BLAST

Basic Local Alignment Search Tool

С.А. Спирин, 19 апреля 2022

BLAST – алгоритм для нахождения участков локального сходства между последовательностями

Алгоритм сравнивает входную последовательность с последовательностями в базе данных, ищет сходные последовательности в базе данных и оценивает статистическую значимость находок.

Напоминание: сходство и гомология

Гомология — общность происхождения

- У гомологичных белков можно говорить о парах гомологичных остатков
- В эволюционно правильном выравнивании все остатки в одной колонке гомологичны друг другу

Признак гомологии — сходство последовательностей

- Для выявления сходства последовательности надо выровнять
- Подбирают оптимальное выравнивание, то есть имеющее наибольший вес
- Оптимальное выравнивание существует для любых последовательностей, в том числе негомологичных
- Для двух последовательностей можно рассматривать или глобальное, или локальное выравнивание

Алгоритмы и программы оптимального парного выравнивания

- Оптимальное **глобальное** выравнивание: алгоритм **Нидлмана – Вунша**
  Needleman & Wunsch (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". *Journal of Molecular Biology*. **48** (3): 443–53
  В оригинальной работе предлагалось оценивать выравнивание с линейными штрафами за гэпы (одинаковый штраф за каждый гэп) и описывался алгоритм нахождения оптимального выравнивания с таким весом. Позднее был предложен вес с аффинными штрафами и алгоритм модифицирован для этой ситуации, ещё позднее введены матрицы замен.
  Программы в EMBOSS: **needle** и **stretcher**
- Оптимальное **локальное** выравнивание: алгоритм **Смита – Уотермена** (Смита – Ватермана)
  Smith & Waterman (1981). "Identification of Common Molecular Subsequences". *Journal of Molecular Biology.* **147** (1): 195–197
  Программа в EMBOSS: **water**
- Близкий алгоритм Уотермена – Эггерта выдаёт не одно оптимальное, а заданное число лучших выравниваний (Waterman & Eggert (1987). *Journal of Molecular Biology*. **197** (4): 723-728)
  Программа в EMBOSS: **matcher**

Параметры этих программ = параметры вычисления веса:

- матрица замен (для белков)
- штраф за первый гэп инделя (gap opening penalty)
- штраф следующие гэпы инделя (gap extension penalty)

Идея поиска гомологов в банке последовательностей

На входе — последовательность, для которой хочется найти гомологичные («запрос»), и банк

Выровняем запрос с каждой последовательностью банка, посчитаем веса этих парных выравниваний

Отберём те последовательности банка («находки»), для которых вес **существенно выше, чем мог бы быть по случайным причинам.**

Почему локальное выравнивание?

Глобальное выравнивание следует применять только в случае заранее известной гомологии последовательностей по всей длине.

Часто у последовательностей гомологичны только отдельные части (примеры: гомеобелки, полипротеины, …)

Если про белки заранее ничего не известно, то более информативным будет локальное выравнивание. Поэтому именно оно применяется при поиске в банках данных.

Protein BLAST: поиск гомологов данного белка в банке аминокислотных последовательностей

Алгоритмы
- BLASTP
- Quick BLASTP
- PSI-BLAST
- PHI-BLAST
- DELTA-BLAST

Алгоритм Смита – Уотермена не используется, он слишком медленный для этой задачи

Алгоритм BLASTP **не гарантирует** нахождение оптимального выравнивания, но зато работает очень быстро

Можно использовать:
- из командной строки (standalone BLAST)
- через веб-интерфейс

Что подаётся на вход программе BLAST?

- Последовательность запроса
- Банк последовательностей
- Параметры:
  - параметры выравнивания: матрица аминокислотных замен, штрафы за гэпы;
  - параметры поиска: длина слова и другие (см. далее);
  - параметры выдачи: максимальное число находок, пороги на качество выравнивания, форма выдачи (обычная, табличная, формат ASN, …)

Что выдает BLAST?

Выдача самой программы состоит из четырёх частей:

- заголовок с описанием программы, банка, запроса (query);
- список находок;
- выравнивания запроса с находками;
- несколько строк со статистическими показателями.

Веб-интерфейсы тем или иным способом перерабатывают выдачу программы. Раздел со статистикой обычно не показывается. Часто вставляется графическое изображение находок.

