Sequence Assembly Plan

Research Methods in Biomedical Informatics

Christian Anderson

17 February 2026

CONTENTS

Overview

Problem

“Create a program that takes as input the set of all next-generation sequencing reads identified in a sample and an initial query sequence and returns the largest sequence contig that can be constructed from the reads that contains the initial query sequence.”

What are kmers?

"In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort."

–*The Hobbit* [1]

What are kmers?

“In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort.”

–*The Hobbit* [1]

inaholeinthegroundtherelivedahobbitnotanastydirty...

What are kmers?

"In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort."

–*The Hobbit* [1]

inaholeinthegroundtherelivedahobbitnotanastydirty...

Simulate Reads by taking random substrings of different length

inahole
intheground
edahobbitno
itnotanastydirty ...

What are kmers?

"In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort."
–*The Hobbit* [1]

inaholeinthegroundtherelivedahobbitnotanastydirty...

Simulate Reads by taking random substrings of different length

inahole
intheground
edahobbitno
itnotanastydirty ...

Chop into k length 'mers

$k = 5$
inaho, nahol, nahole, aholei, holein, ... itnot, tnota, notan, otana ...

What are kmers?

"In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort."

–*The Hobbit* [1]

inaholeinthegroundtherelivedahobbitnotanastydirty...

Simulate Reads by taking random substrings of different length

inahole
intheground
edahobbitno
itnotanastydirty ...

Chop into k length 'mers

How do we work backwards?

$k = 5$
inaho, nahol, nahole, aholei, holein, ... itnot, tnota, notan, otana ...

What is a graph?

A graph is a structure used to encode connections between things.

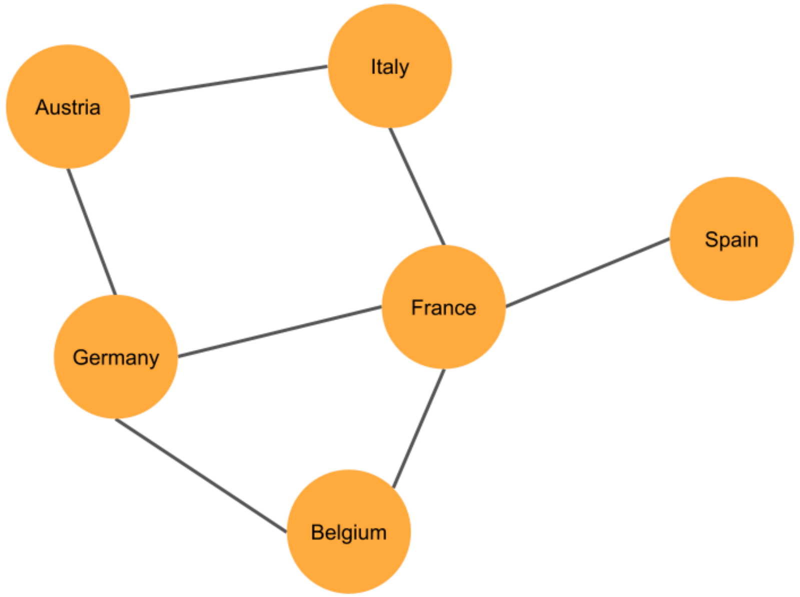


Figure 1: A simple graph used to represent how countries share borders [2] .

I can use a graph to connect kmers that are similar to each other and reconstruct the original string by following each connection.

What is a graph?

Using the previous example, I can plot the graph connecting each kmer to every other kmer that has an overlap.

