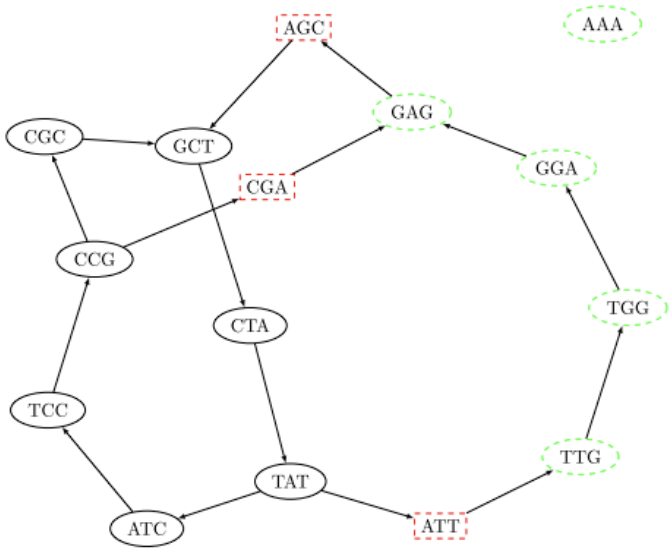
Asembler

Algorytmy Analizy Danych Genomicznych

Younginn Park

Minia, AbySS

- Filtr Blooma trzymający k-mery
- Filtr ma w sobie ukrytą strukturę grafu de Bruijna



(a)

$a_1 \ldots a_k$	$\sum_{i=1}^{k} a_i^i \bmod 10$
ATC	0
CCG	0
TCC	5
CGC	6
...	...

(b)

Bloom filter
1
0
0
0
0
1
1
0
0
0

(c)

Nodes self-information:

$$\lceil log_2 \binom{4^3}{7} \rceil = 30 \text{ bits}$$

Structure size:

$$\underbrace{10}_{\text{Bloom}} + \underbrace{3 \cdot 6}_{\text{False positives}} = 28 \text{ bits}$$

(d)

Chikhi et al. 2013

Pomysł

- Hashmap (słownik pythonowy)
  - Szybkie znajdowanie przynależności k-merów w zbiorze
  - Przetrzymywanie informacji o zliczeniach
- Zajmuje więcej pamięci niż filtr Blooma, ale za to zachowuje wartość zliczeń i nie ma ryzyka fałszywych pozytywów

