example [6] or [2]). It has been noted that for some text algorithms (like the Knuth-Morris-Pratt pattern matching algorithm), the string itself is not considered but rather its structure meaning that two strings with the same prefix or border table are treated in the same manner. For instance, strings `abbbbb`, `baaaaa` and `abcdef` are the same in this aspect.

The study of these tables has become topical. In fact several recent articles in literature (cf. [5, 3, 1, 4]) focus on the problem of validating prefix and border tables, that is the problem of checking if an integer array is either the prefix or the border table of at least one string. In a previous paper [7] the authors represented distinct border tables by canonic strings and gave results on generation and enumeration of these string for bounded and unbounded alphabets. Some of these results were reformulated in [4] using automata-theoretic methods. Note that different words on a binary alphabet have distinct prefix/border tables. This gives us a trivial lower bound in 2^{n-1} (since exchanging the two letters of the alphabet does not change tables). This is no longer true as soon as the alphabet has cardinality strictly greater than 2: for instance, words `abb` and `abc` admit the same prefix table $[3, 0, 0]$.

In this paper we are interested in giving better estimations on the number of prefix/border tables p_n of words of a given length n, that those known in literature.

For this purpose, we define the combinatorial class of *p-lists*, where a p-list $L = [\ell_1, \ldots, \ell_k]$ is a finite sequence of non negative integers.

We constructively define an injection ψ from the set of prefix tables to the set of p-lists which are easier to count. In particular we furnish an algorithm associating to a prefix table a p-list. We define *prefix lists* as p-lists that are images of prefix tables under ψ. We moreover describe an "inverse" algorithm that associates to a prefix list $L = \psi(P)$ a word whose prefix table is P. This result confirms the idea that prefix-lists represent a more concise representation for prefix tables.

We then deduce a *new upper bound and a new lower bound on the number p_n of prefix tables* (see Table 1 for first numerical values) for strings of length n or, equivalently, on the number of border tables of length n.

Let $\varphi = \frac{1}{2}(1 + \sqrt{5}) \approx 1.618$, the golden mean, we have:

Proposition 1 (Upper bound) *The number of valid prefix tables p_n can be asymptotically upper bounded by the quantity $\frac{1}{2}\left(1 + \frac{\sqrt{5}}{5}\right)\ (1 + \varphi)^n + o(1)$.*

Proposition 2 (Lower bound) *For any $\epsilon > 0$ there exists a family of prefix tables $(\mathcal{L}_n)_{n\geq 0}$ such that* $\mathrm{Card}(\mathcal{L}_n) = \Omega((1 + \varphi - \epsilon)^n)$.

The problem of finding an asymptotic equivalent for the number of prefix tables is however still open, and would require a very fine understanding of the autocorrelation structure of words.

n	$p_{n,1}$	$p_{n,2}$	$p_{n,3}$	$p_{n,4}$	$p_{n,5}$	p_n
1	1					1
2	1	1				2
3	1	3				4
4	1	7	1			9
5	1	15	4			20
6	1	31	15			47
7	1	63	46			110
8	1	127	134	1		263
9	1	255	370	4		630
10	1	511	997	16		1525
11	1	1023	2625	52		3701
12	1	2047	6824	162		9034
13	1	4095	17544	500		22140
14	1	8191	44801	1467		54460
15	1	16383	113775	4180		134339
16	1	32767	287928	11742	1	332439
17	1	65535	726729	32466	4	824735
18	1	131071	1831335	88884	16	2051307
19	1	262144	4610078	241023	52	5113298

Table 1: First values: p_n is the total number of prefix tables for strings of size n, $p_{n,k}$ is the number of prefix tables for strings of size n with an alphabet of size k which cannot be obtained using a smaller alphabet.

References

[1] J. Clément, M. Crochemore, and G. Rindone. Reverse engineering prefix tables. In S. Albers and J.-Y. Marion, editors, *26th International Symposium on Theoretical Aspects of Computer Science (STACS 2009)*, volume 3 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 289–300, Dagstuhl, Germany, 2009. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[2] M. Crochemore, C. Hancart, and T. Lecroq. *Algorithms on strings.* Cambridge University Press, Cambridge, UK, 2007.

[3] J.-P. Duval, T. Lecroq, and A. Lefebvre. Border array on bounded alphabet. *Journal of Automata, Languages and Combinatorics*, 10(1):51–60, 2005.

[4] J.-P. Duval, T. Lecroq, and A. Lefebvre. Efficient validation and construction of border arrays and validation of string matching automata. *RAIRO-Theoretical Informatics and Applications*, 43(2):281–297, 2009.

[5] F. Franek, S. Gao, W. Lu, P. J. Ryan, W. F. Smyth, Y. Sun, and L. Yang. Verifying a border array in linear time. *Journal on Combinatorial Mathematics and Combinatorial Computing*, 42:223–236, 2002.

[6] D. Gusfield. *Algorithms on strings, trees and sequences: computer science and computational biology.* Cambridge University Press, Cambridge, UK, 1997.

[7] D. Moore, W. F. Smyth, and D. Miller. Counting distinct strings. *Algorithmica*, 23(1):1–13, 1999.

Indexation de séquences d'ADN au sein d'une base de données NoSQL à l'aide d'algorithmes de hachage perceptuel

Jocelyn De Goër De Herve[1-2,3], Myoung-Ah Kang[1,2], Xavier Bailly[3] and Engelbert Mephu Nguifo[1,2]

[1] CNRS, UMR 6158, LIMOS, Université Blaise Pascal, F-63173 AUBIERE
[2] 2 Clermont Université, Université Blaise Pascal, LIMOS, BP 10448, F-63000 CLERMONT-FERRAND
[3] INRA, UR346 Épidémiologie Animale, F-63122 ST GENÈS CHAMPANELLE

Mots-clefs : *Indexation, Stockage, NoSQL, Bioinformatique, Séquences d'ADN, Alignement de séquences, Traitement d'images, Hachage perceptuel, TCD, Distance de Hamming*

1 Contexte

L'arrivée de nouvelles générations de séquenceurs ADN « haut débit » [12] a permis la production de données génomiques avec un débit de plus en plus élevé pour un coût de plus en plus bas. Le volume de donnée à stocker et à analyser par les biologistes a donc connu une évolution exponentielle. En informatique, ces dernières années l'accélération des calculs a été rendue possible grâce à la parallélisation des algorithmes, la multiplication des unités de calculs des processeurs (cœur) [2, 10] ou le calcul générique à base de processeurs graphiques (GPU) [11]. Cependant, en terme d'analyse de données génomiques, l'évolution de ces technologies n'est pas assez rapide. Outre le fait que cette énorme masse d'informations va entrainer l'émergence de nouveaux questionnements biologiques et de nouvelles applications, un des défis majeurs durant ces 10 prochaines années est de faire évoluer les outils informatiques pour les dimensionner en conséquence.

La recherche de similarité entre des séquences ADN stockées au sein de larges banques de données est une étape fondamentale de toutes études en génomique. Ainsi, de nombreux algorithmes ont été développés. On peut citer encore aujourd'hui comme références dans le domaine, des algorithmes d'alignement local (BLAST [1]), d'alignement global (Needleman-Wusch [14]) et d'alignements issus des méthodes de comparaison de textes [5, 8] ou d'indexation. Néanmoins, la plupart de ces méthodes nécessitent un grand nombre d'opérations complexes ou le chargement en mémoire vive de tout ou partie des séquences brutes pour effectuer les tâches de comparaison, ce qui étant donné l'évolution de la quantité de données à traiter demande de plus en plus de ressources matérielles.

2 Objectifs et méthodes

Le travail présenté ici, consiste à proposer une méthode visant à accélérer la recherche de similarités entre une séquence candidate et une base de données de séquences de références. Elle s'appuie sur des méthodes issues du domaine de la recherche d'image par le contenu [3] et plus particulièrement des méthodes de hachage perceptuel. Contrairement aux algorithmes mentionnés ci-dessus, il ne s'agit pas d'effectuer des alignements de séquences, mais d'effectuer un tri en amont permettant de renvoyer pour une séquence candidate toutes les séquences de références ayant une probabilité non nul d'alignement.

Afin d'accélérer les opérations de lecture et d'écriture, la table de hachage correspondant aux séquences est stockée dans une base de données Redis (de type NoSQL [13, 9] clé/valeur). Ce moteur de base de données réparties, a la particularité de maintenir l'intégralité des données dans la mémoire vive des machines. La table ne contenant que les haches (d'une taille de 64 bits) et les identifiants des séquences, elle se révèle de 8 à 30x moins volumineuse que le volume total des séquences brutes.

Le processus d'indexation par hachage est réalisé via des méthodes de Hachage Perceptuel [6]. De façon générale, ces méthodes permettent l'identification via la génération d'une clé unique (clé de hachage) à partir d'un document multimédia (image, son ou vidéo). Se rapprochant par certains aspects des méthodes de hachage cryptographique (algorithme MD5), elles diffèrent cependant, par le fait que les clés de hachage correspondant à deux documents présentant une légère différence sont relativement proches et comparables par des méthodes telles que la distance de Hamming [7]. Elles ne sont donc pas sensibles à l'« effet avalanche » [4] où une différence d'un bit entre deux documents a pour effet de générer deux clés de hachage radicalement différentes, rendant de ce fait le calcul de distance impossible.

