

# The Computer and Natural Language (Ling 445/515) Topic 2: Searching

Markus Dickinson  
Dept. of Linguistics, Indiana  
Autumn 2008

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

1 / 30

## Searching

- ▶ A breathtaking number of information resources are available: books, databases, the web, newspapers, . . .
- ▶ To locate relevant information, we need to be able to search these resources, which often are **written texts**:
  - ▶ Searching in a library catalogue (e.g., using the IUCAT part of OneSearch)
  - ▶ Searching the web (e.g., using Google)
  - ▶ Advanced searching in text corpora (e.g., using regular expressions in Opus)

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

2 / 30

## Searching in speech

- ▶ One might also want to search for **speech**, e.g., to find a particular sentence spoken in an interview one only has a recording (audio file) of.
- ▶ With current technology, this is only possible if the interview is transcribed, using the IPA or another writing system.
- ▶ It is, however, already possible to
  - ▶ detect the language of a spoken conversation, e.g., when listening in to a telephone conversation
  - ▶ detect a new topic being started in a conversation
- ▶ In the following, we focus on searching in text.

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

3 / 30

## Searching in a library catalogue

- ▶ To find articles, books, and other library holdings, a library generally provides a **database** containing information on its holdings.
- ▶ IUCAT is one of the **database frontends** providing access to the library databases at Indiana.
- ▶ This makes it possible to search for the occurrence of **literal strings** occurring in the author, title, keywords, call number, etc. associated with an item held by the library.

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

4 / 30

## Basic searching with IUCAT

- ▶ Literal strings are composed of characters which naturally must be in the same character encoding system (e.g. ASCII, ISO8859-1, UTF-8) as the strings encoded in the database.
- ▶ For literal strings, the search engine does not distinguish between upper and lower-case letters (i.e. they aren't so literal after all ;-)
- ▶ Adjacent words are searched as a phrase.
  - ▶ art therapy
  - ▶ vitamin c
- ▶ In addition to **querying** literal strings, the **query language** also supports the use of
  - ▶ **special characters** to abbreviate multiple options
  - ▶ special **operators** for combining two query strings (boolean operators) or modifying the meaning of a single string (unary operators)

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

5 / 30

## Special characters and operators

Use \$ to truncate the characters at the end of a word

- ▶ ? can be used to replace a single character

The use of *and*, *or*, and the parentheses provides you with basic **boolean expression** operations

- ▶ Use *and* or *or* (or *xor*) to specify multiple words in any field, any order.
  - ▶ art and therapy
  - ▶ art or therapy
  - ▶ art xor therapy (excludes "art therapy")
- ▶ Use *not* to exclude words.
  - ▶ art not therapy
- ▶ Use parentheses to group words together when using more than one operator.
  - ▶ art therapy not ((music or dance) therapy)

See <http://www.libraries.iub.edu/index.php?pageId=1480>

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

6 / 30

# Positional Operators

- ▶ **adj**: terms must be next/adjacent to each other, in the same order as entered
  - ▶ art adj therapy
  - ▶ art adj5 therapy: within 5 words
- ▶ **near**: terms must be next to each other, in either order
  - ▶ art near therapy
  - ▶ art near5 therapy: within 5 words

NB: stop words are ignored in searches, unless enclosed in double quotes (*a, an, as, at, be, but, by, do, for, if, in, is, it, of, on, the, to*)

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

7 / 30

# Searching the web

## A computer user

- ▶ wants to find something on "the web", i.e., in files accessible via the hypertext transfer protocol (http) protocol on the internet
- ▶ goes to a **search engine** = program that matches documents to a user's search requests
- ▶ enters a **query** = request for information
- ▶ gets a list of websites that might be relevant to the query
- ▶ **evaluates the results**: either picks a website with the information looked for or reformulates the query

