Applications of Parallel Programming

Chris Kauffman

Last Updated:
Tue Apr 26 09:38:57 AM CDT 2022

Logistics

A2 Cancelled

- All students get full credit for remaining assignments, no re-weighting of exams or previous assignments
- Reasons
  - There's not enough time to do a large meaningful assignment
  - Kauffman is spent, does not have resources to construct a shorter assignment
  - None of this is ideal but at least it is easy and less stressful than alternatives
- If you want some extra credit to make up for a loss on A1, write a cool MPI program and email it to me

Remaining Classes

Thu 4/21	Applications
Tue 4/26	Applications / Environments
Thu 4/28	Finale
Mon 5/9	Final Exam 8am-10am

Guest Lecture

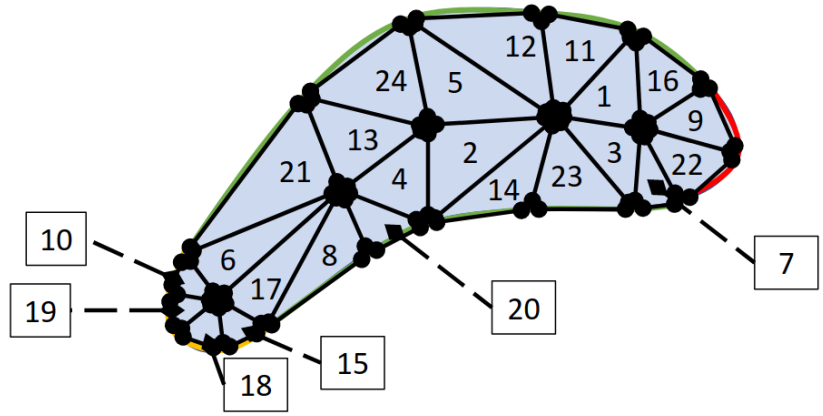
Some Applications of Parallel Computing in Fluid Mechanics

Sreevatsa “Sam” Anantharamu, Aerospace Engineering

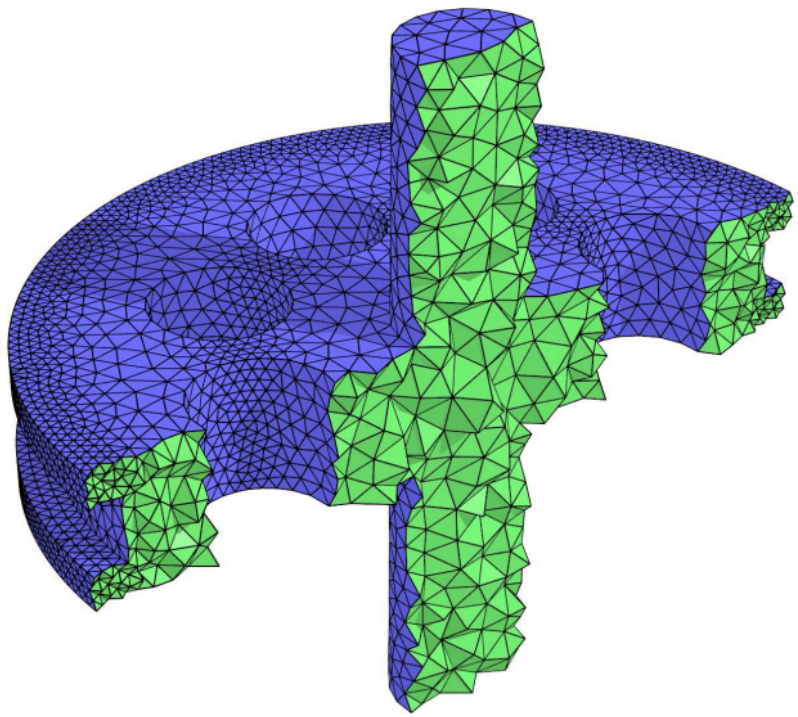
Reviewing Fluid Dynamics: Meshes

Divide the domain into triangles; the set of triangles is called a mesh/grid

Done long in advance of running any simulations



Source: Some applications of parallel computing in fluid mechanics by Sreevatsa Anantharamu

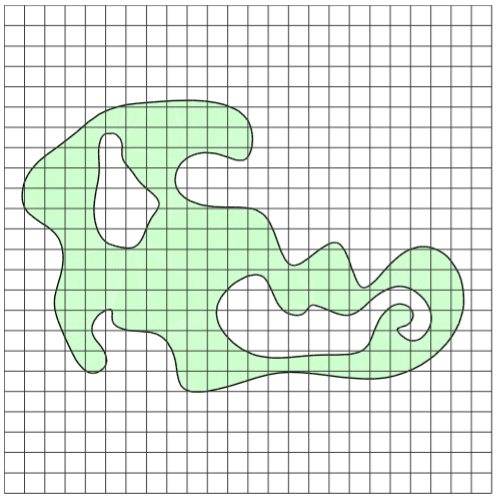


Source: Mesh Generation for Implicit Geometries by Per-Olof Persson (Thesis)

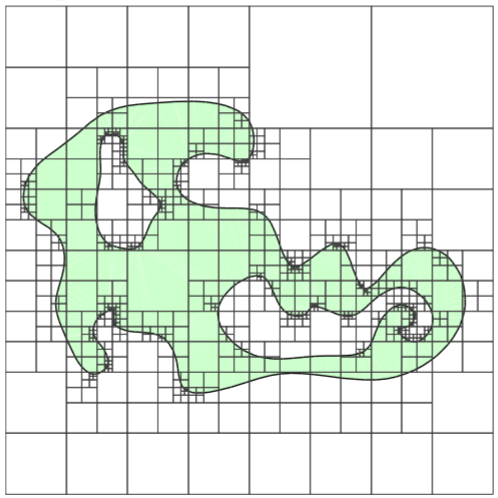
Reviewing Fluid Dynamics: Mesh Generation

