

CSE 160

Lecture 12

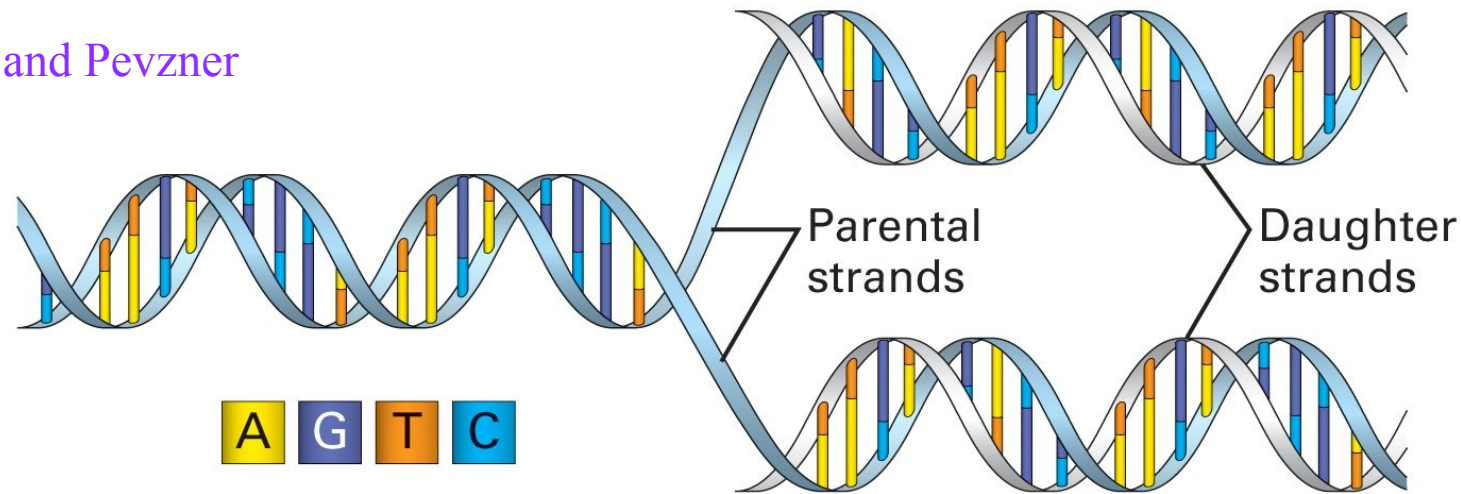
Protein Sequence Querying
Wave front methods

Announcements

- No Lab in APM this Friday

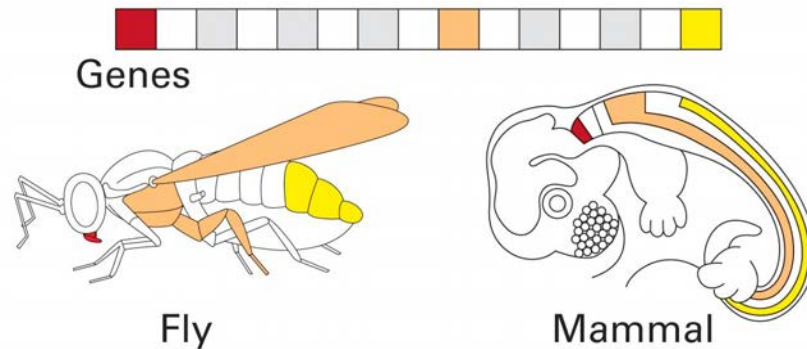
DNA: The Code of Life

Jones and Pevzner



- The structure & 4 genomic letter codes for all living things
- Adenine, Guanine, Thymine, and Cytosine which pair A-T and C-G on complimentary strands
 - Human & mouse genomes: ~ 3 billion DNA base pairs
- Human genome has 24 distinct chromosomes, each contains many **genes**, basic physical and functional units of heredity
- **A gene**: pattern of DNA bases encoding instructions for making **proteins**
- **Proteins** are large, complex molecules made up of smaller subunits called **amino acids**

DNA and the Genetics Makeup



- Genes are inherited and are expressed
 - **genotype** (genetic makeup)
 - **phenotype** (physical expression)
- On the left, is the eye's phenotypes of green and black eye genes.



