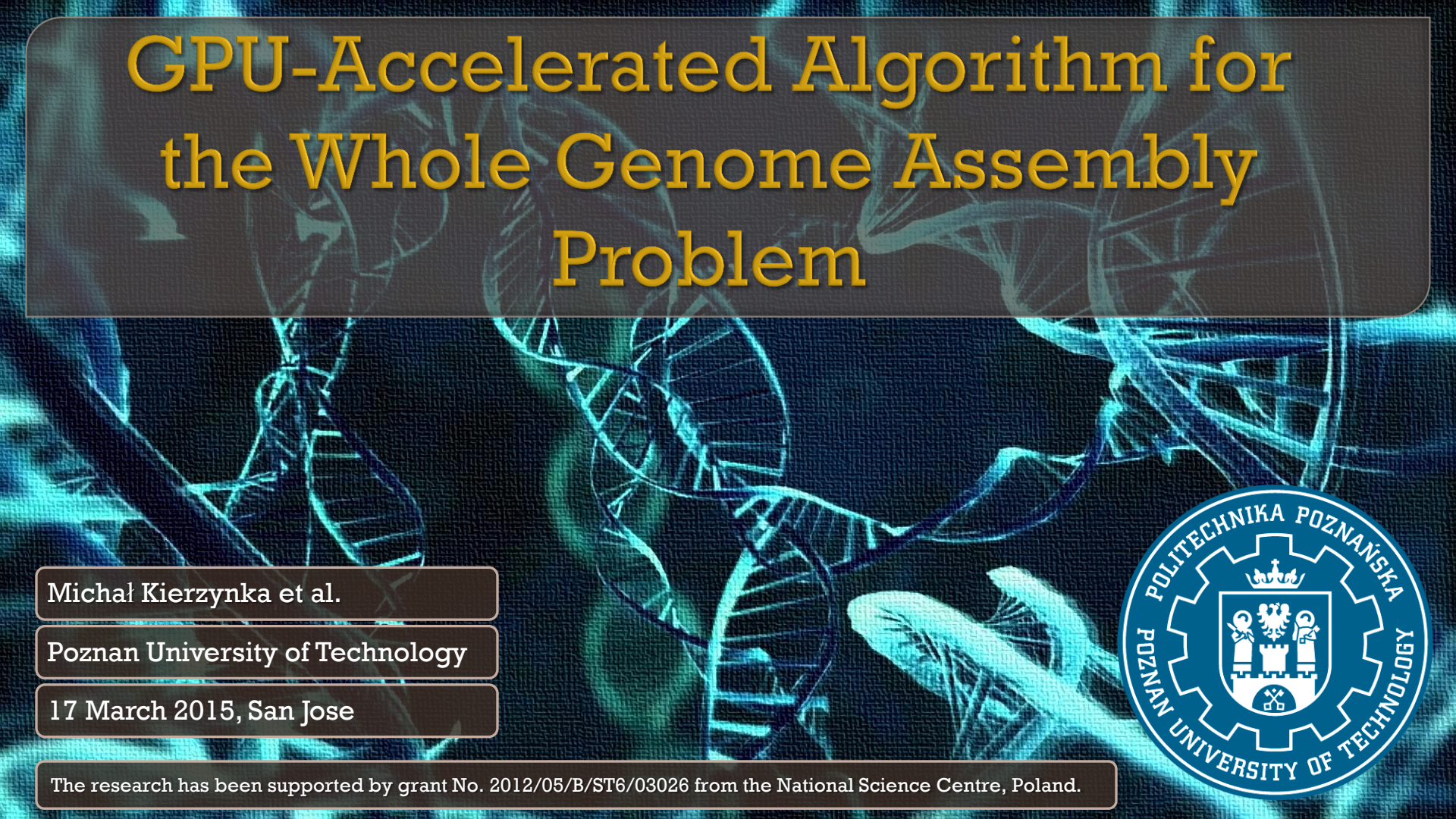
GPU-Accelerated Algorithm for the Whole Genome Assembly Problem



Michał Kierzynka et al.

