Communication and Wall clock time

- Each processor spends CPU time on a job.
- There is also time spent in communication between processors.
- Wall clock time is the temporal duration of a job, which is determined by at what time the job started and the time it ends.
- Hence wall clock time includes the CPU time for the last processor to finish its job and the communication time that causes delays on when a job is started.

Handout 10-12-11

How would you structure a solution of a heuristic in parallel?

- Assume for example we do a GA
  - that it takes F= 1 minute (60 seconds) to evaluate fitness Cost(s) once and
  - 10 seconds to do the heuristic steps for crossover and mutation and select best solution to be used in the next iteration and
  - one second to communicate between processors.
- Assume in each iteration, you want to have a population of 20

Some Definitions for Example Problem

- F– CPU time for one fitness evaluation (1 min)
- C—communication time (1 sec)
- G- time for crossover, mutation, generate new population (10 sec)
- P- number of processors (5)
- GA pop = size of GA pop (20)
- m- number of individuals S for which fitness is computed on each processor
  - How to compute m?

What do we want to calculate?

- Wall Clock Time to compute answer in serial
- Wall ClockTime to compute answer in parallel
- Speed up – $\frac{t_{serial}}{t_{parallel}}$
- Efficiency
- Is this coarse or fine grained parallelism?

(see next graph at back of handout)

Parallel Speed Up, Efficiency and Cost

- $T(p)$ = wallclock time taken to solve a problem with p processors

- Speed up with p procesors $S(p)= T(1)/T(p)$

- Efficiency $E(p)= S(p)/p$ (often expressed as a percentage)
- $0 \leq E(p) \leq 1$ (usually)

- If CPU cost = # of processors* $T(p)$, then

  $Cost(p)= cost(1)/efficiency$

So CPU cost will increase with more processors. Why then would we use $p>1$?