All Life depends on 3 critical molecules

- DNAs
 - Hold information on how a cell works
- Proteins
 - Form enzymes that send signals to other cells and regulate gene activity
 - Form body's major components (e.g. hair, skin, etc.)
- RNAs
 - Act to transfer short pieces of information to different parts of a cell
 - Provide templates to synthesize into protein

Sequence alignment (follows Kaminsky)

Initial sequencing and comparative analysis of the mouse genome

Nature 420, 520-562 (5 December 2002)

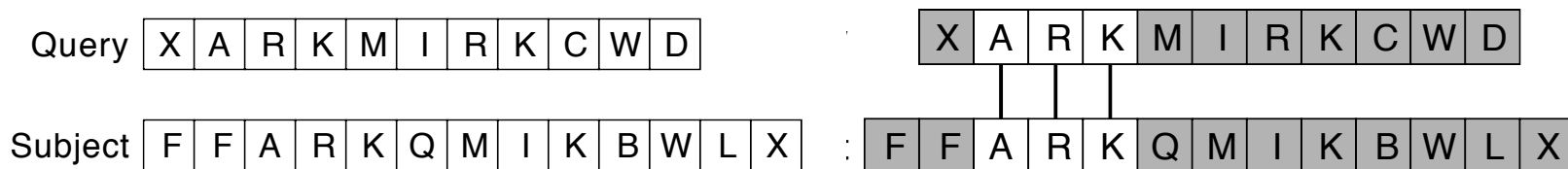
Mouse Genome Sequencing Consortium

- The sequence of the mouse genome is a key tool for understanding the contents of the human genome and for biomedical research
- Comparative analysis of the mouse and human genomes, describing some insights that can be determined from the two sequences
- Comparative genomics allows one to **read evolution's laboratory notebook**. In ~75 million years since the divergence of human and mouse, evolution has altered genome sequences and caused them to diverge ...
- At the nucleotide level **~40% of human genome can be aligned to the mouse genome**. These sequences seem to represent most of the orthologous sequences (speciation) remaining in both lineages from the common ancestor, with the rest likely to have been deleted in one or both genomes.



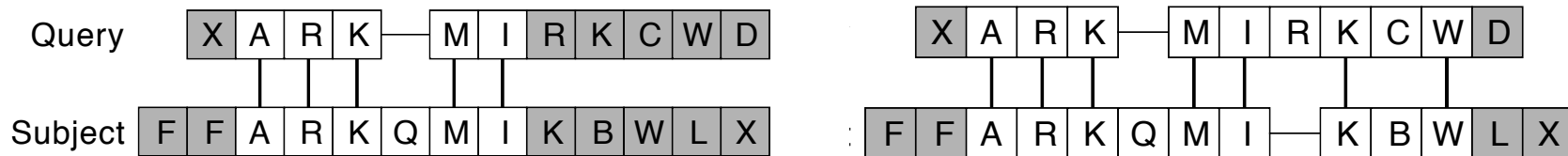
Protein sequence alignment

- Represent proteins as character strings over a 20 symbol alphabet
- Each symbol represents an amino acid
- Goal: form an *alignment*; compare two sequences to gauge their similarity
- Score: $\alpha = +2$ for match $\beta = -1$ for mismatch
- Exclude points outside the sequence



Local alignments and gaps

- We may be able to increase the score by creating *gaps*: $\gamma = -1$
- Score = +11
 - 7 matches, 1 mis-match, 2 gaps



Automation

- There are too many combinations to test by hand
- The Smith-Waterman algorithm (1981) automated procedure to determine the best possible alignment

Smith Waterman algorithm

- Let A be the query sequence, a_i , the i^{th} symbol, and similarly with B and b_j
- $S[i,j]$ = best possible alignment score when a_i is aligned with b_j
- Suppose we have computed the alignment up to, but not including, a_i and b_j
- We therefore know $S[0,0]$ through $S[i-1,j-1]$, $S[i,j-1]$, $S[i-1,j]$
- We compute $S[i,j]$ using a simple formula