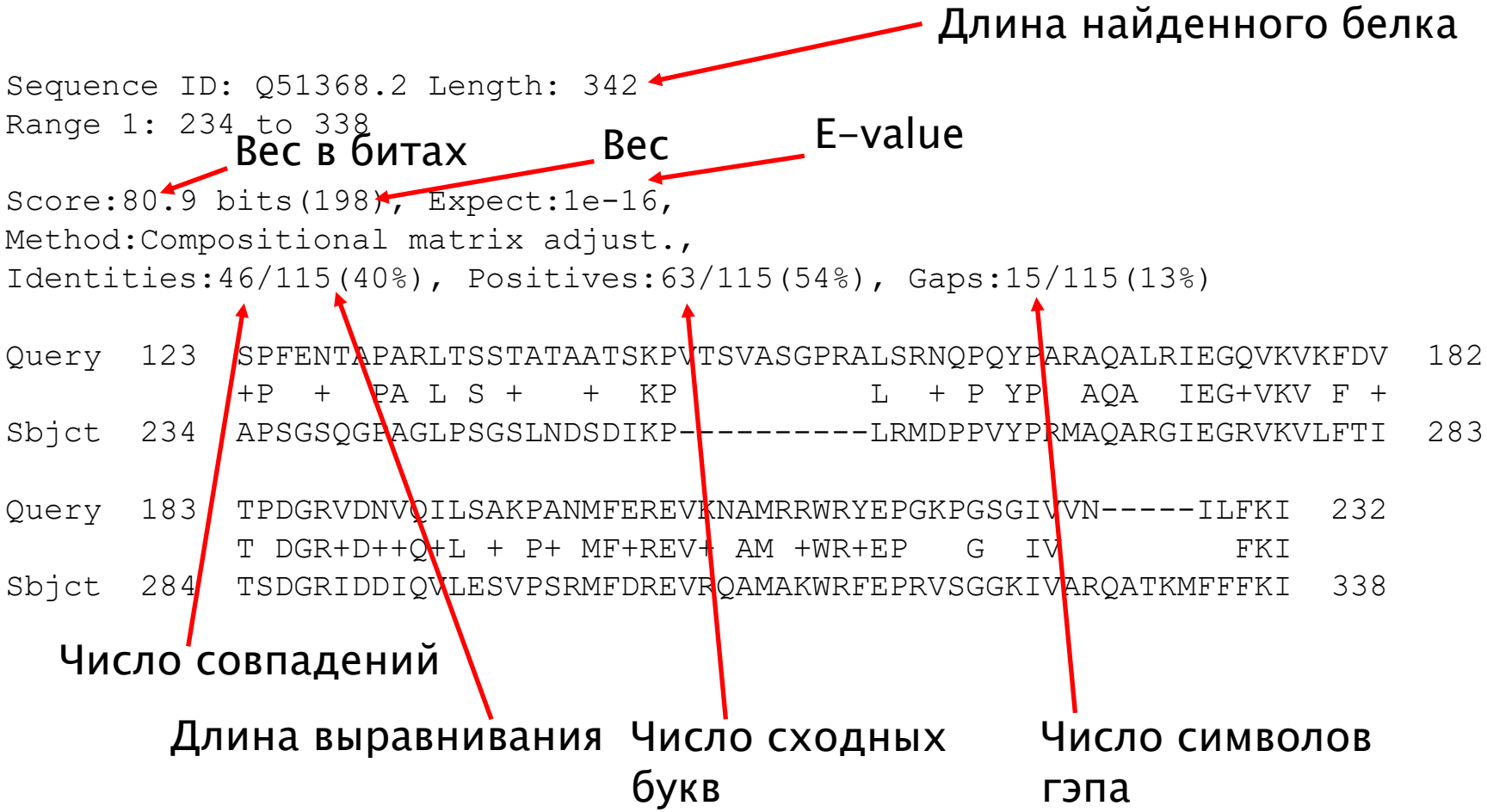
Выравнивание, выданное BLAST

```
Sequence ID: Q51368.2 Length: 342
Range 1: 234 to 338

Score:80.9 bits(198), Expect:1e-16,
Method:Compositional matrix adjust.,
Identities:46/115(40%), Positives:63/115(54%), Gaps:15/115(13%)

Query  123  SPFENTAPARLTSSTATAATSKPVTSVASGPRALSRNQPQYPARAQALRIEGQVKVKFDV  182
            +P  +  PA L S +   +  KP          L  + P YP  AQA  IEG+VKV F +
Sbjct  234  APSGSQGPAGLPSGSLNDSDIKP----------LRMDPPVYPRMAQARGIEGRVKVLFTI  283

Query  183  TPDGRVDNVQILSAKPANMFEREVKNAMRRWRYEPGKPGSGIVVN-----ILFKI  232
            T DGR+D++Q+L + P+ MF+REV+ AM +WR+EP   G  IV         FKI
Sbjct  284  TSDGRIDDIQVLESVPSRMFDREVRQAMAKWRFEPRVSGGKIVARQATKMFFKI  338
```

Участок найденного белка, попавший в выравнивание

Отображение консервативности: между одинаковыми буквами ставится эта же буква, между сходными (positive) — знак +

Выравнивание, выданное BLAST



Словарик BLAST

Identities — совпадения
Positives — сходные буквы, то есть те, для которых значение матрицы положительно
Gaps — знаки гэпа "–" (не индели!)

Для всех трёх приводится их число в виде числителя со знаменателем из длины выравнивания (не длины находки!) и процент от длины выравнивания

Score — вес выравнивания. Приводится в двух видах: сначала в битах (см. далее), затем в скобках обычный = сумма значений матрицы по сопоставлениям минус штраф за гэпы

Expect — E-value, то есть ожидаемое число выравниваний с тем же или большим весом. Запись вида 9e-15 означает $9 \cdot 10^{-15}$.

E-value – ожидаемое количество случайных находок с таким же и лучшим весом выравнивания, при поиске в той же базе данных, со случайным запросом той же длины и состава, с теми же параметрами на вычисление веса выравнивания.

В выдаче BLAST E-value называется “Expect”

Чем **меньше** E-value, тем **выше** значимость находки.

E-value зависит от:
– веса выравнивания (чем больше вес, тем **меньше** E-value);
– размера банка (чем больше банк, тем больше E-value);
– длины запроса (чем длиннее запрос, тем больше E-value);
– параметров, используемых для вычисления веса.

E-value – **ожидаемое** количество **случайных** находок с таким же и лучшим весом выравнивания, при поиске в той же базе данных, со случайным запросом той же длины и состава, с теми же параметрами на вычисление веса выравнивания.

Что означает слово «ожидаемое»?

E-value – **ожидаемое** количество **случайных** находок с таким же и лучшим весом выравнивания, при поиске в той же базе данных, со случайным запросом той же длины и состава, с теми же параметрами на вычисление веса выравнивания.

Что означает слово «ожидаемое»?

Формально это то, что называется «математическое ожидание случайной величины». Случайной величиной в данном случае является **число находок** *(NB! Просьба запомнить!)*

На практике ожидаемое вычисляется как **среднее** по достаточно большому количеству испытаний.

Другое ключевое слово — «случайных». Нам нужно понять, сколько можно ожидать именно случайных, то есть бессмысленных, негомологичных находок, чтобы оценить, насколько надёжно утверждение, что данная находка — действительно гомолог.

Как посчитать E-value

Прямой способ — вычислительный эксперимент:
перемешать буквы в запросе очень много раз, каждый раз запуская BLAST, и посмотреть, сколько в среднем при одном запуске бывает находок с весом выше данного.

Такой способ, естественно, не применяется :)

Стоит подумать: от чего и как может зависеть число случайных находок

Как посчитать E-value

Имеется замечательная теорема (С.Карлина):

$$\text{E-value} = Kmn \cdot e^{-\lambda S}$$

S – Score (вес)
m – длина исходной последовательности
n – размер базы данных (суммарная длина всех последовательностей)
K и λ – две константы

Коэффициенты K и λ зависят от параметров вычисления веса, то есть матрицы и штрафов за гэпы.

BLAST хранит значения K и λ для нескольких наборов параметров вычисления веса (их раз и навсегда нашли посредством вычислительного эксперимента).