Mesh generation has many styles / techniques, is its own (parallel) problem

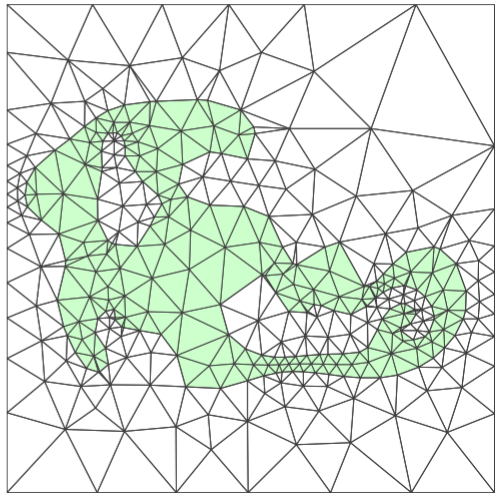
Cartesian



Quadtree/Octree



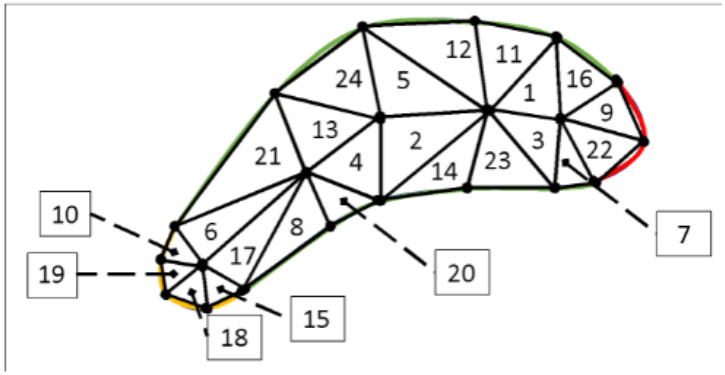
Unstructured



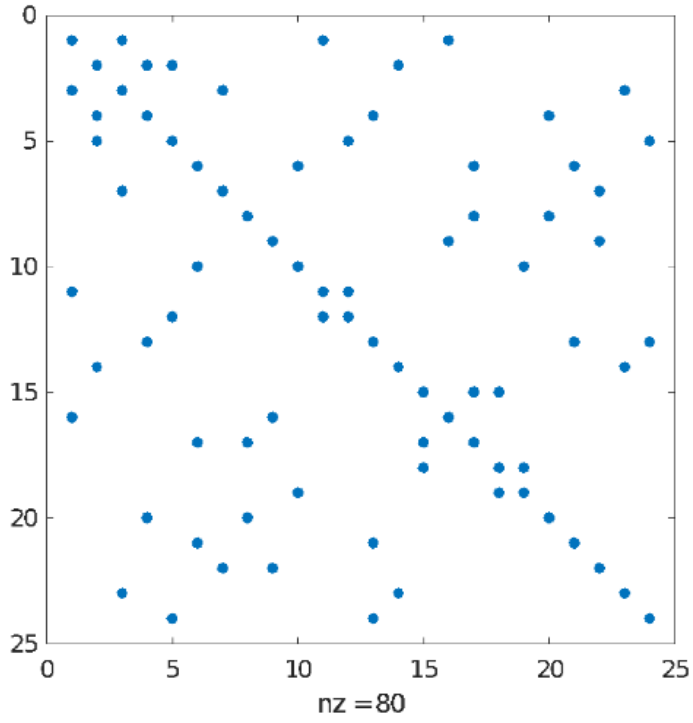
Exercise: Mesh to Matrix

After creating a mesh, Sam alluded to getting a sparse matrix from it.

1. What is this matrix?
2. Why is it sparse?
3. What relation does it have to the Heat problem we have studied several times?
4. What would the matrix for the Heat problem look like?
5. Sam described it as an "unstructured" sparse matrix. What does this mean?



Mesh



Sparsity pattern of A

Answers: Mesh to Matrix

1. What is this matrix?
   Matrix of neighbors: which triangles share a face with others.

2. Why is it sparse?
   Not all triangles share faces. Each triangle only has 3 faces so has at most 3 neighbors.

3. What relation does it have to the Heat problem we have studied several times?
   The heat problem also defined neighbors which dictated heat transfer. Each element in the rod had 2 neighbors (left and right).

Answers: Mesh to Matrix

4. What would the matrix for the Heat problem look like?
   Tri-diagonal: element i has neighbors $i-1$ and $i+1$.

$$T = \begin{pmatrix} a_1 & b_1 & & & \\ c_1 & a_2 & b_2 & & \\ & c_2 & \ddots & \ddots & \\ & & \ddots & \ddots & b_{n-1} \\ & & & c_{n-1} & a_n \end{pmatrix}$$

