LING 408/508: Computational Techniques for Linguists

Lecture 1

Administrivia

1. Syllabus
2. Questions about the Syllabus
3. Introduction

- I will assume everyone has a laptop or desktop …

Syllabus

Description of Course

- An introductory level course to computers and programming for linguists.

Course Pre-requisites

- None!

Instructor and Contact Information

- Instructor: Sandiway Fong, Douglass 311.
- Contact email: sandiway@email.arizona.edu (all homework to be submitted here).
- Instructor: Sandiway Fong, Dept. of Linguistics Office: Douglass 311

Syllabus

Hours:

- make appointments by email: meet online
- Zoom during class time (*best if you have quick Qs*)

Meet:

- on Zoom (watch your email!) and Facebook Messenger (sandiway) with Panopto slides
- No class on Thursday Nov 26th (Thanksgiving)
  - *updated as necessary…*

Course Format and Teaching Methods

- Lecture with slides. Panopto videos (when available) for lecture review.
- All homeworks will be introduced and reviewed in Zoom class.

Syllabus

Course Objectives

Topics covered include:

- Fundamental computer concepts
    - computer organization: underlying hardware, and operating systems (processes, shell, filesystem etc.)
- Operating System:
    - Ubuntu (Linux)
- Introduction to programming
    - data types, different programming styles, thinking algorithmically …
- Programming Languages:
    - *selected examples*: Bash shell, Python, Javascript, Perl, Tcl/Tk, HTML/CSS, cgi-bin etc.

Syllabus

Course Learning Outcomes

After completing this course, students will:

- be familiar with the underlying technology: *what makes a computer tick*?
- acquire the ability to think algorithmically
- acquire the ability to write short programs
- build a graphical user interface (GUI)
- build a web application (with a relational database)
- be equipped to take classes in the Human Language Technology (HLT) program and related classes

Syllabus

Absence and Class Participation Policy

- I expect you to attend lectures (though attendance will not be taken).
- The UA's policy concerning Class Attendance, Participation, and Administrative Drops is available at: http://catalog.arizona.edu/policy/class-attendance-participation-and-administrative-drop.
- Tell me ahead of time so we can make alternative arrangements in the case of missed homeworks. **No homework will be accepted late. Explained below.**
- Absences pre-approved by the UA Dean of Students (or Dean Designee) will be honored. See: https://deanofstudents.arizona.edu/absences.
- The UA policy regarding absences for any sincerely held religious belief, observance or practice will be accommodated where reasonable, http://policy.arizona.edu/human-resources/religious-accommodation-policy.

Syllabus

Required Text

- None

Required or Special Materials

- All required software will be available online at no cost to the student.
- However, students are expected to either have a laptop/desktop capable of handling homework and classwork, or make use of UA lab computers (?)
- Mac, PC (Windows 10) or Linux.

Syllabus

Final Examination or Project

- No examinations, e.g. mid-term or final, are scheduled for this course.

Grading Scale and Policies

- homework exercises (50%)
- a term programming project (50%)
- **ungraded homework exercises too**
- Requests for incomplete (I) or withdrawal (W) must be made in accordance with University policies, which are available at http://catalog.arizona.edu/policy/grades-and-grading-system#incomplete and http://catalog.arizona.edu/policy/grades-and-grading-system#Withdrawal respectively.

Syllabus

Assignments and Examinations: Schedule/Due Dates

- All homeworks will be introduced **and reviewed** in Zoom class.
- Homework submissions by email to me only.
- Late homework will be not accepted since all homeworks will be solved/reviewed in class.
- Quick homeworks are normally due at midnight before the next class, and are generally assigned in class on a **Tuesday** and due **Wednesday** midnight (before **Thursday** class).
- Homeworks not categorized as quick are normally assigned in class on a **Thursday** and due the following **Monday** midnight (before **Tuesday** class). As deemed appropriate by the instructor, some longer homeworks may have an extended due date.
- Students can expect a total of around 8-12 homeworks over the course.
- An in-class quick quiz or two maybe be scheduled. If so, students will be notified ahead of time during the prior class.

Syllabus

Code of Academic Integrity

- You may discuss homework questions with anyone.
- You may look things up on the web and use answers found therein; however, you must write it up yourself (in your own words/own code *etc*.).
- You must cite all (web) references and your classmates (in the case of shared discussion).
- Students are encouraged to share intellectual views and discuss freely the principles and applications of course materials.
- However, graded work/exercises must be the product of independent effort unless otherwise instructed.
- Students are expected to adhere to the UA Code of Academic Integrity as described in the UA General Catalog. See: http://deanofstudents.arizona.edu/academic-integrity/students/academic-integrity.

Syllabus

UA Nondiscrimination and Anti-harassment Policy

- The University is committed to creating and maintaining an environment free of discrimination; see http://policy.arizona.edu/human-resources/nondiscrimination-and-anti-harassment-policy.

Subject to Change Statement

- Information contained in the course syllabus, other than the grade and absence policy, may be subject to change with advance notice, as deemed appropriate by the instructor.

