Лекция 8.
Языки запросов корпусных менеджеров.
Выходные интерфейсы.

1 Понятие и состав языка запросов

Информационный запрос - это словесное выражение определенной информационной потребности. В теории информационного поиска запрос на языке ИПС еще называют поисковым предписанием. Запросы анализируются по своему предметному и формальному содержанию и описываются в терминах словаря языка запросов прикладной программы, работающей с корпусом. Процедура поиска заключается в поочерёдном сопоставлении поискового образа запроса с отдельными элементами корпуса и вычислении их соответствия. При наличии такого соответствия элементы корпуса текстов считаются релевантными и подлежат выдаче.
В общем виде модель языка запросов включает в себя следующие элементы:

- собственно поисковые элементы (термины, выражающие информационную потребность, и т.п.);
- средства морфологической нормализации текстовых элементов запроса;
- булевы операторы;
- средства линейной грамматики (операторы расстояния, позиционные операторы);
- дополнительные условия поиска:
  - поиск в определенных полях корпуса (например, внутри тэгов);
  - ограничение области поиска (по произведениям определенных авторов, по дате создания документов, их типу и т.п.);
- требование на сортировку (ранжирование) выдаваемых результатов;
- требования к форме представления результатов поиска:
  - вид выдаваемых результатов;
  - количество выдаваемых документов.

Далее будет рассмотрен язык запросов одного из наиболее успешных, на наш взгляд, корпусных менеджеров, **Bonito/Manatee**[1] (http://www.textforge.cz/download.html). На примере этой поисковой системы будет продемонстрировано большинство основных элементов языка запросов к корпусам текстов, а также приведены примеры задания запросов к корпусу.

2 Требования к корпусным менеджерам

Корпус текстов становится мощным инструментом в руках лингвиста лишь посредством специализированных средств. Неотъемлемой частью понятия «корпус текстов» является система управления текстовыми и лингвистическими данными, которую в последнее время чаще всего называют ***корпусным менеджером*** (или корпус-менеджером) (англ. corpus manager) (см. также *конкордансер* – лекция 7). Корпусный менеджер – это специализированная поисковая система, включающая программные средства для поиска данных в корпусе, получения статистической информации и предоставления результатов пользователю в удобной форме.

[1] Bonito – название менеджера, Manatee – вся программная подсистема корпусного обеспечения.

Корпусный менеджер должен:

- строить как KWIC, так и полные конкордансные списки;
- искать не только отдельные слова, но и словосочетания;
- осуществлять поиск по шаблонам (сложные запросы);
- сортировать списки по нескольким критериям, выбираемым пользователем;
- давать возможность отображать найденные словоформы в неограниченном контексте;
- давать статистическую информацию по отдельным элементам корпуса;
- отображать леммы, морфологические характеристики словоформ и метаданные (библиографические, типологические) (зависит от степени размеченности корпуса);
- сохранять и распечатывать результаты;
- работать как с отдельными файлами, так и с корпусами, неограниченными по размеру;
- быстро обрабатывать запросы и выдавать результаты;
- поддерживать различные форматы текстовых данных (txt, doc, rtf, html, xml и др.);
- быть легким (интуитивно понятным) в использовании, как для опытного, так и для начинающего пользователя.

3 Корпусный менеджер Bonito

Поисковая система Bonito (корпусный менеджер) представляет собой программное обеспечение для работы с корпусами текстов. Система Bonito состоит из двух частей: сервера (Bonitosrv) и графического пользовательского интерфейса (GUI - graphical user interface) Bonito, работающего на стороне клиента, созданного Павлом Рыхли (Pavel Rychly) и группой NLPlab (Natural Language Processing Laboratory) на факультете информатики Университета им. Масарика.[2]

Для работы с системой нами будет использоваться корпус английских текстов SUSANNE[**] (http://www.grsampson.net/). Данный корпус был создан в Великобритании в Университете Сассекса. Он включает в себя более 130 тысяч слов Брауновского корпуса американского английского (the Brown Corpus of American English), аннотированного согласно схеме SUSANNE.

3.1 Основные особенности системы Bonito

Язык запросов

- поиск отдельных атрибутов (словоформа, лемма, тэг);
- использование регулярных выражений;
- логические операторы;
- средства задания структуры (границы предложения и др.);
- быстрая обработка сложных запросов;
- шаблоны;

Конкордансные списки

---

[2] Faculty of Informatics, Masaryk University, Brno, Czech Republic.

[**] Аббревиатура "SUSANNE" обозначает "Surface and underlying structural analysis of natural English".