La méthode proposée ici permet l'indexation rapide de séquences ADN grâce à un algorithme de Hachage Perceptuel [16] utilisant une Transformée en Cosinus Discrète (TCD) [15]. Afin de pouvoir utiliser la TCD pour calculer les différents haches, les séquences sont préalablement converties sous forme d'une image en niveau de gris, plus précisément une matrice de pixels attribuant une valeur d'intensité lumineuse à chaque types de nucléotides : Adénine 63, Thymine 127, Cytosine 191 et Guanine 255.

3 Évaluation

Nous avons réalisé une première implémentation de cette méthode (octobre 2013) qui montre qu'avec un ordinateur portable (processeur Core i7 et 4 cœurs), la méthode permet d'indexer 500 000 nucléotides à la seconde soit, 8 000 clés de hachage. Une évaluation de la qualité de la méthode est actuellement en cours de réalisation. L'objectif est de réaliser l'alignement via l'outil BLAST entre une base de données de références et des séquences candidates, dans un premier temps sans l'étape d'indexation, puis dans un second temps avec l'étape d'indexation en amont. Enfin nous procéderons à la comparaison des résultats obtenus avec leurs temps d'exécutions.

References

[1] S. Altschul, W. Gish, W. Miller, E. Myers, , and D. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, pages 215 :403–410, 1990.

[2] V. Balaji. Multi-core processors - an overview, arxiv :1110.3535v1 [cs.ar], 2011.

[3] R. da Silva Torres and A. X. Falcao. Content-based image retrieval : Theory and applications, vol. xiii (2) : 165-189, 2006.

[4] H. Feistel. Cryptography and computer privacy. *Scientific American*, pages May, 228(5) : 15–23, 1973.

[5] N. G. and R. M. *Flexible Pattern Matching in Strings - Practical on-line search algorithms for texts and biological sequences.* 2002.

[6] J. Haitsma, T. Kalker, and J. Oostveen. International workshop on content-based multimedia indexing (cbmi). *Robust Audio Hashing for Content Identification*, pages 4 : 117–124, 2001.

[7] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, pages 147–160, 1950.

[8] D. E. Knuth, J. H. M. Jr, and V. R. Pratt. Fast pattern matching in strings. *SIAM J. Comput.*, pages 6(1) : 323–350, 1977.

[9] N. Leavitt. Will nosql databases live up to their promise ? *Computer*, pages 43(2) : 12–14, 2010.

[10] G. Lowney. Why intel is designing multi-core processors. *Proceedings of the Eighteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 113–113, 2006.

[11] D. Luebke, M. Harris, N. Govindaraju, A. Lefohn, M. Houston, J. Owens, M. Segal, M. Papakipos, and I. Buck. Gpgpu : general-purpose computation on graphics hardware, article no. 208. [doi :10.1145/1188455.1188672], 2006.

[12] M. Metzker. Sequencing technologies - the next generation. *Nature Review Genetics*, pages 31–46, 2010.

[13] A. B. M. Moniruzzaman and S. A. Hossain. Nosql database : New era of databases for big data analytics, arxiv :1307.0191 [cs.db], 2013.

[14] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, page 48 (3) : 443–53, 1970.

[15] K. R. Rao and P. Yip. Discrete cosine transform : Algorithms, advantages, applications. *Academic Press, Boston*, 1990.

[16] C. Zauner. Implementation and benchmarking of perceptual image hash functions. master's thesis. *Upper Austria University of Applied Sciences, Hagenberg Campus*, 2010.

Tyrosine-1 phosphorylation of Pol II CTD is associated with antisense promoter Transcription and active enhancers in mammalian cells

Nicolas Descostes, Martin Heidemann, Ahmad Maqbool, Lionel Spinelli, Romain Fenouil, Marta Gut, Ivo Gut, Dirk Eick and Jean-Christophe Andrau

Genome-wide characterization of transcription mechanisms requires the development of adapted bioinformatics approaches. The development of high-throughput sequencing (HTS) technologies such as ChIP-Seq and RNA-Seq offered the possibility to explore the inner conundrum of transcription almost at a base-pair resolution. However, the revealed complexity of molecular processes and sequences organization and composition [2] brought bioinformaticians to deal with more complex data than ever before. In this talk, I will present how we developed original approaches for HTS data treatment and analysis of transcription processes.

I will first present the R packaqe PASHA (Preprocessing of Aligned Sequences from HTS Analyses) we developed for ChIP-Seq, RNA-Seq and MNase-Seq data treatment. This package deals with removal of sequencing artifacts, elongation of sequenced tags, scoring, binning, multiple alignment of reads, nucleosomal positionning and paired-end sequencing. It also yields different statistics and controls about data treatment.

Then through the case study of Tyrosine 1 phosphorylation (Tyr1P) of the carboxy-terminal domain [8, 1, 3] (CTD) of RNA Polymerase II (Pol II), I will then show how epigenetic [5, 9] data, enhancers [4, 7] and Pol II phospho-isoforms analysis led us to develop specific lines of data analyses. For example, we developed specific strategies for isolating enhancers and studying the spatial organization of Pol II isoforms. Finally, I will show that contrary to previously observed association of Tyr1P to transcription elongation in yeast model [6], Pol II post-translational modification in human cells is involved in the initiation phase and antisense transcription as well in enhancer transcription through the analysis of relevant ChIP-Seq, nucleosome (MNase-Seq) and short strand specific RNA-Seq data.

References

[1] S. Egloff and S. Murphy. Cracking the RNA polymerase II CTD code. *Trends Genet*, 24(6):280–288, 2008.

[2] R. Fenouil, P. Cauchy, F. Koch, N. Descostes, J. Cabeza, and C. Innocenti et al. CpG islands and GC content dictate nucleosome depletion in a transcription-independent manner at mammalian promoters. *Genome Res*, 22(12):2399–2408, 2012.

[3] C. Hintermair, M. Heidemann, F. Koch, N. Descostes, M. Gut, and I. Gut et al. Threonine-4 of mammalian RNA polymerase II CTD is targeted by Polo-like kinase 3 and required for transcriptional elongation. *EMBO J*, 31(12):2784–2797, 2012.

[4] F. Koch, R. Fenouil, M. Gut, P. Cauchy, T. Albert, and J. Zacarias-Cabeza et al. Transcription initiation platforms and GTF recruitment at tissue-specific enhancers and promoters. *Nat Struct Mol Biol*, 18(8):956–963, 2011.

[5] T. Kouzarides. Chromatin modifications and their function. *Cell*, 128(4):693–705, 2007.

[6] A. Mayer, M. Heidemann, M. Lidschreiber, A. Schreieck, M.Sun, and C. Hintermair et al. CTD tyrosine phosphorylation impairs termination factor recruitment to RNA polymerase II. *Science*, 336(6089):1723, 2012.

[7] G. Natoli and J. Andrau. Noncoding transcription at enhancers: general principles and functional models. *Annu Rev Genet*, 46:1–19, 2012.

[8] H. Phatnani and A. Greenleaf. Phosphorylation and functions of the RNA polymerase II CTD. *Genes Dev*, 20(21):2922–2936, 2006.

[9] V. Zhou, A. Goren, and B. Bernstein. Charting histone modifications and the functional organization of mammalian genomes. *Nat Rev Genet*, 12(1):7–18, 2011.

Abelian Repetition in Sturmian Words

Gabriele Fici[1], Alessio Langiu[2], Thierry Lecroq[3], Arnaud Lefebvre[3], Filippo Mignosi[4] and Élise Prieur-Gaston[3]

[1] Università di Palermo, Italy
[2] King's College London, UK
[3] Normandie Université, LITIS EA 4108, Université de Rouen, France
[4] Università dell Aquila, Italy

In this talk we investigate abelian repetitions in Sturmian words. We exploit a bijection between factors of Sturmian words and subintervals of the unitary segment that allows us to study the periods of abelian repetitions by using classical results of elementary Number Theory. If k_m denotes the maximal exponent of an abelian repetition of period m, we prove that $\limsup k_m/m \geq \sqrt{5}$ for any Sturmian word, and the equality holds for the Fibonacci infinite word. We further prove that the longest prefix of the Fibonacci infinite word that is an abelian repetition of period F_j, $j > 1$, has length $F_j(F_{j+1} + F_{j-1} + 1) - 2$ if j is even or $F_j(F_{j+1} + F_{j-1}) - 2$ if j is odd. This allows us to give an exact formula for the smallest abelian periods of the Fibonacci finite words. More precisely, we prove that for $j \geq 3$, the Fibonacci word f_j has abelian period equal to F_n, where $n = \lceil j/2 \rceil$ if $j = 0, 1, 2 \bmod 4$, or $n = 1 + \lceil j/2 \rceil$ if $j = 3 \bmod 4$ [1].

References

[1] G. Fici, A. Langiu, T. Lecroq, A. Lefebvre, F. Mignosi, and É. Prieur-Gaston. Abelian repetitions in sturmian words. In M.-P. Béal and O. Carton, editors, *Proceedings of the 17th International Conference on Developments in Language Theory (DLT 2013)*, volume 7907 of *Lecture Notes in Computer Science*, pages 227–238, Marne-la-Vallée, France, 2013. Springer-Verlag, Berlin.