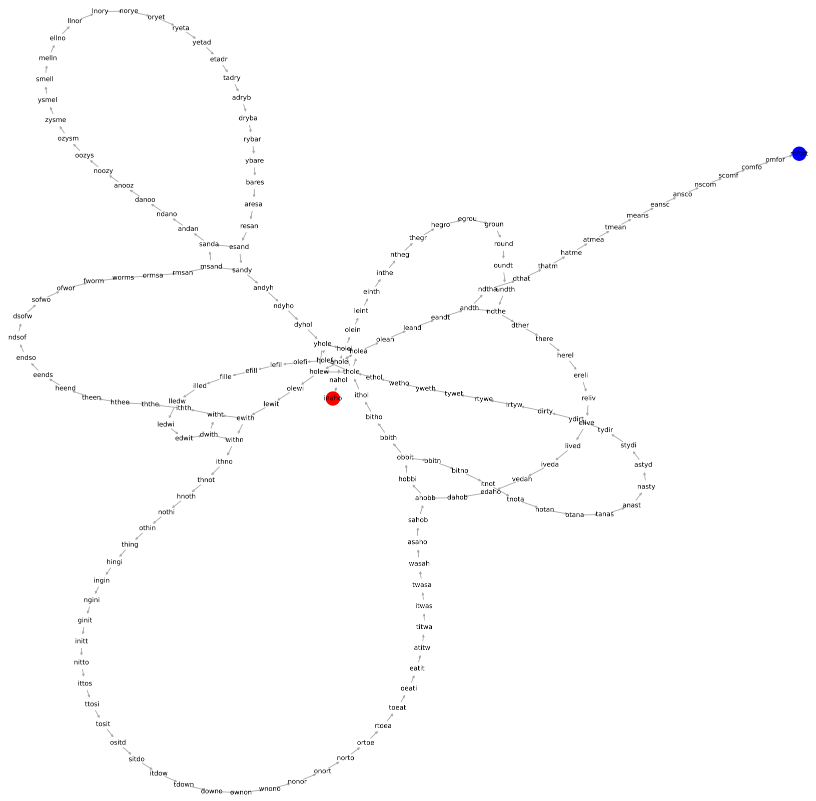


Figure 2: A graph of the 5-length kmers from the first sentence of The Hobbit.

What is a graph?

This kind of graph is composed of kmers with k-1 overlapping characters.
AKA a De Brujin graph:

$k = 5$

```
inaho
 nahol
  ahole
   holei
    olein
     leint
      einth
       inthe ...
inaholeintheground...
```

Following each edge along the graph, the original sentence can be constructed using the last letter of each word.

Technical Approach

Technical Approach

- Process reads into kmers
- Compare each kmer to every other and measure overlap
- Create contiguous strings following branches of kmer overlaps
- Find the longest contigs containing the query sequence

Process reads

- Make k-length strings (kmers) from reads.
- Record kmer frequencies.
- Filter out infrequent kmers below some threshold τ.
- Filter for unique kmers.
- Return list of unique kmers S.

```
let R be a list of strings composed of A, T, C, or G
let k be an integer
let tau be an integer
let results be an empty list
let counter be a dictionary

# trim each string to length k; keep track of string frequencies
for r in R:
  l = len(r)
  for i in range(0, l-k+1):
    r_sub = r[i:i+k]
    results += r_sub
    counter[r_sub] += 1

# save only unique and frequent strings
filtered = counter.keys()[counter.values() > tau]

return filtered
```

Compare Kmers

- Test how each kmer aligns with every other.
- Return an adjacency matrix where a value of *1* means the kmer at the row index matches the string of the column index.

```
let S be a list of k-length strings
let m be a square matrix of zeros with dimensions len(S)

for i in range(len(S)):
  for j in range(len(S)):
    if i does not equal j:
      si = S[i]
      sj = S[j]
      let si_sub be the substring of si from the second character to the last (2 to k)
      let sj_sub be the substring of sj from the first character to the second last (1 to k-1)
      if si_sub equals sj_sub:
        m[i,j] = 1
        else:
          m[i,j] = 0

return m
```

Create Contiguous Sequences

- Concatenate every possible combination of adjacent strings.
- Return a list of contiguous sequences *contigs*

for every string find the next closest strings and for each of these append it to the end of the previous

