Sequence assembly

Radek Píbil

Jaroslav Horáček

MFF UK

2011

Co je to sequence assembly?

- DNA, RNA řetězce nelze zpracovávat najednou
- Nelze je najednou celé přečíst (až stovky miliónů bází - bp)
- Proto se strojově rozsekají na menší kousky, které již přečíst lze
- Chceme-li mít celkový obraz řetězce, musíme ho opět složit

=> Sequence assembly

Jak funguje sequence assembly?

ACGTACGTAGCTGACGTAGCA…

Vstupní řetězec

Fragmentace

ACGTGCAGCATGACGA…
ACGTAGCAGCGAGCAG…

Fragmenty (cca 1K -10K bp)

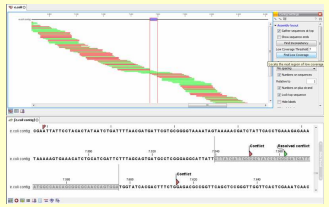
Ready (cca 100-1000 bp)

Single-end ready

```
>read
ACTGCGTA
>read
TCCGATGC
…
```

Paired-end ready

```
>read_pair1
ACTGCGTA
>read_pair2
ACTGATGC
…
```



Assembly

Složený řetězec = Contig

```
>contig
ACGTACGTAGCTGACGTAGCA…
```

Pracovní body

- Vytvořit testovací data
- Otestovat na datech existující assemblery
- Vytvořit vlastní assembler a zjistit čeho všeho je schopný

Příprava testovacích dat

- S pomocí http://crdd.osdd.net/raghava/genomeabc
- Volitelné parametry:
  - Délka fragmentu
  - Délka readu
  - Coverage (= počet_fragmentů * délka_fragmentu / délka_řetězce)

- Umožňuje generovat single-end i paired-end ready (double-barrel shotgun sequencing)
- Umožňuje zanášet mutace
- Možnost otestovat složenou posloupnost s výchozí

Testovací data

- Vygenerováno z reálných data z NCBI

– *Influenza A*	*(ptačí chřipka)*	*1402 bp*
– *Rattus Norwegicus*	*(potkan)*	*7740 bp*
– *Escherichia Coli*	*(střevní bakterie)*	*47773 bp*
– *Ectocarpus Siliculosus*	*(hnědá řasa)*	*199622 bp*

Testovací ready

- Paired-end ready

Původce	Délka fragmentu	Délka readu	Počet readů
Influenza A	100	36	394
Influenza A	400	100	284
Rattus Norwegicus	400	36	1570
Escherichia Coli	2000	500	968
Escherichia Coli	1000	4000	484
Ectocarpus Siliculosus	1000	4000	2008

Výběr assemblerů

- Spousta assemblerů:

  Phrap
  Celera-assembler
  CABOG (modified Celera-assembler for 454)
  Newbler
  Arachne
  AMOS (A Modular Open-Source whole genome assembler)
  ABBA (Assembly Boosted by Amino Acid Sequences)
  MIRA
  ABySS
  Euler
  Velvet
  SOAP denovo
  ...

Výběr assemblerů

- Omezení:
  - Mnoho z nich pracuje s chromatogramovými soubory (.abi, .ab1, .ab2)
  - Mnoho z nich je komerčních
  - Některé z nich je podivně zdokumentovány
  - Hardwarové nároky 13Gb RAM

- Nakonec vybrány:
  - DNA Dragon
  - DNA Baser v3
  - Velvet
  - CodoneCode Aligner

Něco málo o assemblerech

- DNA Dragon
  - komerční (trial 30 dní)
  - Windows
  - podpora různých formátů
  - cena 1299 Euro
  - firma SequentiX (Německo)
- DNA Baser v3
  - komerční (trial 8 hodin)
  - rozsáhlý nástroj pro skládání a vizualizaci sekvencí
  - cena $499
  - firma Heracle BioSoft (Rumunsko)