Co-optimality and Ambiguity, Disentangled

Robert Giegerich and Benedikt Löwes

Faculty of Technology, Bielefeld University, Germany

Summary of the Contribution

Solutions to combinatorial programming problems often are not unique. *Co-optimal* solutions are alternatives that achieve the same (optimal) score, while otherwise, they need not be similar at all. Dynamic programming algorithms can report co-optimal alternatives, or determine at least their number. But they rarely do. If it produces an excessive number of co-optimals, our optimization problem is ill-defined.

While co-optimality is a property of the model, *ambiguity* is a property of the algorithm employed for its solution. An ambiguous algorithm reports the same solution multiple times, often to an exponential degree. Internally creating a larger solution space than actually exists, it also explodes the number of co-optimals. In the presence of ambiguity, we cannot tell whether our model is well-defined or not.

We show how to *disentangle* these issues, using a recent algorithm for minisatellite alignment as a case study. We observe that the ARLEM algorithm produces an exorbitant number of co-optimal alignments. We show that it is ambiguous, and create a variant that is proved to be unambiguous using formal language technology. The unambiguous algorithm demonstrates that, independent of any algorithm, the model allows for a high number of co-optimals. Hence, we conclude that the present model of minisatellite alignment and duplication history reconstruction is not refined enough to yield meaningful alignments.

The presentation at the SEQBIO workshop is closely based on the publication [3], but emphasizes the general approach over the specific findings concerning minisatellite alignments.

Overview of the method

Prerequisites The prerequisite of our method is that we can express our dynamic programming problem at hand in the framework of *algebraic* dynamic programming (ADP) [2].

Its perfect separation of search space and scoring allows us to present our case study on minsatellites as a demonstration of a general method for assessing co-optimality and ambiguity. In ADP, we deal with dynamic programming problems over sequence(s) x. The given problem is encoded by a search space generator $\mathcal{G}$, written in form of a regular tree grammar, and by an evaluation algebra A that includes the objective function h. Solution candidates are trees in $L(\mathcal{G})$ which carry x as their yield sequence. $A(t)$ denotes the score of candidate t. A problem instance is posed by a sequence x (or several sequences), and the problem is solved by computing

$$h[A(t) \mid t \in L(\mathcal{G}), yield(t) = x].$$

The square brackets denote multisets, as there may be co-optimal solutions. The framework of ADP is supported by systems such as Bellman's GAP [4].

Methodical steps

1. Encode the original algorithm for the problem at hand by tree grammar $\mathcal{G}$ and evaluation algebra A (if not already formulated in this way).
2. Design a canonical string representation of solutions. Implement it as a "printing" algebra such that $\mathcal{G}(P, x)$ enumerates the search space in canonical representation, following [1]. Duplicates in the resulting multiset indicate semantic ambiguity, i.e. the *same* solution is found several times.
3. Revise $\mathcal{G}$ into a unambiguous grammar $\mathcal{G}_0$.

4. *Prove* that $\mathcal{G}_0$ is semantically unambiguous, i.e. $\mathcal{G}_0(P,x)$ has no duplicates for all x. (One way to achieve this is to partially evaluate $\mathcal{G}_0$ and P into a context-free string grammar $\mathcal{G}_0^{cfg}$, whose syntactic unambiguity (in the formal language sense) can be shown by an ambiguity checker, and if positive, guarantees semantic unambiguity of $\mathcal{G}_0$.)
5. Compare co-optimal answers of $\mathcal{G}(A,x)$ and $\mathcal{G}_0(A,x)$. If their number is reduced to a tolerable degree, the underlying problem is well-defined.

Fallacies Although systematic, this is by no means an automated method. It requires creativity in all steps and may fail. (1) The given algorithm may be difficult to express in algebraic style. (2) A canonical string representation P may not be obvious and require some clever encoding. (3) Grammar $\mathcal{G}_0$ may be tricky to construct. In fact, if $\{\mathcal{G}(P,x) \mid \text{for all } x\}$ is an inherently ambiguous language, $\mathcal{G}_0^{cfg}$ does not exist. (4) It may be a debatable matter of expert judgement what is to be considerd a tolerable degree of co-optimality.

In the case of minisatellite alignment we observed that $\mathcal{G}_0(A,x)$ has about $10^{|x|}$ co-optimal solutions, and we conclude that the underlying model is ill-defined.

References

[1] R. Giegerich. Explaining and controlling ambiguity in dynamic programming. In *Proceedings of Combinatorial Pattern Matching*, volume 1848 of *Springer Lecture Notes in Computer Science*, pages 46–59. Springer, 2000.

[2] R. Giegerich, C. Meyer, and P. Steffen. A discipline of dynamic programming over sequence data. *Science of Computer Programming*, 51(3):215–263, 2004.

[3] B. Löwes and R. Giegerich. Avoiding ambiguity and assessing uniqueness in minisatellite alignment. In Beissbarth et al., editor, *German Conference on Bioinformatics*, OASIcs, 2013.

[4] G. Sauthoff, M. Möhl, S. Janssen, and R. Giegerich. Bellman's GAP - a language and compiler for dynamic programming in sequence analysis. *Bioinformatics*, 2013.

Les chemins auto-évitants pliés reliés au repliement des protéines

Christophe Guyeux

Mots-clefs : *Chemins auto-évitants ; Repliement des protéines ; Prédiction de conformations*

Commençons par rappeler la définition des chemins auto-évitants [4].

Definition 2 (Chemins auto-évitants) Soit $d \geqslant 1$. Un *chemin auto-évitant* à n pas, de $x \in \mathbb{Z}^d$ dans $y \in \mathbb{Z}^d$, est une fonction $w : [\![0, n]\!] \to \mathbb{Z}^d$ avec :

- $w(0) = x$ et $w(n) = y$,
- $|w(i+1) - w(i)| = 1$,
- $\forall i, j \in [\![0, n]\!]$, $i \neq j \Rightarrow w(i) \neq w(j)$.

Nous nous sommes aperçus que, parmi les méthodes bio-informatiques utilisées pour prédire la conformation 3D des protéines, certaines ne permettaient pas d'atteindre tous les chemins auto-évitants (CAE) quand d'autres le pouvaient. Ainsi, certains outils de prédiction sont dans l'incapacité de produire certaines conformations, quand d'autres atteignent toutes les formes possibles. Partant de ce constat, la question est de savoir ce qui a le plus de sens au niveau biologique. De manière annexe, la preuve de NP-complétude du problème de prédiction du repliement des protéines n'est pas valable dans tous les contextes.

Nous avons donc redécouvert et exploité un résultat de non ergodicité de certains types de transformation de chemins auto-évitants, en avons déduit une sous-famille de chemins auto-évitants pliés (CAEP) qui apparaît naturellement dans les outils de prédiction des formes de protéines, et avons produit de premiers résultats sur ces CAEP. Un des questionnements dirigeant notre recherche est d'estimer le ratio CAE/CAEP, afin de savoir si les outils de prédiction se focalisant sur les CAEP perdent ou non beaucoup de formes possibles pour les protéines.

L'objet de la présentation que je propose est de positionner ce problème qui nous est apparu, de faire le point sur les connaissances que l'on a obtenues sur les CAEP, de lister des problèmes ouverts, et de discuter des conséquences au niveau de la prédiction de la conformation 3D des protéines [1, 2]. Je montrerai notamment que les CAE nulle part dépliables sont en nombre infinis, et sont aussi grand que l'on veut. Qu'en-dessous d'une certaine taille, tous les CAE sont CAEP. Enfin, j'exhiberai le plus petit chemin auto-évitant nulle part dépliable actuellement connu.

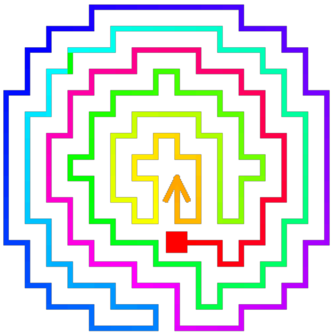


FIGURE 2 – Le premier CAE qui n'est pas CAEP (Madras et Sokal [3])

References

[1] J. Bahi, C. Guyeux, K. Mazouzi, and L. Philippe. Computational investigations of folded self-avoiding walks related to protein folding. *Journal of Bioinformatics and Computational Biology*, 2013. Accepted manuscript. To appear.

[2] C. Guyeux, N. M.-L. Côté, W. Bienia, and J. Bahi. Is protein folding problem really a NP-complete one ? first investigations. *Journal of Bioinformatics and Computational Biology*, 2013. Accepted manuscript. To appear.

[3] N. Madras and A. D. Sokal. The pivot algorithm : A highly efficient monte carlo method for the self-avoiding walk. *Journal of Statistical Physics*, 50 :109–186, 1988.

[4] N. N. Madras and G. Slade. *The self-avoiding walk*. Probability and its applications. Birkhäuser, Boston, 1993.

