Metody bioinformatyki (MBI)

Wykład 4 i 5, sekwencjonowanie, asemblery DNA

Robert Nowak

2025L

Sekwenatory

Genomika - analiza genotypu organizmów

- poznanie sekwencji materiału genetycznego
- mapowanie genomu (znajdowanie pozycji markerów)
- zależności i interakcje wewnątrz genomu
- ewolucja genów i genomów
- zmienność międzyosobnicza genomów

Genotyp informacja genetyczna danego osobnika, najczęściej łańcuch lub zbiór łańcuchów DNA (od 3500 par zasad u wirusów do $3 * 10^{11}$ bp u ameby)

Genom podstawowy materiał genetyczny (DNA jądrowy u eukariotów, $3 * 10^9$ u człowieka)

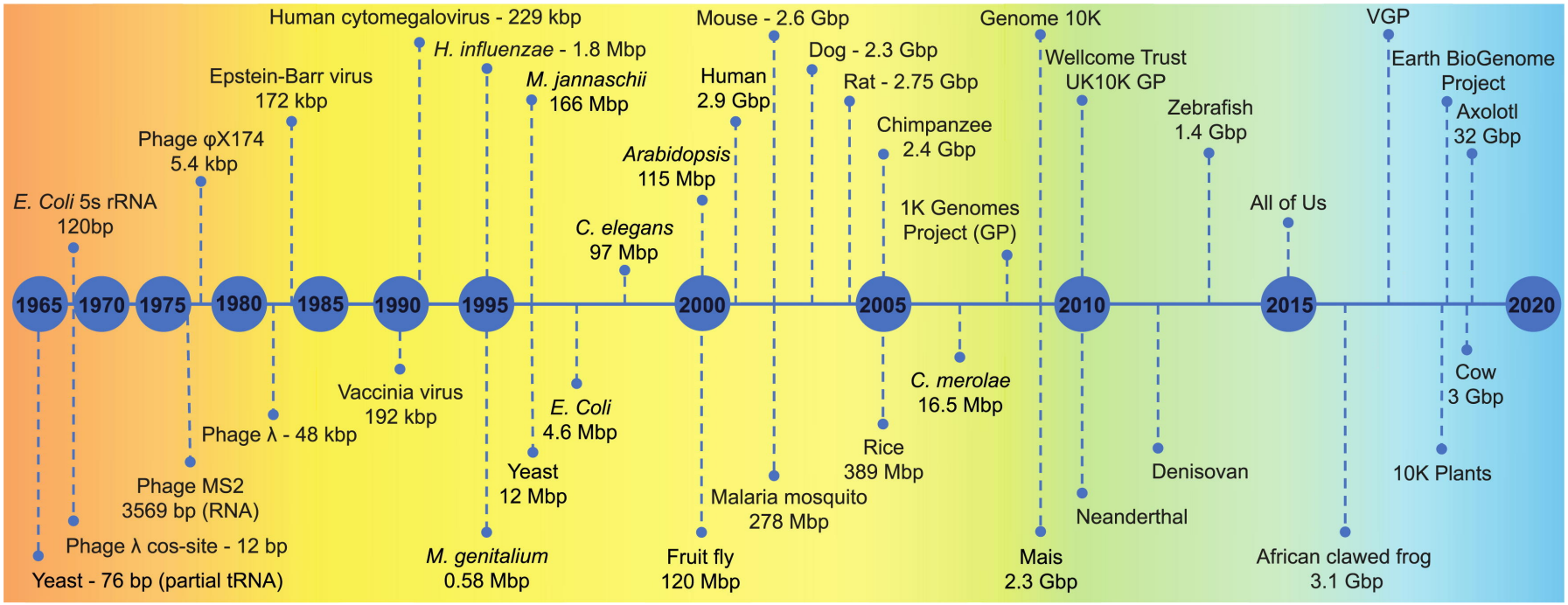
Poznawanie sekwencji genomów - historia

- fag Φ174, 5400 bp - 1977r
- wirus, 170kbp - 1984r
- bakteria, 1,8Mbp - 1995r
- drożdże, 12Mbp - 1996r
- ludzki, 3.3Gbp - 2004r (start w 1990r)
- obecnie 388790 genomów[1]: archea (3856), bakterie (341318), eukariotyczne (33844), wirusy (9772)

Planuje się, że koszt odczytu genomu człowieka spadnie do 100$ (w 2024 jest to 600$) W 2015 przewidywano, że do 2025 roku będzie odczytanych pomiędzy 100 Mln a 2 Mld sekwencji człowieka. Jest ok. 10 Mln. Każdy rekord to 100-150GB.

[1] GOLD (Genomes Online Database), `http://genomesonline.org/`

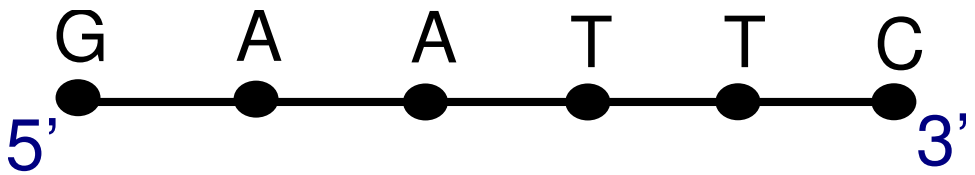
Najważniejsze projekty sekwencjonowania



Sekwencjonowanie

Sekwencjonowanie DNA - proces ustalania kolejności nukleotydów tworzących cząsteczkę DNA.

- sekwencja pierwszorzędowa - nić DNA jest reprezentowana przez napis
- mapa fizyczna

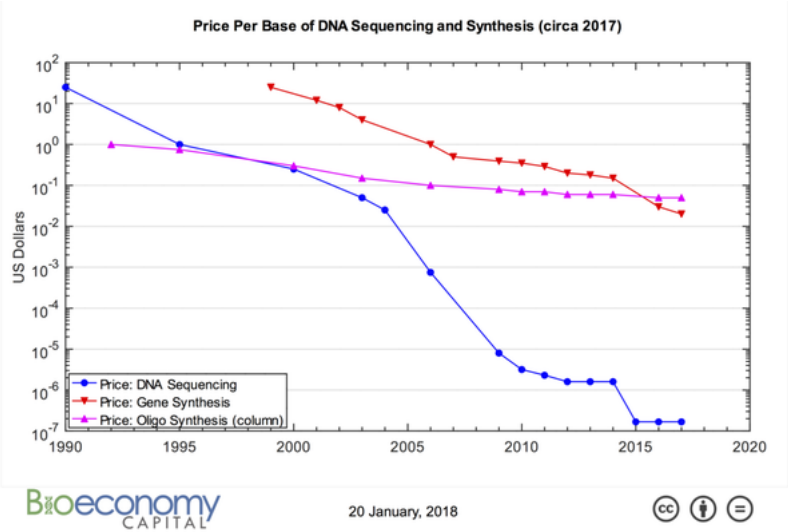


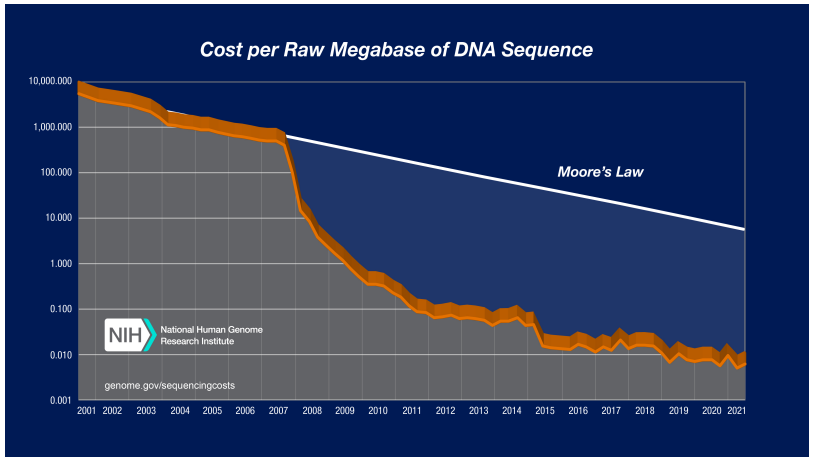
Proces składa się z:

1. odczyt sekwencji fragmentów
2. składanie (asemblacja)
3. wykańczanie

Sekwencjonowanie

Sekwencjonowanie DNA - proces ustalania kolejności nukleotydów tworzących cząsteczkę DNA.



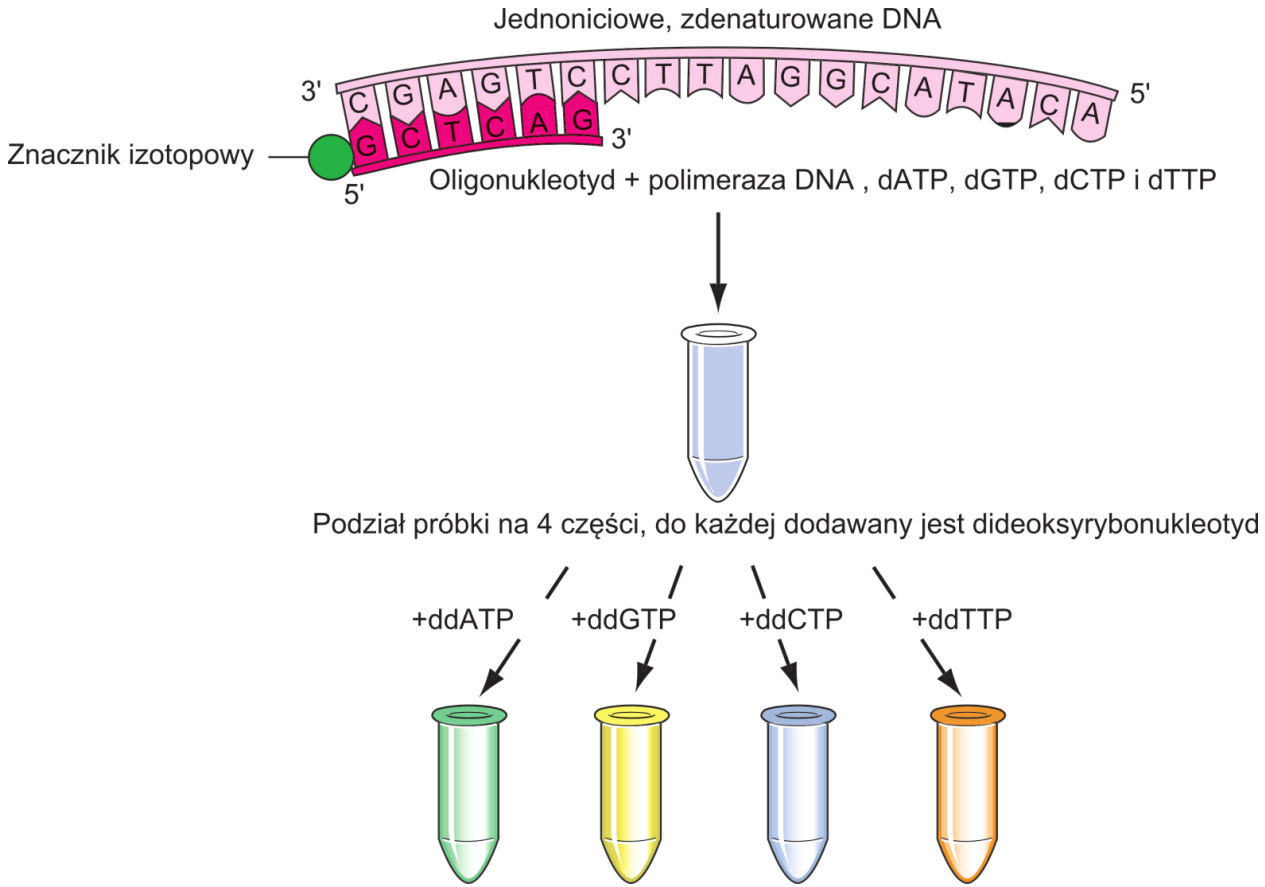


- jeden genom to 360GB odczytów (FASTQ)
- jeden genom to plik 15GB

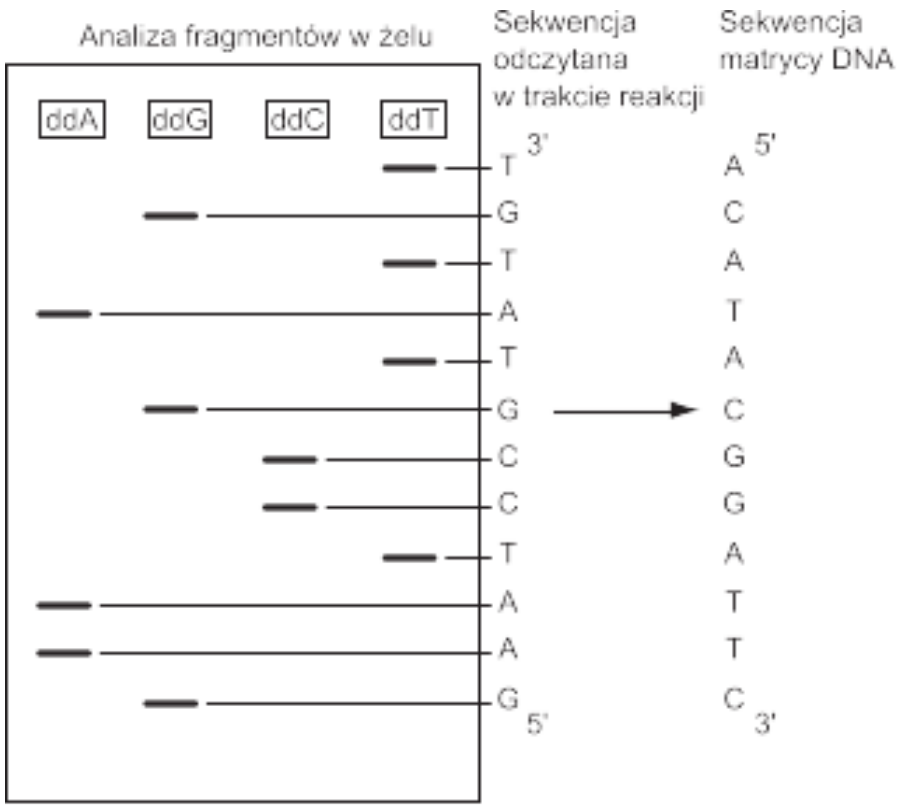
Sekwenatory pierwszej generacji

Metoda Sangera (1) - metoda terminacji łańcucha

Matryca DNA jest denaturowana i dodawany jest znakowany izotopem starter, polimeraza DNA i nukleotydy (dNTP)



