Population Based Search Tools

Ben Simondet and Isaac Koak

3/3/2016

Introduction

In this set of experiments, we compared the performance of random search, uniform crossover, uniform crossover with a tweak, two point crossover, and two point crossover with a tweak.

Our uniform and two point crossovers take two parents and produce a child. The two parents are chosen out of a pool of 50 parents. Each iteration generates a new set of parents. The tweak addition randomly switches one of the choices picked.

Our normal crossover is done by interleaving the two parents and using partition to randomly pick one of each two elements. The two point crossover picks two points over the length of the parent choices, takes from the first parent up to the first point, takes from the second parent from the first point to the second point, and takes from the first parent from the second point to the end of its length.

We chose to use the penalized scoring method, as this was proven to be a very effective scoring method in the last experiment.

Experimental Setup

We tested the above strategies with the following parameters:

- Random search
- Uniform Crossover
- Uniform Crossover with Tweak
- Two Point Crossover
- Two Point Crossover with Tweak

We performed 25 independent runs of 10,000 iterations on each of the following problems for each of the treatment methods:

- `knapPI_11_20_1000_4`
- `knapPI_13_20_1000_4`
- `knapPI_16_20_1000_4`
- `knapPI_11_200_1000_4`
- `knapPI_13_200_1000_4`
- `knapPI_16_200_1000_4`
- `knapPI_16_1000_1000_4`

Note that problems in this report are represented without the leading `knapPI_`.