New software for mapping high-throughput reads for genomic and metagenomic data

Evguenia Kopylova, Laurent Noé, Mikaël Salson and Hélène Touzet

LIFL, UMR8022 CNRS Université Lille 1, and INRIA Lille Nord Europe, France

Context

The arrival of high-throughput sequencing technologies has introduced new problems for read mapping. The main challenges involve efficient processing of large amounts of sequenced read data and delivering robust algorithms generic to different sequencing technologies and their characteristic error types. The nature of the reads to map depends on the sequencing technology. Today, Illumina, 454 and Ion Torrent produce read lengths of 100-1000 bp with the lowest error rates in one round of sequencing, whilst the single-molecule sequencing platform, PacBio (Pacific BioScience), can produce average read lengths of 4600 bp but with a much higher error rate than for other technologies (nearly 15%, mostly indels). Furthermore, new applications such as community metagenomics and metatranscriptomics require sensitive algorithms capable of aligning low-complexity regions and distantly related species.

Read mapping problem

Without applying heuristics, algorithms which identify both substitution and indel errors are computationally expensive for large quantities of data. Although heuristics can effectively speed up the algorithm, many existing alignment tools such as BWA-SW [4], SOAP2 [5] and Bowtie2 [3] (all based on the Burrows-Wheeler-Transform (BWT) [1]) have been optimized for genome resequencing (∼99% sequence similarity) and impose error-free or substitution-only 'seeding' techniques for identifying short homologs prior to extending an alignment using dynamic programming. In the last decade, the application of sequencing technologies has been extended to metagenomics, that is to DNA extracted directly from an environmental sample. Raw samples of microbial organisms can be easily sequenced in parallel and this new culture-independent practice allows for unanimous study of all genomes recovered from an environmental community. For this type of application, where the reference sequences may share ∼75-98% similarity to the query, the aforementioned tools are no longer sensitive enough. Morover, sequencing technologies such as Ion Torrent, 454 and PacBio introduce artifacts into the reads in the form of indel errors. In these contexts, *approximate seeds* allowing mismatch and indel errors anywhere in the seed would serve as an optimal choice (but at some computational cost) and few tools exist today that efficiently implement them.

Methods

In this talk we present a new software called SortMeDNA which can map reads generated by second- and third-generation technologies from genomic and metagenomic studies. SortMeDNA uses approximate seeds (based on the universal Levenshtein automaton [7, 6]) allowing up to 1 error: The error can either be a mismatch or an indel, and its position in the seed is not predetermined. This unique feature gives the seed flexibility for different error types, such as indels in 454 reads, unpredictable error distribution, as readily observed with PacBio reads and capturing similarities between distantly related species. Furthermore, we introduce a new arborescent indexing data structure based on a lookup table and the Burst trie [2], which is specifically tailored to perform fast queries in large texts using this approximate seed. Lastly, SortMeDNA also applies statistical analysis to evaluate the significance of an alignment, as this becomes important to consider when aligning distantly related species.

Experimental results

The performance of SortMeDNA was evaluated against a representative selection of read mappers: Bowtie2, BWA-SW and SHRiMP2. Our tests took into consideration sequencing data produced by Illumina, 454, Ion Torrent and PacBio technologies, and dealt with a wide variety of applications from genomic to metagenomic projects. SortMeDNA has shown to be the most robust tool of the set, quickly and accurately mapping all types of reads using one set of default parameters. However, the tradeoff for maintaining such seeds is the new text indexing data structure which is able to accomodate searches with insertions and deletions, requiring more space than the BWT. We implement an index fragmentation technique which nonetheless allows convenient utilization of the tool.

References

[1] M. Burrows and D. Wheeler. A block-sorting lossless data compression algorithm. Technical report, 1994.

[2] S. Heinz, J. Zobel, and H. E. Williams. Burst tries: A fast, efficient data structure for string keys. *ACM Transactions on Information Systems*, 20:192–223, 2002.

[3] B. Langmead and S. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nat Methods*, 9:357–359, 2012.

[4] H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler Transform. *Bioinformatics*, 25:1754–60, 2009.

[5] R. Li, C. Yu, and Y. Li. SOAP2: an improved ultrafast tool for short read alignment. *Bioinformatics*, 25:1966–1967, 2009.

[6] S. Mihov and K. Schulz. Fast approximate search in large dictionaries. *J. Comput. Ling.*, 30:451–477, 2004.

[7] K. Schulz and S. Mihov. Fast string correction with Levenshtein automata. *IJDAR*, 5:67–85, 2002.

SUMATRA and SUMACLUST: fast and exact comparison and clustering of sequences

Céline Mercier, Frédéric Boyer, Aurélie Bonin and Éric Coissac

LABORATOIRE D'ECOLOGIE ALPINE, UMR5553 CNRS, LECA BP 53, 2233 Rue de la Piscine, 38041 Grenoble, Cedex 9, France

Next-generation sequencing (NGS) has undergone impressive developments within the last decade [11]. Today, an experiment can produce several millions of sequences, and efficient tools are needed to handle these volumes of data in reasonable amounts of time. The development of NGS has found numerous applications in the assessment and description of all forms of genetic diversity, from species, to populations, to individuals [1, 3]. In particular, DNA metabarcoding now allows high-throughput monitoring of biodiversity without requiring the collection/identification of specima in the field [13, 12]. This approach is therefore widely used in environmental microbiology and is becoming fairly popular for many ecological studies involving biodiversity assessment. DNA metabarcoding relies on the extraction of DNA from environmental samples (soil or water samples for example). Once the DNA is extracted, short DNA fragments called markers or barcodes (by analogy with DNA barcoding [7]) are amplified by PCR and sequenced. The tools used to treat the several millions of sequences produced by a DNA metabarcoding experiment have to be efficient, but also adapted to the type of data produced by DNA metabarcoding, i.e. entirely sequenced and short markers.

Classification methods are a key aspect in the analysis of DNA metabarcoding data. Sequences can be assigned to taxa by comparing them to a reference database containing barcodes of described taxa with a supervised classification method. However, such reference databases are not always available or exhaustive (typically for microorganisms). In that case, unsupervised classification enables to cluster highly similar sequences into groups called Molecular Operational Taxonomic Units (MOTUs) [2] which become the unit of measurement for biodiversity. The choice of an adaptated clustering procedure implies to think about the reasons leading to the necessity of the clustering [8]. Indeed, depending on the similarity measure and on the clustering algorithm used, the result of this classification procedure can be highly variable, both in terms of number of clusters and cluster composition.

Here, we present SUMACLUST and SUMATRA, a package of two programs which aim to compare sequences in a way that is fast and exact at the same time, unlike the most popular clustering methods that usually rely on fast heuristics, such as UCLUST [5] or CD-HIT [6]. SUMACLUST and SUMATRA are devoted to the type of data generated by DNA metabarcoding, i.e. entirely sequenced, short markers. There are two components for a clustering process. The first one is the computation of the pairwise similarities between sequences, and the second one is the clustering itself. SUMATRA performs the first step, the computation of the pairwise similarities. The output can then go through a classification process with programs such as MCL [4] or MOTHUR [10]. SUMACLUST performs both the similarities computation and the clustering, using the same clustering algorithm as UCLUST and CD-HIT.

Four elements in particular make SUMACLUST and SUMATRA interesting for handling DNA metabarcoding data:

Clustering algorithm. When representing the similarities between sequences as a graph, with sequences as vertices and similarities as edges, erroneous sequences and the true sequences from which they were generated during the PCR or sequencing steps appear as star-shaped clusters. The 'true' sequence are then the centres of the 'stars', and all the erroneous variants are linked to the centre from which they derive. This form of clustering corresponds to the one produced by the clustering algorithm used by SUMACLUST, UCLUST and CD-HIT, which makes them well-adapted to finding erroneous sequences.

Order of the sequences. Since the clustering procedure implemented in these tools is greedy, cluster composition is highly dependent on the order of the sequences. Clustering sequences sorted by length will lead to cluster centres corresponding to the longest sequences, whereas clustering sequences sorted by count will lead to most abundant sequences as cluster centres. This justifies the fact that SUMACLUST sorts sequences by count, because 'true' sequences should be more abundant than erroneous sequences and should become the centre of their clusters.

Similarity indice. UCLUST and CD-HIT perform semiglobal alignments. Semiglobal alignments aim to find the best possible alignment that includes the whole length of one of the sequences, in their case, the shortest sequence. This method was interesting in the first works of bacterial DNA metarcoding, when barcodes were long and generally not entirely sequenced. Nowadays, barcodes are generally entirely sequenced. Moreover, the size polymorphisms of barcodes are part of the signal allowing to differentiate the studied taxa. They consequently have to be aligned on their whole lengths, with global alignment algorithms.

Speed and exactitude. SUMATRA and SUMACLUST are implemented using a banded alignment algorithm [9], very efficient when high thresholds are used as it is often the case in DNA metabarcoding. A lossless k-mer filter enables to align only the pairs of sequences that potentially present an identity greater than the chosen threshold. Besides, the filter and alignment steps are both parallelized with the use of Simple Instruction Multiple Data instructions, allowing to present execution times similar to those of the most popular programs based on heuristics, while staying exact.

In conclusion, SUMATRA and SUMACLUST combine the speed of heuristic methods with the accuracy of exact methods, and their characteristics make them particularly well adapted to DNA metabarcoding data.

References

[1] H. M. Bik, D. L. Porazinska, S. Creer, J. G. Caporaso, R. Knight, and W. K. Thomas. Sequencing our way towards understanding global eukaryotic biodiversity. *Trends Ecol Evol*, 27(4):233–43, Apr 2012.

