# Iterations

Currently we have max iters parameter. Tells us how many times we are “allowed” to call the cost function, i.e how many times the cost of a tour is calculated. So how many times we can check how good the solution is. The lower the better.

So more birds we have, the faster those iterations will be depleted, as one iteration does not mean every birds moves once. So if we have more birds, then the more often the cost function will likely be called in one round. Round meaning in this case every bird can perform one move. Even though the cost is only calculated after walk and fly, and those are to some degree chosen randomly, on average we will use up more iterations after each round. At the end this means the more birds we have the more potential solution we will be able to find and evaluate, but we won’t be able to explore each solution as deeply, because after all, each bird (and therefore solution) can perform less moves (and therefore less investigation of a solution). This is especially critical for those birds that are moving into the direction of the global minimum, as they will have less opportunities to move further into that direction. **Unfortunately, as there is to mechanism that encourages well performing birds to increase the probability of walking (to further move into the right “direction”), a well performing bird won’t perform more iterations that a bad one on average.**

Consequently, using more birds can be seen as nudging the algorithm into a breadth search, while fewer birds will yield a depth search. Because the birds are configured to search for a (local) minimum, for TSP problems that are big enough fewer birds will generally yield better results, which we see as counterintuitive.

A graph of a number of birds

Description automatically generated A graph with numbers and a line

Description automatically generated

(Show 3 other examples, where this behavior can be seen, so the left plot for one ca. 100 city and 1000+ city problem. The plot right once should be enough)

One could argue that it would be wise to redefine one iteration as “each bird moves once”. While this would yield a more deterministic assessment of runtime for a specific configuration (namely O(n\_birds\*n\_iters) ), a bird number in the hundreds, which is not unlikely, paired with iterations in the millions, also not unlikely for bigger problems, would yield rather long computation times.

This sounds promising, as it would combine depth and breadth search, but it will be close to infeasible for big enough problems.

(Find out what is better, depth or breadth search)

* How long does the combination actually take? And will the result be better?
* For 1000 birds and 1 mill max iterations, will be max one billion iters!

Find ways to better bird behavior, so that iterations will be used more efficiently, instead of giving the algorithm more max iters?!

* Birds need to use their iters better?
* Because joining does not use an iteration, we need to make it better?
* Improve swarm bahavior?