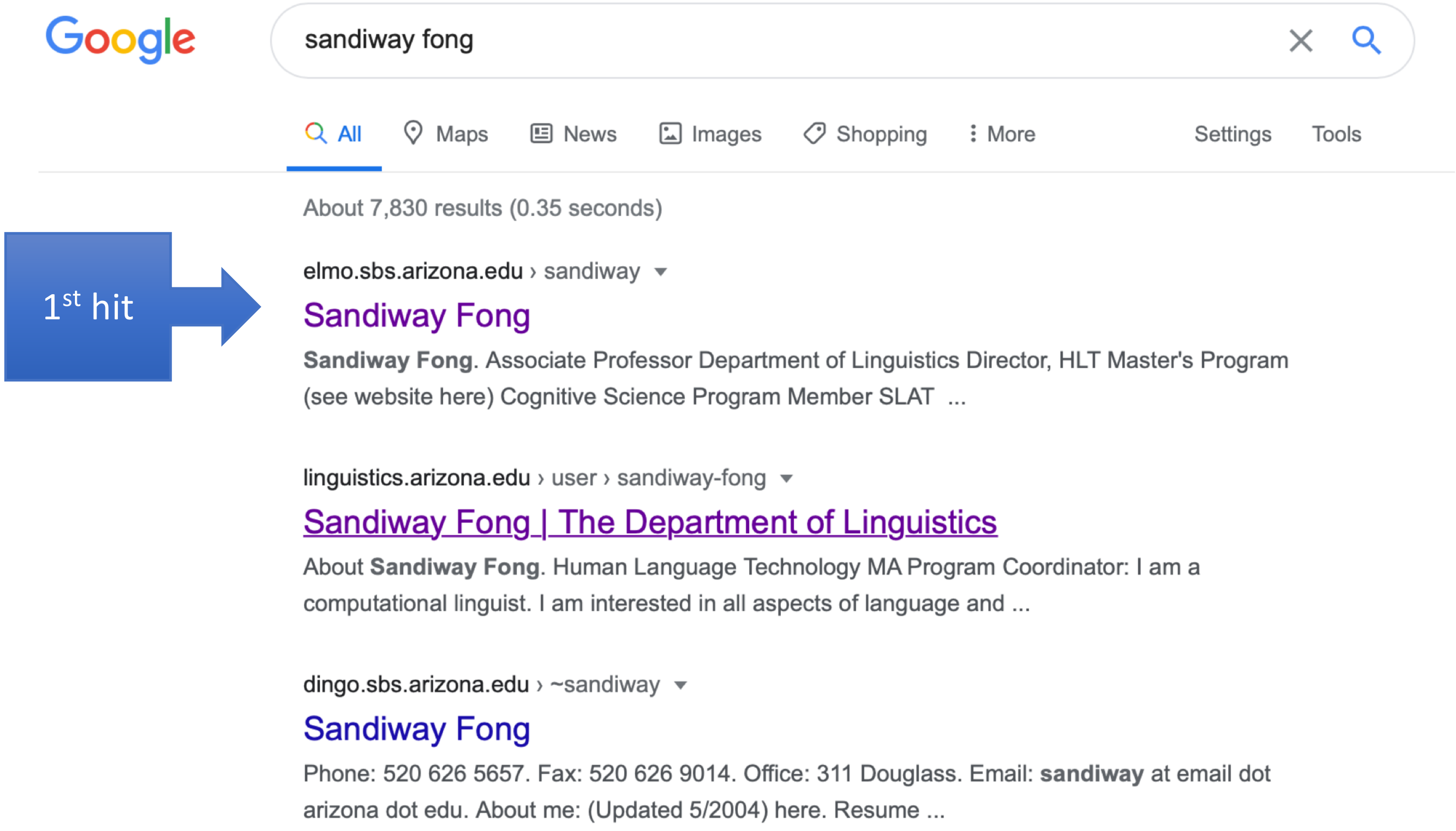
Syllabus

- Questions?

Course website

- Download lecture slides from my homepage
  - http://elmo.sbs.arizona.edu/~sandiway/#courses
  - available from just before class time
    - (afterwards, please look again for updates and corrections)
  - in .pptx (good for animations) and .pdf formats

Course website



1st hit

About 7,830 results (0.35 seconds)

elmo.sbs.arizona.edu › sandiway

Sandiway Fong

Sandiway Fong. Associate Professor Department of Linguistics Director, HLT Master's Program (see website here) Cognitive Science Program Member SLAT ...

linguistics.arizona.edu › user › sandiway-fong

Sandiway Fong | The Department of Linguistics

About **Sandiway Fong**. Human Language Technology MA Program Coordinator: I am a computational linguist. I am interested in all aspects of language and ...

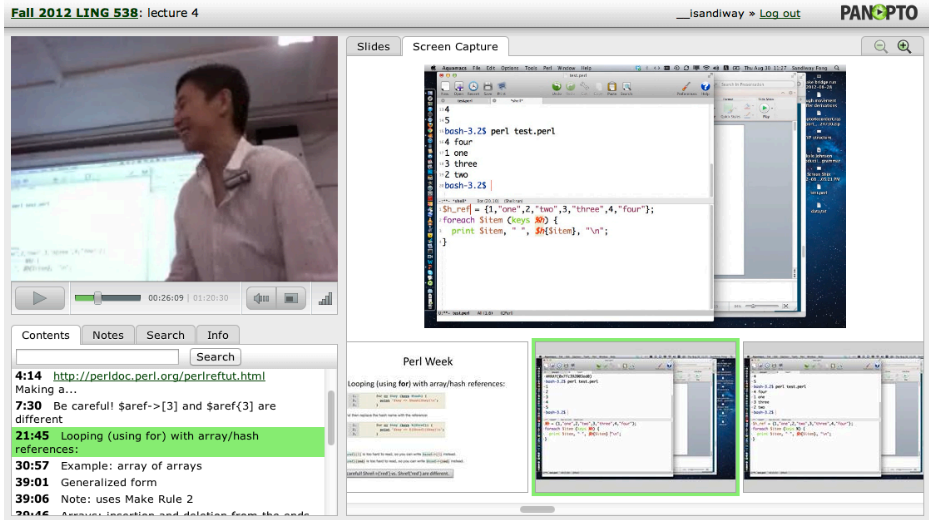
dingo.sbs.arizona.edu › ~sandiway

Sandiway Fong

Phone: 520 626 5657. Fax: 520 626 9014. Office: 311 Douglass. Email: **sandiway** at email dot arizona dot edu. About me: (Updated 5/2004) here. Resume ...

Panopto

- Lectures will be recorded using the panopto system
  - accessible via the course webpage and your browser
  - <span style="color:red">**sometimes crashes**</span>
  - (video, laptop screen, synchronized slides, keyword search)



Fall 2012 LING 538: lecture 4

__isandiway » Log out

```
4
5
bash-3.2$ perl test.perl
4 four
1 one
3 three
2 two
bash-3.2$
```

```
$h_ref = {1,"one",2,"two",3,"three",4,"four"};
foreach $item (keys %h) {
  print $item, " ", $h{$item}, "\n";
}
```

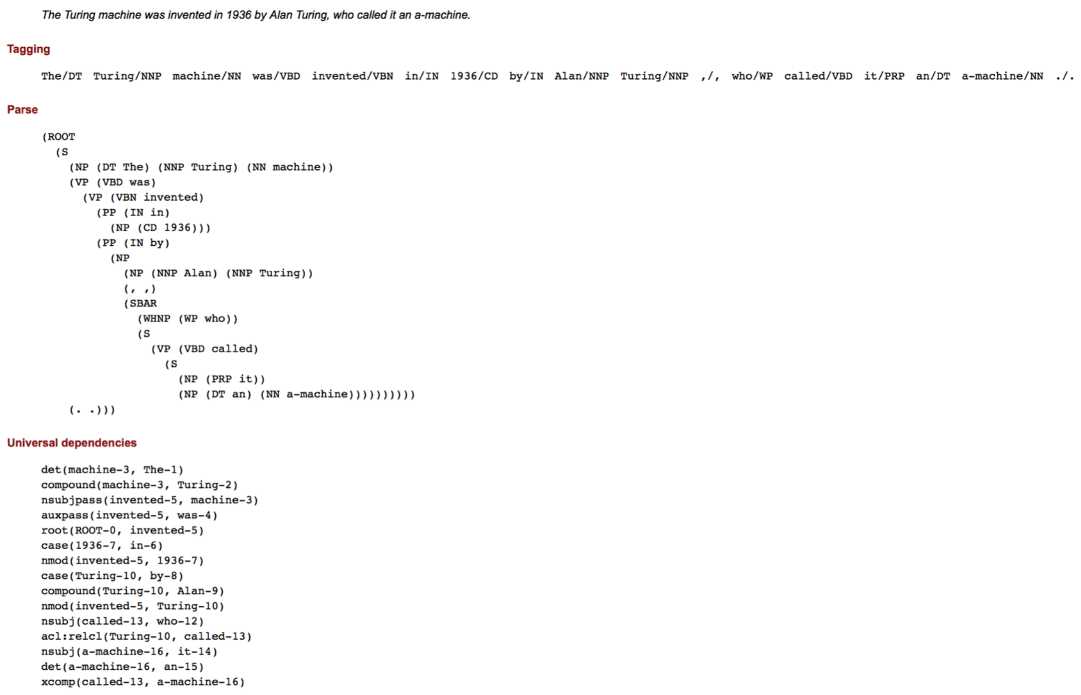
Perl Week

Looping (using **for**) with array/hash references:

4:14 http://perldoc.perl.org/perlreftut.html Making a...
7:30 Be careful! $aref->[3] and $aref{3} are different
21:45 Looping (using for) with array/hash references:
30:57 Example: array of arrays
39:01 Generalized form
39:06 Note: uses Make Rule 2
39:46 Arrays: insertion and deletion from the ends

Example

Stanford Parser: online version
http://nlp.stanford.edu:8080/parser/index.jsp



The Turing machine was invented in 1936 by Alan Turing, who called it an a-machine.

