Parallel Evolutionary Computation

Motivation

- EA requires many evaluations and those evaluations can be expensive
- Easy to parallelize:
  - Independent fitness calculations
  - Parallel evolution of multiple populations
- Parallel evolution is reported as:
  - More reliable
  - Better solutions
  - Super-linear speedups

Overview

- Global parallelism: single population master-slave
- Multiple populations. Coarse grained.
- Many populations. Fine-grained.
- Hierarchical/Hybrid combinations.

Global Parallelism

- Single population. A master controller to execute main generation loop.
- Slaves evaluate the operations, most cases the fitness evaluations.
- Identical to single processor EA except the execution time.
- Synchronous vs. Asynchronous

Coarse Grained

- Multiple subpopulations (demes)
- Migration of individuals among demes.
- Different topologies and migration strategies.

Fine Grained

- Many subpopulations with small number of individuals.
- Populations sizes can be as small as 1.
- Spatial structure of connections.
- Migration is local, only among neighbours.

Hierarchical/Hybrid

- Multiple populations, each works parallel
- Combine the benefits of multiple methods.

Synchronous Master Slave

- Synchronous
  - Wait for evaluations
  - Wait for the last slave to finish
  - Then start the next cycle
- Complete equivalent of single processor.
- CPU time wasted for synchronizing with the slowest evaluation.

Asynchronous Master Slave

- Asynchronous
  - Do not wait for all evaluations to complete.
  - As evaluations are completed they are reflected into the current generation
- Potentially faster, no CPU wasted.
- How many individuals generated. How are they inserted into population?
- EA with small generational gap.
- Steady state genetic algorithms.

Master Slave Execution Time

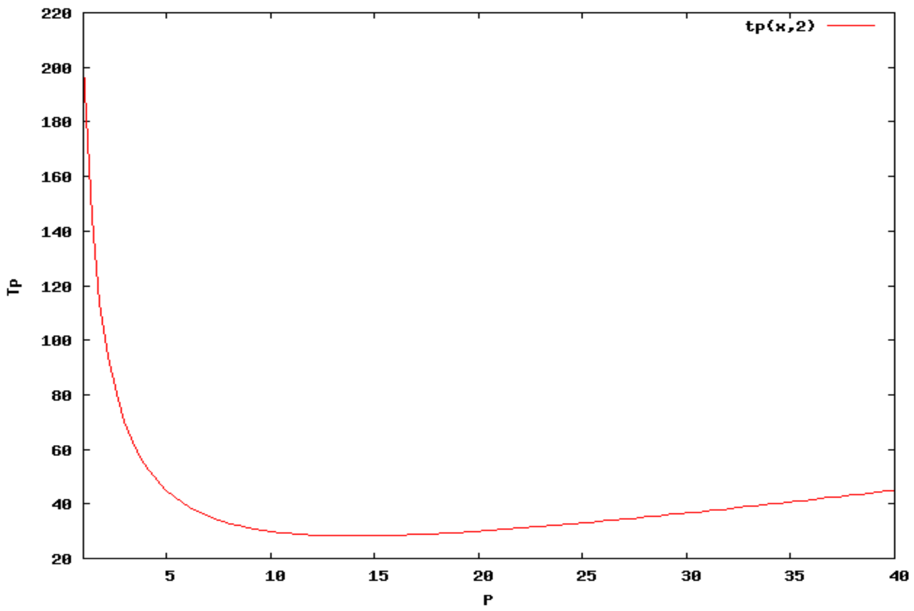
- Computation time: $\frac{\text{pop. size} \times \text{eval time}}{\text{num. of proc.s}} = \frac{nT_f}{P}$
- Communication time:

  $\text{num. of proc.s} \times \text{comm. time} = PT_c$

- Computation time: $\frac{nT_f}{P} + PT_c$

  $\text{Minimize: } P^* = \sqrt{\frac{nT_f}{T_c}}$

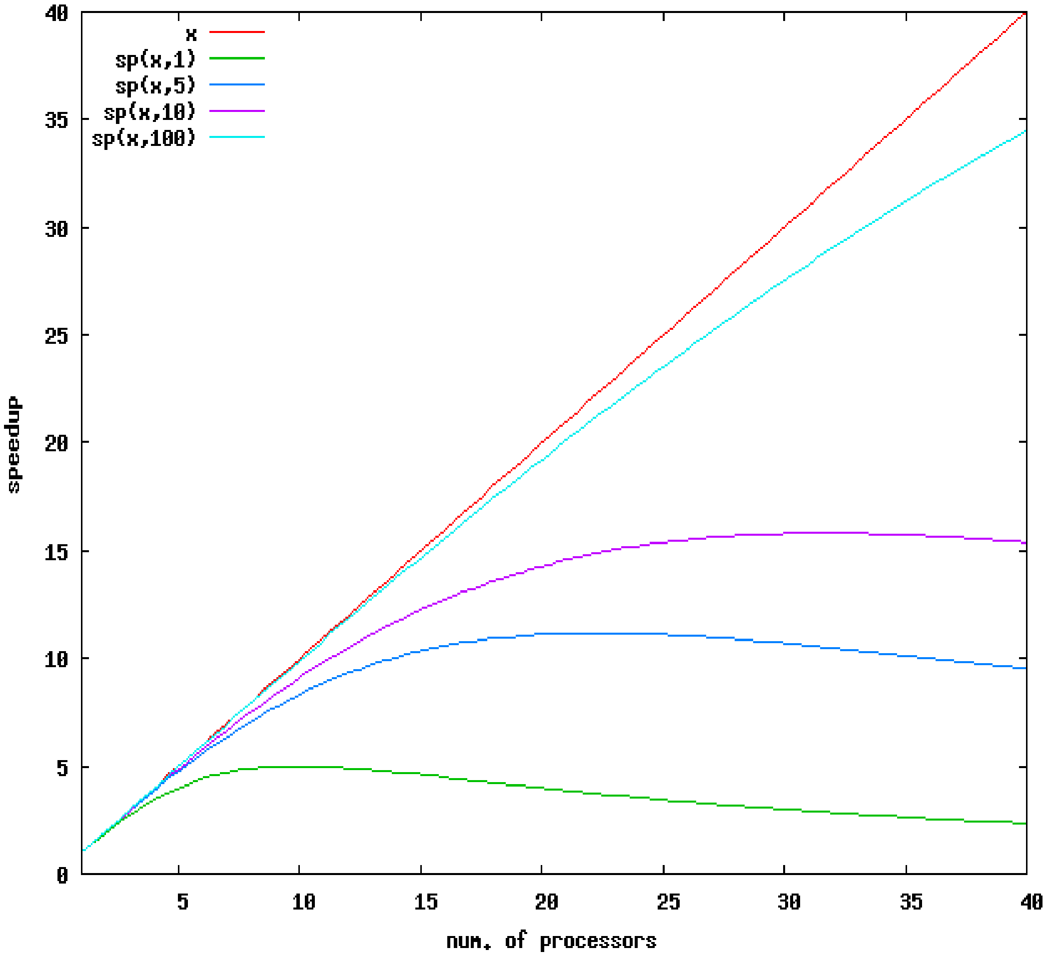
- Computation time should be larger than the communication cost in order to get efficiency.