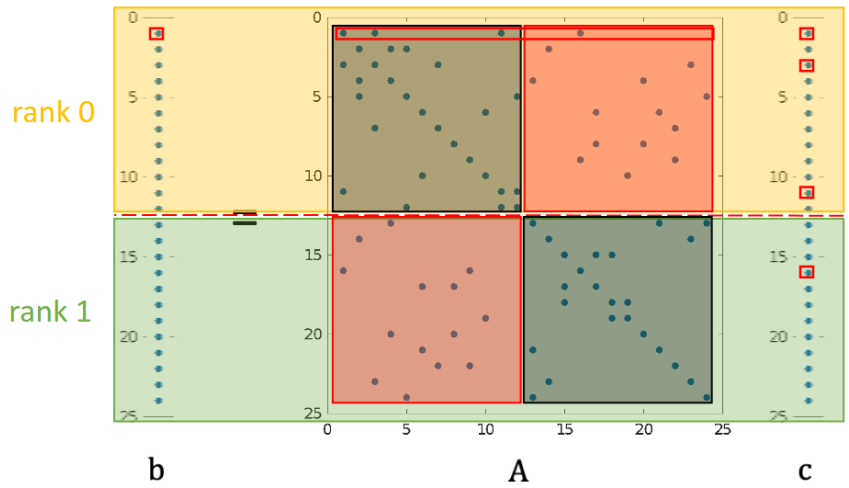
5. Sam described it as an "unstructured" sparse matrix. What does this mean?
   For an arbitrary mesh, the neighbor matrix does not follow a regular pattern like in the Heat problem. Element 4 has neighbors 2, 13, 20; no pattern for the neighbors.

Exercise: Sparse Matrix Vector Multiply

- Sam indicated a Sparse Matrix/Vector multiply is a central operation for the simulation
- Advocated using a simple Row Partition scheme for the matrix, input vector, output vector

Questions

1. What need for communication does this incur?
2. Why is this situation more complex than many matrix problems we have discussed? Give 2 reasons/



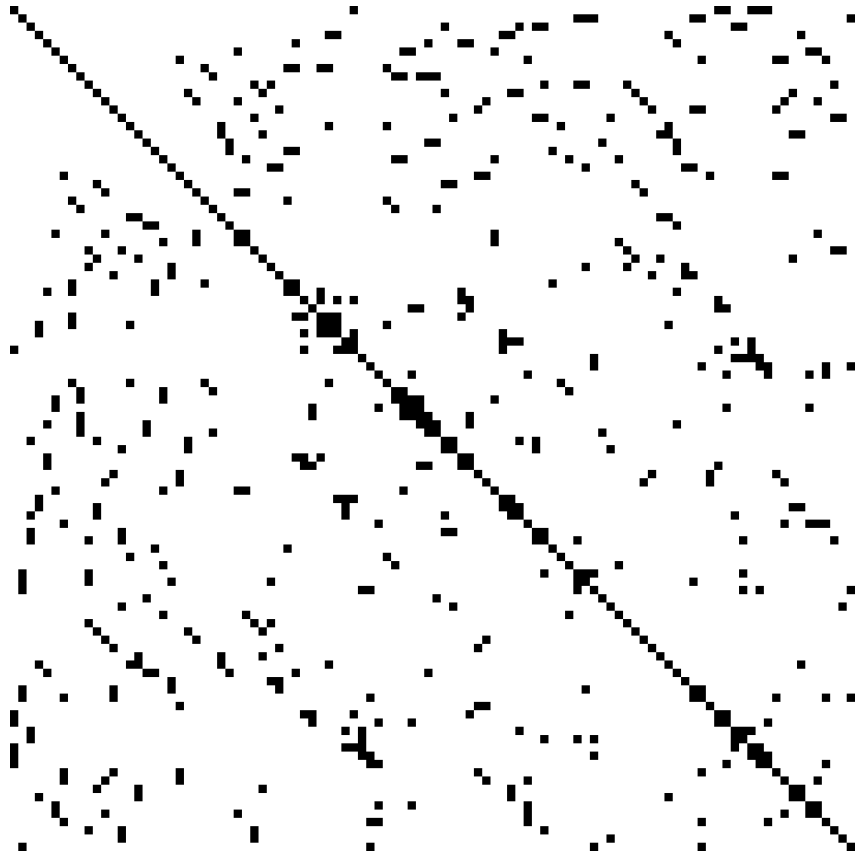
Answers: Sparse Matrix Vector Multiply

1. What need for communication does this incur?
   Procs do not have the entire RHS vector; must communicate with other procs to multiply it. Other procs must know who will ask for data from them.

2. Why is this situation more complex than our work on Page Rank: give 2 reasons?
   - 2.1 *Most matrix problems we discussed are dense so the data communicated were just blocks of dense arrays. In sparse problems must communicate in a format like Compressed Sparse Row (CSR).*
   - 2.2 *We often assumed whole vectors fit on procs so no need to communicate this part; Sam indicated for large problems, cannot do this.*

Exercise: Distribution of Sparse Matrices

- Given a square sparse matrix with N rows
- Have a parallel computer with P procs
- Will partition RHS vector as well
- Which proc should get which rows (matrix/RHS vector)?
- What criteria should be used to dictate partitioning?



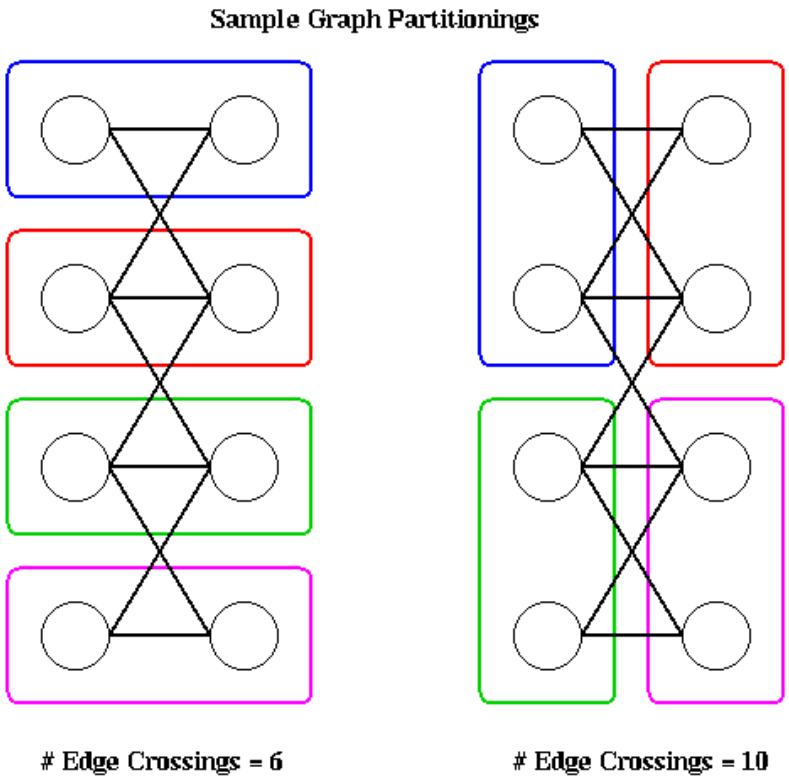
Answers: Distribution of Sparse Matrices

