Bioinformatics - Lecture 03

Bioinformatics

Genomes: assembly, sequences, genes

Martin Saturka

http://www.bioplexity.org/lectures/

EBI version 0.4

Creative Commons Attribution-Share Alike 2.5 License

Genomes

Genome assembly and standard initial genome processing.

- genome assembly and subsequent standard tasks
- sizes: H. sapiens: $3 \cdot 10^9$ bp, D. melanogaster: $1.3 \cdot 10^8$ bp, S. cerevisiae: $2 \cdot 10^7$ bp, E. coli: $4 \cdot 10^6$ bp, Phage λ: $5 \cdot 10^4$ bp

Standard tasks

- fragment assembly
  - standard graph algorithms used
  - advanced methods and repetitions
- exact matching
  - sequence localizations
  - start of heuristic algorithms
- gene finding
  - markers for gene recognition
  - probability methods, comparisons, HMMs

Chromosomes

pairs of antiparallel complementary double-helix strands

ATGC alphabet, 64 possible 3-tuples, 3 of them nonsense ones

- nucleotides:
  - pentose with 3 hydroxyl groups
  - base (attached to 1’ hydroxyl)
  - (mono/di/tri) phosphate (on 5’ hydroxyl)
  - one free 3’ hydroxyl group
  - orientation 5’ $\longrightarrow$ 3’
- bases: adenine, thymine, guanine, cytosine (uracil)
- repetitions (tens of percents of eukaryotic genomes)
  - usually not sequenced

Sequencing methods

main sequencing types

methods:

- enzymatic (polymerization)
- chemical (degradation)
- complementarity hybridization
- new methods - solid phase

enzymatic:

- dideoxyribuncleotides
- pyrosequencing
- bead parallelized

Standard sequencing

shotgun method - libraries of small fragments sequencing

- polymerase chain reaction
- particular ddNTP addition
- fluorescence detection

- sequencing data formats, chromatograms
  - SCF - binary data of fluorescence peaks
- prepared sequences data formats
  - fasta, annotated (GenBank, EMBL, etc.)

```
>Sequence 1
TGAGTAGCGCCATACGTGCTGACTGCATGCATGACTAGTACGTCAGCTAGCTCGGTAGAT
GTAGTAGGCATGCGCCGCGATATCGTAGCATATTAGCGATTTTTAGTAGCTGCATGACTA
...
>Sequence 2
...
```

Fragment assembly

- concatenation of sequenced fragments into contigs
    - fragments cca 500 bp
- overlap-layout-consensus
    - Hamilton paths
    - NP-complete problem
    - standard method used
    - hard to use for repetitions
- overlap graph methods
    - transformation into Euler paths
    - vertex & edge unification
    - error corrections 'inside'
    - used for some bacteria
- branching approach
    - clustering fragments into sequence similarity group
    - first assembly inside groups into larger fragments
    - second assembly the larger fragments into contigs

Overlap-Layout-Consensus

currently the method being used

assembly of all the fragments into a continuous superstring

- shortest superstring:
  - vertices - sequence fragments
  - edges - (maximal) fragment overlaps
- overlap: overlaps of the fragments
- layout: larger contig construction
- consensus: polymorphism abandoning

- error prone, computationally intensive, layout and consensus by multiple checking
  - easy to implement, computer clusters available

Graph paths

- Hamiltonian paths
  - visit each vertex exactly once
  - NP-complete problem
- Eulerian paths
  - visit each edge exactly once
  - linear problem
  - directed graphs: for balanced ones
  - undirected graphs: even degrees for all (but two) vertices
- Eulerian algorithm
  - start (arbitrary) available path
  - augment current path, when finished:
    - all edges used - augment the old path with the new path
    - some edges free - use them for a new path start and augmenting

Overlap sequencing

Sequencing by hybridization

an array of all the l-mers

- hybridization of a fragment on the array
- concatenation of detected l-mers
- not a suitable practical method
    - pradigm for gene expression and SNP arrays!