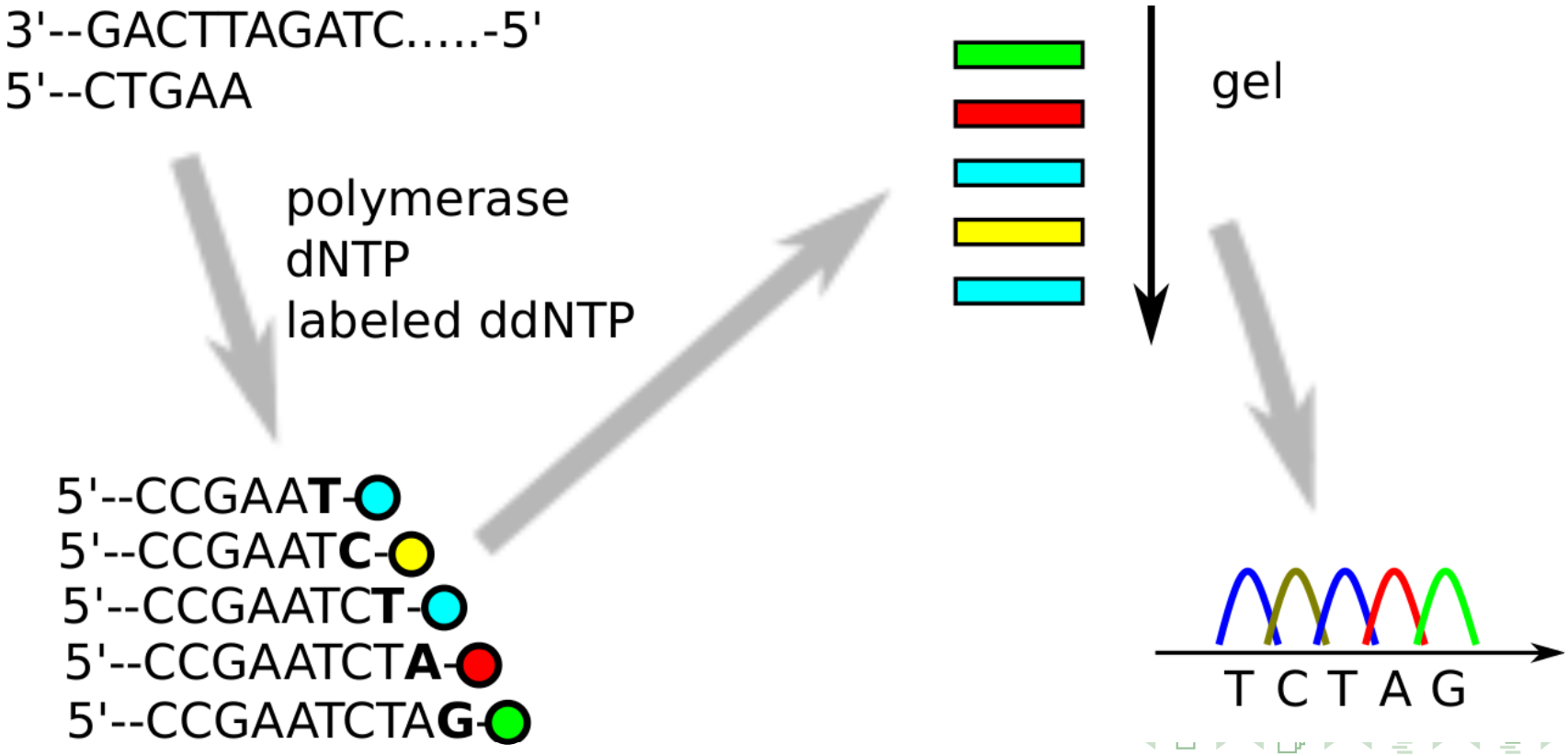
Metoda Sangera (2)



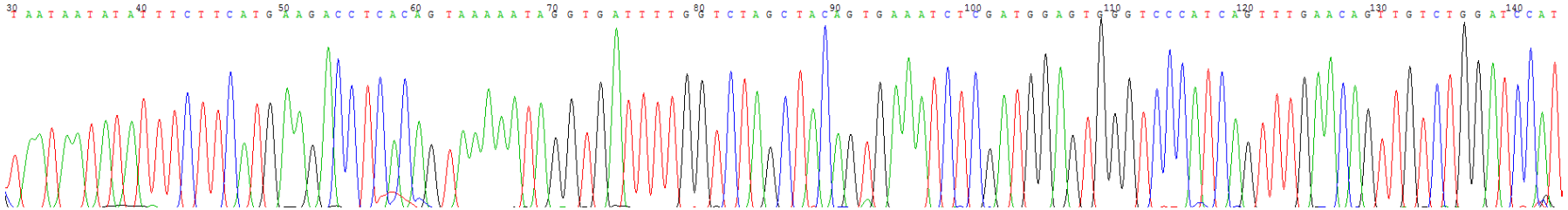
- do rozdziału fragmentów używana jest elektroforeza w żelu poliakrylamidowym
- długość fragmentu odpowiada specyficznym dd-nukleotydom
- sekwencja od 5' do 3' syntezowanej nici jest czytana od dołu żelu

Automatyczne sekwencjonowanie metodą Sangera

- dd-nukleotydy ze znacznikiem fluorescencyjnym
- pojedyncza reakcja i ta sama linia żelu (kapilara)
- sygnał rejestrowany jako chromatogram fluorescencji



Automatyczna metoda Sangera (2)



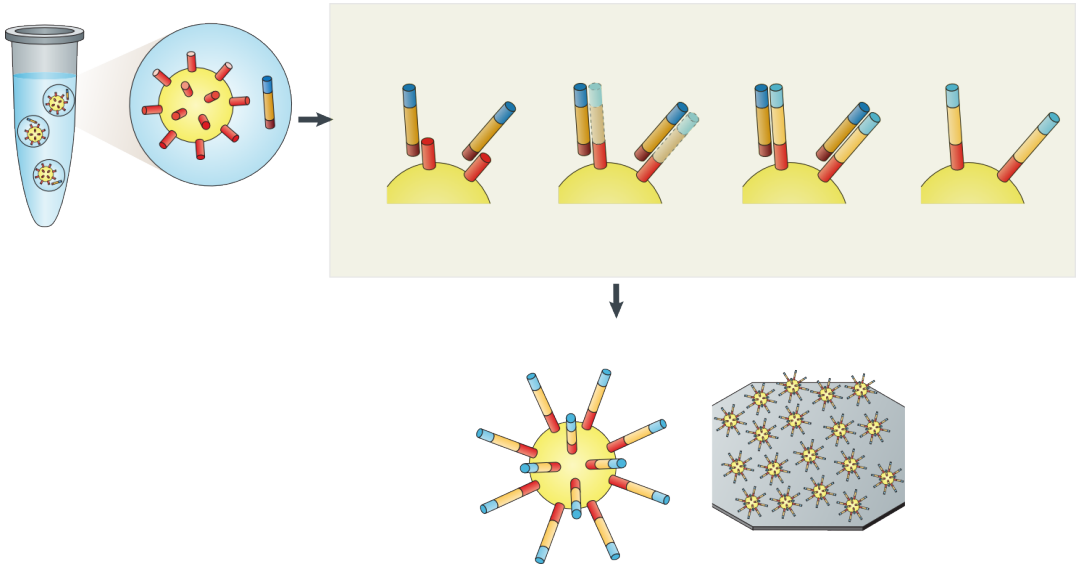
Automatyczna metoda Sangera - sekwencjonowanie pierwszej generacji

- pozwala odczytać sekwencje 2000 bp na reakcję
- dokładność 99.999%
- zdominowała sekwencjonowanie na 20 lat
- użyta do wygenerowania sekwencji genomu ludzkiego
- niska przepustowość i wysokie koszty

Human Genome Project, 2001, 2.7 Mld$

Sekwencjonowanie drugiej generacji

Klonalna amplifikacja fragmentów, emulsyjny PCR

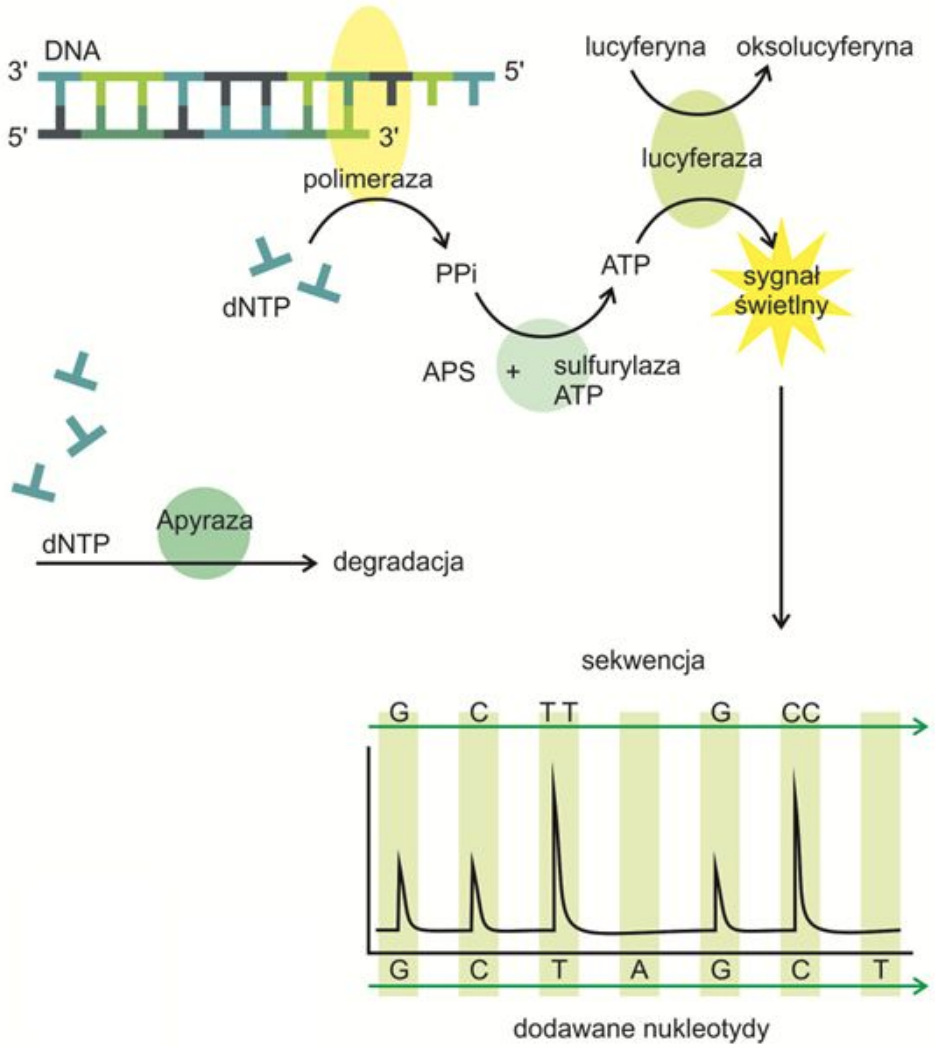


- reaktory: krople w oleju
- kropla zawiera: kulkę, startery, dNTP, matrycę, polimeraza
- jedna cząsteczka DNA na kulkę

- produkty PCR związane do kulek są wtłaczane do studzienek reakcyjnych
- stosowana w urządzeniach: Roche 454, SOLID, Ion Torrent

Piro-sekwencjonowanie, platforma 454

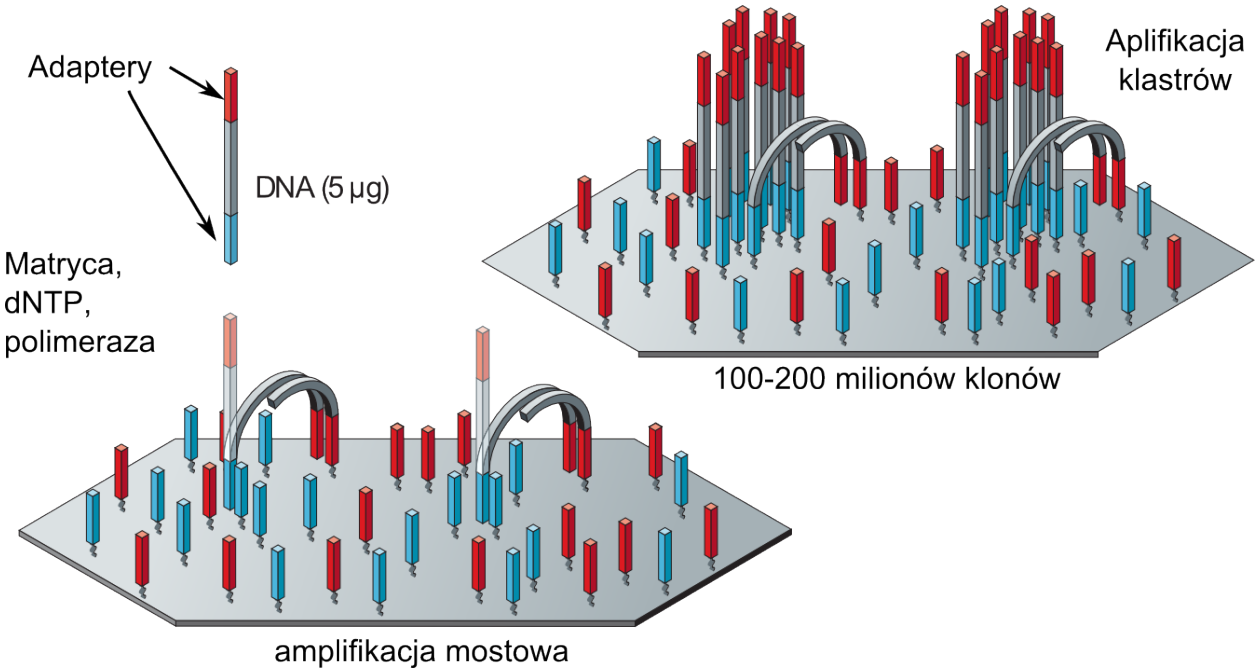
- przeprowadzana jest polimeryzacja z dATP, później dCTP, dGTP, dTTP
- sulfurylaza wychwytuje pirofosforany (PPi) i generuje ATP
- lucyferaza generuje sygnał świetlny
- siła sygnału zależna od ilości zużytego ATP



Platforma Roche 454

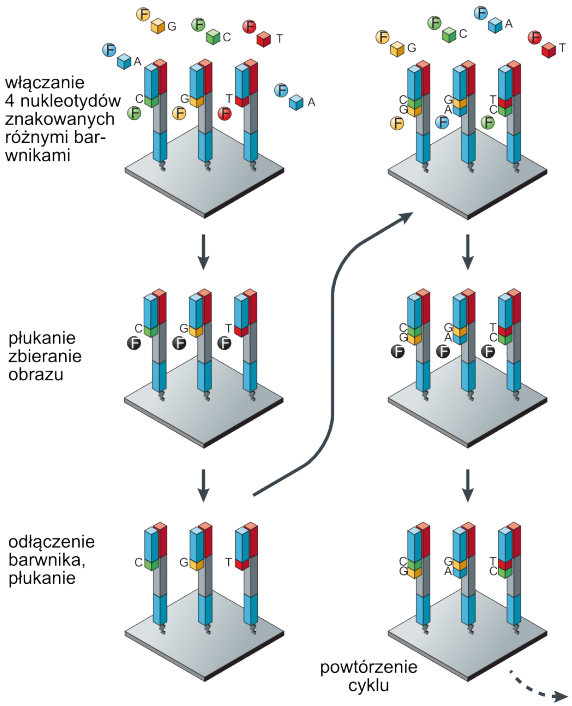
- pierwszy sekwenator drugiej generacji
- początek sprzedaży w 2005 r, koniec produkcji w 2016 r
- sekwencjonowanie $\approx 10x$ tańsze niż metodą Sangera
- $\approx 300x$ bardziej wydajne

Klonalna amplifikacja fragmentów, mostowy PCR, Illumina



- biblioteka losowych fragmentów z adapterami wiązana dwoma końcami do powierzchni płytki
- każdy klaster powiela jeden fragment

