Computational Linguistics 2 (Revised syllabus)

Course	Info
Course#	lin637
Time	T/R 2:30–3:50pm
Location	Zoom (https://lin637.thomasgraf.net/zoom)
Website	<lin637.thomasgraf.net>
Instructor	Thomas Graf
Email	t.graf@stonybrook.edu
Office hours	T 1:00–2:30
	R 1:00–2:30
Office	Zoom (https://officehours.thomasgraf.net)

If you cannot reach your instructor, please email cas_dean@stonybrook.edu

Modality change: Starting Tuesday, March 31, all lectures will be held online via Zoom, and recordings will be made available through the course website. Assignments will continue to be distributed, collected, and graded online as usual. To account for the switch to online instruction, some minor changes have to be made to the course format. Please see the sections **Grading** and **Online component** for details.

Course outline

Bulletin description

An introduction to the theoretical foundation of computational linguistics. The course emphasizes the importance of algorithms, algebra, logic, and formal language theory in the development of new tools and software applications. Empirical phenomena in phonology and syntax are sampled from a variety of languages to motivate and illustrate the use of concepts such as strictly local string languages, tree transducers, and semirings. Students will develop familiarity with the literature and tools of the field.

Full description

This course serves a specific purpose in our program: it acts as the bridge from introductory courses in linguistics (Syntax 1, Phonology 1, Phonetics) and computational methods (Statistics, Mathematical Methods in Linguistics, Computational Linguistics 1) to advanced courses and seminars in computational/mathematical linguistics. In contrast to the NLP courses offered by the department of computer science, this course focuses on studying the properties of natural language from a computationally informed perspective. The question

is not how computers can solve language-related tasks, but how language can be conceptualized as a computational problem. This emphasis is also reflected in the selection of topics for this course.

- **What this course is not about**
  - Computer-assisted research methods in linguistics
  - Software development for natural language tasks
- **What is not covered but benefits from what is covered**
  - Speech recognition
  - OCR
  - Text generation
  - Parsing
  - Semantic analysis
  - Machine translation
- **List of topics**
  - *Phonology and morphology*
    - The role of formalization
    - String languages
    - Subregular hierarchy
    - Regular languages
    - Generative capacity of phonology
    - String transductions
    - 2-level morphology
    - Equivalence of SPE and OT
  - *Syntax*
    - Tree languages
    - Syntax is more complex than phonology
    - Mildly context-sensitive formalisms (TAG, MGs)
    - Tree transductions
    - Regular representations of MCS formalisms
    - Reinterpreting the T-model

A rough outline of the course progression is given in the table below. Many of the topics we cover draw from very specialized areas of formal language theory that even most mathematicians and computer scientists do not know about, e.g. the correspondence between finite-state machines and monadic second-order logic, or the logical characterization of tree transductions. So this course might be of value to you even if you do not particularly care about natural language. Make no mistake, though, we'll talk a lot about language and linguistics — this is not a math class.

Wk	*Formal*	*Linguistics*
1	What is computation?	Marr's Three Levels
2	Formalizing phonology	Why formalize?
3	Strictly local languages	Local dependencies
4	Subregular hierarchy	How powerful is phonology?

Wk	*Formal*	*Linguistics*
5	Regular languages	Abstractness
6	String transductions	SPE-OT equivalence
7	Two-level morphology	Null morphemes
8	(Spring Break)	
9	(Extended Spring Break)	
10	Weak Generative Capacity	Phonology < Syntax
11	Tree languages	Headedness, feature percolation
12	Local tree languages	GPSG
13	Recognizable tree languages	GB
14	TAG and MGs	Minimalist syntax
15	Tree transductions	Reinterpreting the T-model

Prerequisites

The only official prerequisite is Computational Linguistics 1 (Lin 537) or comparable programming skills in Python. Python will be used to illustrate formal concepts, and some of the homeworks will require you to implement an algorithm or procedure in Python. Prior experience with git and markdown is useful for the homework assignments but not required.