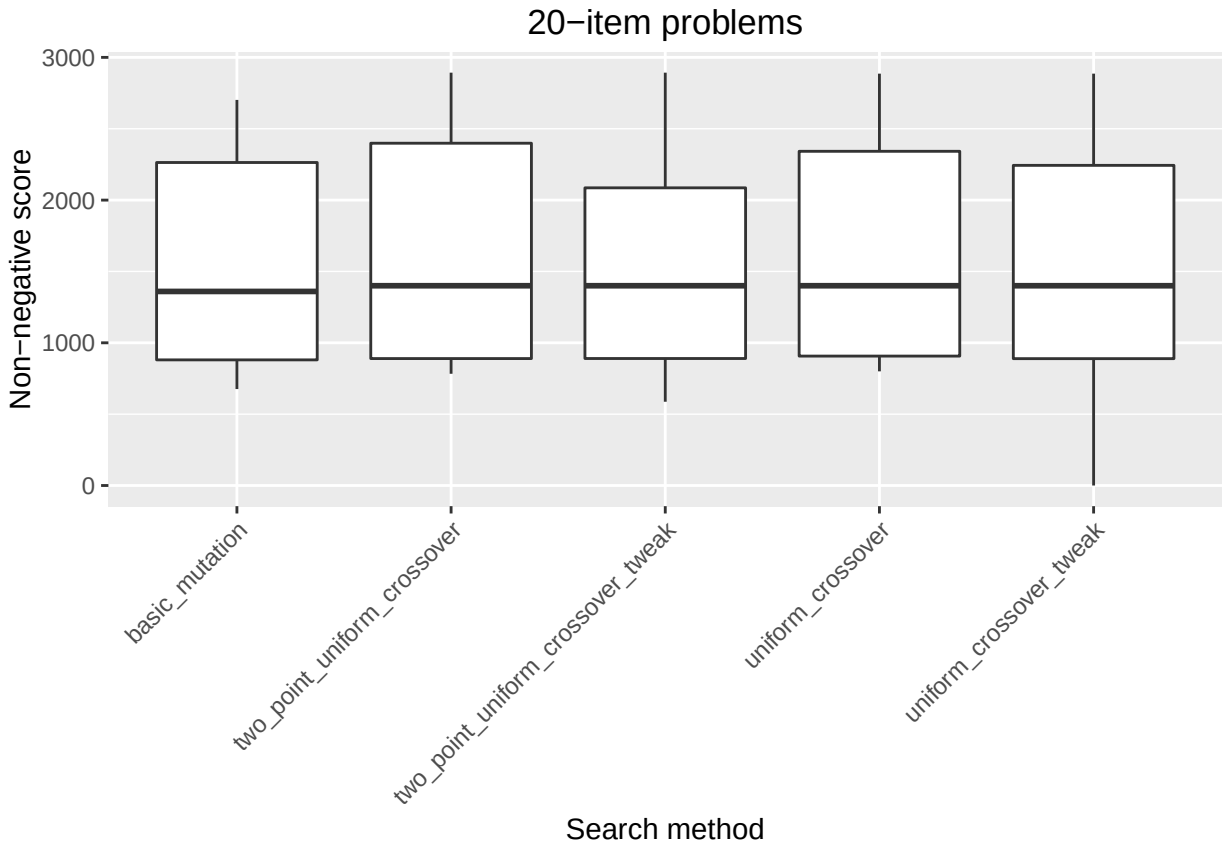
Results

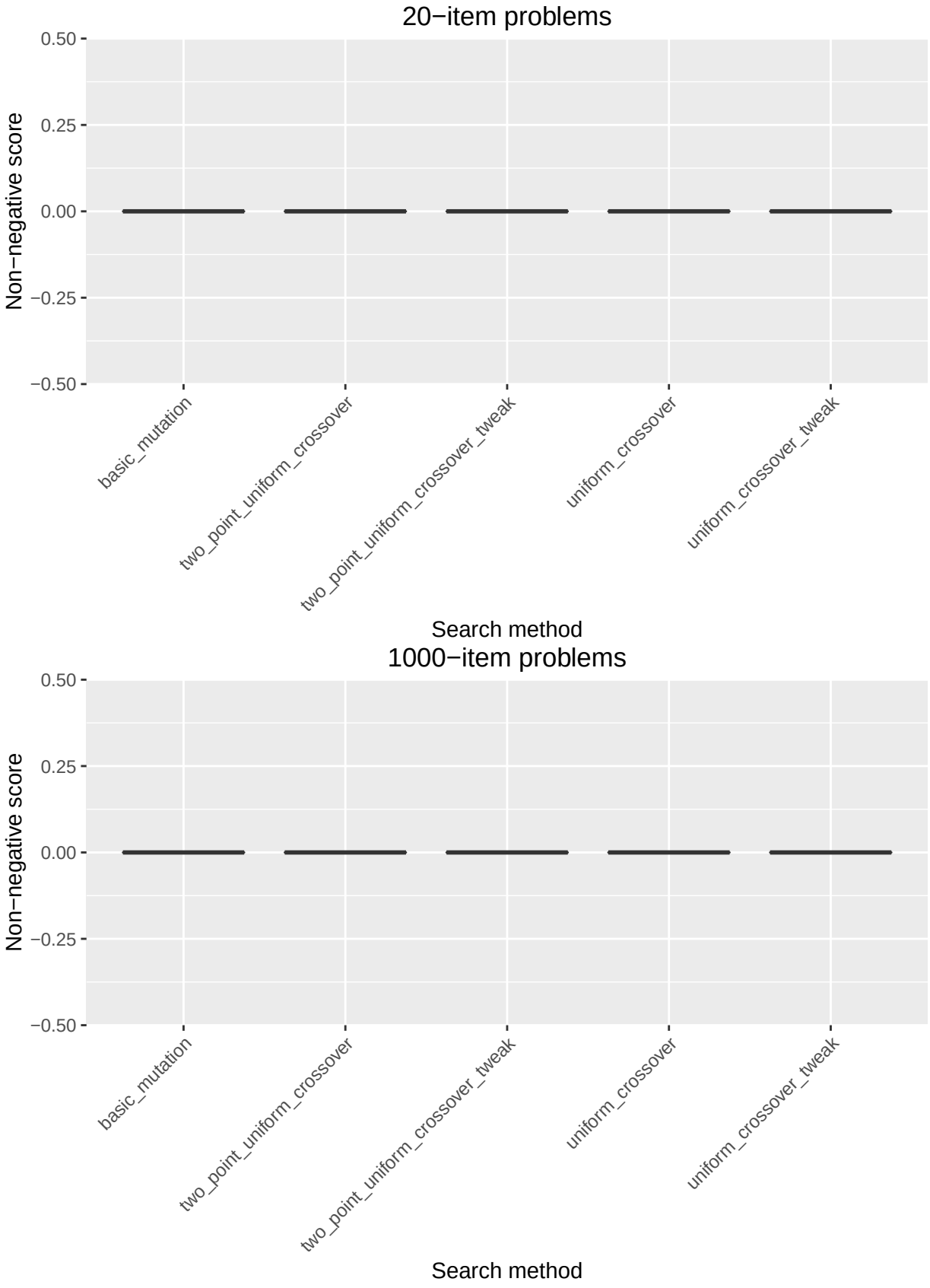
Basic comparison

Whenever the penalized scoring is applied, negative scores can be returned, meaning that no legal solution was found. In situations where the strategy was unlikely to find a legal solution (most notably random search with a large number of items), the negative scores far outweigh the positive ones. We used code from Nic's report to create a new column for the non-negative score:

```
data_25_runs <- read.csv("experiment0303.txt", sep=" ", header=TRUE)
data_25_runs$Non_negative_score = ifelse(data_25_runs$Score<0, 0, data_25_runs$Score)
```

