Информационные процессы, Том 2, № 1, 2002, стр. 55–58.

Алгоритм поиска консервативных вторичных структур в наборе фрагментов РНК[1]

К.Ю.Горбунов, В.А.Любецкий

Институт проблем передачи информации, Российская академия наук
101447, Москва, ГСП-4, Б.Каретный пер., 19, Россия
e-mail: lyubetsk@iitp.ru, gorbunov@iitp.ru
Поступила в редколлегию 26.04.2002

Аннотация—Предложен алгоритм поиска сходных вторичных структур в наборе последовательностей РНК. Сложность алгоритма квадратичная от суммы длин входных последовательностей. Основная идея алгоритма — выравнивание последовательностей плеч возможных спиралей друг относительно друга.

1. ВВЕДЕНИЕ

Регуляторные и функциональные вторичные структуры РНК в родственных геномах во многих случаях обладают определенной консервативностью. На этой основе пытаются решить задачу поиска таких структур в данном наборе подходящих фрагментов РНК как систему количественно близких друг к другу сайтов из каждого фрагмента (или из большей части фрагментов). Сайты — это однотипные конфигурации слов из фрагментов. Такую систему еще называют сигналом. Сигнал может не включать представителей (сайтов) из какого-то числа фрагментов, так как предполагается, что некоторые фрагменты мусорные (ошибочные) и фактически не включают биологически значимых сайтов. Это обстоятельство заметно усложняет задачу. По-видимому, такая задача далека от эффективного алгоритмического решения в общей ситуации.

Вторичная структура состоит из шпилек, где шпилька состоит из двух упорядоченных наборов отрезков нуклеотидов — левого и правого, причём в каждом наборе между соседними отрезками имеются какие-то выпячивания, а между самими наборами имеется некоторая петля. Каждый i-ый отрезок от начала левого набора комплементарен i-ому отрезку от конца правого набора. Концы левого набора будем обозначать A и B, а концы правого набора - C и D (порядок везде от начала фрагмента к его концу). Отрезки от A до B и от C до D называют плечами шпильки (соответственно левым и правым). Таким образом, плечо — это набор отрезков вместе со всеми выпячиваниями в нем. Пару таких i-ых отрезков назовем спиралью, т.е. спираль — это шпилька, у которой плечи содержат по одному отрезку. Всякая шпилька однозначно образуется последовательным присоединением (считая от петли по B к началу, а по C к концу фрагмента) спиралей (а именно, k спиралей в случае шпильки с k отрезками в одном плече). Вторичная структура часто содержит достаточно длинные спирали.

Известны некоторые алгоритмы поиска консервативных вторичных структур. Например, в [1] для каждой пары фрагментов и каждой пары содержащихся в них слов методом динамического программирования строятся по возможности похожие и достаточно мощные структуры (начиная с коротких слов). В [2] для той же цели строится вывод структур в некоторой стохастической контекстно-свободной грамматике, и т.д. Используется полезная идея дерева содержащихся в данном слове шпилек (в этом дереве сыновья — это шпильки, расположенные в петле отца).

[1] Эта работа была поддержана грантом Минпромнауки.

Наш алгоритм основан на другом подходе: каждому фрагменту сопоставляется специальное слово, перечисляющее левые и правые плечи спиралей (вариант — плечи шпилек) из этого фрагмента. А именно, нумеруем спирали по середине их левого плеча (слева направо), затем выписываем знаки i_t, где i — номер спирали, а t — буква л или п, в том (линейном) порядке, в каком идут (например) середины левых и правых плеч спиралей. Так полученное (по каждому фрагменту) слово назовем специальным. Специальные слова образуются, чтобы затем выравнивать их алгоритмом типа Смита-Уотермена. Нами рассматривался и более сложный подход, когда специальное слово сопоставлялось отдельной шпильке из фрагмента, и затем выравнивались специальные слова, соответствующие паре шпилек из двух фрагментов. В консервативных структурах гомологи левых и правых плеч встречаются с сохранением отношения левое-правое. Отсюда возникает идея нашего алгоритма — выравнивание отдельно левых и правых плеч шпилек относительно друг друга.

2. ОПИСАНИЕ АЛГОРИТМА

Сначала для каждого из n фрагментов РНК строится список всех не продолжаемых (в обе стороны) спиралей, имеющих длины плеч больше, чем некоторый параметр, и длину петли в интервале, концы которого также являются параметрами. Получается n списков $L_1, \ldots, L_n$ спиралей. Рассматривался и случай, когда эти списки состояли из шпилек, определенным образом отобранных из соответствующих фрагментов. Для каждой пары спиралей (или шпилек — далее будем говорить только о случае шпилек) из разных списков подсчитывалась степень их сходства (которая заносилась в базу сходств спиралей). В биологической шпильке консервативные нуклеотиды могут лежать в отрезках, в выпячиваниях, в петле или даже на небольшом расстоянии от шпильки; это учитывается в программе. Поэтому для подсчета сходства двух спиралей по ним образовывалось два слова, например окрестность всей шпильки или ее части от A до C, или от A до B и т.п., которые сравнивались алгоритмом Смита-Уотермена; точнее, тем вариантом этого алгоритма, когда ищутся наиболее близкие подслова в данных словах (см., напр., [3]) (здесь нами использовались и другие сходные по цели алгоритмы). Окрестность определяется параметром ϵ, который указывает, сколько нуклеотидов добавляется с каждой стороны (например, $\epsilon = 15$). Наш алгоритм позволяет корректировать таким образом полученные сходства шпилек путем штрафов за различие длин их петель или просто за длинные петли, и т.п. Все сходства меньше заданного порога заменяются числом 0.

