Algorytmy w bioinformatyce sekwencji

Asemblacja

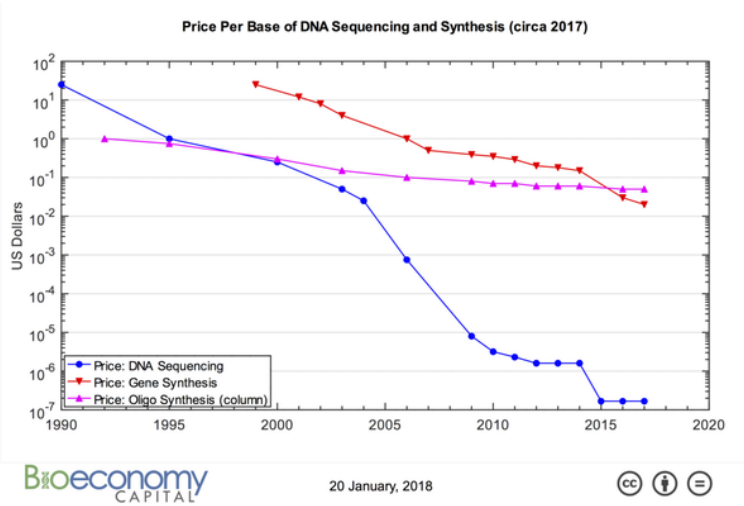
Robert Nowak

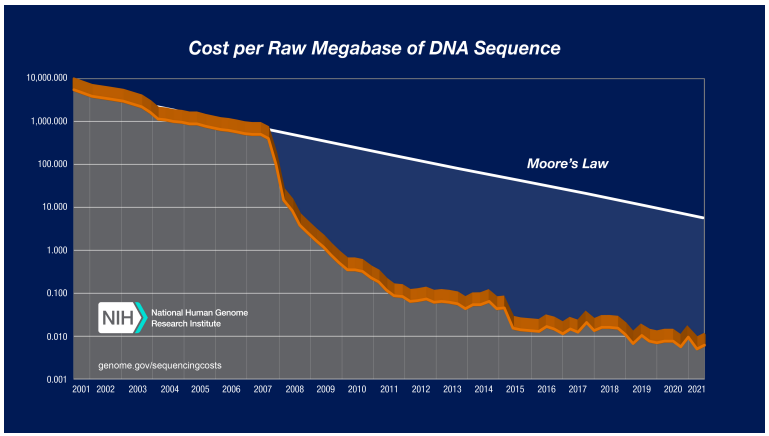
2024

Powtórzenie - sekwencjonowanie

Sekwencjonowanie DNA - proces ustalania kolejności nukleotydów tworzących cząsteczkę DNA.

- liczba poznawanych genomów podwaja się co 15 miesięcy
- wielkość sekwencjonowanych genomów podwaja się co 18 miesięcy
- koszty sekwencjonowania (na bp) zmniejszają się dwukrotnie co 18 miesięcy



