PSO and GA

Commonalities

- Population based stochastic optimization
- Both algorithms start with a group of a randomly generated population
- Fitness values to evaluate the population
- Both update the population and search for the optimum with random techniques
- Both systems do not guarantee success

Differences

- No genetic operators like crossover and mutation. Particles update themselves with the internal velocity
- Particles have memory, which is important to the algorithm
- The information sharing mechanism in PSO is significantly different
    - Info from best to others, GA population moves together
- There is no selection in PSO
    - All particles survive for the length of the run. None die.
    - PSO is the only GA that does not remove candidate population members

Differences continued

- PSO has a memory
  - not just “what” that best solution was, but “where” that best solution was also
- **Quality**: population responds to quality factors pbest and gbest
- **Diverse response**: responses allocated between pbest and gbest
- **Stability**: population changes state only when gbest changes

Advantages

- The fitness function can be non-differentiable (only values of the fitness function are used).
- The method can be applied to optimization problems of large dimensions, often producing quality solutions more rapidly than alternative methods.

Disadvantages

- There is no general convergence theory applicable to practical, multidimensional problems.
- For satisfactory results, tuning of input parameters and experimenting with various versions of the PSO method is sometimes necessary.
- Stochastic variability of the PSO results is very high for some problems and some values of the parameters - C1, C2 and W.

Global Swarm Optimization

- Modification of PSO to reach optimal solution quicker
- In PSO : G_{best} having a better result is used for updation and rest are terminated
- GSO introduces “Experience” concept
- E_{best} : G_{best} value at any iteration will be randomly selected and is used to update the velocity value

```
Initialization
Do
    For each particle
        Calculate fitness value
        If fitness value > pBest :
                pBest = fitness value
        If pBest > gBest :
                gBest = pBest                                   {Save all gBest values}
    End
    eBest = random gBest
    For each particle
        Calculate particle velocity using pBest, gBest and eBest
        Update particle position
    End
While maximum iterations or minimum error criteria is not attained
```

Velocity update in GSO

$v_i(t + 1) = wv_i(t) + c_1r_1[pBest_i(t) - x_i(t)] + c_2r_2[gBest(t) - x_i(t)] + c_3r_3[eBest(t) - x_i(t)]$

- $c_3r_3[eBest(t) - x_i(t)]$ - Improvement Factor
- E_{best} requires no comparison
- The particles will learn and know about different “Experience” for each process
- Improvement Factor (IF) will help the algorithm have the suitable velocity value for updating the next particles’ positions

References

Particle Swarm Optimization :

1. https://en.wikipedia.org/wiki/Particle_swarm_optimization
2. http://www.cs.armstrong.edu/saad/csci8100/pso_tutorial.pdf
3. http://www.mnemstudio.org/particle-swarm-introduction.htm
4. http://www.swarmintelligence.org/tutorials.php
5. http://www.iam.fmph.uniba.sk/ospm/Harman/PSO.pdf

Global Swarm Optimization :

1. http://www.hindawi.com/journals/jam/2014/329193/