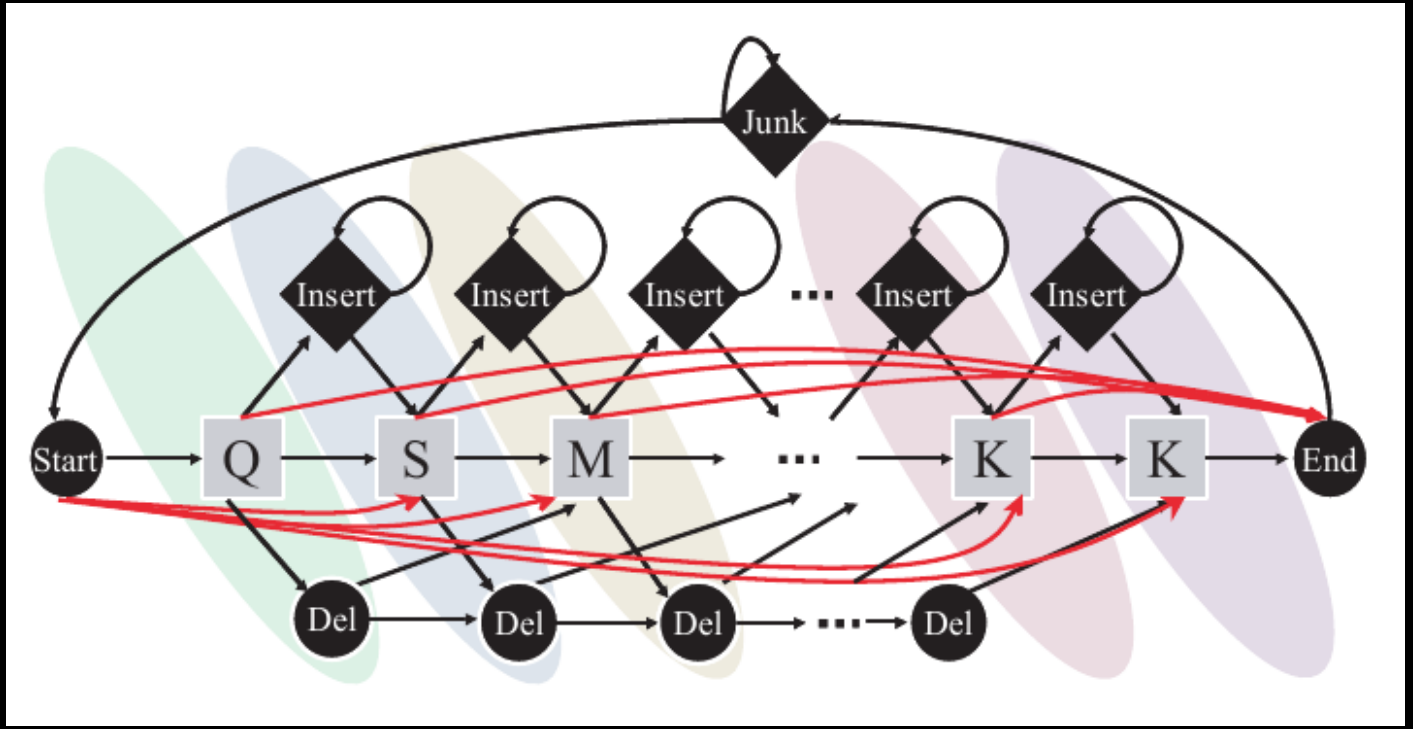
Specialized Hardware for Read Mapping

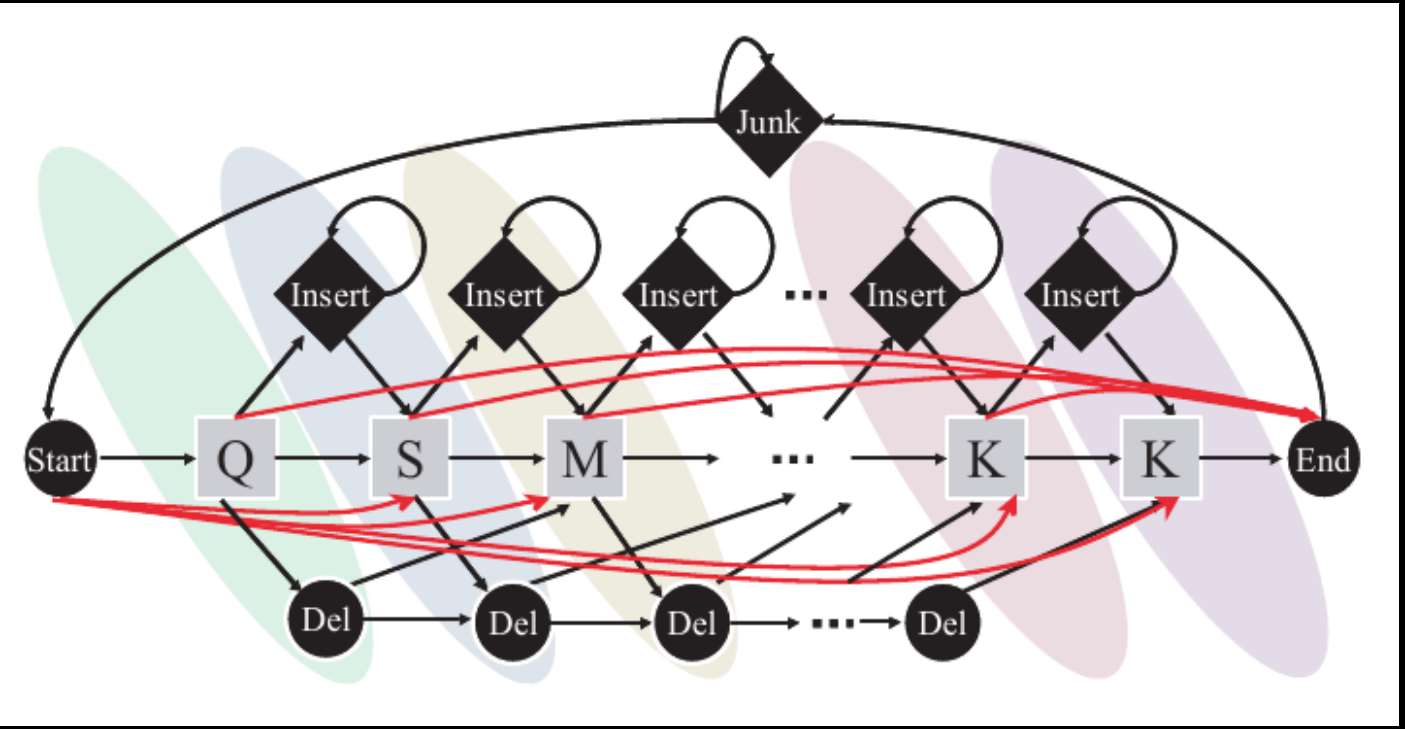