Sekwencjonowanie metodą Illumina



Kamera CCD zbiera obraz i archiwizuje w postaci pliku TIFF

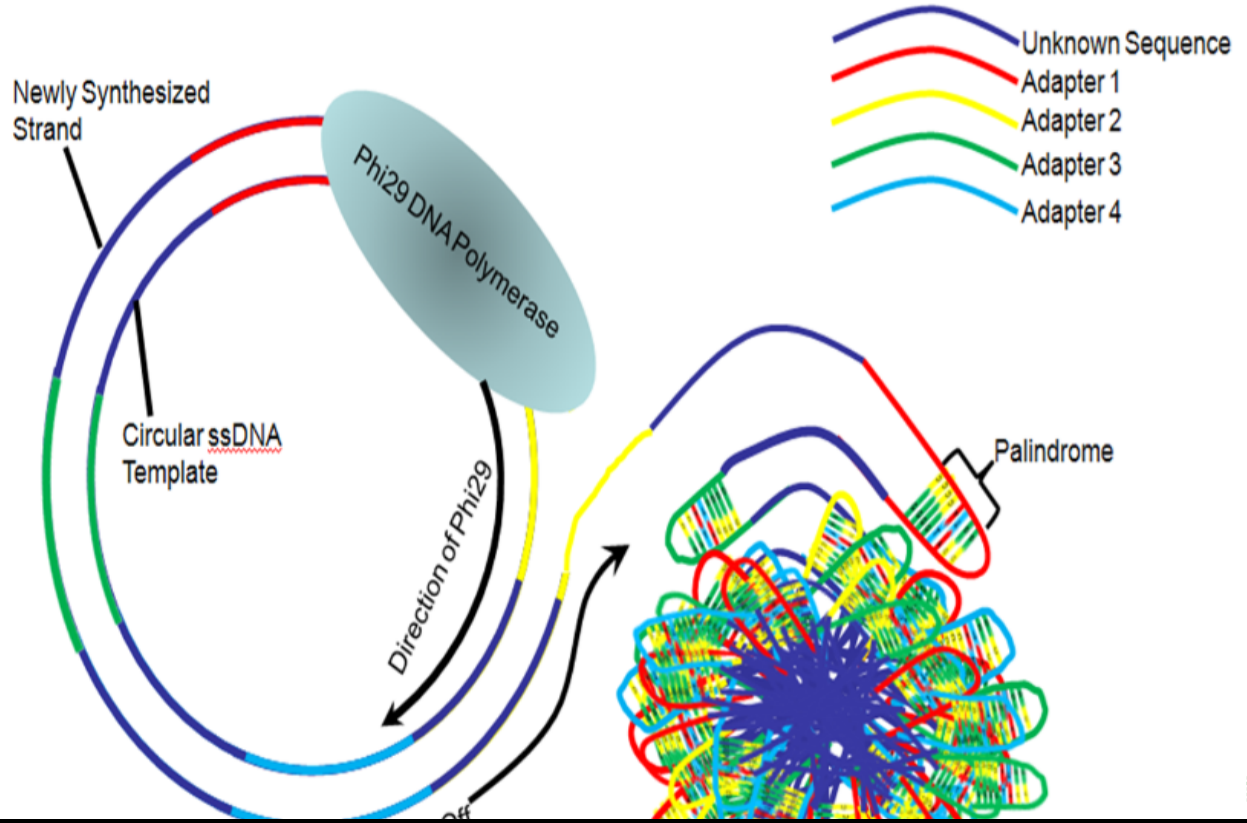
Oprogramowanie "tłumaczy" obraz na sekwencję nukleotydową

sekwencja górna: CATCGT
sekwencja dolna: CCCCCC

- do syntezy używane usuwalne ddNTP wyznakowane fluorescencyjnie
- iteracyjnie skanuje się a następnie usuwa ddNTP

Amplifikacja metodą BGI

- do powielania wykorzystuje koliste DNA zawierające interesującą sekwencję
- synteza tworzy długą nić, która następnie jest wykorzystywana podczas odczytu sekwencji



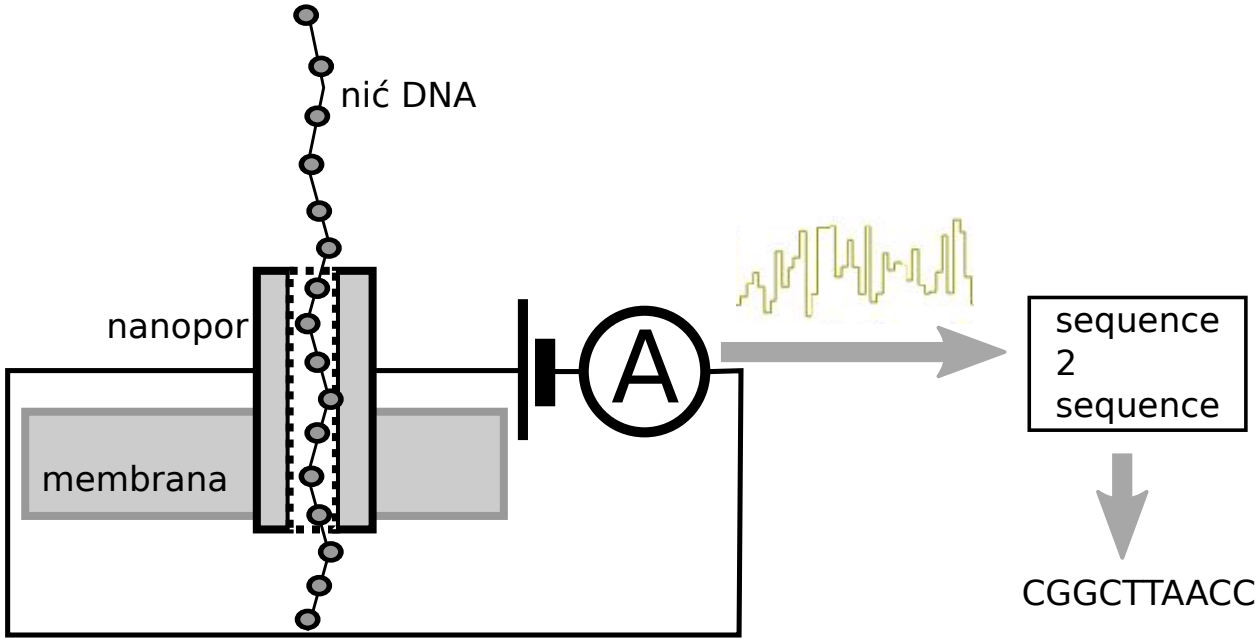
Illumina i BGI

- sekwencjonowanie $\approx 100x$ tańsze niż Roche 454
- $\approx 10x$ bardziej wydajne
- obecnie (2023) Illumina i BGI dominuje na rynku

Przykład: Illumine HiSeq

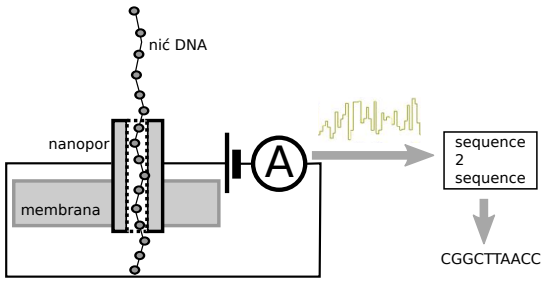
Sekwencjonowanie trzeciej generacji

Pomiar pola elektrycznego w nanoporach



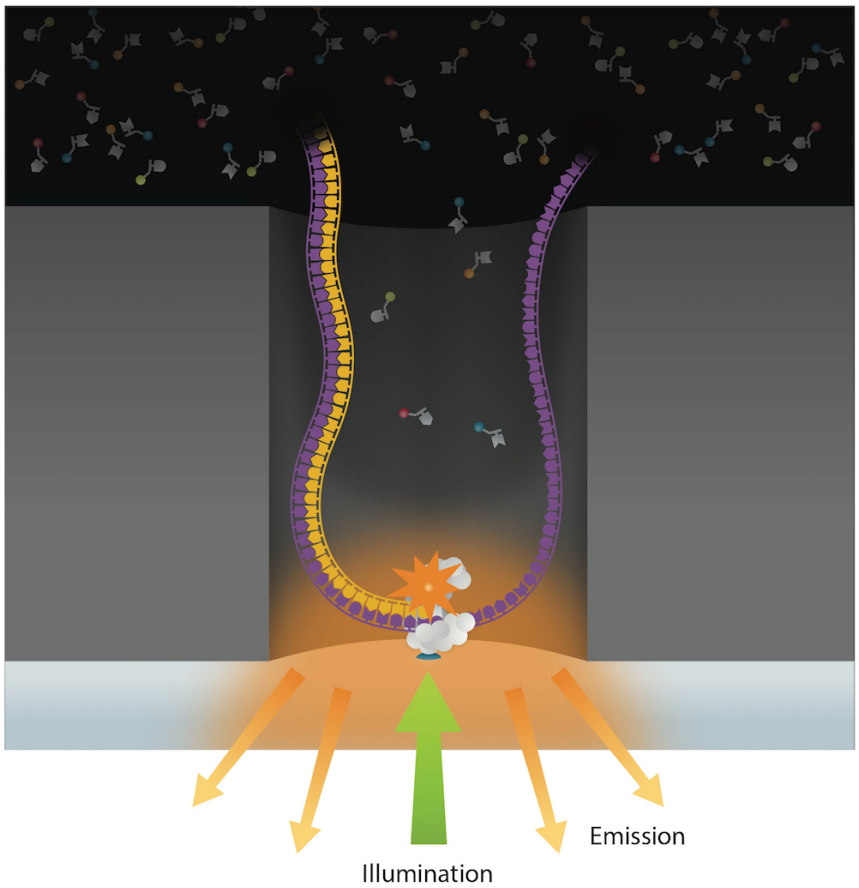
- metoda wykorzystywana przez Oxford Nanopore
- długie odczyty
- 15% błędów

Basecalling - poprawa jakości wyników



- wejście - szereg czasowy, próbki prądu
- metoda - analiza próbek dla kilku sąsiednich symboli
  - ukryte modele markowa
  - sztuczne sieci neuronowe
- osiągnięcia - błąd zredukowany do 3% (Guppy)

Synteza pojedynczych molekuł w czasie rzeczywistym



- reakcja w nano-komorach
- wyznakowane dNTP gdy są przyłączane, emitują światło
- używany PacBio

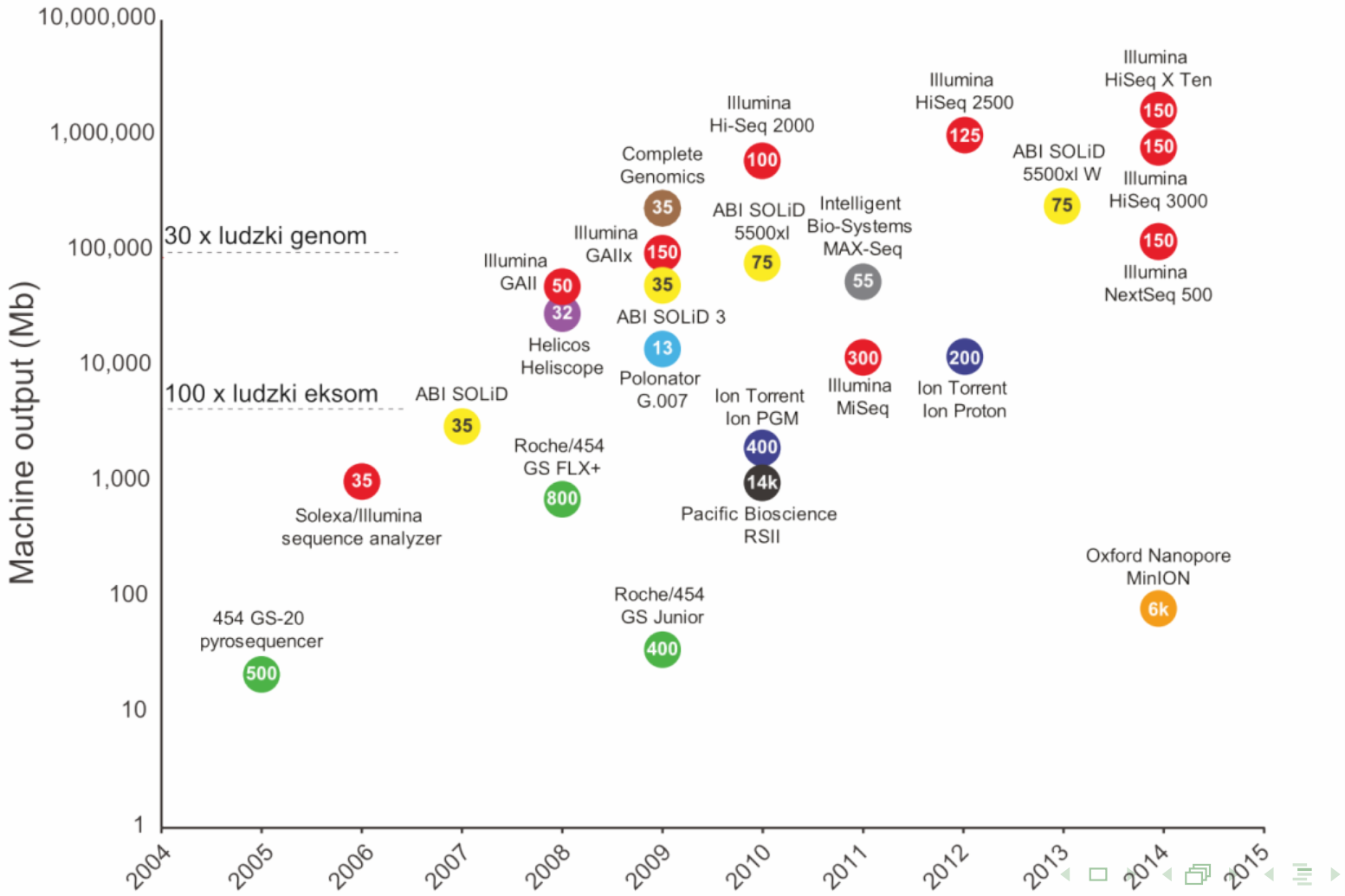
Sekwenatory PacBio i Oxford Nanopore

Sekwencjonowanie DNA - urządzenia

nazwa	długość odczytu	koszt *Mbp*	dokładność
pierwszej generacji			
Sanger	2000bp	2400$	99.999%
drugiej generacji			
Roche 454	500bp	10$	99.9%
Illumina HiSeq	100bp	0.07$	99%
drugiej generacji miniaturowe			
Ilumina, MiSeq	150bp	7$	99%
trzeciej generacji			
PacBio RS	10-100 kbp	10$	85%
Nanopore	50-500 kbp	2$	80%

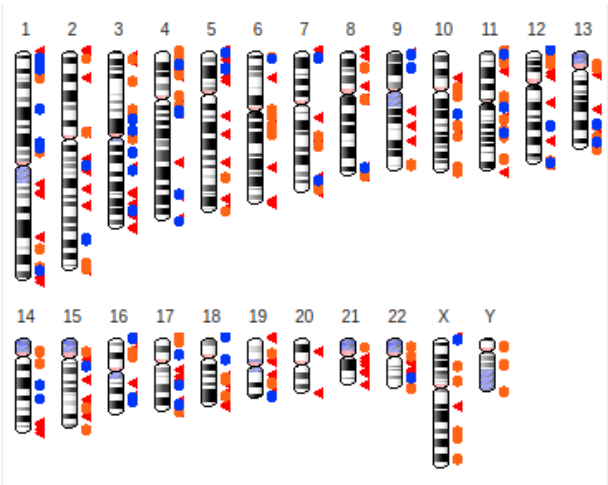
wzrost 21% rocznie, 4.5 mld. $ (2013), 6.6 (2016) 9.7 (2019)