- Want to minimize communication between procs
- If Element 4 has neighbors 2, 13, 20, put rows 2,4,13,20 on same processor
- Leads to a classic NP-Hard problem: **Graph Partitioning**

  In mathematics, a graph partition is the reduction of a graph to a smaller graph by partitioning its set of nodes into mutually exclusive groups. Edges of the original graph that cross between the groups will produce edges in the partitioned graph. If the number of resulting edges is small compared to the original graph, then the partitioned graph may be better suited for analysis and problem-solving than the original.

  – *Wikipedia: Graph Partitioning*

Graph Partitioning



Source: CS267 Lectures 11 and 12 Note by James Demmel, Berkeley

- Graph partitioning seeks a "reduced" graph where multiple vertices are merged into a single vertex group
- Edges within group cost nothing - on same processor
- Seek to minimize edges between groups, the **Cut** of the partition
- Widely used approach is METIS (serial) and ParMETIS (parallel) by our own George Karypis and Vipin Kumar

N-Body Simulations

Several types of Physics and Chemistry systems are simulated in the following way

1. N particles exist in some space
2. A **Force Field** is present which describes attraction/repulsion between particles (and possibly external forces)
3. Particles have their positions initialized in the space
4. Forces acting on each particle are calculated using the Force Field and the current state (position/velocity) of the particles
5. The positions of each particle are adjusted by a small amount based on forces acting on them
6. Repeat 4/5 until bored

In Astrophysics, particles are usually celestial bodies (planets, stars, galaxies) and Force Field is Gravity.

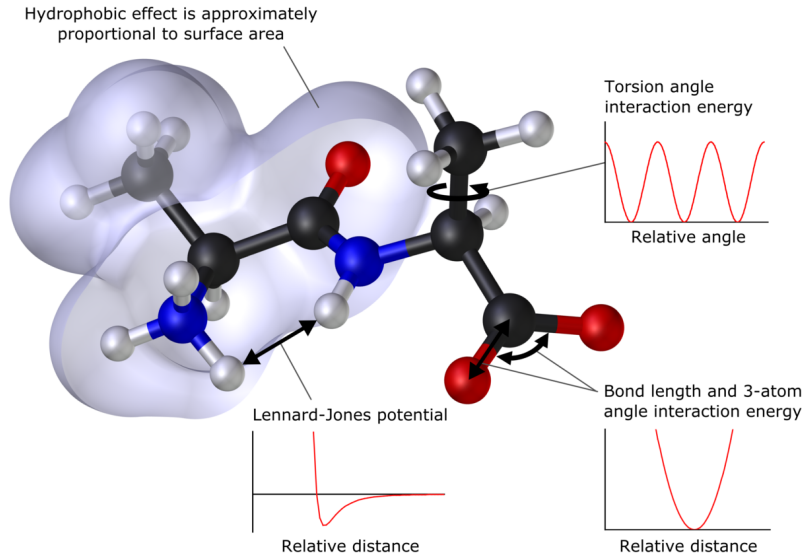
Exercise: Sample N-Body

```
1  # calculate forces for new accelaration
2  for i=0 to N-1:
3    acc[i].x = 0
4    acc[i].y = 0
5    for j=0 to N-1:
6      if i==j: continue
7      dx = pos[j].x - pos[i].x
8      dy = pos[j].y - pos[i].y
9      inv_r3 = (dx*dx + dy*dy)**(-1.5);
10     acc[i].x +=  G * (dx * inv_r3) * mass[j];
11     acc[i].y +=  G * (dy * inv_r3) * mass[j];
12
13 # calculate new position
14 for i=0 to N-1:
15   vel[i].x += acc[i].x
16   vel[i].y += acc[i].y
17   pos[i].x += vel[i].x * dt
18   pos[i].y += vel[i].y * dt
```

- Code at the right represents (very rough) N-body simulation in 2D
- What opportunities do you see for parallelism?