Tim Smith (tsmith@cs)

2008/02/27

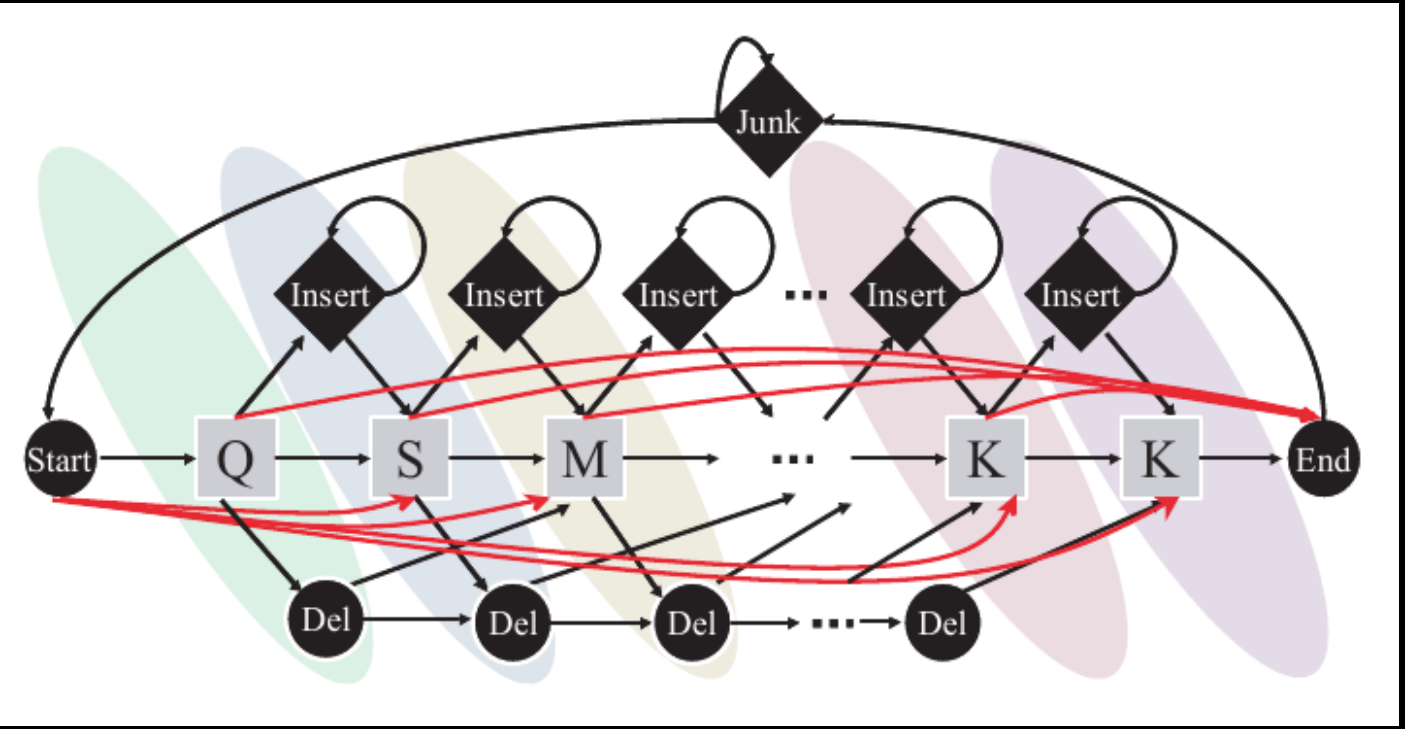
ClawHMMER is based off of HMMer (Eddy et al.)



