Параллельный метод объединения результатов работы программ по сборке генома*

К.В. Романенков[1], А.Н. Сальников[1], А.В. Алексеевский[2,3]

факультет Вычислительной математики и кибернетики МГУ имени М.В. Ломоносова[1], МГУ имени М.В. Ломоносова, Научно-исследовательский институт физико-химической биологии имени А.Н. Белозерского[2], Научно-исследовательский институт системных исследований Российской академии наук[3]

В данной работе проводится исследование в области применения многопроцессорных систем для задачи коррекции сборки генома. Существует большое количество алгоритмических подходов к проблеме сборки генома из набора коротких фрагментов, при этом результаты их работы на одних и тех же экспериментальных данных зачастую существенно разнятся. Вследствие большого объема данных необходима организация вычислений в модели распределенной памяти на вычислительном кластере. Предлагаемый подход использует комбинацию выводов программ сборки геномов, что позволяет сделать результат более корректным в биологическом смысле.

1. Введение

Современные высокотехнологичные автоматические методы расшифровки последовательностей ДНК (секвенирования) позволяют в течении короткого времени (несколько дней) и сравнительно недорого (несколько тысяч долларов) получить сотни миллиардов коротких последовательностей из четырех букв A, T, G, C, полученных прочтением фрагментов входного образца ДНК одного или нескольких организмов.

Расшифровка индивидуальных геномов людей и множества видов организмов, от вирусов и бактерий до сельскохозяйственных и домашних животных, существенно ускоряет прогресс в персонифицированной медицине, биоинженерии и биотехнологии, теории эволюции и других областях биологии [1].

В идеале результатом секвенирования должны быть последовательности хромосом изучаемого организма; в случае человека – 24. Однако на пути от десятков миллиардов последовательностей длины 100-250 до нескольких последовательностей, представляющих полный геном, возникают серьёзные технологические, алгоритмические и вычислительные проблемы [1]. Полностью преодолеть возникающие трудности для больших геномов не удается или удается с большим трудом. Так, последняя сборка *hg38* генома человека состоит из 735 «скэффолдов» – непрерывных последовательностей, – вместо 24-х в идеале. Число нерасшифрованных нуклеотидов оценивается в 160 млн. (расшифрованных – 3 млрд 209 млн 286 тыс 105) [2]. Показательно также, что среди геномов 1542 видов эукариот, заявленных в БД NCBI в разделе секвенированных геномов, лишь 14 отнесены к группе полностью секвенированных геномов [3].

В алгоритмах ассемблирования (сборки геномов из коротких ридов) должны учитываться такие факторы, как неравномерность покрытия геномов ридами, которые по технологии получаются из случайных фрагментов ДНК; возможность и частоту ошибок в ридах; возможность наличия химерных ридов, составленных из разных частей ДНК; наличие в геномах длинных повторов, которые могут приводить к невозможности восстановление полной последовательности даже теоретически.

Наиболее сложной является сборка генома *de novo*. Задача сборки при наличии образ-

*Работа выполнена при частичной поддержке РФФИ (гранты 14-07-00628, 14-07-00654 и 14-04-01693)

ца, например, сборки генома индивидуального человека при наличии референсного генома (генома образца) более простая. Предложены десятки алгоритмов сборки de novo. Большинство из них основаны на построении графа де Брюйна и нахождении Эйлерового пути в нем. Однако из-за разных эвристик, заложенных на разных этапах сборки результаты применения сборщиков существенно отличаются [4], см. также наши результаты далее в тексте. Важным обстоятельством, усложняющим сравнение алгоритмов, является то, что отсутствуют универсальные метрики оценки качества сборки. Причина заключается в том, что нет количественной оценки ошибок разного типа. Например, что лучше: большее число длинных контигов (однозначно расшифрованных непрерывных последовательностей) и скэффолдов (нескольких контигов, склеенных в единую последовательность с возможными пропусками или недостоверно определенными нуклеотидами между ними) либо уменьшение числа химер: контигов и скэффолдов, ошибочно склеенных из фрагментов разных хромосом?

Большой разрыв между желаемым результатом сборки *de novo* и получаемыми результатами, высокая актуальность создания более качественных сборщиков геномов для медицины и биологии ставят эту задачу в первый ряд актуальных вычислительных задач.

2. Задача коррекции сборки

Сборки генома, построенные сборщиками на основе ридов длины 100-150 нуклеотидов фрагментированы и содержат ошибки. Обычная практика при решении задачи сборки генома заключается в запуске нескольких сборщиков с различными параметрами, а затем выбору наилучшего варианта согласно некоторым соображениям. Однако недавние исследования показывают, что достаточно распространена ситуация, при которой одни сборщики показывают лучший результат по одному из статистических критериев в сравнении с другими программами, и уступают им же по другому критерию [5]. С большой уверенностью можно утверждать, что для каждого сборщика существуют фрагменты в геноме, с задачей сборки которых он справляется лучше остальных. В основном, это объясняется деталями реализации стратегии того или иного сборщика по разрешению «проблемных» участков в геноме, например, связанных с повторами.