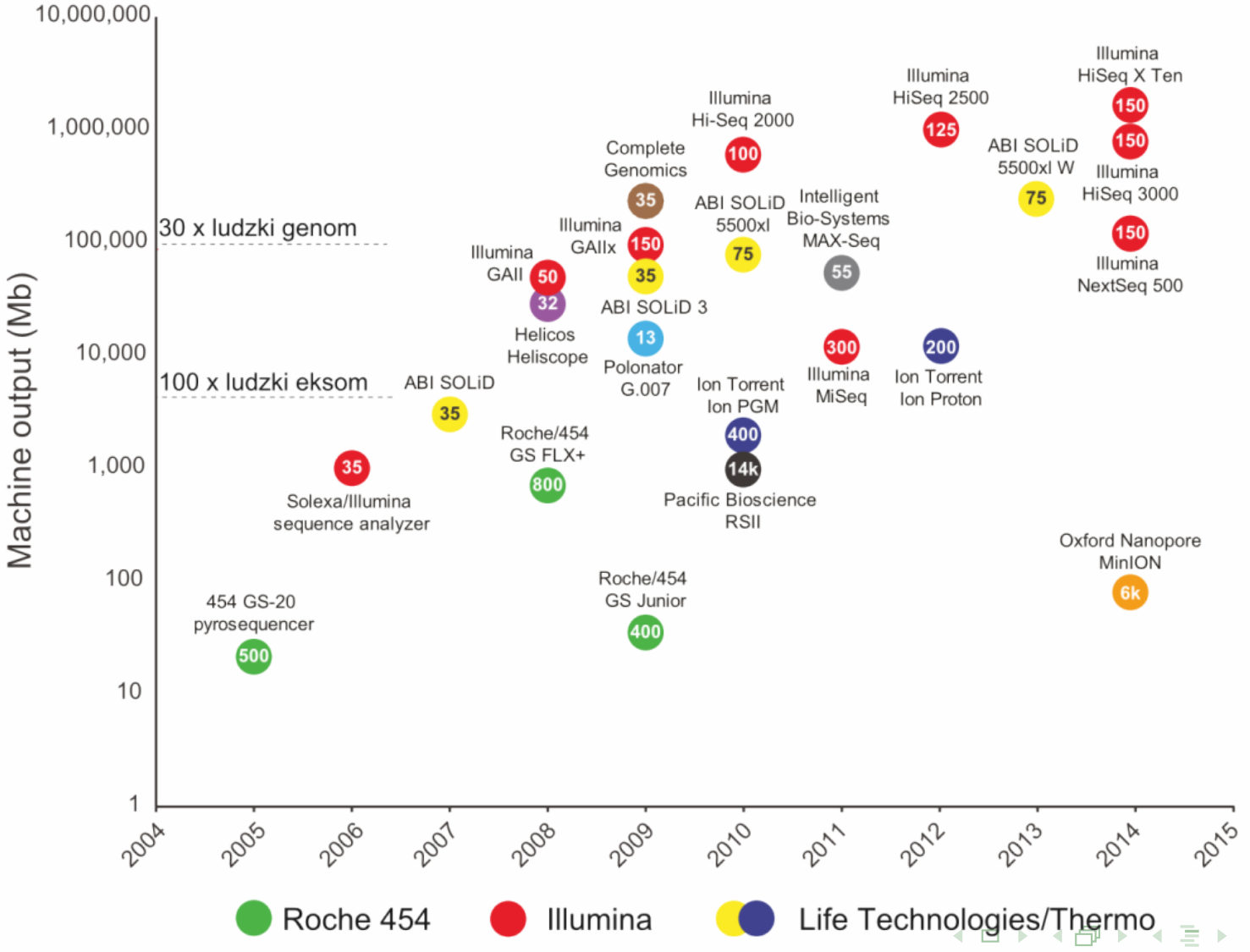


Sekwencjonowanie DNA - urządzenia

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%

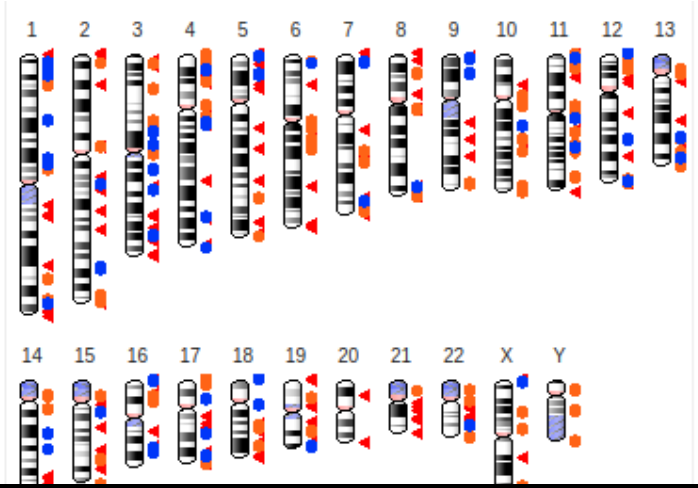
wzrost 12% rocznie, 4.5 mld. $ (2013), 6.6 (2016) 9.7 (2019)

Sekwenatory nowej generacji



Techniki sekwencjonowania

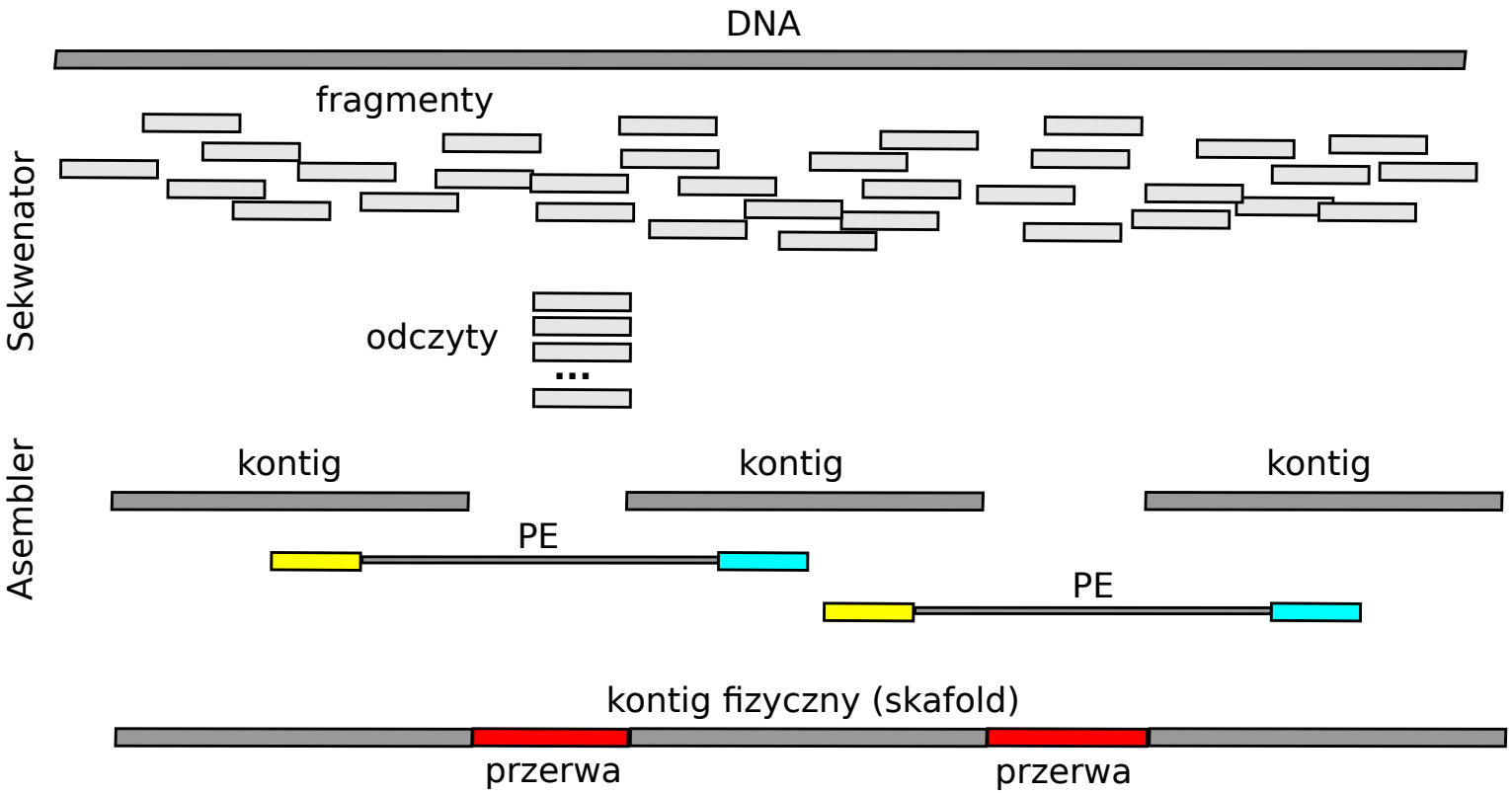
- *de novo*
  - hierarchiczne, np biblioteki klonów
  - losowe fragmenty (shotgun whole genome sequencing, WGS)
- resekwencjonowanie - wymagana sekwencja referencyjna
  - GRCh38 - genom referencyjny człowieka, major release (2013)
  - GRCh38 v.14 - bieżąca wersja genomu (Luty 2022), 434 gaps
  - GRCh38 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

Asemblacja de-novo

Odczyt sekwencji metodą shotgun (WGS)