Sekwenatory nowej generacji



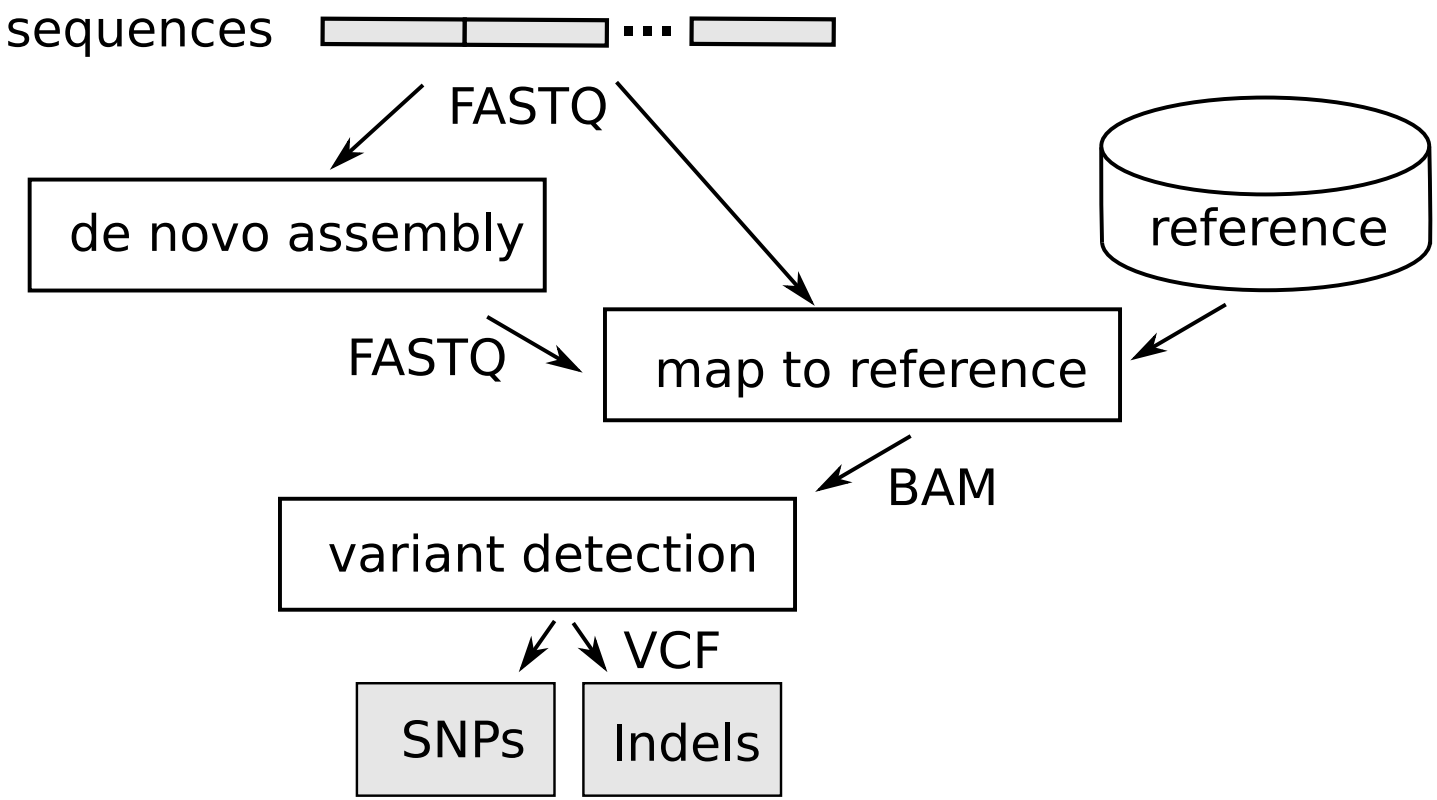
Techniki sekwencjonowania

- *de novo*
  - hierarchiczne, np biblioteki klonów
  - losowe fragmenty (shotgun whole genome sequencing, WGS)
- resekwencjonowanie - wymagana sekwencja referencyjna
  - GRCh38 - genom referencyjny człowieka, major release (2013)
  - GRCh38 v.14 - bieżąca wersja genomu (Luty 2022), 434 gaps
  - GRCh38 v.13 - bieżąca wersja genomu (March 2019), 526 gaps
    `https://www.ncbi.nlm.nih.gov/grc/human/data`
  - udało się uzyskać pełny genom człowieka (31.03.2022),
    `doi:10.1126/science.abj6987`



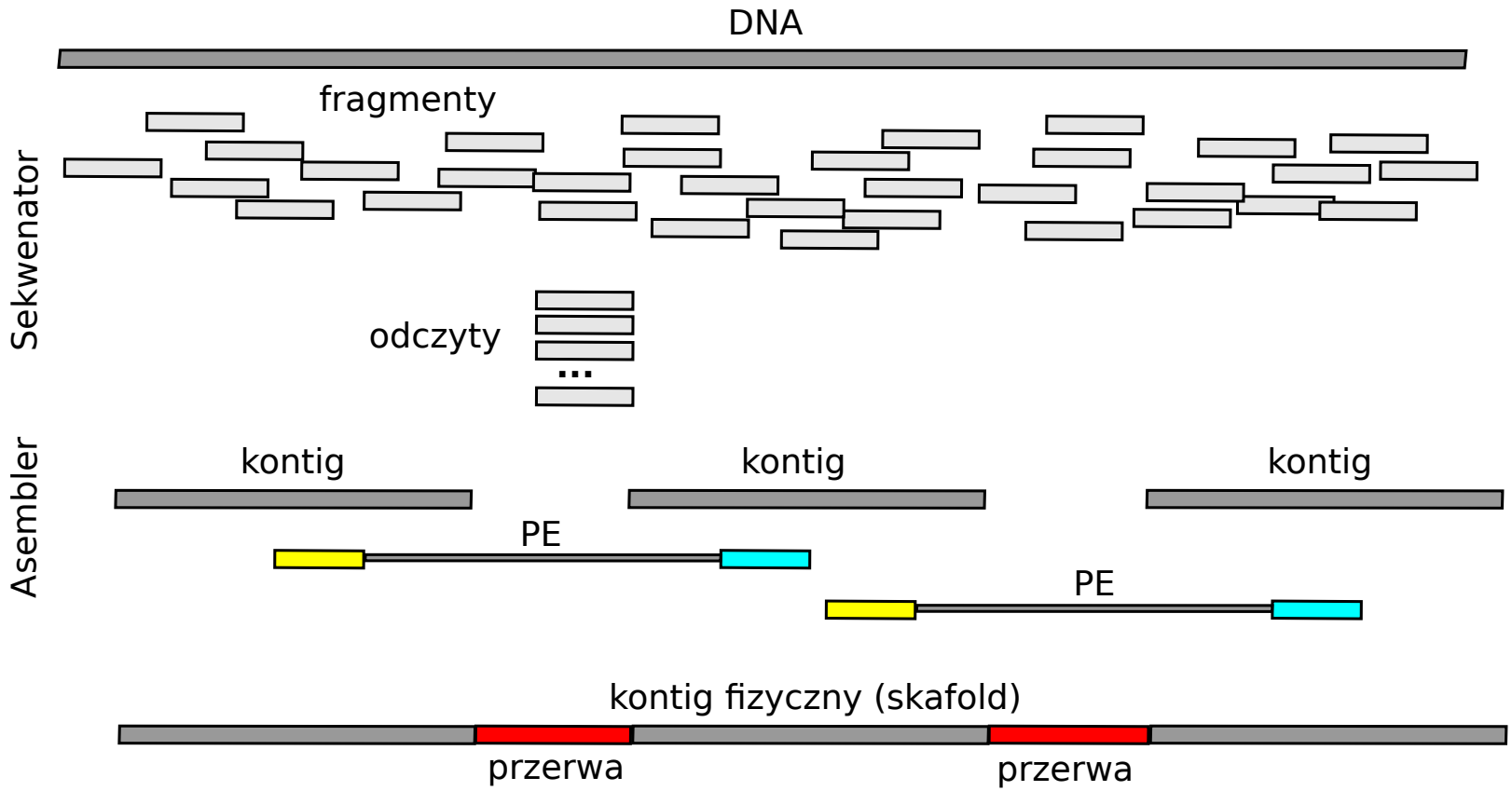
źródło: ncbi

Po co sekwencjonowanie ? analiza wariantów



Nadmiarowość sekwencjonowania

Odczyt sekwencji metodą shotgun (WGS)

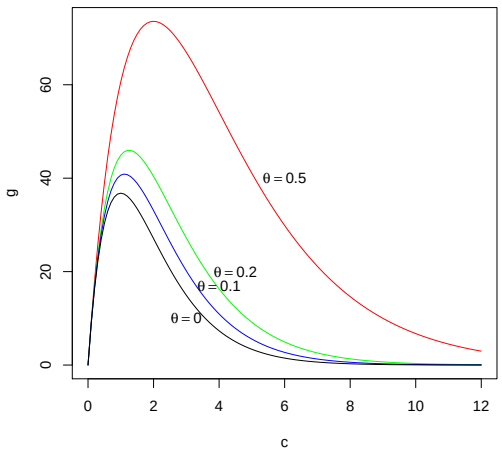


WGS, nieciągłość wyniku – niepełne pokrycie

G długość genomu, L długość fragmentu, N liczba fragmentów
T liczba nukleotydów wymagana do wykrycia pokrywania

$$g = Ne^{-\frac{LN}{G}(1-\theta)}$$

gdzie g - liczba przerw (1988, Lander, Waterman)



$\theta = \frac{T}{L}$ (stosunek długości sekwencji wymaganej do wykrycia pokrywania do długości fragmentu)

$c = \frac{LN}{G}$ (nadmiarowość pokrycia genomu przez fragmenty)

człowiek 3Gbp, fragmenty 100bp, c $= 30x$, $P(g) \approx 10^{-4}$, $N \approx 10^9$, plik $360GB$

losowe fragmenty, liczba bibliotek biologicznych

Biblioteki biologiczne: kolekcja fragmentów DNA

N dla c = 1			
	YAC, 1Mbp	cosmid, 40kbp	fag λ, 15kbp
E.coli	4	100	267
muszka owocowa	130	3250	8667
rzodkiewnik	157	3925	10467
człowiek	3000	75000	200000
prapłatowiec	130000	3250000	8667000

N dla $g \leq 1$ ($\theta = 0$)			
	YAC (1Mbp)	cosmid (40kbp)	lambda (15kbp)
E.coli	9 (c=2.3)	648 (c= 6.5)	2032 (c= 7.6)
rzodkiewnik	1100 (c=7.0)	41761 (c=10.6)	122638 (c=11.7)
człowiek	31028(c=10.3)	1039036 (c=13.9)	2981594 (c=14.9)

WGS, ilość odczytów

$\phi = \frac{T}{L}$	fag(15k) T=9	E.Coli(4M) T=13	muszka (130M) T=16	człowiek (3G) T=18
PacBio	0	0.001	0.002	0.002
Sanger	0.009	0.013	0.016	0.018
454	0.018	0.026	0.032	0.036
Ilumina	0.090	0.130	0.160	0.180

N dla $g \leq 1$

	fag(15k)	E.Coli(4M)	muszka(130M)	człowiek(3G)
PacBio	2 (c=1.3)	3.2k (c= 8.1)	0.15M (c=12.0)	4.6M (c=15)
Sanger	62 (c=4.1)	43k (c=10.7)	1.9M (c=14.4)	53.4M (c=18)
454	151 (c=5.0)	91k (c=11.4)	3.9M (c=15.2)	111M (c=19)
Ilumina	1043 (c=6.9)	527k (c=13.2)	22M (c=16.9)	607M (c=20)

człowiek 3Gbp, Ilumina 100bp, coverage $30x$, $P(g) \approx 10^{-4}$, $N \approx 10^9$, FASTQ 360GB

Asemblery DNA

Algorytmy do składania sekwencji (assembly)

Algorytmy (grafowe) przekształcające ciąg pod-sekwencji (o losowych przesunięciach) nazywanych „odczytami” na zbiór sekwencji (nazywanych kontigami lub kontigami sekwencyjnymi) i zbiór przerw.

Typowe algorytmy:

- graf pokrycia (overlap-layout-consensus, OLC)
- tzw. graf de Bruijn-a (De Bruijn graph, DBG)

Wyjściem nie jest to jedna sekwencja (jeden kontig), ponieważ występują:

- sekwencje powtarzające się dłuższe niż odczyt (OLC) lub rząd grafu (DBG)
- błędy odczytu, błędy sekwencjonowania

Dostępnych jest ok. 50 asemblerów.

Asemblacja de novo. Graf pokrycia.

Graf pokrycia (OLC)

- obliczanie podobieństwa (analiza wszystkich par odczytów)
- konstrukcja grafu (nie wszystkie ścieżki)
- znajdowanie ścieżki Hamiltona w grafie (problem NP-zupełny), więc heurystyki

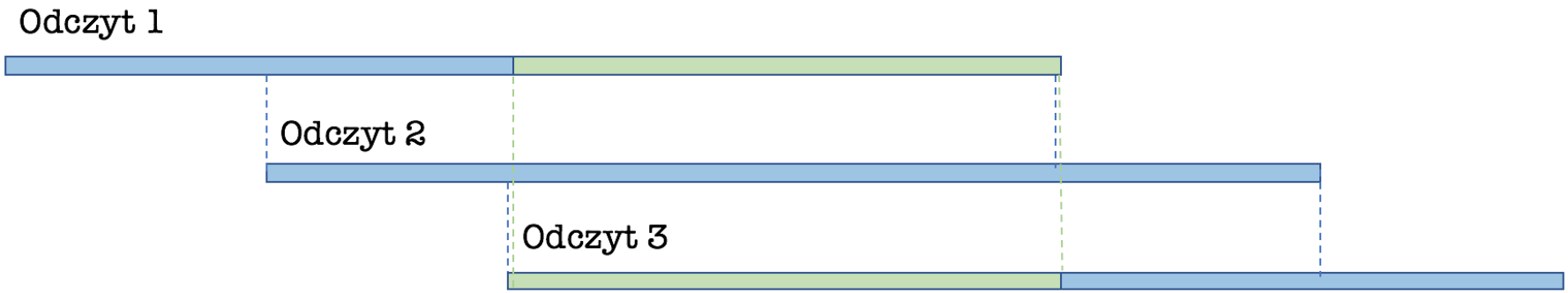
Ścieżka Hamiltona- przechodzi przez każdy wierzchołek grafu dokładnie raz

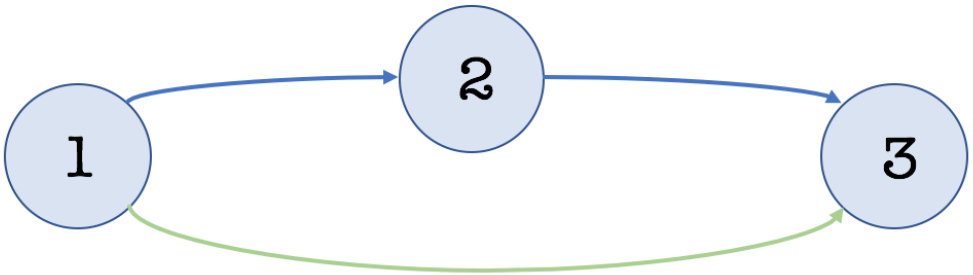
Złożoność: $O(|V|^k)$, gdzie k - maksymalna ilość krawędzi związanych z grafem

- nie nadaje się do NGS (zbyt wolny krok 1)
- próby poprawy: filtr Blooma, algorytm minHash
- lepsze wyniki niż dla DBJ

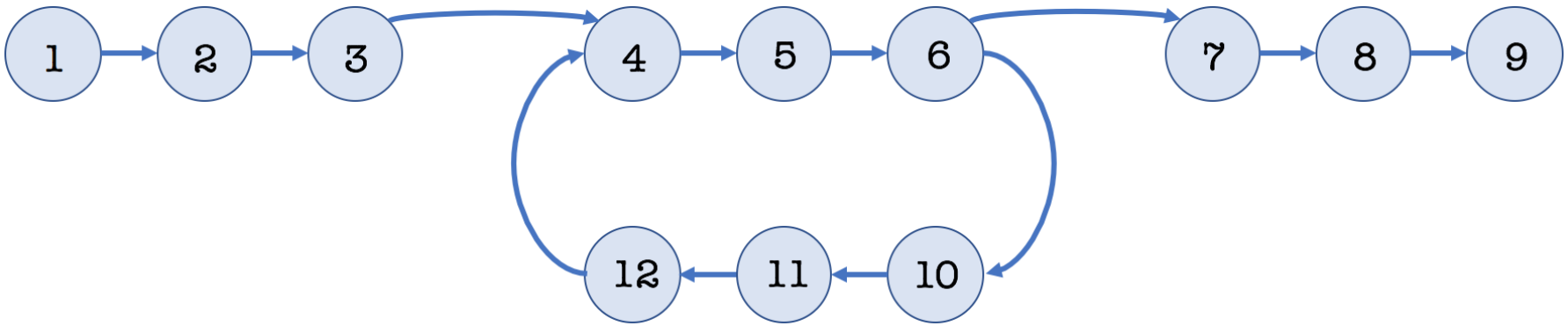
aplikacje: Celera, Arachne, CAP, PCAP, Newbler, CABOG, Edena, Shorty

