Search space reduction for sequence and structure alignments

Christian Otto

Bioinformatics, Leipzig

February 2009

Table of content

Overview

Aim of the project:

Create a filtering method for sequence and structure alignments of RNAs especially with large data sets, e.g. whole genomes.
Thus, it should report only regions with good sequence and structure similarity and thereby, reduce the search space for expensive alignment techniques, e.g. `FOLDALIGN` [4].

Overview

Aim of the project:

Create a filtering method for sequence and structure alignments of RNAs especially with large data sets, e.g. whole genomes.
Thus, it should report only regions with good sequence and structure similarity and thereby, reduce the search space for expensive alignment techniques, e.g. `FOLDALIGN` [4].

Basic ideas:

- only use query regions containing short, local structure motifs for the search
- search is done using fast string matching on sequence and structure rather than alignments

Steps within project

1. preprocessing: encoding of query and genome to include known or predicted structural information
2. search: fast approximate pattern matching on sequence and structure using an enhanced suffix array
3. chaining: chaining of matches at different locations within the query using some gap costs
4. reporting the best chains

Encoding and Cleaning

- **encoding:** converts multiple strings of different alphabets to one string of a combined alphabet (required for suffix array)
- each of those strings represents a feature level and contains specific information, e.g. nucleotide or structural information

Encoding and Cleaning

- **encoding:** converts multiple strings of different alphabets to one string of a combined alphabet (required for suffix array)
- each of those strings represents a feature level and contains specific information, e.g. nucleotide or structural information
- **cleaning:** mark of regions without interest for search
- it can be used in combination with mapping to expose local structure motifs

Encoding and Cleaning

- **encoding:** converts multiple strings of different alphabets to one string of a combined alphabet (required for suffix array)
- each of those strings represents a feature level and contains specific information, e.g. nucleotide or structural information
- **cleaning:** mark of regions without interest for search
- it can be used in combination with mapping to expose local structure motifs
- both operations has to be done in advance for query as well as for the genomic sequence

Example for encoding

Encoding of tRNA-95 in *Homo sapiens* on chr6 [3]:

```
GGGGAATTAGCTCAAGCGGTAGAGCGCTCCCTTAGCATGCGAGAGGTAGCGGGATCGACGCCCCCATTCTCTA   sequence string
(((((((..((((........)))).(((.(........).))).....(.(((........))).)))))))).   structure string
```

↓ cleaning structure

```
GGGGAATTAGCTCAAGCGGTAGAGCGCTCCCTTAGCATGCGAGAGGTAGCGGGATCGACGCCCCCATTCTCTA   sequence string
*********((((........))))*(((.(........).)))*****(.(((........))).)********   cleaned structure string
```

} feature levels

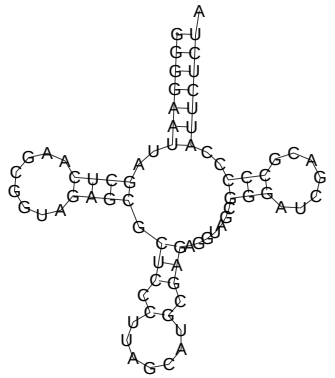
↓ encoding

```
LLLLDDPPDIEMECCKGKKOCJBJFLEMEGEOOCKGCOJGJBJDLLPDIGIIICOGKCGKFFFGFDPPHPHPD   encoded string
```

encoding matrix:

	(	)	.	*
A	A	B	C	D
C	E	F	G	H
G	I	J	K	L
U	M	N	O	P
N	Q	R	S	T

structure of tRNA-95:



Introduction to search

- **mapping:** defines how matching during the search will be done: two characters of the encoded string match on a feature level if the mapped character on this level is the same

Introduction to search

- **mapping:** defines how matching during the search will be done: two characters of the encoded string match on a feature level if the mapped character on this level is the same
- mapped character '0' expresses no feature at this level
  $\Rightarrow$ only regions in the query containing features at each level will be included in the search $\Rightarrow$ possiblity to split up the query into short regions for the search

Introduction to search

- **mapping:** defines how matching during the search will be done: two characters of the encoded string match on a feature level if the mapped character on this level is the same
- mapped character '0' expresses no feature at this level
  $\Rightarrow$ only regions in the query containing features at each level will be included in the search $\Rightarrow$ possiblity to split up the query into short regions for the search

possible mappings:

encoded char	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
mapping 1	A	A	A	A	C	C	C	C	G	G	G	G	U	U	U	U	N	N	N	N
	(	)	.	0	(	)	.	0	(	)	.	0	(	)	.	0	(	)	.	0
mapping 2	A	A	A	A	C	C	C	C	G	G	G	G	U	U	U	U	N	N	N	N
	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0

