```
# Create the population
n_to_optimize = len(atom_numbers)
comp = InteratomicDistanceComparator(
    n_top=n_to_optimize)
population = Population(
    data_connection=data,
    population_size=20, comparator=comp)

pair = CutAndSplicePairing(slab,
    n_to_optimize, blmin).get_new_individual
mutate = PermutationMutation(
    n_to_optimize).get_new_individual

# Test 50 new candidates
for i in range(50):
    # Parents a and b:
    a, b = population.get_two_candidates()
    # Mutate (30 %) or pair parents a
    # and b to obtain child candidate:
    if random() < 0.3:
        child, desc = mutate([a])
    else:
        child, desc = pair([a, b])
    if child is None:
        continue
    data.add_unrelaxed_candidate(child,
        description=desc)

    relax_and_add(child, data)
    population.update()

write('all_candidates.traj',
      data.get_all_relaxed_candidates())
```