К.Ю.Горбунов, В.А.Любецкий

АЛГОРИТМ ВЫЯВЛЕНИЯ РЕГУЛЯТОРНОГО СИГНАЛА В НАБОРЕ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

Abstract. *An algorithm of quadratic (or near) complexity that searches for a common signal in a given set of sequences (in particular DNA sequences), i.e. a system of similar words of a fixed length and satisfying some certain conditions, has been developed.*

Идея этой работы обсуждалась с профессором В.А.Смирновым – светлой памяти Владимира Александровича авторы посвящают ее. Здесь рассматривается задача, имеющая большую область весьма современных приложений (в частности, в компьютерной генетике). В скобках указаны значения параметров этой задачи, характерные для приложения в генетике, которое обсуждается ниже в пункте 2.

1. Дан алфавит (например, из четырех букв A и T, C и G, буквы из этих пар называются соответственно комплементарными друг другу). Дан набор из k последовательностей (примерно) одинаковой длины n каждая. Системой назовем набор слов фиксированной длины l, по одному слову (или по фиксированному числу слов – сейчас для краткости рассмотрим случай одного слова) из одной последовательности; в систему должны включаться слова, по крайней мере, из 2/3 всех исходных последовательностей; система должна состоять из как можно более попарно похожих друг на друга слов (например, в смысле суммы попарных расстояний Хэмминга между словами системы или в смысле какой-то другой фиксированной метрики между словами, или в смысле максимизации какого-то фиксированного «качества системы»). Например, для приложения из пункта 2 естественно говорить о максимизации качества системы как суммы попарных «расстояний» между словами системы, вычисляемых с помощью функции $F(x,y)$, которая для двух слов x и y длины l отражает прежде всего степень их похожести между собой, а затем и степень их палиндромности (т.е. похожести x на $pal(x)$, см. ниже) и некоторые другие их свойства (например, похожесть слов x и y на обучающую выборку слов длины l, когда таковая дана). В качестве примера укажем на функцию $F(x,y) = S(x,y) + 0.5\,(\max(S(x,pal(x)),\ S(x',pal(x'))) + \max(S(y,pal(y)),\ S(y',pal(y'))))$, где $S(x,y)$ – количество совпадающих

букв в словах x и y, а $pal(x)$ – слово, полученное из x обращением с заменой каждой буквы на комплементарную, x' – слово x без последней буквы.

Интуитивно такая система понимается как «сигнал», а последовательность – как «регуляторная область»; проблема в том, что некоторые регуляторные области не содержат сигнала (они ошибочно выписаны); сигнал во всех «истинных» регуляторных областях (т.е. таких, которые его содержат) служит для одновременного запуска процесса, соответствующего набору всех «истинных» регуляторных областей. Более содержательная интерпретация дается в пункте 2.

Описание алгоритма. Сначала образуем вспомогательный граф G, который остается фиксированным в алгоритме. Граф G состоит из k вершин и всех ребер, которые возникают в процессе выполнения следующей процедуры. На первом шаге все вершины графа G разбиваются на две равные (с точностью до единицы, если k нечетное) части и между этими частями проводятся два ребра (A,B) и (C,D), не выходящие из одной вершины (пусть, скажем, A и C находятся в одной части, а B и D в другой). Любое из этих ребер (скажем, (A,B)) назовем *основным относительно этого разбиения*, а другое – *вспомогательным*. Также проводятся два «диагональных» ребра: (A,D) и (C,B). Каждую из двух полученных частей снова разбиваем на две (в том же смысле) равные части так, что A и C, как и B и D, находятся в разных частях этих разбиений. Основные ребра относительно уже этих разбиений определены однозначно: это (A,C) и (B,D), а вспомогательные ребра (по возможности, не выходящие из той же вершины, что и основные) выбираются произвольно. И так далее, каждую появившуюся в этой процедуре неодновершинную часть P разбиваем на две равные части так, чтобы основные ребра двух текущих разбиений соединяли концы основного и вспомогательного ребер предыдущего разбиения. Процедура разбиений прекращается, когда все части P станут одновершинными; на самом деле, можно остановиться, когда эти части станут мелкими (из 1-3 вершин).