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

8 / 30

# The nature of the web

- ▶ Web pages are generally less structured than a record in a library database (with title, author, subject, and other fields).
- ▶ One generally searches for words found anywhere in the document.
- ▶ It is, however, possible to include **meta data** in a web page.
- ▶ Meta data is additional, structured information that is not shown in the web page itself: e.g., the language a web page is in, its character encoding, author, keywords, etc.
- ▶ Example for a **meta tag**: `<META name="keywords" lang="en-us" content="vacation, Greece">`

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

9 / 30

# Search engines

- ▶ Search engines (e.g., Google)
  - ▶ store a copy of all web pages
  - ▶ create an **index** to provide efficient access to this large number of pages (e.g., Google currently searches over 8 billion pages)
  - ▶ compute a rank for each web page to be able to rank the query results
- ▶ Some ways in which search engines can differ:
  - ▶ Treatment of word tokens:
    - ▶ **stemming**: treat *bird* and *birds* as the same or not
    - ▶ **capitalization**: treat *trip* and *Trip* the same or not
  - ▶ Options for searching: use of **operators** or special interface for advanced searching
  - ▶ how search results are **ranked** and potentially **clustered** (group similar results)

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

10 / 30

# Google: Operators (I)

- ▶ **+**: Require a word to occur in the result  
e.g., To find a restaurant that serves both tofu and BBQ one could try
  - ▶ +tofu +BBQ
- ▶ **-**: Disallow a word from occurring in the result  
e.g., As a *potatos* purist :-), I search for
  - ▶ potatos -potatoes
- ▶ **~**: Include synonyms of the word
- ▶ Quotation Marks (phrases)  
e.g., looking for sites on *What Cheer, Iowa* with
  - ▶ "What Cheer"
- ▶ **\*** match anything (wildcard), e.g., "who's that \*\*"

See: <http://www.google.com/help/refinesearch.html>

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

11 / 30

# Google: Operators (II)

- ▶ **intitle**: Find words used in a title
  - ▶ e.g., `intitle:Hoosier` finds only web pages which has this word in the title
- ▶ **inurl**: Find words used in the url
  - ▶ e.g., `inurl:ling` returns more linguistics webpages than `ling` does
- ▶ **link**: Find pages that link to a certain page
  - ▶ e.g., `link:www.indiana.edu` to show pages linking to the main Indiana web page
- ▶ **site**: Find pages that are part of a single domain
  - ▶ e.g., I want to find strange attractions involving fish. Knowing one site which has such stuff, one can try `fish site:www.roadsideamerica.com`.

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

12 / 30

## Google: Advanced searching

More elaborate **web forms** are provided as alternative to using operators:

- ▶ **match all**: matches all terms in your query
- ▶ **match any**: matches as many terms in your query as it can find
  - ▶ most search engines return *match all* followed by *match any* results
- ▶ **exclude**: eliminate documents which contain certain words

You can also specify file format, date updated, etc.

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech  
Searching in a Library Catalogue  
Operators  
Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results  
Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

13 / 30

## Improving searching (I)

How can I make my searches better?

- ▶ Be on the watch for **ambiguity** = one word has multiple meanings
  - ▶ e.g., *bed*: flower bed, sleeping bed, truck bed
- ▶ Use **synonyms** and other related words
  - ▶ e.g., *plant*: building, complex, works, power (distinguish from flora)
- ▶ Be aware of **stop words** = words that some search engines ignore because they are “uninformative,” such as *the, of,* and so on

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech  
Searching in a Library Catalogue  
Operators  
Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results  
Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

14 / 30

## Improving searches (II)

- ▶ Exclude problematic words
  - ▶ e.g., “jefferson airplane -starship” (if you don’t want info on the Starship years)
- ▶ Be aware of **parts of speech** and what other guises they come in.
  - ▶ e.g., *plant*: planting, planter, planted (distinguish from *power plant*)
- ▶ Continually narrow your focus (using the feedback)
  - ▶ e.g., Want to find information on the game *Hearts*
  - ▶ *hearts* is too vague, but looking over results indicates *cards* or *tricks* could help

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech  
Searching in a Library Catalogue  
Operators  
Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results  
Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

15 / 30

## Ranking of results

- ▶ Ideally, the webpages matching a query are returned as an ordered list based on a page’s **relevance**.
- ▶ How can a search engine, which does not understand language, determine the relevance of a particular page?

