Разработка алгоритма секвенирования с использованием paired-end данных

Владислав Исенбаев

Санкт-Петербургский государственный университет информационных технологий, механики и оптики

Научный руководитель: д.т.н., профессор Шалыто А. А.
Рецензент: к.ф.-м.н., Женило С. В.

Санкт-Петербург
2010 г.

ДНК

Дезоксирибонуклеиновая кислота (ДНК).

- Двойная спираль из двух полимерных цепочек.

ДНК

Дезоксирибонуклеиновая кислота (ДНК).

- Двойная спираль из двух полимерных цепочек.
- Составные элементы — нуклеотиды (A, T, G и C).

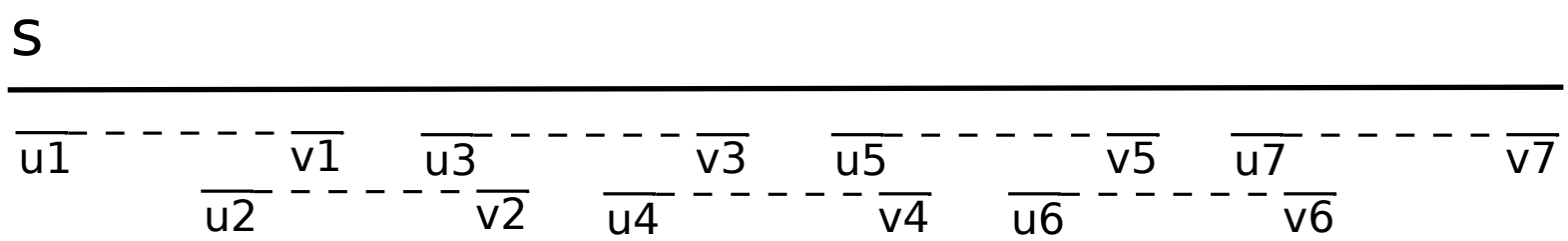
ДНК

Дезоксирибонуклеиновая кислота (ДНК).

- Двойная спираль из двух полимерных цепочек.
- Составные элементы — нуклеотиды (A, T, G и C).
- Комплементарность (A-T, G-C).

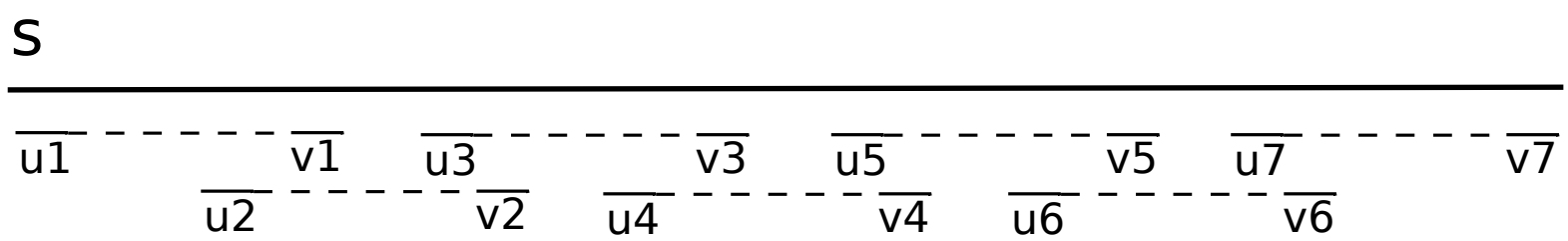
Физический процесс секвенирования double-ended shotgun

- Амплификация исходной ДНК.



