Hill climbing with distributed restarts: an alternative to random-restart hill climbing

Myles Gavic, Kyle Hakala

November 03, 2016

Contents

While looking for alternatives to the hill climbing algorithm, we initially chose to implement hill climbing with random restarts (sometimes referred to as shotgun hill climbing). Classic random-restart hill climbing is built on top of the original hill climbing algorithm but does so iteratively with an initial condition x0 being chosen at random. In the case of the knapsack problem, the best solution xm is kept until a future repetition potentially finds a better solution xm than the previous best and replaces it.

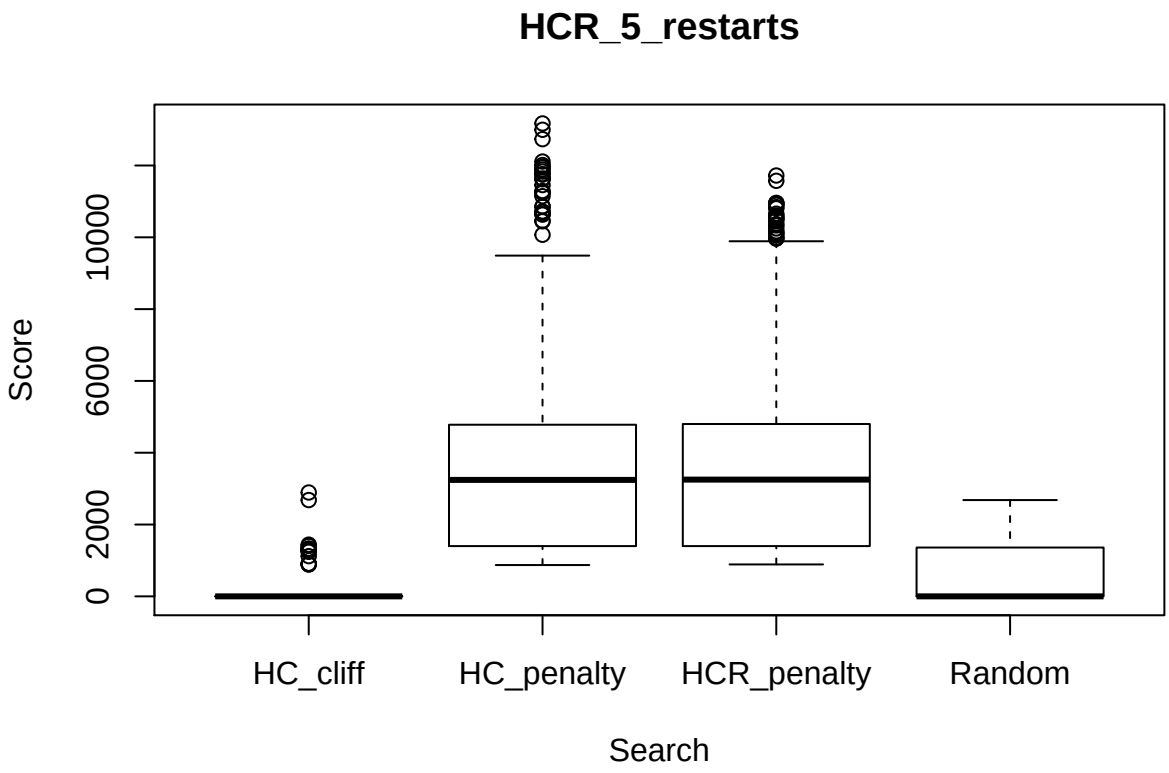
Our approach differs from random-restart hill climbing in that the initial condition is not random. Instead, the initial condition is calculated by taking the max number of tries and dividing it by the number of restarts that we are going to execute. The goal of this variant on hill climber is to place our bets on spending the time to explore the space rather than optimizing from an initial condition, and we can hoped to accomplish this without using a random condition. Ideally, this would yield results that were better than classic hill climbing.

```
data_30_runs_HC <- read.csv("~/EC-AI/simple-search/data/data_30_runs_HC.txt", sep="")
data_30_runs_HC$Non_negative_score = ifelse(data_30_runs_HC$Score<0, 0, data_30_runs_HC$Score)
data_30_runs_HCR_5 <- read.csv("~/EC-AI/simple-search/data/data_30_runs_HCR_5.txt", sep="")
data_30_runs_HCR_5$Non_negative_score = ifelse(data_30_runs_HCR_5$Score<0, 0, data_30_runs_HCR_5$Score)
data_30_runs_HCR_15 <- read.csv("~/EC-AI/simple-search/data/data_30_runs_HCR_15.txt", sep="")
data_30_runs_HCR_15$Non_negative_score = ifelse(data_30_runs_HCR_15$Score<0, 0, data_30_runs_HCR_15$Sco
data_30_runs_5_restarts <- read.csv("../data/data_30_runs_5_restarts.txt", sep="")
data_30_runs_5_restarts$Non_negative_score = ifelse(data_30_runs_5_restarts$Score<0, 0, data_30_runs_5_
data_30_runs_15_restarts <- read.csv("../data/data_30_runs_15_restarts.txt", sep="")
data_30_runs_15_restarts$Non_negative_score = ifelse(data_30_runs_15_restarts$Score<0, 0, data_30_runs_
```