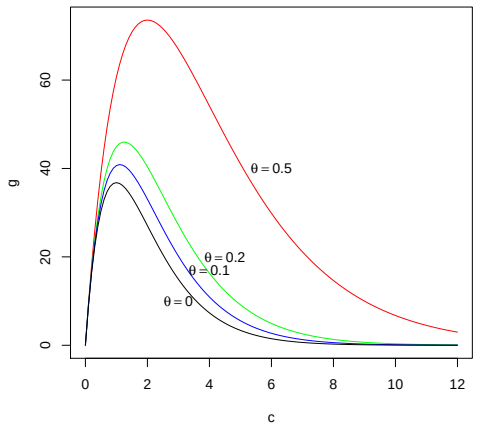
WGS, nieciągłość wyniku – niepełne pokrycie

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, ilość materiału w bibliotekach biologicznych

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

Algorytmy do składania sekwencji, asemblery

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

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

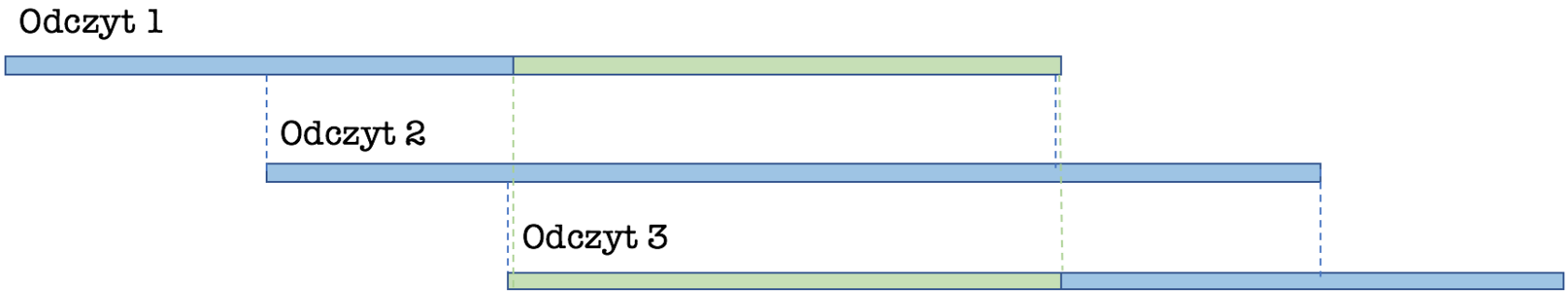
Ścieżka Hamiltona- przechodzi przez każdy wierzchołek grafu dokładnie raz

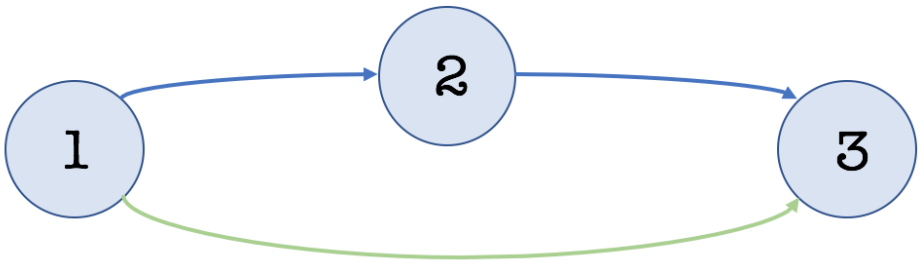
Złożoność: $O(|V|^k)$, gdzie k - maksymalna ilość krawędzi związanych z grafem

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

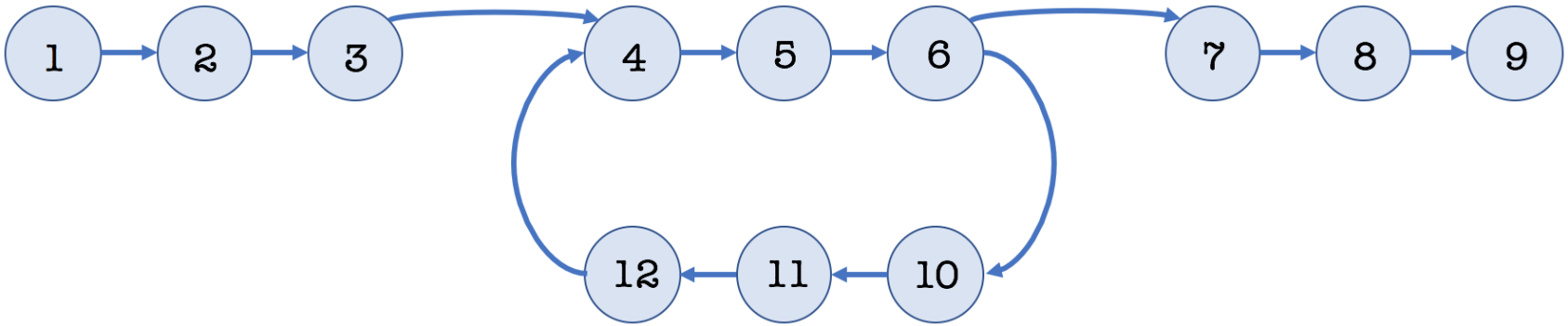
aplikacje: Celera, Arachne, CAP, PCAP, Newbler, CABOG,

Graf pokrycia - budowa grafu





Graf pokrycia - znajdowanie ścieżek



Graf pokrycia - znajdowanie sekwencji dla ścieżek

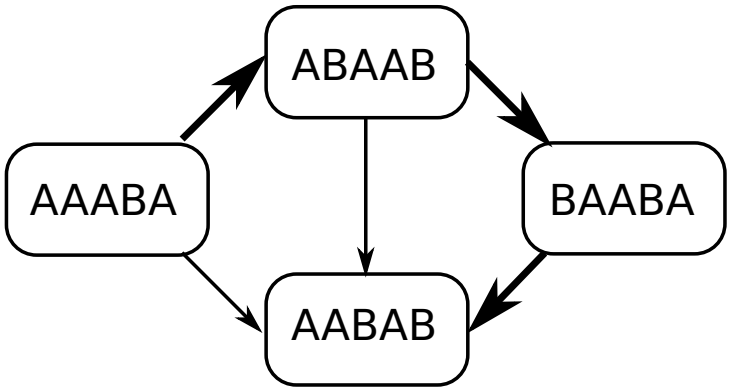
```
1. T A A T C T -- A C
2.       A T C -- G A C T T A
3.             C T G A C T -- A C C G
-------------------------------------
   T A A T C T G A C T T A C C G
```