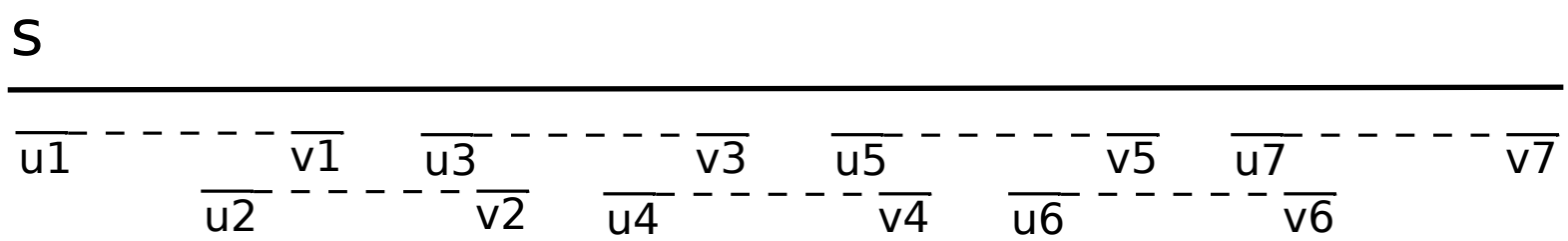
Физический процесс секвенирования double-ended shotgun

- Амплификация исходной ДНК.
- Разрезание ДНК на фрагменты.



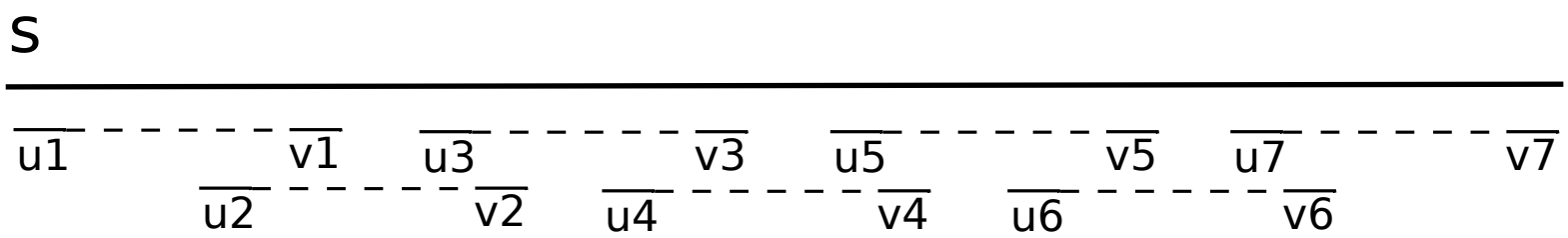
Физический процесс секвенирования double-ended shotgun

- Амплификация исходной ДНК.
- Разрезание ДНК на фрагменты.
- Выделение фрагментов подходящей длины (200 оснований).



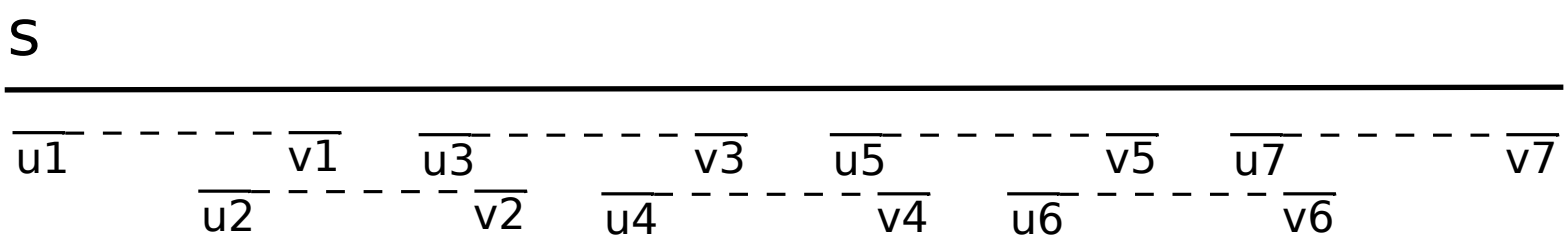
Физический процесс секвенирования double-ended shotgun

- Амплификация исходной ДНК.
- Разрезание ДНК на фрагменты.
- Выделение фрагментов подходящей длины (200 оснований).
- Чтение префикса и суффикса каждого фрагмента (по 36 оснований каждый).



Физический процесс секвенирования double-ended shotgun

- Амплификация исходной ДНК.
- Разрезание ДНК на фрагменты.
- Выделение фрагментов подходящей длины (200 оснований).
- Чтение префикса и суффикса каждого фрагмента (по 36 оснований каждый).
- Обработка полученных данных.