Poznan University of Technology

17 March 2015, San Jose

The research has been supported by grant No. 2012/05/B/ST6/03026 from the National Science Centre, Poland.

DNA *de novo* assembly

- input: short reads (35-150bp)
- output: contigs (assembled parts of a genome)

Illumina Genome Analyzer II sequencer

AGCA ATCAAGCAAC
GACTC TAGAA TTTGCC

TTAGCACAGGAACTCTA
TTTGC-C GA-CTC
AGCA TTCTA
ATCA-AGCAAC

DNA *de novo* assembly

Input sequences:

- a multiset of overlapping reads over alphabet {A, C, G, T}
- may contain misreadings/errors
- come from both strands of the DNA double helix
- reverse complement sequences

Problems:

- large data sets: millions of reads (e.g. ~300GB for *homo sapiens*)
- exact algorithms are exponential
- quality of heuristics is often limited

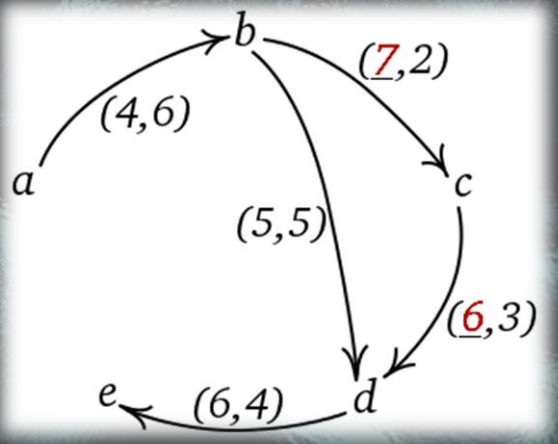
DNA *de novo* assembly

DNA overlap graph:

- each read represented by a vertex
- overlapping sequences connected by an arc
- weights, e.g. corresponding alignment scores
- result: a Hamiltonian path for each connected component

Selection of overlapping sequences!

consensus sequence:	**AGATGTATTATTATTCTACGAGTGG**
sequence a:	**AGATGTATTA**
sequence b:	**ATTATTATTC**
sequence c:	**TATTGTTCTA**
sequence d:	**TATTCTACGA**
sequence e:	**CTACGAGTGG**



DNA overlap graph construction

Selection of overlapping sequences:

- not feasible to compare every sequence with each other $O(n^2)$
- *promising pairs* - pairs of sequences that are likely to overlap
  - fast preselection of *promising pairs*
  - overlaps verification (greatly increases precision)

```
ACGGGTA
   GGGTACT
```

score 5, overlap 2

```
CTGGAGT
  TGGAGTCC
```

score 6, overlap 1

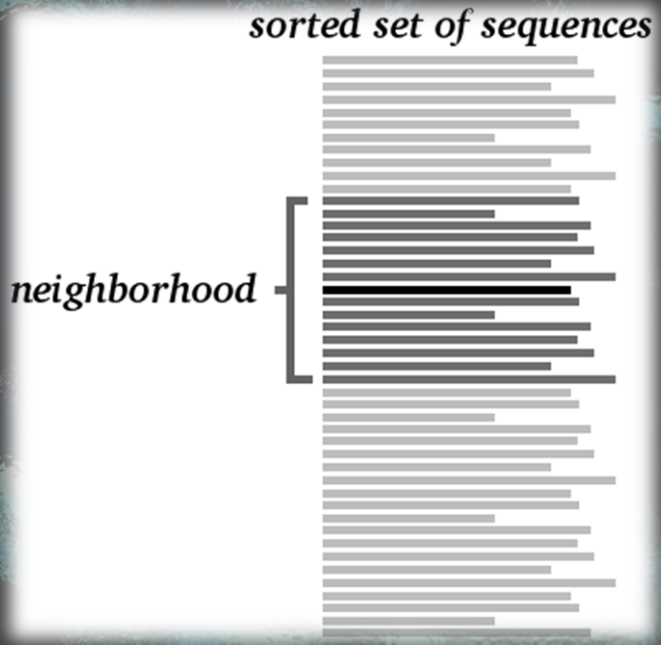
```
CTGGAGT
CTGAACCG
```

score 1, overlap 0

DNA overlap graph construction

DNA overlap graph:

- sort sequences in the way that similar sequences are close to each other
  O(n log n)
- verify which of the neighbouring sequences are really similar using exact sequence comparison



How to sort sequences properly?

DNA overlap graph construction

k-mer – a substring of k consecutive nucleotides from a sequence

For each sequence the algorithm computes its k-mer characteristic:

1) extracts every possible k-mer (k is fixed)
2) sorts k-mers descending on their frequencies of occurrence

GAACGAACTGAA

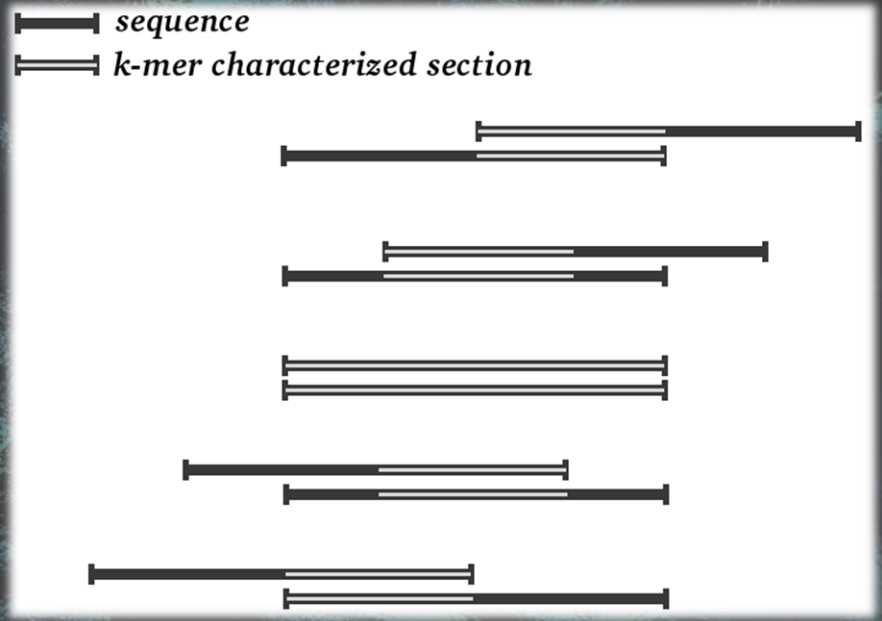
1) K=3: 2xAAC, ACG, ACT, CGA, CTG, 3xGAA, TGA
2) 3xGAA, 2xAAC, ACG, ACT, CGA, CTG, TGA

Finally, sort all the sequences alphabetically according to their characteristics (similar to a dictionary).

DNA overlap graph construction

Partial k-mer characteristics:

- a set of short characteristics computed for each sequence
- purpose: to detect also the pairs with short overlaps

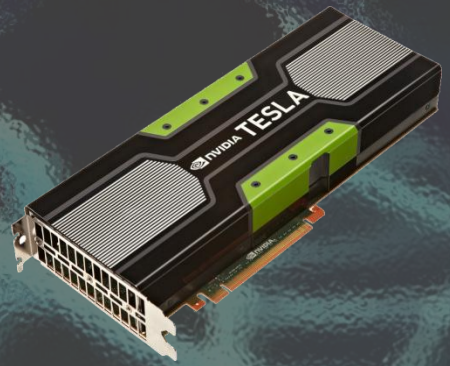


DNA overlap graph construction

Neighborhood verification by sequence alignment:

- computationally heavy (Needleman-Wunsch)
- no solution on the market
  - not a database scan
  - alignment of selected pairs only
- perfect for GPUs

```
TTAGCACAGGAAC-CTA      shift=4
    CACAG-AACTCTAGG    score=9
```



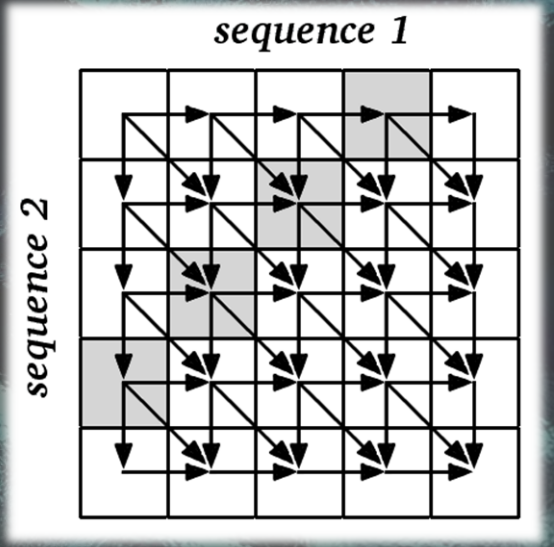
Ultra fast implementation on GPU!

DNA overlap graph construction

NW and dynamic programming (DP):

- data dependencies: left, upper and diagonal elements are needed

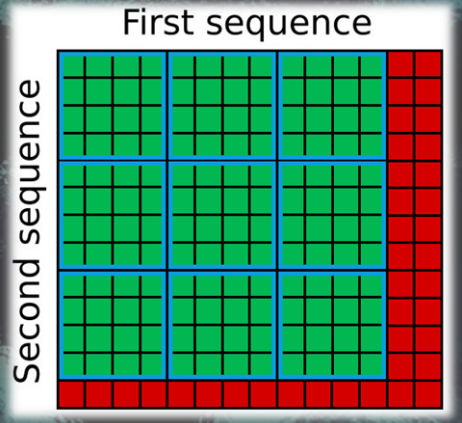
$$H[i,j] = \max \begin{cases} H[i-1,j] - G_{penalty} \\ H[i,j-1] - G_{penalty} \\ H[i-1,j-1] + SM(s_1[i], s_2[j]) \end{cases}$$



DNA overlap graph construction

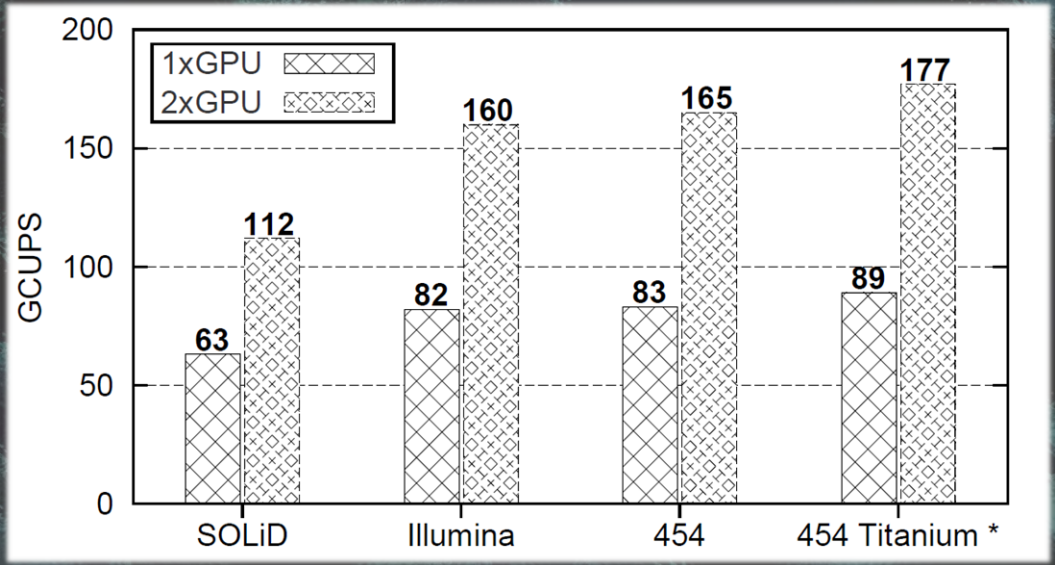
Key GPU optimizations:

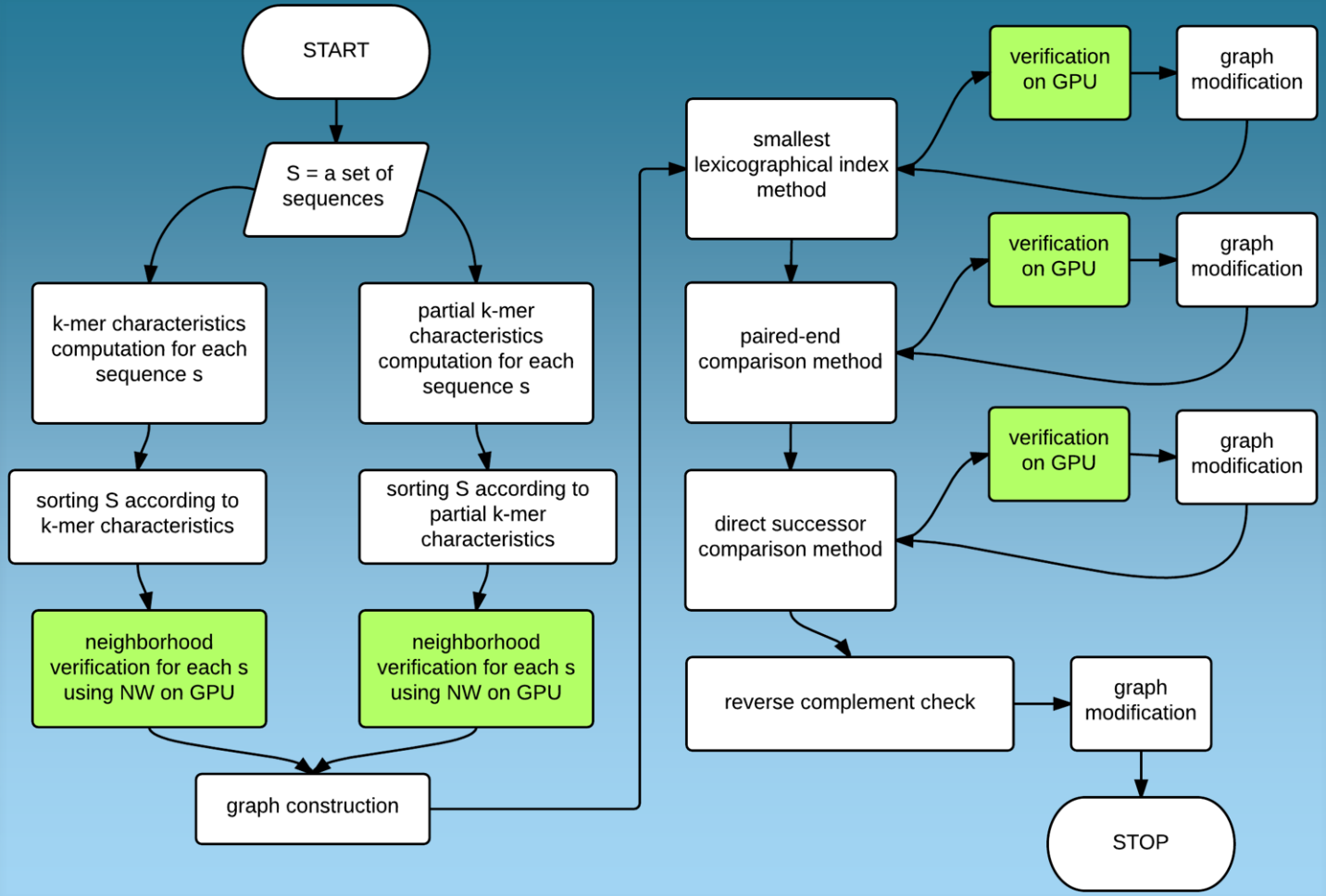
- bitwise compression of sequencing data
  - optimized for nucleotide sequences
- extremely efficient memory access:
  - coalesced access + data prefetch
  - up to 256 cells computed from a single int fetch
  - compute bound
- loop unrolling!
  - DP features nested loops
  - 28 kernels with unrolled loops for various sequence lenghts



