Previously accomplished:

Graphics set up, greedy solution generator, cross breed, best solution distance: 19608.

Monday, February 15, 2016:

1. Increase Population size from 43 to approximately 976. 43 solutions are greedy solutions and the rest is randomly generated.
2. File i/o random indexes into R.py so that the population remains the same each time we run main.

Since the breeding population is sorted from the shortest distance to the longest distance, the more optimal solution should have a better chance at being selected for breeding. In this study, linear and exponential function will be explored as models for the probability.

Let linear function denote the probability of solution with index x in the population of size getting picked for cross breeding, since the sum of the probabilities for the each individual solution in the entire breeding population needs to equal 1, constants and can be found given the following integral,

The probability of the last solution of index n being picked should approach 0, given that an expression of in terms of and can be established as

Given , and and be found with the system of equations shown.

Let exponential function denote the probability function. Then,

Similar to the linear model, an expression of in terms of and and be found as

Again and and be found given .

For example, a breeding population of size 781 would give and 739 with 6 significant figures for and .

As seen from the graph above, the optimal solution has a chance ten times higher in the exponential model than the linear model, which allows the greedy solutions that are locally optimized to be picked far more often than the random solutions. However, the disadvantage is the lack of randomness as the solutions towards to end have probabilities approaching 0.

It is worth noting that although the solving the integral approximates the model quite well, realistically the model needs to be fine tuned since is infinitely small. In reality, we want the Riemann sum to be 1 instead of the definite integral.

Tuesday, February 15

1. Finished probability distribution, translated into python and attempted to breed.

Starting:

best: 19608 worst: 96341 average: 78160 range: 76733 size: 976

Stats for exponential probability distribution model cross breeding, no mutation, 100 iteration:

best: 19097 worst: 27949 average: 21247 range: 8852 size: 899

best: 19390 worst: 28021 average: 21330 range: 8631 size: 896

best: 19323 worst: 27565 average: 21230 range: 8242 size: 936

Stats for exponential probability distribution model cross breeding, no mutation, 10 iteration:

best: 19571 worst: 29012 average: 22665 range: 9441 size: 930

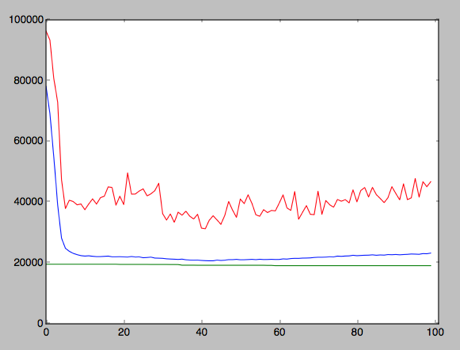
best: 19548 worst: 29000 average: 22351 range: 9452 size: 922

best: 19571 worst: 30492 average: 22408 range: 10921 size: 932

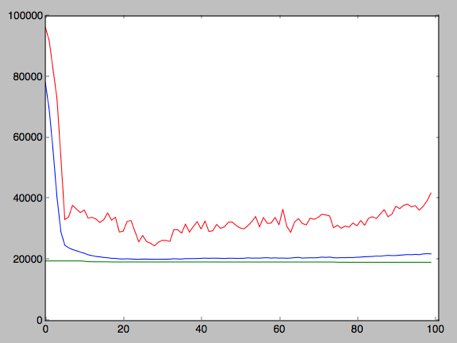
exponential probability distribution model, cross breeding, 100 iteration:

Mutations:

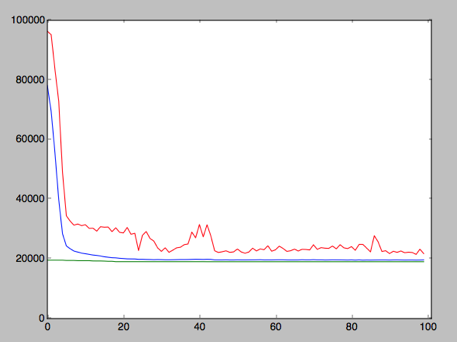
1. 1 swap mutation in solution:
   1. 70% of parent1
      1. best: 19125 worst: 46482 average: 23151 range: 27357 size: 1178