- история запросов пользователя;
- просмотр морфологических характеристик словоформы;
- отображение леммы;

Операции над конкордансом

- сохранение списков в файл;
- печать списков;
- сортировка по ключевым словам, контексту;
- интерактивное неограниченное расширение контекста;
- фильтрация (удаление части построенных конкордансов);
- удаление повторений;

Частотное распределение

- частоты слов и других атрибутов в корпусе, контексте;
- неограниченное число уровней группировки;

Другие особенности

- выбор кодировок;
- создание пользовательских подкорпусов;
- произвольный набор тэгов;
- английская и чешская версии графического интерфейса;
- возможность подключения других языков.

3.2 Запросы

Пользователь может ввести собственно запрос, сформулированный по правилам языка запросов системы, или шаблон (готовый или созданный пользователем) в окно запросов (см. Рис. 1).

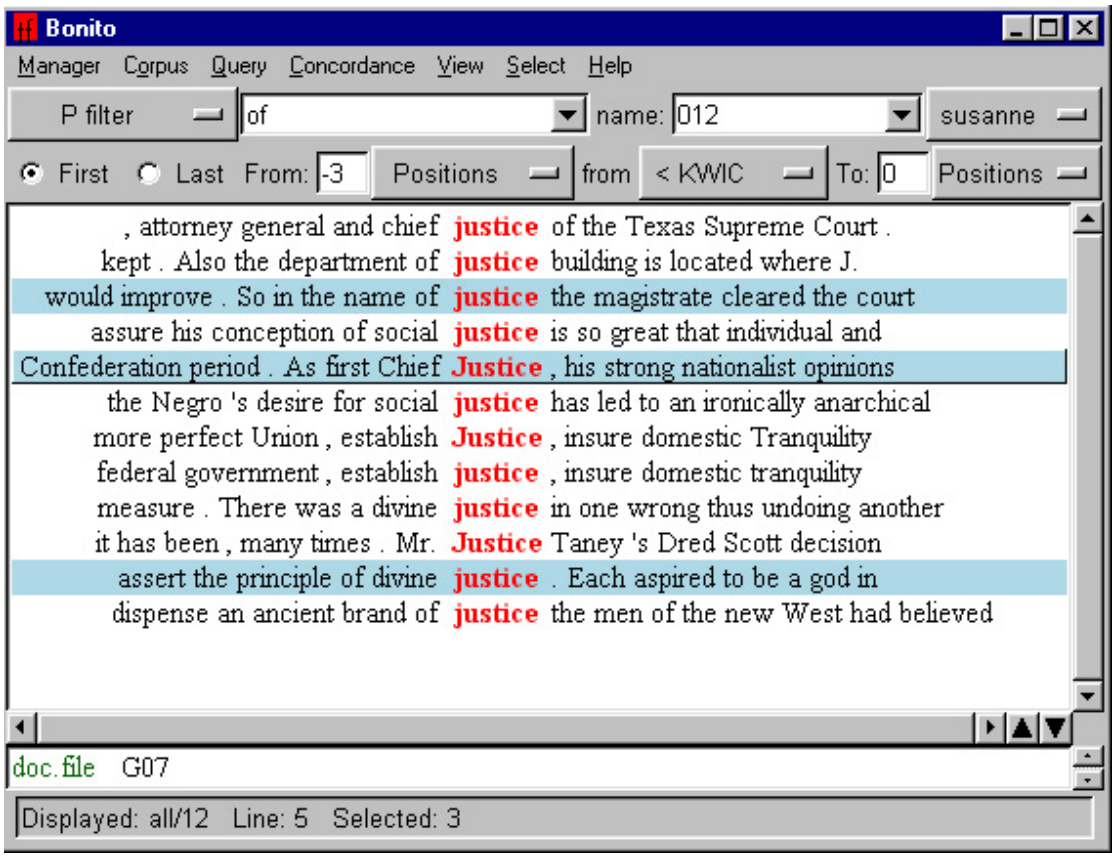


Рис. 1. Окно ИПС **Bonito** с конкордансом
для КС "justice"

Отрицательный фильтр (N-filter) - совпадающие с запросом строки удаляются из конкордансного списка;

Расположение (Collocation) - удовлетворяющие запросу позиции (КС на заданном интервале) в конкордансе выделяются цветом.