Обработка полученных данных: особенности

- Большой объем данных.

Обработка полученных данных: особенности

- Большой объем данных.
- Ошибки чтения.

Обработка полученных данных: особенности

- Большой объем данных.
- Ошибки чтения.
- Длинные повторы.

Проблема длинных повторов

$X = P \cdot R \cdot A \cdot R \cdot B \cdot R \cdot S$
$Y = P \cdot R \cdot B \cdot R \cdot A \cdot R \cdot S$
Если $|R|$ большой, X и Y неразличимы.
Следствие — отсутствие однозначного решения на реальных данных.

Два подхода к решению задачи секвенирования при использовании paired-end данных

- Разработка алгоритма «с нуля» (автор одной из работ — М. Дворкин)
- Сведение задачи к обычной задаче секвенирования (без использования paired-end информации).

В настоящей работе используется второй подход.

Пример использования первого подхода

- Фильтрация ошибок.

Пример использования первого подхода

- Фильтрация ошибок.
- Решение задачи без использования paired-end данных.

Пример использования первого подхода

- Фильтрация ошибок.
- Решение задачи без использования paired-end данных.
- Синтез полученных контигов в более длинные последовательности опираясь на paired-end данные.

Предлагаемое решение — сведение задачи с paired-end даными к обычной задаче секвенирования

- Фильтрация ошибок.

Предлагаемое решение — сведение задачи с paired-end даными к обычной задаче секвенирования

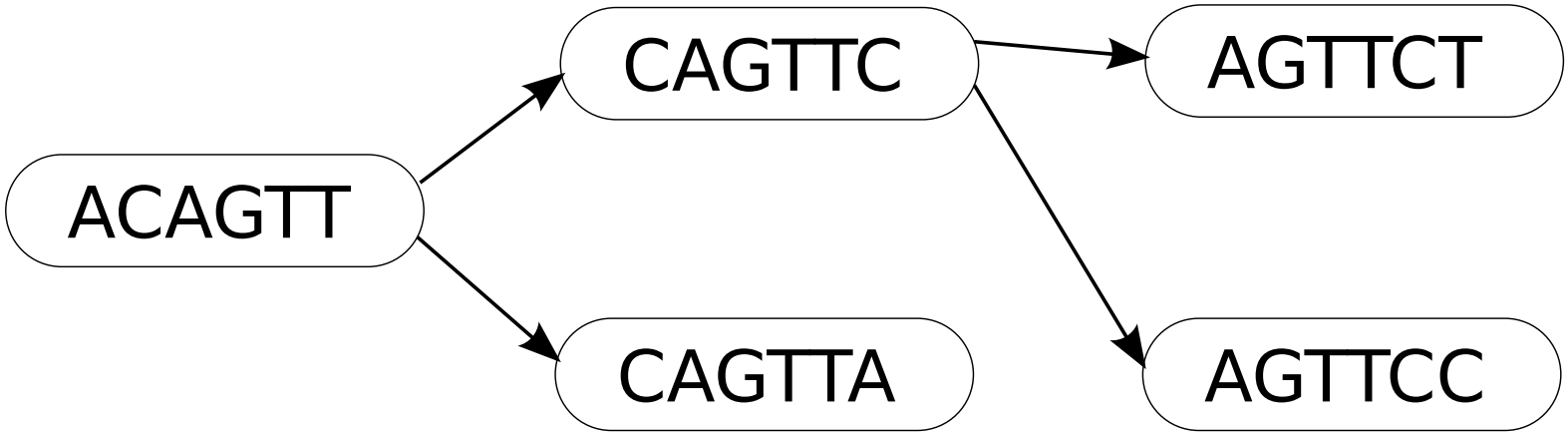
- Фильтрация ошибок.
- Сведение задачи с paired-end данными к обычной задаче секвенирования.

Предлагаемое решение — сведение задачи с paired-end даными к обычной задаче секвенирования

- Фильтрация ошибок.
- Сведение задачи с paired-end данными к обычной задаче секвенирования.
- Применение известных алгоритмов для задачи без paired-end данных.