The next alignment score

- 4 cases
 - ▶ a_i aligns with b_j
 $S[i,j] = S[i-1,j-1] + \alpha$ if match, else β
 - ▶ Introduce a gap in B
 $S[i,j] = S[i-1,j] + \gamma$
 $S_{i,j} = \text{Max}(S_{i,j} + f(a_i, b_i), S_{i-1,j} + \gamma, S_{i,j-1} + \gamma)$
 - ▶ Introduce a gap in A
 $S[i,j] = S[i,j-1] + \gamma$
 $a_i = b_i \rightarrow f(a_i, b_i) = \alpha$
 $a_i \neq b_i \rightarrow f(a_i, b_i) = \beta$
 - ▶ Neither a nor b are in the alignment: $S[i,j] = 0$
- We take the maximum value
- We do this for all i and j

Filling in the alignment matrix

- Set up a matrix to keep track of the work
- We continue for all i and j
- What algorithmic motif is this?

$$S_{i,j} = \text{Max} \left\{ \begin{array}{l} S_{i,j} + f(a_i, b_i) \\ S_{i-1,j} + \gamma \\ S_{i,j-1} + \gamma \end{array} \right\}$$

$$a_i = b_i \rightarrow f(a_i, b_i) = \alpha$$

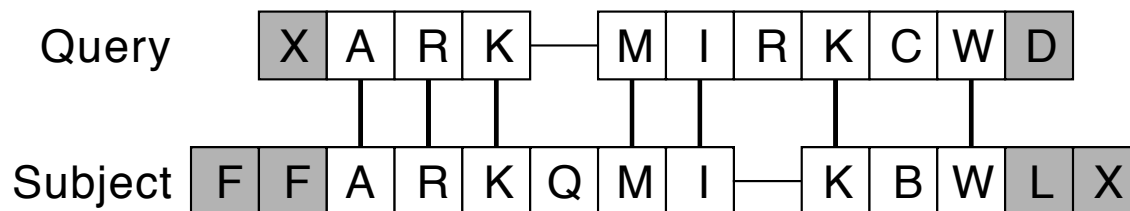
$$a_i \neq b_i \rightarrow f(a_i, b_i) = \beta$$

		Subject Sequence														
		F	F	A	R	K	Q	M	I	K	B	W	L	X		
Query Sequence	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	A	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1
	R	0	0	0	2	1	0	0	0	0	0	0	0	0	1	2
	K	0	0	0	1	4	3	2	1	0	0	0	0	0	0	3
	M	0	0	0	0	3	6	5	4	3	2	1	0	0	0	4
	I	0	0	0	0	3	6	5	7	6	5	4	3	2	1	5
	R	0	0	0	0	2	5	5	7	6	5	4	3	2	1	6
	K	0	0	0	0	1	4	4	6	9	8	7	6	5	4	7
	C	0	0	0	0	2	3	3	5	8	8	7	6	5	4	8
	W	0	0	0	0	1	4	3	4	7	10	9	8	7	6	9
	D	0	0	0	0	0	3	3	3	6	9	9	8	7	6	10
		0	0	0	0	0	2	2	2	5	8	8	11	10	9	11
	0	1	2	3	4	5	6	7	8	9	10	11	12	13		

Traceback

- Maximum score yields the best alignment:
symbol a_{10} is aligned with subject b_{11}
- Compute *traceback* to determine the
remaining match pattern

		Subject Sequence														
		F	F	A	R	K	Q	M	I	K	B	W	L	X		
Query Sequence	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	A	0	0	0	2	1	0	0	0	0	0	0	0	0	1	2
	R	0	0	0	1	4	3	2	1	0	0	0	0	0	0	3
	K	0	0	0	0	3	6	5	4	3	2	1	0	0	0	4
	M	0	0	0	0	2	5	5	7	6	5	4	3	2	1	5
	I	0	0	0	0	1	4	4	6	9	8	7	6	5	4	6
	R	0	0	0	0	2	3	3	5	8	8	7	6	5	4	7
	K	0	0	0	0	1	4	3	4	7	10	9	8	7	6	8
	C	0	0	0	0	0	3	3	3	6	9	9	8	7	6	9
	W	0	0	0	0	0	2	2	2	5	8	8	11	10	9	10
D	0	0	0	0	0	1	1	1	4	7	7	10	10	9	11	
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	