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	-	-	-	-

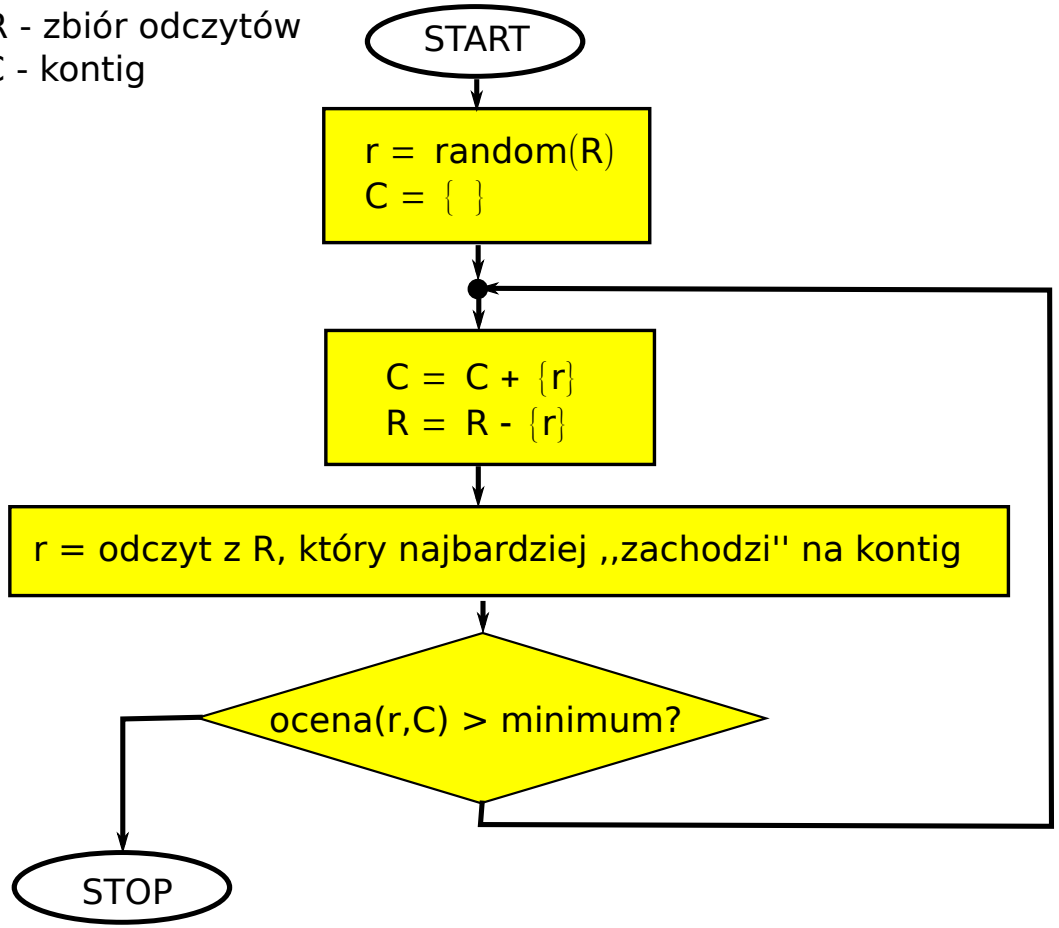



```
A  A  A  B  A  -  -  -  -
-  -  A  B  A  A  B  -  -
-  -  -  B  A  A  B  A  -
-  -  -  -  A  A  B  A  B
```

wynik: **AAABAABAB**

Algorytm zachłanny - jedna z odmian grafu pokrycia

we: R - zbiór odczytów
wy: C - kontig

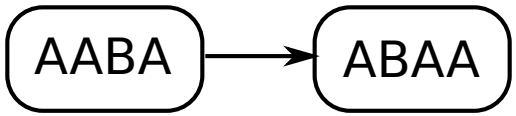


aplikacje: SSAKE, SHARCGS, VCAKE

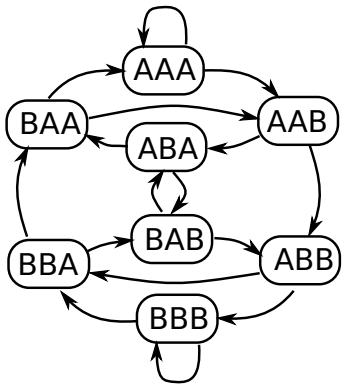
Asemblacja de novo, (pod)graf de Bruijna

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 de Bruijna, jest to graf pełny
$\mathbf{A} = \{A, B\}$, $n = 4$



asemblery wykorzystują podgraf grafu 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, $O(|E|^2)$, algorytm Hierholzer'a $O(|E|)$

aplikacje[1]: Euler, Velvet, ABySS, AllPaths, SOAPdenovo

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

Asemblacja w pod-grafie de Bruijn

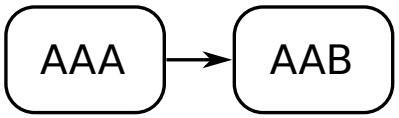
Nie musimy liczyć podobieństwa (nie analizujemy wszystkich par odczytów!), tworzenie grafu ma złożoność $O(N)$.