Сведение задачи к известной (1)

Построим по исходным данным граф де Брейна.



$$(X_i, Y_i) \Leftrightarrow F_i = X_i \cdot Z_i \cdot Y_i$$

$$|F_i| \approx m, |X_i| = |Y_i| = k$$

Тогда Z_i соответствует пути в графе де Брейна из вершины X_i в вершину Y_i длиной $\approx m - k$.
Найдем все такие пути перебором.

Сведение задачи к известной (2)

$\textsc{Walk}(G, u, v, foundPaths)$

```
1  return RECURSIVE-WALK(G, G.vertices[u], G.vertices[v], 0, u, foundPaths
```

$\textsc{Recursive-Walk}(G, u, v, depth, pathString, foundPaths)$

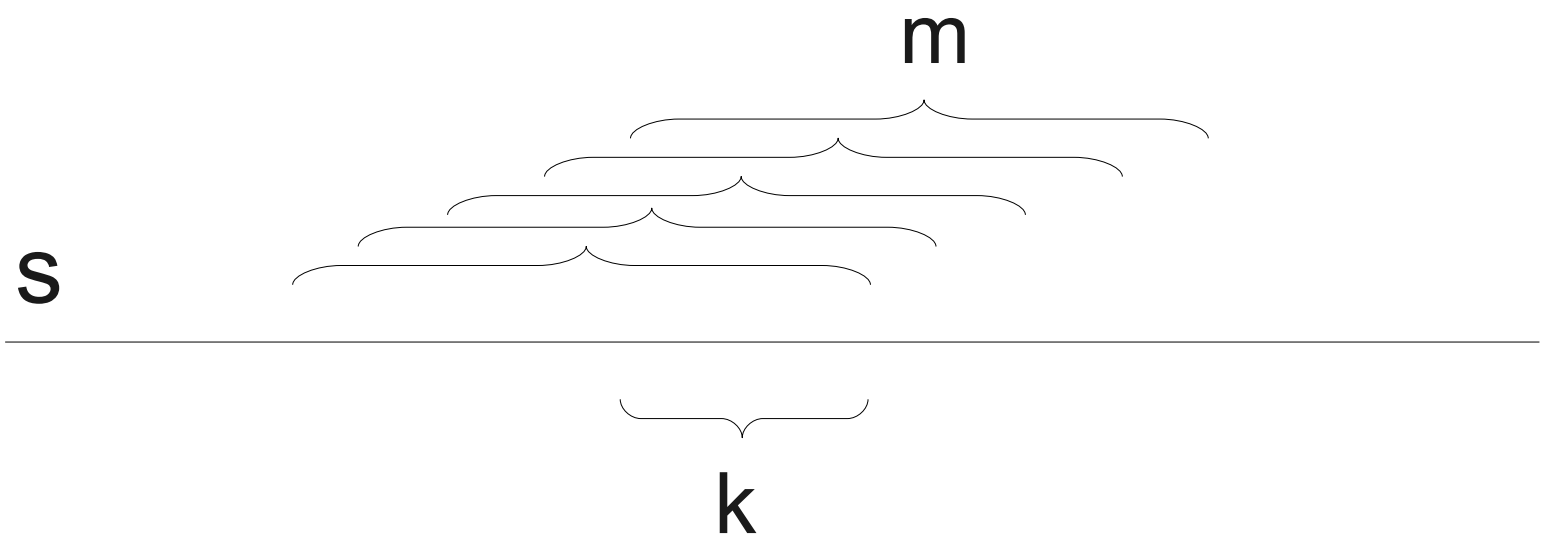
```
 1  if depth > maxDepth
 2        // отсечение перебора при превышении максимальной длины
 3        return
 4  if u == v and depth ⩾ minDepth
 5        // найден путь
 6        foundPaths.add(pathString)
 7  for (u, u') ∈ G.edges
 8        // перебираем очередное ребро в пути
 9        newPathString = pathString + (u, u').symbol
10        // аттрибут symbol ребра (u, v) содержит символ c
11        // такой, что (u + c)[2..u.length] == v
12        RECURSIVE-WALK(G, u', v, depth + 1, newPathString, foundPaths)
```