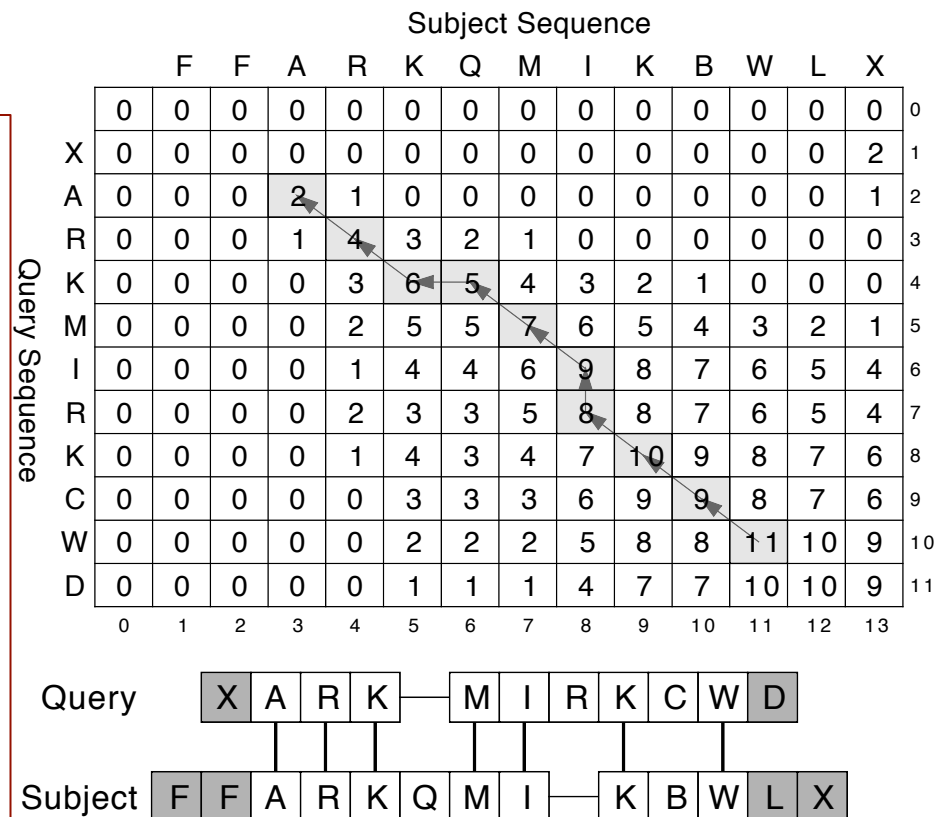


Computing the Traceback

- Start with the maximum score:
 a_{10} is aligned with b_{11}
- Work backwards to the start of A and B

