

## 15-110 Week11-14 Notes Sheet

### Parallel Programming

*Concurrency:* running multiple things at the same time on a computer

*CPU:* a piece of hardware that runs the actions taken by a program

*Parallel Programming:* running the same program on multiple CPUs concurrently

*Scheduler:* the computer part that decides who gets to run next on a CPU

*Throughput:* the amount of work a computer can do in a length of time

*Multitasking:* alternate rapidly between programs on a single CPU to simulate concurrency

*Multiprocessing:* run multiple programs concurrently on multiple CPUs on the same computer

*Deadlock:* occurs when 2+ programs are both waiting on resources that one of the other programs in the group already has

*Total steps:* the actual number of steps taken in a concurrent process

*Time steps:* the number of concurrent steps taken by a concurrent process

*Pipelining:* make an algorithm concurrent by splitting the process into steps. Each step is assigned to one worker (CPU), data is passed between CPUs.

### Distributed Computing and the Internet

*Distributed Computing:* run multiple programs concurrently on multiple computers that are connected together

*Cloud Computing:* using distributed computing to store data/services online

*MapReduce:* make an algorithm concurrent by splitting the data into many smaller datasets.

*Mapper:* part of MapReduce. Takes a small piece of data, processes it, and returns a result.

*Reducer:* part of MapReduce. Takes a collection of results and processes them into the final result.

*Manager:* part of MapReduce. The manager moves data through the system and outputs the final result.

*Browser:* program that receives data from the internet and displays it as a webpage

*Router:* a device that can send data to other machines connected to it via cables. Routers form the core of the internet.

*ISP:* a service that connects individuals' computers to the internet

*IP Address:* a series of numbers that forms the 'real name' of a computer. Can be used to find a computer on the internet

*DNS Server:* a computer on the internet that can map URLs to IP Addresses.

*Protocol:* a standard for communicating information between machines. Examples include HTML and HTTP.

*Packet:* a small piece of data that is sent from one computer to another across the internet. Has sender, receiver, and data.

## 15-110 Week11-14 Notes Sheet

### Fault Tolerance and Security

*Fault tolerance:* the internet is designed to be error-tolerant by being decentralized with many redundancies

*Bottleneck:* a single point where the communication process can be controlled

*Net Neutrality:* a policy stating that all internet traffic should be treated equally

*Data Privacy:* we may want to have control over who has access to our data and what others do with it

*Data Security:* we may want to keep some communications accessible only by the sender and receiver

*DDOS:* a security attack where a server is overwhelmed by a large number of messages, which blocks regular traffic

*Man-in-the-middle:* a security attack where a router intercepts data that passes through it to read and/or change it

*Authentication:* a process to verify someone's identity. Can be done with passwords and certificates.

*Encryption:* a process to encode data so that only the sender and receiver can read it. Original messages are plaintext; encrypted messages are ciphertext.

*Password:* a piece of information that shows you are who you say you are

*Certificate:* a piece of information that shows an organization is who it says it is

*Plaintext:* a regular message

*Ciphertext:* a message that has been encoded using encryption

*Key:* a piece of information used to encode a message. Keys can be symmetric (known by both parties) or asymmetric (split into public/private parts)

*Encode/Encrypt:* take plaintext and change it with a key into ciphertext.

*Decode/Decrypt:* take ciphertext and change it back to plaintext using the key.

*Break:* an attempt to decipher plaintext out of ciphertext without the key

*Caesar Cipher:* an encryption algorithm where you shift each letter by a set amount. The shift amount is the key.

*RSA:* an encryption algorithm where you raise the message to a power  $e$  and mod by  $n$  to encrypt, then raise it to a power  $d$  and mod by  $n$  to decrypt. Uses asymmetric keys. Public key is  $(e, n)$ ; private key is  $(d, n)$ .

### Large Code Projects & Libraries

```
f = open(fname, mode) # open file
f.read() # read contents as string
f.readlines() # read list of lines
f.write(text) # write text to file
f.close() # close file when done
```

*External library:* a library outside of the main Python language that can be installed into Python.

*Documentation:* instructions on how to use a library available online. Describes existing functions and what they do.

*Helper functions:* when given a complicated task, break it into subtasks and assign each subtask to a separate function to simplify the program.

## 15-110 Week11-14 Notes Sheet

### Data Analysis (I and II)

*Data Analysis*: gaining insights about data using computation.

*Categorical*: a type of data that falls into multiple separate categories

*Ordinal*: a type of data that falls into categories which can be compared

*Numerical*: a type of data that can be represented by numbers

*CSV*: data stored in a table-like format, separated by newlines and commas

*JSON*: data stored in a particular nested format similar to a dictionary

*Plaintext*: data that does not match a known protocol but can be read directly

```
import csv # CSV library
csv.reader(f) # read f into data
list(reader) # data -> 2D list
csv.writer(f) # write data as CSV
writer.writerows(data)
# actually write the data
```

```
import json # JSON library
json.load(f) # read file into data
json.dump(data, f) # write data
```

You can extract data from plaintext using *string operations and methods* like slicing, `split`, `index`, and `strip`.

