As there are many ways to combine each configuration, we select a few configurations by examining them trough simulations.

The set of model-configuration contains 216 elements. We choose to run each model-configuration for 50 evolution cycles. For all model-configurations, we use the same 4 factual process instances, which are randomly sampled from the test set. We ensure, that the outcomes of these factuals are evenly divided. We decide to limit the population size to a maximum of 1000 counterfactuals. Within each evolutionary cycle, we generate 100 new offsprings. We keep the mutation rate at 0.01 for each mutation type. Hence, across all cases that are mutated, the algorithm deletes, inserts, and changes 1% of events per cycle. We collect the mean viability and its components across the iterative cycles of the model.

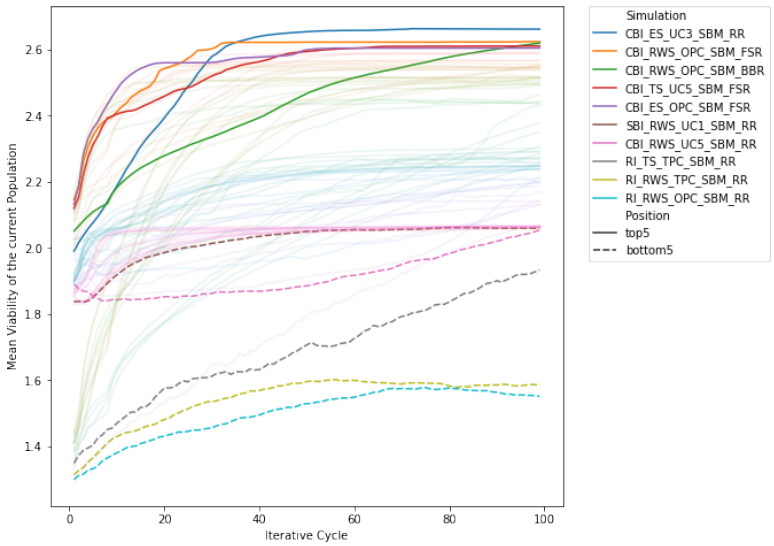

Figure 1: This figure shows the average viability of the 5 best and worst model-configurations. The x-axis shows how the viability evolves for each evolutionary cycle.

Figure 1 shows the bottom and top-5 model-configurations based on the viability after the final iterative cycle. We also show how the viability evolves for each iteration. The results reveal a couple of patterns. First, all of the top-5 algorithms use either **CaseBasedInitiator or SampleBasedInitiator** as initiation operation. In contrast, the bottom-5 all use **RandomInitiator** as initialisation. Hence, the initiaiation appears to be majorly important for the algorithm. Second, we see that most of the top-5 algorithms use the **ElitismSelector**. The complete table of results is in **??**.

In Figure 2, we see the effects of each operator type, except the mutation operation.

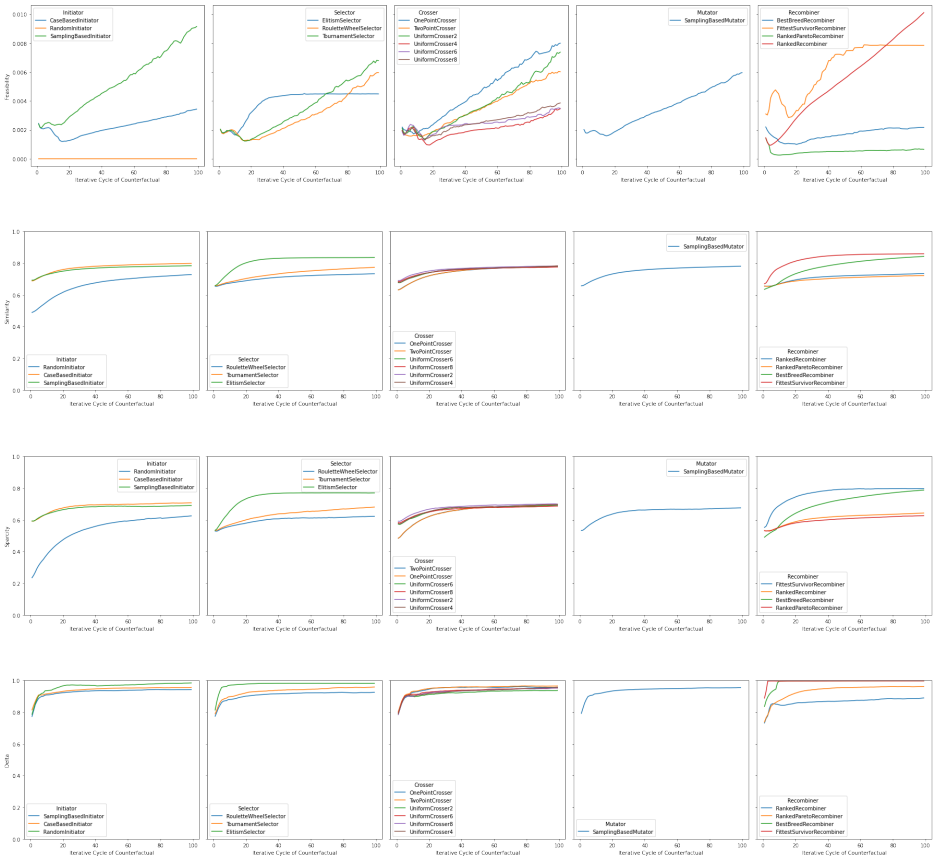

Figure 2: The evolution of each viability measure over the entire span of iterative cycles. Each figure adjust the respective operator type by taking the average over all other operator types.

Starting with some commonalities across operator-type and measure, the figure shows that the initiator heavily determines the starting point for each measure. For instance, the **RandomInitiator** starts well below the other initiator operations when it comes to sparcity and similarity. Another noteworthy general observation is the delta measure. Here, for each operator type we see a movement towards the highest possible delta value. Hence, most configurations are capable of changing the source class to the desired class.

In terms of viability Figure **??**, shows an increase only for cases, in which the **SampleBasedInitiator** is used. Similar holds for recombination with **FittestSurvivorRecombiner**.

The results for the selection operator type are undeniably in favor of **ElitismSelector** for all viability measures. The same holds for the recombination operation. Here, the **FittestSurvivorRecombiner** yields better results.

When it comes to the crossing operation, the results indicate, the differences between **OnePointCrosser** and **TwoPointCrosser** are inconclusive for all viability measures except feasibility. One can explain that by noting, that both operations are very similar in nature. However, cutting the sequence only once produces less impossible sequences for the child sequences.