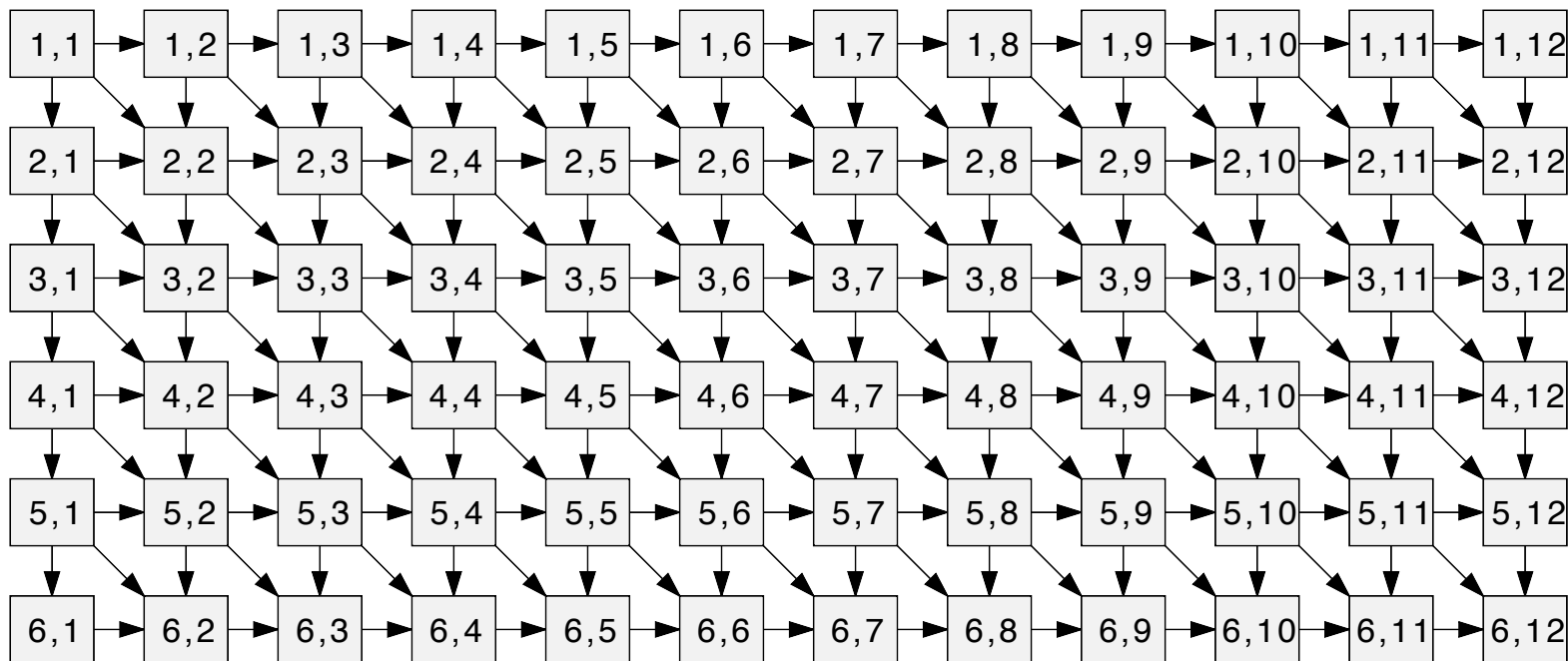
```

(i,j) = largest entry
while  $S_{i,j} \neq 0$ 
  If  $a_i = b_j$   $\delta = \alpha$  else  $\beta$ 
  If  $S_{i,j} = S_{i-1,j-1} + \delta$ 
     $a_i$  aligned with  $b_j$ 
     $(i,j) \leftarrow (i-1,j-1)$ 
  Else if  $S_{i,j} = S_{i-1,j} + \gamma$ 
     $a_i$  aligned with gap in B
     $(i,j) \leftarrow (i-1,j)$ 
  Else
     $b_j$  aligned with gap in A
     $(i,j) \leftarrow (i,j-1)$ 
  
