Instructors: 杜韬(Tue 14-15, FIT 1-202) 吴翼(Fri 11.30-12.30);

TAs: 胡昌宇 吕康博 崔景植 欧阳宇涛

Lecture 1: Intro to AI & RL – Yi Wu

1950, Alan Turing, “computing machinery and intelligence”, play chess, speak English;

1956, Dartmouth, John McCarthy(1971 Turing)-AI proposal, Marvin Minsky(Turing 1969)-“solve in a generation”

McCarthy: Math logic for AI. **LISP**(1950s):based on lambda-calculus, a self-modifying program. First chess program.

1st AI spring w/ logic programming (1970s-1980s): **expert systems**, if-else, 1994 Turing award Feigenbaum & Reddy. Baidu(<2008). Deep Blue(1997) chess search machine.

Noam Chomsky, “father of modern linguistics”, 1950s, a set of rules to describe & generate **natural language**. Google translate < 2016.

1st AI winter: 1970s(perception abandon of connectionism).2nd AI winter: mid 80’s-mid 90’s(end of symbolism).

Minsky (1969 Turing), 1966 MIT Summer Vision Project.

Frank Rosenblatt, 1958, **perceptron**. Cannot solve XOR.

Post AI winter, late 1990s-2000s. statistical methods,human-spec rule.

**Machine learning:** 1990s

**Data-Driven Methods:** Judea Pearl(2011 Turing). probabilistic approaches & causality, Bayesian machine learning(late 2000s).

**Deep Learning:** AlexNet, 2012 by Alex, Ilya & Geoffrey Hinton. “Revival of connectionism”.

Turing @ 2018: **Geoffrey Hinton** (backprop, makes DL work, **1986** Nature “Learning Representations by **Back-propagating** Errors”); **Yann** **LeCun** (CNN, modern pattern recognition); **Yoshua Bengio** (neural language model & attention).

RL – agent, environment, policy. Domain randomization(sim2real): force the agent to adapt via randomizing env. Human like behavior.

Human-AI cooperation with language, language controlled robot.

Lecture 2: Intro to TCS - Yau Chi-Chih

**Yao’s Millionaires’ problem:** secure multi-party computation problem, who is richer without revealing their actual wealth. A puts money into envelopes 1-a, B checks bth.

**2024**: Blockchain(internet), ChatGPT(AI), Quantum computing(QC), DNA-synthesis (CRISPR, molecular biology)

**1950’s-1980’s**: IBM. PARC invented first PC (Alto), 1973.

Ethernet, at Xerox PARC in 1973-1974 for communication; @ 1980: 10Mb/s, Dave Redell and John Shoch. Laser Printer 1971.

Network computing: distributive comp->blockchain. Cryptography.

**QC:** Heisenberg/Schrödinger, 1925/6, birth of **quantum mechanics**.

Feynman, 1982, simulating physics with computers, “can we make a quantum computer?”

Classic Turing machine: deterministic, transition table (rules of state transition) cannot change, finite, discrete, distinguishable, Alan Turing 1936, universal Turing machine.

**Byzantine General Problem:** n generals (1 commander), t traitors. Loyal general reach agreement at last. Everytime pick n-t generals send message, and take majority message.

Lecture 3: Practical Skills in CS Research – Tao Du

Dive to subfield: talk expert; find and read online; code example

Read paper:

Abstract: problem; related work; method; result

Main paper:Intro->related work->experiment and discussion -> method->conclusion

Intro:context of research problem, challenges of existing methods, technical insight, experimental result