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech  
Searching in a Library Catalogue  
Operators  
Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results  
Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

16 / 30

## Information used to rank results

- ▶ Counting the number of links to and from a page, to determine how popular a page is.
  - ▶ As a result, unpopular or new pages require a more specific query to be found.
- ▶ Keeping track of the nature of links to a page; linked pages might be thematically related.
  - ▶ e.g., Even if I never mention Sinclair Lewis on a page describing his book *Babbitt*, it can be identified if many Sinclair Lewis sites link to my page.
- ▶ bonuses/penalties for known sites of high/low quality
- ▶ looking for **keywords in metadata**
- ▶ counting how often a web result was clicked on by a user (**click-through measurement**)
- ▶ various secret ingredients

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech  
Searching in a Library Catalogue  
Operators  
Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results  
Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

17 / 30

## Evaluating search results

What measures can one use to evaluate how successful a query is?

- ▶ **precision**: How many of the pages returned are the ones we want?
  - ▶ e.g., Google gives me 400 hits for a query, 200 of which are related to the topic I want; precision = 50%.
- ▶ **recall**: How many pages on the topic we wanted were actually given? (hard to calculate for web searching)
  - ▶ e.g., Google gave me 200 pages I wanted, but there were actually 1000 pages on that topic out there somewhere on the internet; recall = 20%.

We saw earlier how to use our initial results to refine our query and improve precision

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech  
Searching in a Library Catalogue  
Operators  
Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results  
Advanced searches with regular expressions  
Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

18 / 30

# Motivating regular expressions

If one wants to be able to describe more complex patterns of words and text, sometimes boolean expressions aren't enough:

- ▶ In a large document I want to find addresses with a zip code starting with 911 (around Pasadena, CA); but clearly we would not want to report back all occurrences of emergency phone numbers in the document.
- ▶ I want to find all Indiana email addresses which occur in a long text.

Anything where you have to match a complex pattern so-called **regular expressions** are useful.

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions

Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

19 / 30

# Regular expressions: What they are

- ▶ A regular expression is a compact description of a set of strings, i.e., a language (in **formal language theory**).
- ▶ They can be used to search for occurrences of these strings
- ▶ Regular expressions can only describe so-called **regular languages**.
- ▶ This means that some patterns cannot be specified using regular expressions, e.g., finding a string containing matching left and right parentheses.
- ▶ Note that just like any other formalism, regular expressions as such have no linguistic contents, but they can be used to refer to strings encoding a **natural language** text.

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions

Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

20 / 30

# Regular expressions: Tools that use them

- ▶ A variety of unix tools (grep, sed, ...), editors (emacs, jEdit, ...), and programming languages (perl, python, Java, ...) incorporate regular expressions.
- ▶ Implementations are very efficient so that large text files can be searched quickly; but not efficient enough for web searching → no web search engine offers them (yet).
- ▶ The various tools and languages differ w.r.t. the exact syntax of the regular expressions they allow.

# The syntax of regular expressions (I)

Regular expressions consist of

- ▶ strings of literal characters: `c`, `A100`, `natural language`, `30 years!`
- ▶ disjunction:
  - ▶ ordinary disjunction: `devoured|ate, famil(y|ies)`
  - ▶ character classes: `[Tt]he, bec[oa]me`
  - ▶ ranges: `[A-Z]` (any capital letter)
- ▶ negation:
  - `[^a]` (any symbol but a)
  - `[^A-Z0-9]` (not an uppercase letter or number)

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions

Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

21 / 30

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions

Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

22 / 30

# The syntax of regular expressions (II)

- ▶ counters
  - ▶ optionality: `colou?r`
  - ▶ any number of occurrences: `*` (Kleene star) `[0-9]* years`
  - ▶ at least one occurrence: `+` `[0-9]+ dollars`
- ▶ wildcard for any character: `.`  
`beg.n` for any character in between `beg` and `n`

