Locality-Sensitive Hashing für Read Mapping

David Losch

Locality-Sensitive Hashing

- **Idee:** Pre-Processing beim Read-Mapping auf Nearest-Neighbour Search / Locality-Sensitive Hashing reduzieren
- Kandidaten für exakteres String-Matching per Hashing herausfiltern
- Gratis: Variantentoleranz durch band-basiertes Hashing

Referenzgenom

- Referenzgenom in Abschnitte aufteilen, die ungefähr in der Größenordnung eines Reads liegen

Teil 1	**Teil 2**	**Teil 3**
ACGT...GTAA	GTCT...AGTA	CGCC...TAGA

- In jedem Abschnitt q-Gram-Index erzeugen

Teil 1		**Teil 2**		**Teil 3**	
AC	1	GT	1, ...	CG	1
CG	2	TC	2	GC	2
GT	3, ...	CT	3	CC	3
⋮	⋮	⋮	⋮	⋮	⋮

Referenzgenom

Teil 1	**Teil 2**	**Teil 3**
ACGT...GTAA	GTCT...AGTA	CGCC...TAGA

- Abschnitt im Referenzgenom $\widehat{=}$ *Dokument* in vorgestelltem LSH-Algorithmus
- Idee: Reads auch als *Dokument* zwischen die anderen „untermischen“

Pseudo-Dokumente

Referenzgenom	
Teil 1	**Teil 2**
ACGT...GTAA	GTCT...AGTA

Reads	
Read	**Read**
TGGC...ATAC	GCCA...CGAT

- Q-Gramme auch für Reads berechnen und auf gleiche Weise behandeln
- **Min-Hashing**-Verfahren anwenden (Reduktion der Information auf kürzere Signaturen)
- **Locality-Sensitive-Hashing** über Min-Hash-Signaturen ausführen
- **Problem:** Auf diese Weise können auch Reads auf andere Reads abgebildet werden

Probleme

- **Problem:** Auf diese Weise können auch Reads auf andere Reads abgebildet werden
- **Abhilfe:** Zusätzlich zur Signatur ein Bit als Indikator (Referenzgenomabschnitt/Read) speichern
- Auch im Hash-Bucket das Bit abspeichern
- Bei Kollision im Hash-Bucket wird nur ein Kandidat eingetragen, falls sich das Bit unterscheidet

Probleme

Referenzgenom	
Teil 1	**Teil 2**
ACGT...GTAA	GTCT...AGTA

Reads	
Read	**Read**
TGGC...ATAC	GCCA...CGAT

- Weiteres **Problem:** Durch die feste Unterteilung des Referenzgenoms werden abschnittsübergreifende Reads nicht erkannt
- **Abhilfe**: Zweite Iteration mit verschobenen Fenstern/Abschnitten durchführen

Vorteile

- Großer **Vorteil** des Verfahrens:
- Algorithmus stark parallelisierbar
- Hashing-Operationen sehr gut kompatibel zu Grafikkarten
- Verfahren dient als Sieb für nachfolgende teurere Ähnlichkeitsvergleiche (hier könnten natürlich herkömmliche Verfahren zum Einsatz kommen)

Probleme

- Zu guter Letzt: „Wie soll das denn bitte auf 'ne Grafikkarte passen?!“
- Eventuell den Algorithmus abschnittsweise ausführen
- Dabei müssen immer **alle** Reads in der Teilmenge vorhanden sein, aber es könnten weniger Referenzgenomabschnitte in Betracht gezogen werden