[2] M. Blaxter, J. Mann, T. Chapman, F. Thomas, C. Whitton, R. Floyd, and E. Abebe. Defining operational taxonomic units using DNA barcode data. *Philos Trans R Soc Lond B Biol Sci*, 360(1462):1935–43, Oct 2005.

[3] P. I. Diaz, A. K. Dupuy, L. Abusleme, B. Reese, C. Obergfell, L. Choquette, A. Dongari-Bagtzoglou, D. E. Peterson, E. Terzi, and L. D. Strausbaugh. Using high throughput sequencing to explore the biodiversity in oral bacterial communities. *Mol Oral Microbiol*, 27(3):182–201, Jun 2012.

[4] S. V. Dongen. *Graph clustering by flow simulation*. PhD thesis, University of Utrecht, Utrecht, The Netherlands, 2000.

[5] R. C. Edgar. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, 26(19):2460–1, Oct 2010.

[6] L. Fu, B. Niu, Z. Zhu, S. Wu, and W. Li. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–2, Dec 2012.

[7] P. D. N. Hebert, A. Cywinska, S. L. Ball, and J. R. deWaard. Biological identifications through DNA barcodes. *Proc Biol Sci*, 270(1512):313–21, Feb 2003.

[8] S. M. Huse, D. M. Welch, H. G. Morrison, and M. L. Sogin. Ironing out the wrinkles in the rare biosphere through improved OTU clustering. *Environ Microbiol*, 12(7):1889–98, Jul 2010.

[9] J. B. Kruskal. An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM review*, 25(2):201–237, 1983.

[10] P. D. Schloss, D. Gevers, and S. L. Westcott. Reducing the effects of PCR amplification and sequencing artifacts on 16s rRNA-based studies. *PLoS One*, 6(12):e27310, 2011.

[11] J. Shendure and H. Ji. Next-generation DNA sequencing. *Nat Biotechnol*, 26(10):1135–45, Oct 2008.

[12] P. Taberlet, E. Coissac, F. Pompanon, C. Brochmann, and E. Willerslev. Towards next-generation biodiversity assessment using DNA metabarcoding. *Mol Ecol*, 21(8):2045–50, Apr 2012.

[13] A. Valentini, F. Pompanon, and P. Taberlet. DNA barcoding for ecologists. *Trends Ecol Evol*, 24(2):110–7, Feb 2009.

Nouveaux Algorithmes d'Extraction de Motifs et de Pondération pour le Classement de Protéines

Faouzi Mhamdi, Mehdi Kchouk and Salma Aouled El Haj Mohamed

Laboratoire de Recherche en Technologies de l'Information et de la Communication & Génie Electrique, ESTT- Université de Tunis, Tunisie

Faouzi Mhamdi : Faouzi.mhamdi@ensi.rnu.tn
Mehdi Kchouk : mehdi.kchouk@gmail.com
Salma Aouled El Haj Mohamed : aouledelhaj.salma@gmail.com

Résumé : Dans leur structure primaire, les données biologiques sont représentées sous forme d'une suite de caractéres. Le but de ce travail est de classer automatiquement les séquences de protéines. Pour ce faire, il semblait judicieux d'utiliser un processus très connu dans la fouille des données : c'est le processus de l'ECD (Extraction des Connaissances à partir des Données). Nous nous intéressons à la première phase de l'ECD celle de prétraitement et nous nous concentrons à la tâche d'extraction de motifs. L'extraction de motifs se traduit par la génération d'un ensemble de descripteurs que l'on présente aux algorithmes d'apprentissage supervisé pour faire la classification. La méthode d'extraction la plus connue est le n-gramme, qui correspond à une suite de caractères de longueur n. Les algorithmes des n-grammes consiste à fixer *a priori* la valeur de n, extraire des n-grammes, et à travailler avec cette valeur tout au long du processus de classement de protéines. Dans ce papier, nous proposons un algorithme de construction de n-grammes afin d'obtenir des descripteurs de taille variable ensuite on propose une nouvelle de pondération basée sur la programmation dynamique. Les performances de ces nouvelles propositions sont évaluées par le taux d'erreur obtenu avec le classifieur SVM linéaire. Les expérimentations sur des données biologiques réelles donnent des bons résultats en les comparant avec des antèrieurs.

Mots-clefs : *ECD, Extraction d'attributs, n-gramme, Pondération d'attributs, Classement des séquences de protéines, SVM, Données biologiques*

1 Algorithme d'extraction de motifs

Notre algorithme est un algorithme d'extraction de motifs adoptant une démarche hiérarchique « descendante ». Cette dernière, consiste à construire des n-grammes (motifs) de taille variable. D'une manière générale, hiérarchiquement, on extrait les $n - i$ grammes à partir des n-grammes tant que $n - i \geq 2$. Nous utilisons le terme descendant car nous commençons par l'extraction des motifs de taille n (avec n donné par l'utilisateur), puis comme deuxième étape, nous extrairons les motifs de taille $n - 1$ et nous répétons la procédure et à chaque étape nous diminuons la taille des motifs extraits de $n = n - 1$ jusqu'à $n = 2$. Par exemple : si $n = 5$ l'algorithme commence à extraire les motifs de taille 5 puis 4, 3 enfin, 2. L'algorithme extrait hiérarchiquement des descripteurs á partir des séquences de protéines regroupées en familles et nous tentons de prouver par cette idée que nous pouvons augmenter la taille des n-grammes afin de mélanger le plus possible de descripteurs afin d'obtenir de meilleurs résultats.

2 Méthode de pondération

Afin de réaliser notre pondération, nous construirons une matrice de pondération dont les lignes présentent les séquences protéiques de deux familles de protéines différentes. Quant aux colonnes, elles contiendront les attributs extraits à partir de la technique de n-grammes. L'intersection séquence/attribut n'est autre qu'un appel de la fonction qui calcule le score d'alignement S/W que nous avons adapté à nos besoins. Ce score est

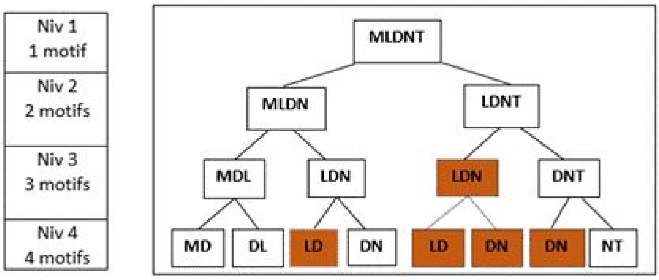
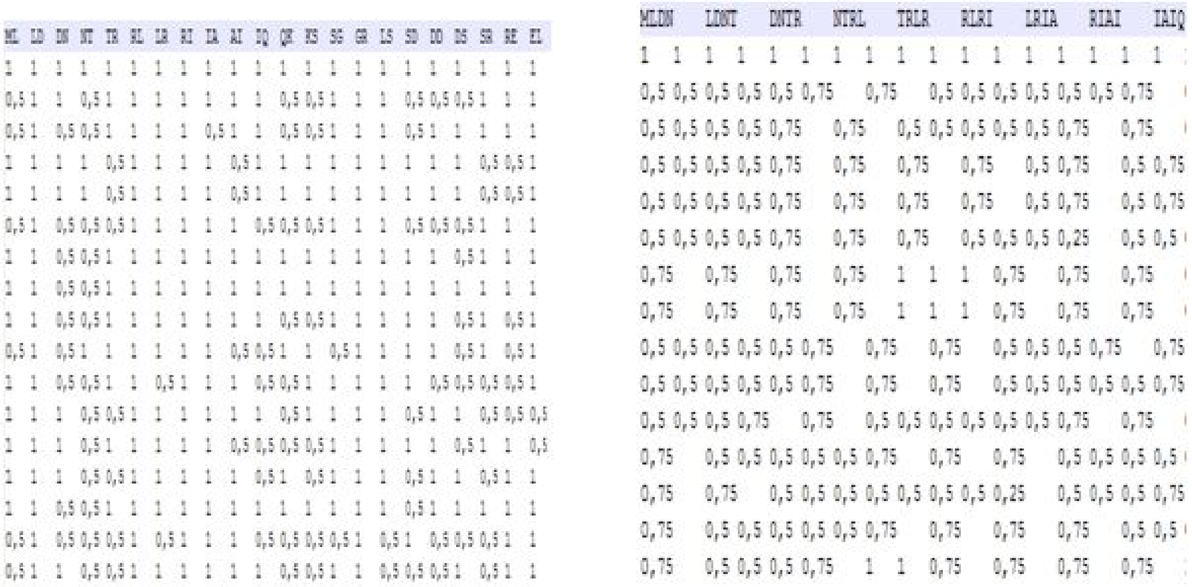


FIGURE 1 – *Processus descendant de construction des descripteurs.*

exprimé en pourcentatge et détermine si l'attribut est existant dans la séquence en entier ou uniquement une partie de l'attribut y existe. Bien évidemment, si l'attribut existe en entier, son score sera alors égal à 100% et plus le nombre de caractères inexistants augmente, plus le score diminue. Dans notre cas les scores seront compris entre 0 et 1. La dernière colonne de la matrice désignera la famille d'appartenance de la séquence. Cette méthode nous permettra d'être plus précis quant au classement des séquences et leur affectation à la famille adéquate. Afin de démontrer l'efficacité de cette pondération, nous devons nous comparer aux autres types de pondération existants (la pondération Booléenne, la pondération par Occurrence, la pondération par Fréquence et la pondération par TF*IDF).