Вес в битах

Вес в битах B зависит от обычного веса S и параметров вычисления веса. Эта зависимость подобрана так, чтобы

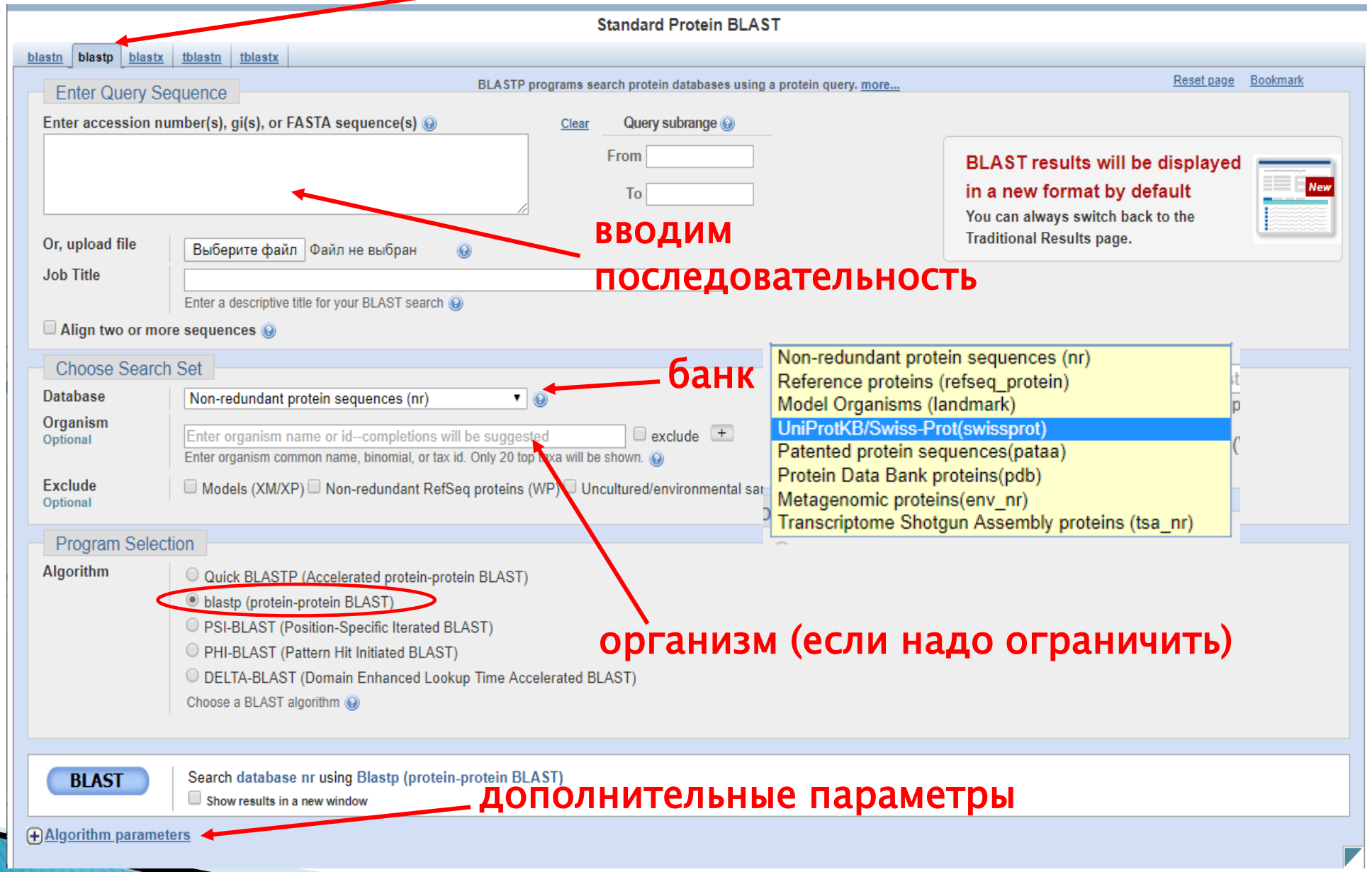
$$\text{E-value} = mn \cdot 2^{-B}$$

m – длина исходной последовательности
n – размер базы данных
(констант K и λ теперь нет, они “загнаны внутрь B”)

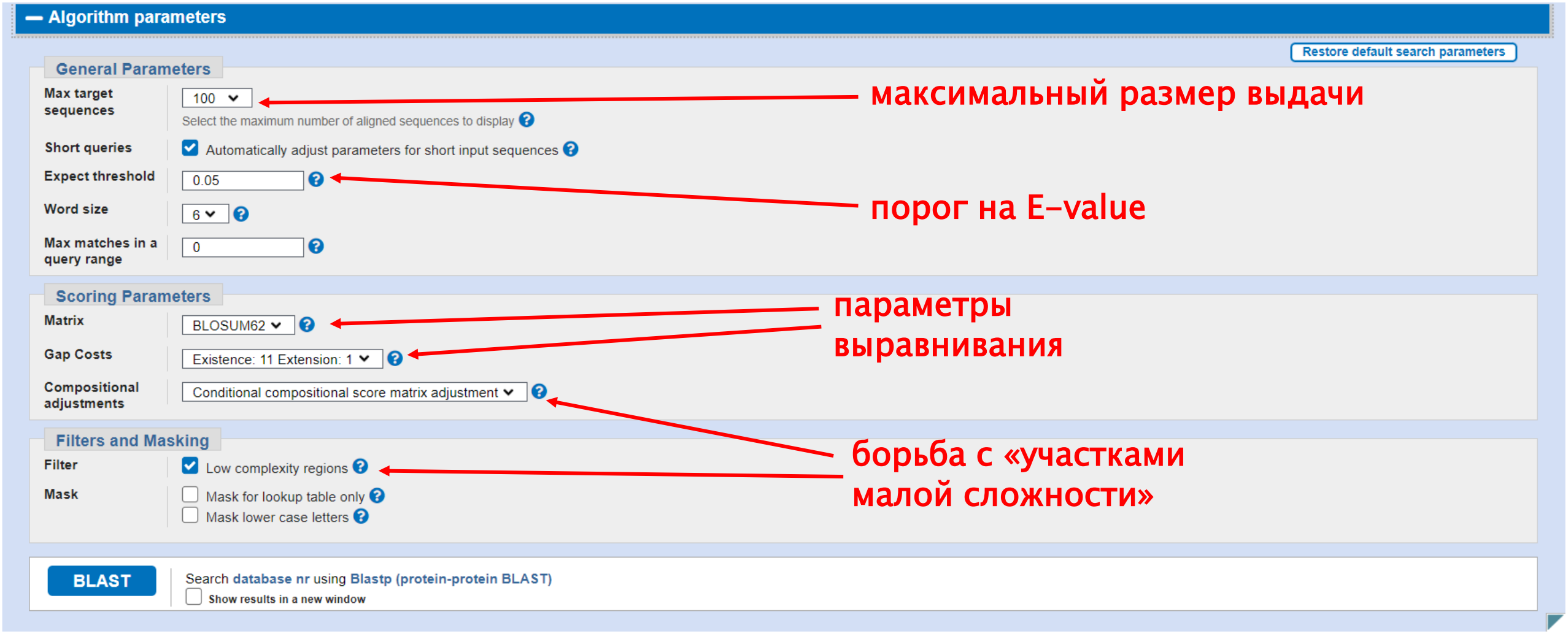
Нетрудно подсчитать, что $B = (\lambda S - \ln K)/\ln 2$