HMMer uses HMMs to determine the probability of a protein sequence belonging to a protein family

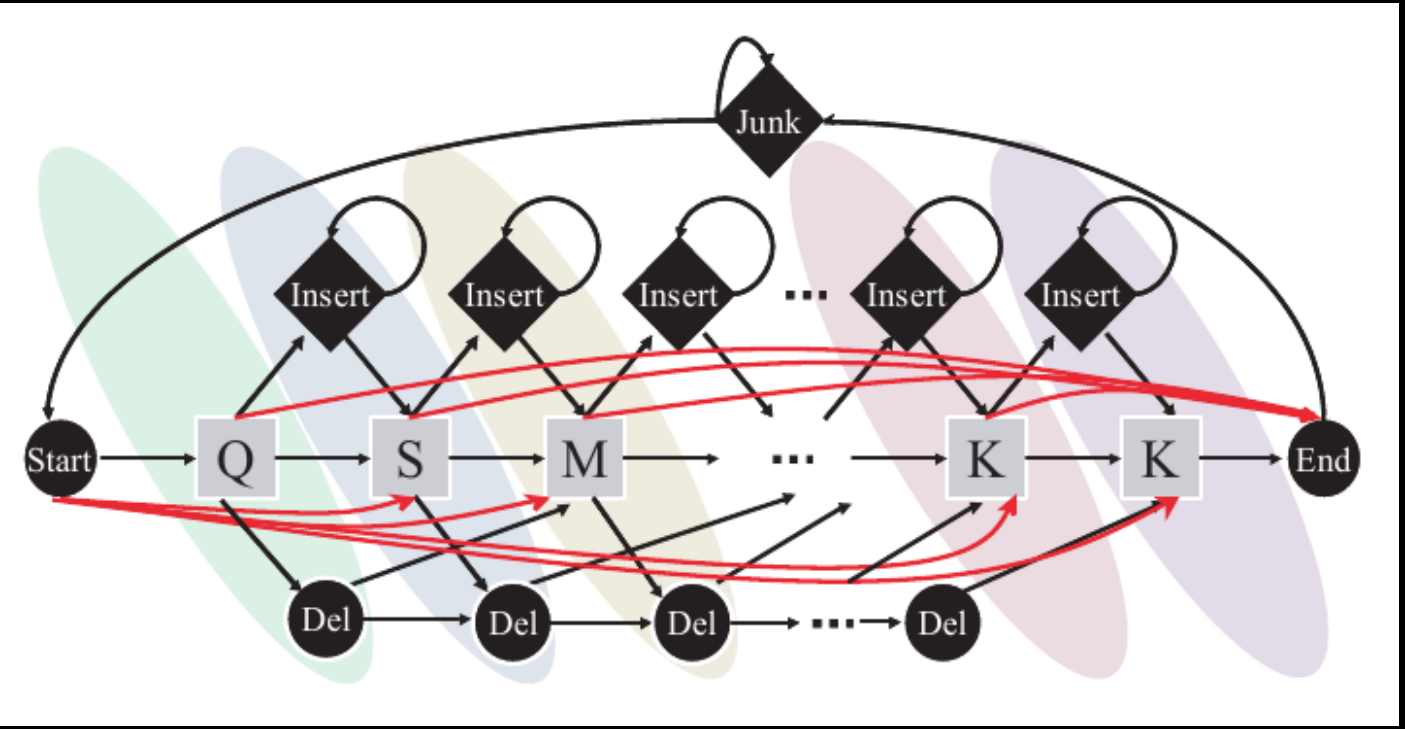


Step 1: Build a “Profile HMM” from a multiple alignment of protein sequences



“Normal” states represent amino acids; transition probabilities are calculated using Bayesian probability theory

Step 2: Use Viterbi algorithm to find "matching" domains in one or more search sequences



ClawHMMer re-implements Step 2 using “Stream Processing” (GPGPU)

Have:

```
for i = 1 to length(observ): //observation loop
    for s in states:         //state loop
        for t in states: //transition loop
            tmp=p_transition(t,s)·prob[i-1][t]
            prob[i][s]=max(prob[i][s],tmp·p_emit(s, observ[i]))
call traceback(prob[][])
```

Need:

```
input stream i,j
output stream o

for each element of i:
    for each element of j:
        o = i <op> j
    end for
end for
```