Molecular Dynamics

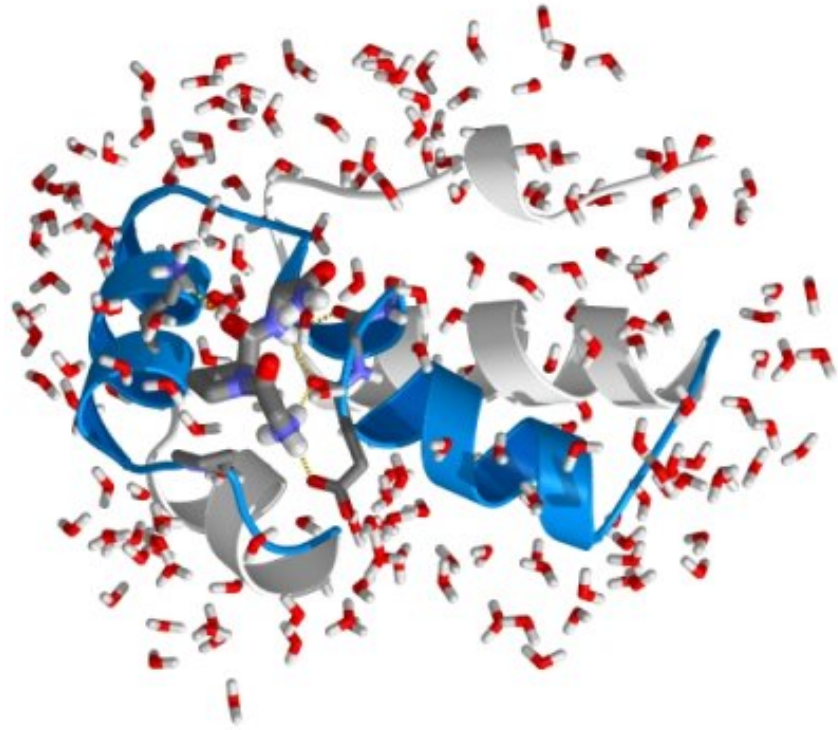
- Chemistry simulations can use a similar approximation to N-body but are focused on microscopic systems: particles are Atoms and Force Field is electrical interactions
- **Bonds** between atoms which act like springs which stretch/compress and allow twisting
- **Van der Waals** forces between non-bonded atoms which exerts attractive/repulsive forces



Source: Wikip Molecular Mechanics

Protein Folding

- Protein Folding is one heavily studies area of Molecular Dynamics
- Through experimental techniques / genetics can determine the bonded atoms a protein: 1,000-25,000 atoms with water around it
- Proteins are a large linear molecule that "folds" due to interactions with surrounding water + self attraction



Source: Intro to protein simulations

Segue to Machine Learning

- Protein Folding is a tough physic simulation
- Has led to various approximations, heuristic, computational approaches. Notably, recent late Deep Learning approaches...

AlphaFold

From Wikipedia, the free encyclopedia

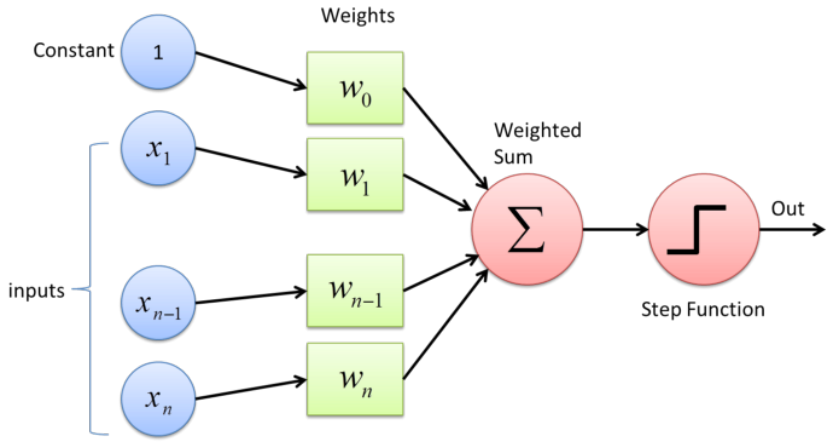
AlphaFold is an artificial intelligence (AI) program developed by Alphabets's/Google's DeepMind which performs predictions of protein structure.[1] The program is designed as a deep learning system.[2]

AlphaFold AI software has had two major versions. A team of researchers that used AlphaFold 1 (2018) placed first in the overall rankings of the 13th Critical Assessment of Techniques for Protein Structure Prediction (CASP) in December 2018. The program was particularly successful at predicting the most accurate structure for targets rated as the most difficult by the competition organisers, where no existing template structures were available from proteins with a partially similar sequence.

A team that used AlphaFold 2 (2020) repeated the placement in the CASP competition in November 2020.[3] The team achieved a level of accuracy much higher than any other group.[2] It scored above 90 for around two-thirds of the proteins in CASP's global distance test (GDT), a test that measures the degree to which a computational program predicted structure is similar to the lab experiment determined structure, with 100 being a complete match, within the distance cutoff used for calculating GDT.[2][4]

