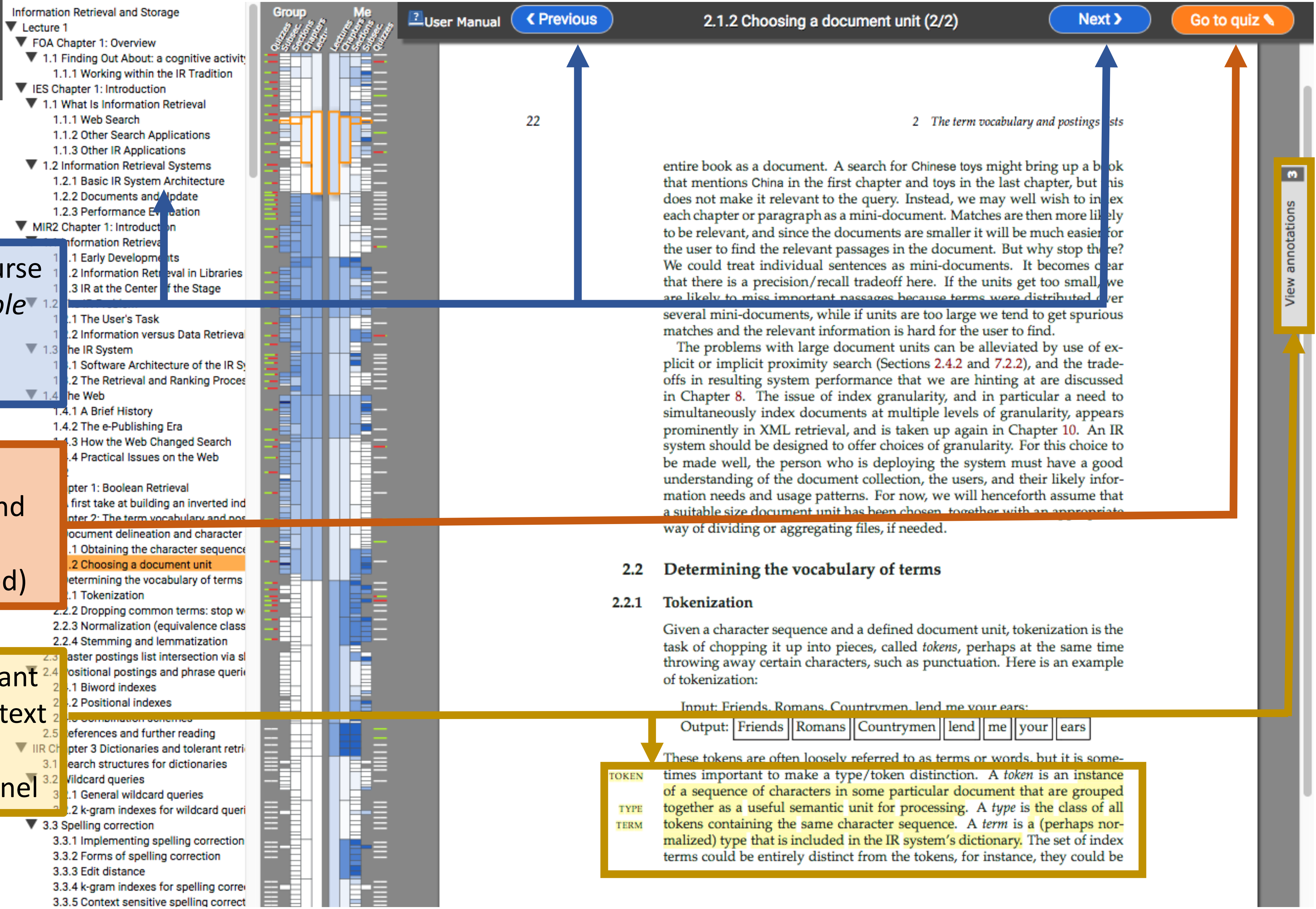
1. GENERAL FEATURES



2.1.2 Choosing a document unit (2/2)

Information Retrieval and Storage
▼ Lecture 1
▼ FOA Chapter 1: Overview
▼ 1.1 Finding Out About: a cognitive activit
1.1.1 Working within the IR Tradition
▼ IES Chapter 1: Introduction
▼ 1.1 What Is Information Retrieval
1.1.1 Web Search
1.1.2 Other Search Applications
1.1.3 Other IR Applications
▼ 1.2 Information Retrieval Systems
1.2.1 Basic IR System Architecture
1.2.2 Documents and Update
1.2.3 Performance Evaluation
▼ MIR2 Chapter 1: Introduction
nformation Retrieva
1.1 Early Developments
1.2 Information Retrieval in Libraries
1.3 IR at the Center of the Stage
▼ 1.2
1.1 The User's Task
1.2 Information versus Data Retrieval
▼ 1.3 he IR System
1.1 Software Architecture of the IR Sy
1.2 The Retrieval and Ranking Proces
▼ 1.4 he Web
1.4.1 A Brief History
1.4.2 The e-Publishing Era
1.4.3 How the Web Changed Search
.4 Practical Issues on the Web
pter 1: Boolean Retrieval
first take at building an inverted ind
pter 2: The term vocabulary and pos
ocument delineation and character
.1 Obtaining the character sequence
.2 Choosing a document unit
etermining the vocabulary of terms
.1 Tokenization
2.2.2 Dropping common terms: stop w
2.2.3 Normalization (equivalence class
2.2.4 Stemming and lemmatization
2.3 aster postings list intersection via sl
2.4 ositional postings and phrase queri
2 .1 Biword indexes
2 .2 Positional indexes
2.5 eferences and further reading
▼ IIR Ch pter 3 Dictionaries and tolerant retri
3.1 earch structures for dictionaries
3.2 ildcard queries
3 .1 General wildcard queries
3 .2 k-gram indexes for wildcard queri
▼ 3.3 Spelling correction
3.3.1 Implementing spelling correction
3.3.2 Forms of spelling correction
3.3.3 Edit distance
3.3.4 k-gram indexes for spelling corre
3.3.5 Context sensitive spelling correct

22 2 *The term vocabulary and postings lists*