Для Положительного, Отрицательного фильтров и Расположения необходимо задавать интервал, в пределах которого следует искать совпадающие позиции для каждой строки конкорданса. Пользователь задает границы интервала (окна ввода "**From:**" и "**To:**"). Если значения положительные, то поиск организуется вправо от исходной позиции, если отрицательные, то влево. Исходной позицией может служить начало КС, конец КС, начало N-ой позиции, конец N-ой позиции. Очень важно отметить, что все введенные запросы сохраняются в так называемой Истории запросов (**Query History**), но если запрос идентичен одному из предыдущих, он не попадает в Историю запросов. Достаточно нажать стрелку "вниз" в окне запроса, чтобы проследить всю Историю, а если необходимо, то вернуться к одному из предыдущих введенных запросов.
Если ввести имя запроса в окне "**name:",** запрос сохраняется в списке "названных" (проименованных) запросов (**named queries**).

3.3 Шаблоны

Шаблон - это вид запроса, который упрощает ввод однотипных запросов. Это означает, что сложный запрос необходимо создать только один раз и сохранить как шаблон, а затем просто вводить значения для данного шаблона.
Например, шаблон для всех словоформ правильного английского глагола "play" мог бы выглядеть так:

```
[word="$1" | word="$1s" | word="$1ed" | word="$1ing"]
```

В этом шаблоне использовалась переменная, состоящая из значка "$" и цифры "1". Количество переменных в шаблоне не ограничено. При использовании шаблона первый вводимый параметр соответствует переменной $1, второй - $2 и т.д. Параметры вводятся через пробел.
Когда шаблон активизируется, он автоматически записывается в окно запроса. Отличие от обычного запроса состоит лишь в следующем: первый знак строки - это восклицательный знак (**!**), далее идет имя шаблона, двоеточие (**:**) и параметры, разделяемые пробелами. Если бы имя приведенного выше шаблона было "regular verb", то строка запроса для всех форм глагола "play" выглядела бы так:

```
!regular verb: play
```

3.4 Примеры запросов

В приведенных ниже примерах мы наглядно продемонстрируем элементы языка запросов корпусного менеджера Bonito.

Пример 1. Поиск конкретной словоформы (КС)

В окно запроса вводится КС "run".
Выдается:

```
    announced that he would not <run> for reelection . Georgia
medical benefits paid out would <run> 1 billion or more in the
    May , said today Jones will <run> well ahead of his GOP opponents
 reports that he had decided to <run> and wanted Mr. Screvane ,
investigation Street car tracks <run> down the center of Pennsylvania
```

Система ищет полное соответствие запрашиваемому слову и выдает результат. Иных словоформ для КС "run" не будет найдено.

Пример 2. Поиск синтагмы

2.1
В окно запроса вводится "run in".
Выдается:

```
contest . The Orioles got a <run in> the first inning when Breeding
   record in the 600 - yard <run in> the Knights of Columbus track
 The Bears added their last <run in> the sixth on Alusik 's double
 for the third Indianapolis <run in> the ninth . Despite the 45
  's first major league home <run in> the fifth put the Sox back
```

Словоформы ищутся в строго заданном (линейном) порядке, как неразрывная синтагма.

2.2
Допустим, мы хотим найти разрывную синтагму "take (smth) out".
В окно запроса вводится "take". Строится конкорданс для данного КС.
Выбирается тип запроса Положительный фильтр (P-filter). В оба окна "From:" и "To:" вводится значение "2", что соответствует второй позиции справа от найденного слова для "оторванной" части синтагмы (у нас "out"). В окно запроса вводим "out".

Выдается:

```
  for governor would force it to <take> petitions  out  into voting
the peasant . Nonetheless , they <take> time  out  -- much time --
Mis-ter McBride . You do that or <take> you  out  a permit right now
```

Разумеется, можно придумать и более сложные варианты подобных запросов с неоднократным применением Положительного фильтра.

Пример 3. Поиск различных форм слова

В окно запроса вводится "runs? in".
Выдается:

```
 tied the game , and single <runs in> the eighth and ninth gave
   record in the 600 - yard <run in> the Knights of Columbus track
   their eight hits for two <runs in> the sixth . Chuck Hinton
 The Bears added their last <run in> the sixth on Alusik 's double
 's first major league home <run in> the fifth put the Sox back
```

В данном запросе используется *управляющий символ*[3] "?", который означает, что предшествующая ему буква "s" может встретиться ноль или один раз. Полученный результат подтверждает это.

Пример 4. Поиск различных форм слова

В окно запроса вводится "run(|s|ning)".
Выдается:

```
    announced that he would not <run> for reelection . Georgia
medical benefits paid out would <run> 1 billion or more in the
    the group are interested in <running> on the required non -
   lawyer and former FBI man is <running> against the Republican
      tied the game , and single <runs> in the eighth and ninth gave
```

Здесь используются группирующие скобки и оператор альтернативы (|) (логическое "или"). То есть, системе дается команда найти КС "run" или "runs" или "running".