Graf pokrycia - budowa grafu





Graf pokrycia - znajdowanie ścieżek



Graf pokrycia - znajdowanie sekwencji dla ścieżek

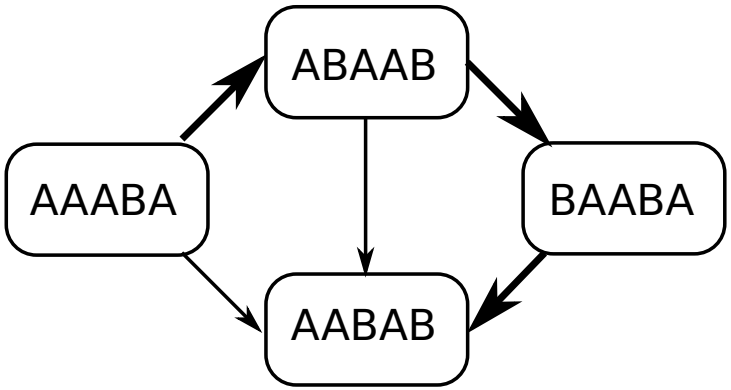
```
1. T A A T C T -- A C
2.       A T C -- G A C T T A
3.           C T G A C T -- A C C G
-------------------------------------
   T A A T C T G A C T T A C C G
```

Algorytm OLC - przykład

4 odczyty, każdy l = 5, brak błędów odczytu, brane podobieństwo z oceną > 2,

a) AAABA
b) ABAAB
c) BAABA
d) AABAB

-	**a**	**b**	**c**	**d**
a	-	3	2	4
b	-	-	4	3
c	-	-	-	4
d	-	-	-	-

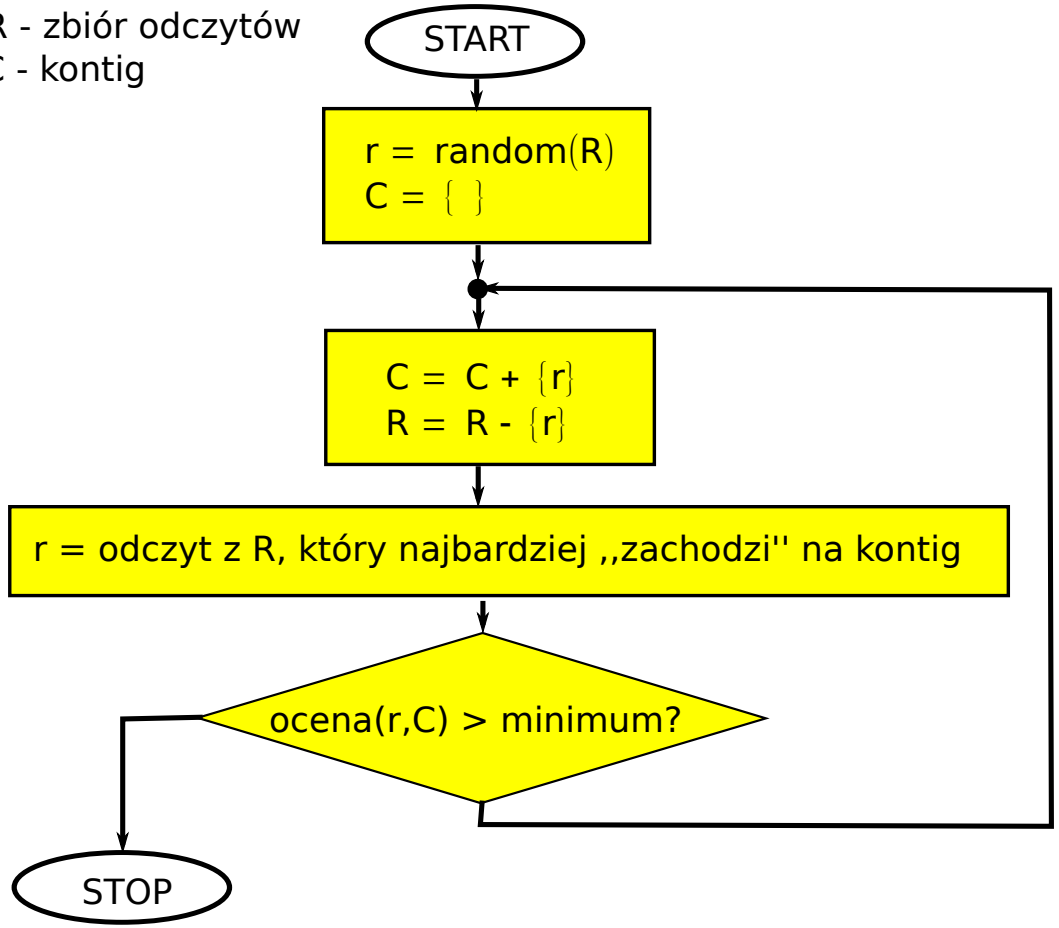



```
A  A  A  B  A  -  -  -  -
-  -  A  B  A  A  B  -  -
-  -  -  B  A  A  B  A  -
-  -  -  -  A  A  B  A  B
```

wynik: **AAABAABAB**

Algorytm zachłanny - jedna z odmian grafu pokrycia

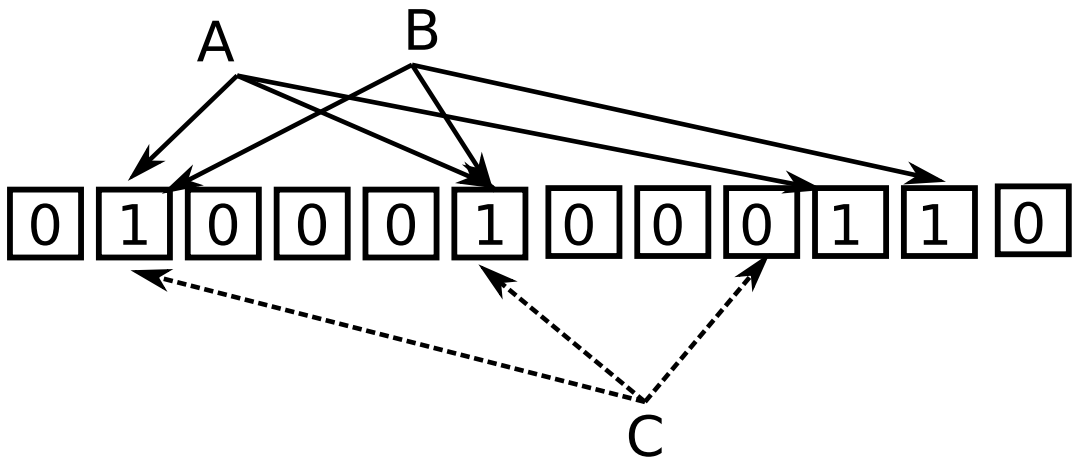
we: R - zbiór odczytów
wy: C - kontig



aplikacje: SSAKE, SHARCGS, VCAKE

Filtr Blooma - zbiór n elementów w tablicy m bitów

Przykład: m = 12, k = 3:

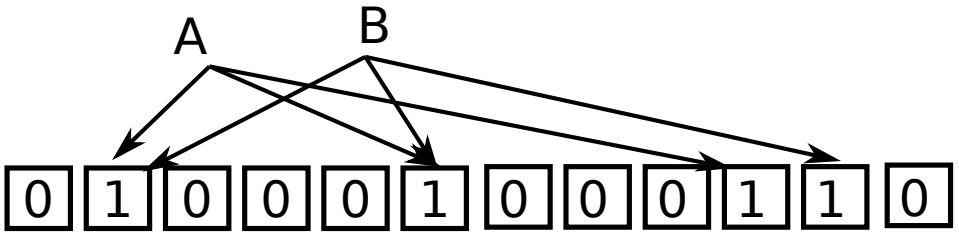


- wykorzystuje k funkcji skrótu (#, haszujących) $k = \lfloor \ln 2 * \frac{m}{n} \rfloor$
- mogą wystąpić błędy FP (false positives)

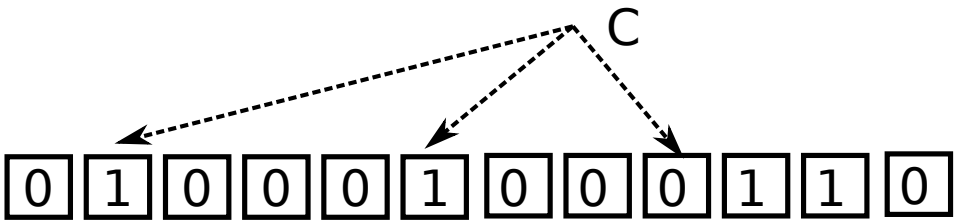
Przykład:
Dla genomu $G = 3 * 10^9$, aby zbadać, czy występuje k-mer o danej długości (unikalny), $n \approx 3 * 10^9$, zakładając tablicę $4GB$ ($m \approx 4 * 8 * 10^9$) użyjemy $k = 11$ funkcji #. Odpowiedź, że dany k-mer nie występuje w sekwencji po 11 obliczeniach #.

Filtr Blooma - operacje

- inicjacja: wszystkie m komórek mają wartość 0
- dodawanie elem.: k krotnie oblicza #, ustawia odp. bity
  Przykład: m = 12, k = 3



- badanie elementu: k krotnie oblicza #, bada, czy wszystkie bity mają wartość 1



Indeks Jaccarda

Indeks Jaccarda - miara podobieństwa między zbiorami

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Przykład:

$$A = \{1, 3, 8, 13, 51\}$$
$$B = \{1, 2, 12, 13, 20, 50, 52\}$$
$$A \cup B = \{1, 2, 3, 8, 12, 13, 20, 50, 51, 52\}$$
$$A \cap B = \{1, 13\}$$
$$J(A,B) = \frac{2}{10}$$

Indeks Jaccarda, k-spektrum

k-spektrum: reprezentacja sekwencji $s_0 s_1 ... sn-1$ przez zbiór $n-k+1$ pod-sekwencji o długości k (k-merów) $\{s_0 ... s_{k-1}, s_1 ... s_k, ..., s_{n-k} ... s_{n-1}\}$

Przykład:
k-spektrum (k=4) dla $s =$ `ACTTATCA`
{ `ACTT`, `CTTA`, `TTAT`, `TATC`, `ATCA` }
dla $t =$ `ACTATC`
{ `ACTA`, `CTAT`, `TATC` }

$$J(s,t) = \frac{|\{TATC\}|}{|\{ACTT, ACTA, ATCA, CTAT, CTTA, TATC, TTAT\}|}$$
$$J(s,t) = \frac{1}{7}$$

Obliczanie indeksu Jaccarda jest złożone obliczeniowo dla rzeczywistych sekwencji, gdzie $n \approx 10^6$, $k \approx 50$, więc zbiory mają $\approx 10^6$ elementów

algorytm MinHash

Algorytm MinHash, przybliżenie indeksu Jaccarda. Bada niektóre elementy zbiorów, jest wydajny obliczeniowo.

Wykorzystuje m funkcji skrótu ($\#$), najczęściej `XOR`
Sygnatura $\star(s)$ dla napisu s to wektor m liczb całkowitych.

Algorytm obliczania $\star(s)$

1: **for** $kmer \in \{s_0 \ldots s_{k-1}, \ldots, s_{n-k} \ldots s_{n-1}\}$ **do**
2: **for** $i \in \{0, \ldots, m-1\}$ **do**
3: $w_i = min(w_i, \#_i(kmer))$
4: **end for**
5: **end for**

algorytm MinHash

$$J(s,t) \approx sim(\star(s), \star(t)) = \frac{1}{m} \sum_{i=0}^{m-1} \sigma(\star_i(a), \star_i(b))$$

gdzie

$$\sigma(x,y) = \begin{cases} 1 & \text{dla } x = y \\ 0 & \text{dla } x \neq y \end{cases}$$

Przykład, $m = 3$ funkcje skrótu $\#_0, \#_1, \#_2$, $k = 4$

$$s = \texttt{ACTTATCA}, \qquad \star(s) = (1, 8, 2)$$
$$t = \texttt{ACTATC}, \qquad \star(t) = (2, 5, 2)$$
$$J(s,t) \approx sim(\star(s), \star(t)) = \frac{1}{3}$$

algorytm MinHash - podsumowanie

Algorytm minHash:

- dla każdego łańcucha przechowuje sygnaturę, wektor m liczb całkowitych (zamiast k-spektrum)
- wiele funkcji skrótu poprzez funkcję `XOR`

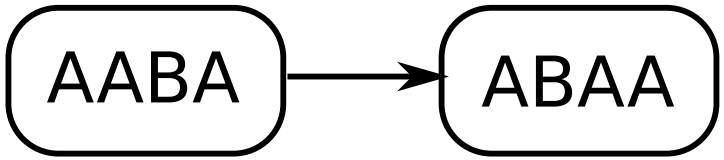
Dokładność algorytmu rośnie gdy:

- więcej funkcji skrótu
- krótsze k (dla k-merów)

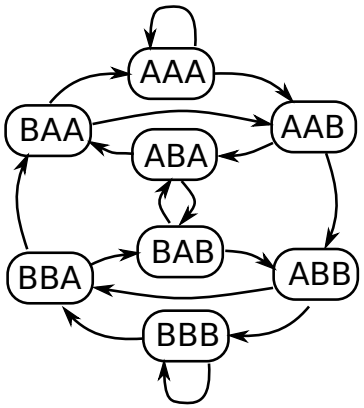
Asemblacja de novo. Pod-graf de Bruijna

Graf de Bruijna

n-wymiarowym grafem de Bruijna dla alfabetu **A** jest graf skierowany, którego wierzchołki odpowiadają słowom o długości $n-1$, natomiast krawędzie łączą wierzchołek $u = s_1 s_2 \ldots s_{n-1}$ z $v = s_2 \ldots s_{n-1} s_n$, gdzie $s_i \in \mathbf{A}$



Przykład grafu pełnego:
$\mathbf{A} = \{A, B\}$, $n = 4$



Asemblery wykorzystują podgraf grafu de Bruijna, ale mówi się skrótowo, że wykorzystują graf de Bruijna

Graf de Bruijna (2)

Aplikacje wykorzystują podgraf grafu de Bruijna:

- nie wymagają oceny wszystkich par odczytów,
- wymagają wyznaczenia ścieżki Eulera (problem rozwiązywany wielomianowo).