You can *reformat* data by adding, removing, and reinterpreting existing data using destructive actions like `append`, `pop`, and index assignment.

```
import statistics # stats library
statistics.mean(data) # mean
statistics.median(data) # median
statistics.mode(data) # mode
```

Calculate probabilities over data using `count` and `len`

*Visualization*: representing data in a visual format. Plot type can be chosen based on the number of dimensions of data (one, two, or three) and the data types being used (categorical, ordinal, numerical).

*Visualization options*: bar chart, box-and-whiskers plot, bubble plot, colored scatter plot, histogram, pie chart, scatter plot, scatter plot matrix

*Matplotlib*: a library that enables building visualizations. Plots can be found by searching the APIs and examples.

```
import matplotlib.pyplot as plt
plt.scatter(x, y) # create plot
plt.show() # render visualization
```

## 15-110 Week11-14 Notes Sheet

### Simulation (I and II)

*Simulation*: an automated imitation of a real-world event

*Model*: a computational representation of the real world

*Components*: information in a model that describes the state of the world

*Rules*: information in a model that describes how it changes over time or due to events

*MVC (Model-View-Controller)*: a framework for programming simulations where functions work in tandem using a shared data structure instead of running sequentially. Store components in the *model*; update graphics from the *view*; call rule functions from the *controllers*.

```
# set up initial model
# data["var"] = value
makeModel(data)

# display current model
# use data["var"] in canvas call
makeView(data, canvas)

# update model over time
# update data["var"] each call
runRules(data, call)

# update model due to key event
# check event.char, event.keysym
keyPressed(data, event)

# update model due to mouse event
# check event.x, event.y
mousePressed(data, event)
```

*True randomness*: randomly generated numbers that are impossible to predict in a way that allows a winner in the long run

*Pseudo-randomness*: numbers generated randomly by an algorithm. Can be predicted if you know the algorithm.

*Monte Carlo method*: solve a problem by running a simulation many many times and averaging the results

```
# Monte Carlo structure
def getExpectedValue(numTrials):
    count = 0
    for trial in range(numTrials):
        result = runTrial()
        if result == True:
            count = count + 1
    return count / numTrials
```

## 15-110 Week11-14 Notes Sheet

### Machine Learning (I and II)

*Machine Learning:* algorithmically find patterns in data and automatically develop a model for the data based on them.

*Supervised learning:* learn from *labeled* data to predict label outputs based on the given information. Can be used to make predictions on future data.

*Unsupervised learning:* group *unlabeled* data into categories by finding data points that are similar to each other. Helps find natural structures, but hard to test.

*Reinforcement learning:* help an *AI agent* solve a goal by repeatedly checking whether the agent is closer too or further from the goal.

*Classification:* given labeled data, produce a model that can find *categorical or ordinal* results.

*Regression:* given labeled data, produce a model that can find *numerical* results.

*Clustering:* given unlabeled data, group similar data points into separate *clusters*.

*Training:* use most of the original dataset to look for groups of features that correctly predict results. Build these features in a model to make future predictions.

*Validation:* while training, repeatedly test on a subset of the data to make sure the model isn't overfitted to the known data

*Testing:* when the model is finished, test it once on a reserved subset of the data to see how well it performs. Since the model has never seen this data before, it should perform similarly on new, unlabeled data.

*Artificial Intelligence (AI):* computational techniques that attempt to mimic signs up human intelligence using programming

*Agent:* a model trained by an AI algorithm to accomplish a specific task. Works through perception, reason, action cycle.

*Perception:* gather information about the problem being solved

*Reason:* given the current information, decide what should be done next, often using search algorithms

*Action:* take the chosen action to move closer to the goal

*Game Tree:* a tree that represents all the possible states of a two-player game. Nodes are game states (possible board configurations), edges are actions taken by players.

*Minimax:* an algorithm that can be applied to a game tree to help the AI choose the best possible move. Scores leaves based on whether the AI wins (1) or loses (-1); for inner nodes, chooses the maximum child score if the AI is taking the turn, or the minimum score if the user is taking the turn.

*Heuristic:* heuristics can be applied to game trees by only moving down a certain number of levels, then scoring inner nodes with a heuristic function instead of creating the full game tree.

## 15-110 Week11-14 Notes Sheet

### Computer Science History

#### *Introduction of theoretical computers*

- Analytical Engine
- Program over Bernoulli numbers
- Church-Turing Thesis

#### *Construction of first hardware & software*

- Electronic circuitry
- WWII impact
- ENIAC
- von Neumann architecture
- compiler

#### *Transition from corporate to personal*

- Transistor
- Integrated circuit
- Microprocessor
- Mother of All Demos

#### *Connection of computers via internet*

- ARPANET
- TCP/IP and HTML

### Computer Science Ethics

*Data Collection:* data is collected from the user, the user's browser, and from other sources like tracking cookies. This data is used to provide hyper-targeted ads.

*Ethics:* there are debates over what kinds of data should be protected.

*Notice-and-choice Model:* some countries try to regulate data collection by requiring companies to notify the consumer when data is being collected and have the consumer choose to agree. The effectiveness of this is questionable.

*Automated Decision Making:* some AI algorithms are used to make important decisions. It is often hard to explain why these decisions are reached due to how the models work. Some algorithms have also shown bias across race and gender.

*Ethics:* there are debates over who should be held responsible for decisions made by algorithms, and what should be done about algorithms causing job loss.

### Computer Science Future

*Add your own notes here based on the topics that are selected for the lecture*