Perceptrons

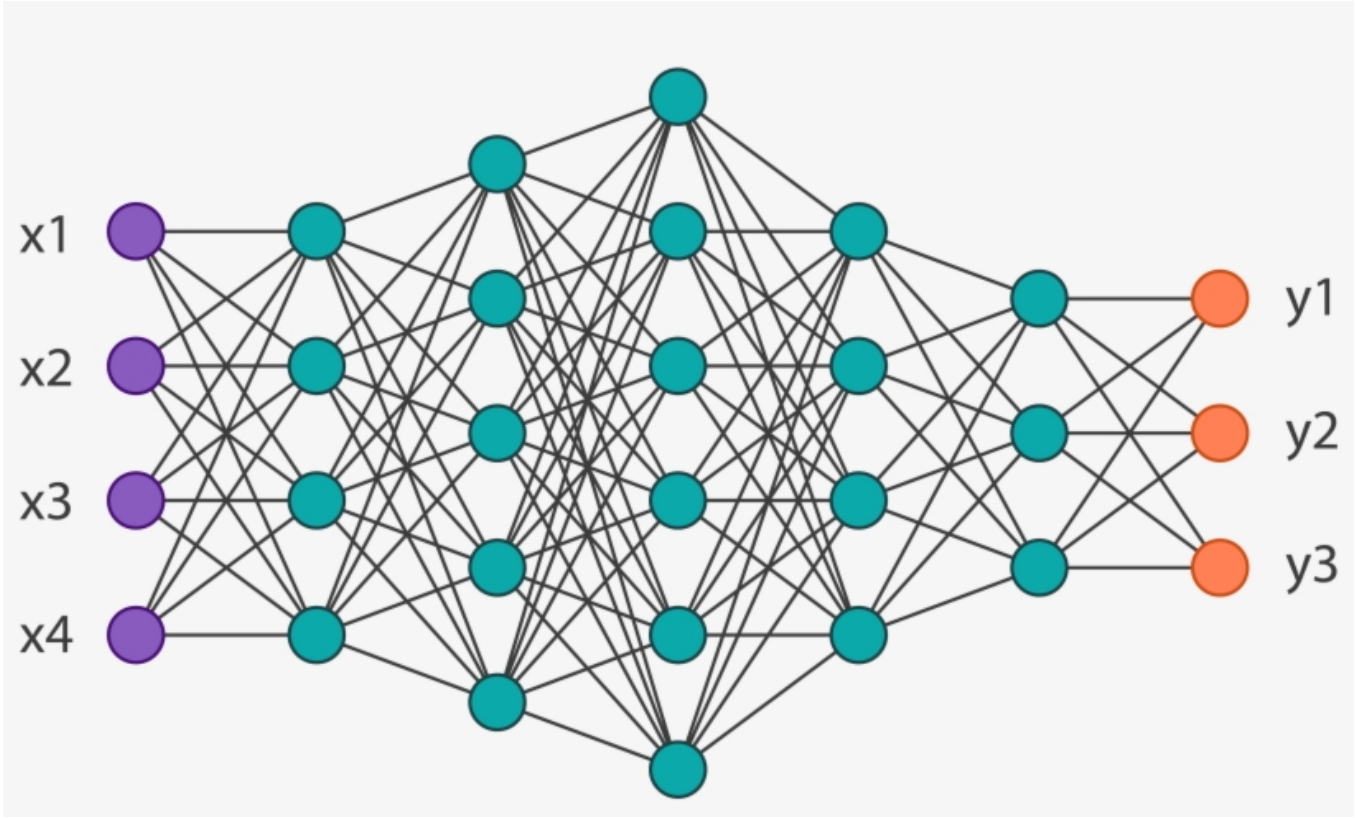
- Neural networks are comprised of a series of **Perceptrons**
- Input data vector is multiplied by a Weight vector and summed
- Basically a dot product followed by a (somewhat) arbitrary Activation function
- In a Single Perceptron, there is a vector of weights that are learned through gradient descent to predict an output value



Source: "What the Hell is a Perceptron" by Towards Data Science

- Will focus on Parallelism in computing outputs for neural networks - also parallelism in **backpropagation** to learn weights