Пример 5. Поиск всех форм слова по лемме

В окно запроса вводится "[lemma="be"] within <head>".

---

[3] Более подробно об управляющих символах, как классе управляющих команд языка регулярных выражений, мы поговорим в следующем разделе данной главы.

Выдается:

```
.       <head>DECISIONS <ARE> MADE</head>Asked to elaborate
      <head>LEADERSHIP <IS> HOPEFUL</head>The housing
Nations .<head>FORMULA <IS> DUE THIS WEEK</head>The Advisory
     year .<head>COULD <BE> SCRAMBLE</head>Some predict
    ends .<head>CHOICE <WAS> EXPECTED</head>The selection
     <head>TOBACCO ROAD <IS> DEAD . LONG LIVE TOBACCO
```

Возможность не только искать все словоформы по лемме, но и находить их в заданных полях документа (в данном примере в заголовочном поле, обозначенном тэгом <head>). Соответственно, если мы введем несколько лемм подряд, то получим все варианты таких словосочетаний.

Пример 6. Поиск по морфологическим признакам

В окно запроса вводится "[tag="VVZv"]".
Выдается:

```
 charge of the election , " <deserves> the praise and thanks of the
 However , the jury said it <believes> " these two offices should be
   of Fulton County , which <receives> none of this money " . The
    when the new management <takes> charge Jan. 1 the airport be
  face is a state law which <says> that before making a first
```

Пример демонстрирует замечательную возможность корпусного менеджера искать словоформы по морфологическим признакам. Код "VVZv" означает, что это третье лицо ед.ч. (Zv) значимого глагола (VV). Такая кодировка предложена схемой аннотирования SUSANNE. Следовательно, данная возможность будет успешно использоваться теми, кто знаком с принципами данной схемы аннотирования.

Пример 6. Отображение морфологических признаков и леммы

В пункте командного меню выбирается "**View ⇒ Attributes…**" и отмечаются пункты "**lemma**" и "**tag**".
В окно запроса вводится "[lemma="be"]".
Выдается:

```
manner in which the election <was/be/VBDZ> conducted . The September
     - October term jury had <been/be/VBN> charged by Fulton Superior
stration and election laws " <are/be/VBR> outmoded or inadequate
  " these two offices should <be/be/VB0> combined to achieve greater
Department, the jury said, " <is/be/VBZ> lacking in experienced
```

В конкордансе для каждого вхождения КС показана его исходная форма и ряд морфологических признаков в виде кода.

4 Язык регулярных выражений RegEx

Языки запросов корпусных менеджеров, представленные в той или иной форме (формализованный язык запросов или оконный интерфейс), как правило, базируются на формализме, который получил название язык регулярных выражений. Бóльшую часть запросов на языке RegEx "скрывают" от пользователя в программном коде, реализовав их в виде удобного интерфейса. Пользователю необходимо лишь заполнить определенные поля формы (web-страница с ячейками для заполнения), и его запрос будет осуществлен.

Но все же для задания сложных запросов полезно знать основы языка регулярных выражений.

Наверняка вам приходилось сталкиваться с ситуацией, когда в операционной системе необходимо найти все файлы с заданным расширением. Вы вызываете функцию поиска файлов и в поле "Имя" вводите: `*.jpg` . Тем самым вы сообщаете поисковой машине, что хотите найти файлы, имя которых состоит из любого количества любых символов (`*`), а расширение должно быть "`jpg`". В данном примере вы используете регулярные выражения.

Строковые записи, задающие правила поиска на особом языке, и есть *регулярные выражения*. Мы имеем выражение и какую-либо строку (слово, массив текстов, записи в полях базы данных и т.д.). Операцию проверки, удовлетворяет ли строка выражению, будем называть *сопоставлением* строки и выражения (сравнение). Если какая-то строка или часть строки успешно сопоставилась с выражением, назовем это *совпадением* (соответствием). Например, при сопоставлении выражения "группа букв, окруженная пробелами" и строки "помню чудное мгновенье" совпадением будет строка "чудное" (ведь только она удовлетворяет нашему выражению).

Существует несколько разновидностей языков, используемых для записи регулярных выражений и работы с ними. У них есть много общего, но отдельные части все же отличаются. В популярном языке программирования PHP4 и СУБД MySQL реализован язык регулярных выражений RegEx. Его мы выберем для для изучения.

