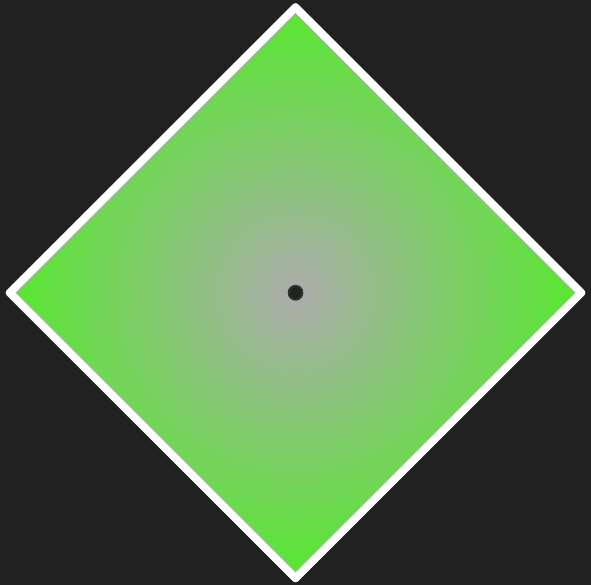
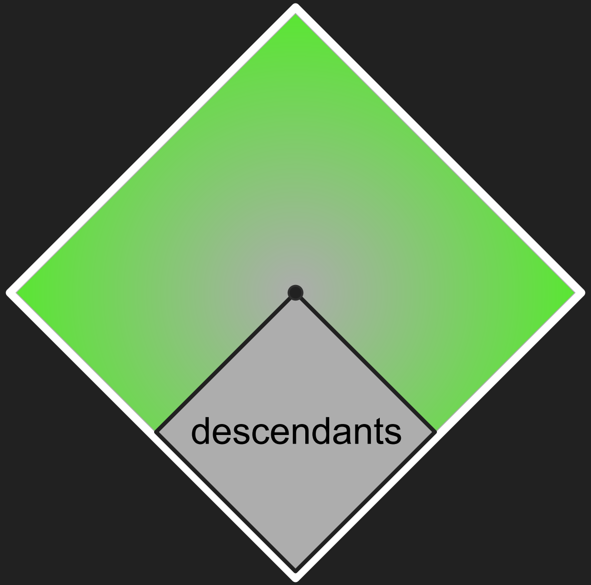
Naïve search

Whether we need a more specific, generic, or a sibling function, we always search the entire function space (works for around 6 variables, but not for 8).



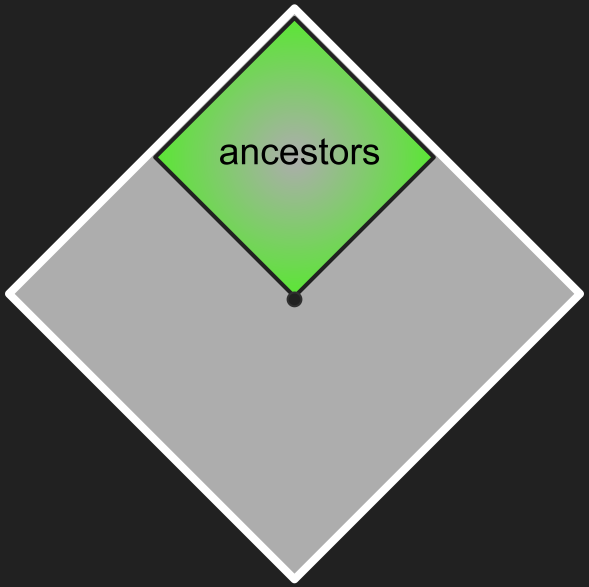
Not-so-naïve search

If we need a more generic function we will search everywhere except where we already know we won't find it.



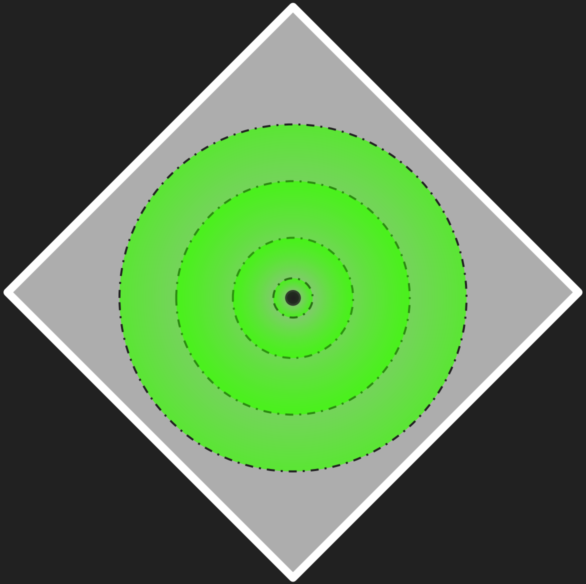
Near-optimal search (proposed by Filipe)

If we need a more generic function we will search only in that function’s ancestors (smallest function space, but we risk missing out on some optimum candidates, or even fail to find a consistent candidate even though one may exist).



Iterative optimal search

Whether we need a more specific, generic, or a sibling function, we always search for the closest possible function to the original function. We start by looking at the first closest functions, then second closest, so and and so forth until we find a suitable candidate.



Iterative optimal search

Pros

- smaller search space than naïve search (possibly even smaller than not-so-naïve)
- is guaranteed to find an optimal function

Cons

- not all candidates will be found
- search space might still be too large
- we will be searching for some functions that won’t matter (doesn’t take into account whether we’re looking for ancestors, descendants or siblings)