Results

For the summarys I am using data sets with only penalty scores because those are what I want to compare.
##Comparison of the Searches 5 restarts

```
plot(main = "HCR_5_restarts", data_30_runs_5_restarts$Non_negative_score ~ data_30_runs_5_restarts$Searc
     xlab="Search", ylab="Score")
```




```
#Hill-Climber_penalty
summary(data_30_runs_HC$Non_negative_score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     872    1400    3244    4145    4780   13170
```

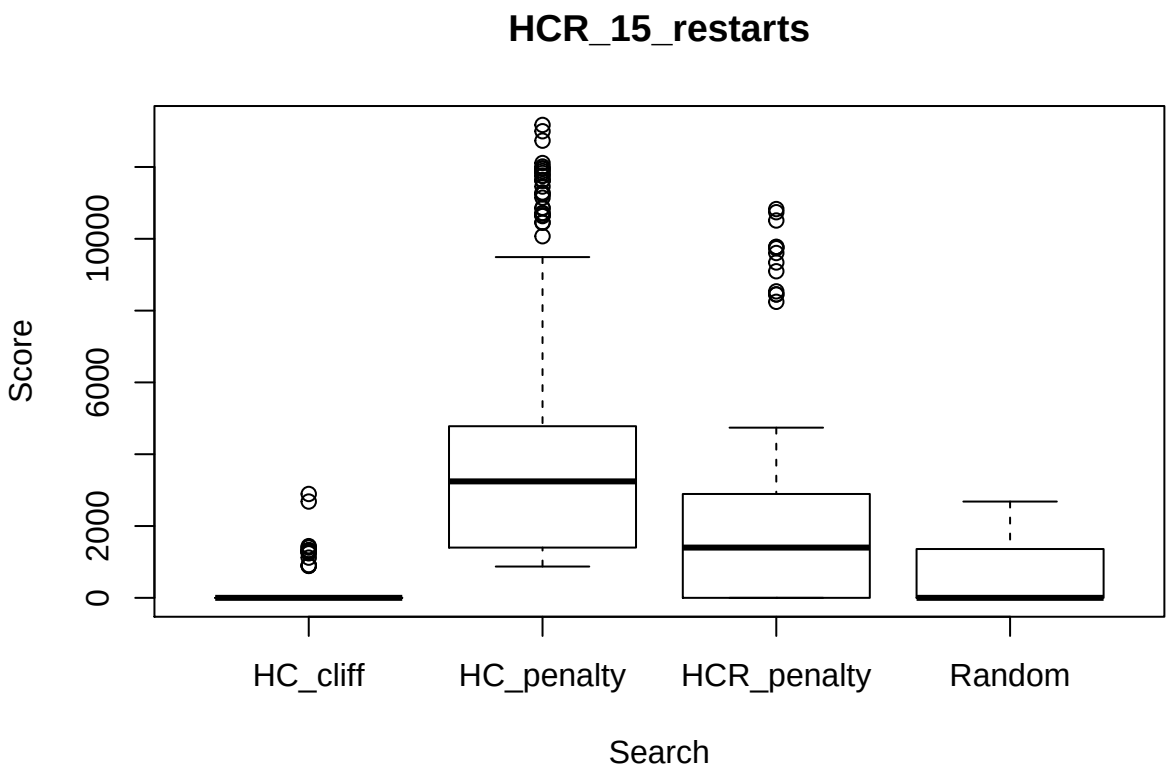
```
#Hill-Climber-5-restarts_penalty
summary(data_30_runs_HCR_5$Non_negative_score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     889    1400    3252    3993    4790   11720
```

From the plot and summarys, we can see that (HC_penalty) is returning better reslts then (HCR_penalty). However what is interesting is that (HCR_penalty) returns a higher Median, 4th Qu, and 3rd Qu. then (HC_penalty) even though it is by a small amount, it's still larger.