Перейдем непосредственно к языку RegEx. Вот что он предлагает. Каждое выражение состоит из одной или нескольких управляющих команд. Некоторые из них можно группировать, и тогда они считаются за одну команду. Все управляющие команды разбиваются на три класса:

- *простые символы*, а также *управляющие символы*, играющие роль их заменителей;
- *управляющие конструкции* (квантификаторы повторений, оператор альтернативы, группирующие скобки и т.д.);
- так называемые *мнимые символы* (в строке их нет, но они "помечают" какую-то часть строки - например, ее конец).

4.1 Простые символы

Класс простых символов, действительно, самый простой. А именно, любой символ в строке на языке RegEx обозначает сам себя, если он не является управляющим. К управляющим символам причисляются следующие:

`.*?+[]{}|$^`

Например, регулярное выражение "`abcd`" будет "реагировать" на строки, в которых встретится последовательность "`abcd`".

Группы символов

Одним из самых важных управляющих символов является точка "**.**", обозначающая один любой символ. Например, выражение "`л.к`" имеет совпадение для строк "`лик`", "`лук`", "`лак`". Позже мы рассмотрим, как можно с помощью точки обозначить ровно один (или, к примеру, ровно пять) любых символов.

Возможно, мы захотим искать не любой символ, а один из нескольких указанных. Для этого их нужно заключить в квадратные скобки. К примеру, выражение "`л[иуа]к`" соответствует строкам, в которых есть подстроки из трех символов, начинающиеся с "`л`",

затем одной из букв "и,у,а" и, наконец, "к". Если букв-альтернатив много, и они идут подряд (в алфавитном порядке), то не обязательно перечислять их все. Достаточно указать через дефис первую и последнюю. Например, выражение "[а-я]" обозначает любую букву от "а" до "и", а выражение "[а-я0-9]" задает любой алфавитно-цифровой символ.

Существует и другой, иногда более удобный способ задания больших групп символов. В языке RegEx в квадратных скобках могут встречаться специальные выражения, обозначающие сразу группу символов:

- [:alpha:] - буква;
- [:digit:] - цифра;
- [:alnum:] - буква или цифра;
- [:space:] - пробельный символ;
- [:punct:] - знак пунктуации.

Отрицательные группы

Иногда, когда альтернативных символов много, бывает довольно утомительно перечислять их всех в квадратных скобках. Особенно если нас устраивают все символы, кроме нескольких. В этом случае следует воспользоваться конструкцией "[^]", которая обозначает любой символ, кроме тех, что перечислены после "[^" и до "]". Например, выражение "м[^ао]х" будет соответствовать всем строкам, содержащим буквы "м" и "х", разделенные любым символом, кроме "а" или "и".

4.2 Квантификаторы повторений

Перейдем к рассмотрению так называемых квантификаторов - спецсимволов, использующихся для уточнения действия предшествующих им символов первого класса.

Ноль и более совпадений
Звездочка "*" обозначает, что предыдущий символ может быть повторен ноль или более раз. Например, выражение "19*8" соответствует строке, в которой есть цифра "1", затем ноль или более цифр "9" и, наконец, "8".

Одно и более совпадений
Символ плюса "+" обозначает одно или более совпадений предшествующего символа или группы. Вот пример выражения, который определяет слова, написанные через дефис**:** "[а-я]+-[а-я]+".

Ноль или одно совпадение
Иногда используют еще один квантификатор - знак вопроса "?". Он обозначает, что предыдущий символ может быть повторен ноль или один (но не более!) раз. Например, выражению "Петров[аы]?" будут соответствовать строки "Петров", "Петрова" и "Петровы".

Заданное число совпадений
Рассмотрим последний квантификатор повторения. - фигурные скобки "{}". С его помощью можно реализовать все перечисленные выше возможности. Существует несколько форматов его записи:

- A{n,m} - указывает, что символ "A" может быть повторен *от* n *до* m раз;
- A{n} - символ "A" должен быть повторен *ровно* n раз;
- A{n, } - символ "A" может быть повторен n *или более* раз.

4.3 Мнимые символы

Мнимые символы - это просто участок строки между соседними символами, удовлетворяющий некоторым свойствам. Фактически, мнимый символ - это некая позиция в строке. Например, символ "^" соответствует началу строки, а "$" - ее концу.
Например, выражение "^пере" будет соответствовать любой строке, начинающейся на "пере", выражение "ть$" строке оканчивающейся на "ть", а выражение "^перенять$" точному совпадению со строкой "перенять" (эквивалентно сравнению на равенство двух строк).