Něco málo o assemblerech

- Velvet
  - open source
  - navržen pro 64-bit Linux, měl by fungovat i na ostatních
  - hlavně pro krátké ready
  - využívá hashování
  - European Bioinformatics Institute

- CodoneCode Aligner
  - komnerční (trial 30 dní)
  - Windows, Mac
  - Grafické prostředí s využitím Phrap
  - firma CodonCode Corporation (USA)
  - cena $720

Výsledky čas

Vzorek	DNA Dragon	DNA Baser	Velvet	CodoneCode Aligner
L= 36 #394	1 s	-	1 s	1 s
L =100 #284	1 s	151 s	1 s	1 s
L = 36 #1570	5 sec	-	1s	3 s
L = 500 #968	6 min	~1 hod	2 s	10 s
L = 4000 #484	11 min	-	2 s	8 s
L = 4000 #2008	1 hod 3 min	-	-	47 s

contigů
Max. contig

Výsledky contigy

Vzorek	DNA Dragon	DNA Baser	Velvet	CodoneCode Aligner
1402 bp L= 36 #394	3 1276	-	10 757	15 195
1402 bp L =100 #284	1 1393	1 1393	70 903	1 1393
7740 bp L = 36 #1570	2 7457	1 2063	11 1974	12 2352
47773 bp L = 500 #968	1 47668	-	25 22689	1 47668
47773 bp L = 4000 #484	1 47082	-	13 32524	1 47082
199622 bp L = 4000 #2008	3 32782	-	-	8 51506

Poznatky

- DNA řetězec často není kompletně pokrytý ready!
- DNA Dragon měl poměrně dlouhý čas skládání oproti ostatním, ale výsledky byly velmi dobré (0% chyba)
- DNA Baser měl příliš dlouhé časy skládání, přestože v reklamě psali, že složení proběhne dřív než se vrátíme z přestávky na kafe
  - Možná je způsobeno špatným nastavením parametrů (je jich mnoho)
  - Možná tím, že je to trial verze
- Velvet i na velkém počtu readů běžel jen několik vteřin, ale menší přesnost složení (občas se vyskyly chyby tak kolem 1%)
- Velvet na dlouhé úseky (1000bp) a mnoho readů již nestačil
- Výsledek Velvetu závisí na nastavení parametru k – velikost hashovaného k-meru
- CodonCode běhá též velmi rychle a jeho přesnost složení je větší než u Velvetu, nicméně občas se též vyskytnou chyby ve složení

Vlastní „assembler“

SHOTGUN SEQUENCING

Vlastní „assembler“

- Pokus o Shotgun Sequencing
- Zdroje:
  - http://www.cbcb.umd.edu/papers/ADCOM6006.pdf
  - Staré slidy č. 10 k přednášce z BioInf Alg
- Nepovedlo se plně dokončit
- Generování delších podsekvencí