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`, $k = 4$

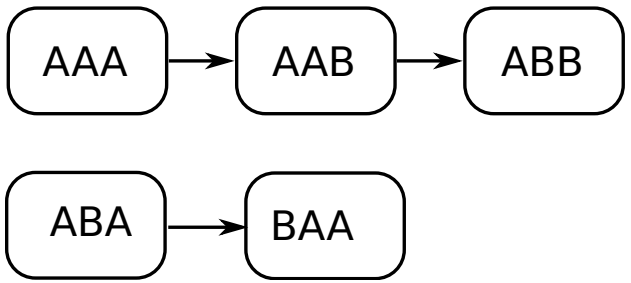
`AAABB`: `AAAB`, `AABB`

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`

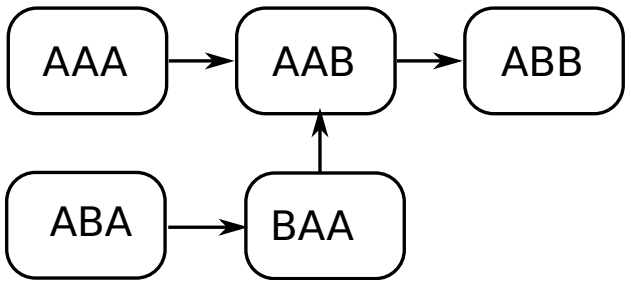


Asemblacja w pod-grafie de Bruijn

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`

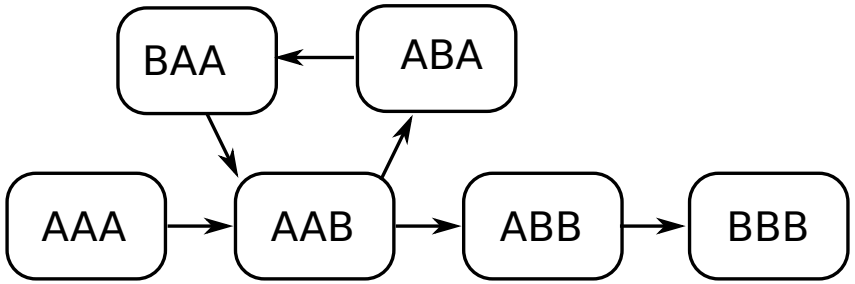


Asemblacja w pod-grafie de Bruijn

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
```

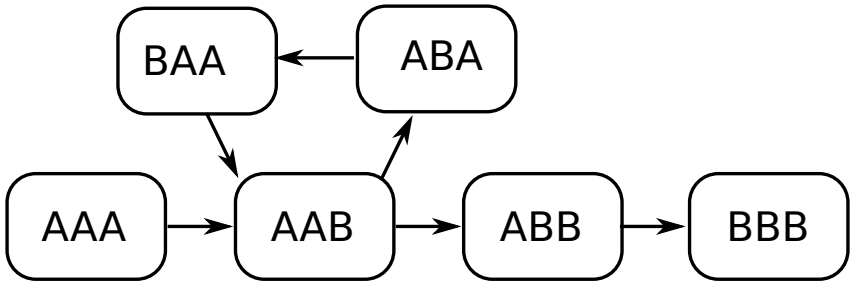


Asemblacja w pod-grafie de Bruijn

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

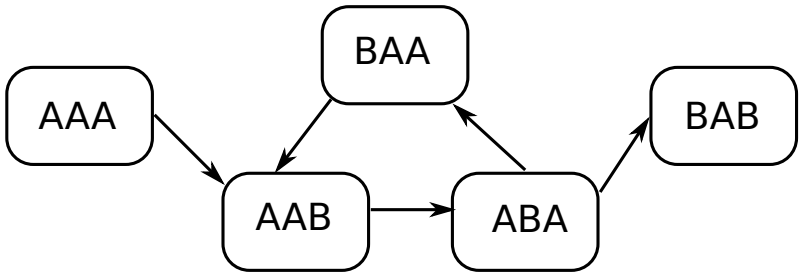
Przykład: odczyty: `AAABB`, `ABAAB`, `AABBB`, `BAABA`, $k = 4$

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



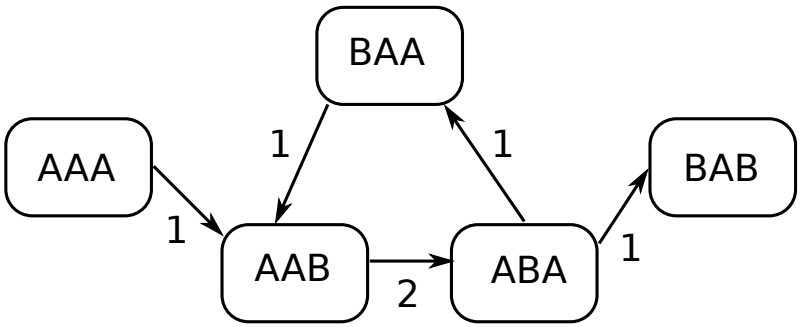
Wynik: `AAABAABBB`

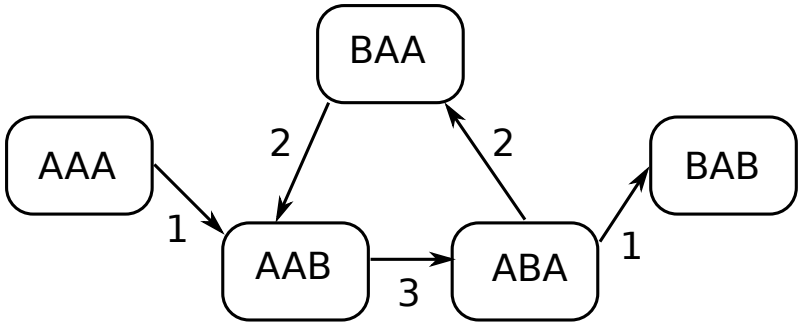
Multi-graf de Bruijna



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

dodatkowe założenie: pozycje początkowe dla odczytów mają rozkład jednostajny



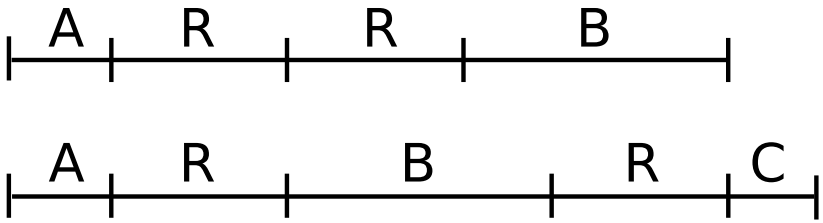


`AAABAABAB` (po lewej) and `AAABAABAABAB` (po prawej)

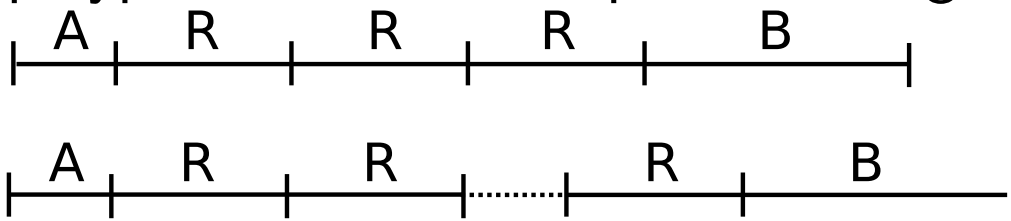