ML	LD	DN	NT	TR	RL	LR	RI	IA	AI	IQ	QR	RS	SG	GR	LS	SD	DD	DS	SR	RE	EL
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0,5	1	1	0,5	1	1	1	1	1	1	1	0,5	0,5	1	1	1	0,5	0,5	0,5	1	1	1
0,5	1	0,5	0,5	1	1	1	1	0,5	1	1	0,5	0,5	1	1	1	0,5	1	1	1	1	1
1	1	1	1	0,5	1	1	1	1	0,5	1	1	1	1	1	1	1	1	1	0,5	0,5	1
1	1	1	1	0,5	1	1	1	1	0,5	1	1	1	1	1	1	1	1	1	0,5	0,5	1
0,5	1	0,5	0,5	0,5	1	1	1	1	1	0,5	0,5	0,5	1	1	1	0,5	0,5	0,5	1	1	1
1	1	0,5	0,5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0,5	1	1	1
1	1	0,5	0,5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0,5	0,5	1	1	1	1	1	1	1	0,5	0,5	1	1	1	1	1	0,5	1	0,5	1
0,5	1	0,5	1	1	1	1	1	1	0,5	0,5	1	1	0,5	1	1	1	1	0,5	1	0,5	1
1	1	0,5	0,5	1	1	0,5	1	1	1	0,5	0,5	1	1	1	1	1	0,5	0,5	0,5	0,5	1
1	1	1	0,5	0,5	1	1	1	1	1	1	0,5	1	1	1	1	0,5	1	1	0,5	0,5	0,5
1	1	1	0,5	1	1	1	1	1	0,5	0,5	0,5	0,5	1	1	1	1	1	0,5	1	1	0,5
1	1	1	0,5	0,5	1	1	1	1	1	0,5	1	0,5	1	1	1	0,5	1	1	0,5	1	1
1	1	0,5	0,5	1	1	1	1	1	1	1	1	1	1	1	1	0,5	1	1	1	1	1
0,5	1	0,5	0,5	0,5	1	0,5	1	1	1	0,5	0,5	0,5	0,5	1	0,5	1	0,5	0,5	0,5	1	1
0,5	1	1	0,5	0,5	1	1	1	1	1	1	0,5	0,5	1	1	0,5	0,5	0,5	1	0,5	1	1

MLDN		LDNT		DNTR		NTRL		TRLR		RLRI		LRIA		RIAI		IAIQ
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0,5	0,5	0,5	0,5	0,5	0,75		0,75		0,5	0,5	0,5	0,5	0,5	0,5	0,75	
0,5	0,5	0,5	0,5	0,75		0,75		0,5	0,5	0,5	0,5	0,5	0,75		0,75	
0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,75		0,5	0,75		0,5	0,75
0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,75		0,5	0,75		0,5	0,75
0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,5	0,5	0,5	0,25		0,5	0,5
0,75		0,75		0,75		0,75		1	1	1	0,75		0,75		0,75	
0,75		0,75		0,75		0,75		1	1	1	0,75		0,75		0,75	
0,5	0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,5	0,5	0,5	0,75		0,75
0,5	0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,5	0,5	0,5	0,5	0,5	0,75
0,5	0,5	0,5	0,75		0,75		0,5	0,5	0,5	0,5	0,5	0,5	0,75		0,75	
0,75		0,5	0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,5	0,5	0,5	0,5
0,75		0,75		0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,25		0,5	0,5	0,5	0,75
0,75		0,5	0,5	0,5	0,5	0,5	0,75		0,75		0,75		0,75		0,5	0,5
0,75		0,5	0,5	0,5	0,75		1	1	0,75		0,75		0,75		0,75	

FIGURE 2 – Exemple de pondération basée sur l'alignement S/W avec 2 et 4-grammes

3 Conclusion

Nous avons présenté dans ce papier un nouvel algorithme d'extraction d'attributs à partir des séquences biologiques et une nouvelle méthode de pondération de ces attributs. L'objectif est d'améliorer le taux d'erreur de classification. Nous avons obtenu des bons résultats par rapport à des travaux antérieurs.

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. *Basic local alignment search tool.* J Mol Biol, 1990.

[2] C. Cortes and V. N. Vapnik. *Support Vector Networks.* Machine Learning J, 1995.

[3] M. Maddouri and M. Elloumi. Encoding of primary structures of biological macromolecules within a data mining perspective. *Journal of Computer Science and Technology*, pages 78–88, 2004.

[4] F. Mhamdi, R. Rakotomalala, and M. Elloumi. *Textmining, Features Selection and Datamining for Proteins Classification.* ICTTA, IEEE Catalog Number, Damascus, Syria, 2004.

[5] F. Mhamdi, R. Rakotomalala, and M. Elloumi. An hierarchical approach of n-grams extraction from proteins classification. *SITIS (Hammamet, Tunisia)*, 1, 2006.

[6] T. F. Smith and M. S. Waterman. In *Identification of common molecular subsequences*, pages 195–197. J Mol Biol, 1981.

Suffix tree and suffix array of an alignment

Joong Chae Na[1], Heejin Park[2], Sunho Lee[3], Minsung Hong[3], Thierry Lecroq[4], Laurent Mouchard[4] and Kunsoo Park[3]

[1] Sejong University, Korea
[2] Hanyang University, Korea
[3] Seoul National University, Korea
[4] Normandie Université, LITIS EA 4108, Université de Rouen, France

The huge amount of sequencing of individual genomes rises the problem of indexing highly similar sequences. Classical indexing structures such as generalized suffix trees or generalized suffix arrays do not offer satisfying solutions to this problem because they do not sufficiently exploit redundancies inside these data.

In this talk we present two new recently introduced data structures aimed at addressing the problem: the suffix tree of an alignment [1] and the suffix array of an alignment [2].

The alignment of two sequences $x = \alpha\beta\gamma$ and $y = \alpha\delta\gamma$ is denoted by $\alpha(\beta/\delta)\gamma$ where α and γ are respectively the longest prefix and the longest suffix common to x and y (β and δ are thus different). The suffix tree and the suffix array of an alignment of two sequences $x = \alpha\beta\gamma$ and $y = \alpha\delta\gamma$ enable to represent all the suffixes of x and y by storing only once the suffixes common to the two sequences.

The suffix tree of the alignment $\alpha(\beta/\delta)\gamma$ can be constructed in time $O(|x| + |\alpha^*| + |\delta| + |\hat{\gamma}|)$ where α^* is the longest suffix of α that occurs twice in x or y et $\hat{\gamma}$ is the longest prefix of γ such that $d\hat{\gamma}$ occurs twice in x and y (d is the symbol before γ).

The suffix array of the alignment $\alpha(\beta/\delta)\gamma$ can be constructed in time $O(|x| + |\alpha^*| + |\delta| + |\gamma^*|)$ where γ^* is the longest prefix of γ that occurs twice in x or y.

The two structures can be extended to several non-common regions and to more than two sequences.

Pattern matching in these two structures can be done with similar complexities than in classical structures.

References

[1] J. C. Na, H. Park, M. Crochemore, J. Holub, C. S. Iliopoulos, L. Mouchard, and K. Park. Suffix tree of an alignment: An efficient index for similar data. In T. Lecroq and L. Mouchard, editors, *Proceedings of the 24th International Workshop on Combinatorial Algorithms (IWOCA 2013)*, number 8288 in Lecture Notes in Computer Science, Rouen, France, 2013. Springer-Verlag, Berlin. To appear.

[2] J. C. Na, H. Park, S. Lee, M. Hong, T. Lecroq, L. Mouchard, and K. Park. Suffix array of alignment: A practical index for similar data. In O. Kurland, M. Lewenstein, and E. Porat, editors, *Proceedings of the 20th International Symposium on String Processing and Information Retrieval (SPIRE 2013)*, number 8214 in Lecture Notes in Computer Science, pages 243–254, Jerusalem, Israel, 2013. Springer-Verlag, Berlin.

Comparison of sets of homologous gene transcripts

Aïda Ouangraoua[1], Krister M. Swenson[2] and Anne Bergeron[3]

[1] INRIA Lille, LIFL, Université Lille 1, Villeneuve d'Ascq, France
[2] Université de Montréal and McGill University, Canada
[3] LaCIM, Université du Québec à Montréal, Montréal, Canada

Aïda Ouangraoua: aida.ouangraoua@inria
Krister M. Swenson: swenson@lirmm.fr [1]
Anne Bergeron: bergeron.anne@uqam.ca

One of the most intriguing and powerful discoveries of the post-genomic era is the revelation of the extent of alternative splicing in eukaryote genomes, where a single gene sequence can produce a multitude of transcripts [2, 3]. The "one gene, one protein" dogma of the last century has been shattered into pieces, and these pieces tell a story in which genome sequences acquire new function not only by mutation, but by being processed differently.

Most studies on the analysis of gene transcript variants are based on cataloging splicing events between pairs of transcripts. For instance, in a recent paper [5], an analysis was carried on hundreds of transcripts from over three hundred genes in human, mouse and other genomes, yielding dozens of conserved or species-specific splicing events. The results are given as combined statistics by species or group of species, and cataloged as one of 68 different kinds of splicing events.