4.4 Оператор альтернативы

При описании простых символов мы рассматривали конструкцию "[…]", которая позволяла указывать, что в нужном месте строки должен стоять один из указанных символов. Это ни что иное, как оператор альтернативы, работающий с отдельными символами.
Но в языке RegEx есть возможность задавать альтернативы не одиночных символов, а сразу их групп. Это делается при помощи оператора `"|"`. Вот несколько примеров его работы:

- `"1|2|3"` - полностью эквивалентно выражению `[123]`;
- `"^пре|^пере"` - строки, которые начинаются с "`пре`" или "`пере`";
- `"давать|давал|давала|давало|давали"` - соответствует подстрокам, разделенным символом альтернативы `"|"`.

4.5 Группирующие скобки

В последнем примере подстрока "дава" встретилась в выражении пять раз. Для управления оператором альтернативы существуют группирующие круглые скобки `"()"`. Нетрудно догадаться, что выражение из последнего примера с их помощью можно было записать так: `"дава(ть|л|ла|ло|ли)"`. Скобки могут иметь произвольный уровень вложенности.

5 Выходные интерфейсы

Результаты поиска (выдача) в корпусных менеджерах обычно представлена в виде конкорданса.

"Конкорданс", согласно толковому словарю иностранных слов, это:
"*Алфавитный перечень слов или понятий с указанием их смысла и всех случаев употребления их в данном тексте*. "
В словаре Collins Cobuild English Dictionary мы встретим следующее толкование слова "concordance":
"*An alphabetical list of the words in a book or a set of books which also says where each word can be found and often how it is used*. "

Что же такое конкорданс в корпусной лингвистике? Самый простой способ ответить на этот вопрос - посмотреть на пример, приведенный ниже. Это фрагмент конкорданса для слова "имение" из текста "Дубровский" А.С.Пушкина.

Конкорданс для слова "имение":

вес в губерниях, где находилось его	**имение.**	Соседи рады были угождать малейшим его
грубиян; я хочу взять у него	**имение,**	как ты про то думаешь? - Ваше
, чтобы безо всякого права отнять	**имение.**	Постой однако ж. Это имение принадлежало
отнять имение. Постой однако ж. Это	**имение**	принадлежало некогда нам, было куплено

Теперь посмотрим на тот же конкорданс, но отображаемый в более широком контексте (здесь 12 слов справа и слева от КС):

1 род и связи давали ему большой вес в губерниях, где находилось его **имение.** Соседи рады были угождать малейшим его прихотям; губернские чиновники трепетали при его

2 меня сосед есть, - сказал Троекуров, - мелкопоместный грубиян; я хочу взять у него **имение**, - как ты про то думаешь? - Ваше превосходительство, коли есть какие-нибудь документы или…

3 На то указы. В том-то и сила, чтобы безо всякого права отнять **имение.** Постой однако ж. Это имение принадлежало некогда нам, было куплено у какого-то

4 и сила, чтобы безо всякого права отнять имение. Постой однако ж. Это **имение** принадлежало некогда нам, было куплено у какого-то Спицына и продано потом отцу

И конкорданс KWIC, и более полный контекст могут оказаться полезными, в зависимости от того, что мы хотим делать с данным материалом. Для построения любого конкорданса необходимы две составляющих: текстовая база и процедура. Процедура – это и есть компьютерные программы, которые называются конкордансерами (см. лекция 7) или корпусными менеджерами.

Пример стандартного конкорданса см. выше на рис. 1.

Простой конкордансер может построить конкорданс отдельных слов, словосочетаний, частей слов, знаков пунктуации и т.д. в контекстном окружении. Но более сложные программы (корпусные менеджеры) способны строить полные конкордансы, включающие в себя не только слова, но и другие элементы корпуса. Действительно, существует множество параметров, которые бывает необходимо получить из корпуса. Это лемма и морфологические характеристики слова; позиция слова в предложении и в структуре размеченного текста (HTML, XML); библиографические и типологические признаки документа, из которого выбран контекст (автор, название, источник, год издания, тип текста и т.д.); статистические данные и многое другое.

Еще один режим, характерный для корпусных менеджеров, это выдача информации о так наз. коллокациях – количественных мерах совместной встречаемости.

Пример запроса на коллокации и пример выдачи см. на рис. 2 и 3.