Shotgun Sequencing

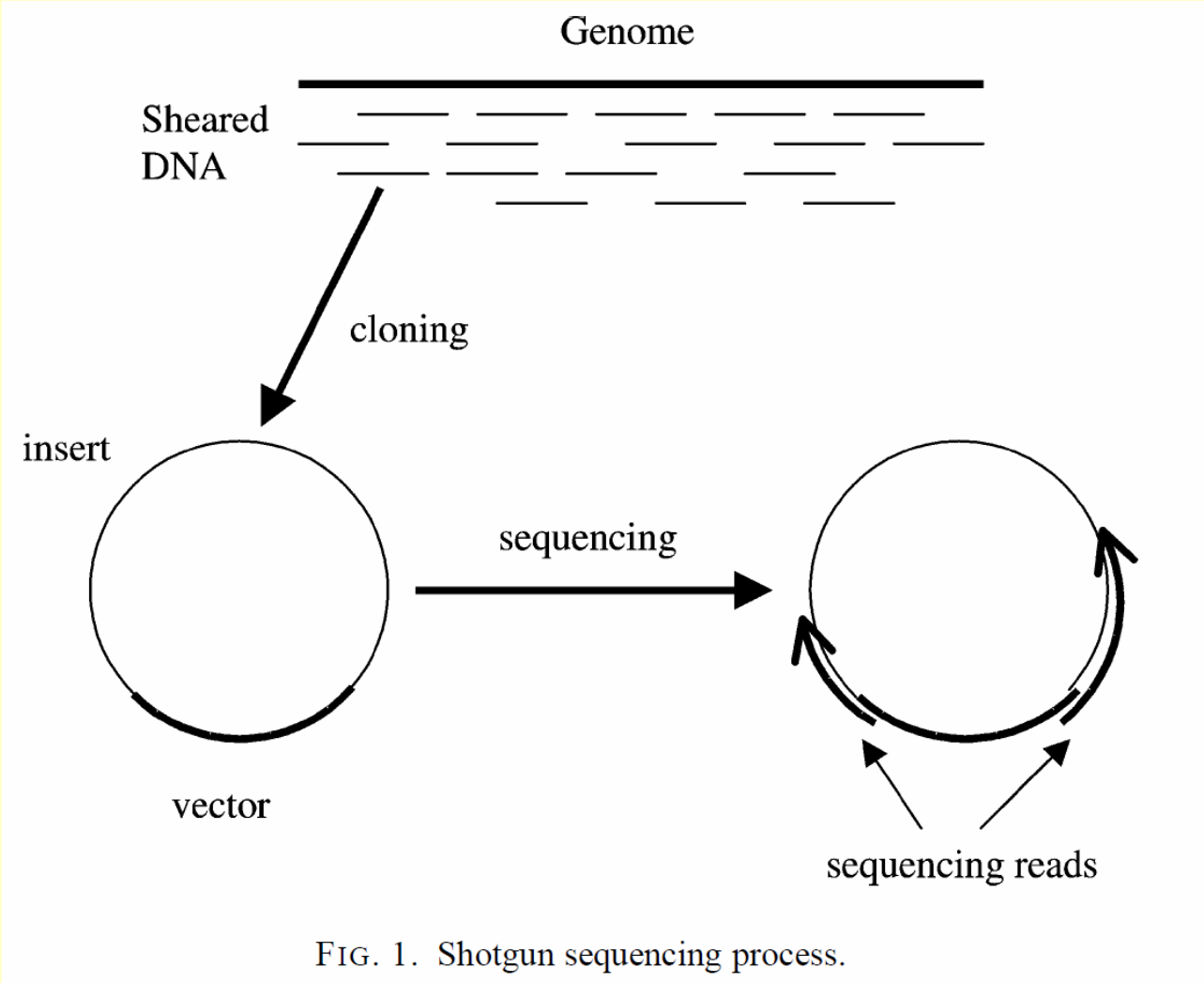
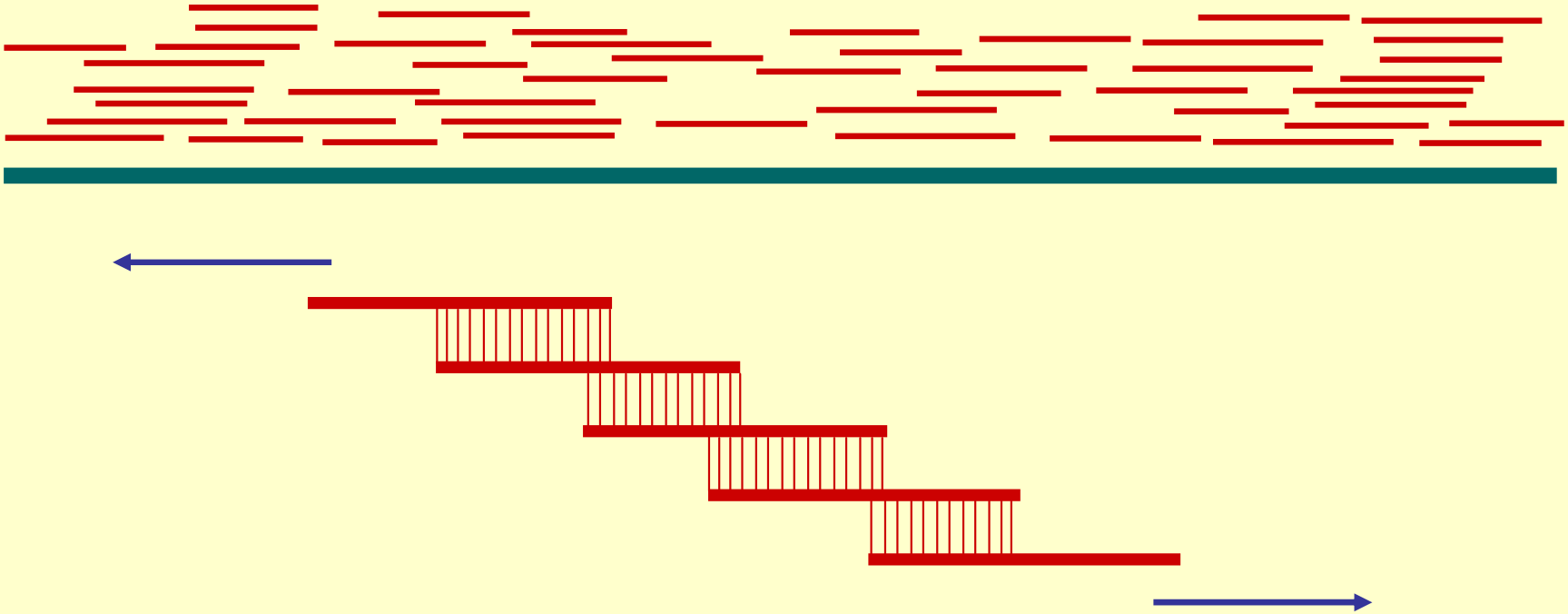