Tagging

```
The/DT  Turing/NNP  machine/NN  was/VBD  invented/VBN  in/IN  1936/CD  by/IN  Alan/NNP  Turing/NNP  ,/,  who/WP  called/VBD  it/PRP  an/DT  a-machine/NN  ./.
```

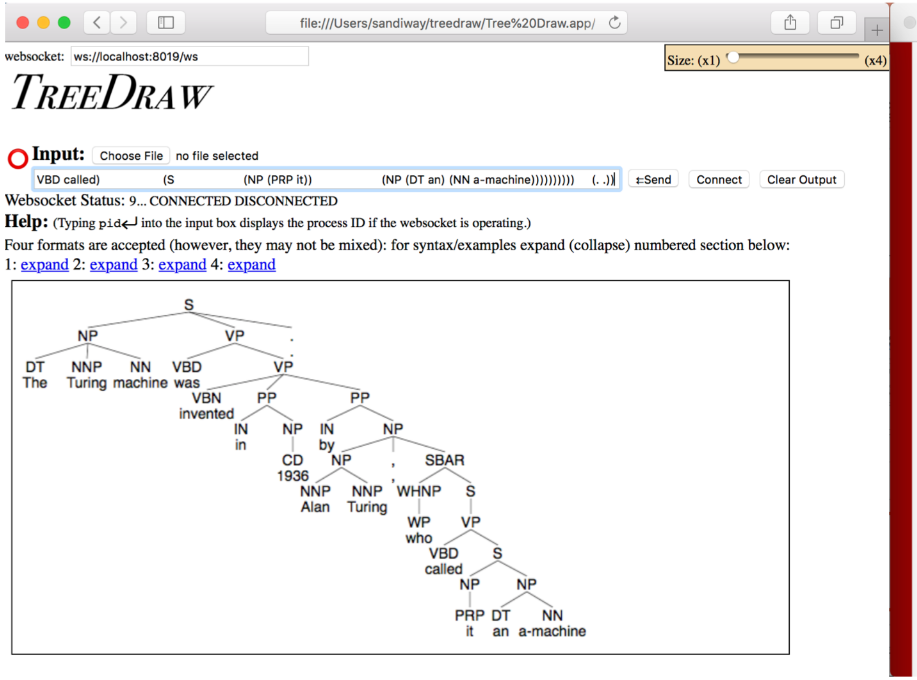
Parse

```
(ROOT
  (S
    (NP (DT The) (NNP Turing) (NN machine))
    (VP (VBD was)
      (VP (VBN invented)
        (PP (IN in)
          (NP (CD 1936)))
        (PP (IN by)
          (NP
            (NP (NNP Alan) (NNP Turing))
            (, ,)
            (SBAR
              (WHNP (WP who))
              (S
                (VP (VBD called)
                  (S
                    (NP (PRP it))
                    (NP (DT an) (NN a-machine))))))))))
    (. .)))
```

Universal dependencies

```
det(machine-3, The-1)
compound(machine-3, Turing-2)
nsubjpass(invented-5, machine-3)
auxpass(invented-5, was-4)
root(ROOT-0, invented-5)
case(1936-7, in-6)
nmod(invented-5, 1936-7)
case(Turing-10, by-8)
compound(Turing-10, Alan-9)
nmod(invented-5, Turing-10)
nsubj(called-13, who-12)
acl:relcl(Turing-10, called-13)
nsubj(a-machine-16, it-14)
det(a-machine-16, an-15)
xcomp(called-13, a-machine-16)
```

Example



websocket: ws://localhost:8019/ws

TREEDRAW

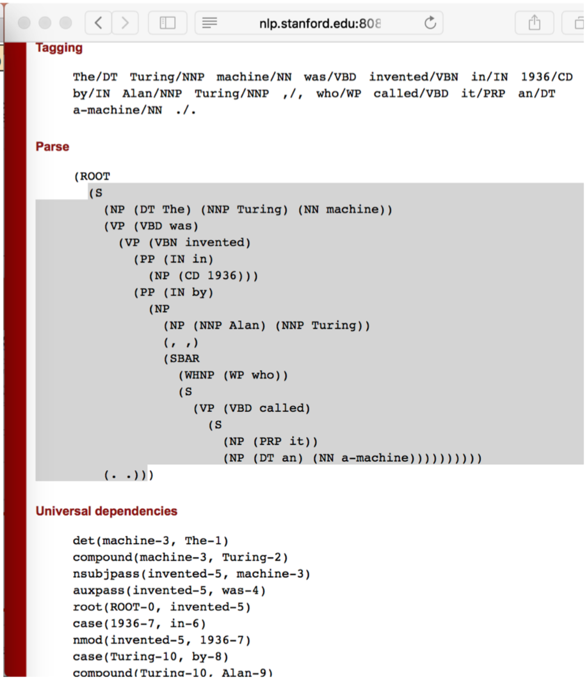
Input: no file selected

VBD called)   (S   (NP (PRP it))   (NP (DT an) (NN a-machine)))))))))   (. .))

Websocket Status: 9... CONNECTED DISCONNECTED

Help: (Typing `pid`↵ into the input box displays the process ID if the websocket is operating.)

Four formats are accepted (however, they may not be mixed): for syntax/examples expand (collapse) numbered section below:
1: expand 2: expand 3: expand 4: expand



Tagging

```
The/DT  Turing/NNP  machine/NN  was/VBD  invented/VBN  in/IN  1936/CD
by/IN  Alan/NNP  Turing/NNP  ,/,  who/WP  called/VBD  it/PRP  an/DT
a-machine/NN  ./.
```

