Effects of Parameters on a One-Max Genetic Algorithm

Contents

- Changes to One Max to turn it into a function we can use
- Parameters Settings
- Run Experiments
- Results
- Conclusion

Changes to One Max to turn it into a function we can use

function [bestFitness, medianFitness] = oneMaxFunc(p)

popSize = p.popSize;

nGenes = p.nGenes ;

rateMut = p.rateMut;

rateXov = p.rateXov;

maxGens = p.maxGens;

Parameters Settings

```
p(1).popSize = 18;
p(1).nGenes  = 100;
p(1).rateMut = 1/p(1).nGenes;
p(1).rateXov = 0.9;
p(1).maxGens = 150;
name{1} = 'BaseLine';

p(2) = p(1);
p(2).rateMut = p(1).rateMut * 3;
name{2} = 'Triple Mutation';

p(3) = p(1);
p(3).popSize = p(1).popSize / 3;
p(3).maxGens = p(1).maxGens * 3;
name{3} = 'Triple Gens';

p(4) = p(1);
p(4).popSize = p(1).popSize * 3;
p(4).maxGens = p(1).maxGens / 3;
name{4} = 'Triple Pop';

p(5) = p(1);
p(5).rateMut = p(1).rateMut / 3;
name{5} = 'One Third Mutation';
```

Run Experiments

As we are testing a stochastic algorithm, we run each parameter set 100 times and take the median result, by changing a 'for' into a 'parfor' it tells MATLAB to run each iteration on a different core.

```
tic; %Start a timer
for pset=1:5
    clear bestFitness medianFitness
    parfor run=1:100
        [bestFitness(run,:), medianFitness(run,:)] = oneMaxFunc(p(pset));
    end
    p(pset).fitMax = median(bestFitness,1);
    p(pset).fitMed = median(medianFitness,1);
end
toc; %End a timer
```

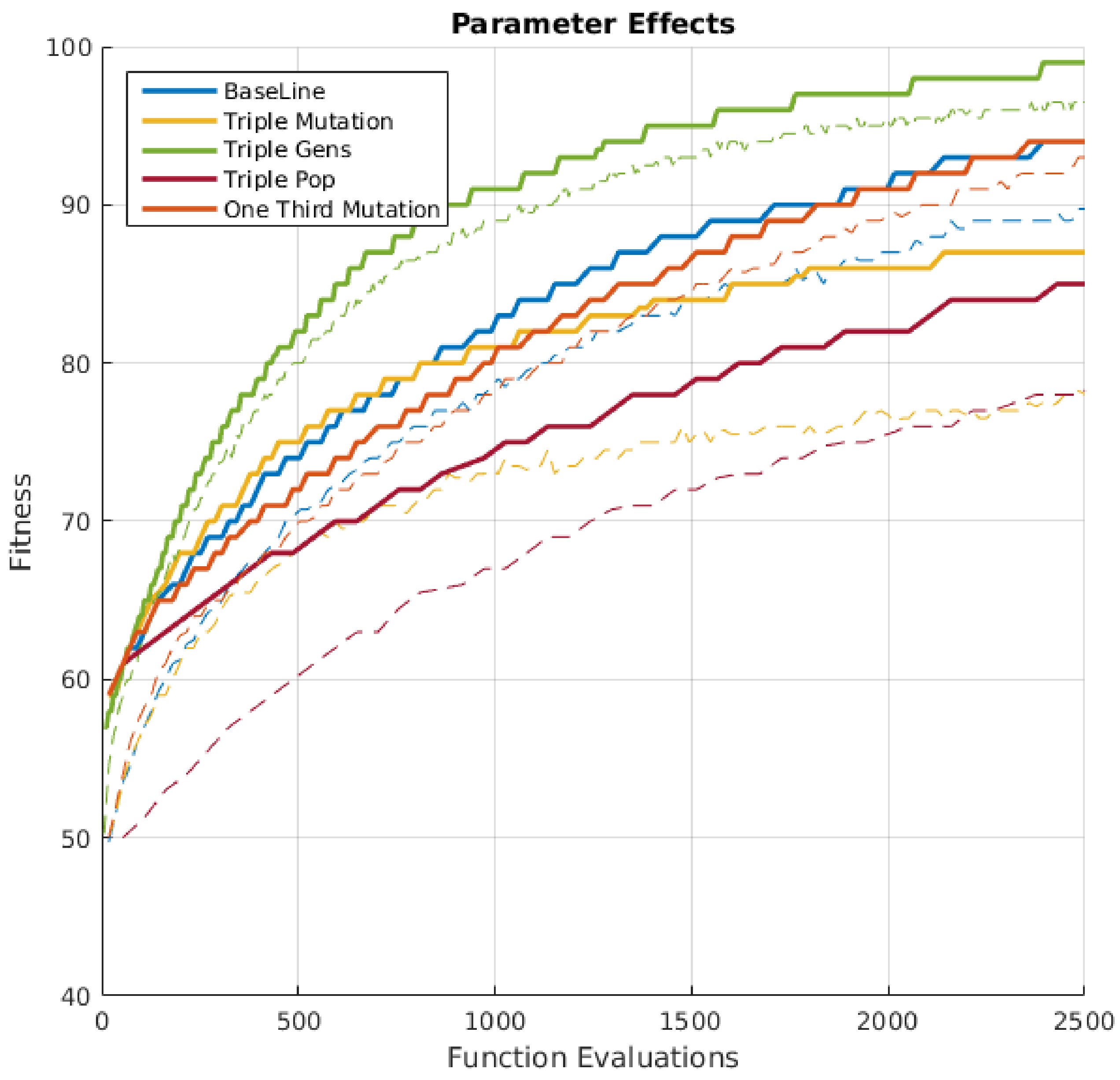
```
Elapsed time is 3.030387 seconds.
```

Results

Since we are using different values for the generations and populations, ensure that we line up the different runs on the measure that matters: fitness evaluations

```
figure(1); clf; hold on;
for pset=1:5
    fitnessEvals = [p(pset).popSize: p(pset).popSize : p(pset).popSize*p(pset).maxGens];
    lineHandles(pset) = plot(fitnessEvals, p(pset).fitMax,'-','LineWidth',2);
    plot(fitnessEvals, p(pset).fitMed,'--','Color',get(lineHandles(pset),'Color'))
end

legend(lineHandles,name,'Location','NorthWest')
xlabel('Function Evaluations');ylabel('Fitness');grid on;
set(gca,'XLim',[0 2500]); title('Parameter Effects')
```



Conclusion

It can be seen from the graph here that in the One-Max problem that spending function evaluations on more generations of a smaller population results in much faster convergance speed and better resulting fitness. We also see while a 3% mutation rate, while initially outperforming the baseline 1% and reduced 0.3% mutation rate, was too high to allow convergence on higher fitness solutions.

Published with MATLAB® R2015b