Syllabus for Combinatorial Algorithms

Raphael Reitzig

Winter term 2015/16

2015-10-21

Overview

During this course, you will learn about selected topics out of the area of *combinatorial algorithms.* Specifically, we will investigate

- efficient *string matching*, i. .e. finding words in texts,
- methods of *compressing* strings,
- algorithms for the graph problems of finding *matchings* and *maximum flows*, and
- *random generation* of numbers and combinatorial structures.

The goal of the course is that you will be able to

- explain the problems and their solutions in your own words, using formalisms as necessary,
- apply the algorithmic solutions to simple examples, and
- attack new problems using the methods you have learned as well as reason formally about issues and solution attempts.

Teaching Agreement

The **teachers** will support the students' learning process by acting as guides, reference and source of feedback. Specifically, we make the following promises.

- Provide a reasonable set of resources for the students to work with.
- Answer to email inquiries in a timely fashion (max. 48 hours).
- Offer weekly written feedback on students' learning progress, provided there is sufficient input (see below).
- Be available for at least three hours of exclusive interaction per week, at a fixed time and place (which we will call "meetings").
- Communicate updates on content or organisational details in a timely fashion.

The **students** will take responsibility over their own learning and use the provided resources to actively engage with the material. Specifically, we expect the following.

- Work through the given resources on your own time.
- Prepare a short, machine-written essay ($\approx$ one page) every week that answers the current guidance question(s) to the best of your ability.

  Hint: If you do *not* have an answer, *explain why not*! Nailing down the issue(s) you are having is a useful exercise in itself.
- Be prepared for the meetings, both mentally[1] and physically[2].
- Attend the meetings and engage with the other students.
- Work on a significant subset of the exercises and hand in clearly presented (attempts at) solution – and retry on failure!
- Check their email inbox frequently.

Organization

Your guide for this course is Raphael Reitzig[3].

We will meet every *Thursday* at *15:30* o'clock in *48-654*. This time is designated for Q&A regarding the material you prepare as well as the exercises, and collaborative tinkering on the same. There will be quick assessments at the beginning to start off the discussion.

You will hand in your essays before or during the meeting. You can submit LaTeX-ed PDFs or handwritten notes. Re-submissions are possible.

We have compiled a set of exercises that build upon the material you work through and should help you clarify it. We will give you all exercises up front; you decide when you work on which. You can hand in anything you want feedback on. Re-submissions are possible and encouraged.

Important: Whenever you hand something in, please

- put your name on every sheet,
- leave a margin of at least 4 cm along every edge,
- structure your writings appropriately, and
- write clearly.

We will not be able to provide feedback on work that does not follow these humble standards.

[1]Be awake. Bring clear understanding and/or questions about the current material.
[2]Bring your notes and the material.
[3]`http://wwwagak.cs.uni-kl.de/home/staff/raphael-reitzig`

Content

This section is organized by week. We will identify a week by its *index* in the reading period and by the *date of its class*; this means that you will have to deal with significant parts of the work on the listed material *before* that date.

For each week, we give you three things.

Main references: The sources you *should* read. In particular, these (together with the exercise problems) define the scope of exam questions.

Supplementary references: Sources you *may want to* read, e. g. if you found the main references unclear or are interested in further material.

You are invited to extend this list by yourself; make it as long as you need it to be, that is until you feel that you have understood the main reference sufficiently!

Guiding question: A question you should be able to answer after absorbing the specified material. Defines the topic of your weekly essay.

Week 0: Kick-Off

We will discuss how this course will be organised, in particular the content of this here document.

Session: 29.10.2015

Week 1: String Matching I

About naive string matching, string matching with automata, and the algorithm of Knuth-Morris-Pratt.

Main references:
Nebel [Neb12, Sections 6.1.1–.3]

Supplementary references:
Sedgewick and Wayne [SW11, Section 5.3 up to p. 769],
Sedgewick and Wayne [SW15, Week 4.2, lectures 1–3]

Guiding Question:
What are the similarities and differences of how SMA and KMP work, respectively?

Session: 05.11.2015

Week 2: String Matching II

More algorithms for string matching.