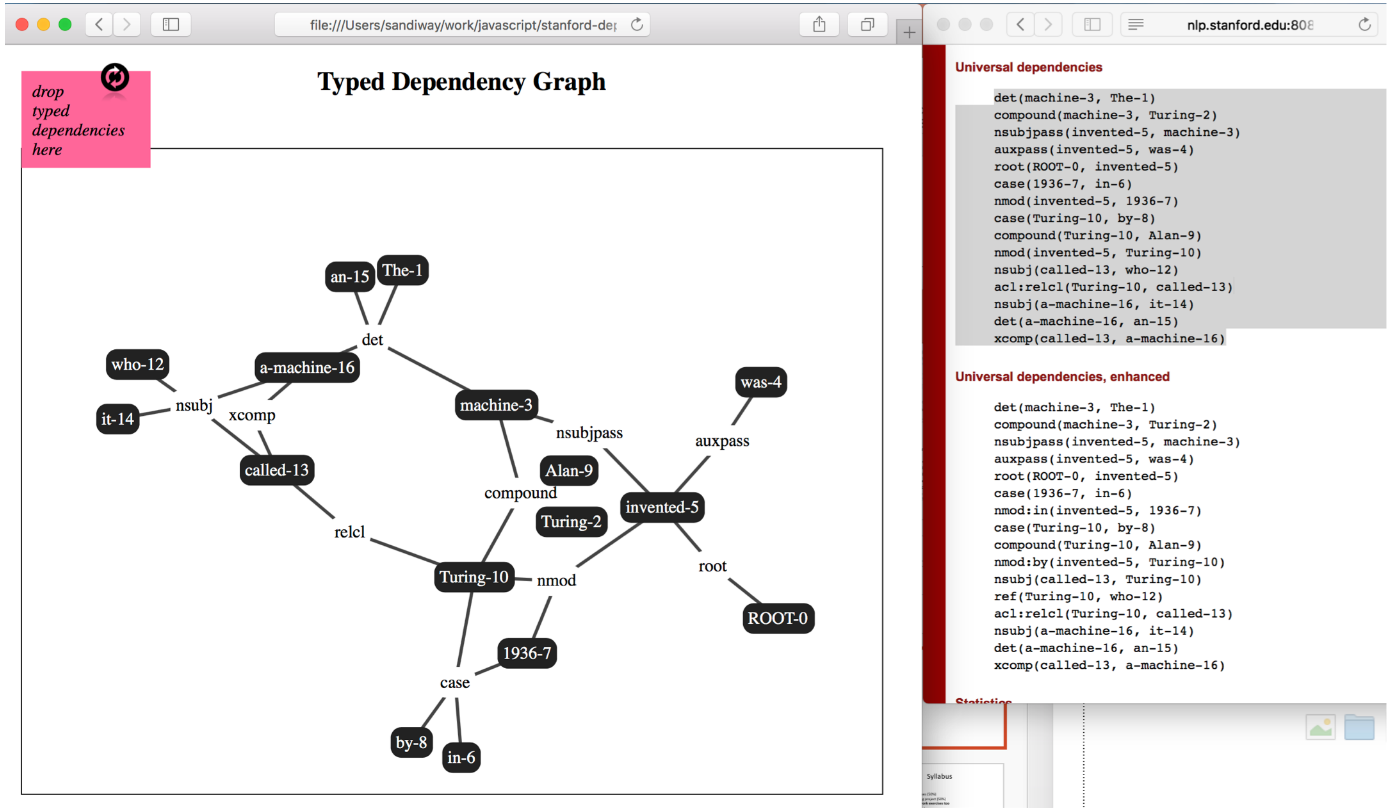
Parse

```
(ROOT
  (S
    (NP (DT The) (NNP Turing) (NN machine))
    (VP (VBD was)
      (VP (VBN invented)
        (PP (IN in)
          (NP (CD 1936)))
        (PP (IN by)
          (NP
            (NP (NNP Alan) (NNP Turing))
            (, ,)
            (SBAR
              (WHNP (WP who))
              (S
                (VP (VBD called)
                  (S
                    (NP (PRP it))
                    (NP (DT an) (NN a-machine)))))))))))
    (. .)))
```

Universal dependencies

```
det(machine-3, The-1)
compound(machine-3, Turing-2)
nsubjpass(invented-5, machine-3)
auxpass(invented-5, was-4)
root(ROOT-0, invented-5)
case(1936-7, in-6)
nmod(invented-5, 1936-7)
case(Turing-10, by-8)
compound(Turing-10, Alan-9)
```

Example



Universal dependencies

```
det(machine-3, The-1)
compound(machine-3, Turing-2)
nsubjpass(invented-5, machine-3)
auxpass(invented-5, was-4)
root(ROOT-0, invented-5)
case(1936-7, in-6)
nmod(invented-5, 1936-7)
case(Turing-10, by-8)
compound(Turing-10, Alan-9)
nmod(invented-5, Turing-10)
nsubj(called-13, who-12)
acl:relcl(Turing-10, called-13)
nsubj(a-machine-16, it-14)
det(a-machine-16, an-15)
xcomp(called-13, a-machine-16)
```

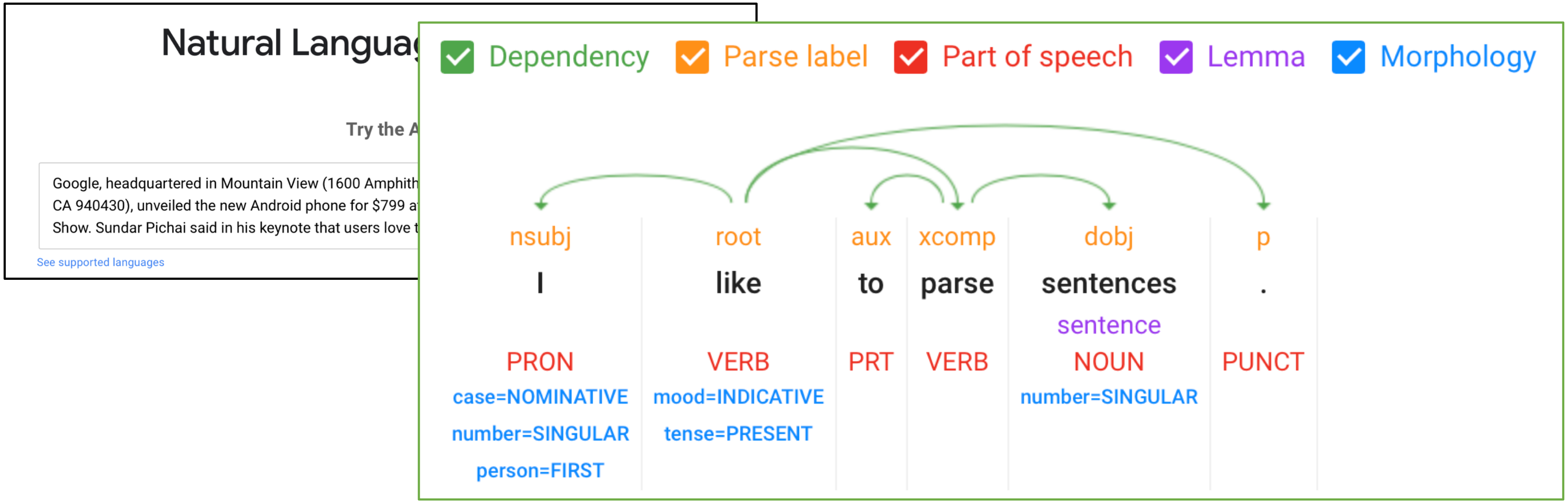
Universal dependencies, enhanced

```
det(machine-3, The-1)
compound(machine-3, Turing-2)
nsubjpass(invented-5, machine-3)
auxpass(invented-5, was-4)
root(ROOT-0, invented-5)
case(1936-7, in-6)
nmod:in(invented-5, 1936-7)
case(Turing-10, by-8)
compound(Turing-10, Alan-9)
nmod:by(invented-5, Turing-10)
nsubj(called-13, Turing-10)
ref(Turing-10, who-12)
acl:relcl(Turing-10, called-13)
nsubj(a-machine-16, it-14)
det(a-machine-16, an-15)
xcomp(called-13, a-machine-16)
```

Statistics

Syntactic Parsing

- Google Natural Language
- https://cloud.google.com/natural-language/



Natural Languag

Try the A

Google, headquartered in Mountain View (1600 Amphith
CA 940430), unveiled the new Android phone for $799 a
Show. Sundar Pichai said in his keynote that users love t