Внешний цикл алгоритма состоит во взаимно однозначном приписывании каждой вершине графа G одной из исходных последовательностей (одну из таких текущих расстановок последовательностей по вершинам графа G обозначим r). Вопрос о том, как лучше выбирать такое приписывание (такую расстановку), является центральным. Суть дела в том, что качество системы s (иными словами, качество «сечения» s над G) определяется как сумма значений функции $F(x,y)$ по всем парам вершин графа G (т.е. как сумма «по всем ребрам» графа G, как если бы он был пол-

ным графом; это качество обозначим *H(s)*; здесь рассматривается случай, когда в систему входит по слову из каждой последовательности). И мы хотим приблизить это качество, беря вместо *H(s)* величину *H(s,r)*, определяемую как сумма значений функции *F(x,y)* по всем тем парам вершин графа *G*, которые в нем действительно соединены ребром. (В сумме *H(s,r)* гораздо меньше слагаемых, чем в сумме *H(s)*, и ее вычисление происходит гораздо быстрее.)

Конкретно, для данного *r* выполняется цикл (называемый «сборкой»), который мы опишем индукцией по глубине разбиений. Индуктивный шаг: пусть для двух частей *P1* и *P2* с основными ребрами соответственно *(A,C)* и *(B,D)*, полученными разбиением подграфа *P* с основным ребром *(A,B)*, уже определены два набора из *t* лучших сечений как продолжений с их основных ребер (точнее, для любых двух слов из последовательностей над *A* и *C* с качеством большим некоторого фиксированного порога определены *t* лучших продолжений соответственно на все множество *P1*; и аналогично для *P2).* Тогда для любых слов (скажем, *x* и *y)* из последовательностей над *A* и *B* с качеством большим того же порога определим *t* лучших продолжений на все множество *P* (т.е. на объединение множеств *P1* и *P2*) следующим образом. В цикле рассмотрим все слова (скажем, *x1* и *y1)* из последовательностей над *C* и *D,* для которых качество слов *x* и *x1* и *y* и *y1* выше этого порога; и для *x,x1* и *y,y1* выберем соответствующие продолжения, объединяя их подходящим образом, получим *t* лучших сечений над всем *P.* Если условие пороговости не может быть обеспечено, то соответствующее значение сечения над *P* считается по определению равным нулю; иными словами, в этом случае возникают частично определенные сечения над *G*. Кроме того, чтобы среди *t* лучших сечений, получаемых на различных шагах сборки, было меньше таких, которые не дают новых сигналов (по сравнению с уже «утвержденными» сечениями), проводится проверка каждого нового сечения *s* на существенность относительно списка *S* уже «утвержденных» сечений (с той же областью определения). Для этого, если на достаточно большой доле последовательностей сечение *s* не имеет новых по сравнению с *S* слов, предполагаем, что среди этих «не новых» слов из *s* значительную часть составляет сигнал, и смотрим, насколько близко к этой «сигнальной» совокупности каждое оставшееся слово из *s*. Если среди них не обнаружено близких к этой совокупности слов, то отвергаем *s* и переходим к следующему кандидату в список.

Цикл, состоящий в расстановке последовательностей по вершинам графа *G,* работает, по крайней мере, до тех пор, пока любая

пара последовательностей хотя бы раз не соединится ребром в графе *G*. Процедура расстановки устроена так, чтобы на каждой итерации по *G* «покрыть» ребрами графа *G* больше пар последовательностей, не покрытых на предыдущих итерациях (по *G*). Это обеспечивает разумное количество итераций этого цикла при достаточном разнообразии распределений последовательностей по вершинам графа *G*. Последнее важно для получения достаточно представительной статистики в следующем, последнем цикле работы алгоритма.