Programy komputerowe:
EULER+, dnaasm.sourceforge.net

Sekwencje powtarzające się dłuższe niż rząd grafu

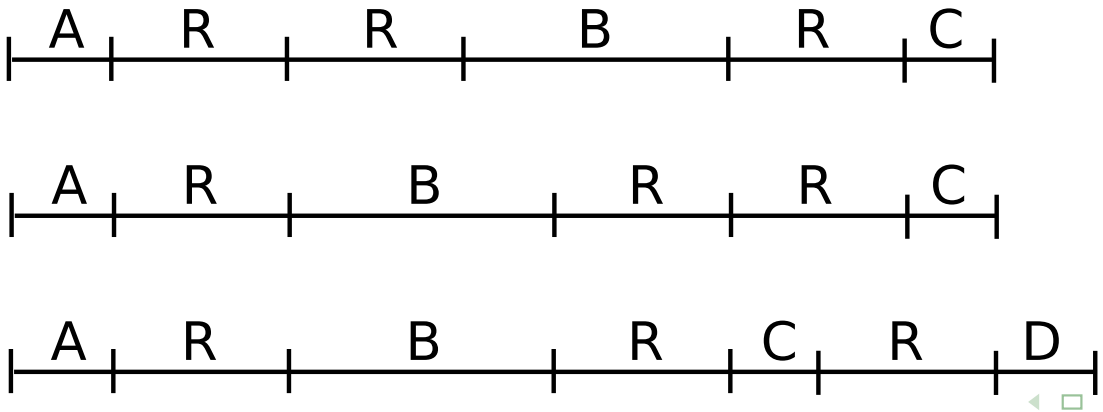
- sytuacje odtwarzane poprawnie przez typowy algorytm



- przypadki odtwarzane przez multi-graf de Bruijna

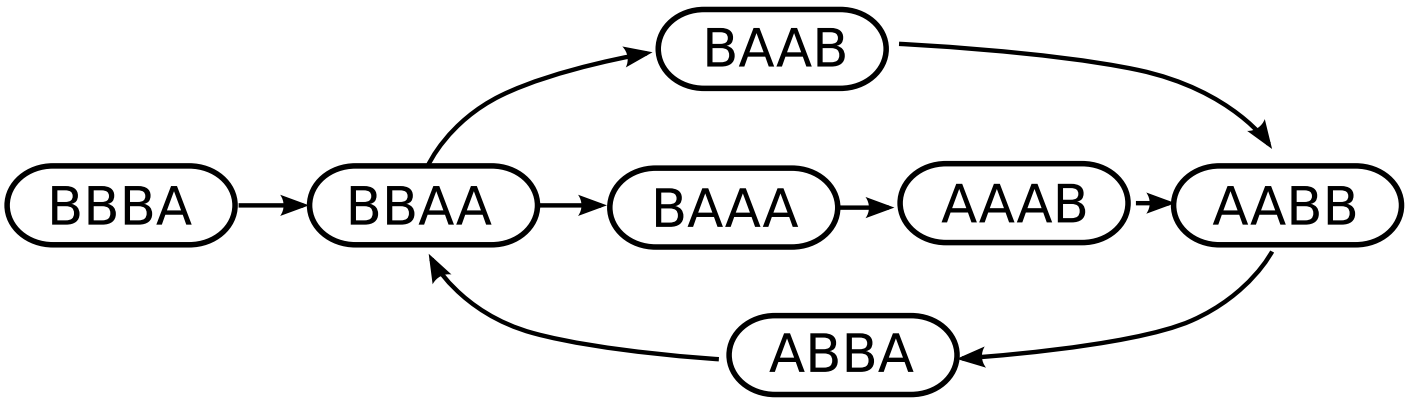


- przypadki kłopotliwe



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

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.

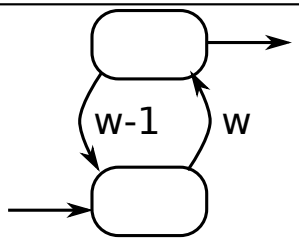
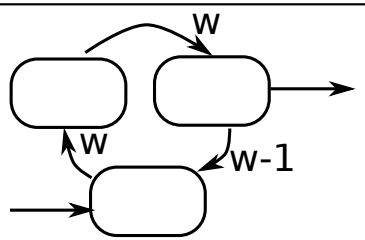
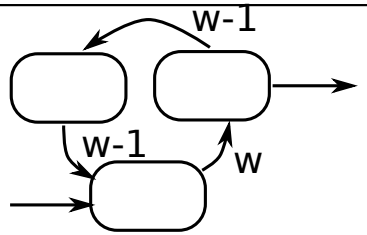
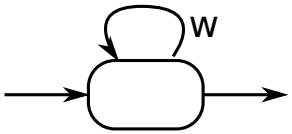
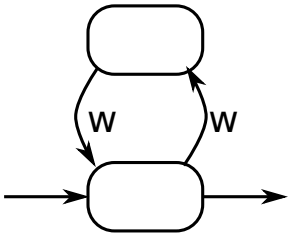
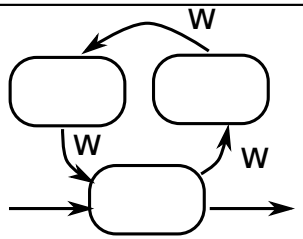
Sekwencja powtarzająca się S, $|S| = n$, $S = m_0 \ldots m_{d-1} m_0 \ldots m_{(n-1) \bmod d}$, motyw $m = m_0 m_1 \ldots m_{d-1}$, $|m| = d < n$.

Przykłady:

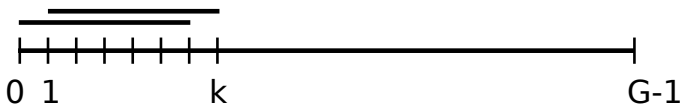
- `ACGTACGTACG`, $d = 4, n = 11$,
- `AAAAAAA` $(d = 1, n = 7)$.

Powtórzenia motywów w multi-grafach de Bruina

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

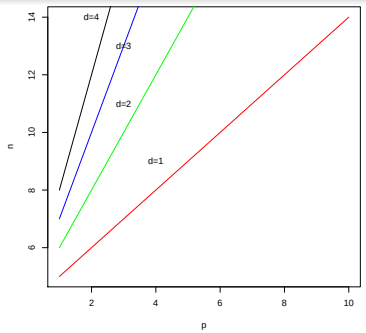
	$d = 1$	$d = 2$	$d = 3$