See supported languages

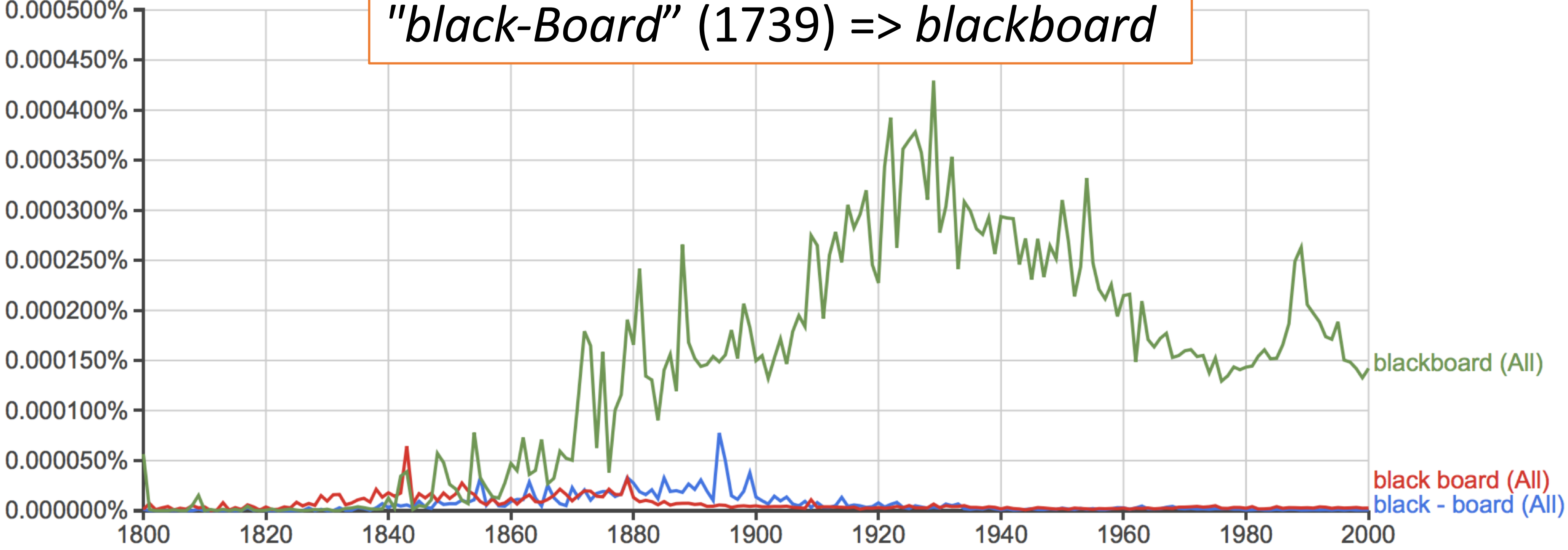
☑ Dependency ☑ Parse label ☑ Part of speech ☑ Lemma ☑ Morphology

nsubj	root	aux	xcomp	dobj	p
I	like	to	parse	sentences	.
				sentence	
PRON	VERB	PRT	VERB	NOUN	PUNCT
case=NOMINATIVE number=SINGULAR person=FIRST	mood=INDICATIVE tense=PRESENT			number=SINGULAR	

Google n-grams

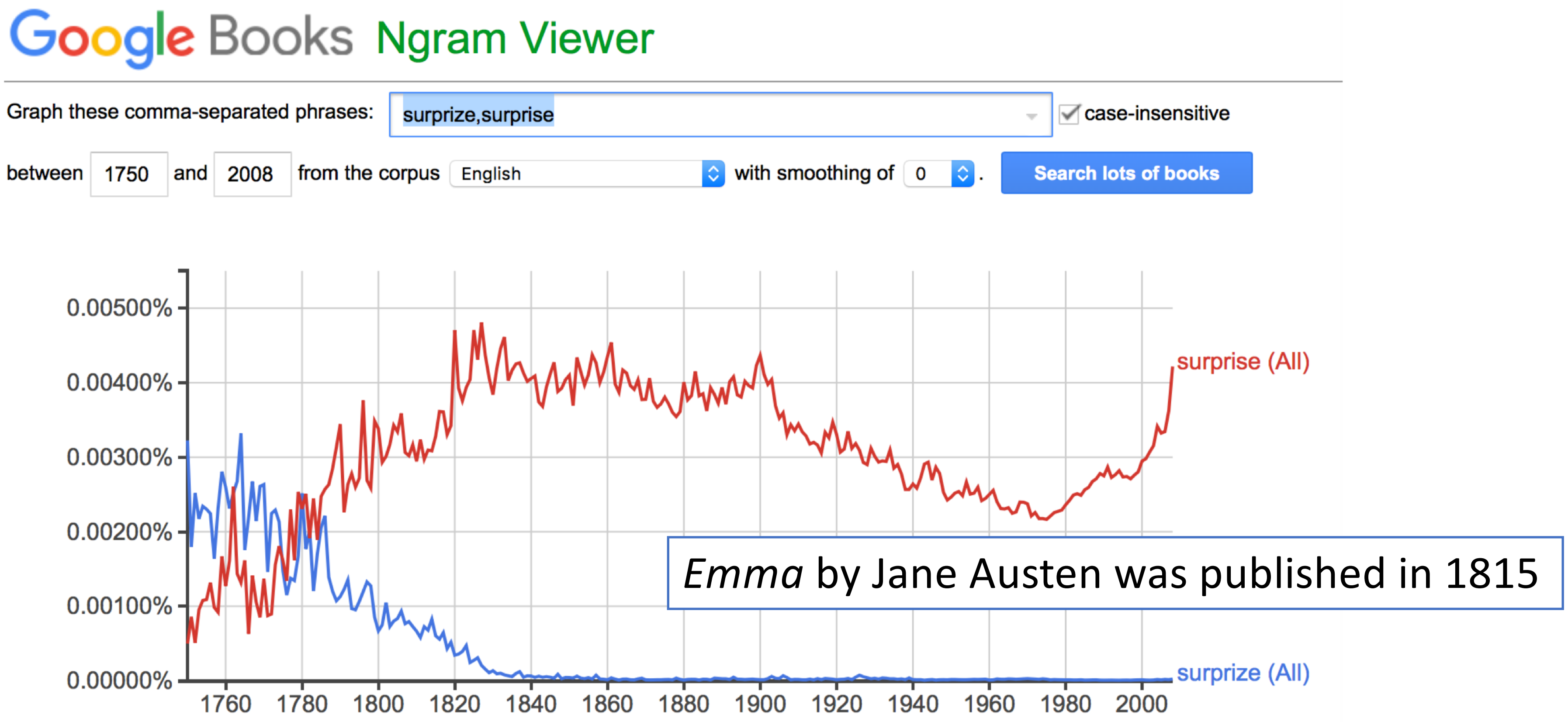
Staged word compound formation?

"black-Board" (1739) => *blackboard*



https://books.google.com/ngrams

Google: relative frequency of two spellings



Google Books Ngram Viewer

Graph these comma-separated phrases: surprize,surprise ☑ case-insensitive

between 1750 and 2008 from the corpus English with smoothing of 0 .