```



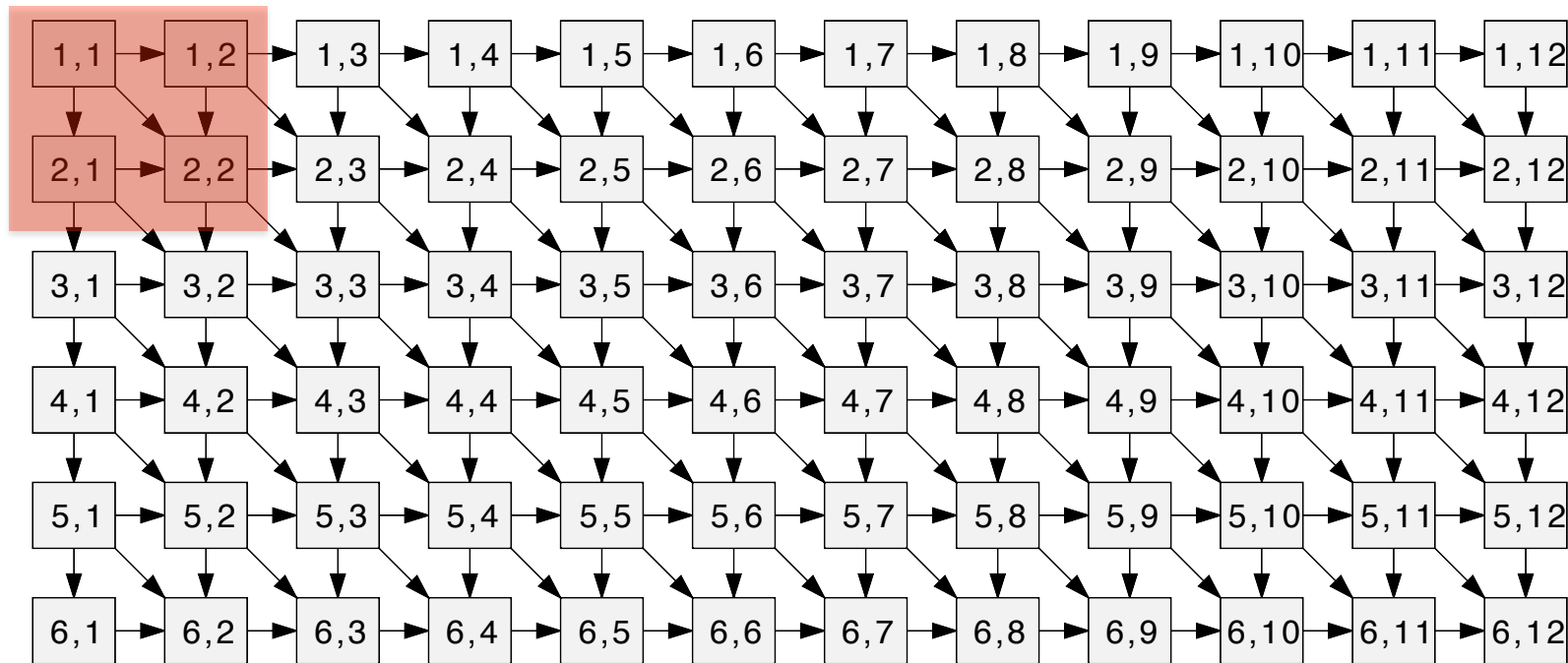
Computing the traceback in parallel

- We must preserve the dependencies to ensure correctness
- How can we realize parallelism?