GA in Parallel

time | Time

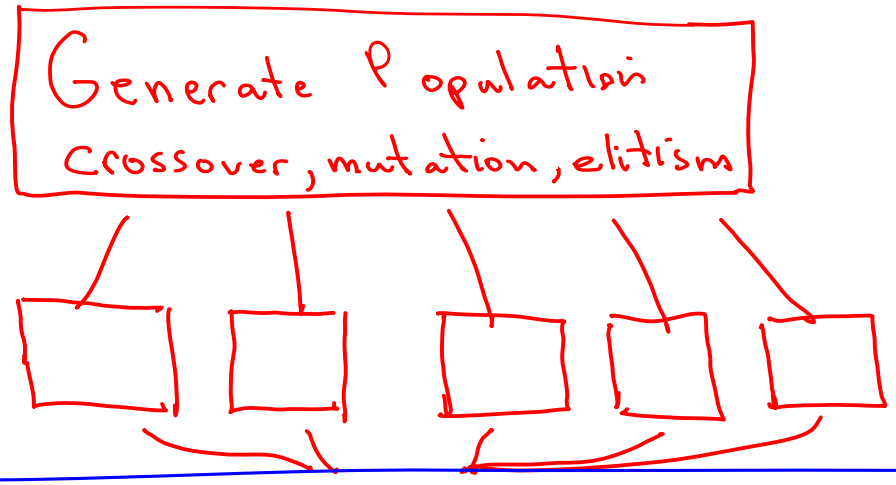
Serial 10 sec

Communication 1 sec

Parallel (60 sec) m

Communication $\frac{1}{2}$ sec

Iteration 1



serial 10 sec

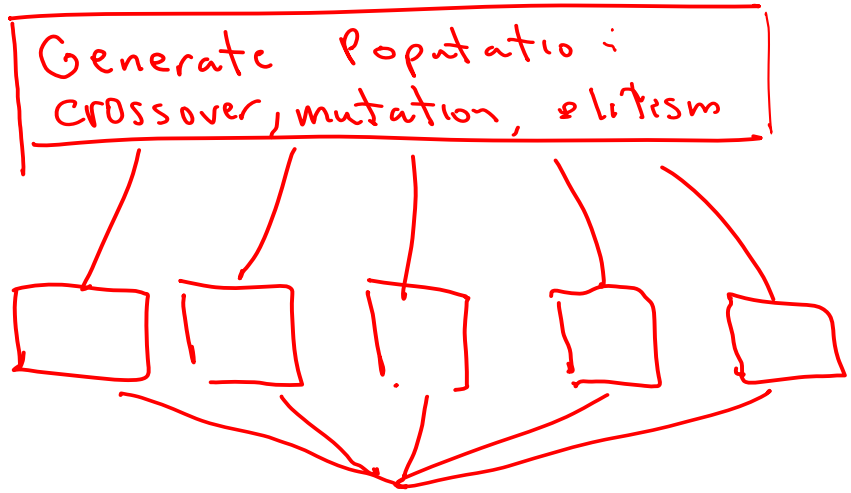
time

Communication $\frac{1}{2}$ sec

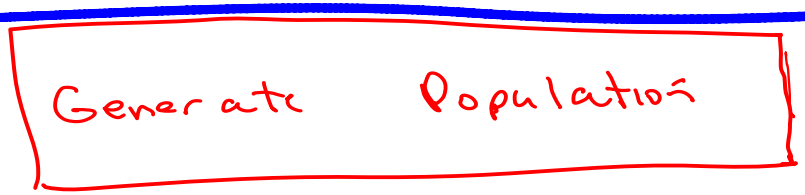
Parallel 60 sec

Communication $\frac{1}{2}$ sec

Iteration 2



Iteration 3

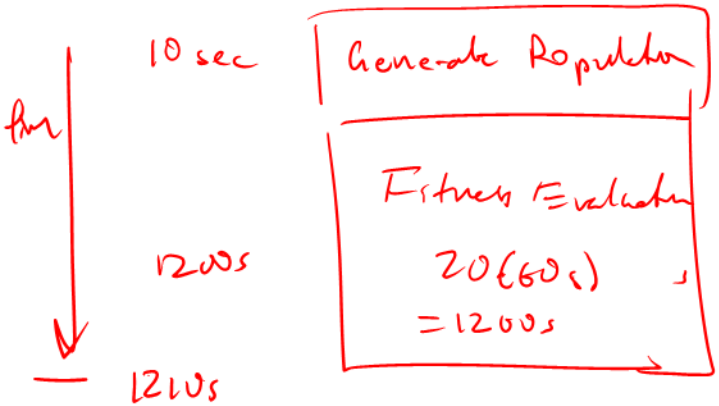


recall mP = GA population size, P = # processors

m = number of fitness functions done per processor

P=5, GA pop= 20, communication =1/2 sec, fitness eval = 60 sec., gen pop=10 sec.

Serial GA



Engineer Time vs CPU Time:
\$100/hr \$0.05/hr

$$\Rightarrow \text{speedup} = \frac{\text{serial}}{\text{parallel}} = \frac{1210}{252} = 4.8$$

$$\text{efficiency} = \frac{\text{speedup}}{\#\text{processors}} = 96.6\%$$

$\neq 100\%$ because of comm time

Time per iteration in parallel (5-processor)

Wall Clock Time: $2C + mR + G$ ← serial time

communication time

# of evals/processor

fitness eval time

if #processors = 5,
pop = 20,
$m = \frac{20}{5} = 4$

$$= 2(1) + 4(60s) + 10$$

$$= 252s \ll 1210s$$

$\Rightarrow$ parallel better than serial iff $R \gg C$

- Additional measures of parallel efficiency:
  - Total CPU time used to solve problem (usually the basis for computing charge)
  - %Overhead
- Assume that calculation of J(s) is fairly time consuming. Do you think that heuristic methods are coarse or fine grained problems?
- What would happen if the amount of time it takes to solve one simulation is variable?
- How do you determine how many processors you will use?

Parallelism on Your computer

- There are two ways parallelism affects your computation:
  - Operating System—if you have a computer with multiple cores on one or more chips, your operating system will automatically (e.g. without instructions from you) do mulitple jobs on multiple processors (e.g. allow you to read email while it is downloading a number of files to your external drive).
  - In this case your computer does not know what things it is going to be asked to do milleseconds into the future so it might not be the most efficient.

- Parallelism you control through your programming-
  - You can use languages like “MPI” or parallel Matlab to direct the computation done in your algorithm.
  - This is what you would do to implement a parallel Genetic Algorithm or simulated annealing search procedure. You would instruct you code to

Programming a GA in Parallel

- 1. You instruct the code to partition the decision variables in the population into P equally sized packets.
- 2. Then each of these packets are sent to one of the processors (cores) which is responsible for computing the fitness functions. Store these values
- 3. In serial you perform tournament/roulette selection, crossover, and mutation to generate the new population and then go to 1.

Efficiency

- When you write your own parallel code, it is expected to be vastly more efficient that the automatic code that exists on your computer .
- This is because you have told the computer how to partition the work in a way that is efficient relative to your algorithm.

What kinds of algorithms are efficient in parallel?

- You want to be able to efficiently distribute the expensive parts of the calculation among many parallel processors.
- For somewhat expensive fitness functions, Genetic Algorithms in parallel are efficient in comparison to genetic algorithms in serial because GAs do many fitness evaluations simultaneously and so these can be distributed among processors.
- How well do you expect parallel processing to do with simulated annealling or tabu search.
- What is the relationship between the length of the binary string and parallel processing?