Согласуясь с этим предположением, в последнее время были созданы программные средства, реализующие стратегию согласования для пары результатов работы геномных сборщиков. Обычно один набор контигов считается «ведущим», а второй «ведомым», задача согласования заключается в объединении контигов из пары наборов, при этом необходимо обнаружить и изолировать проблемные регионы. Одной из первых программ для объединения набора контигов от разных сборщиков стала Reconciliator [6].

В ней производится поиск участков, являющихся уникальными как для ведомой, так и для ведущей последовательности. На следующем этапе закрываются пропуски в контигах из первого набора с использованием последовательностей из второго набора. В случае наличия нескольких вариантов выбирается тот, который отвечает лучшему статистическому критерию. Похожий подход использует GAA [5], строящий граф соответствия между наборами контигов, используемый для объединения сборок, и ZORRO [7], который предваряет шаг объединения этапом фильтрования контигов, содержащих ошибки. Программа GAM-NGS [4] ищет в контигах блоки соответствия, которые зависят от количества ридов, картирующихся на сравниваемые последовательности — таким образом, не проводят процедуру выравнивания каждого контига с каждым. Исходя из информации, полученной на стадии картирования ридов, строится граф сборок, анализируя который, можно установить участки несоответствия между наборами контигов.

Процесс объединения результатов работ геномных сборщиков является ресурсоемкой задачей. Время работы программы Reconciliator на одном из входных наборов данных составляет 24 часа, потребление оперативной памяти при этом составляет больше 100 Гб [6].

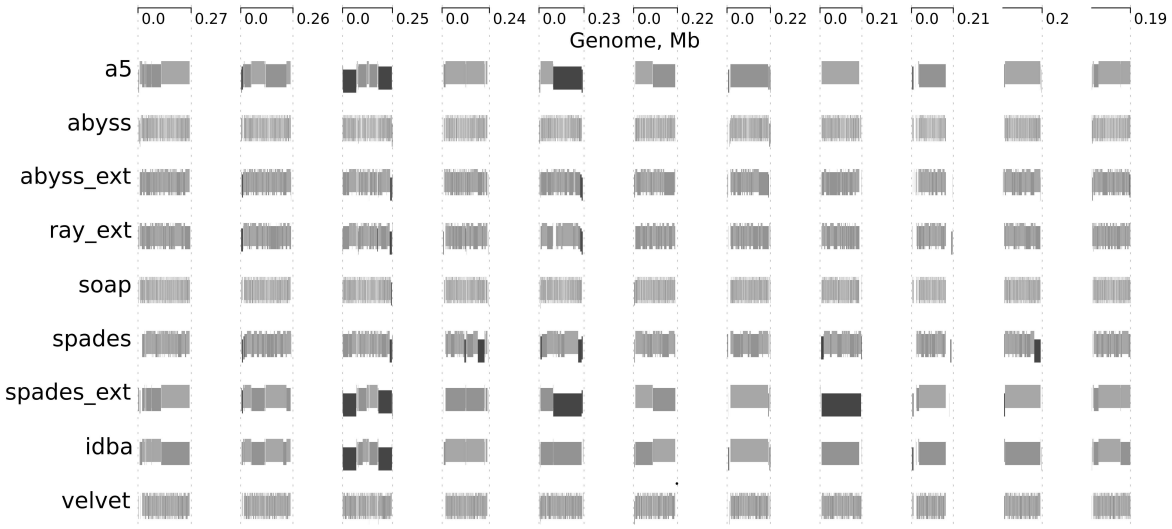

Рис. 1. Сравнение контигов, полученных от различных сборщиков, и референсных последовательностей. Объяснения см. в тексте

Все вышеперечисленные программные продукты являются последовательными, либо работают в контексте модели общей памяти, что ограничивает их масштабируемость.

3. Описание входных данных

Для тестирования предложенного метода был выбран геном гриба Encephalitozoon cuniculi, содержащий 11 хромосом, общая длина которых составляет около 2,5 миллионов нуклеотидов. Парные риды длиной 100 символов, полученные в результате секвенирования генома данного организма, доступны на сайте Европейского биоинформатического института [8], референс, состоящий из 11 последовательностей хромосом, опубликован на сате NCBI [9]. Из парных ридов нижеперечисленными сборщиками были получены последовательности контигов, которые и служили входными данными для описываемого метода. Наличие референсной последовательности для данного набора данных позволяет оценить корректность полученных сборок, поэтому параметры для запуска сборщиков подбирались таким образом, чтобы полученные ими результаты максимально соответствовали референсу. На рисунке 1 показан пример картирования набора контигов, полученного от разных сборщиков на референсную последовательность, построенный с помощью программы QUAST [10].