# The syntax of regular expressions (III)

- ▶ Escaped characters: to specify a character with a special meaning (`*`, `+`, `?`, `(`, `)`, `|`, `[`, `]`) it is preceded by a backslash (`\`)  
e.g., a period is expressed as `\.`
- ▶ Operator precedence, from highest to lowest:
  - parentheses `()`
  - counters `*` `+` `?`
  - character sequences
  - disjunction `|`

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions

Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

23 / 30

Computers and Language  
Topic 2: Searching

Introduction  
Text  
Speech

Searching in a Library Catalogue  
Operators

Searching the web  
Operators  
Improving searching  
Ranking of results  
Evaluating search results

Advanced searches with regular expressions

Syntax of regular expressions  
Grep: An example for using regular expressions  
Text corpora and searching them

24 / 30

<h2>Grep</h2> <ul style="list-style-type: none"> <li>▶ <code>grep</code> is a powerful and efficient program for searching in text files using regular expressions.</li> <li>▶ It is standard on Unix, Linux, and Mac OSX, and there also are various ports to Windows (e.g., <a href="http://gnuwin32.sourceforge.net/packages/grep.htm">http://gnuwin32.sourceforge.net/packages/grep.htm</a>, <a href="http://www.interlog.com/~tcharron/grep.html">http://www.interlog.com/~tcharron/grep.html</a> or <a href="http://www.wingrep.com/">http://www.wingrep.com/</a>).</li> <li>▶ The version of <code>grep</code> that supports the full set of operators mentioned above is generally called <code>egrep</code> (for extended <code>grep</code>).</li> </ul>	<p>Computers and Language Topic 2: Searching</p> <p>Introduction Text Speech</p> <p>Searching in a Library Catalogue Operators</p> <p>Searching the web Operators Improving searching Ranking of results Evaluating search results</p> <p>Advanced searches with regular expressions</p> <p>Syntax of regular expressions Grep: An example for using regular expressions Text corpora and searching them</p> <p>25 / 30</p>	<h2>Grep: Examples for using regular expressions (I)</h2> <p>In the following, we assume a text file <code>f.txt</code> containing, among others, the strings that we mention as matching.</p> <ul style="list-style-type: none"> <li>▶ Strings of literal characters: <code>egrep 'and' f.txt</code> matches <u>and</u>, <u>Ayn Rand</u>, <u>Candy</u> and so on</li> <li>▶ Character classes: <code>egrep 'the year [0-9][0-9][0-9][0-9]' f.txt</code> matches the year 1776, the year 1812, the year 2001, and so on</li> <li>▶ Escaped characters: <code>egrep 'why\?'</code> <code>f.txt</code> matches <code>why?</code>, whereas <code>egrep 'why?'</code> <code>f.txt</code> matches <code>why</code> and <code>wh</code></li> </ul>	<p>Computers and Language Topic 2: Searching</p> <p>Introduction Text Speech</p> <p>Searching in a Library Catalogue Operators</p> <p>Searching the web Operators Improving searching Ranking of results Evaluating search results</p> <p>Advanced searches with regular expressions</p> <p>Syntax of regular expressions Grep: An example for using regular expressions Text corpora and searching them</p> <p>26 / 30</p>
<h2>Grep: Examples for using regular expressions (II)</h2> <ul style="list-style-type: none"> <li>▶ disjunction (<code> </code>): <code>egrep 'couch sofa' f.txt</code> matches <code>couch</code> or <code>sofa</code></li> <li>▶ grouping with parentheses: <code>egrep 'un(interest excit)ing' f.txt</code> matches <code>uninteresting</code> or <code>unexciting</code>.</li> <li>▶ Any character (<code>.