It is also helpful to have some basic familiarity with linguistics (phonemes, phrase structure rules, syntactic trees) and mathematics (sets, functions, relations, and first-order logic as covered in Semantics 1, for instance). You can take an online survey to identify weaknesses, and several introductory readings on these topics are available on the course website.

- **Survey URL:** https://testmoz.com/432409

Teaching goals

- **Practical skills**
  - conceptualize a problem in mathematical terms
  - optimize your programs through the use of adequate algorithms and data structures (dynamic programming techniques, hash tables, etc.)
  - a more abstract and theoretically informed perspective on current tools and techniques in NLP
  - an understanding for how linguistic insights can be invoked to simplify NLP tasks
- **Research skills**
  - assess linguistic phenomena from a computational perspective
  - evaluate linguists' claims about computational efficiency
  - basic overview of current research in theoretical computational linguistics
  - use computational concepts to identify new empirical generalizations
  - bring linguistic data to bear on computational claims

– mathematically informed understanding of linguistic theories

Grading

This course cannot be taken for variable credit. Students who want to take the course for less than 3 credits should contact me as soon as possible.

Student grades are determined by the following components. Note that **all components contribute equally to the final grade**. There is no final exam.

1. **Class participation** (20%)
   As the shift to remote teaching will inevitably introduce some technical hurdles for some of you, attendance is no longer mandatory. However, you are still expected to contribute to the course on a regular basis, in particular online discussion and collaboration. I will post Github issues with specific topics for you to discuss and work on together with your classmates. The procedure is the same as for providing feedback on the lecture notes (see below).

   Of course you are still highly encouraged to attend the Zoom meetings at our regular class time to discuss the materials. Zoom meetings will also be recorded and shared with you on the course website.

2. **Class summary** (20%)
   Due to the shift of remote teaching, oral class summaries are no longer required. Students instead have to do one (1) the following:

   - provide a short summary (at most 1 page) of an online student discussion, or
   - design an extra set of exercises for the lecture notes, or
   - pick one of the optional reading assignments and send me a 1-page summary, to be uploaded to the readings repository, or
   - design a short tutorial on a specific tool (e.g. xfst).

   These items are listed in increasing order of expected workload. Only pick the lower items if you like the idea of spending time on a topic you truly love even though it might mean more work for you.

3. **Assignments** (20%)

   - exercises, programming assignments, or critical evaluations of assigned readings
   - Homework submission and grading is done via GitHub.
   - No late hand-ins!
   - Code that does not run will be returned ungraded.
   - Collaboration on homework problems is encouraged as long as you write up the solutions by yourself, using your own words, examples, notation, and code.

- Under no circumstances should you copy code from the internet. That's a severe case of academic dishonesty.

4. **Readings** (20%)
    - at most two readings per week, but sometimes none beyond the lecture notes
    - Readings will be announced and distributed via GitHub.
    - It is presupposed in the lectures that you have done the required readings.
    - Reading comprehension may be tested as part of the homework.
5. **Lecture Note Feedback** (20%)
    - I plan to publish the lecture notes as an open-access textbook with *Language Science Press*.
    - This is the last time I teach the course before the submission deadline, so I want feedback.
    - Every week, you should file issues on GitHub for the relevant units, where you spot typos, suggest exercises, pictures, examples, etc.
    - This is a collaborative enterprise: comment on other student's suggestions if you (dis)approve, expand their ideas, and so on.

Online component

This class uses some online tools to facilitate homework collaboration and submission, student discussions, and dynamic lecture evaluation. In addition, all class meetings will be held online after Spring break.

Online meetings

- *How it works:* After Spring break, we will use Zoom to meet at our regular class time. Tuesday sessions will still operate like a regular class, with me presenting materials in an interactive fashion (assuming my dual-camera setup works as intended). Thursday sessions will be a free-style discussion and QnA format where you can ask about materials in the lecture notes, homework assignments, and so on.
- *Why we do it:* Strictly speaking because there's not alternative, but I think this will open up some exciting opportunities for us nonetheless.
- *What you'll need:* If at all possible, try to join the Zoom meetings instead of watching the recordings online. Make sure you have the Zoom client installed (it's available for Windows, OS X, and Linux). You can also join a Zoom meeting through your browser or by calling in with your phone, but those are suboptimal experiences that lack some key features of the client.