$n - k \equiv 0 \pmod{d}$			
$n - k \equiv 1 \pmod{d}$			
$n - k \equiv d - 1 \pmod{d}$			

K-spectrum jako wejście asemblera



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

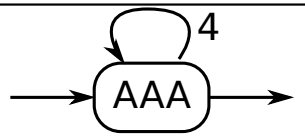
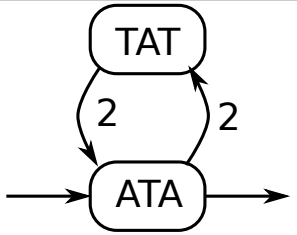
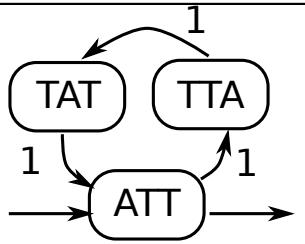
$$w = \frac{\Delta}{d}, \text{ gdzie } \Delta \equiv 0 \pmod{d}, \Delta = n - k + 1$$



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

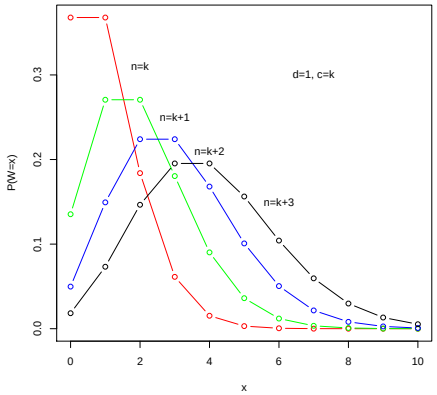
$$n = wd + k - 1, \text{ gdzie } n - k \equiv d - 1 \pmod{d}$$

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	$k = 4, d = 3, n = 6, w = 1$	

Rozkład wag krawędzi

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, $n > k$

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

dla

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

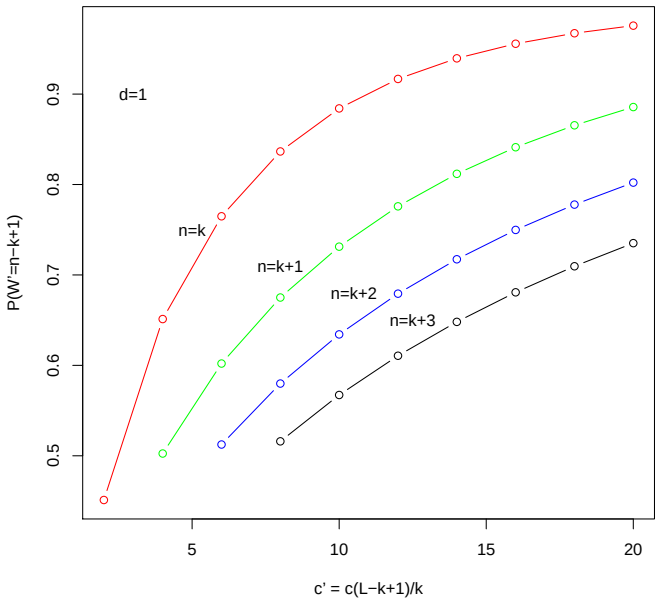
$$W \sim \text{Poisson}\left(\lambda\right), \lambda = \frac{c'\Delta}{d}, c' = \frac{c(L - k + 1)}{k}, \Delta = n - k + 1$$

Estymacja błędu normalizacji wag krawędzi

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

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