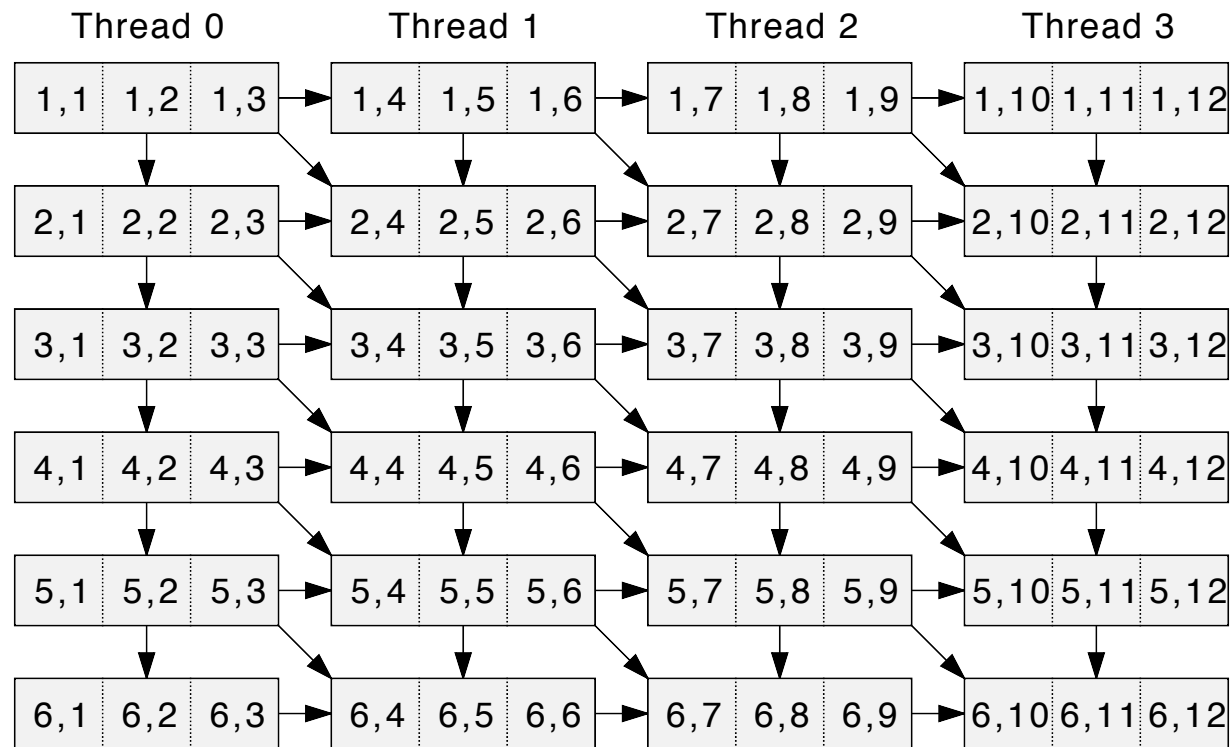
Insight

- If we group the entries into blocks, we maintain the same dependence structure, but at a coarse level



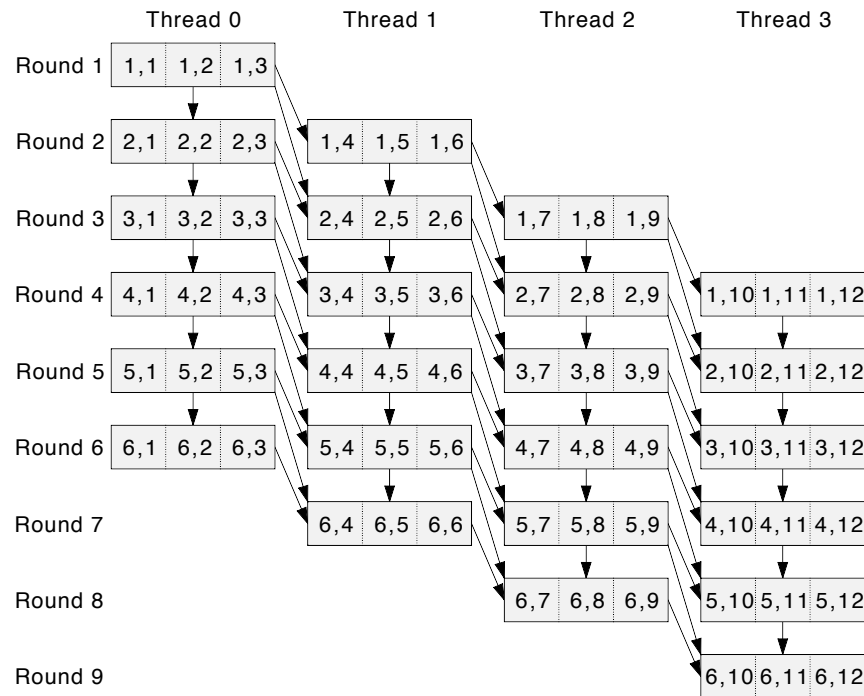
Parallelization strategy

- Let's partition the columns of S across the threads



Computational Rounds

- Redraw the diagram to uncover parallelism
- We organize into rounds
- At round 4, all 4 threads are engaged



Fin

In class exercises

Questions

1. ~~Iteration to thread mapping~~
2. ~~Removing data dependencies~~
3. ~~Dependence analysis~~
4. ~~Time constrained scaling~~
5. Tree Summation
6. Performance

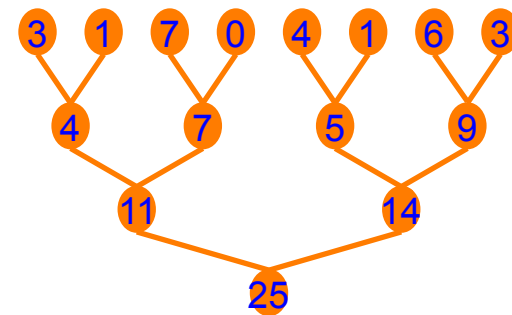
5. Tree Summation

- Input: an array $x[]$, length $N \gg P$
- Output: Sum of the elements of $x[]$
- Goal: Compute the sum in $\lg P$ time