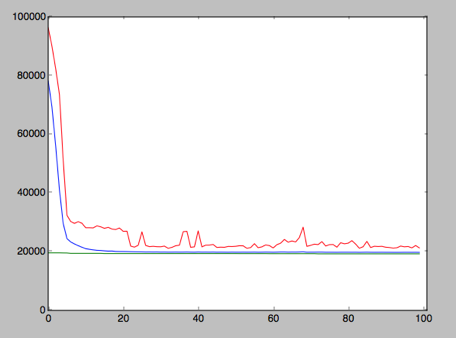
* 1. 50% of parent1
     1. best: 19160 worst: 40477 average: 21981 range: 21317 size: 1164



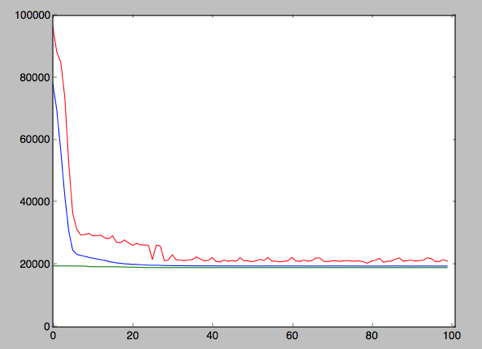
* 1. 30% of parent 1
     1. best: 19071 worst: 22588 average: 19648 range: 3517 size: 869



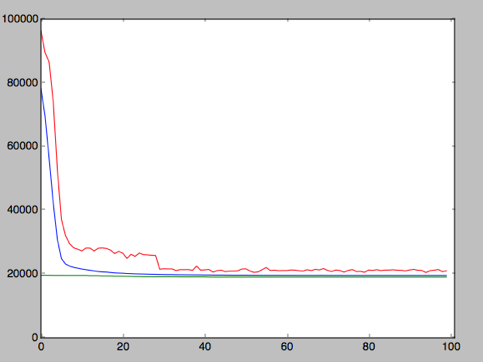
* 1. 15% of parent1
     1. best: 19281 worst: 21533 average: 19822 range: 2252 size: 825



1. 1 swap mutation of child:
   1. 30%
      1. best: 19071 worst: 21265 average: 19588 range: 2194 size: 820



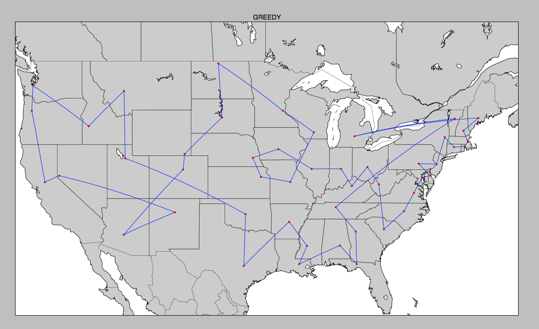
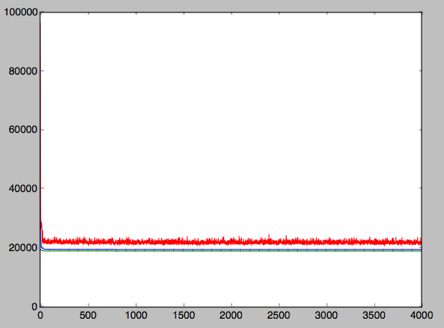
* 1. 15%
     1. best: 19071 worst: 20957 average: 19567 range: 1886 size: 790



Wednesday February 17

4000 iteration 30% 1 swap child mutation exponential probability half cut cross breed

