Practical universal $k$-mer sets for minimizer schemes

*Dan DeBlasio*, Fiyin Gbosibo, Carl Kingsford and Guillaume Marçais

slides: dandeblasio.com/bcb2019
Minimizer Schemes

Roberts, et al. (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Roberts, et al. (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Minimizer Schemes

Roberts, et al. (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Roberts, et al. (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Minimizer Schemes

Roberts, et al. (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Roberts, *et al.* (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Roberts, et al. (2004) introduced minimizer schemes as a way to decrease the time needed for sequence overlap computation.
Minimizer Schemes

- Minimizer schemes have two special properties:
  - two sequences with a long exact match must select the same $k$-mers
  - there are no large gap between selected $k$-mers

- Use in $k$-mer counting, *de Bruijn* graph construction, data structure sparsification, etc.
Minimizer Schemes

For a window of \( w \) consecutive \( k \)-mers from a sequence \( S \), a minimizer scheme selects the minimum according to an ordering \( o \) as a representative.
Minimizer Schemes

For a window of $w$ consecutive $k$-mers from a sequence $S$, a minimizer scheme selects the minimum according to an ordering $o$ as a representative.
Minimizer Schemes

For a window of $w$ consecutive $k$-mers from a sequence $S$, a minimizer scheme selects the minimum according to an ordering $\circ$ as a representative.
Minimizer Schemes

- Changing the ordering used can greatly impact the number of unique minimizers
- Can we find an order that minimizes the number of minimizer locations

Only some $k$-mers are used as minimizers
Universal $k$-mer Set

A universal $k$-mer set $U_{k,w} \subseteq \Sigma^k$ is a set of $k$-mers such that any window of $w$ consecutive $k$-mers must contain at least one element from the set.
Universal $k$-mer Set and Minimizer Ordering

- A universal $k$-mer set induces a family of compatible orderings
- Orderings based on universal sets have better performance than lexicographic or random orders (Marçais, et al., 2017)
A universal $k$-mer set induces a family of compatible orderings.

Orderings based on universal sets have better performance than lexicographic or random orders (Marçais, et al., 2017).
Universal $k$-mer Set and Minimizer Ordering

- Set Size
  - Fraction of all $k$-mers in the universal set

- Density
  - Normalized count of minimizer locations in $S$
Universal $k$-mer Set and Minimizer Ordering

- **Set Size**
  - Fraction of all $k$-mers in the universal set

- **Density**
  - Normalized count of minimizer locations in $S$

- **Sparsity**
  - Normalized count of windows in $S$ with only one umer (universal $k$-mer)
Universal $k$-mer Set and Minimizer Ordering

- **Set Size**
  - Fraction of all $k$-mers in the universal set

- **Expected** Density
  - Normalized count of minimizer locations in $B_L$

- **Expected** Sparsity
  - Normalized count of windows in $B_L$ with only one umer (universal $k$-mer)

$B_L$ is the de Bruijn sequence of order $L$, it contains each window exactly once.
Universal $k$-mer Set and Minimizer Ordering

- A universal $k$-mer set induces a family of compatible orderings.
- Orderings based on universal sets have better performance than lexicographic or random orders (Marçais, et al., 2017).
- Current methods cannot construct sets for values of $k$ and $w$ used in practice.

Can we construct universal $k$-mer sets that are practical for use in minimizer schemes?
Universal $k$-mer Set Extension

The naïve extension $U_{k,w} \cdot \Sigma$ of a universal set $U_{k,w}$ is universal

create $|\Sigma|$ new $(k+1)$-mers from each $k$-mer by concatenating each character from $\Sigma$ to the end

Example:

$\text{ACCTG} \in U_{k,w} \rightarrow \{\text{ACCTGA, ACCTGC, ACCTGT, ACCTGG}\} \in U_{k,w} \cdot \Sigma$
Universal $k$-mer Set Extension

The naïve extension $U_{k,w} \cdot \Sigma$ of a universal set $U_{k,w}$ is universal.