entire book as a document. A search for Chinese toys might bring up a book that mentions China in the first chapter and toys in the last chapter, but this does not make it relevant to the query. Instead, we may well wish to index each chapter or paragraph as a mini-document. Matches are then more likely to be relevant, and since the documents are smaller it will be much easier for the user to find the relevant passages in the document. But why stop there? We could treat individual sentences as mini-documents. It becomes clear that there is a precision/recall tradeoff here. If the units get too small, we are likely to miss important passages because terms were distributed over several mini-documents, while if units are too large we tend to get spurious matches and the relevant information is hard for the user to find.

The problems with large document units can be alleviated by use of explicit or implicit proximity search (Sections 2.4.2 and 7.2.2), and the tradeoffs in resulting system performance that we are hinting at are discussed in Chapter 8. The issue of index granularity, and in particular a need to simultaneously index documents at multiple levels of granularity, appears prominently in XML retrieval, and is taken up again in Chapter 10. An IR system should be designed to offer choices of granularity. For this choice to be made well, the person who is deploying the system must have a good understanding of the document collection, the users, and their likely information needs and usage patterns. For now, we will henceforth assume that a suitable size document unit has been chosen, together with an appropriate way of dividing or aggregating files, if needed.

2.2 Determining the vocabulary of terms

2.2.1 Tokenization

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called *tokens*, perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenization:

Input: Friends, Romans, Countrymen, lend me your ears;
Output: Friends | Romans | Countrymen | lend | me | your | ears

These tokens are often loosely referred to as terms or words, but it is sometimes important to make a type/token distinction. A *token* is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing. A *type* is the class of all tokens containing the same character sequence. A *term* is a (perhaps normalized) type that is included in the IR system's dictionary. The set of index terms could be entirely distinct from the tokens, for instance, they could be