Comparison of the Searches 15 restarts

```
plot(main = "HCR_15_restarts", data_30_runs_15_restarts$Non_negative_score ~ data_30_runs_15_restarts$S
     xlab="Search", ylab="Score")
```




```
#Hill-Climber_penalty
summary(data_30_runs_HC$Non_negative_score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     872    1400    3244    4145    4780   13170
```

```
#Hill-Climber-15-restarts_penalty
summary(data_30_runs_HCR_15$Non_negative_score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0    1400    2001    2893   10830
```

From the plot and summarys, we can see that just like the above plot with 5 restarts (HC_penalty) is returning better reslts then (HCR_penalty). What's very interesting about (HCR_penalty) this time around is that the 1st Qu. is nonexistant. The mean, median, and upper Qu. are also significantly lower the standard (HC_penalty).

Summary: One reason this might be is that in the data for HCR, there is a significand portion of the scores that are returning negative scores. We are removing negative scores from the data set and making the data set smaller in comparison to standard HC.

Parswise Wilcox test

5 restarts

We are just going to look at the 5 restarts data set because it gives us a more significant result that we can compare to HC_penalty.

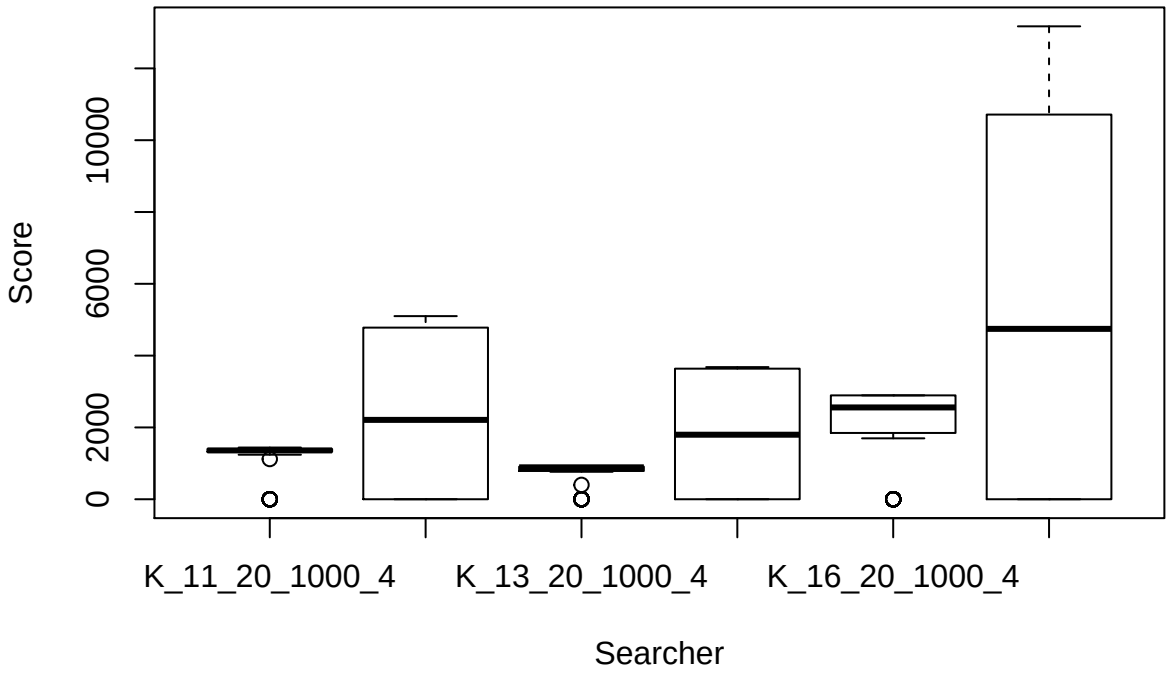
```
pairwise.wilcox.test(data_30_runs_5_restarts$Non_negative_score, data_30_runs_5_restarts$Search_method)
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test
##
## data:  data_30_runs_5_restarts$Non_negative_score and data_30_runs_5_restarts$Search_method
##
##             HC_cliff HC_penalty HCR_penalty
## HC_penalty  < 2e-16  -          -
## HCR_penalty < 2e-16  0.86       -
## Random      6.7e-16  < 2e-16    < 2e-16
##
## P value adjustment method: holm
```