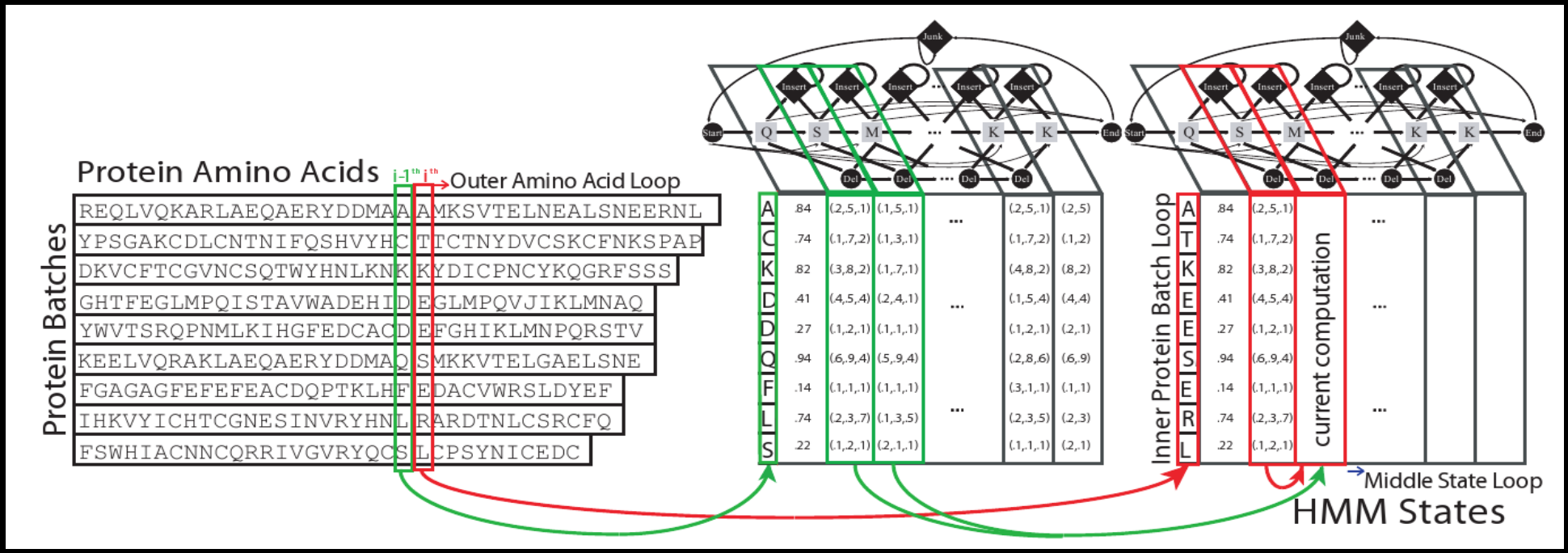
Stream processes must minimize data dependencies and maximize “arithmetic intensity”

```
input stream i,j
output stream o

for each element of i:
    for each element of j:
        o = i <op> j
    end for
end for
```

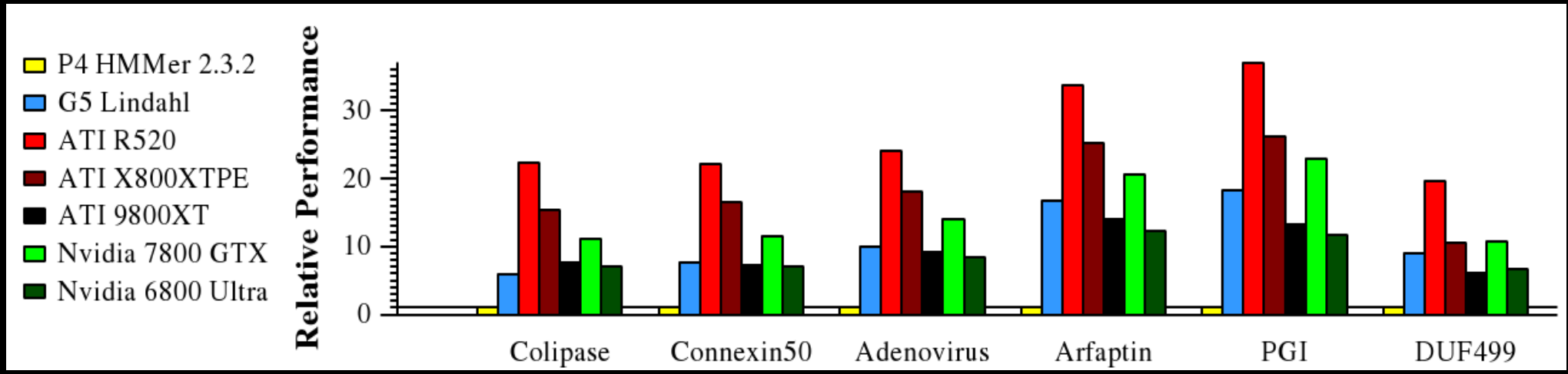
o cannot depend on i or j

<op> must require enough computation to “hide” the latency of memory fetches for i and j, as well as the store of o

