

# DSC 102 Systems for Scalable Analytics

Rod Albuyeh

Topic 1: Basics of Machine Resources Part 1: Computer Organization

Ch. 1, 2.1-2.3, 2.12, 4.1, and 5.1-5.5 of CompOrg Book

# DSC 102 Systems for Scalable Analytics

Logistics:

PA0 update

PA group sign ups

"Look under the hood" videos

# Survey Says...

What do you want to learn from this course?
AWS, Spark, job applicable skills, ambivalent...
Experience with cluster + cloud computing?
Mostly none.

Linux + Shell scripting?

Mostly none.

Anything else you'd like us to know?

Excitement, intimidation, first 8am class

## Outline



- Digital Representation of Data
  - Processors and Memory Hierarchy
- Basics of Operating Systems
  - Process Management: Virtualization; Concurrency
  - Filesystem and Data Files
  - Main Memory Management
- Persistent Data Storage

### (REVIEW) Digital Representation of Data

- The size and interpretation of a data type depends on PL
- A **Byte** (B; 8 bits) is typically the basic unit of data types
- Boolean:
  - Examples in data sci.: Y/N or T/F responses
  - Just 1 bit needed but actual size is almost always 1B, i.e., 7 bits are wasted!
- Integer:
  - Examples in data science: # of friends, age, # oflikes
  - Typically 4 bytes; many variants (short, unsigned, etc.)
  - Java int can represent  $-2^{31}$  to  $(2^{31} 1)$ ;
  - C unsigned int can represent 0 to  $(2^{32} 1)$ ;
  - Python3 *int* is effectively unlimited length (PL magic!)

### (REVIEW) Digital Representation of Data

**Q:** How many unique data items can be represented by 3 bytes?

- Given k bits, we can represent 2<sup>k</sup> unique data items
- ♦ 3 bytes = 24 bits => 2<sup>24</sup> items, i.e., 16,777,216 items
- Common approximation: 2<sup>10</sup> (i.e., 1024) ~ 10<sup>3</sup> (i.e., 1000);
   kibibyte (KiB) = 1024 bytes, vs kilobyte (KB) = 1000 bytes

### **Q:** How many bits are needed to distinguish 97 unique items?

- $\, lpha \,$  For k unique items, invert the exponent to get  $\, \log_2(k) \,$
- But #bits is an integer! So, we only need  $\lceil \log_2(k) \rceil$
- So, we only need the next higher power of 2
- So... 7 bits

### (REVIEW) Digital Representation of Data

#### 1. Given decimal n

- if n is power of 2 (say, 2<sup>k</sup>), put 1 at bit position k; if k=0, stop; else pad with trailing 0s till position 0
- if n is not power of 2, identify the power of 2 just below n (say, 2<sup>k</sup>); #bits is then k; put 1 at position k
- 2. Reset n as n 2<sup>k</sup>; return to Steps 1-2
- 3. Fill remaining positions in between with 0s

	7	6	5	4	3	2	1	0	Position/Exponent of 2
Decimal	128	64	32	16	8	4	2	1	Power of 2
<b>5</b> 10						1	0	1	
<b>47</b> 10			1	0	1	1	1	1	
<b>163</b> 10	1	0	1	0	0	0	1	1	
<b>16</b> 10				1	0	0	0	0	

- Hexadecimal representation is a common stand-in for binary representation; more succinct and readable
  - Base 16 instead of base 2 cuts display length by ~4x
  - Digits are 0, 1, ... 9, A (1010), B, ... F (1510)
  - Each hexadecimal digit represents 4 bits.

Decimal	Binary	Hexadecimal	
510	1012	516	Alternative notations 0xA3 or А3н
<b>47</b> 10	10 11112	<b>2 F</b> 16	
<b>163</b> 10	1010 00112	A 316	
<b>16</b> 10	1 00002	<b>1 0</b> 16	

Let's unpack:

Base 10...

0123456789

Base 2...

0 1

Base-16 Hexadecimal... 0 1 2 3 4 5 6 7 8 9 A B C D E F

10 11 12 13 14 15

#### Float:

- Examples in data sci.: salary, scores, model weights
- IEEE-754 single-precision format is 4B long; double-precision format is 8B long. Single precision is ~ 8 decimal digits. Double precision is ~ 16 decimal digits.
- Java and C float is single; Python float is double!



Using Automatic Mixed Precision for Major Deep Learning Frameworks

### Float:

Standard IEEE format for single (aka binary32):



$$(-1)^0 \times 2^{124-127} \times (1+1 \cdot 2^{-2}) = (1/8) \times (1+(1/4)) = 0.15625$$

(Note: Converting decimal reals/fractions to float is NOT on exams)

Due to representation imprecision issues, floating point arithmetic (addition and multiplication) is not associative!