FIG. 1. Shotgun sequencing process.

Obrázek lepší, než slide plný slov

Shotgun Sequencing



Shotgun Sequencing

- Fragment – oříznutý kus DNA
- Read – přečtená část fragmentu
- Contig – spojitá část DNA vytvořená assemblerem
- Overlap – překryv read
- Repeat – Eukaryoty mají mnoho sekcí opakujících se

Shotgun Sequencing

- Nalézt podobné ready
  - Dove-tail nebo podřetězec
- Spočítat úroveň překryvů
  - Dobrý hint pro repeaty (když je násobně více překryvů, než je coverage)
- Pokud mám paired-end, mohu jej jednoduše využít k nalezení pořadí (vím, který read byl dřív a který později)
- Ze slabších překryvů mezí většími, u nichž znám pořadí, mohu odvodit vhodné množiny řetězců mezi nimi
- Najdu konsensus
  - I blbé metody fungují

Nalezení podobných readů

- V ideálním případě nejsou chyby čtení
  - Mohu použít suffixový strom, či cokoliv podobného efektivního
- Realita
  - Dynamické programovaní – skoro lokální alignment
    - Hledám ready s nejmenší editační vzdáleností, kde jeden může navazovat na druhý
    - V našem pokusu využíváme přesně toto [vysvětlit na tabuli]

Shotgun Sequencing

- Vyfiltrování chyb
- Odhalení repeatů
- Hledání konsensu
  - Typicky hloupě iterativní, stačí to
  - My jsme použili hloupé počítání výskytů pro každou pozici a zvolení nejčastější báze

Naše vlastní

- Docela náročné udělat to správně, z důvodu ne zcela dostatečných podkladů

- Máme:
  - Upravený alignment pro páry readů
  - Vyfiltrování horších
  - Nalezení dobrých overlapů
  - Nalezení konsensu mezi nimi