With the negative scores removed, a simple comparison (separated by item count) looks like this:





Here are some things these plots suggest:

- Something went wrong with our 200 and 1000 datasets. We tried relentlessly to get these numbers to return positively, but were unable to.
- On our 20 item sets, two point crossover seems to do the best job followed by normal crossover.
- Algorithms with tweaks generally didn't perform as well.

To explore the significance of the results, we run a pairwise Wilcoxon rank sum test:

20-item problems

```
##
##  Pairwise comparisons using Wilcoxon rank sum test
##
## data:  items20$Score and items20$Search_method
##
##                                  basic_mutation
## two_point_uniform_crossover      1
## two_point_uniform_crossover_tweak 1
## uniform_crossover                1
## uniform_crossover_tweak          1
##                                  two_point_uniform_crossover
## two_point_uniform_crossover      -
## two_point_uniform_crossover_tweak 1
## uniform_crossover                1
## uniform_crossover_tweak          1
##                                  two_point_uniform_crossover_tweak
## two_point_uniform_crossover      -
## two_point_uniform_crossover_tweak -
## uniform_crossover                1
## uniform_crossover_tweak          1
##                                  uniform_crossover
## two_point_uniform_crossover      -
## two_point_uniform_crossover_tweak -
## uniform_crossover                -
## uniform_crossover_tweak          1
##
## P value adjustment method: holm
```

200-item problems

```
##
##  Pairwise comparisons using Wilcoxon rank sum test
##
## data:  items200$Score and items200$Search_method
##
##                                  basic_mutation
## two_point_uniform_crossover      1.00
## two_point_uniform_crossover_tweak 1.00
## uniform_crossover                1.00
## uniform_crossover_tweak          0.95
##                                  two_point_uniform_crossover
## two_point_uniform_crossover      -
## two_point_uniform_crossover_tweak 1.00
```

```
## uniform_crossover                 1.00
## uniform_crossover_tweak           1.00
##                                   two_point_uniform_crossover_tweak
## two_point_uniform_crossover       -
## two_point_uniform_crossover_tweak -
## uniform_crossover                 1.00
## uniform_crossover_tweak           1.00
##                                   uniform_crossover
## two_point_uniform_crossover       -
## two_point_uniform_crossover_tweak -
## uniform_crossover                 -
## uniform_crossover_tweak           1.00
##
## P value adjustment method: holm
```