> (base) rodalbuyeh@Rods-MacBook-Pro ~ % python Python 3.9.12 (main, Apr 5 2022, 01:53:17) [Clang 12.0.0 ] :: Anaconda, Inc. on darwin Type "help", "copyright", "credits" or "license" for more information. >>> 0.1 + 0.2 0.3000000000000004 >>> (0.1 + 0.2) + 0.7 1.0 >>> 0.1 + (0.2 + 0.7) 0.999999999999999999999999

- In binary32, special encodings recognized:
  - Exponent 0xFF and fraction 0 is +/- "Infinity"
  - Exponent 0xFF and fraction <> 0 is "NaN"

- More float standards: double-precision (float64; 8B) and halfprecision (float16; 2B); different #bits for exponent, fraction
- Float16 is now common for *deep learning* parameters:
  - Native support in PyTorch, TensorFlow, etc.; APIs also exist for weight quantization/rounding post training
  - NVIDIA Deep Learning SDK support mixed-precision training; 2-3x speedup with similar accuracy!
- New processor hardware (FPGAs, ASICs, etc.) enable arbitrary precision, even 1-bit (!).

#### Representing Character (char) and String:

- Represents letters, numerals, punctuations, etc.
- A string is typically just a variable-sized array of char
- C char is 1 byte; Java char is 2 bytes; Python does not have a char type (use str or bytes)
- American Standard Code for Information Interchange (ASCII) for encoding characters; initially 7-bit; later extended to 8-bit (1 byte)
  - Examples: 'A' is 65 (dec), 'a' is 97 (dec), '@' is 64 (dec), etc.
- ♦ Unicode UTF-8 is now most common; subsumes ASCII; 4 bytes for ~1.1 million "code points" incl. many other language scripts, math symbols ∜, emojis <sup>(1)</sup>/<sub>2</sub>, etc.

- All digital objects are *collections* of basic data types (bytes, integers, floats, and characters)
  - SQL dates/timestamp: string (w/ known format)
  - ML feature vector: *array* of floats (w/ known length)
  - Neural network weights: set of multi-dimensional arrays (matrices or tensors) of floats (w/ known dimensions)
  - Graph: an abstract data type (ADT) with set of vertices (say, integers) and set of edges (pair of integers)
  - Program in PL, SQL query: string (w/ grammar)
  - DRAM addresses: array of bytes (w/ known length)
  - Instruction in machine code: array of bytes (w/ ISA)
  - Other data structures or digital objects?

### Serialization and Deserialization:

- A data structure often needs to be persisted (stored in a file) or transmitted over a network
- Serialization is the process of converting a data structure (or program objects in general) into a neat sequence of bytes that can be exactly recovered; deserialization is the reverse, i.e., bytes to data structure
- Serializing bytes and characters/strings is trivial
- 2 alternatives for serializing integers/floats:
  - As byte stream (aka "binary type" in SQL)
  - As string, e.g., 4B integer 5 -> 2B string as "5"
  - String ser. common in data science (CSV, TSV, etc.)

# Serialization and Deserialization in ML

- We often convert a trained model into a format that can be stored or transmitted. This involves transforming it into a sequence of bytes that can be written to disk or sent over network (i.e. we have to serialize it).
- Deserialization is the process of converting a serialized model back to its original data structure so that it can be used for inference. We load it back into memory for inference or evaluation purposes.
- Can be implemented in various formats, such as JSON, protocol buffers, or Apache Avro.
- We can serialize any other ML related artifacts like transformers, data, metadata, etc.

### A Common Serialization Scenario...

```
import pandas as pd
from sklearn.linear_model import LogisticRegression
import pickle
```

```
# Load the Iris dataset
iris_df = pd.read_csv('iris.data', header=None)
X = iris_df.iloc[:, :-1]
y = iris_df.iloc[:, -1]
```

```
# Initialize a logistic regression model
clf = LogisticRegression(random_state=0, max_iter=1000)
```

```
# Fit the model on the data
clf.fit(X, y)
```

```
# Serialize the model to disk
filename = 'logistic_regression_model.pkl'
with open(filename, 'wb') as file:
    pickle.dump(clf, file)
```

print(f'Saved the model to {filename}')

### **Review Questions**

- What is the difference between data and code?
- What kind of software is TensorFlow? Linux?
- Why do computers use binary numbers?
- What is a byte?
- How many integers can you represent with 5 bits?
- How many bits do you need to represent 5 integers?
- What is the hexadecimal representation of 2010?
- Why is a floating point standard needed?
- Why should a data scientist know about float formats?
- What does "lower precision" mean for a float weight in DL?
- Why is serialization needed on a computer?
- Is code a string? Is a string code?
- Is reality a computer simulation? :)

## Outline

- Basics of Computer Organization
  - Digital Representation of Data
  - Processors and Memory Hierarchy
- Basics of Operating Systems
  - Process Management: Virtualization; Concurrency
  - Filesystem and Data Files
  - Main Memory Management
- Persistent Data Storage

### In class activity

What is 64 in base 2 notation? If you think you might need partial credit, show your work.

- () 110001
- () 111111
- () 1010000
- () 1000000
- () 1001000

Which of the following is NOT a potential disadvantage of serializing a machine learning model? If you think you might need partial credit, explain your justification.

() Increase in the model's size due to the inclusion of architecture and state information.
() Reduced compatibility with different programming languages and versions.
() Loss of interpretability and ability to modify the model's internal workings.
() Risk of privacy breaches due to unauthorized access to the model's data or parameters.
() Increased computational overheads during model inference due to serialization and deserialization processes.