However, beyond recognizing that two transcripts are conserved between species, or that a specific alternative splicing event is conserved, there is no formal setting for the comparison of two or more sets of transcripts that are transcribed from homologous genes of different genomes. The most widely used approach is to resort to comparing all pairs of transcripts within a set, or between two sets (see [11] for a review).

There are several hurdles on the way to a good representation. The first comes from the fact that, when alternate transcripts were scarce, much of the focus was directed towards the representation of alternative splicing events: splicing graphs [4] or pictograms [1] are adequate but do not scale easily to genes that can have dozens of transcripts, or to comparison between multiple species. Other representation techniques, such as bit matrices and codes (see [6, 9] and references therein), proposed for the identification and the categorization of alternative splicing events are often more appropriate for computers than for human beings. A second problem is the identification of the features to compare. The splicing machinery is entangled with a myriad of bits and pieces that can vary within and between species: transcripts, coding sequences, exons, introns, splicing donor and acceptor sites, start and stop codons, untranslated regions of arbitrary lengths, frame shifts, etc. Ideally, a model would capture as much as is known about transcripts, including the underlying sequences. In that direction, the goal of the Exalign method [8, 10] is to integrate the exon-intron structure of transcripts with gene comparison, in order to find "splicing orthology" for pairs of transcripts. What about integrating the whole structure of orthologous sets of transcripts in gene comparison, and the discovery of homologous genes?

Here we propose a switch from the paradigm of comparing single transcripts between species, to comparing all transcripts from a global perspective, rather than focussing on specific splicing events. We describe a representation of sets of transcripts that is straightforward, readable by both humans and computers, and that can incorporate the various mechanisms that drive transcript evolution (see [7] for a detailed description of the model). This representation yields very flexible tools to compare sets of transcripts. It serves as a powerful representation for the reconstruction of evolution histories of sets of transcripts, and for the evaluation of the structural similarity between potentially homologous genes. On the other hand, the model has a precise, formal specification that insures its coherence, consistency and scalability. We show several applications, among them a comparison of 24 Smox gene transcripts across five species.

[1] current affiliation: Institut de Biologie Computationnelle, Montpellier, France

References

[1] D. Bollina, B. Lee, T. Tan, and S. Ranganathan. ASGS: an alternative splicing graph web service. *Nucleic Acids Res.*, 34:W444–447, Jul 2006.

[2] P. Carninci, T. Kasukawa, S. Katayama, and . others. The transcriptional landscape of the mammalian genome. *Science*, 309:1559–1563, Sep 2005.

[3] T. E. P. Consortium. Identification and analysis of functional elements in 1% of the human genome by the encode pilot project. *Nature*, 447:799–816, 2007.

[4] S. Heber, M. Alekseyev, S. Sze, H. Tang, and P. Pevzner. Splicing graphs and EST assembly problem. *Bioinformatics*, 18 Suppl 1:S181–188, 2002.

[5] J. Mudge, A. Frankish, J. Fernandez-Banet, T. Alioto, T. Derrien, C. Howald, A. Reymond, R. Guigo, T. Hubbard, and J. Harrow. The origins, evolution and functional potential of alternative splicing in vertebrates. *Molecular biology and evolution*, 28:2949–2959, Oct 2011.

[6] H. Nagasaki, M. Arita, T. Nishizawa, M. Suwa, and O. Gotoh. Automated classification of alternative splicing and transcriptional initiation and construction of visual database of classified patterns. *Bioinformatics*, 22(10):1211–1216, 2006.

[7] A. Ouangraoua, K. M. Swenson, and A. Bergeron. On the comparison of sets of alternative transcripts. In *ISBRA, LNCS 7292*, pages 201–212, 2012.

[8] G. Pavesi, F. Zambelli, C. Caggese, and G. Pesole. Exalign: a new method for comparative analysis of exon-intron gene structures. *Nucleic Acids Res*, 36:e47, May 2008.

[9] M. Sammeth, S. Foissac, and R. Guigo. A general definition and nomenclature for alternative splicing events. *PLoS Computational Biology*, 8:e1000147, 2008.

[10] F. Zambelli, G. Pavesi, C. Gissi, D. Horner, and G. Pesole. Assessment of orthologous splicing isoforms in human and mouse orthologous genes. *BMC Genomics*, 11:534, 2010.

[11] M. Zavolan and E. van Nimwegen. The types and prevalence of alternative splice forms. *Curr. Opin. Struct. Biol.*, 16:362–367, Jun 2006.

Disentangling homeologous contigs in tetraploid assembly: application to durum wheat

Vincent Ranwez[1*], Yan Holtz[2], Gautier Sarah[2], Morgane Ardisson[2], Sylvain Santoni[2], Sylvain Glémin[3], Muriel Tavaud-Pirra[1] and Jacques David[1]

[1] Montpellier SupAgro, UMR AGAP, F-34060 Montpellier, France
[2] INRA, UMR AGAP, F-34060 Montpellier, France
[3] Institut des Sciences de l'Evolution de Montpellier (ISE-M), UMR 5554 CNRS Université Montpellier II, place E. Bataillon, CC 064, 34 095 Montpellier cedex 05, France

[*] Corresponding author

Vincent Ranwez : ranwez@supagro.inra.fr
Yan Holtz : holtz@supagro.inra.fr
Gautier Sarah : gautier.sarah@cirad.fr
Morgane Ardisson : Morgane.Ardisson@supagro.inra.fr
Sylvain Santoni : Sylvain.Santoni@supagro.inra.fr
Sylvain Glémin : sylvain.glemin@univ-montp2.fr
Muriel Tavaud-Pirra : Muriel.Tavaud@supagro.inra.fr
Jacques David : Jacques.David@supagro.inra.fr

Contexte

En utilisant les outils de séquençage haut débit, la détection de SNP est devenue relativement routinière pour les espèces diploïdes. Elle reste cependant problématique concernant les espèces polyploïdes, notamment suite aux confusions entre locus homéologues qui peuvent être assemblés de manière erronée en un seul contig. Nous proposons une méthode permettant de séparer efficacement de tels contigs chimériques en deux contigs homologues sur la base du différentiel d'expression de ces deux copies. Le logiciel HomeoSplitter, qui implémente cette solution, permet de gérer efficacement ces problèmes de mélange d'homéologues à l'aide d'une approche par maximum de vraisemblance.

Nous avons validé HomeoSplitter sur des données RNAseq réelles issues de trente accessions de blé dur (*Triticum turgidum*, tétraploïde contenant les génomes A et B, 2n=4x=28). Les transcriptomes des espèces diploïdes donneuses des génomes élémentaires, *Aegilops speltoides* (proche du génome B) et *Triticum urartu* (proche du génome A) ont été utilisés comme élément de comparaison afin de valider la méthode.

Les millions de reads ont été assemblés par assemblage *de-novo* et les contigs obtenus clusterisés. Les 2505 clusters contenant des séquences homologues de blé dur, de *urartu* et de *speltoides*, ont constitué un jeu de test permettant de confirmer l'apport d'HomeoSplitter. HomeoSplitter permet, sur ces données, d'obtenir des contigs de blé dur plus proches de ceux de ses ancêtres diploïdes. Le mapping des reads sur ces nouveaux contigs, plutôt que directement sur ceux issus de l'assemblage *de novo*, permet de multiplier par 4 le nombre de SNP fiables identifiés (762 SNP parmi 1360 sites polymorphes au lieu de 188 parmi 1620).

Le programme HomeoSplitter est disponible gratuitement à l'adresse http://davem-193/homeoSplitter/. Cet outil constitue une solution pratique résolvant les problèmes de mélange des homéo-génomes pour les espèces allo-tétraploïdes, et permet une détection des SNP plus performante chez ces espèces.

Ce travail a été accepté pour publication dans [1].

References

[1] V. Ranwez, Y. Holtz, G. Sarah, M. Ardisson, S. Santoni, S. Glémin, and M. Tavaud-Pirra. *BMC Bioinformatics*, 14(Suppl 15) :S15. (RECOMB-CG 2013 special issue). Accepted.

An algorithm for pattern occurrences P-values computation

Mireille Régnier[1*], Evgenia Furletova[2*], Victor Yakovlev[2,4] and Mikhail Roytberg[2,3,4*]

[1] INRIA team AMIB, LIX-Ecole Polytechnique and LRI-UPSud, 1 rue d'Estienne d'Orves, 91 120 Palaiseau, France
[2] Institute of Mathematical Problems of Biology, 142290, Institutskaya, 4, Pushchino, Russia
[3] Laboratoire J.-V. Poncelet (UMI 2615), 119002, Bolshoy Vlasyevskiy Pereulok, 11, Moscow, Russia
[4] National Research University "Higher School of Economics", 101978, Myasnitskaya str., 20, Moscow, Russia

[*] corresponding author

Mireille Régnier: mireille.regnier@inria.fr
Evgenia Furletova: furletova@lpm.org.ru
Victor Yakovlev: v.yacovlev@gmail.com
Mikhail Roytberg: mroytberg@lpm.org.ru

Our study is related to finding of new functional fragments in biological sequences, that is an important problem in bioinformatics. Methods addressing this problem commonly search for clusters of pattern occurrences that are statistically significant. A measure of statistical significance is the P-value of the number of pattern occurrences, i.e. the probability to find at least S occurrences of words from a pattern $\mathcal{H}$ in a random text of length N generated according to a given probability model. All words of the pattern are supposed to be of a same length.