- overlap graph approach
    - concatenation of the l-mers
    - Hamiltonian path problem
- subsequences approach
    - concatenation of sub $l - 1$-mers
    - Eulerian path problem

Eulerian path approach

usage of inner vertex and edge structures

- both verices and edges are sequences
- de Bruijn graphs
  - super-graphs where edges are vertices of the old graph
  - construction of the repeat and de Bruijn graphs with many obstacles
- HBS like approach
  - short k-mers made out of the sequenced fragments
  - good for genomes with false repetions
  - used for some hard assemble bacterial genomes
    - *N. meningitidis*
  - not used for human genome: large, real repetitions

DNA publishing

- data available
  - data state
  - sequenced regions
  - masked regions
  - tagged sequences
  - polymorphic sequences
  - gene sites
- masking coordinates
  - various repetition classes - not suitable for search
- STSs - sequence-tagged sites
  - unique chromosomal sequences (200 - 500 bp)
  - ESTs - expressed sequence tags
    - similar, but from cDNA sequences, i.e. from mRNA

Polymorphism

highly polymorphic sites

- SNPs [snips]
    - single nucleotide polymorphism
    - human: every 100-300 bp (2/3 of them: C $\rightarrow$ T)
    - cca 90% of human genetic variation
    - TSC - The SNP Consortium
        - DNA of 24 individuals, and more
- forensic usage
    - SNPs, some repetitive sequences
- GATTACA phenomemon
    - genetical ’brave new world’

Sequence search

sequence localization

- exact matching
  - where a sequence is exactly located
  - standard algorithms available
  - linear time search
- approximate matching
  - allowed (similar) mismatches
  - with or without insertions, deletions
  - 'fuzzy' definition of similarity

Exact matching

where to use the exact matching

usage:

- location of a given sequence
    - STS set localization
    - faster than on a whole genom

- multiple search
    - restriction (palindromic) sites
        - `G|AATTC` for EcoRI
    - start of heuristic algorithms
        - motif search, comparison

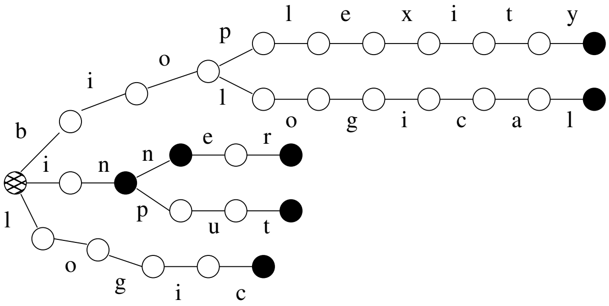
Algorithms

- Boyer-Moore
  - standard algorithm for single word search
  - simple kind of constraint programming
- Aho-Corasick
  - standard algorithm for multiple word search
  - efficient trie dictionary usage
- other algorithms
  - Rabin-Karp
    - uses hashing
    - plagiarism detection

Tries

re'trie'val structures - dictionary storage

- biological, bioplexity, in, inn, inner, input, logic



Aho-Corasick algorithm

- automaton creation - trie, three functions
    - match case transition
        - *goto* function - trivial
        - always set for the initial node
    - fail case transition
        - *fail* function
        - construction through a queue
    - *output* function
        - initial values - trivial
        - update through a queue
- automaton search
    - reads character by character
    - match case: pass through the *goto* function
    - fail case: pass through the *fail* function
    - write by the *output* function

Automaton creation

- make the initial trie with *goto* and initial *output* functions
- first cycle on the initial trie node:
    - take all the considered symbols
    - if there is a way out of the initial symbol by the symbol
        - put the entered new node on the top of the queue
        - set the *fail* (on the new node) to lead to the initial node
- then cycle in breadth first manner:
    - while is not the queue empty, take off first node and for all the considered symbols, and if *goto* leads somewhere
        - put the newly enetered node on the top of the queue
        - take *fail* node of the current node and
          while there is no *goto* way on the *fail* node
          take one more backward *fail* node (of the *fail* node)
        - set the *fail* on the current node as *goto*
          of the finally found *fail* node and the current symbol
        - add to *output* of the current node *output* of the
          newly found *fail* on the current node

Automaton search

search steps:

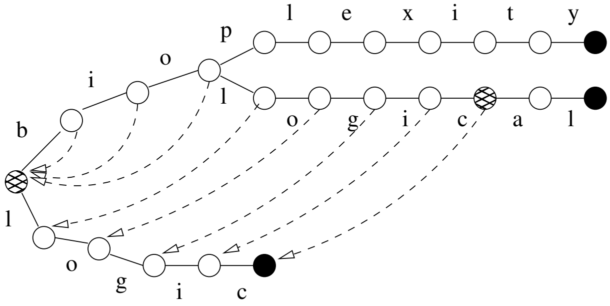
- start in the initial node
- read the searched string charcter by character
- while (if) is no output of *goto* on a current node and the read character
  - set the new node as *fail* result on the current node
- pass through the *goto* function
  - it is always defined on the initial node
    - at least as a stand by
- write *output* if is something to write

Algorithm complexity

- linear complexity $O(m+n)$
  - m size of the searched (target) string
  - n total size of the set of the searched for strings
- automaton creation
  - trie construction - simply linear
  - queue cycle:
    - linear in number of nodes
    - *fail* construction reuses *fail* on previous nodes thus it goes fast backward
- automaton search
  - total amount of *fail* transitions is bounded by the total amount of (matched) *goto* transitions
  - total amount of *goto* transitions is bounded by the length of the searched string