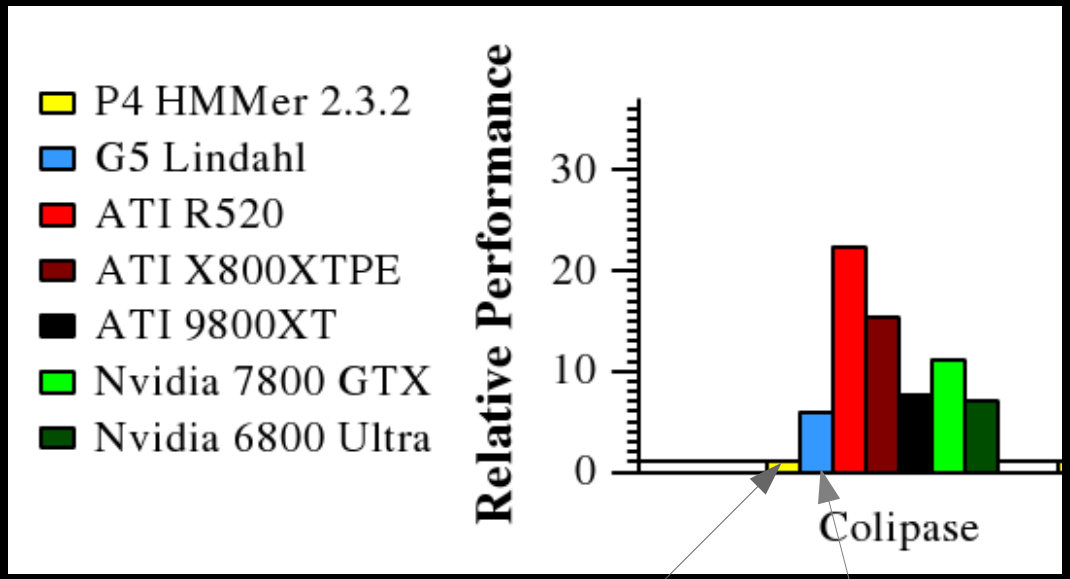



```
for i = 1 to length(longest protein):        //observation loop
  for s in greyMatchStates:                  //state loop
    j = previousInsertState(s)
    q = previousDeleteState(s), d = nextDeleteState(s)
    parallel_for protein in database: //database loop
      curProb=allProb[protein][i mod 2]
      prevProb=allProb[protein][(i + 1) mod 2]
      //transition loop (unrolled)
      tmp_s=max(s's 4 incoming trans. in prevProb)
      curProb[s]=tmp_s
      curProb[d]=max(tmp_s · p_trans(),curProb[q]·p_trans())
      curProb[j]=max(j's 2 incoming trans. in prevProb)
      curProb[end]=max(prevProb[end],tmp_s·p_trans())
for protein in database:
  if (allProb[protein][last][endstate]>globalThreshold):
    call generalViterbi(protein) for traceback
```

The results were promising



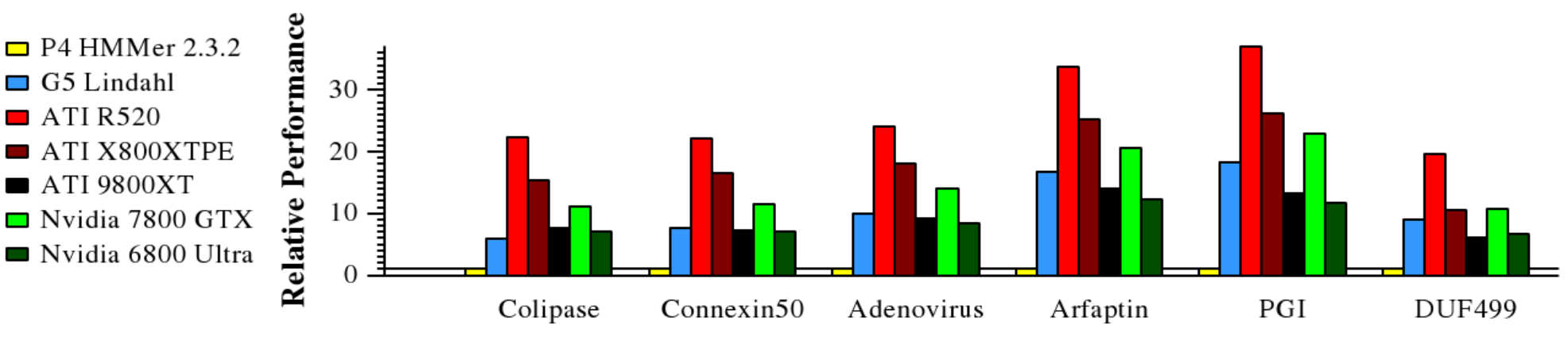
On small HMMs, ClawHMMer was 3-25 times as fast as CPU implementations