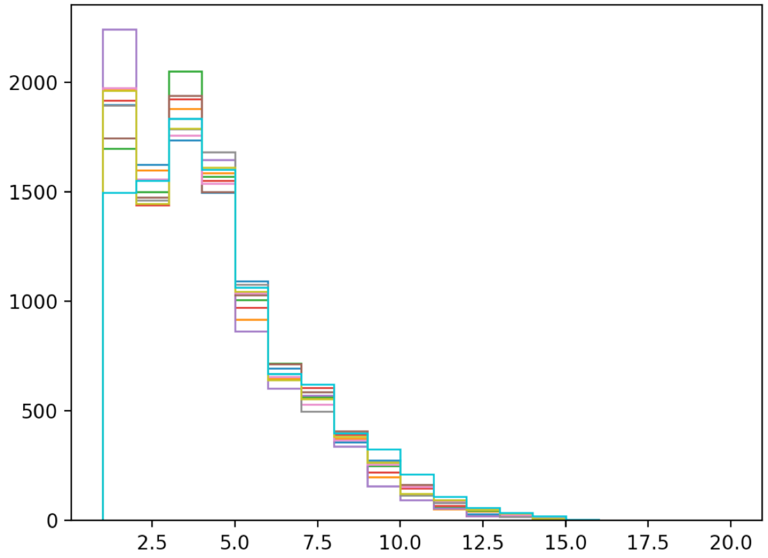
Search lots of books

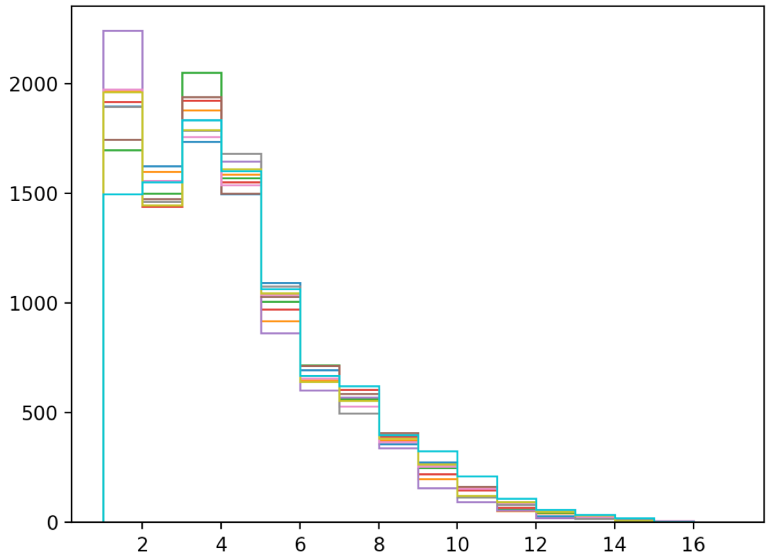
Emma by Jane Austen was published in 1815

Stylometry: compare word length distribution

```
len1s = [len1[i*10000:i*10000+10000] for i in range(10)]
for l in len1s:
    plt.hist(l, bins=np.arange(min(l),max(l)+1), histtype='step')
plt.show()
```

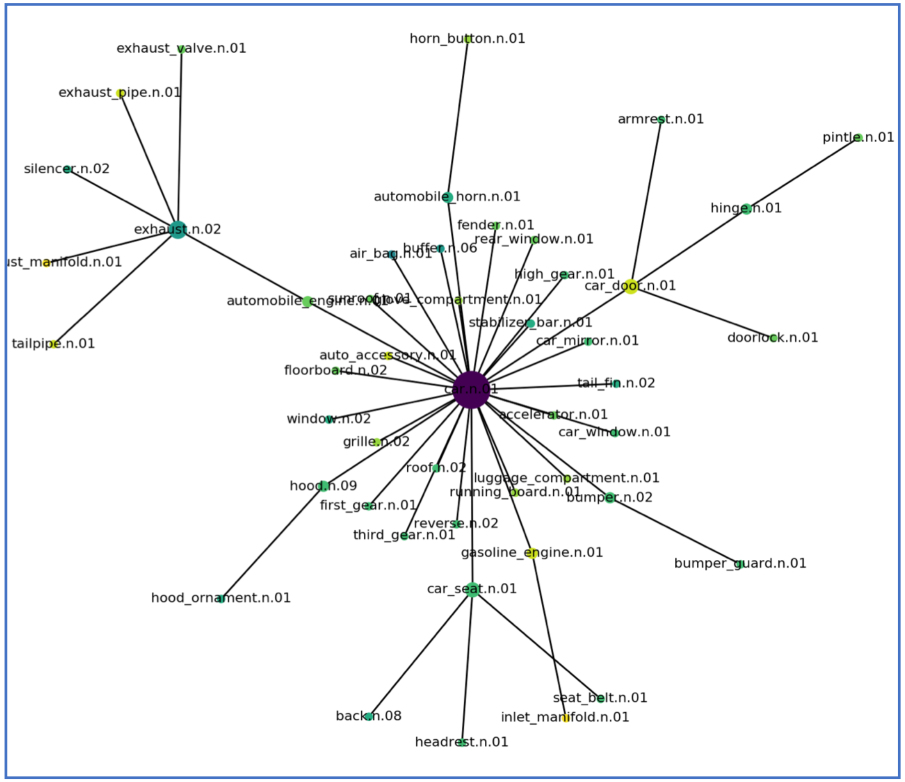
Forensic linguistics





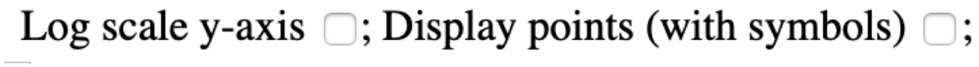
WordNet relations: parts of a car

```
from nltk.corpus import wordnet as wn
c = wn.synset('car.n.01')
g = graph(c, 'part_meronyms')
graph_draw(g)
```



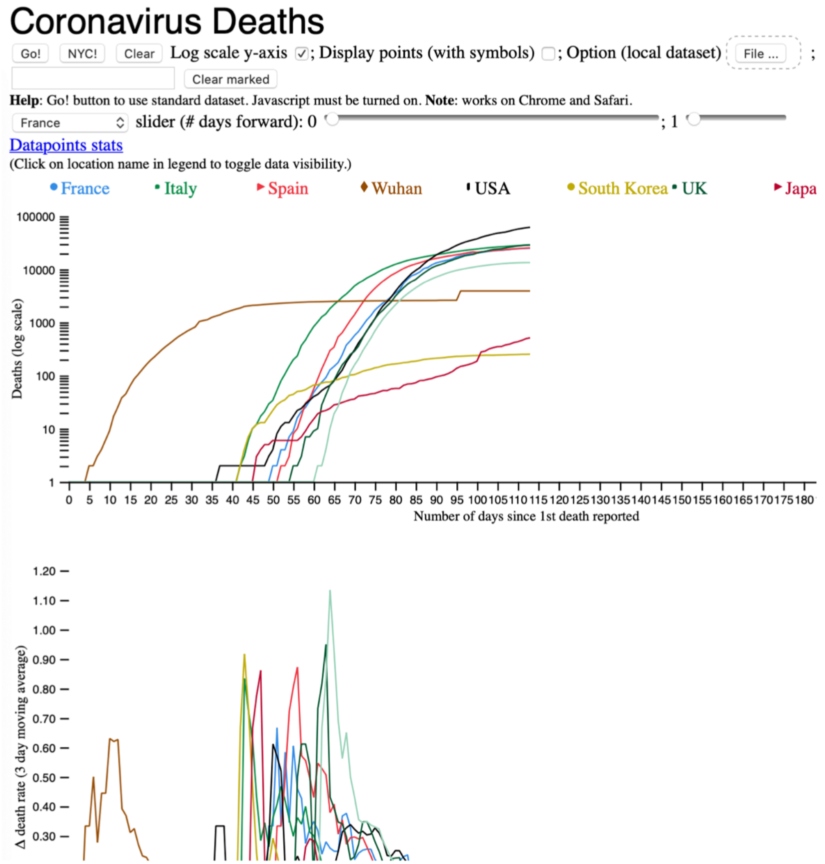
Browser language: Javascript

- An example:
  - http://elmo.sbs.arizona.edu/sandiway/lockdown/index.html
  - Toggle the Log scale y-axis check