Теперь задача — так проредить списки $L_1, \ldots, L_n$, чтобы в них остались только шпильки, образующие искомую вторичную структуру в своем фрагменте. Первый, предварительный этап прореживания состоит в выбрасывании из каждого списка шпилек, для которых лишь в малом числе других списков найдутся шпильки со сходством выше порога.

Второй этап прореживания состоит в следующем. Множество шпилек из списка превращаем в специальное слово из их левых и правых плеч, как это описано во введении. В случае, когда списки состоят из спиралей, каждому фрагменту сопоставляем соответствующее специальное слово. Для каждой пары специальных слов применяем тот же вариант алгоритма Смита-Уотермана (или сходный по цели алгоритм). При этом левым плечам разрешается выравниваться лишь с левыми, а правым — с правыми плечами. Приз за выравнивание двух плеч равен сходству соответствующих спиралей, которое берётся из упомянутой базы. Учитывая, что в исходных списках обычно присутствует большое число лишних спиралей, задается нулевой или небольшой штраф за соответствие плечу пробела.

Затем каждой спирали (шпильке) ставится в соответствие качество, равное сумме весов всех ее выравниваний с другими спиралями (шпильками). При этом в качество добавляется приз за полное спаривание спирали, когда её плечи спарились с плечами какой-то одной спирали. Ещё приз даётся спирали за то, что две спарившиеся с ней (обоими плечами) спирали спарились и между собой. Использовались еще некоторые такого рода правила для спиралей и шпилек.

Шпильки с качеством ниже порога удаляются из списков; как и сами фрагменты, для которых сумма качеств их шпилек (назовем ее качеством фрагмента) ниже порога. Оставшиеся шпильки

участвуют во второй итерации аналогичных выравниваний, после которой качества шпилек и фрагментов подсчитываются заново. Вторая итерация имеет два отличия. Во-первых, сходство спиралей не просто берется из базы, а корректируется с учётом их качеств, полученных на первой итерации. Во-вторых, при подсчете качеств спиралей приз дается лишь за одновременное спаривание левых и правых плеч. Программа допускает любое число таких итераций, но наше тестирование показало, что обычно достаточно двух итераций. После завершения первого этапа прореживания списков составляется общий для всех фрагментов список L спиралей (шпилек) в порядке убывания их качеств до некоторого порога; снова удаляются фрагменты с качеством ниже порога.

Затем начинается второй этап алгоритма — построение отдельных вторичных структур в каждом из оставшихся фрагментов из спиралей, оставшихся в этом фрагменте. В каждом из этих фрагментов структура строится модифицированным нами алгоритмом Нуссинов-Джекобсона. Как известно, этот алгоритм [4] строит в данном фрагменте (и в каждом его слове) наиболее мощную структуру методом динамического программирования (начиная с коротких слов). В нашем случае структура строится, во-первых, из плеч спиралей, входящих в список L (не из нуклеотидов), а, во-вторых, максимизируется сумма качеств спиралей (не мощность структуры). Если известно, что искомая структура имеет вид, например, клеверного листа, т.е. одну спираль с длинной петлёй, на которой последовательно расположены несколько спиралей, с относительно короткими петлями (скажем, случай тРНК), то алгоритм предусматривает следующее уточнение (имеется аналогичная возможность и для других типов вторичных структур). На втором этапе левое плечо спирали склеивается с правым лишь, если у спирали короткая или соответственно длинная петля, на которой уже имеется заданное число спиралей. Такое усовершенствование предусмотрено и на этапе выравнивания плеч. В алгоритме предусмотрен дополнительный учет консервативных нуклеотидов.

3. РЕЗУЛЬТАТЫ СЧЕТА

Приведём результат тестирования алгоритма на 18 фрагментах тРНК кишечной палочки. Ниже после номера и названия антикодона перечисляются найденные нашим алгоритмом спирали этой вторичной структуры в каждом фрагменте. Буква Н обозначает нижнюю спираль (черенок), Л — левую, В — верхнюю, П — правую спирали; отсутствие буквы означает, что соответствующая спираль не найдена; числа после буквы в квадратных скобках означают погрешности по концам петли, их отсутствие означает, что соответствующая максимально продолженная спираль найдена точно. Число в круглых скобках указывает число ложных полученных алгоритмом спиралей.