Далее описан интерфейс, установленный на «родине» BLAST: National Center for Biotechnology Information (NCBI) в США, http://blast.ncbi.nlm.nih.gov/

https://blast.ncbi.nlm.nih.gov/ → Protein BLAST



Дополнительные параметры



Участок малой сложности

Ищем по белку P02929
Если отключить “Compositional adjustment” и фильтр, то среди прочих выдаётся следующее выравнивание:

```
Query: P02929 TONB_ECOLI; Subject: Q95P09 TSEP_GLOMM
 Score = 63.5 bits (153),  Expect = 1e-09
 Identities = 32/76 (42%), Positives = 47/76 (62%), Gaps = 6/76 (8%)

Query  56   EPPQAVQPPPEPVVEPEPEPEPIPEP-PKEAPVVIEKPKPKPKPKPKPVKKVQEQPKRDV  114
            EP    +P PEP  EPEPEPEP PEP P+  P    +P+P+P+P+P+P  + + +P+ +
Sbjct  243  EPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEPEP  302

Query  115  KP-----VESRPASPF  125
            +P      ES+P S F
Sbjct  303  EPEPQPEPESKPNSLF  318
```

в исходном белке имеется участок, содержащий очень много пролина (P) и глутаминовой кислоты (E)

Данное выравнивание **не свидетельствует о гомологии**, несмотря на хорошее значение E-value (10^{-9})

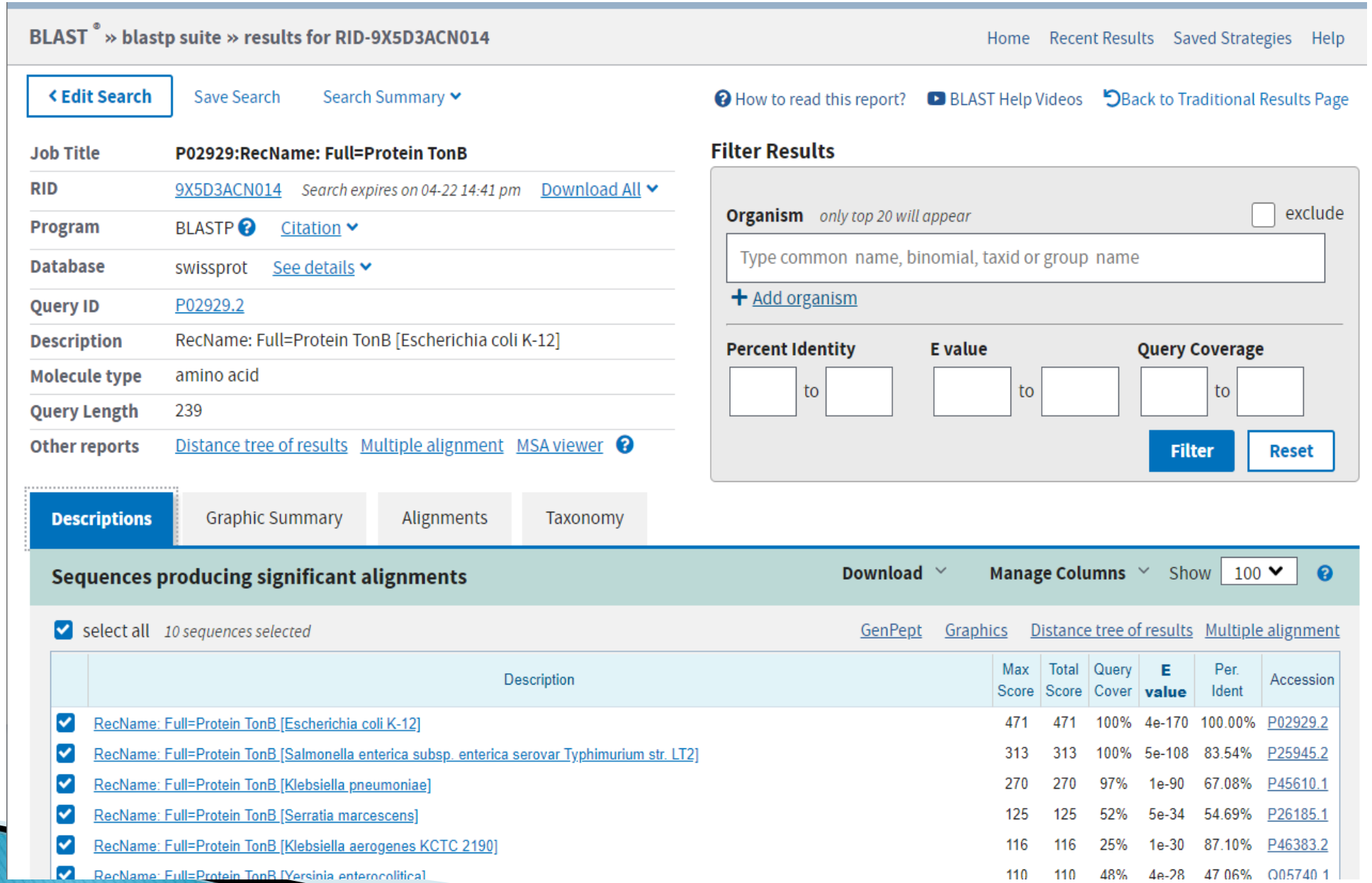
Участок малой сложности

Определяется как участок с смещенным составом (biased composition)