Speedups

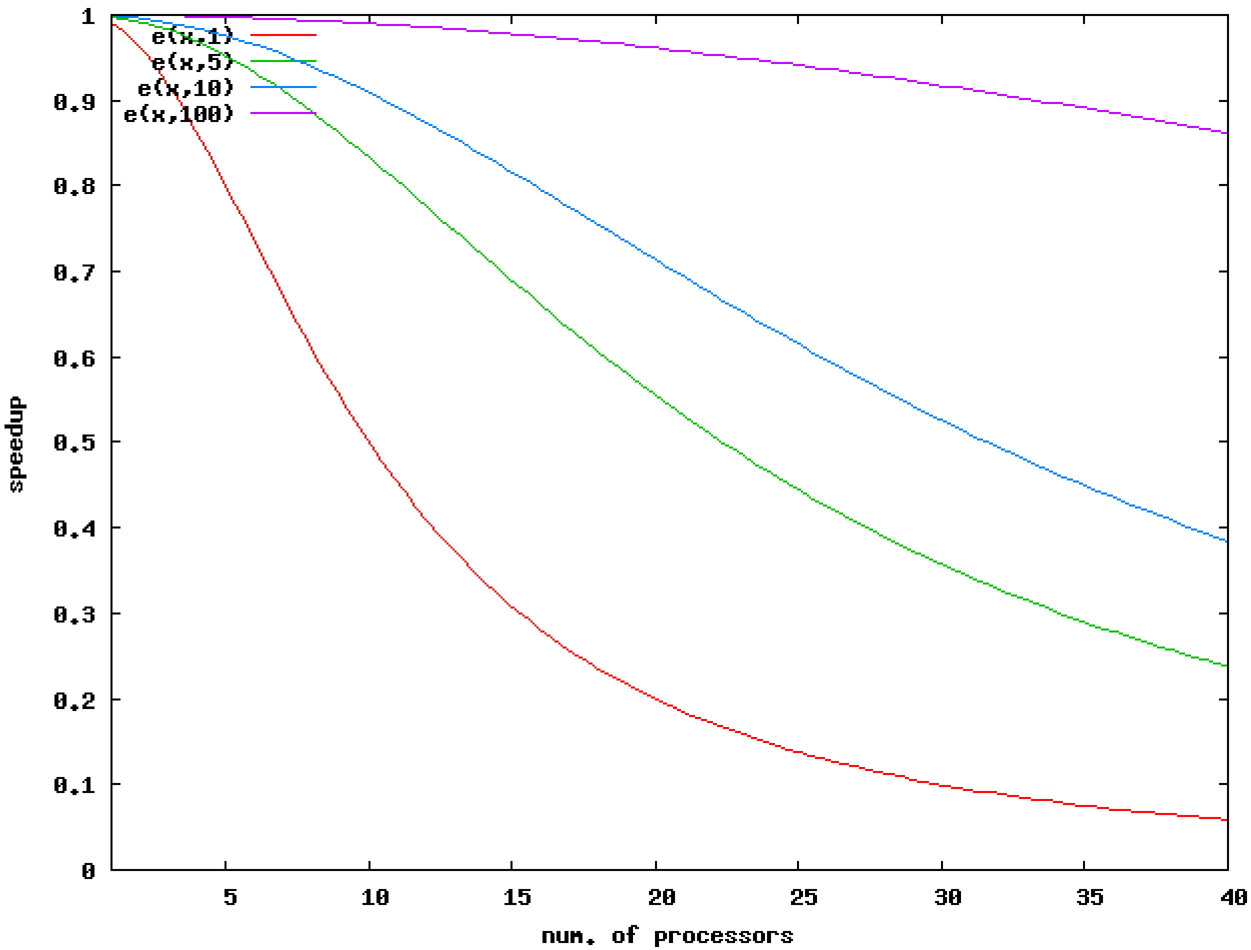
- $S_p = \frac{T_s}{T_p} = \frac{n T_f}{\frac{n T_f}{P} + P T_c}$
- Speedups for different computation/comm. Ratio.
- Near linear speedups until:

$$P \approx \left(\frac{n T_f}{T_c} \right)^{\frac{1}{3}}$$



Efficiency

- $E = \frac{S_p}{P}$



Parallelization of Genetic Operators

- Genetic operators are very fast linear procedures.
- Does it worth parallelizing?
- Selection: global process, modification required.
- Usually communication cost is larger than the computation.

Parallel EA Design

- Design choices:
  - Single/Multiple populations
  - Size of the populations
  - Number of populations
  - Isolated or connected populations
  - Connection policy
  - Migration rate
  - Migration policy
- Many parameters to compare .
- Different varieties of dynamically changing parameters.

Isolated Demes

- Multiple simple GAs
- Execute in different processors, than collect the best results.
- Quality of solution has binomial distribution. Taking the best of a binomially distributed solutions. $\sqrt{2 \ln P}$
- Speedup is the reduction in number of generations required to get to the same solution quality in a single processors. It is proportional to

Fully Connected Demes

- All demes can migrate to each other
- Migration frequency:
  - At each generation (gets similar to master slave)
  - At the end of epoch.
- Algorithm:
  - Run isolated until some convergence
  - Swap a ratio of current population with every other deme.
  - Restart the demes

Migration Policies

- Choose emigrants/replacements:
  - Randomly
  - Based on fitness
- Fitness based migration increases selection pressure.
- Converge faster and can make super inear speedups.
- Sometimes too fast: premature convergence?

- Migration frequency:
  - Every generation vs with some convergence period.
- Best migrate, replace worst
- Random migrate replace worst.

Superlinear Speedups: Why?

- Reduction of total work, the search space?
- Increase selection pressure while keeping diversity?
- Needs more investigation.

Fine-Grained Parallel EA's

- Different names:
  - Cellular GAs
  - Diffusion model GAs
- Spatially structured population.
- Local selection
- Local mating (migration)
- Implementations in Massively Parallel Computers (SIMD) and Distributed Parallel Machines (MIMD).

- ASPARAGOS
  - Ladder population structure, ends are connected to each other (Torus)
  - Local hill climbing
  - Combinatorial optimization problems.
- Parameters:
  - Population size
  - Structure (topology)
  - Mating strategy
  - Neighborhood size

Mating Strategies

- Choose mates:
  - Proportional selection
  - Tournament selection
  - Random walk
- Choose replacements
  - Always replace local
  - Replace local if worse.
- Neighborhood Sizes and Shapes
  - Selection pressure.
  - Neighborhood radius/radius of whole grid.