Main references:
Nebel [Neb12, Sections 6.1.4–.5]

Supplementary references:
Sedgewick and Wayne [SW11, Section 5.3 from p. 770],
Sedgewick and Wayne [SW15, Week 4.2, lectures 4–5],
Nebel [Neb12, Section 6.2]

Guiding Question:
Suppose you want to search with *wildcards*. How do you go about that, and why?

Session: 12.11.2015

Week 3: String Compression I

About two of the most fundamental compression algorithms, LZ77 and LZ78.

Main references:
Ziv and Lempel [ZL77; ZL78]

Supplementary references:
Kaiser [Kai14, Chapters 2–3],
Lempel and Ziv [LZ76],
Salomon and Motta [SM09, Chapter 6]

Guiding Question:
How do the two algorithms relate in terms of design choices? Which are still apt/relevant today (contrasted with 1980)?

Session: 19.11.2015

Week 4: String Compression II

About a third algorithm in the LZ family, namely LZW.

Main references:
Sedgewick and Wayne [SW11, Section 5.5, from p. 839]

Supplementary references:
Kaiser [Kai14, Chapter 4],
Sedgewick and Wayne [SW15, Week 5.2, lecture 4],
Salomon and Motta [SM09, Chapter 6]

Guiding Question:
How does LZW improve upon its older cousins? Which would you implement *today* and why?

Session: 26.11.2015

Week 5: Network Flows I

About the maximum flow problem and the augmenting paths meta algorithm.

Main references:
Krumke and Noltemeier [KN12, Sections 9.1–.5]

Supplementary references:
Sedgewick and Wayne [SW11, p. 886–902],
Sedgewick and Wayne [SW15, Week 3.1],
Cormen et al. [CLRS09, Sections 26.1–.2]

Guiding Question:
Why are “oscillation graphs” a problem for Ford-Fulkerson but not for Edmonds-Karp?

Session: t. b. d.

Week 6: Network Flows II

About push-relabel (also: preflow-push) algorithms for finding maximum flows.

Main references:
Krumke and Noltemeier [KN12, Section 9.7]

Supplementary references:
Cormen et al. [CLRS09, Section 26.4]

Guiding Question:
Can different strategies for picking *edges* influence the performance of push-relabel algorithms?

Session: 10.12.2015

Week 7: Assignments & Matchings

About the marriage problem and related issues.

Main references:
Krumke and Noltemeier [KN12, Sections 9.10.1–.2, 10.1–.3, 10.6]

Supplementary references:
Cormen et al. [CLRS09, Section 26.3],
“Blossom algorithm” on Wikipedia

Guiding Question:
Can we use the M-augmenting path algorithm [KN12, p. 279] for general graphs?

Session: 17.12.2015

Week 8: Randomness

About what randomness is and what we can hope to achieve.

Main references:
Knuth [Knu01, Sections 3.1, 3.5]

Guiding Question:
What is randomness, intuitively? How can this intuition be caputured in mathematical terms?

Session: 07.01.2016

Week 9: Generating Random Numbers

On generating random numbers assuming we have *some* uniform source of randomness.

Main references:
Knuth [Knu01, Section 3.4]

Supplementary references:
Knuth [Knu01, Section 3.2],
"Mersenne Twister" on Wikipedia,
Knuth [Knu00, Chapter 34]

Guiding Question:
Given an arbitrary random distribution on numbers, how can we sample values according to it?

Session: 14.01.2016

Week 10: Testing Random Number Generators

Given a candidate random number generation, how can we determine if it is good? And what does *good* mean here?

Main references:
Knuth [Knu00, Chapter 2],
Knuth [Knu01, Sections 3.3.1–.3]

Supplementary references:
Knuth [Knu01, Section 3.3.4]

Guiding Question:
How can we make sure that a given random number generator is correct? Can we make sure it is broken?

Session: 21.01.2016

Week 11: Symbolic Method

Main references:
Sedgewick and Flajolet [SF13, Sections 5.1–.3]

Supplementary references:
Sedgewick and Flajolet [SF13, Section 5.4],
Sedgewick [Sed13, Week 5, lectures 1–2]

Guiding Question:
How does the symbolic method relate to formal grammars?

Session: 28.01.2016

Week 12: Random Generation of Combinatorial Objects

On generating combinatorial objects assuming we have *some* uniform source of randomness.