Homework submission

- *How it works:* Each homework assignments is in its own private GitHub repository. You can fork this repo and upload your own code, or checkout other students' forks to see how they dealt with the problem. In order to submit a homework you upload your solution to your fork and issue a pull request. After the due date, I'll upload my solution to the repository.
- *Why we do it:* This setup mimics the modern workflow in collaborative development projects. Git is one of the best-known version control systems, and GitHub is the biggest online service for hosting git repositories. Familiarity with version control systems is an essential job requirement for computational linguists.

  Git skills are also very helpful for academic work. See this discussion on Stackoverflow for some ideas how git can be used in conjunction with Latex: http://stackoverflow.com/questions/6188780/git-latex-workflow
- *What you'll need:* A GitHub account (the free tier is enough) and a way of uploading your code to a GitHub repository. Linux users can install git via the command line. Windows and Mac users should download and install the provided virtual machine image. Please do not use the GitHub app, its GUI makes things more complicated than just using the command line directly.

Homework feedback and discussion

- *How it works:* when you submit a pull request, I can add specifics comments to your code that will be visible to you through the GitHub interface.
- *Why we do it:* Once again this is an essential part of modern software development. And it is also a lot more convenient than anything Blackboard has to offer.
- *What you'll need:* The willingness to master GitHub's user interface.

Class announcements

- *How it works:* All announcements are published as issues in the main repository. New readings are assigned as issues in the readings repository. Make sure to subscribe to both repositories (using the **Watch** button) to get notifications.
- *Why we do it:* Again because issues are a cornerstone of modern software development.

Lecture notes

- *How it works:* The lecture notes are made available online Monday and Wednesday before 3pm. You can look at them on your laptop/tablet or

make a hardcopy before class. I will not bring handouts to class.

- *Why we do it:* Mostly because I just don't like paper. Also keep in mind that you can fork the lecture notes repository and include your own notes directly in the Latex source files. Then you can compile a version of the lecture notes with your own notes already included.

Policies

Contacting me

- Emails should be sent to t.graf@stonybrook.edu. Disregarding this policy means late replies and is a sure-fire way to get on my bad side.
- Reply time $<$ 24h in simple cases, possibly more if meddling with bureaucracy is involved.
- If you want to come to my office hours and anticipate a longer meeting, please email me so that we can set apart enough time and avoid collisions with other students.

Disability support services

If you have a physical, psychological, medical, or learning disability that may impact your course work, please contact the Student Accessibility Support Center, 128 ECC Building, (631) 632-6748, or at sasc@Stonybrook.edu. They will determine with you what accommodations are necessary and appropriate. All information and documentation is confidential.

Students who require assistance during emergency evacuation are encouraged to discuss their needs with their professors and the Student Accessibility Support Center. For procedures and information go to the following website: https://ehs.stonybrook.edu/programs/fire-safety/emergency-evacuation/evacuation-guide-people-physical-disabilities and search Fire Safety and Evacuation and Disabilities.

Academic integrity

Each student must pursue his or her academic goals honestly and be personally accountable for all submitted work. Representing another person's work as your own is always wrong. Faculty is required to report any suspected instances of academic dishonesty to the Academic Judiciary. Faculty in the Health Sciences Center (School of Health Technology & Management, Nursing, Social Welfare, Dental Medicine) and School of Medicine are required to follow their school-specific procedures. For more comprehensive information on academic integrity, including categories of academic dishonesty please refer to the academic judiciary website at http://www.stonybrook.edu/commcms/academic_integrity/index.html.

Critical incident management

Stony Brook University expects students to respect the rights, privileges, and property of other people. Faculty are required to report to the Office of University Community Standards any disruptive behavior that interrupts their ability to teach, compromises the safety of the learning environment, or inhibits students' ability to learn. Faculty in the HSC Schools and the School of Medicine are required to follow their school-specific procedures. Further information about most academic matters can be found in the Undergraduate Bulletin, the Undergraduate Class Schedule, and the Faculty-Employee Handbook.