Производительность

Если средняя исходящая степень вершины в графе де Брюина равна d, алгоритм будет обрабатывать один фрагмент в среднем за $O(d^{m-k})$. Если считать, что все подстроки длины k равновероятны, можно получить оценку $d \approx 1 + \frac{N}{2^{k-3}}$, где N — длина исходной последовательности.
При $N = 10^8, m = 200, k = 36$ имеем $d^{m-k} \approx 7$, то есть алгоритм ведет себя достаточно эффективно.
Для больших значений N может потребоваться использование более эффективного метода поиска путей нужной длины, например meet-in-the-middle, дающий оценку $O(d^{\frac{m-k}{2}})$

Фильтрация ошибок

С помощью этого же алгоритма можно искать данные с ошибками чтения. Действительно, если строка прочитана без ошибок, через соответствующую вершину в графе должно проходить примерно $m - k$ путей-фрагментов (кроме строк, близких к началу или концу, их необходимо обработать отдельно).



Результаты

На симулированных данных без ошибок для генома Escherichia Coli алгоритм восстанавливает 98% фрагментов, что обеспечивает 99.5% покрытия исходной последовательности.
На симулированных сильно зашумленных данных (60% чтений с хотя бы одной ошибкой) для генома Haemophilus Influenzae алгоритм возвращает 0.01% ошибочных фрагментов и обеспечивает 99.5% покрытия исходной последовательности.

Выводы

- разработано два алгоритма сведения задачи секвенирования с использованием paired-end данных к обычной задаче секвенирования, первый из которых является простым в реализачии, а второй — более эффективен;

Выводы

- разработано два алгоритма сведения задачи секвенирования с использованием paired-end данных к обычной задаче секвенирования, первый из которых является простым в реализачии, а второй — более эффективен;
- разработан алгоритм устранения ошибок чтения, не использующий эвристики;

Выводы

- разработано два алгоритма сведения задачи секвенирования с использованием paired-end данных к обычной задаче секвенирования, первый из которых является простым в реализачии, а второй — более эффективен;
- разработан алгоритм устранения ошибок чтения, не использующий эвристики;
- проведен теоретический анализ эффективности созданных алгоритмов;

Выводы

- разработано два алгоритма сведения задачи секвенирования с использованием paired-end данных к обычной задаче секвенирования, первый из которых является простым в реализачии, а второй — более эффективен;
- разработан алгоритм устранения ошибок чтения, не использующий эвристики;
- проведен теоретический анализ эффективности созданных алгоритмов;
- разработанные алгоритмы реализованы ввиде программы для ЭВМ;

Выводы

- разработано два алгоритма сведения задачи секвенирования с использованием paired-end данных к обычной задаче секвенирования, первый из которых является простым в реализачии, а второй — более эффективен;
- разработан алгоритм устранения ошибок чтения, не использующий эвристики;
- проведен теоретический анализ эффективности созданных алгоритмов;
- разработанные алгоритмы реализованы ввиде программы для ЭВМ;
- проведены численные эксперименты на синтетических данных, построенных из реальных ДНК живых организмов и проведен анализ полученных результатов.

Перспективы

Реализация, созданная в рамках данной работы, является прототипом и недостаточно эффективна для использования на больших наборах данных (например для секвенирования ДНК, входящих в геном человека). Требуется следующие улучшения.

- Реализовать программу на языке *C++* с использованием эффективных по памяти хеш-таблиц для реализации графа де Брейна;

Перспективы

Реализация, созданная в рамках данной работы, является прототипом и недостаточно эффективна для использования на больших наборах данных (например для секвенирования ДНК, входящих в геном человека). Требуется следующие улучшения.

- Реализовать программу на языке *C++* с использованием эффективных по памяти хеш-таблиц для реализации графа де Брейна;
- Сделать систему распределенной (время выполнения алгоритма уменьшается пропорционально количеству процессоров из-за того, что обработка каждой пары независима).

Конец

Спасибо за внимание!