best: 19031 worst: 22087 average: 19552 range: 3056 size: 852



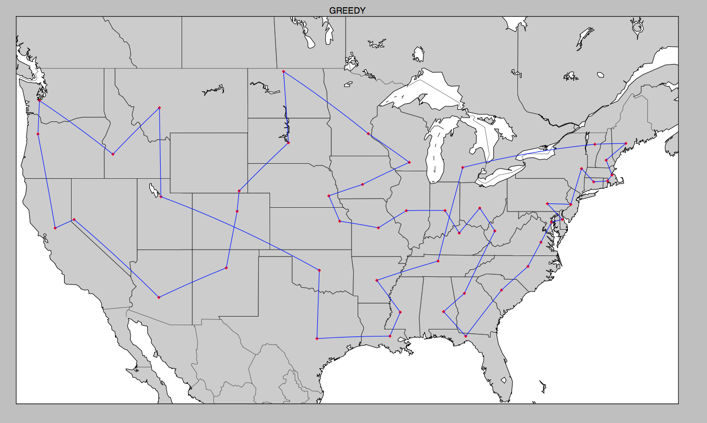
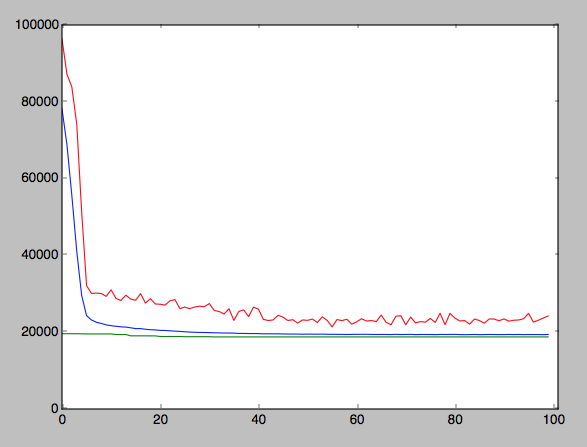
Thursday, February 18

Developed new mutation called reverse. Randomly chooses segments of code of length between 2-5 and reverse the order in attempt to untie knots.

100 iteration, exp prob, rev2-5 mut on child 30%, half cut cross breed

best: 18772 worst: 23654 average: 19391 range: 4882 size: 840

it has only been 100 iterations and the best solution has already improved by nearly 1000, and the average is already lower than the best solution in original population. Graph looks similar to the previous mutation



The graph is two knots away from passing the stupidity test!!!

Friday February 19

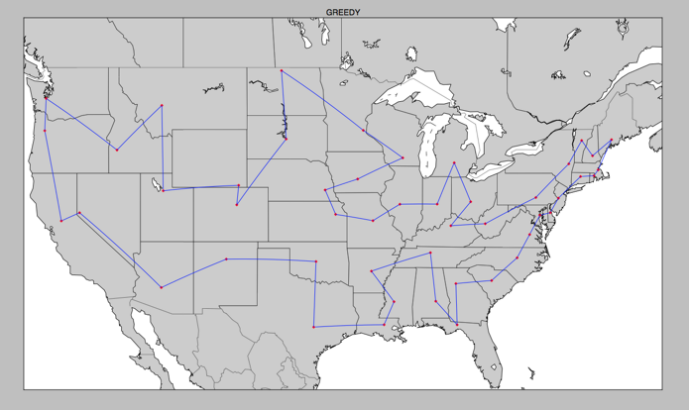
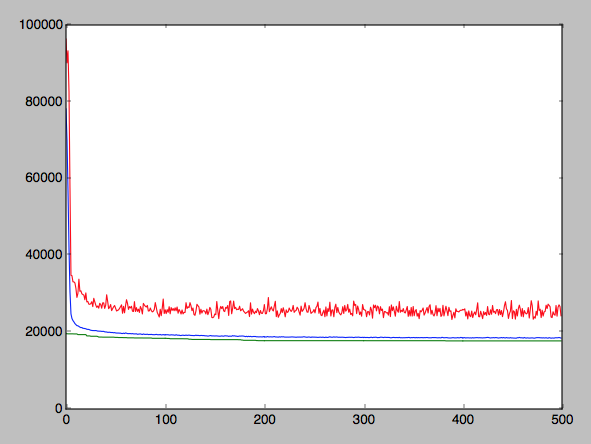
100 iteration exp prob, rvrs2-10 mut on child 30%, half cut cross breed.

best: 18501 worst: 25732 average: 19291 range: 7231 size: 875

500 iteration exp prob rvrs2-19 mut on child 30%, half cut cross breed.

best: 17737 worst: 25360 average: 18481 range: 7623 size: 886

OPTIMAL SOLUTION PASSED THE STUPIDITY TEST.



500 iteration exp prob rvrs2-25 mut on child 30%, half cut cross breed.

best: 17591 worst: 27628 average: 18633 range: 10037 size: 924

We see that as the possible segment for reversal increase, the best solution improves but the range increases and the worst solution stays high, though the average is still low.

Wednesday February 24

Worked on weave() function using recursion, lots of issues.

Thursday February 25

Day 8 met with Mr.G.