P4 implementation barely registers

Optimized PowerPC implementation 3x slower than fastest GPU

As HMMs get larger, the speed-up gets smaller



Architecture	Best Codepath	Colipase 139 states Time(s)	Connexin50 211 states Time(s)	Adenovirus 427 states Time(s)	Arfaptin 739 states Time(s)	PGI 1543 states Time(s)	DUF499 3271 states Time(s)
P4 2.8 GHz	HMMer 2.3.2	1589.1	2372.8	5030.6	12236.9	28423.6	29601.7
G5 2.5 GHz	HMMer 2.3.2	539.9	729.0	1300.5	2229.1	4547.9	11778.4
G5 2.5 GHz	Lindahl	279.4	320.8	546.0	836.5	1574.9	3341.6
R520	Brook 4-output	71.48	107.1	209.0	362.6	768.9	1508.4
X800XTPE	Brook 4-output	107.5	148.5	301.0	555.5	1088.9	2800.6
9800XT	Brook 2-output	217.2	324.2	594.4	988.7	2175.9	4861.3
7800 GTX	Brook 4-output	141.9	206.4	361.4	594.3	1236.1	2747.2
6800 Ultra	Brook 4-output	233.1	330.4	643.1	1142.8	2451.0	4461.0

GPUs and CPUs have progressed since 2005

Architecture
P4 2.8 GHz G5 2.5 GHz G5 2.5 GHz
R520 X800XTPE 9800XT
7800 GTX 6800 Ultra

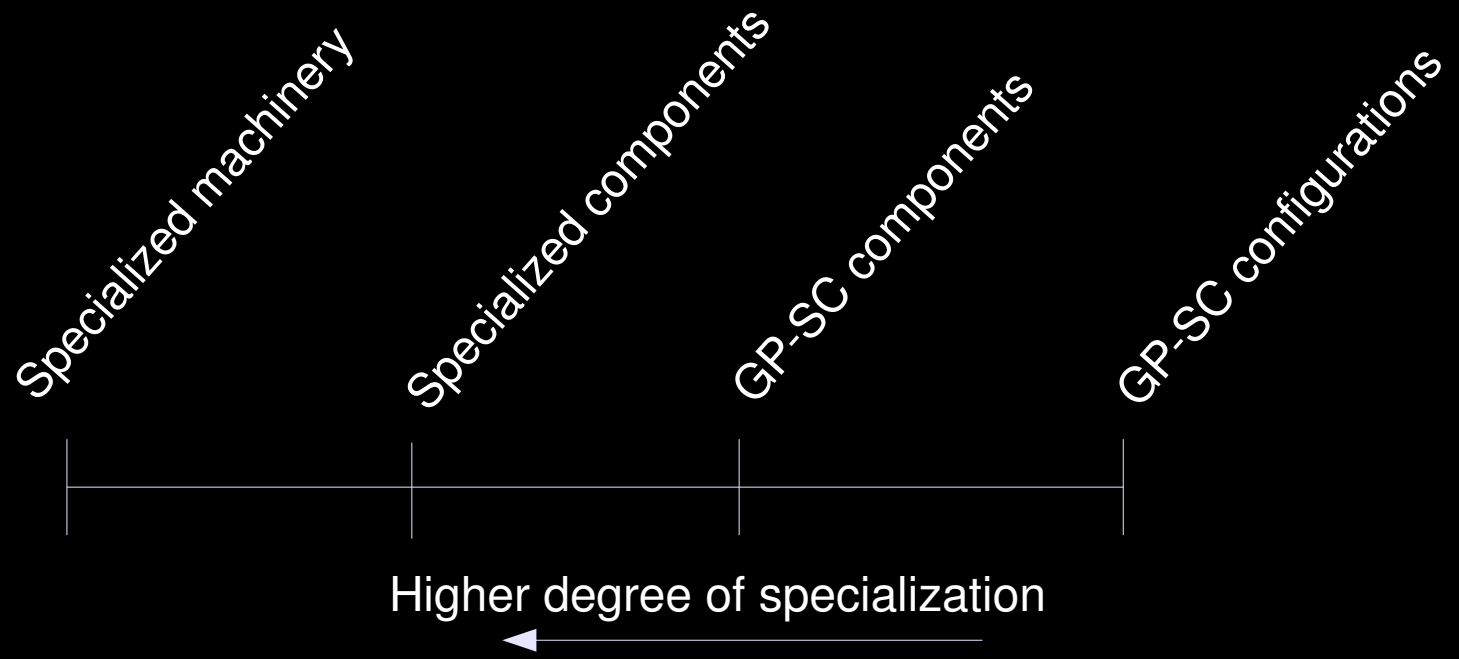
P4 has been replaced with dual, quad-core P3