Search and extending

- the enhanced suffix array, introduced by Abouelhoda et al., allows exact pattern matching in linear time according to query length [1]
  $\Rightarrow$ can be extended to allow errors
- implemented search mode: $\vec{k}$-difference search with at most k_i errors (mismatches, insertions and deletions) at level i

Search and extending

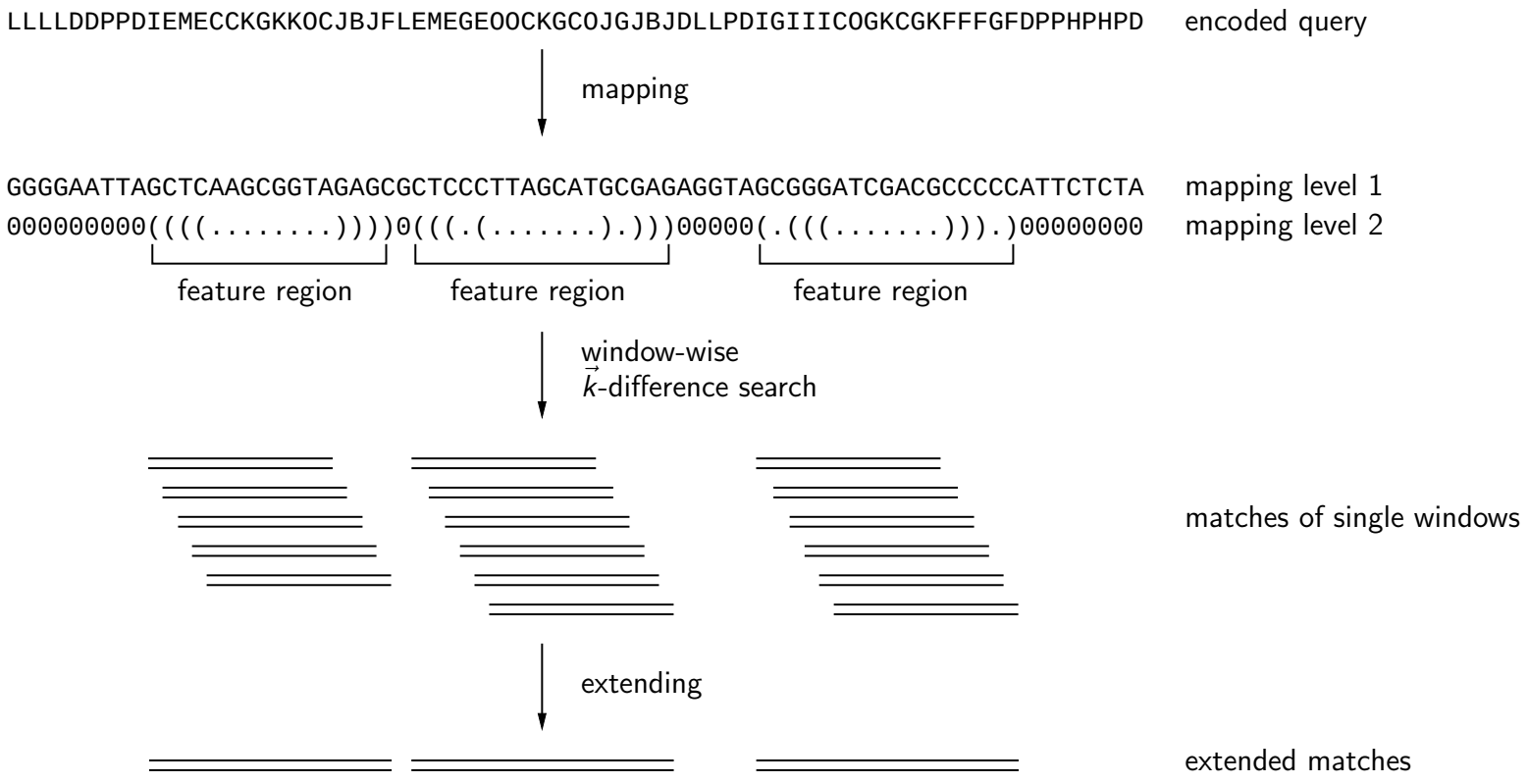
- the enhanced suffix array, introduced by Abouelhoda et al., allows exact pattern matching in linear time according to query length [1]
  $\Rightarrow$ can be extended to allow errors
- implemented search mode: $\vec{k}$-difference search with at most k_i errors (mismatches, insertions and deletions) at level i
- sliding window search to avoid the problem of adjusting $\vec{k}$ to different lengths of search regions