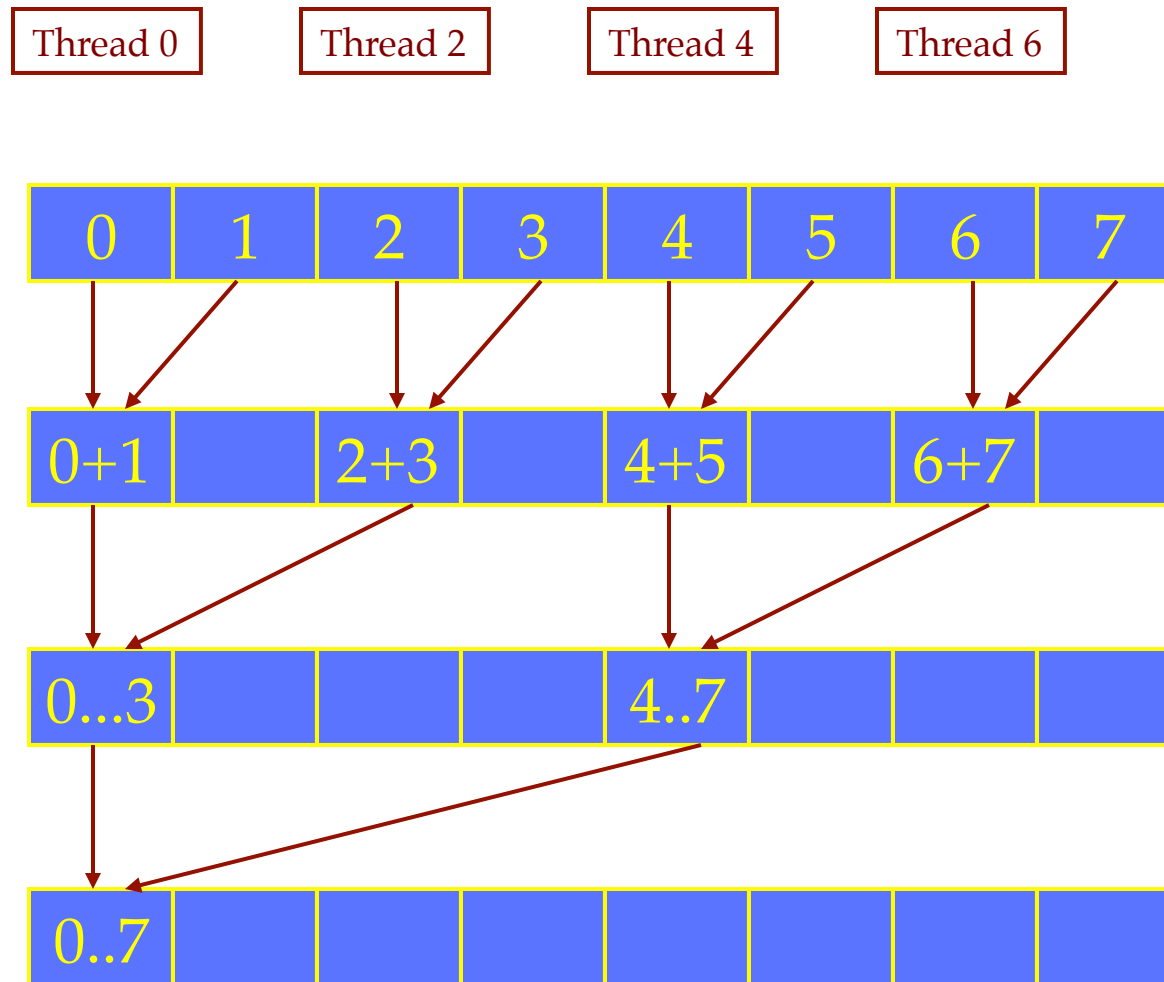
```
sum = 0;  
for i=0 to N-1  
    sum += x[i]
```

- Assume P is a power of 2, $K = \lg P$
- Starter code
for $m = 0$ to $K-1$ {

}



Visualizing the Summation



6. Performance

- You observe the following running times for a parallel program running a fixed workload N
- Assume that the only losses are due to serial sections
- What is the speedup and efficiency on 8 processors?
- What will the running time be on 4 processors?
- What is the maximum possible speedup on an infinite number of processors?
- What fraction of the total running time on 1 processor corresponds to the serial section?
- What fraction of the total running time on 2 processors corresponds to the serial section?

NT	Time
1	10000
2	6000
8	3000