- Гомополимерные участки
- Короткие повторы
- Перепредставленность отдельных остатков

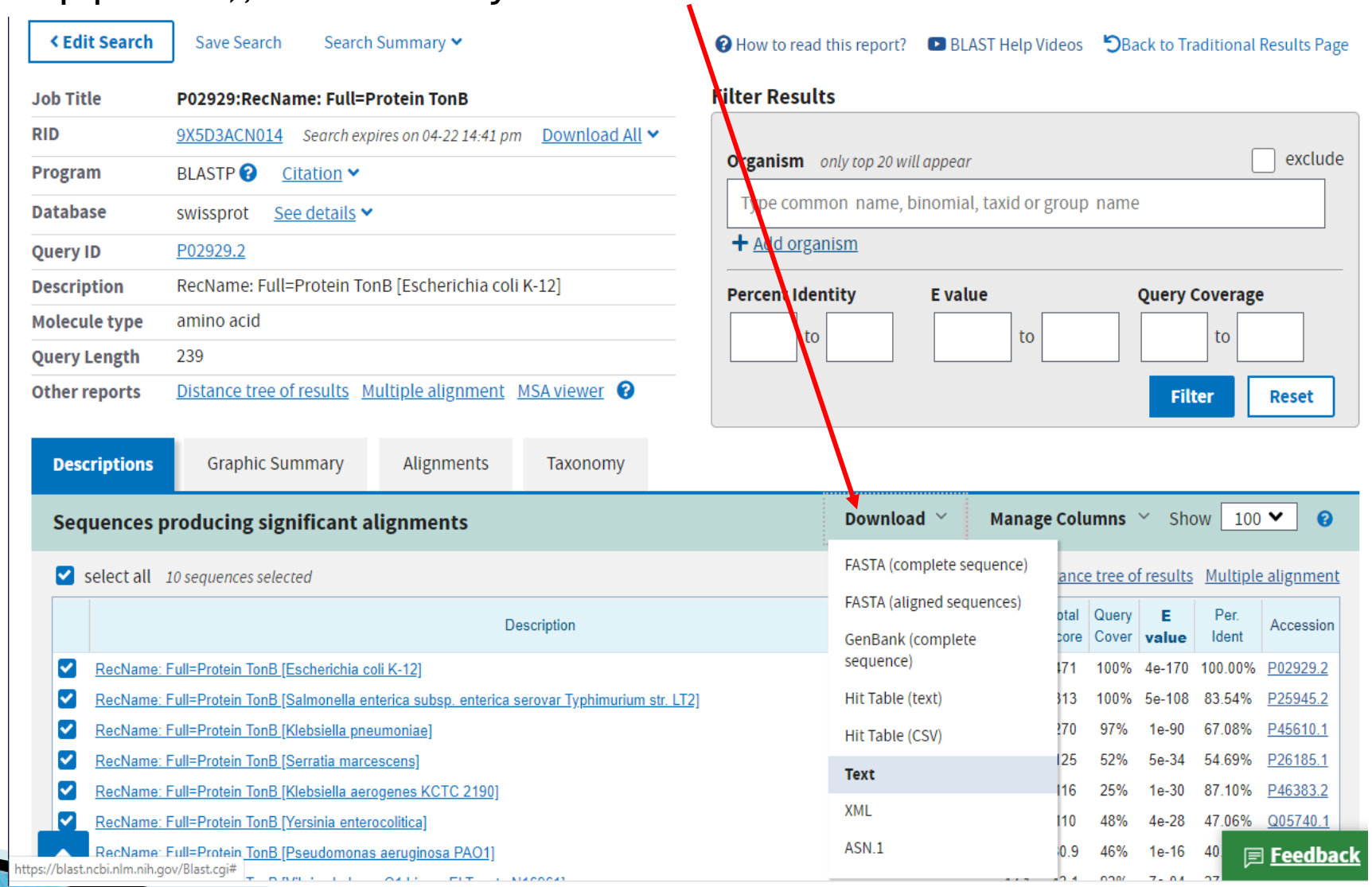
- ✓ Может мешать анализу последовательностей
- ✓ Вычисление E-value (параметры *K* и λ) опирается на средние по всем белкам частоты аминокислотных остатков, поэтому на участках малой сложности оно становится некорректным
- ✓ Обычно ведет к ложным предсказаниям гомологии (false positives)
- ✓ Лучше использовать «Compositional adjustment» (по умолчанию включен)

Выдача BLAST в интерфейсе NCBI



Переход к текстовому виду

Чтобы скачать выдачу самой программы (а не её обработку интерфейсом), можно поступить так:



Текстовая выдача BLAST

```
RID: 9X5D3ACN014
Job Title:P02929:RecName: Full=Protein TonB
Program: BLASTP
Query: RecName: Full=Protein TonB [Escherichia coli K-12] ID: P02929.2(amino acid) Length: 239
Database: swissprot Non-redundant UniProtKB/SwissProt sequences

Sequences producing significant alignments:
                                                                    Max    Total Query    E    Per.
Description                                                         Score  Score cover Value  Ident  Accession
RecName: Full=Protein TonB [Escherichia coli K-12]                  471    471   100%  4e-170 100.00 P02929.2
RecName: Full=Protein TonB [Salmonella enterica subsp. enteric...   313    313   100%  5e-108 83.54  P25945.2
RecName: Full=Protein TonB [Klebsiella pneumoniae]                  270    270   97%   1e-90  67.08  P45610.1
RecName: Full=Protein TonB [Serratia marcescens]                    125    125   52%   5e-34  54.69  P26185.1
RecName: Full=Protein TonB [Klebsiella aerogenes KCTC 2190]         116    116   25%   1e-30  87.10  P46383.2
RecName: Full=Protein TonB [Yersinia enterocolitica]                110    110   48%   4e-28  47.06  Q05740.1
RecName: Full=Protein TonB [Pseudomonas aeruginosa PAO1]            80.9   80.9  46%   1e-16  40.00  Q51368.2
RecName: Full=Protein TonB [Vibrio cholerae O1 biovar El Tor...     43.1   43.1  92%   7e-04  27.50  O52042.2
RecName: Full=Protein TonB [[Haemophilus] ducreyi 35000HP]          33.5   33.5  17%   1.2    36.36  O51810.1
RecName: Full=Translation initiation factor IF-2 [Laribacter...     31.6   31.6  13%   6.8    53.12  C1D8X2.1

Alignments:
>RecName: Full=Protein TonB [Escherichia coli K-12]
Sequence ID: P02929.2 Length: 239
Range 1: 1 to 239

Score:471 bits(1211), Expect:4e-170,
Method:Compositional matrix adjust.,
Identities:239/239(100%), Positives:239/239(100%), Gaps:0/239(0%)

Query  1    MTLDLPRRFPWPTLLSVCIHGAVVAGLLYTSVHQVIELPAPAQPISVTMVTPADLEPPQA  60
            MTLDLPRRFPWPTLLSVCIHGAVVAGLLYTSVHQVIELPAPAQPISVTMVTPADLEPPQA
Sbjct  1    MTLDLPRRFPWPTLLSVCIHGAVVAGLLYTSVHQVIELPAPAQPISVTMVTPADLEPPQA  60

Query  61   VQPPPEPVVEPEPEPEPIPEPPKEAPVVIEKPKPKPKPKPKPVKKVQEQPKRDVKPVESR  120
            VQPPPEPVVEPEPEPEPIPEPPKEAPVVIEKPKPKPKPKPKPVKKVQEQPKRDVKPVESR
Sbjct  61   VQPPPEPVVEPEPEPEPIPEPPKEAPVVIEKPKPKPKPKPKPVKKVQEQPKRDVKPVESR  120

Query  121  PASPFENTAP                                                    180
            PASPFENTAPARLTSSTA                                           180
Sbjct  121  PASPFENTAPARLTSSTATAATSKPVTSVASGPRALSRNQPQYPARAQALRIEGQVKVKF  180
```

Текстовая выдача BLAST

```
RID: 9X6N7P7G016
Job Title:ORF1ab
Program: BLASTP
Query: ORF1ab ID: lcl|Query_23045(amino acid) Length: 7050
Database: swissprot Non-redundant UniProtKB/SwissProt sequences

Sequences producing significant alignments:
```

Description	Max Score	Total Score	Query cover	E Value	Per. Ident	Accession
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	12928	12928	100%	0.0	85.90	P0C6X7.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	12882	12882	100%	0.0	85.76	P0C6W2.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	12867	12867	100%	0.0	85.52	P0C6V9.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	12861	12861	100%	0.0	85.50	P0C6W6.1
RecName: Full=Replicase polyprotein 1a; Short=pp1a; AltName:...	7476	7476	61%	0.0	80.46	P0C6U8.1
RecName: Full=Replicase polyprotein 1a; Short=pp1a; AltName:...	7461	7461	61%	0.0	80.20	P0C6F8.1
RecName: Full=Replicase polyprotein 1a; Short=pp1a; AltName:...	7436	7436	61%	0.0	79.80	P0C6F5.1
RecName: Full=Replicase polyprotein 1a; Short=pp1a; AltName:...	7431	7431	61%	0.0	79.71	P0C6T7.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	6323	6323	95%	0.0	48.41	P0C6W5.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	6135	6135	99%	0.0	45.73	P0C6W4.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5990	6347	99%	0.0	50.39	K9N7C7.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5608	6235	92%	0.0	55.76	P0C6W1.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5599	6237	93%	0.0	55.66	P0C6W3.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5435	5608	93%	0.0	49.37	P0C6W8.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5434	5606	93%	0.0	49.39	P0C6W7.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5418	5554	83%	0.0	48.88	P0C6X8.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5404	5574	93%	0.0	49.26	P0C6X6.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5397	5555	87%	0.0	48.81	P0C6X9.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5395	5565	93%	0.0	49.23	P0C6W9.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5346	5484	93%	0.0	48.37	P0C6Y0.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5305	5483	91%	0.0	48.52	P0C6X3.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5301	5477	91%	0.0	48.52	P0C6X4.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5299	5474	91%	0.0	48.53	P0C6X2.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	5267	5435	87%	0.0	48.86	P0C6X0.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	4319	4397	72%	0.0	46.73	P0C6W0.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	4282	4358	70%	0.0	46.53	P0C6Y4.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	4266	4344	70%	0.0	46.72	P0C6X5.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	4227	4303	73%	0.0	45.46	P0C6X1.1
RecName: Full=Replicase polyprotein 1ab; Short=pp1ab; AltName:...	4179	4252	72%	0.0	45.66	P0C6Y5.1

Словарик (таблица находок BLAST)

Max Score: самый большой из весов (в битах) выравниваний запроса с данной находкой

Total Score: суммарный вес (в битах) всех выравниваний запроса с данной находкой

Query cover: процент длины запроса, покрытого выравниваниями

E Value: в таблице находок это E-value, посчитанное по особой формуле на основе **всех** выравниваний запроса с данной находкой