Dicsussed future goals: potential blinking nodes, simulated annealing

Simulated annealing: start from very random solution, and gradually decrease the randomness through picking chunks of solution, randomize the chunks and see if it improves.

New direction for weave(), mark the cities already in the child as false and ignore any false cities instead of removing them as cross breed happen. This result in less runtime as removing elements costs a lot of time.

New mutation, pick a random chunk and move it to a random place.

Settled on exp probability, no need to change n.

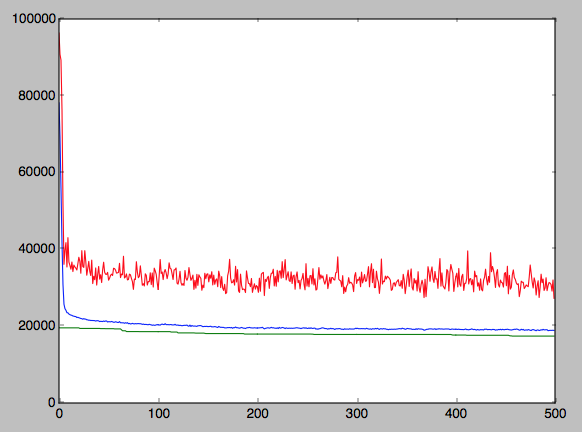
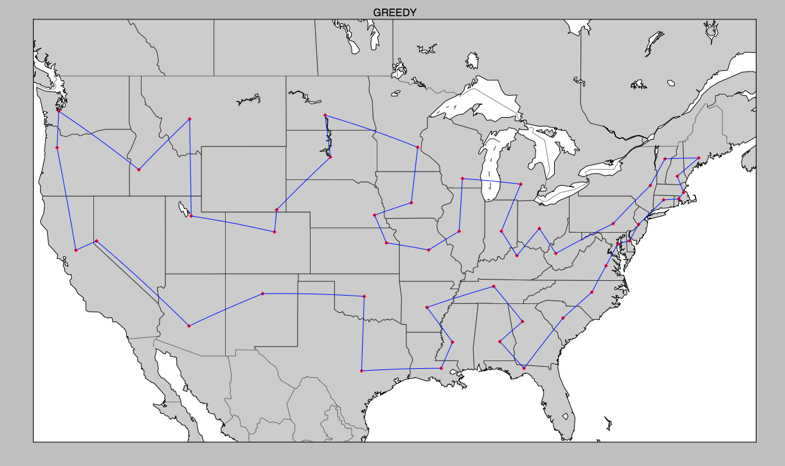
New population idea, different types of greedy algorithm with not only the shortest city but the second shortest, third shortest and so on.

Fix Bismarck coordinates.

Wednesday March 9, 2016

Bismarck coordinates fixed, weave function functional. The resulting best is about 17423, which is not far than our goal of 17088.

best: 17423 worst: 31705 average: 18495 range: 14282 size: 948



100 iterations, no greedy, randomization, mutate reverse 2-25, exp, weave 50% size 6, half-cut 60%