Ścieżka Eulera- przechodzi przez każdą krawędź grafu dokładnie raz

Złożoność: algorytm Fleury'ego, $O(|E|^2)$, alg. Hierholzer'a $O(|E|)$

aplikacje[2]: Euler, Velvet, ABySS, AllPaths, SOAPdenovo

[2] J.Miller and oth, Assembly algorithms for next-generation sequencing data

Asemblacja w pod-grafie de Bruijn

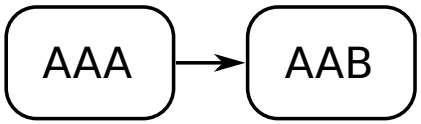
Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

Asemblacja w pod-grafie de Bruijn

Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$



`AAABB:  AAAB`

Asemblacja w pod-grafie de Bruijn

Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

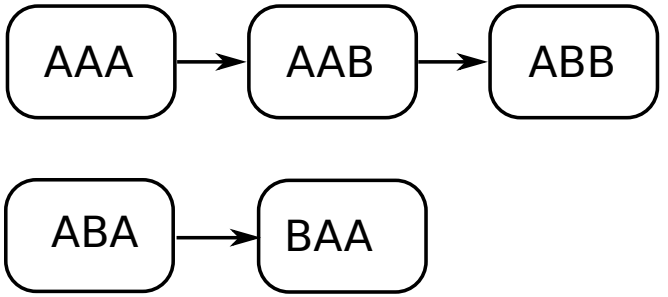
`AAABB`: `AAAB`, `AABB`

Asemblacja w pod-grafie de Bruijn

Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

`AAABB`: `AAAB`, `AABB`
`ABAAB`: `ABAA`

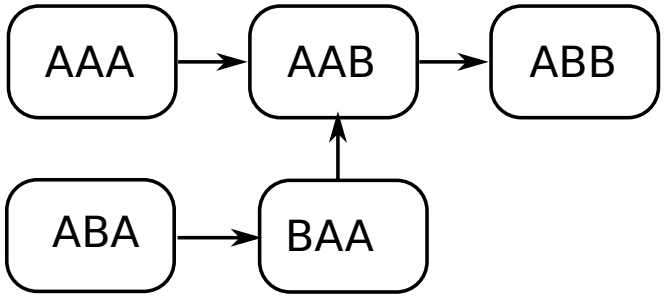


Asemblacja w pod-grafie de Bruijn

Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

`AAABB`: `AAAB`, `AABB`
`ABAAB`: `ABAA`, `BAAB`

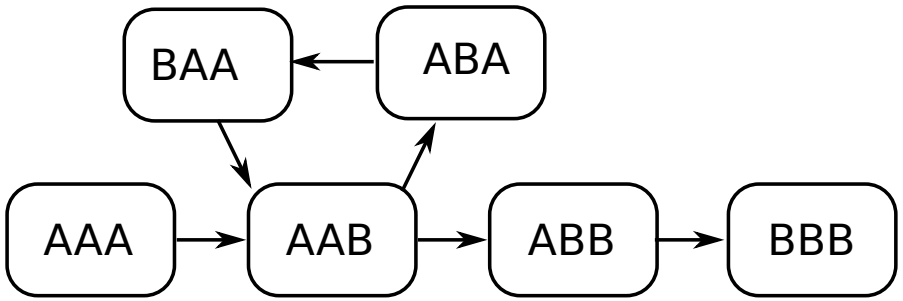


Asemblacja w pod-grafie de Bruijn

Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

`AAABB`: `AAAB`, `AABB`
`ABAAB`: `ABAA`, `BAAB`
`AABBB`: `AABB`, `ABBB`
`BAABA`: `BAAB`, `AABA`

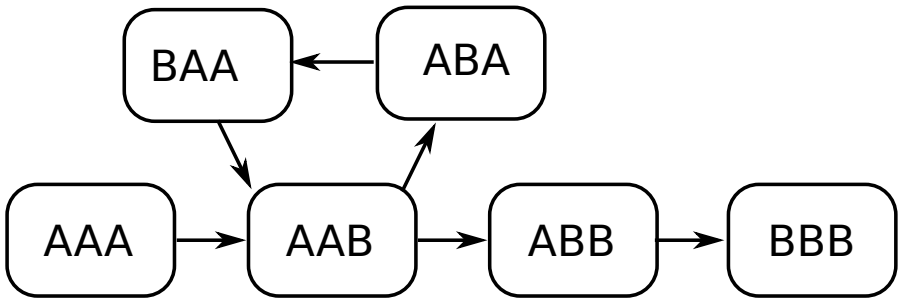


Asemblacja w pod-grafie de Bruijn

Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

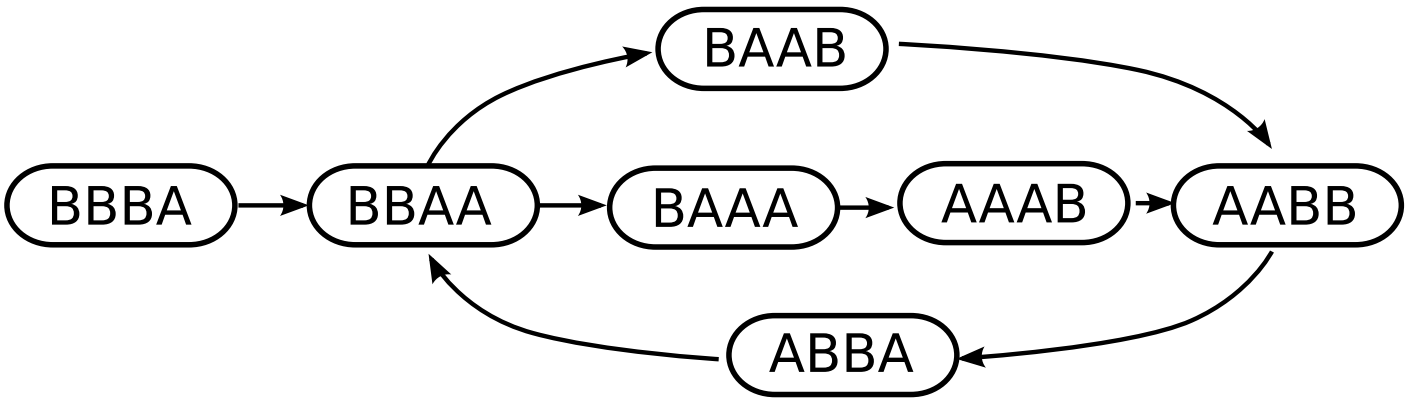
`AAABB`: `AAAB`, `AABB`
`ABAAB`: `ABAA`, `BAAB`
`AABBB`: `AABB`, `ABBB`
`BAABA`: `BAAB`, `AABA`



Wynik: `AAABAABBB`

Składanie w multi-grafie de Bruijna

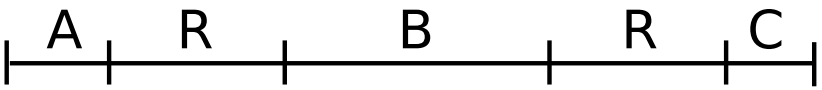
Dla zestawu bezbłędnych odczytów: `AAABB`, `AABBA`, `ABBAA`, `BAAAB`, `BAABB`, `BBAAA`, `BBAAB`, `BBBAA`, zbuduj multi-graf de Bruijna 5 rzędu, a następnie podaj wynikową sekwencję.



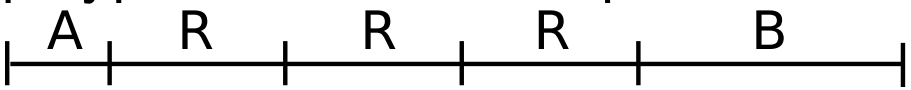
Wynik: `BBBAABBAAABB` lub `BBBAAABBAABB`

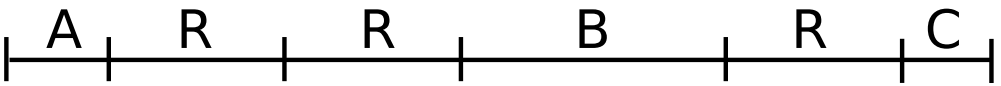
Sekwencje powtarzające się dłuższe niż rząd grafu

- sytuacje odtwarzane poprawnie przez typowy algorytm

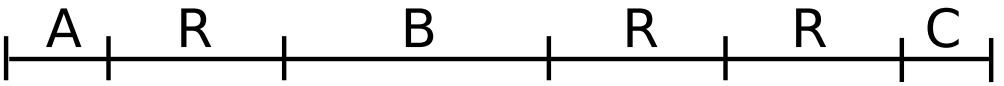


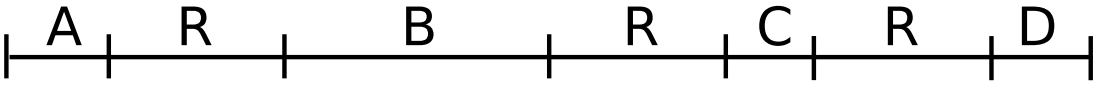
- przypadki odtwarzane przez multi-graf de Bruijna



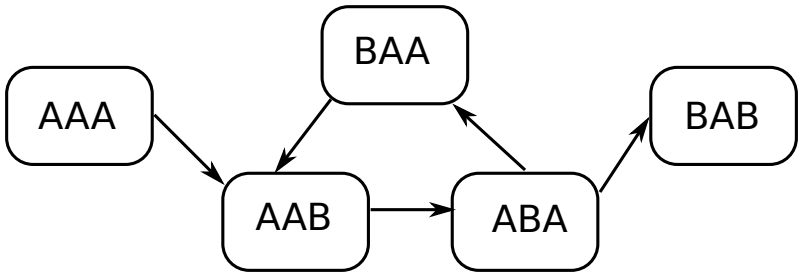


- przypadki kłopotliwe



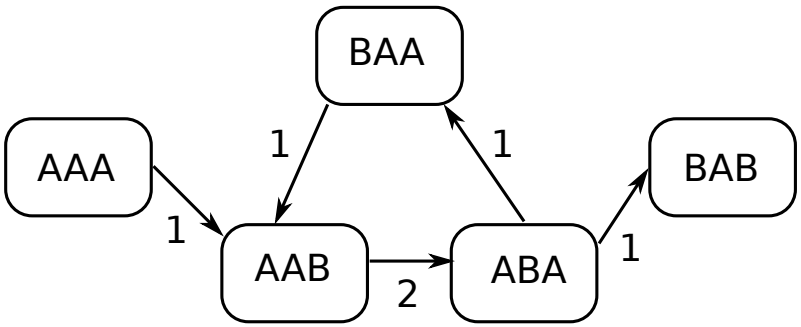


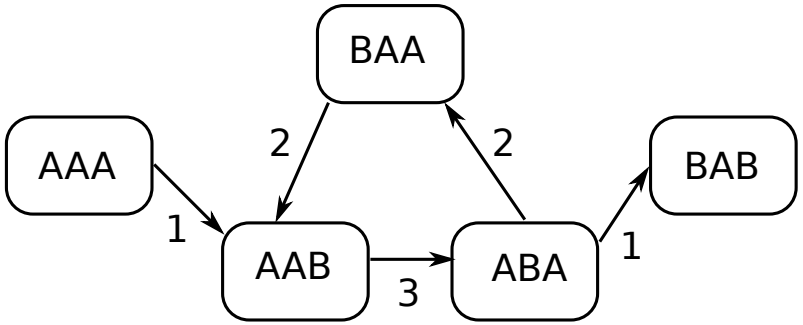
Multi-graf de Bruijna



reprezentuje sekwencje:
AAABAABAB, AAABAABAABAB,
AAABAABAABAABAB, ...

dodatkowe założenie: pozycje początkowe dla odczytów mają rozkład jednostajny





AAABAABAB (po lewej) and AAABAABAABAB (po prawej)

Programy komputerowe:
EULER+, dnaasm.sourceforge.net

Odtwarzanie powtórzeń motywów

Most - krawędź pomiędzy silnie spójnymi składowymi.

- Pozycja mostów jest taka sama na wszystkich ścieżkach Eulera dla danego grafu;
- wierzchołek z dokładnie 2 krawędziami wyjściowymi, gdy jedna z nich jest mostem, pozwala jednoznacznie odtworzyć ścieżkę Eulera.

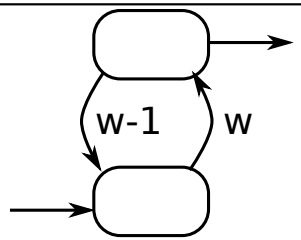
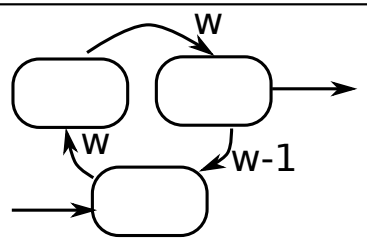
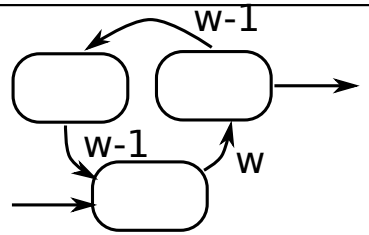
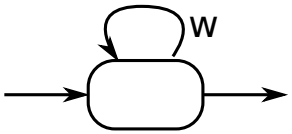
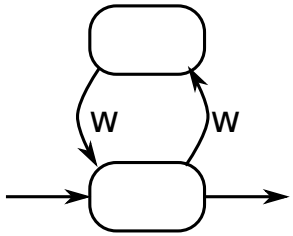
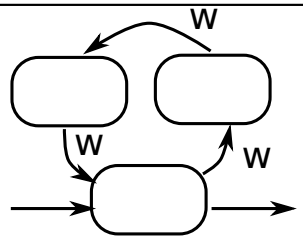
Sekwencja powtarzająca się S, $|S| = n$, $S = m_0 \ldots m_{d-1} m_0 \ldots m_{(n-1) \bmod d}$, motyw $m = m_0 m_1 \ldots m_{d-1}$, $|m| = d < n$.

Przykłady:

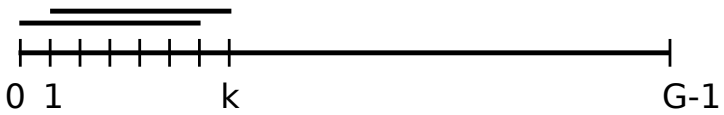
- `ACGTACGTACG`, $d = 4, n = 11$,
- `AAAAAAA` $(d = 1, n = 7)$.

Powtórzenia motywów w multi-grafach de Bruina

k rząd grafu, w waga krawędzi, d długość motywu
n długość sekwencji powtarzającej się, $n > k$

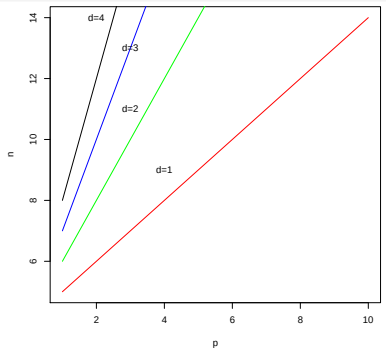
	$d = 1$	$d = 2$	$d = 3$
$n - k \equiv 0 \pmod{d}$			
$n - k \equiv 1 \pmod{d}$			
$n - k \equiv d - 1 \pmod{d}$			

K-spectrum jako wejście asemblera



- dokładnie jeden odczyt z każdej pozycji
- długość odczytu równa rzędowi grafu k

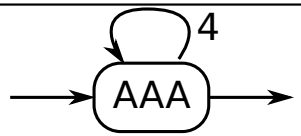
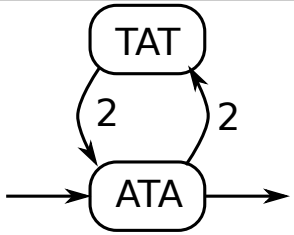
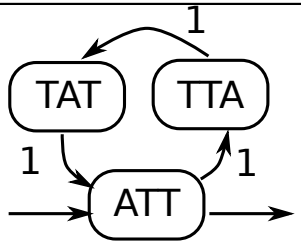
$$w = \frac{\Delta}{d}, \text{ gdzie } \Delta \equiv 0 \pmod{d}, \Delta = n - k + 1$$



k	rząd grafu
d	długość motywu
n	długość skewencji z powtórzeniami
w	waga krawędzi

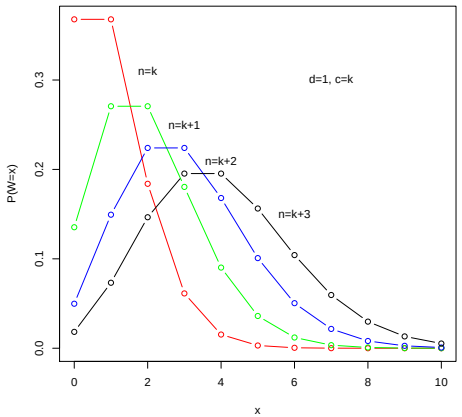
$$n = wd + k - 1, \text{ gdzie } n - k \equiv d - 1 \pmod{d}$$

Multigraf de-Bruijna dla k-spektrum, przykłady

sequence	parameters	graph
AAAAAAA	$k = 4, d = 1, n = 7, w = 4$	
ATATATA	$k = 4, d = 2, n = 7, w = 2$	
ATTATT	$k = 4, d = 3, n = 6, w = 1$	

Wagi krawędzi dla jednostajnego rozkładu fragmentów

W to zmienna losowa, waga krawędzi wewnątrz sekwencji reprezentującej motyw.



k	rząd grafu
G	długość genomu
L	długość fragmentu, tutaj $L \geq k$
N	liczba fragmentów
c	nadm. sekwencjonowania $c = \frac{LN}{G}$
d	długość motywu
n	dł. sekw. z powtórzeniami, $n > k$

$$W \sim B(N', p), N' = N(L - k + 1), p = \frac{\Delta}{dG}, \Delta = n - k + 1$$

