Informatics 2A 2016–17

Lecture 32

Revision Lecture

John Longley
Adam Lopez

Reminder: pass criteria

By 4pm tomorrow, you will have completed your coursework. This accounts for 25% of the course mark.

The remaining 75% of the course mark is provided by the exam.

For a pass in Inf2A, you need a combined mark of at least 40%.

(No separate exam and coursework hurdles this year.)

The 2016 Inf2A Exam

December exam time and location:

> INFR08008 - Informatics 2A
> Location: Patersons Land - G1 (Surname A - PAR), Patersons Land - G.21 (Surname PAT - TO), Patersons Land - G.42 (Surname TS - Z)
> Date/Time: Saturday 10/12/2016, 09:30:00-11:30:00 (02:00:00)

This is copied from the Registry exam timetable

`http://www.scripts.sasg.ed.ac.uk/registry/examinations/`

which is the official exam timetable. Make sure that you use this link to double-check all your exam times (including Inf2A).

A resit exam will be held in August 2017.

Exam structure

The exam is pen-and-paper, and lasts 2 hours.
Calculators may be used, but you must bring your own. It must be one from an approved list of models specified by College:

`http://edin.ac/1RNRSfa`

The exam consists of:

- Part A: 5 compulsory short questions, worth 10% each.

  Guideline time per question: 10 minutes
- Part B: a choice of 2 out of 3 longer questions, worth 25% each.

  Guideline time per question: 30 minutes

The guideline times allow 10 minutes for reading and familiarising yourself with the exam paper.

Part A questions

The 5 compulsory short questions were new in 2012 and replaced 20 multiple-choice questions in previous years.

The questions will be similar in style and length (but not necessarily in topic) to the questions on this week's Tutorial 9.

The multiple-choice questions of previous years still provide good revision material in terms of coverage of topics.

Examinable material

Examinable material: formal language thread

Lecture 2 (the course roadmap) should be considered examinable.

All of the material on regular and context-free languages (Lectures 3–14) is examinable, except:

- Use of finite automata in verification (Lecture 7, slides 12–15)
- Specific details of Micro-Haskell (Lecture 13, slides 7–12)
- Examples of English palindromes (Lecture 14, slides 16–19)

Examinable material: formal language thread (contd.)

Lecture 28 (semantics of programming languages, in particular MH) may be considered non-examinable.

Lecture 29 (context-sensitive languages): mostly examinable – but for the context-free pumping lemma, just the general idea will suffice.

Lecture 30 (Turing machines, linear bounded automata): general ideas examinable but not detailed definitions/proofs.

Lecture 31 (undecidability): not examinable.

Kinds of exam question: formal language thread

Broadly speaking, 2 styles of question in exam.

Algorithmic problems: Minimizing a DFA, converting NFA to DFA, executing a PDA, LL(1) parsing using parse table, generating parse table from LL(1) grammar, . . .

When the algorithm is complex (e.g., minimization, calculating first and follow sets), it may be easier to work with your understanding of the concepts rather than following the algorithm strictly to the letter.

Non-algorithmic problems: Converting DFA to regular expression, designing regular expression patterns, applying pumping lemma, designing CFGs, converting CFG to LL(1), parsing using CSG or noncontracting grammar, . . .

Examinable material: natural language thread

The main thing being tested is your ability to understand and apply the *methods* for solving certain standard kinds of problems.

Algorithmic problems:

- POS tagging via bigrams or Viterbi algorithm (lecture 17).
- CYK and Earley parsing (lectures 20, 21).
- Probabilistic CFGs; probabilistic CYK; inferring probabilities from a corpus (lectures 22, 23).
- Computing semantics, including β-reduction (lecture 25).

Examinable material: natural language thread (continued)

Non-algorithmic problems **(simple examples only!)**

- Design of a transducer for some morphology parsing task (lecture 15).
- Design of context-free rules for some feature of English. (Includes parameterized rules for agreement — lecture 23.)
- Converting an English sentence to a formula of FOPL (lecture 24).
- Adding semantic clauses to a given context-free grammar (lecture 25).

In all such cases, you will be given specific instructions on what the designed system should compute (input and output).

Examinable material: natural language thread (continued)

General topics

- The language processing pipeline (lecture 2).
- Kinds of ambiguity (lectures 2, 16, 19, 25).
- The Chomsky hierarchy, and where human languages sit (lectures 2, 26, 27).
- The *general idea* of parts of speech (lecture 16).
- Word distribution and Zipf's law (lecture 16).

Recursive descent parsing (lecture 19) is only weakly examinable.

Non-examinable material: natural language thread

- Specific knowledge of linguistics (everything you need will be given in the question).
- Details of particular POS tagsets; ability to do POS tagging by hand (lectures 16, 17).
- Fine-grained typing, e.g. selectional restrictions on verbs (lecture 23).
- Mildly context-sensitive grammars (lecture 27).
- Human parsing (lecture 27).

