NYU School of Engineering
Computer Science and Engineering
CS 6913, Fall 2015

September 1, 2015

CS 6913: Web Search Engines

Course Prerequisites: Solid Computer Science background. Excellent programming skills, preferably in C/C++. General experience with the web and with HTML is expected. Knowledge of algorithms (similar to CS6033 or CS3414) is strongly recommended. Useful but not required are basic knowledge of networking and of Unix network and systems programming, scripting languages, and OS and database concepts.

Time and Location: W 6:00–8:30 pm, room RH 603.

Instructor: Prof. Torsten Suel, email: `torsten.suel@nyu.edu`, office: 10.046 in 2 MetroTech Center.

Office Hours: T 4:30–5:40 in 10.046, or by appointment (send email).

Course Webpage: `http://cse.poly.edu/cs6913/`

Textbooks: The following books are recommended as background reading for this course, but are not required. See also the course page for additional resources.
– Introduction to Information Retrieval, by Manning, Raghavan, and Schutze, Cambridge University Press, 2007 (available online for free).
– Mining the Web, by S. Chakrabarti, Morgan Kaufmann 2002.
– Modern Information Retrieval (2nd Edition), by Baeza-Yates and Ribeiro-Neto, ACM Press, 2011.

Grading Policy: The final grade will depend on the course projects and assignments (60%), and the final exam (40%). The final project also requires writing a longer paper describing the work and results.

Assignments. There will be several written or programming assignments. These assignments have to be done individually by each student or in teams of two, **as to be announced**. Discussions between students and help in using the programming environment are permitted and encouraged. However, no code or solutions may be copied! In the second half, students will work on a larger project, individually or in small groups. A list of available projects will be presented, and students may also propose relevant projects of their own interest.

Course Outline. This course will cover a variety of topics related to web search technology. The main focus will be on large-scale web search engines (such as Google, Bing, or Baidu) and the underlying architectures and techniques. You will learn how search engines work, and get hands-on experience in building search engines from the ground up. You will also learn about the fundamental challenges and bottlenecks in current search technologies, and about research efforts that address these issues. In this context, we will also cover fundamentals of information retrieval, data compression, and computing with massive data sets. Here are some typical topics:

- basic web protocols
- search engine architecture
- web crawling, web exploration, and web surveillance

- indexing of very large text data sets
- text and data compression
- I/O-efficient computing and computing with massive data sets
- search engine query execution and ranking functions
- use of link structure in web search and data analysis
- data mining on the web
- computational advertising
- search engine manipulation and spam
- query log mining and personalization
- advanced search engine architecture

The course will cover both algorithmic techniques and implementation aspects. Students will be required to complete several substantial programming projects in C/C++, Java, or the Python scripting language. Students will also have to read a number of research papers.

Policies and Suggestions. Following are a few tips on how to approach the assignments.

(1) Start on the assignments early! Assignments may be more substantial than they appear on first sight. But others may be easier, once you figure out the necessary parts. But they all need some planning, so a last-minute approach is discouraged.

(2) Do not reinvent the wheel! For example, Python has a lot of useful functions in its standard modules. Check it out. You many also find other useful libraries on the web. But ask me for permission if you would like to use other outside code libraries.

(3) Do not skip programming assignments! You should do all of them. Subsequent assignments may reuse code from earlier ones. Make sure you write your code such that you can modify it easily later.

(4) Discuss the assignments with classmates. Seek and provide help about things such as the programming environment, library functions, design decisions. Don't get stuck for hours because you cannot figure out, e.g., how to run a Python program or how to use a particular library - ask somebody. Ask other students, then ask the instructor if necessary.

(5) But, again, do not copy code. There are tools that will discover this!

(6) If you are in serious danger of falling behind, talk to the instructor BEFORE it is too late.

(7) Make use of the office hours. If needed, there will also be special project discussion hours, to meet as a group and discuss any common problems with the assignments.