DNA overlap graph construction

- the fastest software in its class worldwide
- up to 89 GCUPS on a single GPU



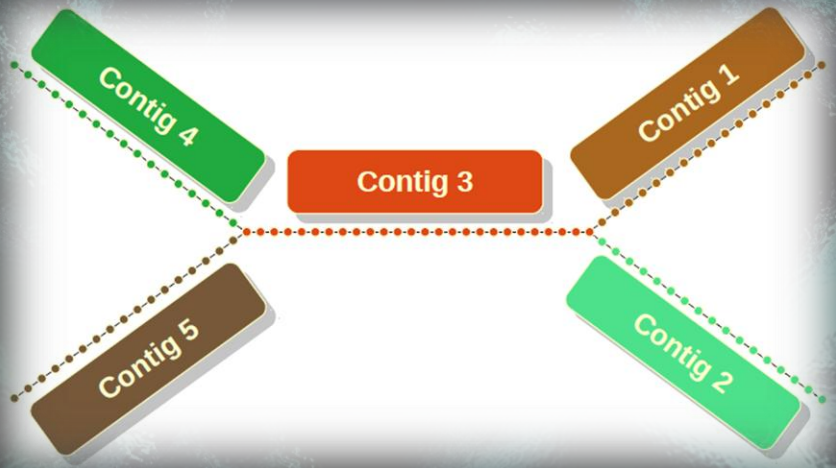

START
S = a set of sequences
k-mer characteristics computation for each sequence s
partial k-mer characteristics computation for each sequence s
sorting S according to k-mer characteristics
sorting S according to partial k-mer characteristics
neighborhood verification for each s using NW on GPU
neighborhood verification for each s using NW on GPU
graph construction
smallest lexicographical index method
paired-end comparison method
direct successor comparison method
reverse complement check
verification on GPU
graph modification
verification on GPU
graph modification
verification on GPU
graph modification
graph modification
STOP

DNA overlap graph construction

- high accuracy of graph construction:
  - sensitivity up to 99%
  - precision: ca. 97%
  - pairs with min. overlap of 40% are well detected
  - very good error handling
- ultra fast reads alignment on GPU makes it possible to check more promising pairs in a reasonable time

Graph traversal

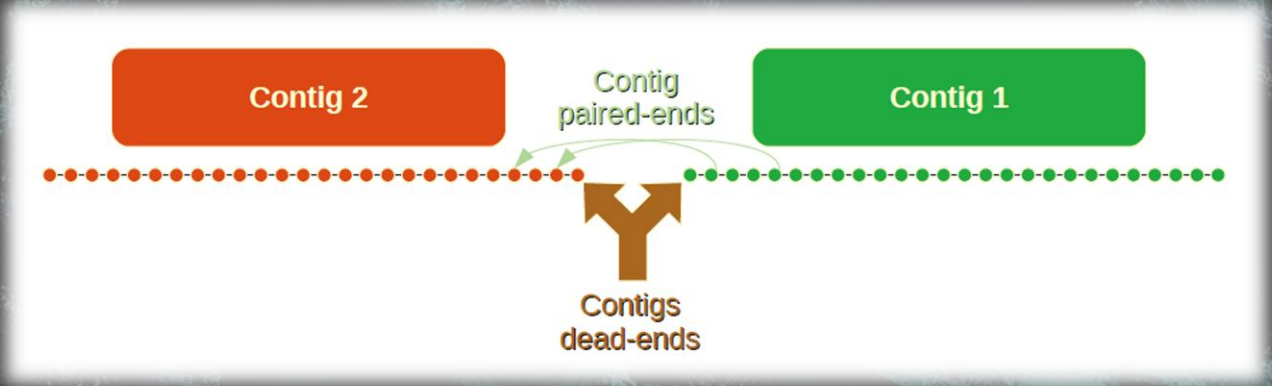
- custom greegy algorithm visits every node
- visited nodes – a sequence of consecutive reads (contig)
- key difficulty – repetitive genome regions
- a dedicated algorithm detecting branches
- graph of contigs