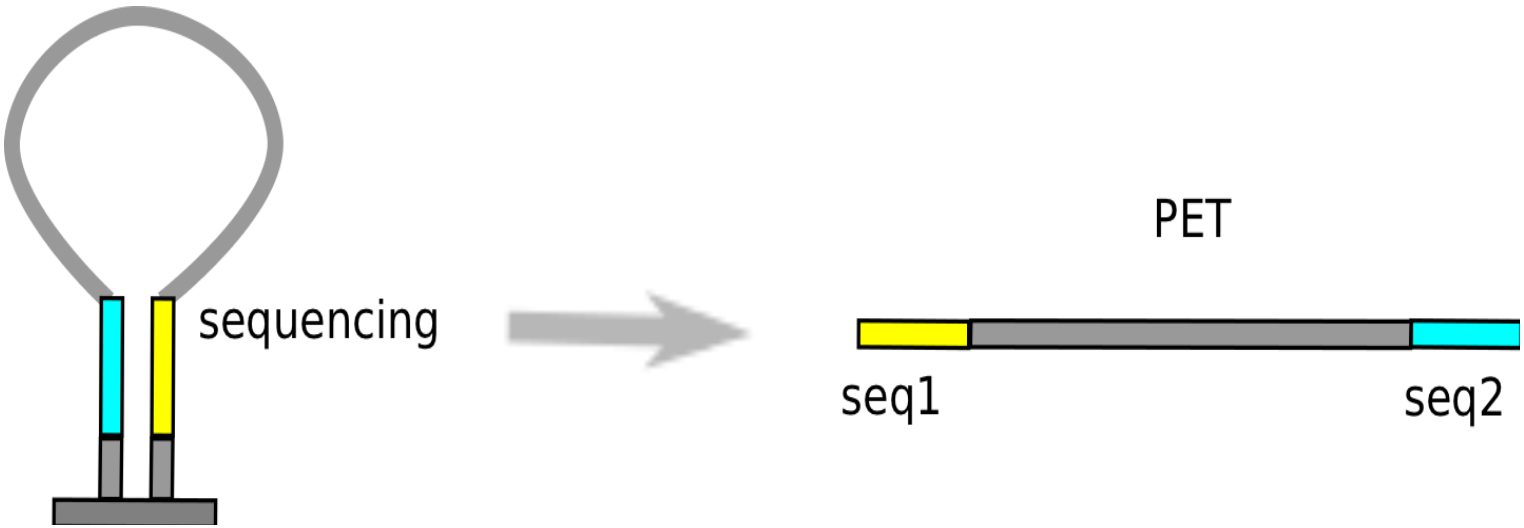
$\Phi_P(x, \lambda)$ jest dystrybuantą rozkładu Poissona



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, $n > k$

Algorytmy dla sparowanych końców

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



algorytmy osiągają wyniki jak dla odczytów o długości cząsteczki

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

Miary jakości asemblacji

Miary jakości asemblacji

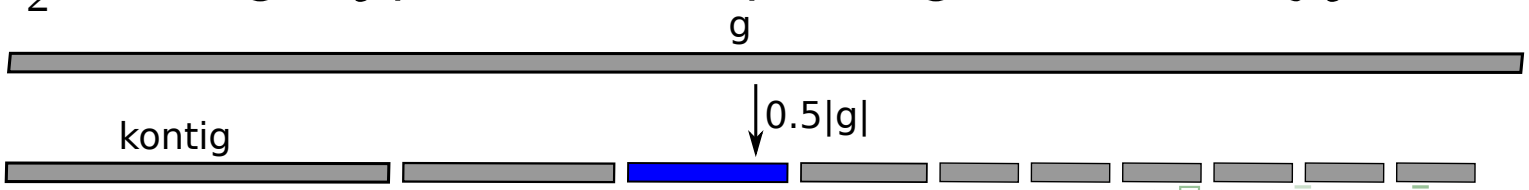
Wynik asemblacji genomu g to zbiór n kontigów $\{c_0, c_1, \ldots, c_{n-1}\}$

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 $t = 1000nt$)

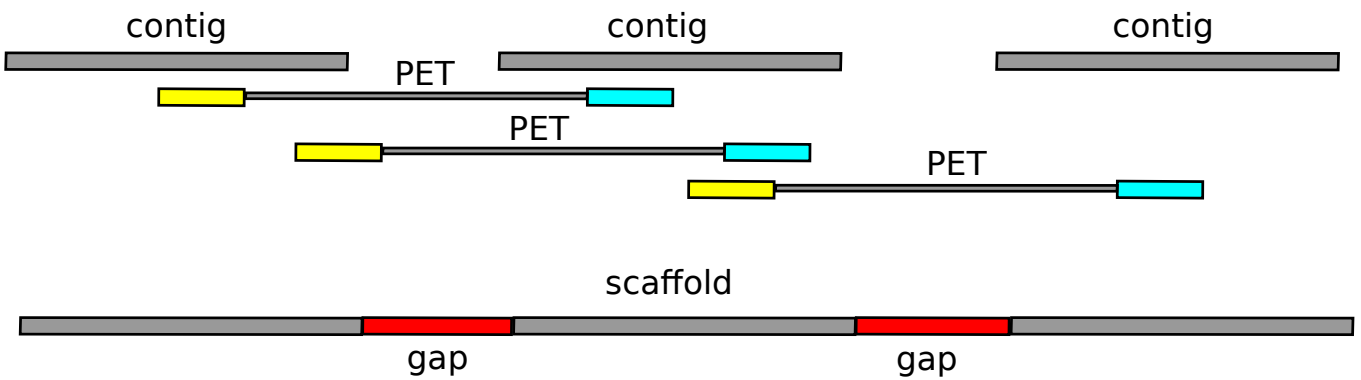
$$s = \sum_{i=0}^{n-1} |c_i|, \text{dla } |c_i| \geq t$$

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



Wykańczanie (scaffolding)

Wykańczanie (scaffolding)



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

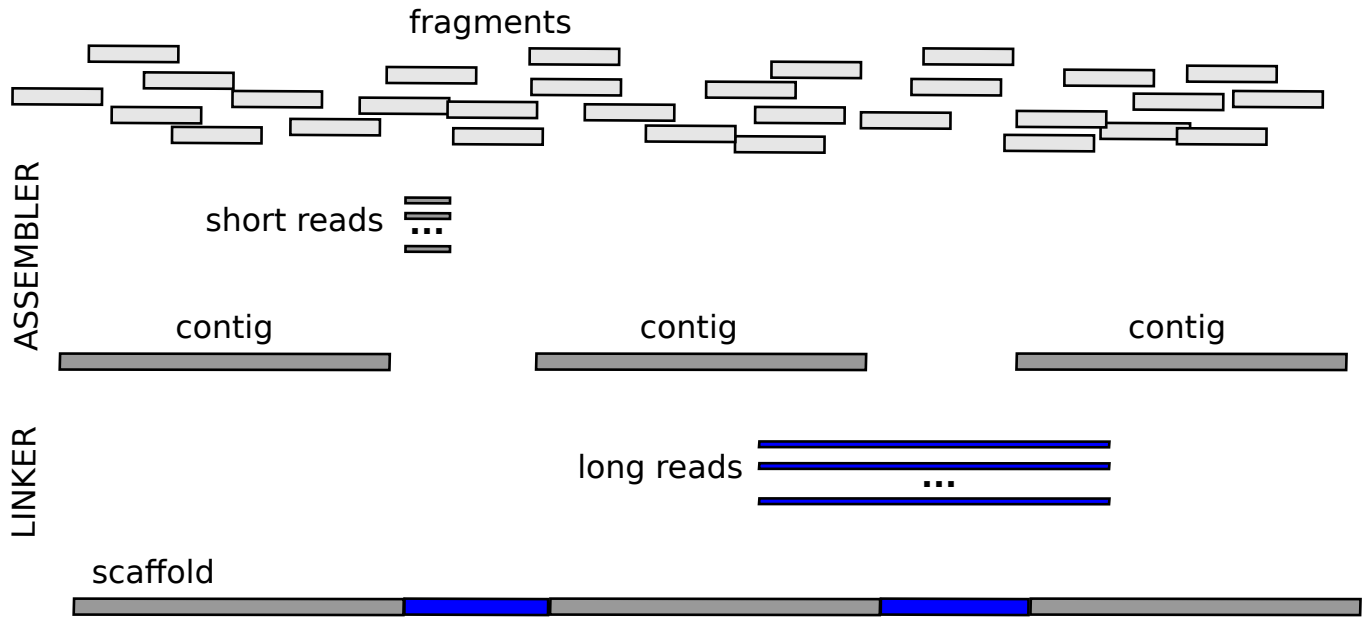
Algorytmy łączone: wykorzystują odczyty II generacji (krótkie, wysoka jakość) i III genreracji (długie, niska jakość)

- 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

Dziękuję