DA1660 TGC: Н,Л,В,П (1); DA1661 GGC: Н,Л,В,П (1); DC1660 GCA: Н,В,П (0);
DD1660 GTC: Н,В,П (1); DE1660 TTC: Н,П (2); DF1660 GAA: Н,Л,В,П (0);
DG1660 TCC: Н,В,П (1); DG1661 GCC: Н,Л,В,П (1); DG1662 CCC: Н,Л,В,П (0);
DH1660 GTG: Н,Л,В,П[1,2] (1); DI1660 GAT: Н,Л,В,П (0); DI1661 CAT: Н,Л,В,П (1);
DK1660 TTT: Н,Л,В,П (0); DL1660 CAG: В,П (2); DL1661 TAG: Н,П (2);
DL1662 CAA: Н,В,П (2); DL1663 GAG: Н,В,П (1); DL1664 TAA: Н,В,П (0).

Было проведено тестирование алгоритма и для других случаев: RFN-структуры [5], Т-боксы. RFN-структура состоит из черенка и четырех спиралей-листьев (другие биологические спирали обычно не обладают консервативностью и не рассматриваются). Ниже приведен результат запуска алгоритма на выборке из 20 RFN-фрагментов. Цифра 1 означает черенок, цифры 2,3,4,5 — соответсвенно спирали-листья (занумерованные по часовой стрелке). Перед каждым результатом указана аббревиатура генома и название гена, из 5’ области которого вырезан соответствующий фрагмент. Смысл остальных обозначений тот же, что и выше.

BS ribD: 2[1,4],3[2,2],5[2,2](3); BQ ribD: 2[9,6],3,5(5); BE ribD: 1,3[0,12],4[0,3],5[2,2](4);
HD ribD: 2,5[2,2](4); CA ribD: 1,2,3[3,0],5[2,2](3); DF ribD: 1,2,3[2,2],4[0,6],5[2,2](4);
SA ribD: 1,2[5,0],3[2,2],5[2,2](3); LLX ribD: 1,2,3[2,2],4,5(4); PN ribD: 1,2,3,4[0,6],5[2,2](4);

TM ribD: 1,2[1,4],3,5[2,2](4); BS ypaA: 3,5[2,2](6); BQ ypaA: 1,2[1,4],4,5[2,2](4);
BE ypaA: 3[1,16],4[12,2],5[4,5](4); SAypaA: 1,2[1,4],3[2,2],5[2,2](3);
DF ypaA: 1,2[6,0],3[2,2],4,5[2,2](4); EF ypaA: 1,3,5[2,2](4); LLX ypaA: 2,3[1,11],5(1);
ST ypaA: 1,2,3,5(3); SA ypaA: 2[5,0],3[2,2],4[2,5],5[2,2](3); AMI ypaA: 2,3[1,5](2).

Как видно, четвертая спираль находится существенно хуже остальных. Это объясняется низкой консервативностью соответствующего участка РНК.

4. ЗАКЛЮЧЕНИЕ

Программа допускает ещё один третий этап: сравнение между собой построенных структур с целью оценки качества полученного сигнала и построения консенсусной вторичной структуры вместе с её (частичными) отображениями в построенные структуры. Отображения позволяют предсказывать шпильки, которые не были включены алгоритмом в построенные им ответы. Отметим, что первый этап алгоритма работает и в случае, когда ищется структура с псевдоузлами, т.е. ищутся шпильки, содержащие в своей петле лишь одно плечо другой шпильки. В этом случае на втором этапе (вместо аналога алгоритма Нуссинов-Джекобсона) можно использовать аналогичным образом аналог недавно предложенного алгоритма из [6], который уже допускает псевдоузлы. Алгоритм из [6] в худшем случае имеет большое время работы (шестая степень), но, поскольку у нас на втором этапе остаётся мало спиралей, модификация алгоритма из [6] дает хороший результат в качестве второго этапа нашего алгоритма.

Авторы благодарят М.С. Гельфанда и А.А. Миронова за помощь и многочисленные разъяснения биологического содержания задачи.

СПИСОК ЛИТЕРАТУРЫ

1. Gorodkin J., Heyer L.J. and Stormo G.D. Finding the Most Significant Common Sequence and Structure Motifs in a Set of RNA Sequences. *Nucleic Acids Res.*, 1997, vol. 25, pp. 3724–3732.
2. Eddy S., Durbin R. RNA Sequence Analysis Using Covariance Models. *Nucleic Acids Res.*, 1994, vol. 22, pp. 2079–2088.
3. *Математические методы для анализа последовательностей ДНК.* Пер. с англ. Под ред. М.С. Уотермена. М.: Мир, 1999. 349 с.
4. Nussinov R., Jacobson A.B. Fast Algorithm for Predicting the Secondary Structure of Single-Stranded RNA. *Proc. Natl Acad. Sci. USA*, 1980, vol. 77, pp. 6309–6313.
5. Vitreschak A.G., Rodionov D.A., Mironov A.A., Gelfand M.S. Regulation of Riboflavin Biosynthesis and Transport Genes by a Conserved RNA Structural Element. *In press.*
6. Rivas E. and Eddy S.R. A Dynamic Programming Algorithm for RNA Structure Prediction Including Psevdoknots. *J. Mol. Biol.*, 1999, vol. 285, pp. 2053–2068.

Статью представил к публикации член редколлегии В.А. Любецкий