Summary:The test shows us that tere is a strong dierence with $p < 2^{-16}$ in all cases except when comparing hcr_penalty with HC_penalty. This resulkts with a $p < 0.86$ that indicates a weak diference between the two search results.

Plot with searcher problems

```
plot(data_30_runs_5_restarts$Non_negative_score ~ data_30_runs_15_restarts$Problem,
     xlab="Searcher", ylab="Score")
```

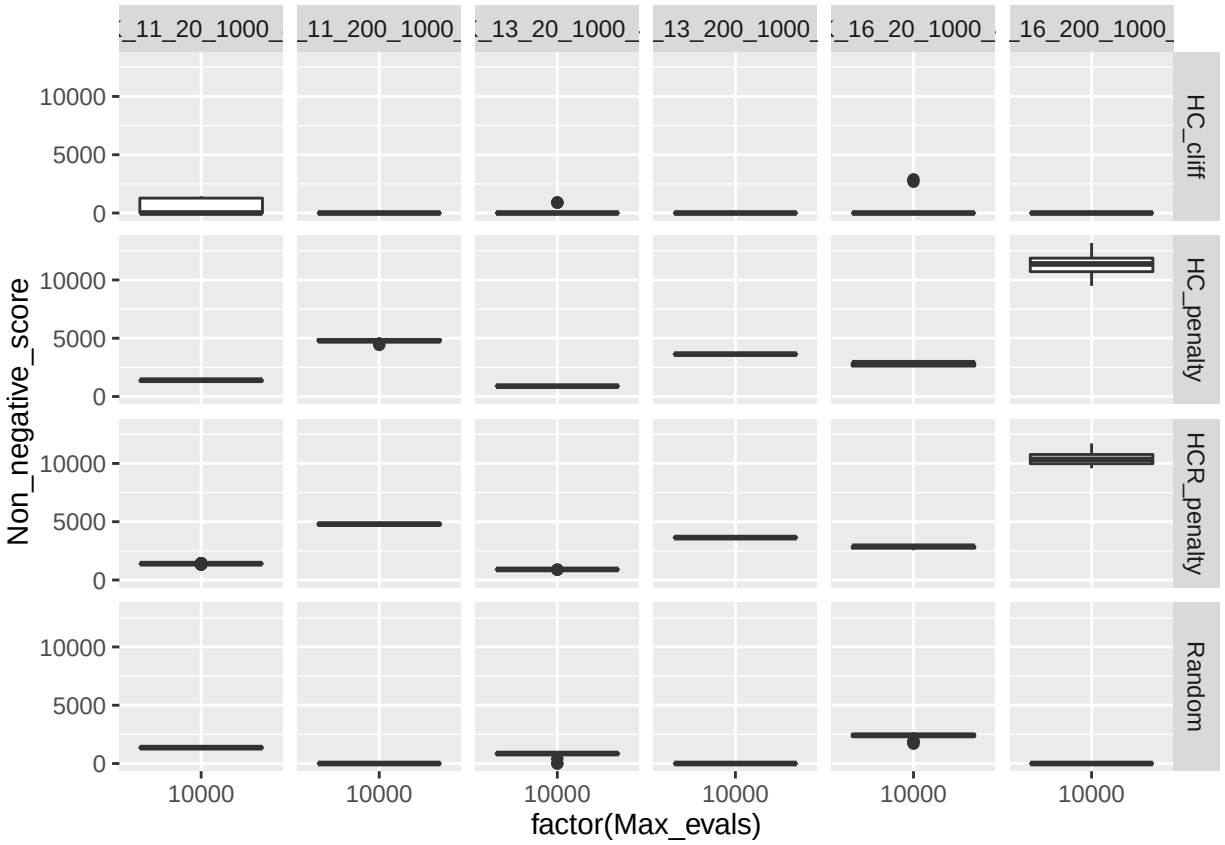


Summary: The Plot shows us that the rightmost plot is acheiving the higest values while the rest are indicating more about overall performance. The boxplots that are acheiving relitvly high valuse with respectable medians have a reduced dificulty possibly due to the addition of restarts.