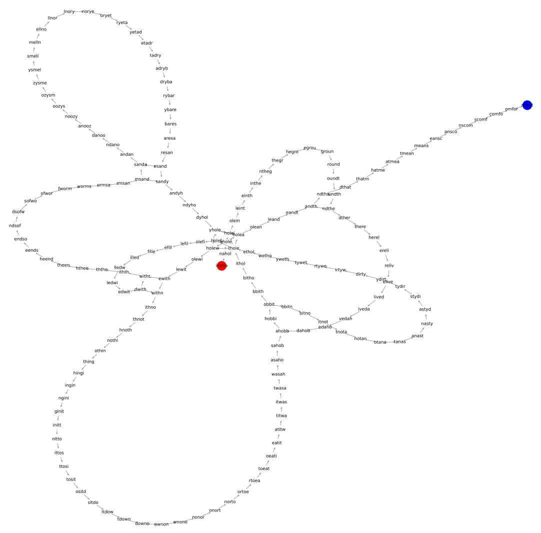


Figure 3: A graph of the 5-length kmers from the first sentence of The Hobbit.

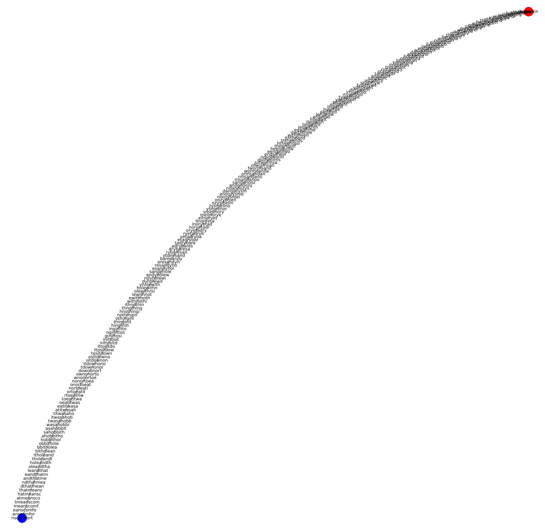


Figure 4: A graph of the 9-length kmers from the first sentence of The Hobbit.

5-letter versus 9-letter kmers

Create Contiguous Sequences

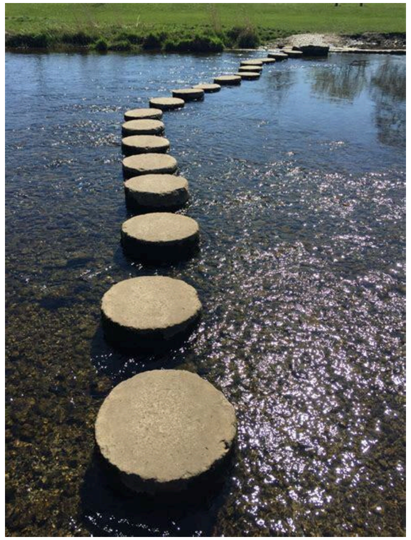


Figure 5: Stones in a river arranged in a line.

