Hello,

My solution to explore all of the design domain consists mostly of the following four ideas:

1) Random sampling: for each result, I randomly sample combinations until I find one that satisfies all constraints. This method is flexible enough to work for any constraints despite not being that efficient.

2) Crowding distance: as results are randomly generated, another constraint is applied so that all results are at least a specific distance apart from each other. This way I am using the already known information to explore the design domain and also avoid physically equivalent results. Distances are non-dimensionalized based on the current known domain, not the [0,1] bounds.

3) Inflation/deflate: I initially increase (inflate) the number of desired results to increase diversity. Less diverse solutions are periodically eliminated based on the crowding distance, which is more computationally efficient than eliminating at the end. This second step also guarantees that the distribution is more uniform in the explored areas (look at plot generated)

4) Mutation: as an alternative to random sampling, another selecting algorithm is used. Mutation consists of slightly changing a known feasible solution. Inspired by genetic optimization, it helps in problems where most of the design domain is unfeasible. Dominant selection algorithm when there is an implicit ingredient or an equality constraint.

There are two parameters, crowding_distance, inflation, mutation_period (determines how often a mutation is done relative to random sampling) that could be tweaked to get more uniformly distributed results. For alloy.txt, I was not able to find all of the feasible domain.

The attached code is also available on Github.

Thanks,

Pedro Leal

Graduate Research Assistant at Texas A&M University