Exercise: Neural Networks and Parallelism

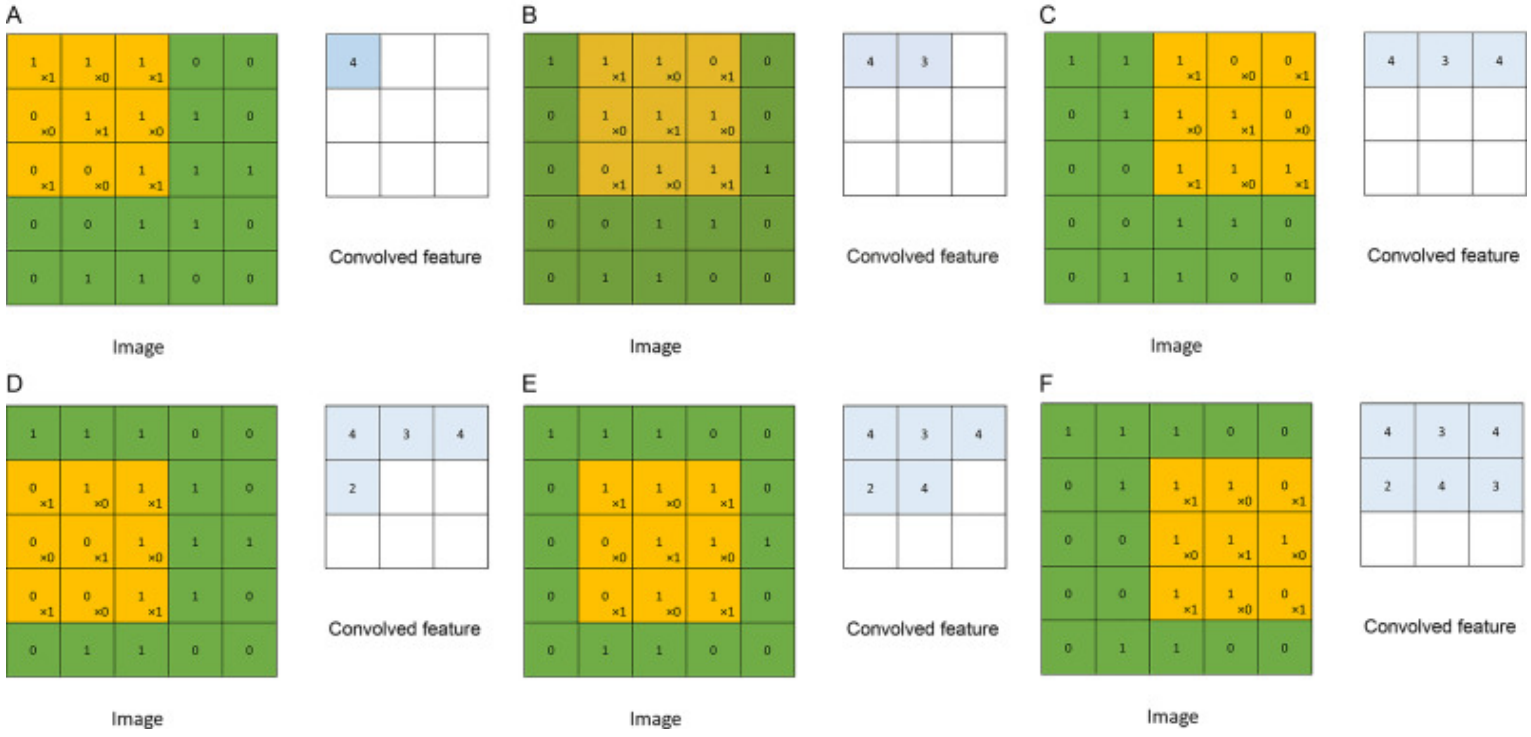


- Neural nets are layers of Perceptrons connected in some pattern: outputs in one layer become inputs in another
- Where can one find parallelism when calculating inputs to outputs in a neural net?

Examine: Pytorch Source Code

```
// pytorch/aten/src/ATen/native/cuda/Activation.cu
template <typename scalar_t>
inline void _rrelu_with_noise_cuda_train(...)
{  ...;
  rrelu_with_noise_cuda_kernel<scalar_t, 2><<<grid, block, 0, stream>>>(
      numel,
      rng_engine_inputs,
      output_data,
      input_data,
      ...);
}
__global__ void rrelu_with_noise_cuda_kernel(
    int numel,
    PhiloxCudaState philox_args,
    scalar_t* output,
    scalar_t* input,
    ...)
{ ...;
  int idx = blockIdx.x * blockDim.x + threadIdx.x;
  int grid_stride = blockDim.x * gridDim.x * unroll_factor;
  ...;
  for (int linear_index = idx;
       linear_index < rounded_size;
       linear_index += grid_stride)
  {
    ...;
  }
}
```

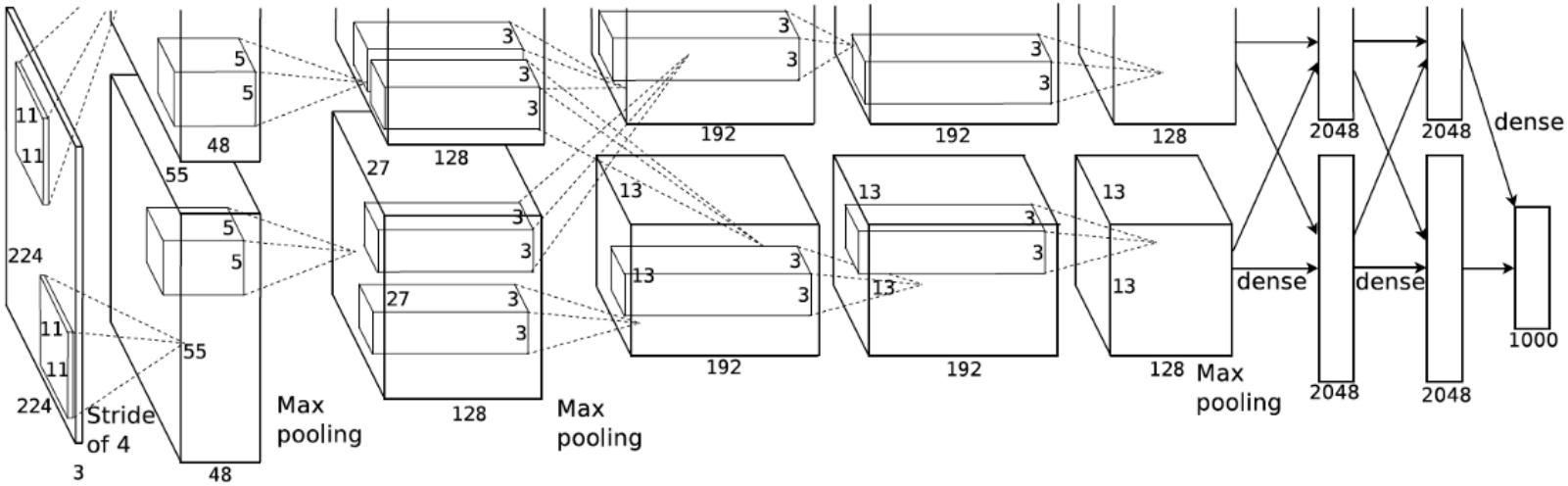
Convolution Neural Networks: Rife with Parallelism



Deep Neural Networks for Natural Language Processing Ehsan Fathi, Babak Maleki Shoja, Handbook of Statistics 2018

- In image processing, a **Convolution** is the application of a transformation matrix to an input image to produce a new matrix - repeated matrix inner product (Frobenius)
- Serve as the basis of Convolution Neural Networks where
  - Transformation reduces the size of the original image
  - Output matrix is used as input to deeper layers

Example CNN: AlexNet



ImageNet Classification with Deep Convolutional Neural Networks by Krizhevsky et al, Advances in Neural Information Processing Systems 25 (NIPS 2012)