best: 18215 worst: 37113 average: 21136 range: 18898 size: 1069

greedy 1-10

best: 18131 worst: 34143 average: 20427 range: 16012 size: 1049

greedy 1-10 with weave size 8

best: 18125 worst: 36378 average: 20540 range: 18253 size: 1065

greedy 1-10 no weave all half cut mutate reverse 2-25, 100 iteration

best: 17902 worst: 27249 average: 19170 range: 9347 size: 920

greedy 1-10 no weave all half cut mutate reverse 2-25 500 iterations

best: 17576 worst: 25909 average: 18601 range: 8333 size: 924

April 9

Testing effect of different weave size on population

100% weave with size 2,4,6,8,12,24, 100 iteration, greedy 10, mut rev 2-25, exp

best: 19608 worst: 58154 average: 23636 range: 38546 size: 1096

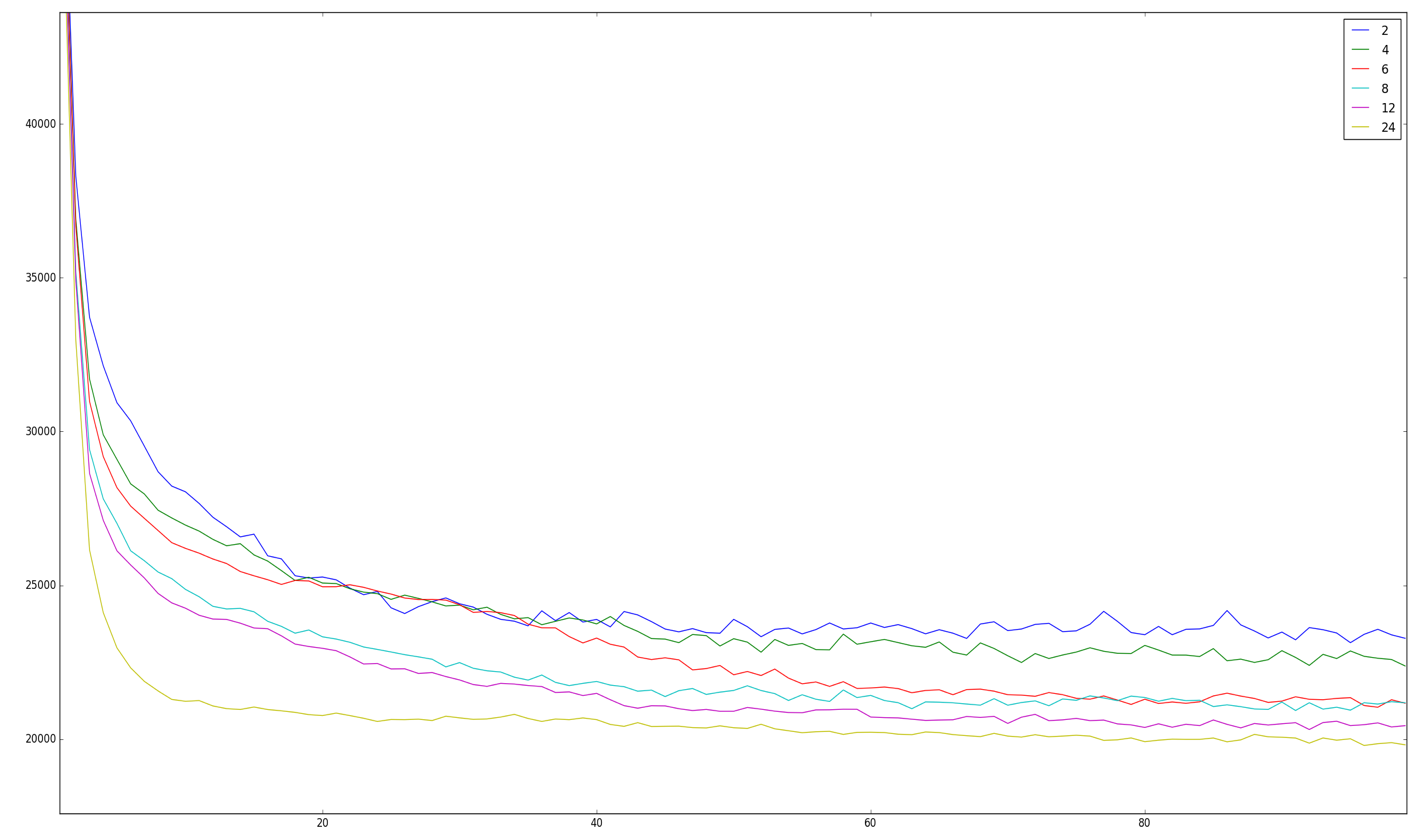
best: 19488 worst: 42522 average: 22441 range: 23034 size: 1013

best: 18780 worst: 37092 average: 21127 range: 18312 size: 1056

best: 18681 worst: 35229 average: 21187 range: 16548 size: 1081

best: 18540 worst: 32838 average: 20360 range: 14298 size: 989

best: 18713 worst: 27631 average: 19825 range: 8918 size: 926



We can see that with 100% weave, the optimal weave size is 24 for sure.