На горизонтальной оси располагаются результаты картирования наборов контигов на 11 хромосом референсной последовательности. Вдоль вертикальной оси расположены наборы контигов, полученные от разных сборщиков или от одного сборщика, но с разными параметрами. Участки, выделенные темным цветом, соответствуют ошибкам сборки (например, перестановке участков генома местами или объединению участков генома, которые в референсной последовательности не следуют друг за другом). Выравнивание контига и референсной последовательности изображается на рисунке сплошным прямоугольником соответствующей длины; таким образом, если в результате работы сборщика получилось много коротких контигов, это выражается в сильной фрагментированности диаграммы выравнивания.

4. Описание сборщиков

Ниже перечислены геномные сборщики, результаты работы которых объединяет предложенный метод. Все они используют концепцию графа де Брюйна и фигурируют в научных работах последнего времени, посвященных сравнению качества работы геномных сборщиков [11].

A5 — автоматически выбирает параметры сборки, внутри использует сборщик IDBA.

ABYSS — собирает геном в модели распределенной памяти, используя библиотеку MPI. Для определения участков перекрытий использует распределенную хеш-таблицу.

IDBA — итеративно изменяет параметры сборки, на каждой итерации дополняя риды контигами, полученными на предыдущей итерации.

RAY — может собирать геном в модели распределенной памяти, используя библиотеку MPI. Использует оригинальную эвристику для выявления повторов в геноме.

SOAPdenovo — использует разреженную хеш-таблицу для экономии оперативной памяти, что может приводить к ошибкам сборки. Для этого сборщика характерно относительно малое время работы.

SPAdes — использует различные параметры сборки для участков генома с различной глубиной покрытия.

Velvet — один из первых и самых распространенных сборщиков для коротких ридов. Необходимо вручную задавать большой набор параметров сборки.

5. Описание метода объединения

Прежде чем перейти к непосредственному описанию, необходимо отметить, что все перечисленные в разделе 2 программы объединения результатов работы геномных сборщиков позволяют сопоставлять только два набора контигов, причем зачастую один из них сформирован с использованием сборщика, принимающего на вход несколько наборов коротких фрагментов [4], что выражается в более аккуратном результате. Однако получение дополнительных наборов ридов связано с финансовыми и временными затратами, поэтому для большей части организмов доступен только один набор ридов. Описываемый метод позволяет объединять контиги, полученные с использованием только одного набора коротких фрагментов, для произвольного количества сборщиков.

Основная идея алгоритма заключается в построении распределенного взвешенного графа контигов, используемого для этапа согласования последовательностей. Реализация параллельного алгоритма создана на основе библиотеки MPI. Каждый MPI-процесс хранит в памяти часть вершин и ребер построенного графа. Вес ребра, соединяющего пару вершин, соответствующих контигам, определяется, в частности, исходя из результата парного выравнивания контигов. Для этого используется программа MEGABLAST, которая является вариацией программы BLAST, отличающейся от неё более высокой скоростью работы и ориентацией на поиск участков сходства с малым числом неточных совпадений. Одним из результатов работы MEGABLAST является score — значение, соответствующее степени сходства двух последовательностей.

Общая схема предлагаемого метода выглядит следующим образом:

1. Картирование ридов на наборы контигов с помощью программы REAPR [12] и выделение в контигах участков мало покрытых короткими чтениями. Такие участки могут сигнализировать о потенциальных ошибках сборки, поэтому, встретив такой контиг, REAPR разделяет его с целью уменьшения числа последовательностей, содержащих участки, не следующих друг за другом в референсном геноме.

2. Распределение контигов по процессам, сбор статистики: максимум, минимум, медиана длины контигов и т.д.

3. Каждый MPI-процесс для каждого контига производит выравнивание программой MEGABLAST со всем множеством контигов на всех процессорах. Каждый процесс строит матрицу значений score. При этом каждый процесс запрашивает у всех остальных процессов последовательности контигов, а в ответ отправляет номера своих контигов и вычисленные значения score.

4. Построение распределенного взвешенного графа контигов, где вершина – контиг, а ребро обозначает ситуацию перекрытия, каждому ребру поставлен в соответствие вес. Вес зависит от значения score и от длин пары перекрывающихся контигов.

5. Разбиение построенного графа с помощью эвристического алгоритма на непересекающиеся области так, чтобы максимизировать вес внутри каждой области и минимизировать вес ребер между областями. Если ребро рассматриваемого контига находится в памяти другого процесса, то ему отправляется сообщение типа точка-точка, ответом на которое будет служить вес ребра. Предполагается, что каждая область будет содержать максимальное число участков, соответствующих фрагментам одной хромосомы.