Lecture 4: Computer Architecture – Gao Mingyu

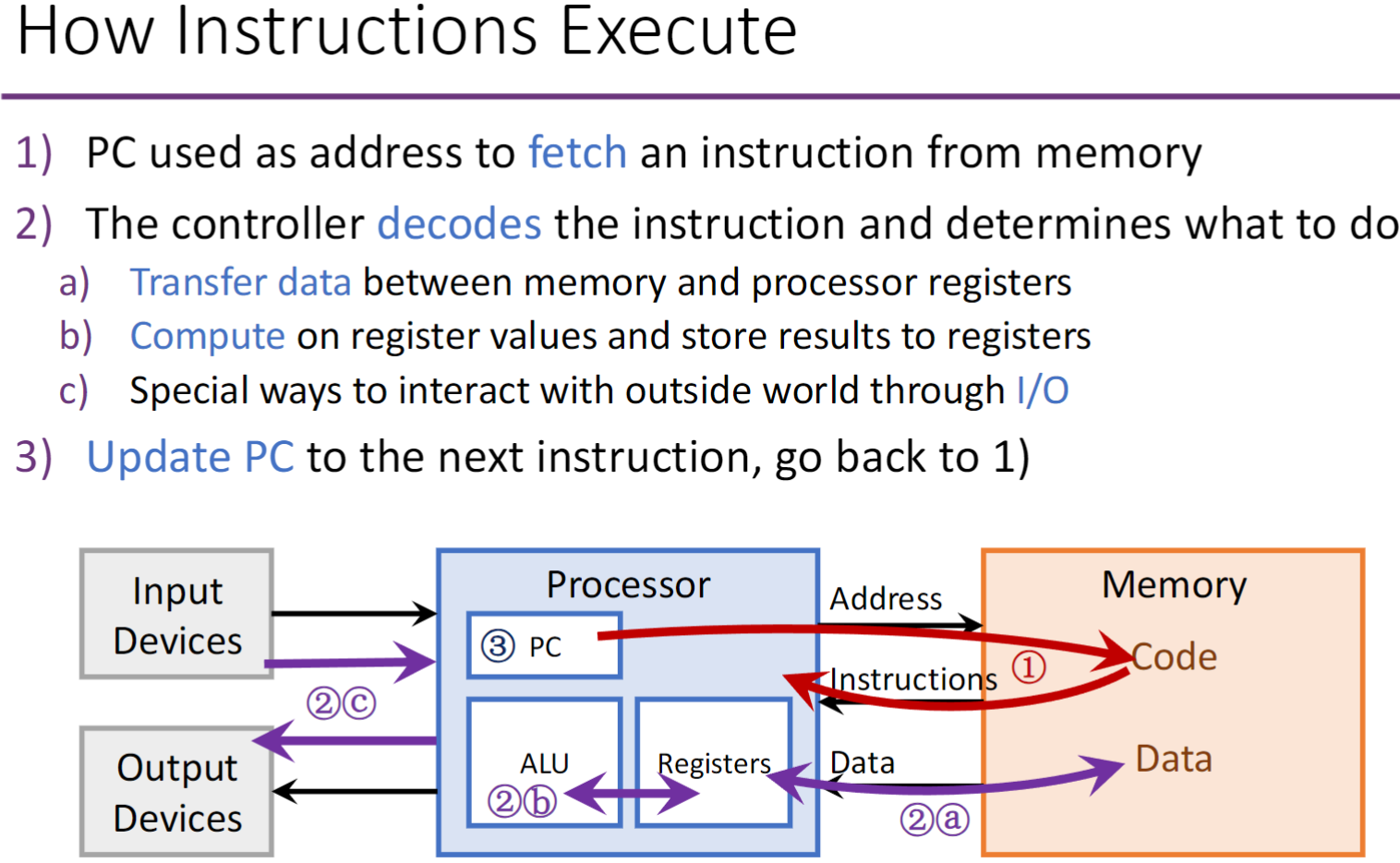
Von Neumann Architecture: memory for both data & programs, compute unit (arithmetic/logic units ALU – digital logic for computation; PC – address of next instruction; program counter – address of the next instruction), control unit (coordinate program flow), input/output. Key point: separate processor and memory.

Code compilation: compiler 🡪 assembly program 🡪 assembler 🡪 machine object 🡪 linker 🡪 executable 🡪 loader🡪processor memory.

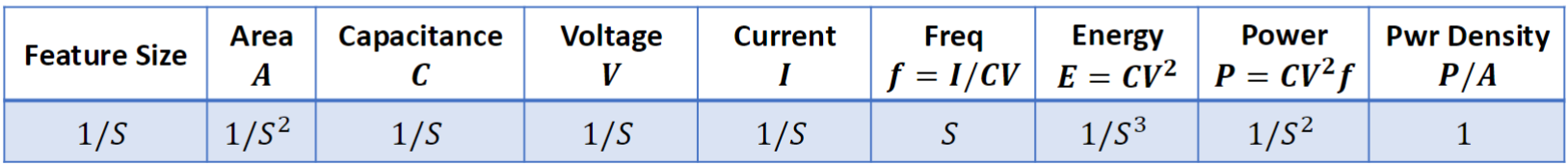
Machine code: byte sequence encodes program instructions; Assembly code: text representation of machine code; Assembler: binary encode instruction; Linker: inter-file references.