Automaton example

- dashed arrows for the relevant part of the *fail* function
- the *output* on the node of position of ’biologic’ is ’logic’



Generalizations

usage of Aho-Corasick algorithm for more complex situations

- wild card matching with bounded amount of the wild cards
    - search for non wild card parts
    - checks for appropriate distances between occurences
- limited amount of mismatches on any position
    - enlarging the set of the strings searched for according to the possible mismatches

Gene prediction

howto find unknown genes inside sequences

genomes do not contain explicit information on gene locations

- human genome: cca 30 thousand genes
- search for putative genes
  - ORF open reading frame - can be translated
- gene finding
  - de novo - according to gene characteristics
  - search for transcribed genes
  - comparisons - with usage of known genes
- gene markers
  - for gene expression purposes
  - promoters, transcription factors binnding sites, etc.
  - cells themselves have to find genes somehow

Prokaryotic gene markers

transcription

- promoter sequences
  - Pribnow box (-10 location)
    `T(77%) A(76%) T(60%) A(61%) A(56%) T(82%)`
  - TTGACA sequence (-35, 17 nt off the Pribnow box)
    `T(69%) T(79%) G(61%) A(56%) C(54%) A(54%)`
  - other specific (SOS box, etc.) promoters
- systematics of transcription factor binding sites
- termination harpin loops - palindromes

translation

- Shine-Dalgarno sequence `AGGAGG`
  - upstream of the first coding `AUG`

Eukaryotic gene markers

transcription

- (methylation) CpG islands
- cis regulatory elements
  - TATA box, is not necessary

mRNA maturation

- polyadenylation (AAUAAA sequence)
- splicing marks
  - donor site - 5’ of an intron, acceptor site - 3’ of an intron
  - intron: `GU` (donor site) - `AG` (acceptor site)
    - for vast most of introns
  - intron: branch site (20-50 bp upstream of acceptor site) `CU(A/G)A(C/U)`
    - the middle A is conserved
  - exons: `(A/C)AG` (donor site), `G` (acceptor site)
    - cca 60% of exon/intron borders

translation

- Kozak consensus sequence `(A/G)CCACC`
  - upstream (-1 to -15) of the first coding `AUG`

Gene cotents

- triplet frequences
    - biased triplet frequenes inside genes
- tri-nucleotide auto-correlations
    - slight 3-repetition signal inside genes
- stop-codons
    - outside genes 3 of 64 triplets should be a stop codon every cca 60-75 bp inframe
- entropy measure
    - GC content and conditional probabilities biased for gene rich/poor sites for specific species

HMM search

- combination of the known signals and content probabilities
- probabilities of inside-a-gene along a given sequence
    - selectoin of regions with high (smoothened) in-gene probabilities, usage of hierarchized HMMs

- Prokaryotes
    - known systematics on (strong) signals
    - works fairly well

- Eukaryotes
    - weak, various, poorly known signals
    - mediocre results

Comparison search

every species similar to every species

- to search sequences related to known genes
  - many organisms have (partially) sequenced genomes
  - approximate search of various gene sequences
- comparative genomics
  - successful for yeast species
  - under hard work for the human genome

Extrinsic gene prediction

Protein and mRNA based approaches

- mRNA detection
  - relatively simple task - polyA tails detection
- ESTs, cDNA libraries
  - many mRNAs reversely transcribed into cDNAs
- protein sequences
  - genetic code usage for coding sequences prediction

Non-coding sequences

not only protein codin genes

- repetitions and ncRNA search

- long time known RNAs: rRNA, tRNA
- non-coding RNAs with substantial regulatory functions
- structural RNA motifs
  - UNCG and GNRA tetraloops, uridine turns, CTAG tetramers

- some repetitive sequences probably functional too

Binding sites

- regulatory motifs (cca 4-10 bp)
  - usually in near upstream sequences
  - sometimes in near downstream sequences
  - many in far upstream sequences
- regulatory canonical sequence search
  - a short sequence present in most given sequences
  - the least total amount of mismatches on a subset
- median string search
  - rename ATGC into 0123 numbers
  - a table of all the 8-mers is about 1 MB large
  - an improvement of the search
    - stop a local comparison if initial (terminal) part too distant

Items to remember

Nota bene:

gene structure, marker types

- Sequence localization
  - STS, EST
  - Aho-Corasick algorithm

- Gene sequencing
  - Overlap-Layout-Consensus
  - Euler, Hamilton paths