The sparsity of $U_{k,w} \cdot \Sigma$ is equal to that of $U_{k,w}$.

The density of a compatible order for $U_{k,w} \cdot \Sigma$ is less than or equal to the density of a compatible order for $U_{k,w}$ if the orderings are compatible with each other.
$M_u$ and $r e M_u$ val

- the minimum co-occurrence count for $u \in U$
  \[ M_u = \min_{\omega \in W_u} |\omega \cup U| \]

- For any $u \in U$ such that $M_u > 1$, $U \setminus u$ is universal
$M_u$ and $\text{rem} M_u$ val

- The minimum co-occurrence count for $u \in U$
  \[ M_u = \min_{\omega \in W_u} |\omega \cup U| \]

- For any $u \in U$ such that $M_u > 1$, $U \setminus u$ is universal

- The universal set after the removal of $u$ has:
  - smaller size, and
  - higher (possibly equal) sparsity
Optimal re\(M_u\)val

- Not all umers with \(M_u > 1\) can be removed from \(U\),

- Integer linear programming (ILP) is used to find the minimum number of \(k\)-mers to retain

- The ILP is deceptively simple

\[
\begin{align*}
\text{minimize} & \quad \sum_{u \in U} y_u \\
\text{subject to} & \quad \sum_{u \in \omega \cap U} y_u \geq 1 \quad \forall \omega \in W \\
& \quad y_u \in \{0,1\} \quad \forall u \in U
\end{align*}
\]

All of the umer co-occurrence information is encoded in \(W\)
Practical Universal $k$-mer Set Construction

DOCKS (Orenstein, et al., 2017) $U_{k',w}$ $j=k'+1$

Naïve Extension $\hat{U}_{j,w}$$j=j+1$

reMoval $U_{j,w}$

$\text{yes} \rightarrow U_{k,w}$

$\text{no}$

$j==k?$
Improvements Over DOCKS Sets

- **Set Size Fraction**
  - (a) Graph showing a decrease in set size fraction as $k$-mer length increases.
  - DOCKS vs Random
  - Better performance for DOCKS in most cases.

- **Density**
  - (b) Graph showing density values for different $k'$ values.
  - DOCKS vs Random
  - Better performance for DOCKS in most cases.

- **Sparsity**
  - (c) Graph showing sparsity values for different $k'$ values.
  - DOCKS vs Random
  - Better performance for DOCKS in most cases.

Differences by $k'$:
- $k' = 2$
- $k' = 3$
- $k' = 4$
- $k' = 5$
- $k' = 6$
- $k' = 7$
- $k' = 8$
- $k' = 9$
- $k' = 10$

Note: Better performance is indicated by the green arrow.
Sequence-specific re\textsubscript{$M_u$}val

- Often minimizer schemes are used on a single reference sequence
- Not all windows will appear in that sequence
- The set of windows for re\textsubscript{$M_u$}val \textit{W} can be limited to those from the reference
Improvement Over General $k$-mer Sets

(a) Set Size Fraction

(b) Density

(c) Sparsity
Summary

- Universal $k$-mer sets can be constructed for use as in for minimizer scheme orderings for large $k$

- Reference-specific universal sets have better performance

- Open question if similar methods can be used to extend in $w$
Practical universal $k$-mer sets for minimizer schemes

Dan DeBlasio, Fiyin Gbosibo, Carl Kingsford and Guillaume Marçais

dandeblasio.com
danfdeblasio

Thanks
Kingsford group members
iBRIC administration, especially David Boone

Funding
Gordon and Betty Moore Foundation’s Data-Driven Discovery Initiative (GBMF4554)
US National Institutes of Health (R01GM122935)
US National Science Foundation (CCF- 1256087, CCF-1319998)

slides: dandeblasio.com/bcb2019
code: github.com/Kingsford-Group/remuval