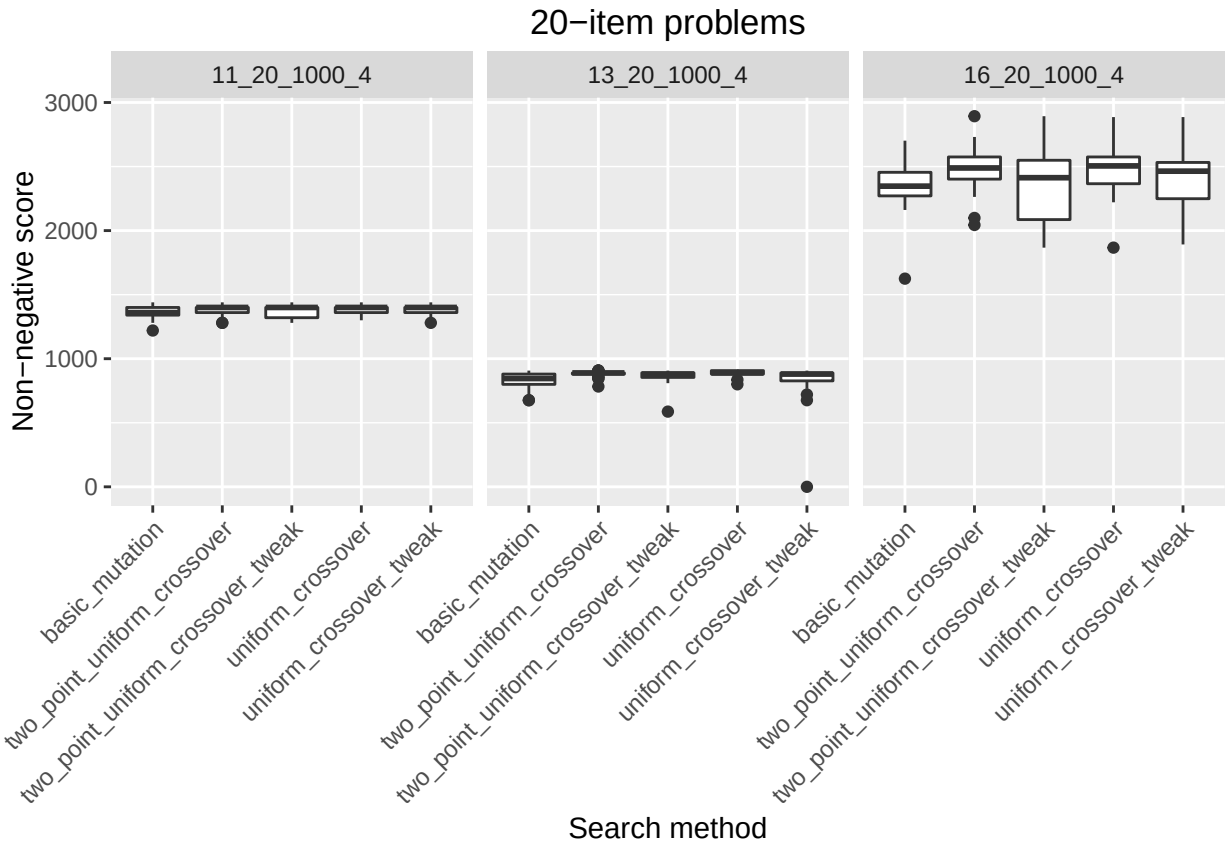
1000-item problems

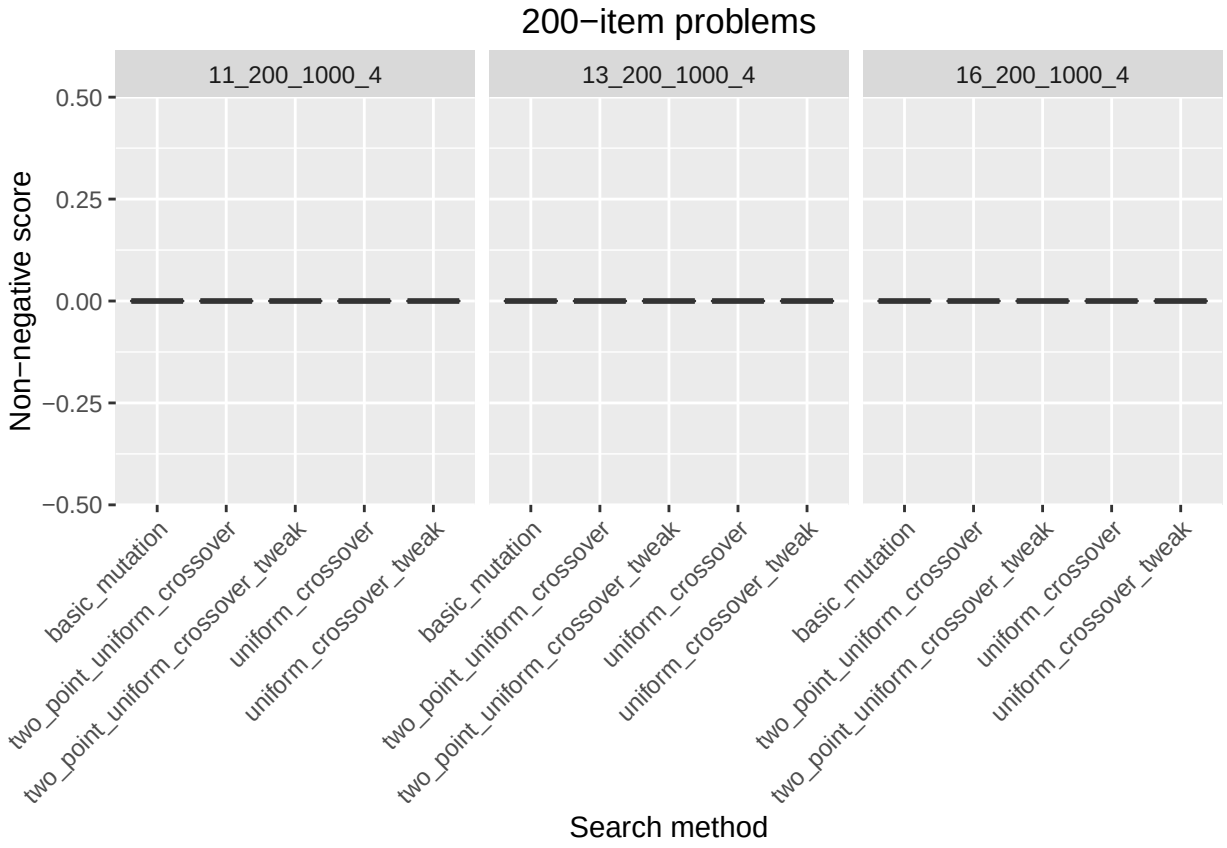
```
## Warning in wilcox.test.default(xi, xj, paired = paired, ...): cannot
## compute exact p-value with ties
```