2005 GPU: 16 ALUs, 256MB VRAM

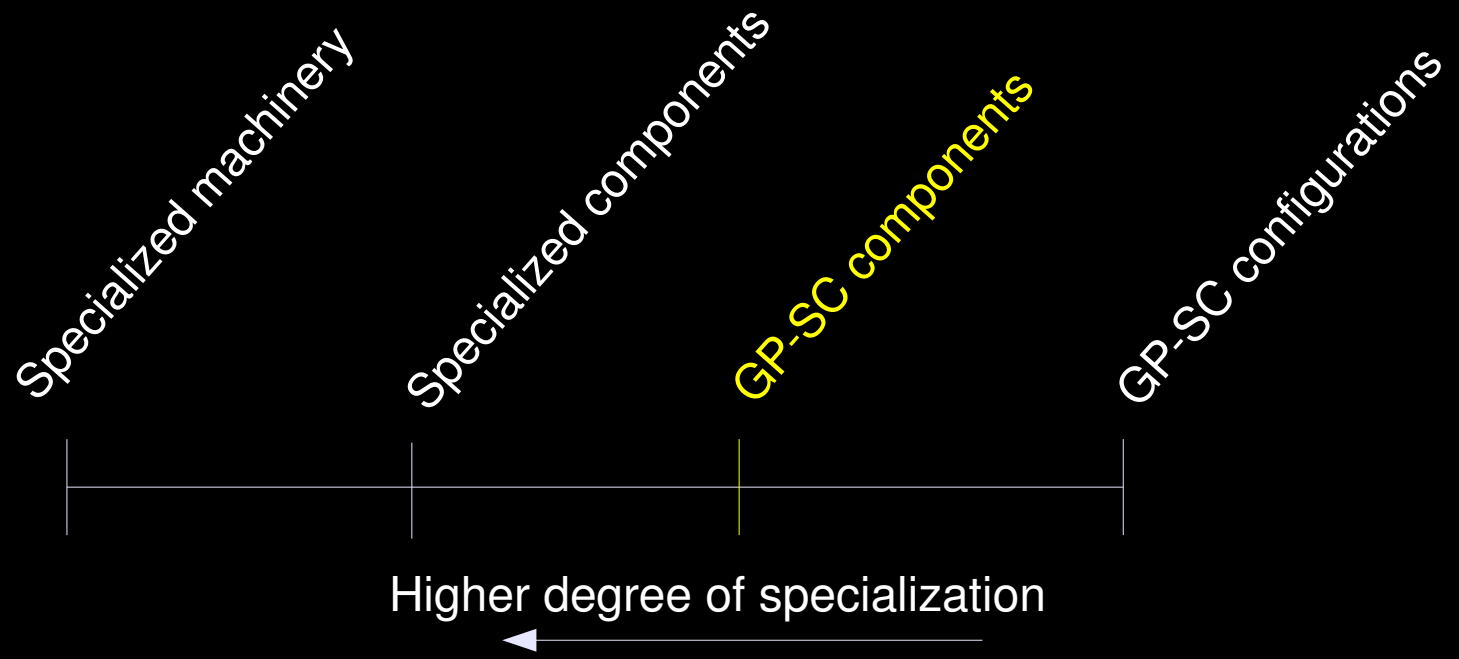
2008 GPU: 320 ALUs, 1GB+ VRAM

GPUs have advanced at a greater rate!