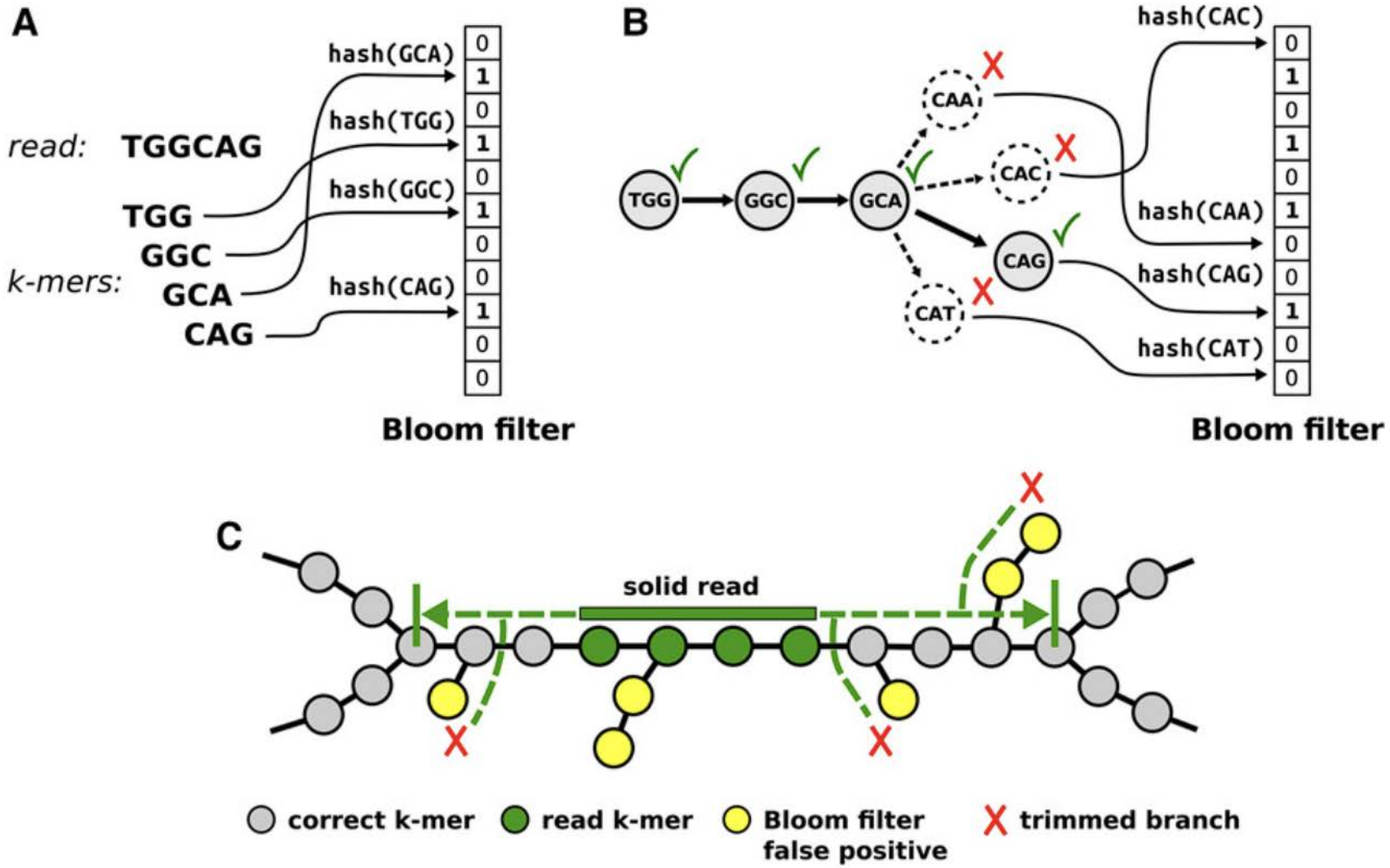
Pre-processing

- Zliczenie k-merów (*k=17*) z wszystkich odczytów i zapisanie wyników do słownika
- Odfiltrowanie rzadkich k-merów (mniej lub równie liczne od *d=2*)
- Utworzenie zbioru seedów (k-mery o 3 najwyższych wartościach liczności – “*solid k-mers*”)

Algorytm zachłanny

- Obustronne rozszerzanie seedów (w przód i w tył)
  - Generowanie następnego/poprzedniego k-meru po alfabecie (tu: ACTG)
  - Sprawdzenie czy taki k-mer występuje w zliczeniach
  - Rozszerzenie seedu o nukleotyd z najliczniejszego k-meru z występujących
    - Zmniejszenie wartości liczności tego k-meru w zliczeniach by uniknąć zapętlenia (tu: dzielenie przez 2)
  - Po rozszerzeniu, sprawdzenie, czy dokładnie taki sam contig już nie był wcześniej “odkryty” i czy contig ma długość >300bp

- Rozszerzenie seedu kończy się gdy żaden z opcji rozszerzeń nie znajduje się w zliczeniach



Jackman et al. 2017

Wyniki asemblacji

Odczyty	Liczba contigów	Liczba uliniowionych contigów	Pokrycie genomu referencyjnego	Pokrycie odczytów	Ocena identyczności
reads1	7	7	70,9%	99,0%	100%
reads2	3	3	15,5%	100%	100%
reads3	2	2	7,0%	100%	100%

Benchmark

Odczyty	Algorytm 1	Algorytm 2	Wynik Projektu
reads1	0.06	0.59	**0.47**
reads2	0.03	0.21	**0.12**
reads3	0.03	0.08	**0.07**

Bibliografia

- Jackman SD, Vandervalk BP, Mohamadi H, et al. ABySS 2.0: resource-efficient assembly of large genomes using a Bloom filter. *Genome Res*. 2017;27(5):768-777. doi:10.1101/gr.214346.116
- Chikhi R, Rizk G. Space-efficient and exact de Bruijn graph representation based on a Bloom filter. *Algorithms Mol Biol*. 2013;8(1):22. Published 2013 Sep 16. doi:10.1186/1748-7188-8-22