</code>): <code>egrep 'o.e' f.txt</code> matches <code>ore</code>, <code>one</code>, <code>ole</code></li> </ul>	<p>Computers and Language Topic 2: Searching</p> <p>Introduction Text Speech</p> <p>Searching in a Library Catalogue Operators</p> <p>Searching the web Operators Improving searching Ranking of results Evaluating search results</p> <p>Advanced searches with regular expressions</p> <p>Syntax of regular expressions Grep: An example for using regular expressions Text corpora and searching them</p> <p>27 / 30</p>	<h2>Grep: Examples for using regular expressions (III)</h2> <ul style="list-style-type: none"> <li>▶ Kleene star (<code>*</code>): <code>egrep 'a*rg'h' f.txt</code> matches <code>argh</code>, <code>aargh</code>, <code>aaargh</code> <code>egrep 'sha(la)*' f.txt</code> matches <code>sha</code>, <code>shala</code>, <code>shalala</code>, or if you're Van Morrison <code>shalalalalalalalala</code></li> <li>▶ One or more (<code>+</code>): <code>egrep 'john+y' f.txt</code> matches <code>johny</code>, <code>johnny</code>, ..., but not <code>johy</code></li> <li>▶ Optionality (<code>?</code>): <code>egrep 'joh?n' f.txt</code> matches <code>jon</code> and <code>john</code></li> </ul>	<p>Computers and Language Topic 2: Searching</p> <p>Introduction Text Speech</p> <p>Searching in a Library Catalogue Operators</p> <p>Searching the web Operators Improving searching Ranking of results Evaluating search results</p> <p>Advanced searches with regular expressions</p> <p>Syntax of regular expressions Grep: An example for using regular expressions Text corpora and searching them</p> <p>28 / 30</p>
<h2>Corpora</h2> <ul style="list-style-type: none"> <li>▶ A <b>corpus</b> is a collection of text.</li> <li>▶ Corpora with the works of various writers, newspaper texts, etc. have been collected and electronically encoded.</li> <li>▶ Corpora can be quite large</li> <li>▶ The British National Corpus is a 100 million word collection representing a wide cross-section of current written and spoken British English.</li> <li>▶ Another example is the European Parliament Proceedings Parallel Corpus 1996–2003.</li> </ul>	<p>Computers and Language Topic 2: Searching</p> <p>Introduction Text Speech</p> <p>Searching in a Library Catalogue Operators</p> <p>Searching the web Operators Improving searching Ranking of results Evaluating search results</p> <p>Advanced searches with regular expressions</p> <p>Syntax of regular expressions Grep: An example for using regular expressions Text corpora and searching them</p> <p>29 / 30</p>	<h2>How corpora can be searched</h2> <ul style="list-style-type: none"> <li>▶ Both the BNC and the European Parliament corpus can be searched using on-line web-forms.</li> <li>▶ Both of the web forms allow <b>regular expressions</b> for advanced searching.</li> <li>▶ To provide efficient searching in large corpora, in these search engines regular expressions over characters are limited to single tokens (i.e. generally words).</li> <li>▶ BNC: <ul style="list-style-type: none"> <li>▶ web form: <a href="http://sara.natcorp.ox.ac.uk/lookup.html">http://sara.natcorp.ox.ac.uk/lookup.html</a></li> <li>▶ regular expressions are enclosed in <code>{ }</code></li> </ul> </li> <li>▶ European Parliament Corpus: <ul style="list-style-type: none"> <li>▶ web form: <a href="http://logos.uio.no/cgi-bin/opus/opuscqp.pl?corpus=EUROPARL;lang=en">http://logos.uio.no/cgi-bin/opus/opuscqp.pl?corpus=EUROPARL;lang=en</a></li> <li>▶ in the simplest case, regular expressions are enclosed in <code>" "</code></li> </ul> </li> </ul>	<p>Computers and Language Topic 2: Searching</p> <p>Introduction Text Speech</p> <p>Searching in a Library Catalogue Operators</p> <p>Searching the web Operators Improving searching Ranking of results Evaluating search results</p> <p>Advanced searches with regular expressions</p> <p>Syntax of regular expressions Grep: An example for using regular expressions Text corpora and searching them</p> <p>30 / 30</p>