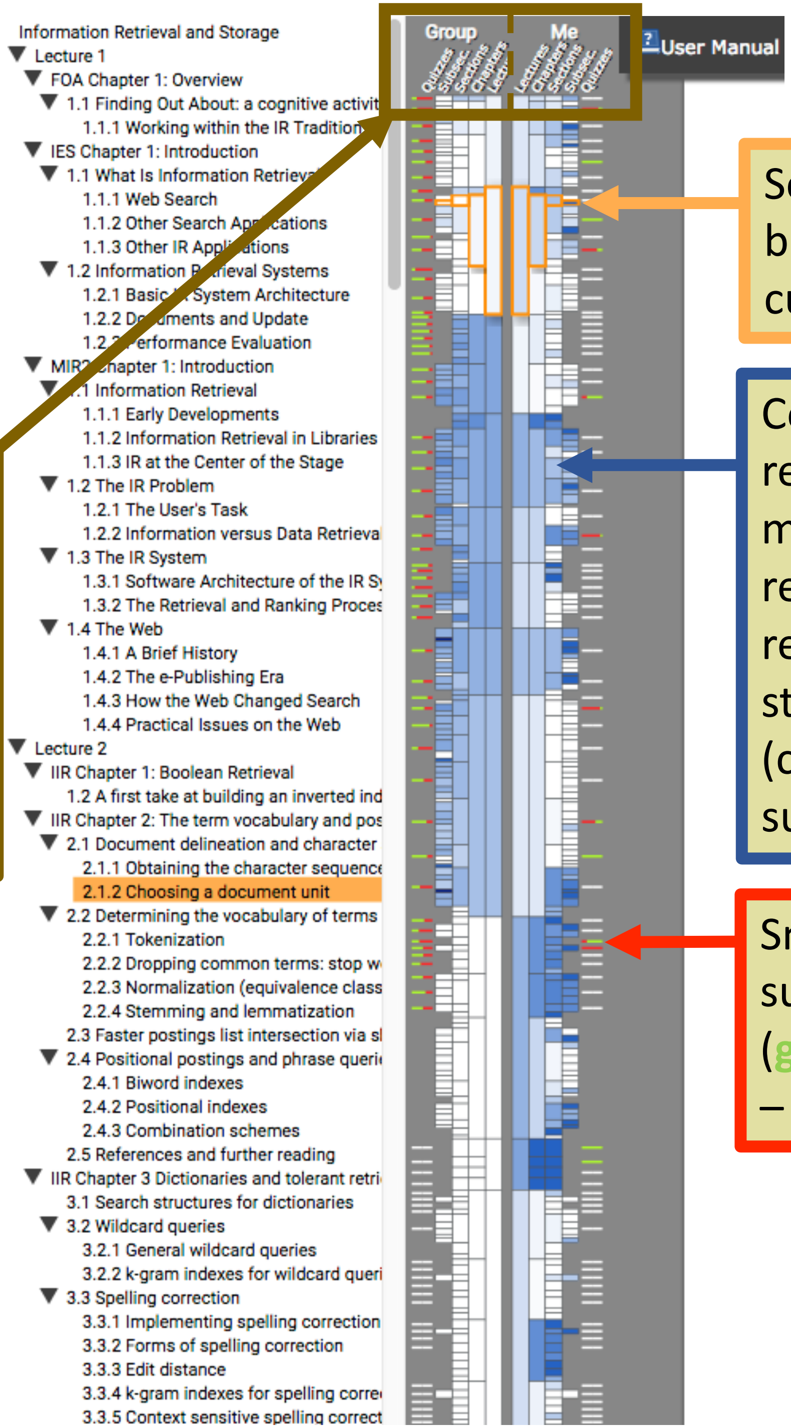
TOKEN
TYPE
TERM

Navigate through the course readings by using the *Table of Contents* or the *Previous/Next* buttons

Access to quizzes to test your knowledge at the end of specific sections (click that button when enabled)

You can annotate important passages by highlighting text and access them later through the right side panel

2. PROGRESS VISUALIZATION



Information Retrieval and Storage
- ▼ Lecture 1
  - ▼ FOA Chapter 1: Overview
    - ▼ 1.1 Finding Out About: a cognitive activit
      - 1.1.1 Working within the IR Tradition
  - ▼ IES Chapter 1: Introduction
    - ▼ 1.1 What Is Information Retrieva
      - 1.1.1 Web Search
      - 1.1.2 Other Search Ap cations
      - 1.1.3 Other IR Appl ations
    - ▼ 1.2 Information R ieval Systems
      - 1.2.1 Basic System Architecture
      - 1.2.2 D uments and Update
      - 1.2. erformance Evaluation
  - ▼ MIR hapter 1: Introduction
    - ▼ 1 Information Retrieval
      - 1.1.1 Early Developments
      - 1.1.2 Information Retrieval in Libraries
      - 1.1.3 IR at the Center of the Stage
    - ▼ 1.2 The IR Problem
      - 1.2.1 The User's Task
      - 1.2.2 Information versus Data Retrieval
    - ▼ 1.3 The IR System
      - 1.3.1 Software Architecture of the IR Sy
      - 1.3.2 The Retrieval and Ranking Proces
    - ▼ 1.4 The Web
      - 1.4.1 A Brief History
      - 1.4.2 The e-Publishing Era
      - 1.4.3 How the Web Changed Search
      - 1.4.4 Practical Issues on the Web
- ▼ Lecture 2
  - ▼ IIR Chapter 1: Boolean Retrieval
    - 1.2 A first take at building an inverted ind
  - ▼ IIR Chapter 2: The term vocabulary and pos
    - ▼ 2.1 Document delineation and character
      - 2.1.1 Obtaining the character sequence
      - 2.1.2 Choosing a document unit
    - ▼ 2.2 Determining the vocabulary of terms
      - 2.2.1 Tokenization
      - 2.2.2 Dropping common terms: stop w
      - 2.2.3 Normalization (equivalence class
      - 2.2.4 Stemming and lemmatization
    - 2.3 Faster postings list intersection via s
    - ▼ 2.4 Positional postings and phrase queri
      - 2.4.1 Biword indexes
      - 2.4.2 Positional indexes
      - 2.4.3 Combination schemes
    - 2.5 References and further reading
  - ▼ IIR Chapter 3 Dictionaries and tolerant retri
    - 3.1 Search structures for dictionaries
    - ▼ 3.2 Wildcard queries
      - 3.2.1 General wildcard queries
      - 3.2.2 k-gram indexes for wildcard queri
    - ▼ 3.3 Spelling correction
      - 3.3.1 Implementing spelling correction
      - 3.3.2 Forms of spelling correction
      - 3.3.3 Edit distance
      - 3.3.4 k-gram indexes for spelling corre
      - 3.3.5 Context sensitive spelling correct

The visualization can show your personal progress (***right side***) and the average progress of the rest of your class (***left side***)

Sections with orange border represent the current page read

Colored table show reading progress (the more **blue** the more read pages). The table reflects the ***hierarchical*** structure of the readings (chapter section subsection)

Small bars indicate success rate on quizzes (**green** correct attempts – **red** wrong attempts)