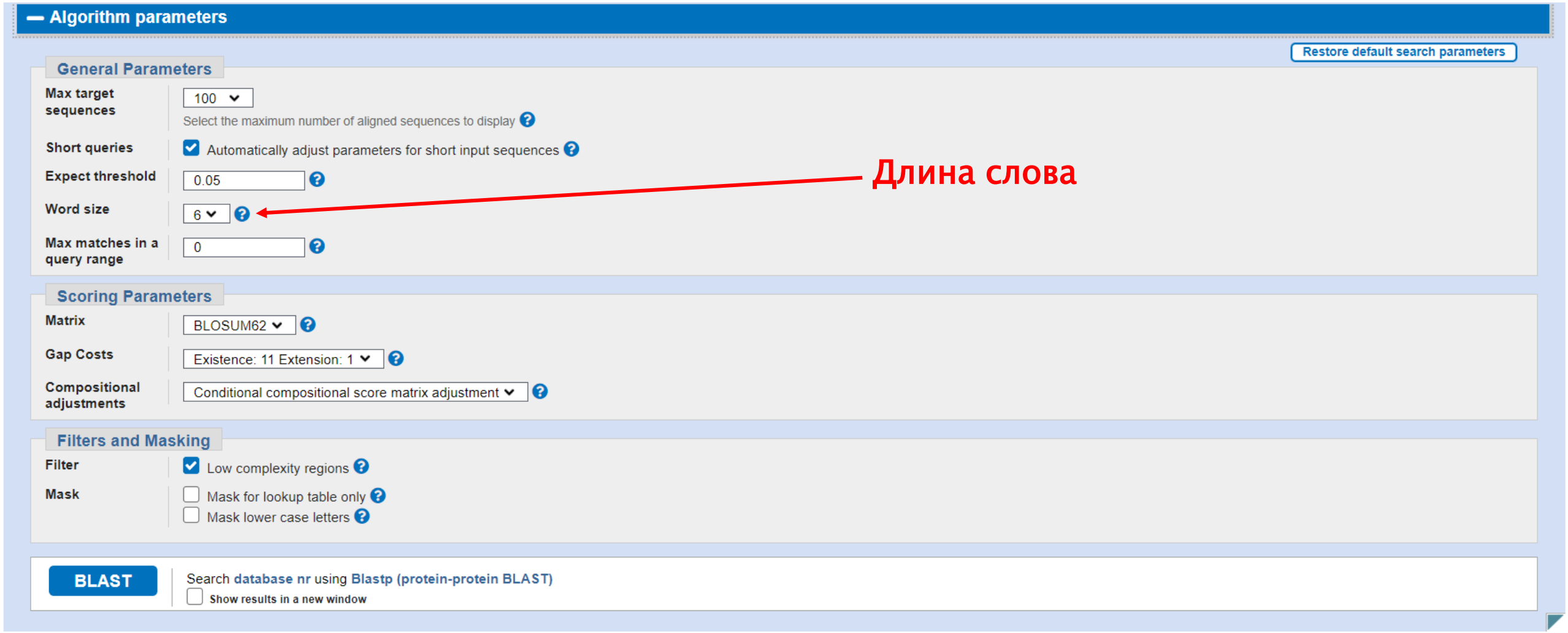
Per. Ident: процент идентичных букв в лучшем (по весу) из выравниваний запроса с данной находкой

BLAST — эвристический алгоритм

Алгоритмы биоинформатики можно разделить на точные и эвристические.
Точные алгоритмы решают какую-либо точно сформулированную формализованную задачу. Пример: алгоритм Нидлмана – Вунша, который для данных последовательностей находит выравнивание с максимальным весом.
Эвристические алгоритмы — те, для которых формальную задачу сформулировать нельзя.

BLAST **не гарантирует** нахождение оптимального локального выравнивания. За счёт этого достигается высокая скорость работы. Но теоретически возможно, что BLAST не найдёт в банке имеющийся там вполне достоверный (судя по выравниванию) гомолог.

Дополнительные параметры



Длина слова

Одним из параметров BLAST является длина слова (word size).

Чем больше длина слова, тем быстрее работает BLAST, но тем меньше его **чувствительность**. Это означает, что вероятность пропустить гомологи возрастает.

Сейчас на сайте NCBI значение длины слова по умолчанию равно 6, доступны значения 2 и 3.

Идея алгоритма BLAST

Нам нужно найти в банке последовательности, хорошо (то есть с большим весом) выравнивающиеся с последовательностью запроса.

Можно было бы это делать алгоритмом Смита – Уотермена, последовательно выравнивая каждую банковскую последовательность с запросом (и такие сервисы существуют, например ssearch на сайте ebi.ac.uk). Но при нынешних объёмах банков это работает слишком медленно.

Идея состоит в том, чтобы заранее **проиндексировать** банк.

Индексы вы видели в конце почти любой научной книги, там имеется **алфавитный** список терминов (или, например, латинских названий растений) с указанием страниц, на которых упоминается этот термин.

В случае BLAST индексами служат **слова** заданной длины из букв, встречающихся в наших последовательнсостях. Например, для белков и при длине слова 3 это AAA, AAC, AAD, …, YYY, всего $20^3 = 8000$ слов.

Перед тем, как запускать собственно поиск, создаётся таблица, в которой для каждого слова указано, в какой последовательности банка и в каком месте это слово встретилось.

Индекс — примерно то же, что алфавитный указатель в книге

АЛФАВИТНЫЙ УКАЗАТЕЛЬ

(цифры обозначают номера экспериментов или параграфов)

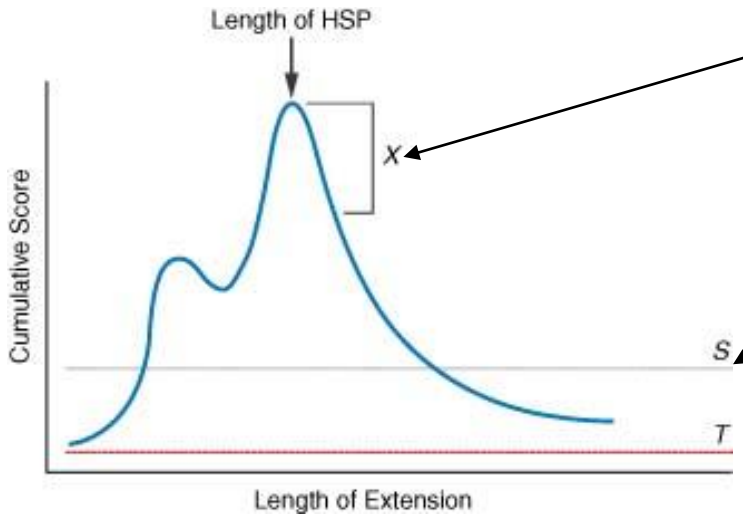
BLAST: отбор слов

- Два параметра:
  - длина слова
    (word_size, ≥2, в standalone по умолчанию 3)
  - порог на сходство слов
    (threshold, ≥0, по умолчанию 11)
- Берутся все слова из запроса (query)
  например, из aacddefg будут взяты (при длине слова 3):
  aac, acd, cdd, dde, def, dfg
- В индексах ищутся слова, имеющие сходство со словами из запроса на уровне не менее threshold