Log scale y-axis □; Display points (with symbols) □;

  - Redraws in real-time on **YOUR** computer



Coronavirus Deaths

Go! NYC! Clear Log scale y-axis ☑; Display points (with symbols) □; Option (local dataset) File ... ;

Clear marked

Help: Go! button to use standard dataset. Javascript must be turned on. **Note**: works on Chrome and Safari.

France slider (# days forward): 0 ; 1

Datapoints stats

(Click on location name in legend to toggle data visibility.)

Introduction

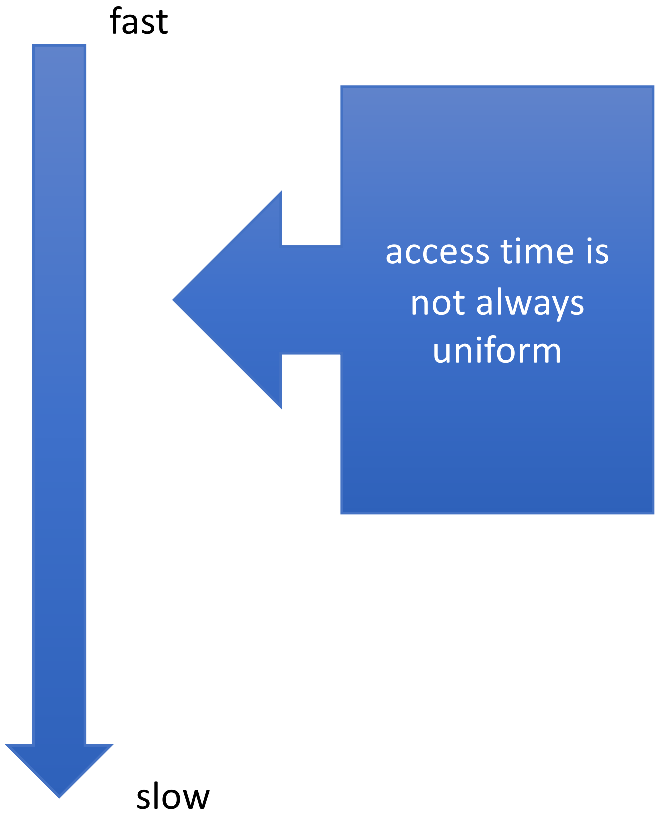
- Computers
  - Memory
    - Programs and data
  - CPU
    - Interprets machine instructions
  - I/O
    - keyboard, mouse, touchpad, screen, touch sensitive screen, printer, usb port, etc.
    - bluetooth, ethernet, wifi, cellular …

Introduction

- Memory hierarchy
  - CPU registers
  - L1/L2 cache
  - L3 cache
  - RAM (sometimes NUMA)
  - SSD/hard drive
  - blu ray/dvd/cd drive
  - LAN
  - Internet

invisible to programmers

open file read/write

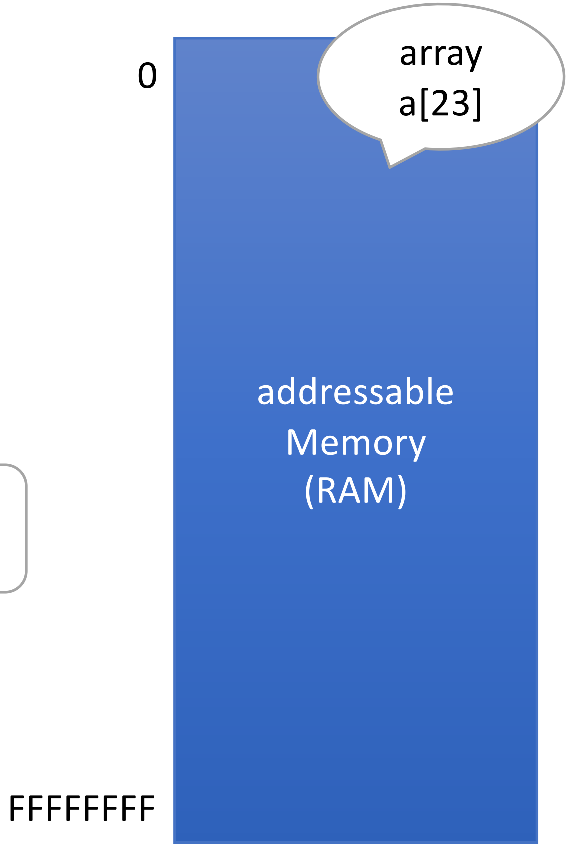


access time is not always uniform

Introduction

- Memory Representation (Revealed)
  - binary: zeros and ones (1 bit)
  - organized into **bytes** (8 bits)
    - memory is byte-addressable
  - **word** (32 bits)
    - e.g. integer
    - (64 bits: floating point number)
  - big-endian/little-endian
    - most significant byte first or least significant byte
    - communication …