Follow-on Informatics courses

Compiling techniques (UG3)

Covers the entire language-processing pipeline for programming languages, aiming at effective compilation: translating code in a high-level source language (Java, C, Haskell, . . .) to equivalent code in a low-level target language (machine code, bytecode)

Syllabus includes lexing and parsing from a more practical perspective than in Inf2A.

Majority of course focused on latter stages of language-processing pipeline. Converting lexed and parsed source-language code into equivalent target-language code.

Currently an assignment-only course, no exam: you build a compiler!

Introduction to theoretical computer science (UG3)

This will look at models of computation (register machines, Turing machines, lambda-calculus) and their different influences on computing practice.

One thread will address the boundaries between what is not computable at all (undecidable problems), what is computable in principle (decidable problems), and what is computable in practice (tractable problems). A major goal is to understand the famous P = NP question.

Another thread will look at the influence lambda-calculus has had, as a model of computation, on programming language design and practice, including LISP, OCaml, Haskell and Java.

Natural Languages: what we've done, what we haven't.

NLs are endlessly complex and fascinating. In this course, we have barely scratched the surface.

There's a world of difference between doing NLP with small toy grammars (as in this course) and wide-coverage grammars intended to cope with real-world speech/text.

- Ambiguity is the norm rather than the exception.
- Empirical and statistical techniques (involving text corpora) come to the fore, as distinct from logical and symbolic ones.

Coping with the richness and complexity of real-world language is still a largely unsolved problem!

Discourse structure.

In this course, we haven't considered any structure above the level of sentences. In practice, higher level discourse structure is crucial. E.g.

> The Tin Man went to the Emerald City to see the Wizard of Oz and ask for a heart. Then he waited to see whether he would give it to him.

Or compare:

- Bob wants Alice to give him a job.
- Charles has graduated. Bob wants Alice to give him a job.

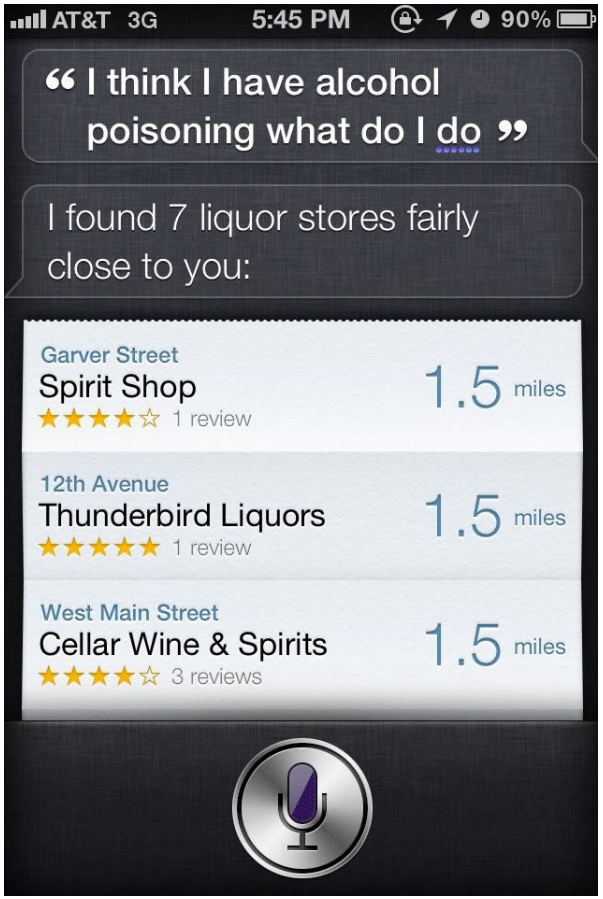
Deep vs. shallow processing.

Roughly, the further we go along the NLP pipeline, the deeper our analysis.

- Many apparently 'shallow' NLP tasks (e.g. spell checking; speech transcription) can benefit from the use of 'deeper' techniques such as parsing.

- On the other hand, for many seemingly 'deep' tasks (e.g. machine translation), current state-of-the-art techniques are surprisingly 'shallow' (e.g. use of N-gram techniques with massive corpora).

Shallow techniques do not always work.



Follow-on courses in NLP

- **Foundations of Natural Language Processing** [UG3]. Empirical rather than theoretical in focus. Material on text corpora, N-grams, the 'noisy channel' model.
- **Machine Translation** [UG4]. Mainly on probabilistic (neural network) models for MT, but includes substantial discussion of how NL phenomena interact with these models. Find out how Google Translate works!
- **Natural Language Understanding** [UG4]. Considers the LP pipeline much in the spirit of Inf2a (i.e. syntax and semantics), but with modern machine learning techniques.
- **Topics in Natural Language Processing** [UG4]. Get acquainted with state of the art in NLP and read cutting-edge research papers in NLP and machine learning.

Thank you!!

Hope you've enjoyed Inf2A,
and good luck with the exam!

Please complete the online course questionnaire when it becomes available.