Graph traversal

Graph of contigs:

- useful to perform scaffolding



G-DNA - whole genome test

Data set	1
organism name	Candidatus Microthrix parvicella Bio17-1
species	bacteria
genome length	4,202,850 bp
chromosome №	1
reference genome in pieces	yes: 13
sequencing technology	Illumina GA II
read length (before trimming)	100 bp
avg. read length (after trimming)	89.5 bp
reads № (after trimming)	2 x 2,208,789
paired-end	yes
avg. coverage	94
insert size	435 bp
σ of insert size	100 bp

G-DNA - whole genome test

- very high quality of contigs expressed as percentage of identity
- superior contig lengths

assembler	Velvet		SOAPdenovo		Celera WGS		SGA		SR-ASM		G-DNA	
number of contigs	430		14260		329		6046		9005		300	
aligned 0 times	6	1.40%	49	0.34%	5	1.52%	24	0.40%	364	4.04%	6	2.00%
aligned 1 time	421	97.91%	13186	92.47%	320	97.26%	5454	90.21%	8574	95.21%	291	97.00%
aligned >1 time	3	0.70%	1025	7.19%	4	1.22%	568	9.39%	67	0.74%	3	1.00%
N50	19674		1801		17027		6987		63912		23156	
contig length sum	3689879		5253873		3518044		4475163		6102827		3688731	
10 longest contigs (length and % of identity)	95205	99.38%	17749	99.86%	92091	98.46%	47470	99.67%	225071	95.37%	141105	99,53%
	87349	98.47%	16851	99.79%	81611	99.70%	45539	99.73%	141460	96.89%	89267	99,48%
	84978	99.63%	13492	99.76%	77990	99.79%	43417	99.74%	132035	97.44%	86285	99,53%
	74155	99.59%	12986	99.82%	68811	99.68%	31063	99.71%	131058	98.53%	72635	99,49%
	70808	99.57%	12883	99.76%	66620	99.67%	30890	99.78%	129931	98.29%	69815	99,54%
	68186	99.67%	12629	99.87%	60334	99.73%	29077	99.66%	121798	98.01%	63101	99,60%
	63679	99.60%	11905	99.86%	54458	99.81%	28333	99.76%	111032	97.30%	55712	99,59%
	62306	99.52%	11336	99.51%	54376	98.61%	27836	99.67%	100720	90.78%	52433	99,54%
	62139	99.56%	11311	99.80%	48096	99.78%	27139	99.65%	100517	96.66%	51496	99,56%
	55979	99.61%	10274	99.90%	45601	99.73%	26203	99.78%	92359	80.52%	49005	99,67%

Conclusios

- heavy GPU computations help to construct high quality DNA overlap graphs
- highly accurate graphs + good traversal method = very high quality contigs
- memory efficient implementation
- ready for next-generation sequencing / big data

Contact information

Michał Kierzynka
michal.kierzynka@cs.put.poznan.pl
http://www.cs.put.poznan.pl/mkierzynka

Please complete the Presenter Evaluation sent to you by email or through the GTC Mobile App. Your feedback is important!