Main references:
Flajolet, Zimmerman, and Van Cutsem [FZV94, Sections 1–4]

Supplementary references:
Duchon et al. [DFLS04, Sections 1–4, 6], Sedgewick and Flajolet [SF13, Section 5.5]

Guiding Question:
What kind of specifications lend themselves badly to *fast* random generation? Can we circumvent the effect?

Session: 04.02.2016

Week 13: Recap & Questions

In our last meeting, we will address any open issues.

Guiding Question:
Consider any overarching principles you have discovered during this class. Which has captured your interest the most, and why? Which has challenged your intuition the most?

Session: 11.02.2016

References

[CLRS09] Thomas H. Cormen et al. *Introduction to Algorithms.* 3rd ed. MIT Press, 2009. ISBN: 978-0-262-03384-8. URL: `http://mitpress.mit.edu/books/introduction-algorithms`.

[DFLS04] Philippe Duchon et al. "Boltzmann Samplers for the Random Generation of Combinatorial Structures." English. In: *Combinatorics, Probability and Computing* 13.4-5 (July 2004), pp. 577–625. ISSN: 1469-2163. DOI: `10.1017/S0963548304006315`.

[FZV94] Philippe Flajolet, Paul Zimmerman, and Bernard Van Cutsem. "A calculus for the random generation of labelled combinatorial structures." In: *Theoretical Computer Science* 132.1-2 (Sept. 1994), pp. 1–35. ISSN: 03043975. DOI: `10.1016/0304-3975(94)90226-7`.

[Kai14] Markus Kaiser. "Lempel-Ziv Komprimierung vorgestellt und empirisch untersucht." Seminar. University of Kaiserslautern, 2014.

[KN12] Sven Oliver Krumke and Hartmut Noltemeier. *Graphentheoretische Konzepte und Algorithmen.* Wiesbaden: Vieweg+Teubner Verlag, 2012. ISBN: 978-3-8348-1849-2. DOI: `10.1007/978-3-8348-2264-2`.

[Knu00] Donald E. Knuth. *Selected Papers on Analysis of Algorithms.* 1st ed. CSLI lecture notes 102. Stanford: CSLI Publications, 2000. ISBN: 978-1-57586-212-5. URL: `http://www-cs-faculty.stanford.edu/~uno/aa.html`.

[Knu01] Donald E. Knuth. *Seminumerical Algorithms.* 3rd ed. Vol. 2. The Art Of Computer Programming. Addison-Wesley Longman Publishing, 2001. 762 pp. ISBN: 0201896842.

[LZ76] Abraham Lempel and Jacob Ziv. "On the Complexity of Finite Sequences." In: *Information Theory, IEEE Transactions on* 22.1 (1976), pp. 75–81. DOI: `10.1109/TIT.1976.1055501`.

[Neb12] Markus E. Nebel. *Entwurf und Analyse von Algorithmen.* Wiesbaden: Vieweg+Teubner Verlag, 2012. ISBN: 978-3-8348-1949-9. DOI: `10.1007/978-3-8348-2339-7`.

[Sed13] Robert Sedgewick. *Analytic Combinatorics, Part I.* 2013. URL: `https://class.coursera.org/introACpartI-001`.

[SF13] Robert Sedgewick and Philippe Flajolet. *An Introduction to the Analysis of Algorithms.* 2nd ed. Addison-Wesley Professional, 2013, p. 592. ISBN: 032190575X.

[SM09] David Salomon and Giovanni Motta. *Handbook of Data Compression.* 5th ed. Springer, 2009. ISBN: 978-1-84882-902-2.

[SW11] Robert Sedgewick and Kevin Wayne. *Algorithms.* Addison-Wesley, Mar. 2011. ISBN: 978-0-321-57351-3.

[SW15] Robert Sedgewick and Kevin Wayne. *Algorithms, Part II.* 2015. URL: `https://www.coursera.org/course/algs4partII`.

[ZL77] Jacob Ziv and Abraham Lempel. "A universal algorithm for sequential data compression." In: *Information Theory, IEEE Transactions on* 23.3 (1977), pp. 337–343. DOI: `10.1109/TIT.1977.1055714`.

[ZL78] Jacob Ziv and Abraham Lempel. "Compression of individual sequences via variable-rate coding." In: *Information Theory, IEEE Transactions on* 24.5 (1978), pp. 530–536. DOI: `10.1109/TIT.1978.1055934`.