Compute units: adder, multiplier…

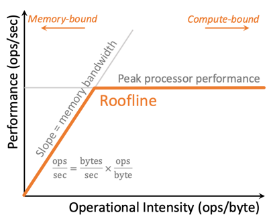
Full adder ,



**Moore’s Law:** original - The complexity for minimum component costs has increased at a rate of roughly a factor of two per year; new: The number of transistors per square inch on integrated circuits is doubling every year; In 1975, revised to doubling every two years.

**Dennard Scaling:** performance & cost. 

**Parallelism:** bit-level(datapath bit width), instruction-level(pipelining, superscalar&OoO), data-level(SIMD), thread-level(multicore).

Post-Dennard Scaling Era: transistor size too small(leakage->Moore without Dennard), power wall (all computers are power limited)(Dark Silison: only utilize part of the chip; slow down freq scaling)

**Roofline model:** performance (ops/sec) vs. operational intensity (ops/byte), low OI slope=memory bandwidth, memory-bound; high OI compute-bound. Memory wall: memory scales more slowly.

**Trends:** domain-specific accelerations(DSA-reduce overhead of general purpose, custom opt, effective parallel, mem access efficient), new memory architectures (Interchip transfer high cost) near-data processing:3D stacking/processing-in-memory:ReRAM), (scaling down? In lecture 14)

Lecture 5: Intro to Quantum Computer Science – 侯攀宇

Trapped ions and color centers in diamond

Quantum sensing, communication, simulation & computing.

Quantum interference: check all possible paths at the same time with good quantum algorithm.

**Shor’s algorithm**, 1994, offers exponential speedup. It is used to factor large integers. Shor’s algorithm combines the Quantum Fourier Transform and modular exponentiation to find the period of large integers.

Platform: Trapped ions, Neutral atoms, Superconductor, Photonics

**NISQ era:** Noisy intermediate-scale quantum computing, no use.

Quamtum error correcting: logic bit.

One time pad: random key used one time + XOR

Lecture 6: Smart Contract – Fan Long, University of Toronto

**Blockchain:** decentralized, transactions are irreversible, cryptographic hash of previous block is contained in all blocks except the first.

**Smart Contract:** a self-executing program on top of blockchain, encode transaction rules, users interact with it by issuing transactions, rely on oracles to access off-chain data.

**Decentralized Finance:** apps that allow users to perform financial transactions. Runtime enforcement of invariants can be used to prevent harmful transactions and states in smart contracts. DeFi lending protocol: have smart contracts to allow people deposit and borrow digital assets.

**Oracle:** provide asset price info for DeFi. Deviation tolerance.

In a **flash loan attack**, the attacker takes out a large, uncollateralized loan and exploits vulnerabilities in decentralized finance (DeFi) protocols to manipulate the prices of assets, profiting from the rapid repayment within the same transaction.

**FlashSyn:** Prune search space of attack vectors, identify action candidates and choices of non-integer arguments.

**Program Synthesis:** e.g., check whether turtle & grass inside rectangle. **Program:** data-structure, the rectangle. **Input:** data-structure, coordinates. **Output:** data-structure, whether inside rectangle. **Interpreter:** function (program, input) 🡪 output. **Specification (Spec):** stating the task so that both human and computer agree on what needs to be done, e.g., a list of input-outputs.

Lecture 7: Computer Vision – Jia Deng, Princeton

AlexNet: CNN & fully connected layers; max pooling layers, ReLU, data augmentation. Structured grid data.

ImageNet: diverse range of categories, large numbers of examples (crawl the web), human worker labelled (with multiple checks).

Visual SLAM(simultaneous localization and mapping): creates a map of the environment while simultaneously keeping track of the camera’s location, uses features extracted from images to build the map and localize the camera, problem of Deep Visual SLAM is that it’s not easy to generalize to new datasets or cameras, low accuracy.

Procedural Synthetic data: 3D vision solution.