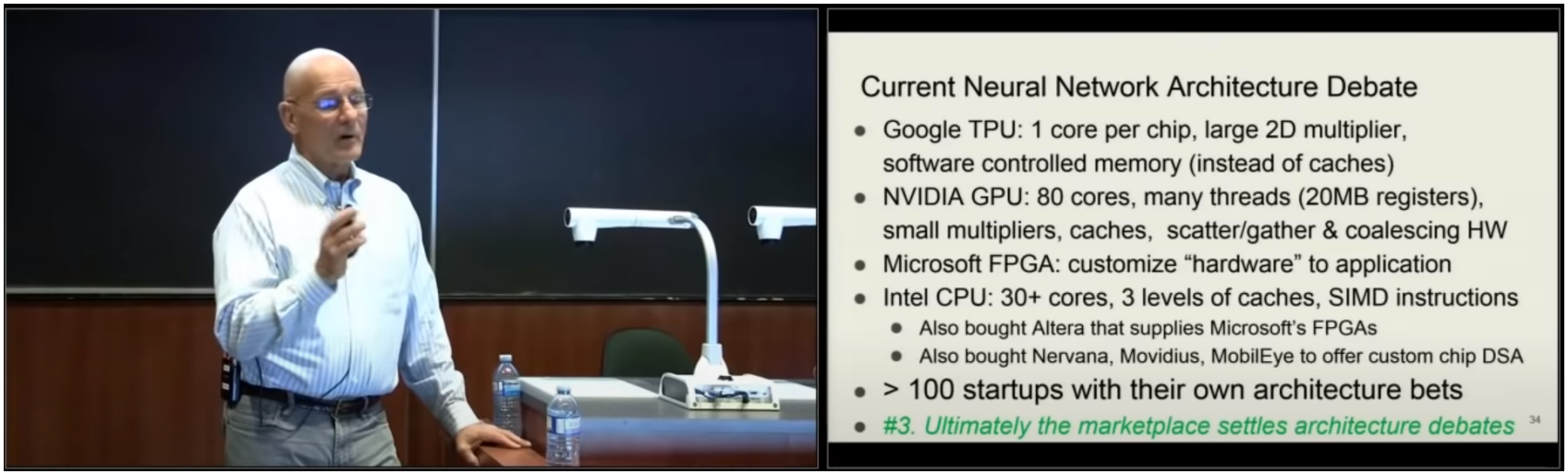
- First Convolution layer has 96 "filters" applied across images in 11x11 tiles
- 11x11 weights are "learned" and applied to image tile to produce 1 internal output - not fully connected
- Later layers have additional convolutions
- Ends with traditional Fully Connected "dense" layers to predict 1 of 1000 classes of images (dog, cat, car, etc.)
- Trained on 2 GPUs via CUDA to exploit parallelism

GPU Acceleration of Neural Nets

- Standard Neural Net layers boil down to a Matrix-Vector multiply then applying a function to each output vector element
- CNNs involve matrix-matrix multiplication followed by activation functions
- Neural nets can be done on distributed machines but it is more common to use single machines with multithreading
- Since GPUs fit the problem extremely nicely and are often used to accelerate Neural Network training and predictions
- Popular packages for neural networks like
  - TensorFlow (by Google)
  - Pytorch (based on Facebook's Torch library)

  all include GPU support for accelerating neural network operations

"A Cornucopia of Architectures"



Source: New Golden Age for Computer Architecture by David Patterson, 2019 (Youtube)

- In addition to GPUs, a wide variety of new computer architecture endeavors are specifically targeting Neural Network applications
- Application Specific Integrated Circuits (ASICs) to exploit parallel structure in training and inference

Cryptocurrency Mining

- A big topic, will look at one specific aspect: **Proof of Work** in Bitcoin
- Alice sends Bob 2 Bitcoins, a **transaction**
- Alice signs transaction with a private key to verify it is legit
- This transaction + 5 other transaction get grouped into a **block** of transactions:

  `[(A-2, B+2), (C-0.5, D+0.5), (E-3,A+1,B+1,C+1),....]`
- **Miners** provide record keeping for the block as a service by calculating the **Proof of Work** (POW) for the block
- Miners **compete** to find the PoW: doing so gets them transaction fees from Alice and others on the transaction AND earns them a small reward - newly minted Bitcoins

Proof of Work

See: https://www.blockchain.com/btc/block/660000

- Block has transaction data in some format
- Block also contains hash of previous block (the "chain")
- Proof of Work for the block is a cryptographic hash of
  - The block data
  - A unique identifier/key for the miner
  - A **nonce**, number that needs to be found

  Such that the hash value has N leading 0's
- Example:

  `0000000000000000008eddcaf078f12c69a439dde30dbb5aac3d9d94e9c18f6`
- Bitcoin adjusts N every 14 days so that miners can find PoW for 1 block every 10 minutes

Basic Approach for Proof of Work

```
int find_nonce(int my_id,
               int *blockdata[], int len,
               int difficulty)
{
  for(int nonce=1; ; i++){
    int hash = hash_data(my_id, nonce, blockdata, len);
    int lead_zeros = count_lead_zeros(hash);
    if(lead_zeros >= difficulty){
      return nonce;
    }
  }
  return 0;
}
```

Where is the parallelism here?

Bitcoin has Issues

The New York Times

Bitcoin Uses More Electricity Than Many Countries. How Is That Possible?

In 2009, you could mine one Bitcoin using a setup like this in your living room.

↓

↓

Amount of household electricity required to mine one coin: **a few seconds' worth**. Bitcoin's value: **basically nothing**.

Today, you'd need a room full of specialized machines, each costing thousands of dollars.

↓

↓

Amount of household electricity required: **9 years' worth**. (Put in terms of a typical home electricity bill: **about $12,500**.) Value

Source: NYTimes 9/3/2021

- Miners receive transaction fees + reward for keeping the block chain publicly verified
- Requiring a certain amount of work to verify blocks means as more miners enter the scheme, more competition results and more power is spent
- Leads to explosion of power usage just to brute force a bunch of hashes
- Some schemes have been suggested for alternate system designs but little movement yet for the main cryptocurrencies