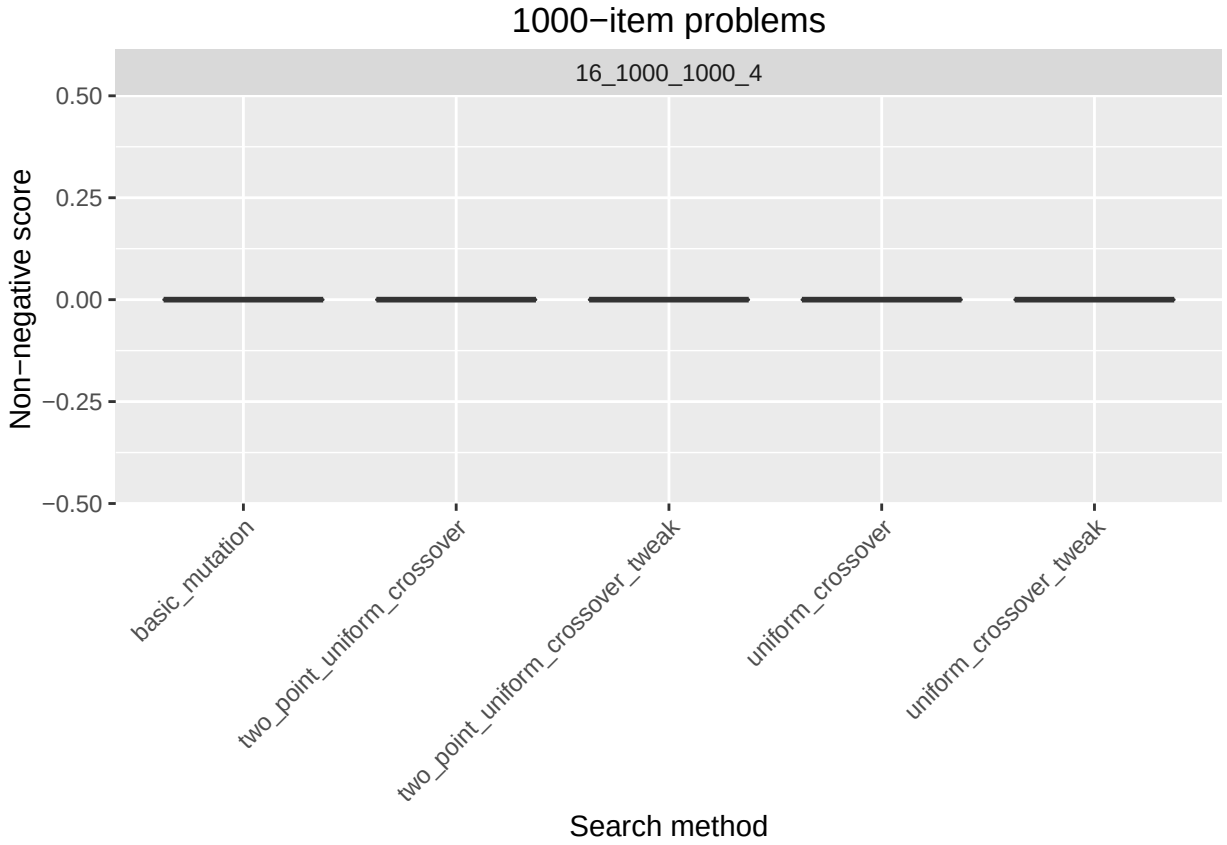
```
##
##  Pairwise comparisons using Wilcoxon rank sum test
##
## data:  items1000$Score and items1000$Search_method
##
##                                   basic_mutation
## two_point_uniform_crossover       1.00
## two_point_uniform_crossover_tweak 1.00
## uniform_crossover                 1.00
## uniform_crossover_tweak           1.00
##                                   two_point_uniform_crossover
## two_point_uniform_crossover       -
## two_point_uniform_crossover_tweak 1.00
## uniform_crossover                 0.65
## uniform_crossover_tweak           1.00
##                                   two_point_uniform_crossover_tweak
## two_point_uniform_crossover       -
## two_point_uniform_crossover_tweak -
## uniform_crossover                 1.00
## uniform_crossover_tweak           1.00
##                                   uniform_crossover
## two_point_uniform_crossover       -
## two_point_uniform_crossover_tweak -
## uniform_crossover                 -
## uniform_crossover_tweak           1.00
##
## P value adjustment method: holm
```