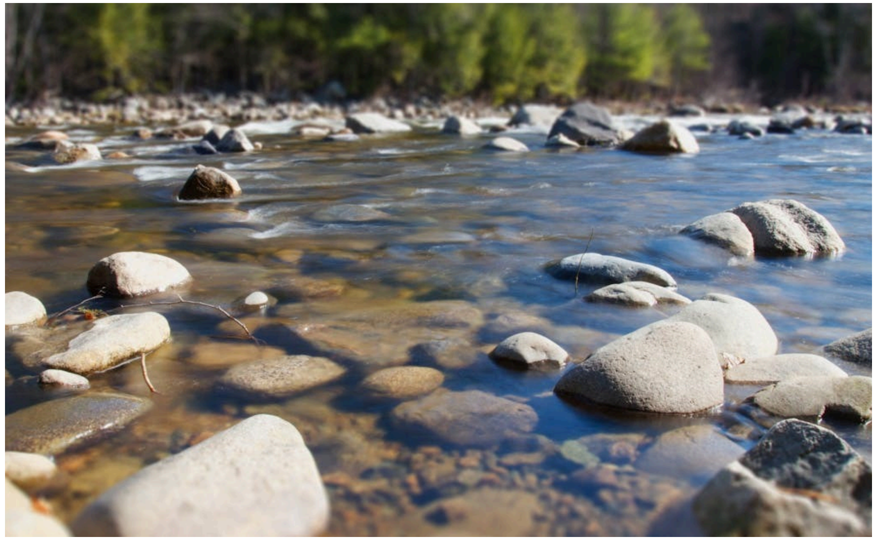


Figure 6: Stones scattered in a river.

Create Contiguous Sequences, cont.

```
let S be as list of k-length strings
let A be an adjacency matrix between strings in S, with row and column names S, where rows encode outgoing and columns incoming edges

let colsums be the column sums of A
let starts be a list of strings of S where colsums equals the minimum value in colsums (S[colsums == min(colsums)])

contigs = []

define _recurse as a function of curr, contig, visited:
  row = A[ S==curr, : ] # ``
  nexts = S[row>0] # ``

  if len(nexts):
    contigs+=contig
    stop function execution

  for next in nexts:
    if next in visited:
      contigs+=contig
      continue to next loop
    new_visited = visited.copy() # ensure each branch only sees within branch visits
    new_visited[next] +=1

    _recurse(next, contig+=next[-1], new_visited) # recursivley run function, each time appending contig with the last character of next

  for start in starts:
    let counter be a dictionary
    counter[start] = 1
    _recurse(start, start, counter)

return contigs
```

Query Contigs

- Find the longest sequence contig that contains a query sequence.

```
let C be the list of contiguous strings
let q be the query string

max_l = 0

for c in C:
  l = len(c)
  if q is in c & l > max_l:
    max_l = l
    longest_with_query = c

return longest_with_query
```

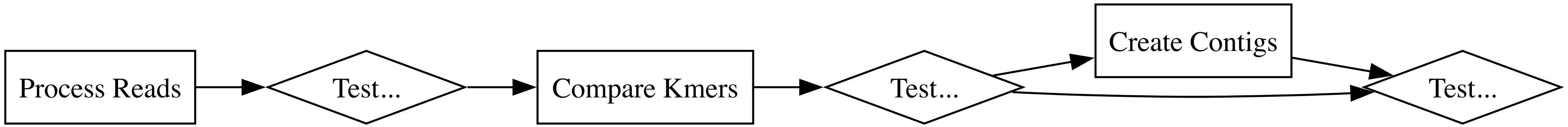
Example Results

Query: dirtywetholefilledwiththeendsofwormsandanoozysmell

Longest contigs with query sequence:

inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresand
inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyhole
inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyhole
inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholewith
inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholewithnothinginittositdownonortoeatitwasahob
inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeandth
inaholeinthegroundtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeandthatmeanscomfort
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresand
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeinthegroundtherelivedahob
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeinthegroundthatmeanscomfort
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyhole
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyhole
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeandtherelivedahob
inaholewithnothinginittositdownonortoeatitwasahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeandthatmeanscomfort
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresand
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeinthegroundth
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholeinthegroundthatmeanscomfort
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyhole
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholewith
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyholewithnothinginittositdownonortoeatitwasahob
inaholeandtherelivedahobbitnotanastydirtywetholefilledwiththeendsofwormsandanoozysmellnoryetadrybaresandyhole

Development and Testing



Assignment

- a clear and concise problem statement and algorithm design description that is targeted at a general scientific audience. Figures such as flow charts and schematics are encouraged.
- Technical jargon is discouraged.
- a detailed development and testing plan that is targeted at a fellow methods developer. This plan should explain how you will decompose your solution into independent modules, how those modules will interact, how they will be tested, and a timeline.

References

[1] J. R. R. Tolkien, *The Hobbit*. HarperCollins, 2012.

[2] Databricks Documentation, “Simple Graph Illustration.” 2025.