ggplot

```
library("ggplot2")

ggplot(data_30_runs_5_restarts,
       aes(x=factor(Max_evals), y=Non_negative_score, group=Max_evals)) +
  geom_boxplot() + facet_grid(Search_method ~ Problem)
```



By reading the plot horizontaly we can see that HC_penalty and HCR_penalty are having much higher values for knapPI_16_200_1000_4. Reading the plot vertically shows for most knapPI, Random and HC___diff searches almost never get past zero except for knapPI_16_20_1000_4 and knapPI_11_200_1000_4.

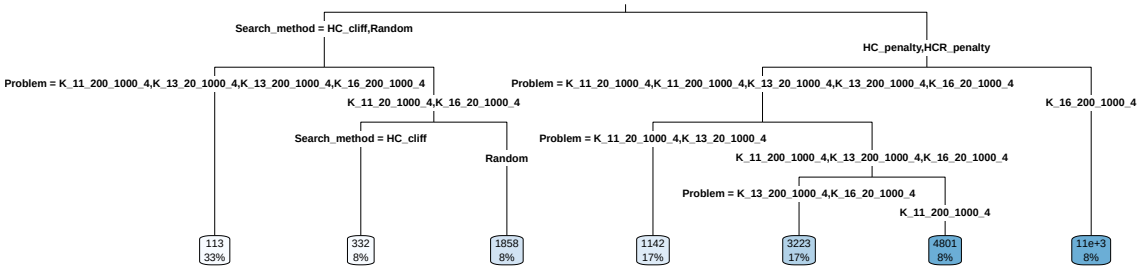
Recursive partitioning

```
library("rpart")
library("rpart.plot")
```

```
rp <- rpart(Non_negative_score ~ Search_method + Problem + Max_evals, data=data_30_runs_5_restarts)
rp
```

```
## n= 720
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 720 6585064000.0  2254.4930
##    2) Search_method=HC_cliff,Random 360  211767200.0   440.2667
##      4) Problem=K_11_200_1000_4,K_13_20_1000_4,K_13_200_1000_4,K_16_200_1000_4 240   20133660.0   11
##      5) Problem=K_11_20_1000_4,K_16_20_1000_4 120  114469900.0  1095.0080
##       10) Search_method=HC_cliff 60   27753570.0   332.2500 *
##       11) Search_method=Random 60   16900290.0  1857.7670 *
##    3) Search_method=HC_penalty,HCR_penalty 360 4003476000.0  4068.7190
##      6) Problem=K_11_20_1000_4,K_11_200_1000_4,K_13_20_1000_4,K_13_200_1000_4,K_16_20_1000_4 300  61
##       12) Problem=K_11_20_1000_4,K_13_20_1000_4 120    7111177.0  1141.6580 *
##       13) Problem=K_11_200_1000_4,K_13_200_1000_4,K_16_20_1000_4 180  122201400.0  3749.0720
##         26) Problem=K_13_200_1000_4,K_16_20_1000_4 120   21661490.0  3222.9420 *
##         27) Problem=K_11_200_1000_4 60     887093.3  4801.3330 *
##      7) Problem=K_16_200_1000_4 60   42579110.0 10881.7800 *
```

```
rpart.plot(rp, type=3, extra=100)
```



This indicates that the searcher is the most important first-order difference in the plot. There is a split between HC_penalty and HCR_pena;ty (on the right) vs. the other two searchers. After that split, though, the problems were the next most important factor along both branches. As predicted, (HC_penalty), knapPI_16_200_1000_4 resulted in the highest result.

Conclusions

Based on our results, Hill Climber penalty with restarts is better but not by much. While standard HC was obtaining a higher values, HCR seemed to be obtaining better scores baised off of the higher median, 3rd, and 4th QU in the boxplots.

However baised on these results we have concern that there is some aspects of our code that is wrong and not giving the results that we want. On paper, HC with restarts should give us a better score then standard HC. What we think is happening is that the Hill cliber is running out of runs well before it actually reaches a value it deems the max. 10000 evaluations with 5 restarts will only give each restart 2000 evalustions where as standart HC will use more.