Search and extending

- the enhanced suffix array, introduced by Abouelhoda et al., allows exact pattern matching in linear time according to query length [1]
  $\Rightarrow$ can be extended to allow errors
- implemented search mode: $\vec{k}$-difference search with at most k_i errors (mismatches, insertions and deletions) at level i
- sliding window search to avoid the problem of adjusting $\vec{k}$ to different lengths of search regions
- extending: extend matches of single windows by merging matches overlapping on genomic sequence and query
  $\Rightarrow$ reduces number of matches for chaining without loosing any crucial information
- possible improvements later on: use of matching statistics, bitvector algorithm and so on ($\nearrow$ Steve's talk)

Example for mapping, search and extending

Search for tRNA-95 of *Homo sapiens*:



Introduction to chaining

- **chaining:** connects fragments or matches to longer chains using defined gap costs
- sum-of-pair gap costs, defined by Myers and Miller [5], are used which allow punishing gaps between two fragments in a flexible way

Introduction to chaining

- **chaining:** connects fragments or matches to longer chains using defined gap costs
- sum-of-pair gap costs, defined by Myers and Miller [5], are used which allow punishing gaps between two fragments in a flexible way

Sum-of-pair gap costs:

Sum-of-pair gap costs between two fragments f' and f are defined as:

$$g(f', f) = \begin{cases} \lambda \cdot \Delta_x(f', f) + (\epsilon - \lambda) \cdot \Delta_y(f', f) & \text{if } \Delta_x(f', f) \geq \Delta_y(f', f) \\ \lambda \cdot \Delta_y(f', f) + (\epsilon - \lambda) \cdot \Delta_x(f', f) & \text{if } \Delta_y(f', f) \geq \Delta_x(f', f) \end{cases}$$

with $\Delta_x(f', f) = |\text{sequence start of f'} - \text{sequence end of f}|$

and $\Delta_y(f', f) = |\text{query start of f'} - \text{query end of f}|$

Chaining algorithm

Overview about algorithm:

- introduced by Abouelhoda et al. [2]
- can chain only non-overlapping fragments

Chaining algorithm

Overview about algorithm:

- introduced by Abouelhoda et al. [2]
- can chain only non-overlapping fragments
- requires data structure for 2-dimensional RMQ
  $\Rightarrow$ use of 2-dimensional range tree enhanced with vanEmdeBoas trees at the lowest dimension
- algorithm takes $O(n \log n \log \log n)$ in time and $O(n \log n)$ in space using these range trees

Chaining algorithm

Overview about algorithm:

- introduced by Abouelhoda et al. [2]
- can chain only non-overlapping fragments
- requires data structure for 2-dimensional RMQ
  $\Rightarrow$ use of 2-dimensional range tree enhanced with vanEmdeBoas trees at the lowest dimension
- algorithm takes $O(n \log n \log \log n)$ in time and $O(n \log n)$ in space using these range trees
- local chaining used: chain fragment/chain with another fragment only if score of resulting chain is greater or equal than the score of previous one
- reports best chains

Concluding remarks

Main aspects:

- different types of information about each position can be handled simultaneously
- fast pattern matching, can be restricted to defined regions
- reducing the number of matches by extending/merging
- connecting nearby matches by chaining

Concluding remarks

Main aspects:

- different types of information about each position can be handled simultaneously
- fast pattern matching, can be restricted to defined regions
- reducing the number of matches by extending/merging
- connecting nearby matches by chaining

$\Rightarrow$ Thus, it could be suitable for other types of applications as well.

References


Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch.
Replacing suffix trees with enhanced suffix arrays.
J. of Discrete Algorithms, 2(1):53–86, 2004.


Mohamed Ibrahim Abouelhoda and Enno Ohlebusch.
Multiple genome alignment: Chaining algorithms revisited.
In *Combinatorial Pattern Matching: 14th Annual Symposium, CPM 2003, Morelia, Michoacn, Mexico, June 25-27, 2003. Proceedings*, volume 2676/2003 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2003.


Patricia P. Chan and Todd M. Lowe.
GtRNAdb: a database of transfer RNA genes detected in genomic sequence.
Nucl. Acids Res., 37:D93–97, 2009.


Jakob H H. Havgaard, Elfar Torarinsson, and Jan Gorodkin.
Fast pairwise structural rna alignments by pruning of the dynamical programming matrix.
PLoS Comput Biol, 3(10), October 2007.


Gene Myers and Webb Miller.
Chaining multiple-alignment fragments in sub-quadratic time.
In *SODA '95: Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 38–47, Philadelphia, PA, USA, 1995. Society for Industrial and Applied Mathematics.

The end

Thank you for listening!

Feel free to ask some questions.