Lecture 8: Intro to Theoretical Computer Science – Lijie Chen,UC Ber

**NP**: YES/NO problems, solution verifiable in poly time. **P**: poly time solvable problems. **PSPACE**: problems solvable in poly(N) bits of space. L: solvable in O(log N) bits of space. **BPP**: in poly(N) randomized time. **IP**: solution can be verified interactively in poly(N) time. We have **IP=PSPACE, NPSPACE**. **NEXP**: solution verifiable in time. **MIP(\*)**: solution can be verified interactively in poly time with two non-communicating (\*: quantum) provers. RE: solution verifiable in whatever time. **NEXP=MIP, MIP\*=RE. PSPACE=NPSPACE**. NSPACE(f(n)): solutions can be determined by a non-deterministic Turing machine with space f(n). NPSPACE: union of NSPACE(n^k). Open: **P=?PSPACE, BPP=?P, L=?P, P=?NP.** Best algorithm for **maximum matching**: . **Shortest path**: O(nlogn+m) for directed graph w/ non-negative weights, O(m polylog(n)) when there are negative weights, O(m sqrt(logn loglogn)) for undirected graphs. **All pair shortest path**: Floyd O(n^3), 2014 O(n^3/2^sqrt(logn)), open: can we do better? Fine-grained complexity

Derandomization: random to deterministic with almost no overhead

Lecture 9: AI for Social Good – Fei Fang, CMU

**Application domains**: healthcare > transportation > security. **Techniques**: machine learning > human computation & crowd sourcing.

**Functionality**: descriptive, predictive, prescriptive. Three **conceptual methods** to identify SG problems: AEC framework, DPP framework, and supply chain analysis.

Security (patrol): Game model and Linear Programming-based solution –– Zero-sum, min Attacker’s max expected utility, Flow-based Representation + Critical Time points

Poacher: Decision Trees + Markov Random Field (challenge: Lack of labeled data + data imbalance), 基于捕猎者模型plan patrol

NewsPanda: Fine tune LM(BERT) to look through articles to identify trends, events of threats.

Identify AI4SG problem: AEC framework, DPP framework, supply chain analysis.

Volunteer-based Food Rescue: stacking model (预测接单)

Optimize Intervention and Notification Scheme (INS) by Branch-and-Bound Algorithm.

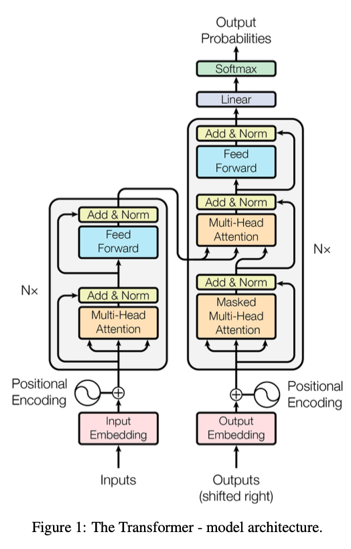
Rescue-Specific Notification: 限定每人最多接到L条通知, 贪心选前k, online planning

Lecture 10: Physical Simulation – Doug L. James, Stanford

Increasing grid resolution does not always make the simulation results converge. **Discretization** is the process of converting continuous-time equations into discrete forms for numerical solving. **Mesh representation** is a common data structure used to describe the surface of 3D objects, typically consisting of vertices and faces. A **rendering algorithm** usually requires additional information of mesh representation, such as normal vectors and UV coordinates. A **Bézier curve** is defined by at least two control points. **Ray tracing** is typically slower than **rasterization** in real-time rendering. However, ray tracing provides more realistic lighting and shadows. The core of **CG** is generating 2D images (or other visual outputs) from scene information. The core of **CV** is inferring scene information from 2D images. In **rigid body dynamics simulation**, the update of velocity and position is typically performed using time integration methods, such as the Euler method or Verlet method. The **Finite Element Method (FEM)** is used to calculate both forces and deformations in elastic simulations. It solves the partial differential equations governing elasticity, providing a way to compute the deformation of an object under applied forces. **Collision handling** involves detecting the exact time and location of collision events and resolving them by adjusting velocities or positions to prevent interpenetration. **UV mapping** maps 2D texture to 3D models.

Progressive Simulation: coarse approximation -> progressive refine, real-time interactivity, adaptive resolution, high fidelity.

Lecture 11: How Undergrads Start AI Research – Lei Li, CMU

**ALGO**: Use brute-force LLM program as oracle; use LLM to generate test input data & oracle to compute outputs 🡪 synthetic test cases. Instruct LLM to generate efficient programs (prompt LLM to random sample an idea & generate the code) & verify correctness via synthetic test cases.