$p \ll 1$ $(G \gg 1)$, $N \gg 1, L \geq k$

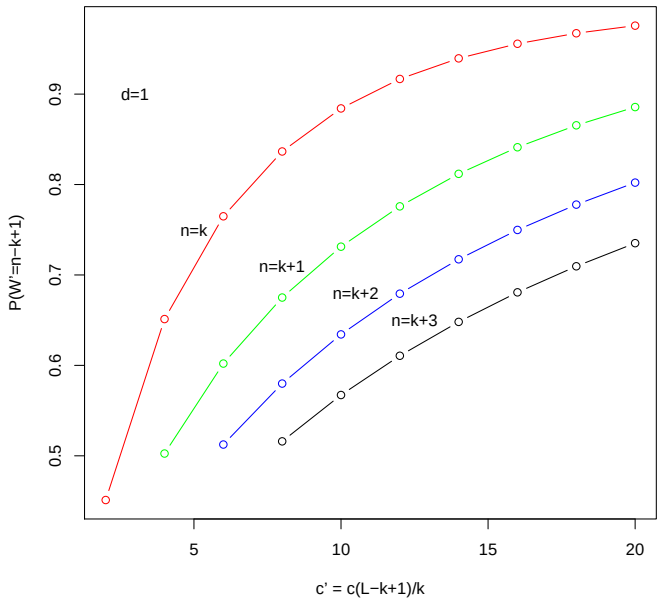
$$W \sim \text{Poisson}(\lambda), \lambda = \frac{c'\Delta}{d}, c' = \frac{c(L - k + 1)}{k}, \Delta = n - k + 1$$

Estymacja błędu normalizacji wag krawędzi

$$w' = \lfloor \frac{c(L-k+1)}{k} w + 0.5 \rfloor, \text{ gdzie } L \geq k$$

$$P(W' = x) = \Phi_P(\lambda + \frac{c'}{2}, \lambda) - \Phi_P(\lambda - \frac{c'}{2}, \lambda)$$

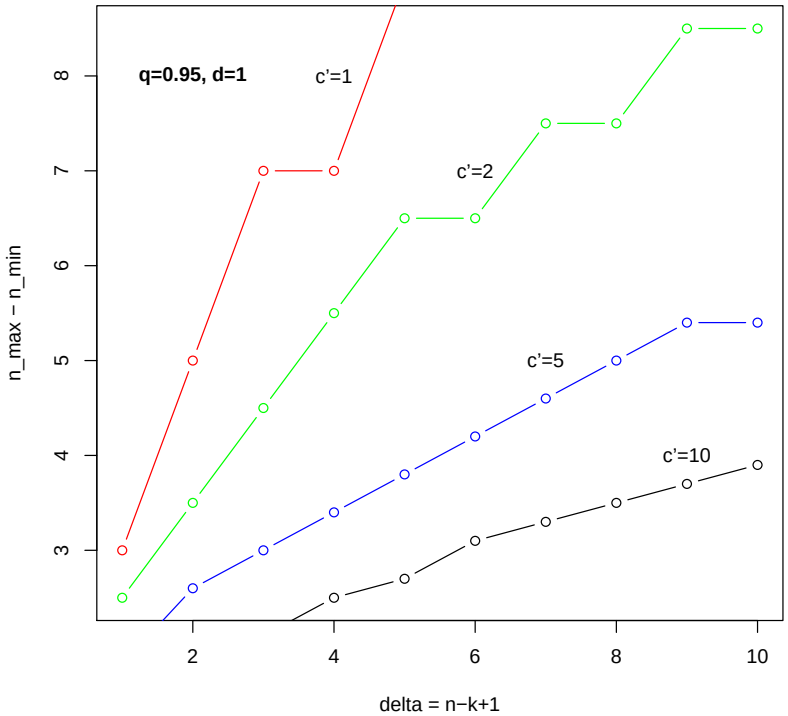
$\Phi_P(x, \lambda)$ jest dystrybuantą rozkładu Poissona

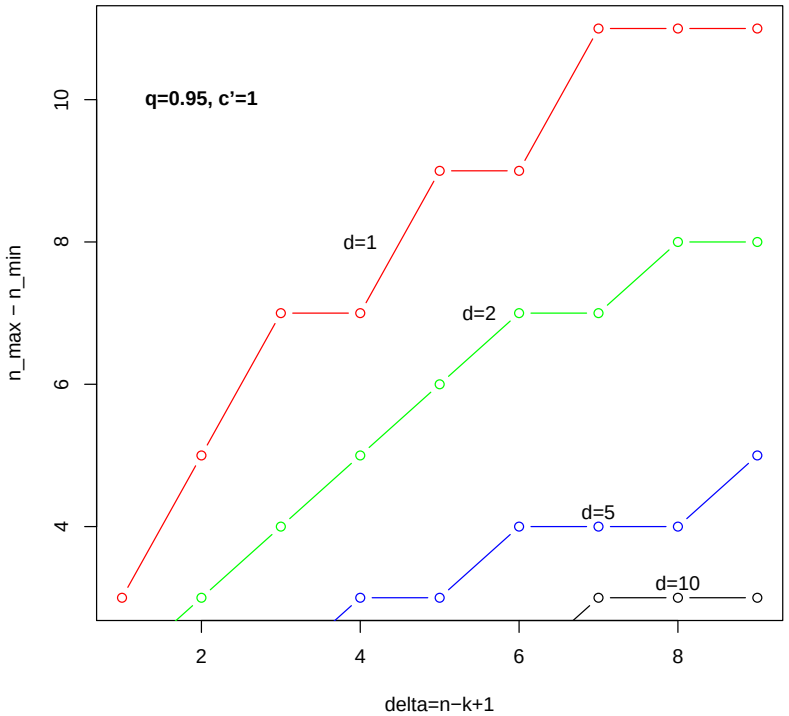


k	rząd grafu
G	długość genomu
L	długość fragmentu, tutaj $L \geq k$
N	liczba fragmentów
c	nadm. sekwencjonowania $c = \frac{LN}{G}$
c'	nadm. dla krawędzi $c' = \frac{c(L-k+1)}{k}$
d	długość motywu
n	dł. sekw. z powtórzeniami, $n > k$

Estymacja błędu normalizacji wag krawędzi (2)

$$\hat{n}_{max} = \frac{1}{c'}\Phi_P^{-1}\left(\frac{1+q}{2}, \lambda\right)$$
$$\hat{n}_{min} = \frac{1}{c'}\Phi_P^{-1}\left(\frac{1-q}{2}, \lambda\right)$$

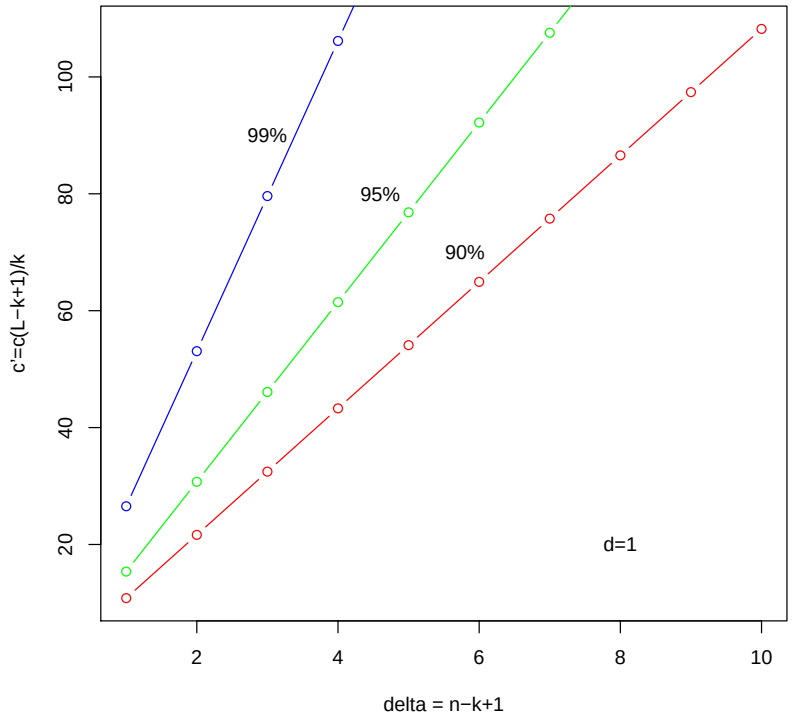


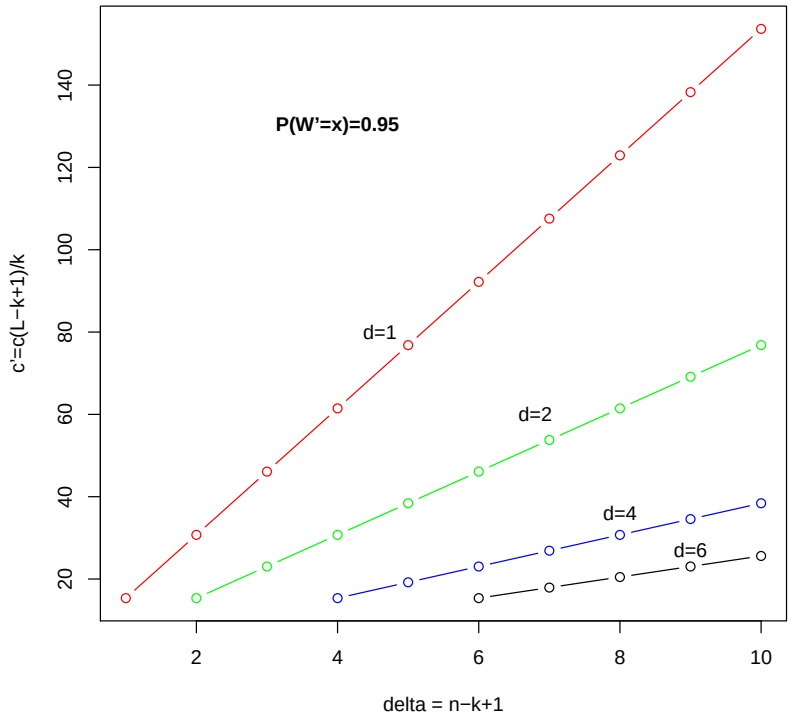


Wymagana nadmiarowość sekwencjonowania

Ponieważ $P(W' = x) \geq 0.9$ dla $c' > 10$

$$W' \sim \mathcal{N}(\mu, \sigma) \text{ gdzie } \mu = \frac{\Delta}{d}, \sigma = \sqrt{\frac{\Delta}{d}}, \Delta = n - k + 1$$





Wymagana nadmiarowość (2) - przykłady

poziom ufności $= 95\%$

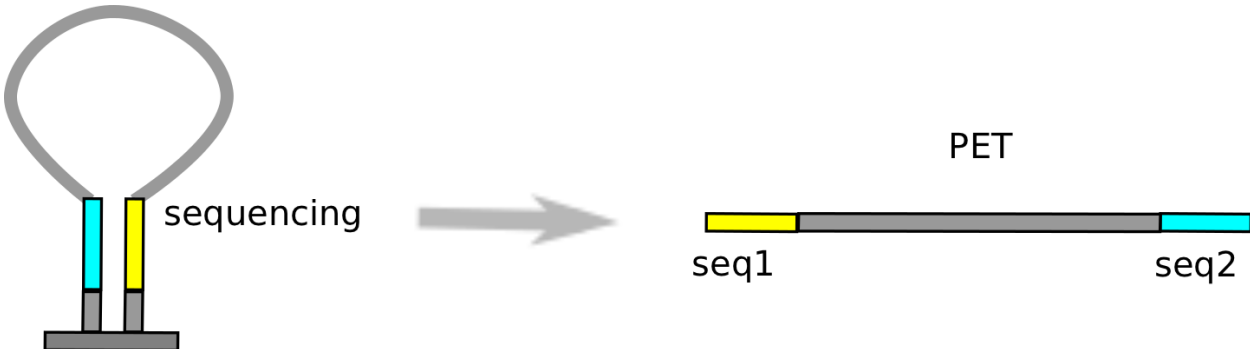
Wymagana nadmiarowość sekwencjonowania c':

	$d = 1$	$d = 2$	$d = 5$	$d = 10$
$\Delta = 1$	15.4	-	-	-
$\Delta = 2$	30.7	15.4	-	-
$\Delta = 5$	76.8	38.4	15.4	-
$\Delta = 10$	154	76.8	30.7	15.4
$\Delta = 100$	1537	768	307	30.7
$\Delta = 1000$	15366	7683	3073	307

Przykład: $L = 80$, $k = 50$, $d = 5$, $n = 150 \rightarrow c' \geq 307$, $c \geq 495$

Algorytmy dla sparowanych końców

- znana długość (np. 3000 nt)
- znane sekwencje na obu końcach



algorytmy osiągają wyniki jak dla odczytów o długości cząsteczki

Miary jakości asemblacji

Miary jakości asemblacji

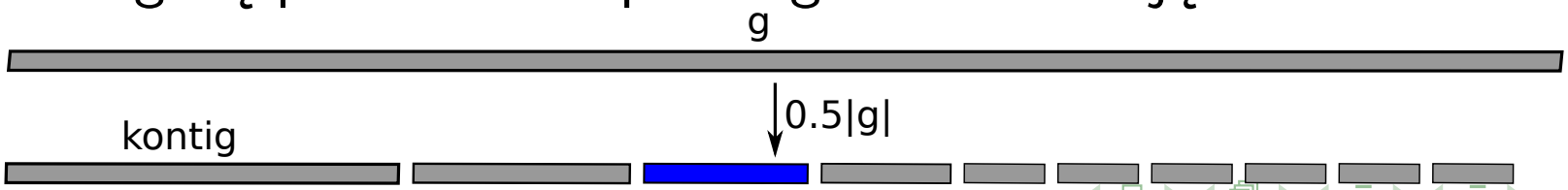
Wynik asemblacji genomu g to zbiór n kontigów $\{c_0, c_1, \ldots, c_{n-1}\}$

Typowe miary jakości asemblacji:

- n – liczba kontigów (im mniej tym lepiej)
- s lub $\frac{s}{|g|}$ – suma długości kontigów dłuższych niż ustalony próg t (typowo $t = 1000nt$)

$$s = \sum_{i=0}^{n-1} |c_i|, \text{dla } |c_i| \geq t$$

- $N50$ – długość kontiga, w którym jest symbol o indeksie $\frac{|g|}{2}$, kontigi są posortowane po długościach malejąco



Wykańczanie

Kontigi sekwencyjne i kontigi fizyczne (ang. *scaffold)*

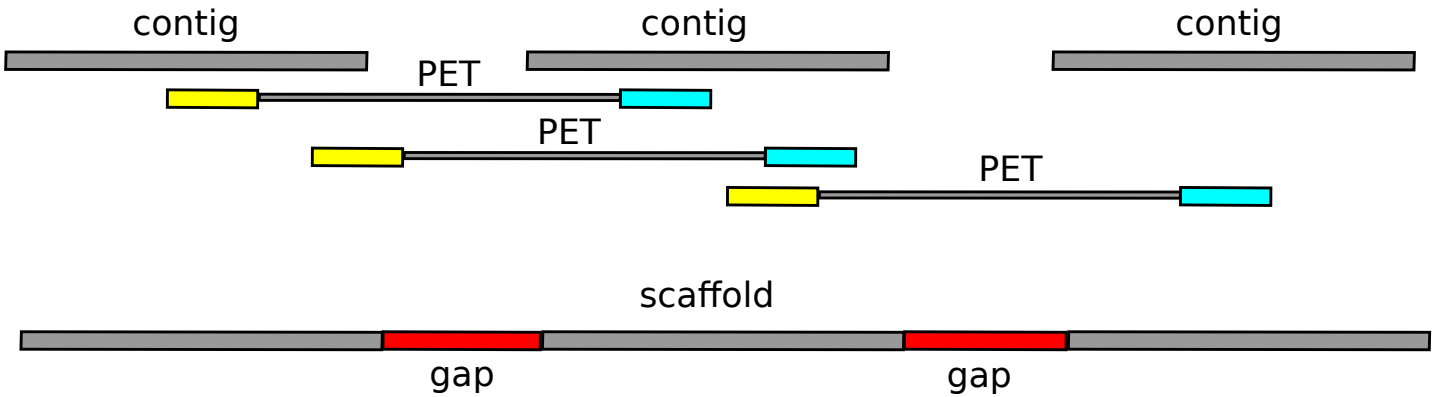
Kontig, kontig sekwencyjny - ciągła sekwencja uzyskana ze złożenia nachodzących na siebie fragmentów