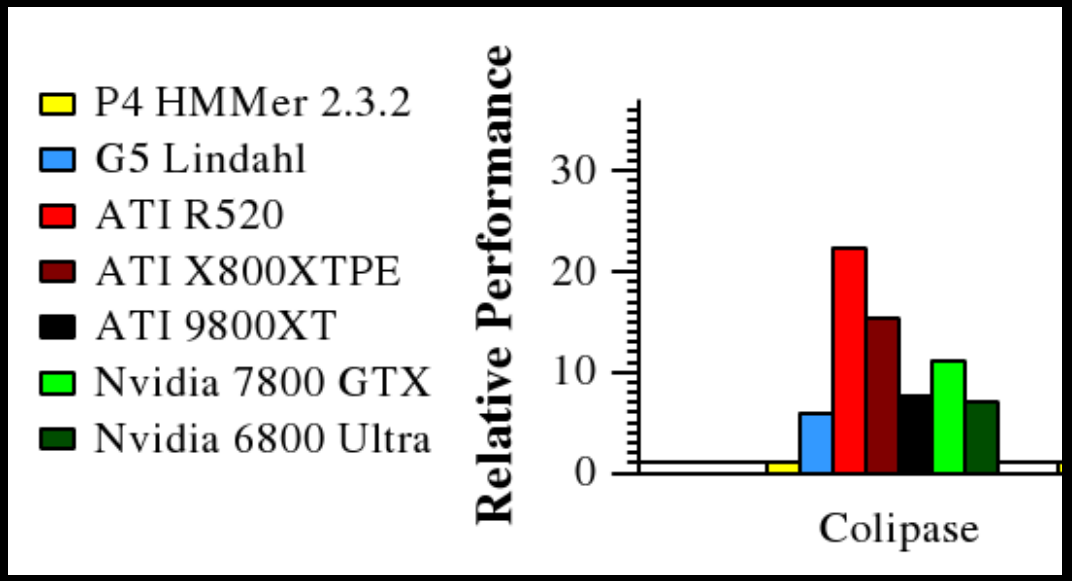
Specialization is a spectrum



ClawHMMer uses general-purpose scientific computing components



Specialization offers the advantage of increased speed



Specialization comes at the price of improved cost, complexity, and development effort, while reducing flexibility

```
for i = 1 to length(observ): //observation loop
    for s in states:         //state loop
        for t in states: //transition loop
            tmp=p_transition(t,s)·prob[i-1][t]
            prob[i][s]=max(prob[i][s],tmp·p_emit(s, observ[i]))
call traceback(prob[][])
```

```
for i = 1 to length(longest protein):      //observation loop
  for s in greyMatchStates:                //state loop
    j = previousInsertState(s)
    q = previousDeleteState(s), d = nextDeleteState(s)
    parallel_for protein in database: //database loop
      curProb=allProb[protein][i mod 2]
      prevProb=allProb[protein][(i + 1) mod 2]
      //transition loop (unrolled)
      tmp_s=max(s's 4 incoming trans. in prevProb)
      curProb[s]=tmp_s
      curProb[d]=max(tmp_s · p_trans(),curProb[q]·p_trans())
      curProb[j]=max(j's 2 incoming trans. in prevProb)
      curProb[end]=max(prevProb[end],tmp_s·p_trans())
for protein in database:
  if (allProb[protein][last][endstate]>globalThreshold):
    call generalViterbi(protein) for traceback
```

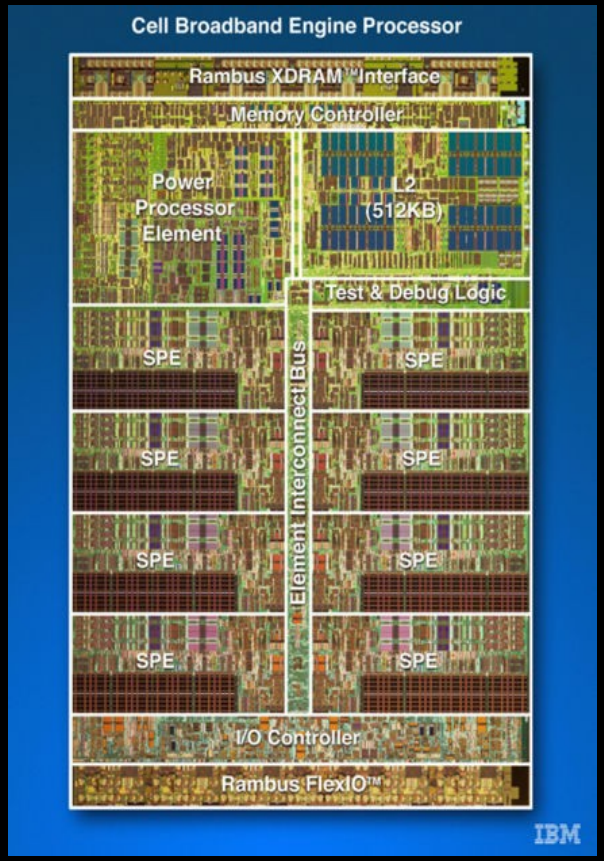
Specialized HW = Not commercially viable

- “Selling accelerators for BLAST/Smith-Waterman/HMMR algorithms to the biotech market is a tough nut to crack. They they already have huge commodity x86 farms for doing this stuff. We have been told by numerous people that their bioinformatics queries are 'cheap enough' and 'fast enough' already.”

  – Beyond3D GPGPU forum post

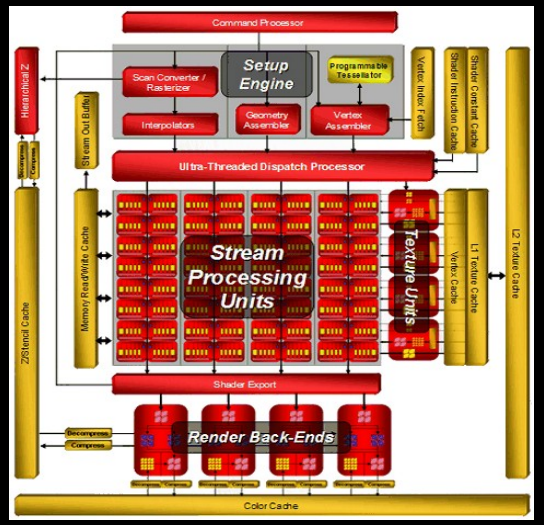
CPUs have only one direction to go

Four cores



Nine cores

.....



60 cores (320 ALUs!)

Today's “specialized hardware” is tomorrow's commodity hardware

Specialized HW 20 years ago

Specialized HW today

Questions?