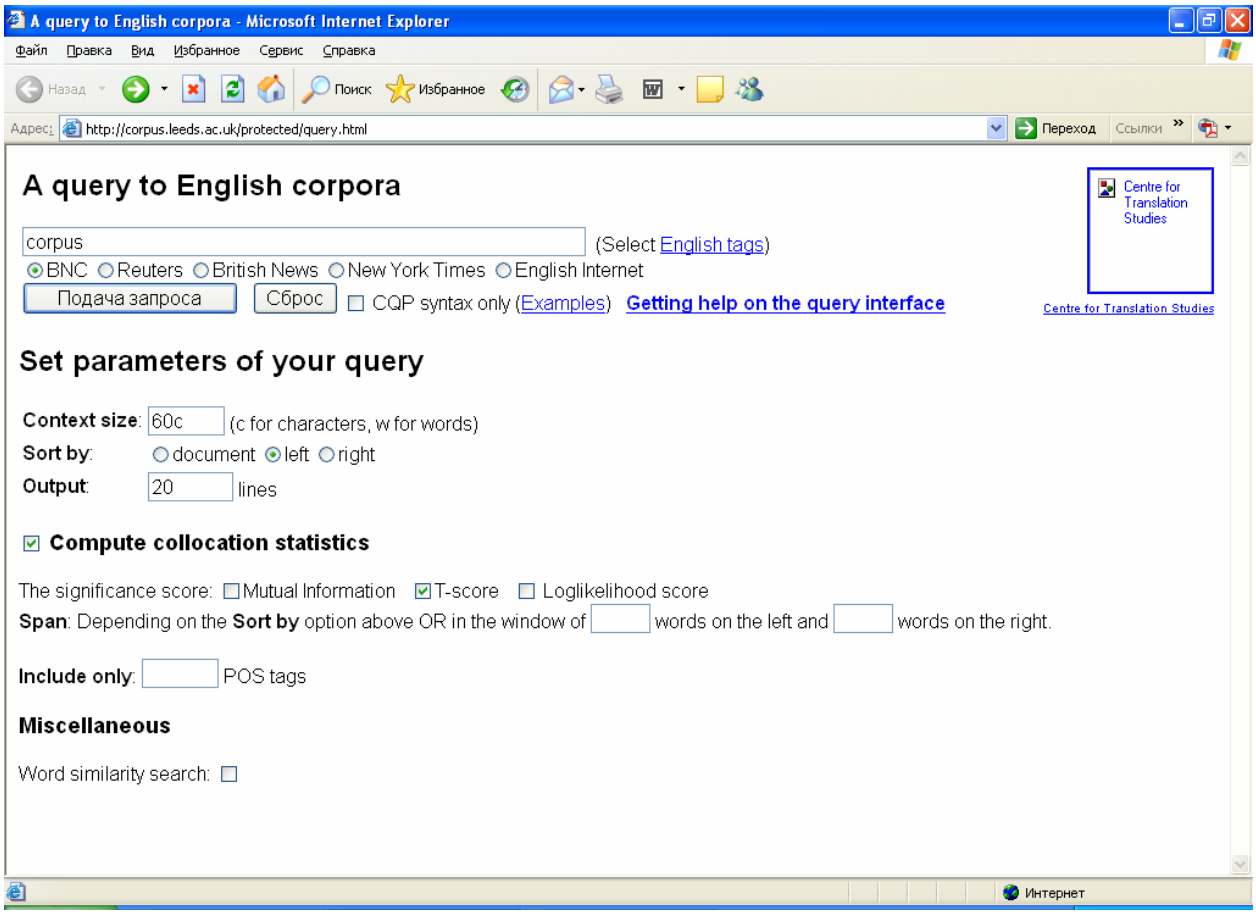
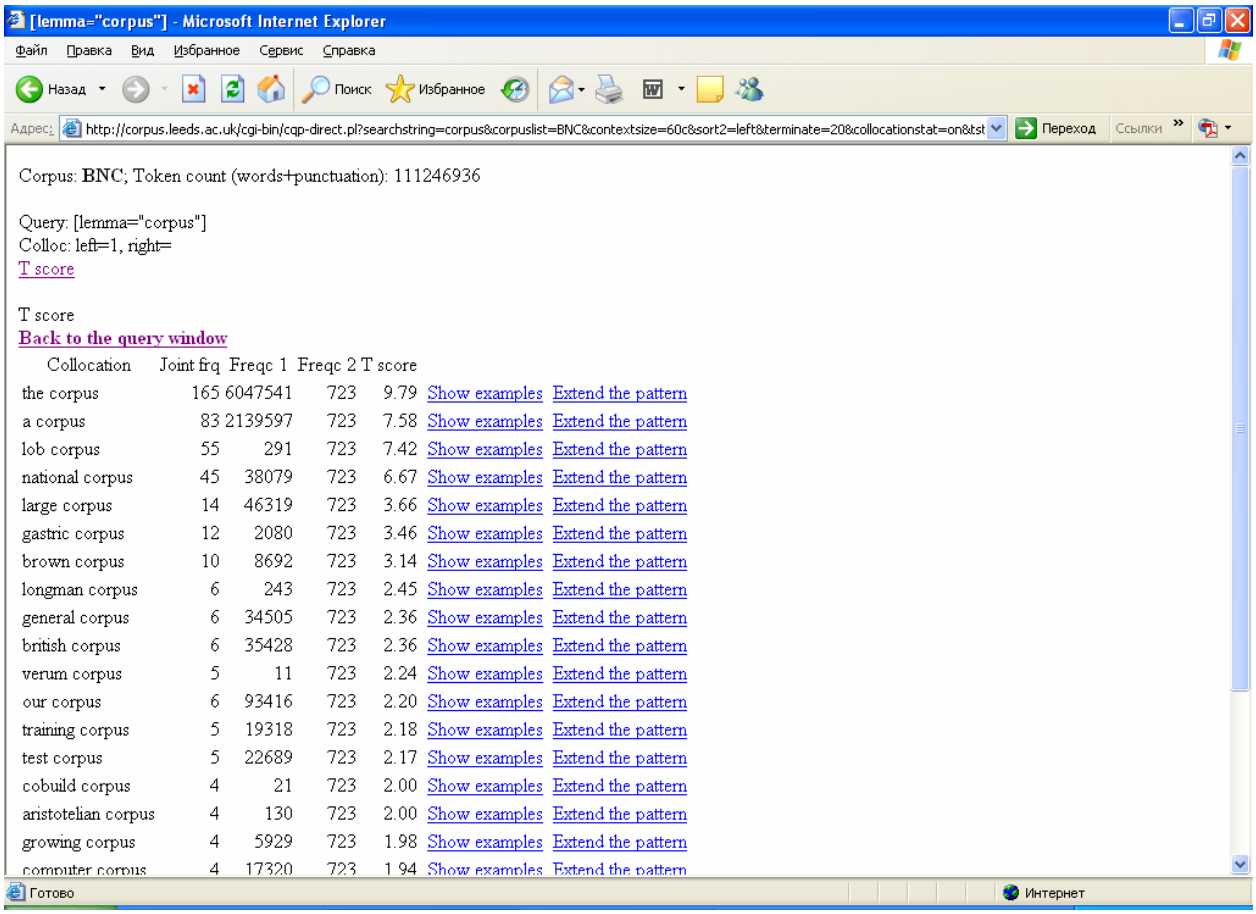