your Intel and ARM CPUs



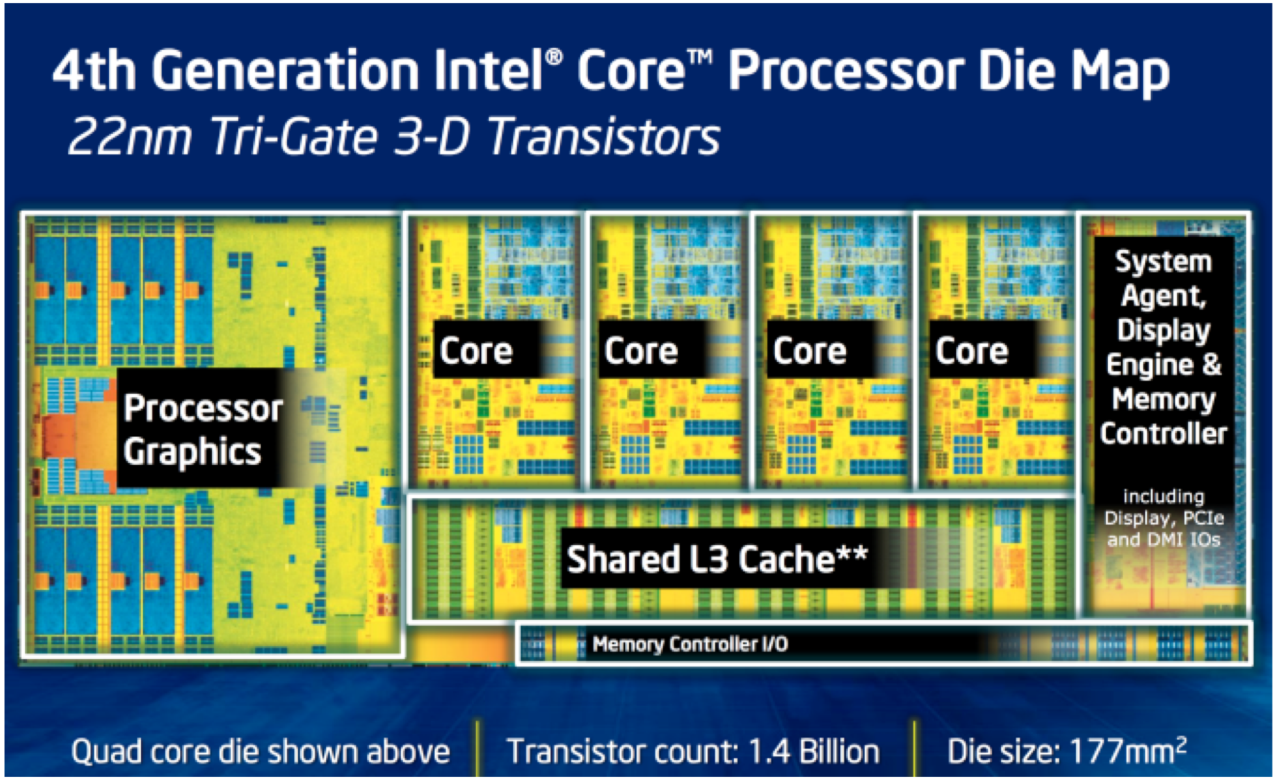
Introduction

- A typical notebook computer (this one)
  - **a 2018 MacBook Pro 15**
  - CPU: Intel® Core™ i9-8950HK
    - approx. $500
    - <7, >3.5 billion transistors (14nm lithography)
    - built-in GPU: Intel® UHD Graphics 630
    - TDP: **45W** (2.9 GHz)
    - Hexa-core (Max Turbo: 4.8 GHz)
    - Hyper-Threaded (12 logical CPUs, 6 physical)
    - 64 bit processor
    - 64 KB (32 KB Instruction + 32 KB Data) L1 cache per core
    - 256KB L2 cache per core
    - 12MB L3 cache **shared**
    - 32GB max RAM (64GB design)

Increased address space and 64-bit registers

Human brain:
86 billion neurons
Herculano-Houzel S. The Human Brain in Numbers: A Linearly Scaled-up Primate Brain . *Frontiers in Human Neuroscience*. 2009;3:31. doi:10.3389/neuro.09.031.2009.

Introduction


4th Generation Intel® Core™ Processor Die Map
22nm Tri-Gate 3-D Transistors
Processor Graphics
Core
Core
Core
Core
System Agent, Display Engine & Memory Controller
including Display, PCIe and DMI IOs
Shared L3 Cache**
Memory Controller I/O
Quad core die shown above
Transistor count: 1.4 Billion
Die size: 177mm²

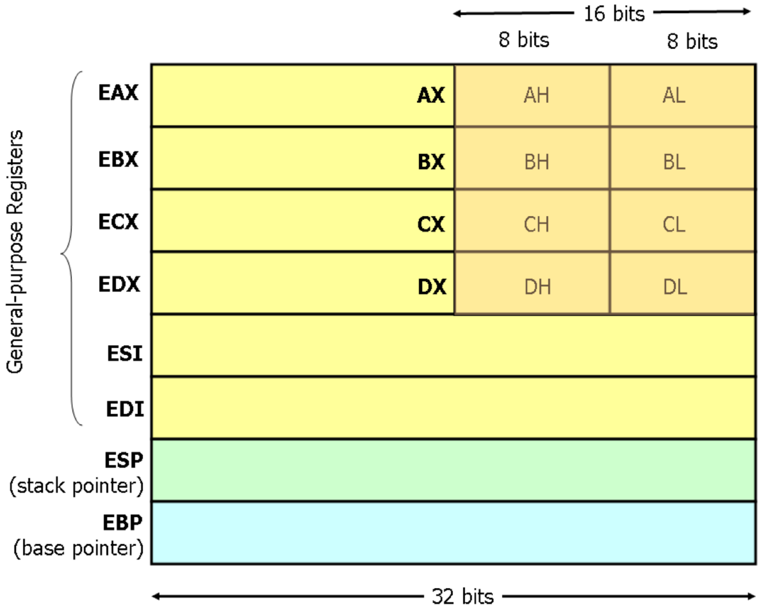
A 4 core machine: 8 virtual

anandtech.com

Introduction

- Machine Language
  - A CPU understands only one language: machine language
    - **all other languages** must be translated into machine language
  - Primitive instructions include:
    - MOV
    - PUSH
    - POP
    - ADD / SUB
    - INC / DEC
    - IMUL / IDIV
    - AND / OR / XOR / NOT
    - NEG
    - SHL / SHR
    - JMP
    - CMP
    - JE / JNE / JZ / JG / JGE / JL / JLE
    - CALL / RET

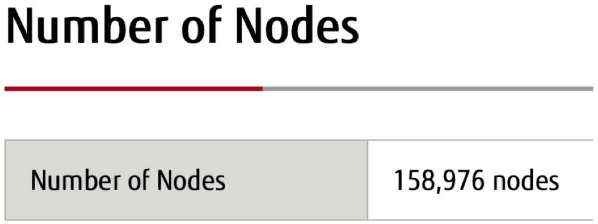
Assembly Language: (this notation)
by definition, nothing built on it is more powerful



http://www.cs.virginia.edu/~evans/cs216/guides/x86.html

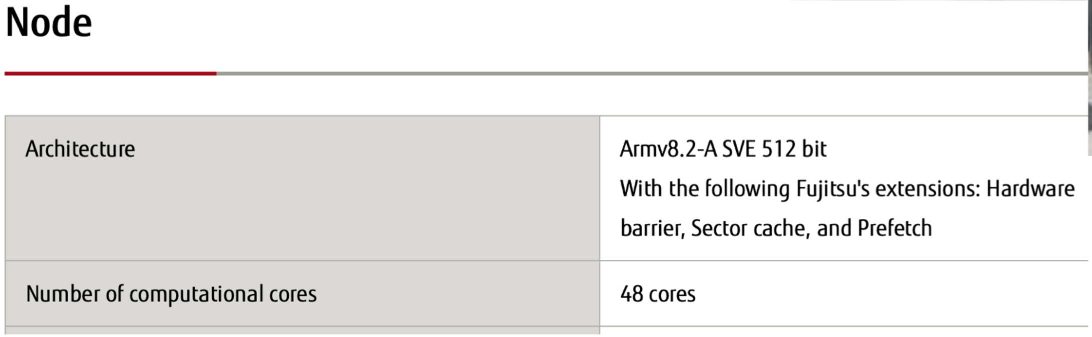
Fugaku supercomputer

- World's fastest computer (ARM instruction set)
- Power consumption of 30 to 40 MWatts
- https://www.fujitsu.com/global/about/innovation/fugaku/



Number of Nodes

Number of Nodes	158,976 nodes



Node

Architecture	Armv8.2-A SVE 512 bit With the following Fujitsu's extensions: Hardware barrier, Sector cache, and Prefetch
Number of computational cores	48 cores