BLAST: от якоря к выравниванию

- Выравнивание начинает строиться, если в запросе есть пара слов на расстоянии, меньшем параметра window_size (по умолчанию 40), для которых нашлась пара сходных слов в одной банковской последовательности на том же расстоянии.
  В результате получаем два якоря — выравнивания длины word_size.
- Второй якорь расширяется без гэпов в обе стороны, пока вес не упадёт на заданную величину от максимально достигнутого (по умолчанию этот параметр xdrop_ungap = 7 бит)
- Если максимально достигнутый вес больше 22 бит, то соответствующее выравнивание расширяется уже с гэпами (аналогично алгоритму Нидлмана – Вунша). Расширение продолжается, пока вес не упадёт ниже максимально достигнутого на величину, большую xdrop_gap, по умолчанию 15 бит

BLAST: расширение якоря



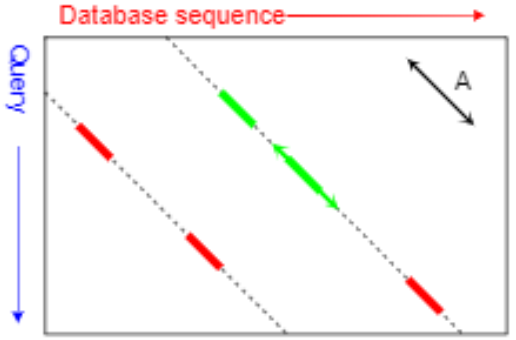
xdrop_ungap, по умолчанию X =7

Минимальный вес хита, по умолчанию S = 22

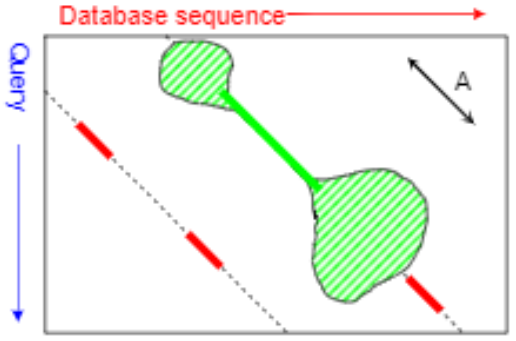
Минимальный вес якоря, по умолчанию T = 11

Это схема расширения в одну сторону; после того, как максимальное значение найдено, точно так же расширяем в другую.

Indexing for Blast (3)



Ungapped extension if:
2 "Hits" are on the same diagonal but
at a distance less than A



Extension using dynamic programming
limited to a restricted region
limited through a score drop-off
threshold

LF, Bezel October 2006

SIB

https://docplayer.net/15013198-Databases-indexation.html
Автор: Laurent Falquet, SIB

BLAST: роль длины слова
(мой эксперимент)

- Вход: последовательность из 466 остатков
- NCBI BLAST (https://blast.ncbi.nlm.nih.gov/)
- Область поиска: Swiss-Prot, белки из бактерий
- Параметры, кроме "Word Size", по умолчанию.
  В частности, порог E-value = 10
- W = 6
  - Найдено 16 последовательностей, в них 18 находок
  - 8 находок с E < 0,001
  - Время работы сервиса NCBI – менее одной минуты
- W = 2
  - Найдено 69 последовательностей, в них 75 находок
  - 12 находок с E < 0,001
  - Время работы сервиса NCBI – около 35 мин

Вопросы и ответы про BLAST

За счёт чего BLAST работает быстро?

За счёт просмотра не всех возможных выравниваний, а только полученных расширением "затравок". Каждая "затравка" получается из слова длины k (k =2, 3, …, 6), встреченного в запросе, и очень сходного слова из какой-либо банковской последовательности.

"Затравки" находятся очень быстро благодаря предварительной индексации всех слов в банке. В результате индексации для каждого слова указано, в каких местах каких банковских последовательностей это слово встечается.

Что может поменяться при изменении параметра "Word size»?

Чем длиннее слово, тем меньше машинного времени займёт поиск.

Чем короче слово, тем чувствительнее поиск (меньше опасность пропустить хорошее выравнивание).

Standalone BLAST

BLAST можно установить на своём компьютере
(а на kodomo он уже установлен)

Предположим, вам нужно найти гомологи белка, чья последовательность — в файле myprot.fasta, в протеоме, содержащемся в файле proteom.fasta (всё в fasta-формате, BLAST других не понимает).

Придётся сначала проиндексировать ваш банк программой makeblastdb, подав ей на вход протеом (читайте makeblastdb -help)

Эта программа создаст несколько файлов, необходимых для поиска, в том числе тот самый индекс якорей (сразу для всех допустимых длин слов)

После этого можно искать программой blastp, указав ей имя файла с запросом и название проиндексированного банка
(читайте blastp -help, нужные опции: -query, -db, -out)

Standalone BLAST

Впрочем, можно использовать BLAST и для обычного локального выравнивания двух последовательностей, безо всякой индексации:

```
blastp -query seq1.fasta -subject seq2.fasta -out result.blastp
```

Но имейте в виду, что BLAST и в таком варианте не гарантирует оптимального выравнивания (это **эвристический** алгоритм)! Зато можно быстро выровнять очень длинные последовательности (команде water может не хватить памяти) и получить не одно, а много локальных выравниваний.

(На самом деле в этом варианте BLAST «на ходу» индексирует вторую последовательность)

BLAST: варианты формата выходного файла

```
-outfmt <String>
  alignment view options:
    0 = Pairwise,
    1 = Query-anchored showing identities,
    2 = Query-anchored no identities,
    3 = Flat query-anchored showing identities,
    4 = Flat query-anchored no identities,
    5 = BLAST XML,
    6 = Tabular,
    7 = Tabular with comment lines,
    8 = Seqalign (Text ASN.1),
    9 = Seqalign (Binary ASN.1),
   10 = Comma-separated values,
   11 = BLAST archive (ASN.1),
   12 = Seqalign (JSON),
   13 = Multiple-file BLAST JSON,
   14 = Multiple-file BLAST XML2,
   15 = Single-file BLAST JSON,
   16 = Single-file BLAST XML2,
   18 = Organism Report
```

0–4 — чтобы смотреть глазами
5–12 — чтобы парсить программами.
6, 7 и 10 можно импортировать в электронные таблицы