Рис. 2. Запрос на выдачу характеристики T-score для слова ‘corpus’.



[lemma="corpus"] - Microsoft Internet Explorer

Файл Правка Вид Избранное Сервис Справка

Назад Поиск Избранное

Адрес: http://corpus.leeds.ac.uk/cgi-bin/cqp-direct.pl?searchstring=corpus&corpuslist=BNC&contextsize=60c&sort2=left&terminate=20&collocationstat=on&st Переход Ссылки

Corpus: **BNC**; Token count (words+punctuation): 111246936

Query: [lemma="corpus"]
Colloc: left=1, right=
T score

T score
Back to the query window

Collocation	Joint frq	Freqc 1	Freqc 2	T score		
the corpus	165	6047541	723	9.79	Show examples	Extend the pattern
a corpus	83	2139597	723	7.58	Show examples	Extend the pattern
lob corpus	55	291	723	7.42	Show examples	Extend the pattern
national corpus	45	38079	723	6.67	Show examples	Extend the pattern
large corpus	14	46319	723	3.66	Show examples	Extend the pattern
gastric corpus	12	2080	723	3.46	Show examples	Extend the pattern
brown corpus	10	8692	723	3.14	Show examples	Extend the pattern
longman corpus	6	243	723	2.45	Show examples	Extend the pattern
general corpus	6	34505	723	2.36	Show examples	Extend the pattern
british corpus	6	35428	723	2.36	Show examples	Extend the pattern
verum corpus	5	11	723	2.24	Show examples	Extend the pattern
our corpus	6	93416	723	2.20	Show examples	Extend the pattern
training corpus	5	19318	723	2.18	Show examples	Extend the pattern
test corpus	5	22689	723	2.17	Show examples	Extend the pattern
cobuild corpus	4	21	723	2.00	Show examples	Extend the pattern
aristotelian corpus	4	130	723	2.00	Show examples	Extend the pattern
growing corpus	4	5929	723	1.98	Show examples	Extend the pattern
computer corpus	4	17320	723	1.94	Show examples	Extend the pattern

Готово Интернет

Рис. 3. Характеристика T-score для слова ‘corpus’.

Заключение

Более подробно с языками запросов корпусных менеджеров и выходными интерфейсами мы ознакомимся в ходе лабораторной работы.