**ToolDec**: Uses finite-state constrained decoding to enable LLM to use tools without syntax errors, zero-shot, and document-free. Fine-tuning can only zero-shot, in-context learning can only document-free.

**LingoLLM**: Training-free method. First, morphological analysis (e.g., cats = cat + plural); second, dictionary matching; last, LLM translation.

No Reinforcement Learning from Human Feedback (RLHF). LLM needs **feedback** (not vague/incorrect), they come from self-gen oracle, another smaller LLM, or separately trained metric.

Lecture 12: Efficient Deep Learning – Song Han, MIT

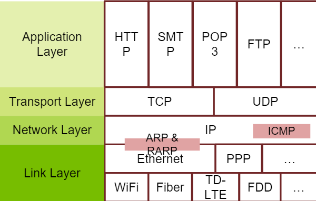
**Deep Compression** reduce the storage and computational requirements of deep learning models to deploy on mobile devices, pruning and quantization. A potential disadvantage of **pruning**: loss of accuracy due to removing important weights. **Network quantization** (reduce precision of weights and activations, 32-bit floating-point to 8-bit integers.) can reduce the size of the model and speed up the inference process. **On-device** (opposed to cloud-based) **training** challenge: quantized graph hard to optimize due to low precision & lack of normalization. **Transfer learning** allow fine-tune pre-trained models with fewer resources.

Huffman coding: lossless coding, more frequent code = shorter, complexity O(n logn).

Deep Learning 3 pillars: algorithm, hardware, data.

Lecture 13: Principles of Computer System Design – Wenguang Chen

**Why difficult**: consistency, fault-tolerance, performance. Scale of system, complexity: requirement, utilization-complex optimization

Common **problems** of systems: emergent properties (not considered at design time,ethernet), propagation of effects(heartbeat), incommensurate scaling (internet, Network Address Translator & Ipv6), trade-offs(waterbed effect: throughput & delay (split logic)).

**Coping with complexity**: modularity (split up system, bug count), abstraction (interface/hiding, avoid propagation of effects, like OOP, virtualization, ISA), layering (gradually build up, TCP/IP), hierarchy (reduce connections, divide-and-conquer).

Lecture 14: 面向下一代人工智能的高能效电路与系统设计-汪玉

**能效提升**：Scaling down，custom accelerators，new devices & models（高能效存内计算新架构） 软硬件协同

软硬件协同优化：面向神经网络，容错度较高, enable analog circuits in accelerator.

应用场景：云端(服务器，计算+存储)，边缘(基站，路由器，计算+通信)，终端(显示+交互)

FPGA (Field-Programmable Gate Array)：circuit that can be reprogrammed to customize hardware logic. Paralleism

Application-Specific Integrated Circuit.

**CPU clock speed** is not directly proportional to its processing power, architecture, cache size, memory bandwidth, and thermal constraints also affect overall processing power. **Processing-in-Memory (PIM)** architecture often requires specialized hardware, such as non-volatile memory or in-memory processing units, to achieve optimal performance.

**存算一体架构**：ReRAM O(1)矩阵向量乘法,无需搬数据

**Analog in-memory computing** uses memristors for matrix computation, with computational complexity related to the input matrix dimensions. **Digital in-memory logic computing** reduces data transfer overhead by performing logical operations directly within the memory cells. It reduces the dependency on external processors or co-processors.

In traditional architectures, the time of data movement overhead is typically O(n).

AI 1.0: 用于判别任务的专用模型 AI 2.0: 生成任务的通用模型

算法创新优化(10 - 30) + 模型计算优化(10 -20) + 算力平台优化(5 - 10) + 硬件推理优化(10 - 20)

CPU which strictly follows the von Neumann architecture only perform 1 instruction at one time.

GPU: **并行加速** NPU: specialized hardware accelerators designed to accelerate AI applications.

**Specialized accelerators** reduce power consumption by optimizing the architecture for specific workloads.

**Memory access** is typically less energy-efficient than computation. Accessing data from off-chip memory (e.g., DRAM) consumes significantly more energy than performing on-chip computations. Therefore, reducing data movement and optimizing memory usage is critical to improving energy efficiency in AI workloads.