50% half\_cut, 50% weave with size 2,4,6,8,12,24, 100 iteration, greedy 10, mut rev 2-25, exp

the stats from weave size 2 to weave size 24 in order:

best: 18647 worst: 59498 average: 22113 range: 40851 size: 1107

best: 18890 worst: 36142 average: 21504 range: 17252 size: 1063

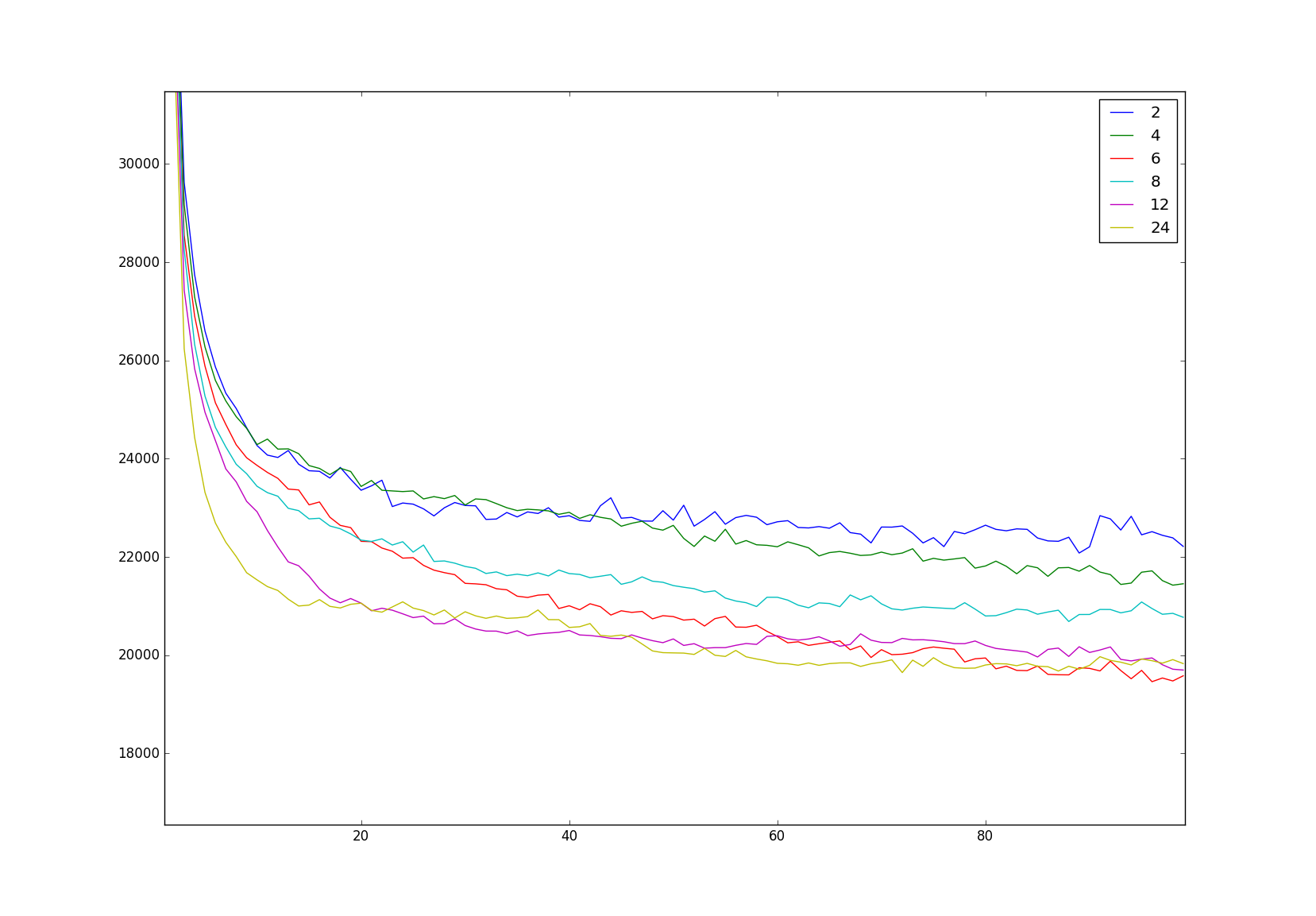
best: 17747 worst: 37290 average: 19589 range: 19543 size: 1035

best: 18518 worst: 35352 average: 20802 range: 16834 size: 1023

best: 18180 worst: 29586 average: 19757 range: 11406 size: 1028

best: 18476 worst: 29967 average: 19873 range: 11491 size: 1013

comparison graph:



Interestingly weave size 6 started out worse than others, however by iteration 100, it has surpassed efficiency of both 12 and 24, ending up with the best average. Weave size of 12 and 24 seems to be tangled and not determined which one is better. Weave size 2 and 4 and 8 is obviously worse than the others.