We present a novel algorithm SufPref computing an exact P-value for three types of probability models: Bernoulli, Markov models of arbitrary order and Hidden Markov models (HMMs). The algorithm computes P-value as the probability of the set of all sequences containing at least S occurrences of $\mathcal{H}$. We described the structure of such set using auxiliary sets of sequences and obtained equations for probabilities of auxiliary sets. The equations allow efficient processing of overlaps between pattern words, that is the main advantage of our algorithm. The information about overlaps and their relations is stored in a specific data structure, the overlap graph. The nodes of the graph are associated with the overlaps of words from $\mathcal{H}$. Edges are associated to the prefix and suffix relations between overlaps. An originality of our data structure is that pattern $\mathcal{H}$ need not be explicitly represented in nodes or leaves. The algorithm inductively traverses the graph. The algorithm relies on the Cartesian product of the overlap graph and the graph of model states; the approach is analogous to the automaton approach from [2]. We carefully analyze the structure of the Cartesian product, e.g. the reachability of vertices; this leads to extra improvement of time and space complexity. Taking into account overlaps between the pattern words significantly decreases space and time complexities. Remark that the nodes of the extensively used structure Aho-Corasick trie, used in particular by the algorithm AhoPro [1], are associated with prefixes of pattern words. The number of prefixes is much larger than the number of overlaps.

The algorithm SufPref was implemented as a C++ program; it can be used both as Web-server and a stand alone program for Linux and Windows; the program is available at http://server2.lpm.org.ru/bio/online/sf/.

The implementation of the algorithm SufPref was compared with program AhoPro for Bernoulli model and first order Markov model. The comparison shows that, for all considered cases, our algorithm is faster than AhoPro in more than four times for Bernoulli models and in more than two times for Markov model. In a vast majority of cases, it outperforms AhoPro in space.

References

[1] V. Boeva, J. Clément, M. Régnier, M. Roytberg, and V. Makeev. Exact p-value calculation for heterotypic clusters of regulatory motifs and its application in computational annotation of cis-regulatory modules. *Algorithms for Molecular Biology*, 2(13):25 pages, 2007. [http://www.almob.org/content/2/1/13].

[2] G. Kucherov, L. Noé, and M. Roytberg. A unifying framework for seed sensitivity and its application to subset seeds. *Journal of Bioinformatics and Computational Biology*, 4(2):553–569, 2009.

Filtering systematic errors in next-generation genotype calls: stranding matters

Laure Sambourg and Nicolas Thierry-Mieg

UJF-Grenoble 1 / CNRS / TIMC-IMAG UMR 5525, Computational and Mathematical Biology (BCM), Grenoble, F-38041, France

Background

Next generation sequencing technologies have enabled the production of massive datasets, opening up new avenues of research. In particular, exome capture and sequencing provides a cost-effective means of genotyping the coding portion of the genome for large cohorts of individuals. However calling genotypes from NGS data remains difficult, and high rates of discrepancies between called genotypes are observed when using different sequencing technologies on the same samples, or different software pipelines on the same raw sequencing data. Some of these difficulties may be due to systematic biases and errors arising at the sequencing, alignment or genotype-calling steps. In order to investigate these possibilities, we analyzed 108 deep exome-seq replicates of non-tumor cells produced by The Cancer Genome Atlas with ABI SOLiD and Illumina GAII platforms, searching for sources of systematic errors.

Results

We aligned the TCGA short-reads using the MAGIC pipeline, which allowed us to distinguish reads aligning on the forward and reverse strands of the genome, and developed a simple and effective algorithm for calling genotypes. This algorithm favors specificity over sensitivity: calls are only made when the data is unambiguous and sufficiently deep. Furthermore, we called genotypes independently on each strand and compared the resulting calls. Surprisingly, we discovered that they disagree in 40.6% of the positions where high-confidence calls can be made on both strands (excluding homozygous reference positions). This observation is not an artifact of the MAGIC aligner, as shown by reanalyzing a published dataset initially studied with GATK. These stranddiscordant positions appear due to the sequence context, which is strand-specific and can lead to systematic sequencing errors on one strand. Furthermore, the TCGA replicates allowed us to identify systematic error-prone positions in the genome, some of which are specific to the ABI or Illumina sequencers and some of which are cross-platform. Filtering these positions significantly improves the genotyping quality.

Conclusions

Our results clearly show that strand-specific read counts should always be provided, and that a reliable genotype can only be called when the two strands are compatible and sufficiently covered. In addition, lists of error-prone positions are provided and should help to filter out systematic errors. Beyond genotype-calling, our findings have implications on the experimental design of exomecapture experiments: capture libraries should be short enough to allow a significant proportion of positions to be sequenced on both strands.

Reconstructing Textual Documents from perfect n-gram Information

Matias Tealdi and Matthias Gallé

Xerox Research Centre Europe

Companies may be interested in releasing part of the data they own for reasons of general good, prestige, harnessing the work of those the data is released to or because it opens access to new sources (in a marketplace setting for instance). However, most of the times it is not possible to release the complete data due to privacy concerns, legal constraints or economic interest. A compromise is to release some statistics computed over this data. In the case of releasing n-gram counts of text documents (the case we study here), two examples are the release of copyrighted material (notably the Google Ngram Corpus) and the exchange of phrase tables for machine translation when the original parallel corpora are private or confidential.

The obvious question that then arises is how much of the information should be released in order to avoid reconstruction from a third party. We analyse here the question of what are the longest blocks that can be reconstructed with total certainty (this is, not considering probabilistic approaches) starting from an n-gram corpus containing *perfect* information (no n-gram or count is omitted, and no noise is introduced).

A similar problem is solved routinely in DNA sequencing mapping the n-grams into a graph (the *de Bruijn* graph) and finding an Eulerian tour in this graph [1]. However, the number of different Eulerian tours can grow worse than exponentially with the number of nodes, and only one of these tours corresponds to the original document. We present a novel reduction of this graph into its most irreducible form, from which large blocks of substrings of the document can easily be read off. In our experiments on books from the Project Gutenberg [1] we were able to obtain blocks of an average size of 53.21 words starting from 5-grams.

1 Definitions

We will be working with directed multigraphs where an edge not only has a multiplicity attached to it, but also a label denoting the substring it represents. This motivates our following definition of graph:

Definition 3 A graph G is a tuple (V, E), with V the set of nodes and E the set of edges, where each edge is of the form $(\langle u, v, \ell \rangle, k)$ with $u, v \in V$; $\ell \in \Sigma^*, k \in \mathcal{N}$; where Σ is the vocabulary of the original sequence.

Given an edge $e = (\langle v, w, \ell \rangle, k)$ we use the following terms to refer to its components: $tail(e) = v$, $head(e) = w$, $label(e) = \ell$, $multiplicity(e) = k$.

The *indegree* of a node v, $d_{in}(v)$ is $\sum_{e \in E: head(e)=v}$ $multiplicity(e)$; and the *outdegree* $d_{out}(v)$ is $\sum_{e \in E: tail(e)=v}$ $multiplicity(e)$.

A graph is *Eulerian* if it is connected and $d_{in}(v) = d_{out}(v)$ for all nodes v. In this case we define $d(v) = d_{in}(v) = d_{out}(v)$.

We furthermore require that the labels determine the edges uniquely. This is, $\forall e_1, e_2 \in E$, if $label(e_1) = label(e_2)$, then $e_1 = e_2$.

An Eulerian cycle in our graphs is then a cycle that visits each edge e exactly $multiplicity(e)$ times. We denote the set of all Eulerian cycles of G with $ec(G)$. Given an Eulerian cycle $c = e_1, \ldots, e_n$, its label sequence is the list $\ell(c) = [label(e_1), \ldots, label(e_n)]$, and the string it represents is the concatenation of these labels: $s(c) = label(e_1).label(e_2). \ldots .label(e_n)$.

Remember that our original problem was to find substrings as long as possible of which we are sure that they appear in the original sequence, given the evidence of the n-grams. Our strategy is to start with the original de Bruijn graph, and to merge some edges iteratively until the final edges correspond exactly to those maximal strings. Formally, given the original graph G, we are interested in a graph G^* that:

[1] http://www.gutenberg.org/

1. is equivalent:

$$\{s(c) : c \in ec(G)\} = \{s(c) : c \in ec(G^*)\} \tag{2}$$

2. is irreducible:

$$\nexists e_1, e_2 \in E^* : [label(e_1), label(e_2)] \textit{ appears in } \text{all } \ell(c), c \in ec(G^*) \tag{3}$$

2 Reduction Steps

To achieve this we will introduce two reduction rules which preserve Eulerian cycles (correctness) and whose successive application ensures an irreducible Eulerian graph (completeness).

The first one – which in practice is applied the most – takes only into consideration local information by analyzing incoming and outgoing edges of a specific vertex. The second one uses global information, through the use of *division points*, which are nodes that are either articulation point [2] or have a self-loop edge.

References

[1] P. E. C. Compeau, P. A. Pevzner, and G. Tesler. How to apply de Bruijn graphs to genome assembly. *Nature biotechnology*, 29(11):987–91, Nov. 2011.

[2] R. Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, 1(2):146–160, June 1972.