6. В каждой области, содержащей достаточное большое количество последовательностей, выполняется объединение контигов жадным алгоритмом, в итоговый результат попадают последовательности длиннее порогового значения.

Процесс разбиения графа позволяет уменьшить число перекрытий, получающихся в результате ошибки сборки или из-за наличия в геноме длинных повторов. Данный подход является масштабируемым, что достигается отказом от хранения графа контигов или матрицы попарного выравнивания в общей памяти.

6. Вычислительный эксперимент

Тестирование предложенного метода проводилось на суперкомпьютере «Ломоносов». Вначале были получены наборы контигов от разных сборщиков, а затем был запущен процесс объединения последовательностей. В Таблице 1 представлены результаты работы метода, для оценки результата использованы следующие метрики: число ошибок сборки (перестановка участков генома местами; объединение участков генома, которые в референсной последовательности не следуют друг за другом), общее число контигов и N50. Метрика N50 определяет величину, при которой контиги длиннее этого значения величины покрывают половину собранного генома и считается стандартом сравнения геномных сборок в случае отсутствия референса.

Из таблицы видно, что в процессе объединения результатов часть контигов была отфильтрована, что объясняется либо их малой длиной, либо малым количеством перекрытий с другими последовательностями.

7. Заключение

В качестве выводов можно отметить оправданность использования вычислений на кластерах, если объем данных на процессор достигает больших значений. В рамках дальнейшего исследования планируются следующие действия.

1) Проверка работы предложенного подхода для объединения сборок геномов, размер которых составляет около 1 Гб.

2) Изучение применимости графовых алгоритмов кластеризации для выделения сильносвязанных областей графа, соответствующих контигам, полученных из одной хромосомы.

3) Изучение возможности хранения контигов в сжатом виде (например, используя FM-индекс [13]) для экономии ОП и возможности работы с большими геномами.

Таблица 1. Результаты работы по объединению набора контигов

Сборщик	**Ошибки сборки**	**Число контигов**	**N50**
A5	14	16774	1401
Abyss	10	30933	9076
Ray	9	15087	10177
SOAPdenovo	2	6340	1558
Spades	8	42236	1883
IDBA	9	1533	65715
Velvet	3	3693	3344
Предложенный метод	5	1274	53187

Литература

1. Miller J.R., Koren S., Sutton G. Assembly algorithms for next-generation sequencing data // Genomics. June 2010. Vol. 95, N. 6. P. 315–327.

2. NCBI, БД Assembly, геном человека:
URL: `http://www.ncbi.nlm.nih.gov/assembly/883148` (дата обращения: 1.12.2014)

3. Метаинформация о геномах эукариотов на сайте NCBI:
URL: `ftp://ftp.ncbi.nlm.nih.gov/genomes/GENOME_REPORTS/eukaryotes.txt` (дата обращения: 1.12.2014)

4. Vicedomini R., Vezzi F., Scalabrin S., Arvestad L., Policriti A. GAM-NGS: genomic assemblies merger for next generation sequencing // BMC Bioinformatics. April 2013. Vol. 14(Suppl.7), N. 1.

5. Yao G., Ye L., Gao H., Minx P., Warren W.C., Weinstock G.M. Graph accordance of next-generation sequence assemblies // Bioinformatics. January 2012. Vol. 28, N. 1. P. 13–16.

6. Zimin A.V., Smith D.R., Sutton G., Yorke J.A. Assembly reconciliation // Bioinformatics. January 2008. Vol. 24, N. 1. P. 42–45.

7. Zorro – The masked assembler URL: `http://lge.ibi.unicamp.br/zorro/` (дата обращения: 1.12.2014)

8. European Nucleotide Archive URL: `http://www.ebi.ac.uk/ena/data/view/SRR122309` (дата обращения: 1.12.2014)

9. Encephalitozoon cuniculi GB-M1
URL: `http://www.ncbi.nlm.nih.gov/genome/39?genome_assembly_id=22671` (дата обращения: 1.12.2014)

10. Gurevich A., Saveliev V., Vyahhi N., Tesler G. QUAST: quality assessment tool for genome assemblies // Bioinformatics. April 2013. Vol. 29, N. 8. P. 1072–1075.

11. Koren S., Treangen T.J., Hill C.M., Pop M., Phillippy A.M. Automated ensemble assembly and validation of microbial genomes // BMC Bioinformatics. May 2014. Vol. 15, N. 1.

12. Hunt M., Kikuchi T., Sanders M., Newbold C., Berriman M., Otto T.D. REAPR: a universal tool for genome assembly evaluation // Genome Biology. May 2013. Vol. 14, N. 5.

13. Simpson J.T., Durbin R. Efficient construction of an assembly string graph using the FM-index // Bioinformatics. June 2010. Vol. 26, N. 12. P. 367–373.