А именно, каждой позиции в каждой последовательности (содержащей, скажем, букву *i*) ставится в соответствие число, которое отражает меру того, что буква *i* входит в искомый сигнал. Это число равно сумме качеств по всем полученным сечениям, которые включают слово, содержащее эту букву. Здесь под качеством понимается качество не всего сечения, а именно данного слова в нем, т.е. сумма значений $F(u,x)$, где u – данное слово (содержащее *i*), а x пробегает все остальные слова этого сечения. Таким образом, позиции букв, входящих в сигнал, будут помечены в исходных последовательностях числами, которые заметно больше чисел, стоящих в других позициях этих же последовательностей.

Счет, проведенный на многих примерах из приложения, описанного в следующем пункте 2, показал, что в случаях, когда сигнал был известен, алгоритм находил его, а в других случаях алгоритм находил вполне содержательный (с точки зрения этого приложения) ответ. Компьютерная реализация и счет были проведены Л.В. Даниловой. Математическое исследование такого рода алгоритмов весьма сложно (в сущности, здесь речь идет о дискретном динамическом процессе); авторы предполагают опубликовать некоторый анализ этого алгоритма.

2. Карта (иными словами, граф) метаболических путей бактерии (например, кишечной палочки) описывает, в принципе, все протекающие в этом организме химические реакции (ребро – химическая реакция, вершина – соответствующее химическое вещество). У родственных бактерий такие карты близки (иногда, с заменой некоторых веществ на гомологичные). Эти химические реакции объединены в более или менее сложные одновременно протекающие каскады реакций (небольшие подграфы в карте метаболических путей, удовлетворяющие определенным условиям) с целью производства веществ, необходимых для жизнедеятельности бактерии (например, пурина или тирозина). Такие каскады иногда пересекаются. Протекание одного каскада начинается сразу после того, как клетка (одновременно)

продуцировала набор ферментов, обеспечивающих этот каскад (можно считать, что каждому ребру приписан соответствующий фермент).

Производство одного фермента соответствует активности, обычно, одного гена бактерии. Поэтому клетка содержит механизмы одновременного запуска многих разных групп генов, соответствующих многим разным каскадам химических реакций. Одним из таких механизмов является механизм совместного запуска (обычно говорят – совместной регуляции), который основан на одновременном связывании многих копий белковой молекулы (в случае эукариотов – нескольких белковых молекул) с определенными участками перед генами из данной их группы. В этом случае группа генов называется регулоном, а эти участки – оператором регулона (они находятся в области генома бактерии перед геном регулона, эта область называется регуляторной); соответствующая молекула называется фактором транскрипции. Итак, перед каждым геном (или группой генов – тогда их называют опероном) из данного регулона в соответствующей ему регуляторной области находится оператор – участок связывания с фактором транскрипции (что и приводит к активации или репрессии регулона).

Оператор состоит из нескольких похожих слов с фиксированным расстоянием между ними. Эти слова могут быть произвольными или описываться как в той или иной мере слабые палиндромы, в достаточной мере отличные от случайного слова. Заметим, что фактор транскрипции продуцируется согласно той же карте метаболических путей; отсюда возникает важная задача нахождения формально-логического языка для описания всей этой сложной ситуации.

Итак, возникает следующая задача, решаемая на основе алгоритма из пункта 1. Дан набор (предполагаемых) регуляторных областей одного регулона (одной или нескольких бактерий – во втором случае для каждой бактерии анализируем результаты отдельно и затем сравниваем; такое сравнение полезно организовывать и в случае одной бактерии, деля исходный материал, скажем, на две части). Часто это набор не всех регуляторных областей данного регулона. В каждой из регуляторных областей оператор может состоять из нескольких слов. Ищем систему похожих друг на друга слов в этих регуляторных областях (с отбраковкой областей, не содержащих сигналов). При этом иногда дана еще обучающая выборка для искомого оператора. Найдя оператор, пополняем регулон в этой бактерии и находим его в других бактериях; затем тестируем найденные регулоны на

метаболическую осмысленность и одинаковость (у разных бактерий) по набору генов.

Авторы сердечно благодарят профессора М.С. Гельфанда за многократные объяснения, касающиеся этой задачи, и обсуждение результатов счета.