Powody uzyskiwania dziur:

- sekwencje powtarzające się dłuższe niż odczyty
- niepełne pokrycie badanej cząsteczki losowymi fragmentami

Dodatkowa informacja pozwala ustalić orientację kontigów i je uszeregować, uzyskujemy kontig fizyczny

Wykańczanie (scaffolding), kontigi fizycznych



kontig fizyczny (ang. scaffold) - sekwencja (niekoniecznie ciągła) uzyskana za pomocą sekwencji sparowanych końców

- wykorzystywane algorytmy przeszukiwania przestrzeni z ograniczeniami (CSP – Constraint satisfaction problems)
- wykorzystanie algorytmów łączonych

Algorytmy łączone

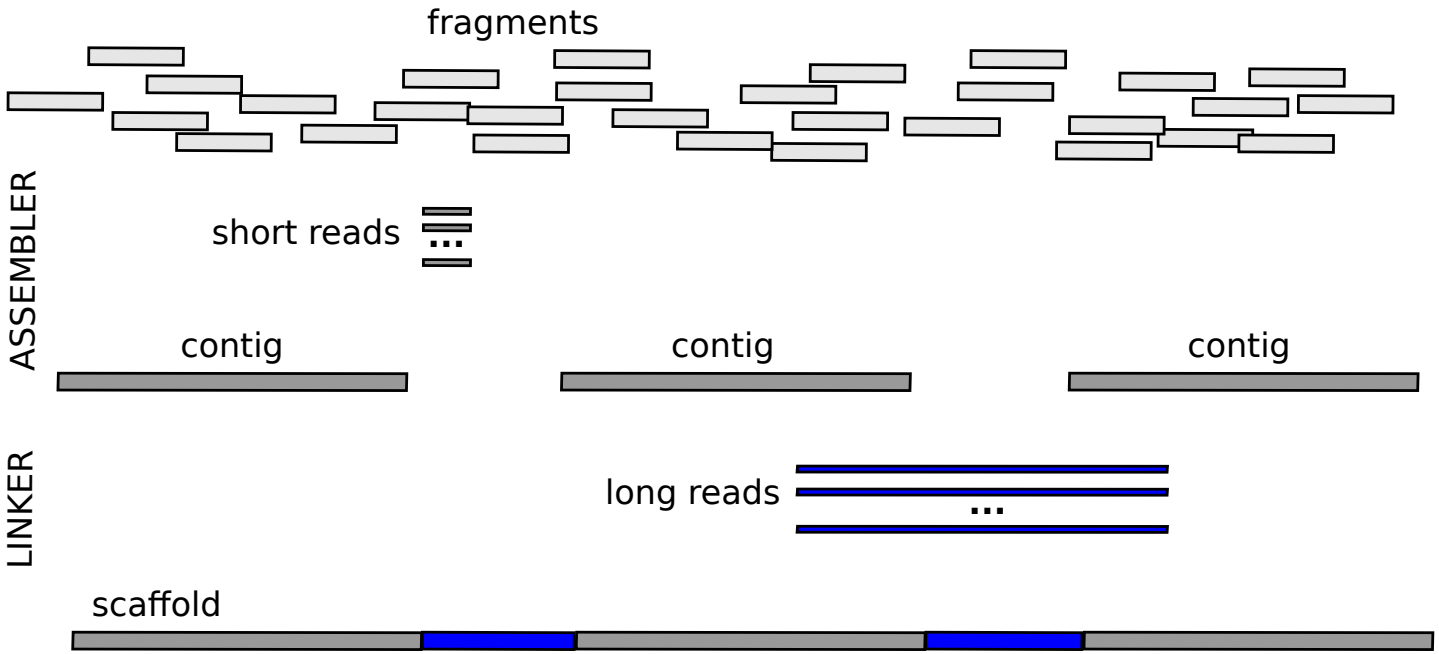
Algorytmy łączone: wykorzystują odczyty II generacji (krótkie, wysoka jakość) i III genreracji (długie, niska jakość)

- korekta długich odczytów przez krótkie odczyty, a później asemblacja
- korekta kontigów z krótkich odczytów za pomocą długich odczytów
- asemblacja długich odczytów, korekta kontigów przez krótkie odczyty
- **asemblacja krótkich odczytów, łączenie kontigów za pomocą długich odczytów**

Obecnie podstawowe zagadnienie w rozwoju assemblerów DNA

Algorytmy łączone (asemblacja hybrydowa)

Metoda łączenia kontigów (wynik semblacji krótkich odczytów) za pomocą długich odczytów



Przykład użycia: asemblacja *de-novo Hymenolepsis dimiuta*

	pokrycie	długość odczytu		długość wstawki	
		mediana	średnia	średnia	std dev [bp]
PET1	156x	100	100	434	88
PET2	340x	100	100	438	92
MP1	80x	100	100	6183	2748
MP2	62x	100	100	6435	2716
ONT1	20x	3762	6378		
ONT2	9x	6073	10116		

dane	kontigi	N50 [kbp]	sum [Mbp]
PET1	8289	43.5	159.2
PET1+2	8194	45.2	162.1
PET12+ MP1	2346	840.6	170.7
PET12+ MP1+2	2342	844.2	170.8
PET12+ MP12+ONT1	815	1770.0	176.6
all	719	2537.0	177.3

Składanie sekwencji - podsumowanie

- uwzględnienie błędnych sekwencji zwiększa złożoność problemu
- stosuje się dodatkowe odczyty (inny rodzaj doświadczenia, sparowane końce) do łączenia kontigów
- problemem są długie sekwencje powtarzające się (dłuższe niż odczyt)
- techniki sekwencjonowania 2 generacji dostarczają dużej ilości odczytu
- istniejące algorytmy wymagają dużej ilości pamięci (64GB) i mają wiele parametrów

Resekwencjonowanie

Resekwencjonowanie - etapy

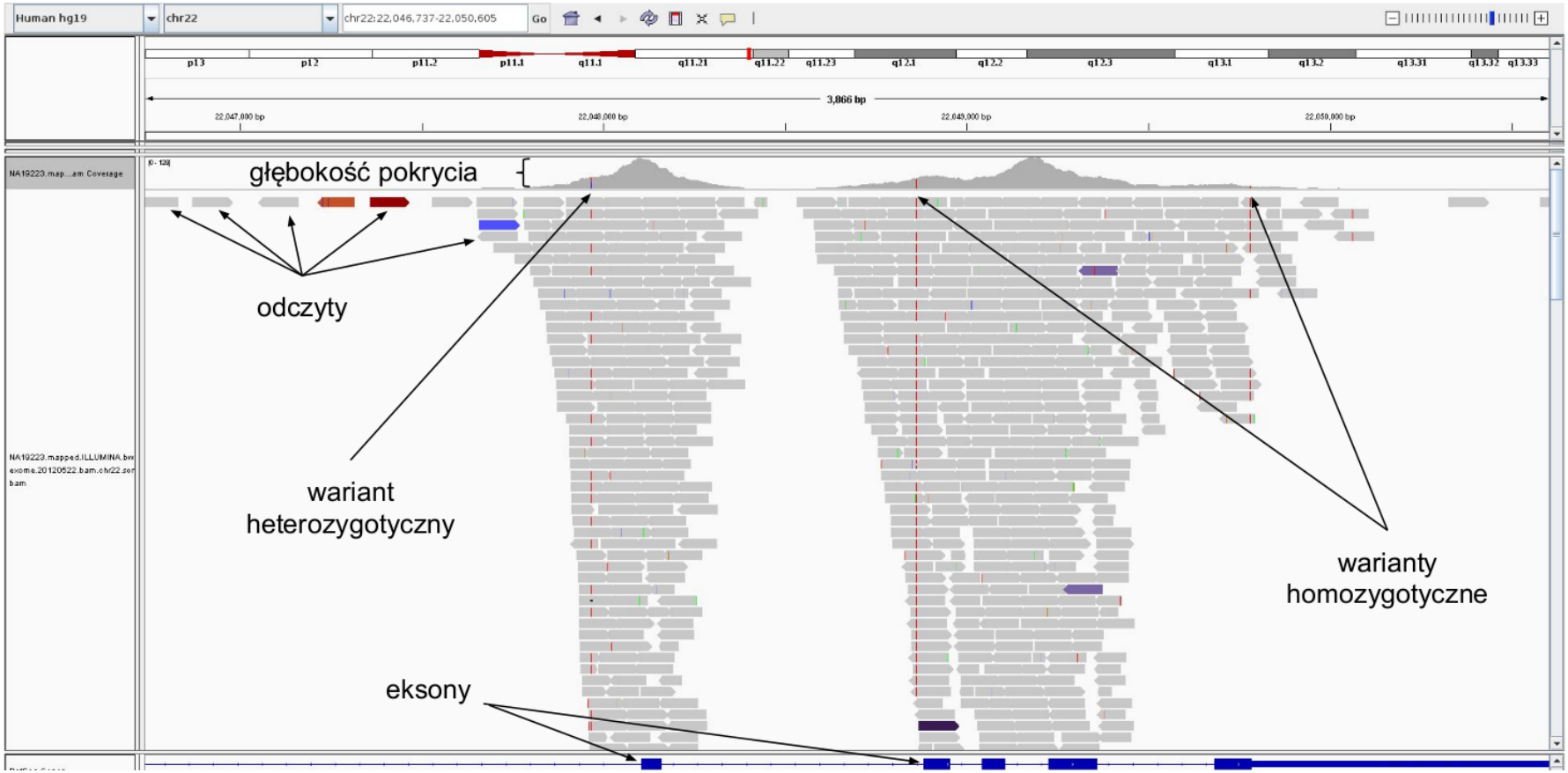
Resekwencjonowanie:

- mapowanie odczytów na genom referencyjny
- oznaczanie duplikatów
- rekalibracja oceny jakości nukleotydów

Później - analiza wariantów

- możliwe analizy dla nadmiarowości (coverage) < 1
- możliwe analizy dla części genomu (np. exony)

Mapowanie odczytów na genom referencyjny



Plik SAM (Sequence Alighment/Map)

mapowanie wykorzystuje transofmatę Burrowsa-Wheelera

plik tekstowy (tabela), zawiera informację o mapowaniu odczytów na genom referencyjny, zawiera:

- identyfikator odczytu
- identyfikator kontiga (na który odczyt jest mapowany)
- pozycja (indeks) mapowania
- jakość mapowania, flagi, dodatkowe informacje (np. insercje i delecje)

Plik BAM: kompresja pliku SAM

transformata Burrowsa-Wheelera

BTW używana w BZIP2, bardzo wydajna do przechowywania sekwencji biologicznych. Znajduje korelacje pomiędzy symbolami.

- generowanie wszystkich rotacji kompresowanego bloku, np. dla napisu 'bioinformatyka'

0	b	i	o	i	n	f	o	r	m	a	t	y	k	a
1	a	b	i	o	i	n	f	o	r	m	a	t	y	k
2	k	a	b	i	o	i	n	f	o	r	m	a	t	y
3	y	k	a	b	i	o	i	n	f	o	r	m	a	t
4	t	y	k	a	b	i	o	i	n	f	o	r	m	a
5	a	t	y	k	a	b	i	o	i	n	f	o	r	m
6	m	a	t	y	k	a	b	i	o	i	n	f	o	r
7	r	m	a	t	y	k	a	b	i	o	i	n	f	o
8	o	r	m	a	t	y	k	a	b	i	o	i	n	f
9	f	o	r	m	a	t	y	k	a	b	i	o	i	n
10	n	f	o	r	m	a	t	y	k	a	b	i	o	i
11	i	n	f	o	r	m	a	t	y	k	a	b	i	o
12	o	i	n	f	o	r	m	a	t	y	k	a	b	i
13	i	o	i	n	f	o	r	m	a	t	y	k	a	b

transformata Burrowsa-Wheelera (2)

- sortowanie wierszy - jeżeli są korelacje, to grupowane są symbole w ostatniej kolumnie

i	i_0														
0	1	a	b	i	o	i	n	f	o	r	m	a	t	y	k
1	5	a	t	y	k	a	b	i	o	i	n	f	o	r	m
2	0	b	i	o	i	n	f	o	r	m	a	t	y	k	a
3	9	f	o	r	m	a	t	y	k	a	b	i	o	i	n
4	11	i	n	f	o	r	m	a	t	y	k	a	b	i	o
5	13	i	o	i	n	f	o	r	m	a	t	y	k	a	b
6	2	k	a	b	i	o	i	n	f	o	r	m	a	t	y
7	6	m	a	t	y	k	a	b	i	o	i	n	f	o	r
8	10	n	f	o	r	m	a	t	y	k	a	b	i	o	i
9	12	o	i	n	f	o	r	m	a	t	y	k	a	b	i
10	8	o	r	m	a	t	y	k	a	b	i	o	i	n	f
11	7	r	m	a	t	y	k	a	b	i	o	i	n	f	o
12	4	t	y	k	a	b	i	o	i	n	f	o	r	m	a
13	3	y	k	a	b	i	o	i	n	f	o	r	m	a	t

- wyjście: ostatnia kolumna, czyli $L =$'kmanobyriifoat' oraz indeks pierwszego znaku p, tutaj $p = 5$
- złożoność: $O(N \log N)$

transformata Burrowsa-Wheelera (3) - dekompresja

- sortowanie L daje pierwszą kolumnę F, tzn:

L	k	m	a	n	o	b	y	r	i	i	f	o	a	t
F	a	a	b	f	i	i	k	m	n	o	o	r	t	y

- budujemy wektor sąsiedztw T z ostatniej kolumny (wejście do dekompresji) i pierwszej kolumny (wektor po sortowaniu):

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13
$T[i]$	2	12	5	10	8	9	0	1	3	4	11	7	13	6

- odtwarzamy ciąg z wektora L oraz T oraz indeksu pierwszego znaku p

```
for(int c = 0, i = P; c < SIZE; ++c, i = T[i]) { //c - licznik
  cout << L[i];
}
```

Plik SAM - przykład

Obok podano fragment pliku SAM dla sekwencji referencyjnej ACTGGGAC-CTA (indeks od 1). Wyznacz potencjalne warianty, podaj pokrycie i określ, czy jest on homo- czy heterozygotyczny.

POS	CIGAR	SEQ
2	5M1D1M	CTGGGC
4	3M1D3M	GTGCCT
5	2M1D4M	TGCCTA
5	2M1D4M	GGCCTA

A	C	T	G	G	G	A	C	C	T	A	
	C	T	G	G	G	*	C				5M1D1M
			G	T	G	*	C	C	T		3M1D3M
				T	G	*	C	C	T	A	2M1D4M
				G	G	*	C	C	T	A	2M1D4M

Plik SAM - przykład

Obok podano fragment pliku SAM dla sekwencji referencyjnej ACTGGGAC-CTA (indeks od 1). Wyznacz potencjalne warianty, podaj pokrycie i określ, czy jest on homo- czy heterozygotyczny.

POS	CIGAR	SEQ
2	5M1D1M	CTGGGC
4	3M1D3M	GTGCCT
5	2M1D4M	TGCCTA
5	2M1D4M	GGCCTA

A	C	T	G	G	G	A	C	C	T	A	
	C	T	G	G	G	*	C				5M1D1M
			G	T	G	*	C	C	T		3M1D3M
				T	G	*	C	C	T	A	2M1D4M
				G	G	*	C	C	T	A	2M1D4M

Sa dwa warianty:

- wariant heterozygotyczny na pozycji 5, G>T
- delecja homozygotyczny na pozycji 7, A>_

Dziękuję