From the Wilcoxon test, we can observe that there isn't very much difference overall. This is most likely due to the problems with the 200 and 1000 datasets.

Comparison by problem



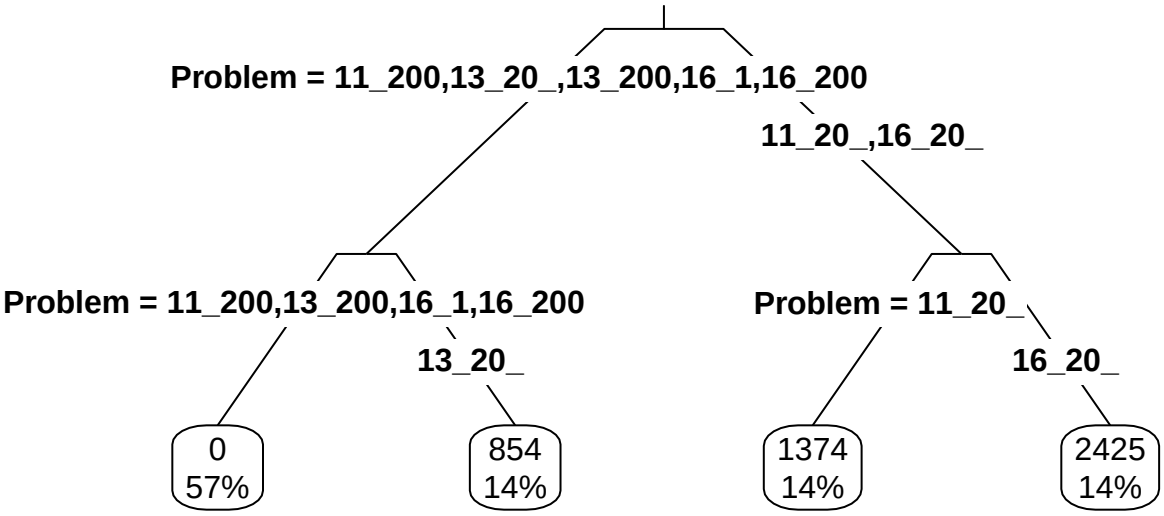

200−item problems
11_200_1000_4
13_200_1000_4
16_200_1000_4
Non−negative score
0.50
0.25
0.00
−0.25
−0.50
basic_mutation
two_point_uniform_crossover
two_point_uniform_crossover_tweak
uniform_crossover
uniform_crossover_tweak
Search method


1000−item problems
16_1000_1000_4
Non−negative score
0.50
0.25
0.00
−0.25
−0.50
basic_mutation
two_point_uniform_crossover
two_point_uniform_crossover_tweak
uniform_crossover
uniform_crossover_tweak
Search method

Comparing gives us a better idea of how individual problems did. For example, `16_20_1000_4` saw the best results with two point uniform crossover, where other problems all did around the same.

Recursive partitioning

To help determine which of these variables matter, we ran a recursive partition:



These results didn't tell us anything especially interesting. The only significant piece is that certain problems were harder than others, or there might be some bug that only allows the uniform crossover to do better.

Conclusions

Based on all of our tests, it's clear that something was wrong with our algorithm, and it was only useful for the 20 point problem.

Our data also suggests that adding a tweak to uniform crossover doesn't add any value to the population search.