Bash: Team Assignment

Bas Terwijn

1 Introduction

In the team assignment we will write Bash scripts to find the best parameters for solvers for the knapsack problem in git repository:

- `https://bitbucket.org/bterwijn/knapsack`

Clone this repository or "git pull" if you reuse an earlier clone so that you use the most recent version. For this assignment you will be mostly working in directory "./team/".

2 Knapsack solutions

Our goal is to find an as good as possible solution for a two dimensional Knapsack problem with 100 items. It is expected the "depth_first" algorithm will not give good solutions for 100 items in reasonable search time. Therefore, we will instead use the iterative "hill_climbing" and "simulated_annealing" algorithms to find our solution.

3 Hill Climbing

The "hill_climbing" algorithm requires two parameters to be set:

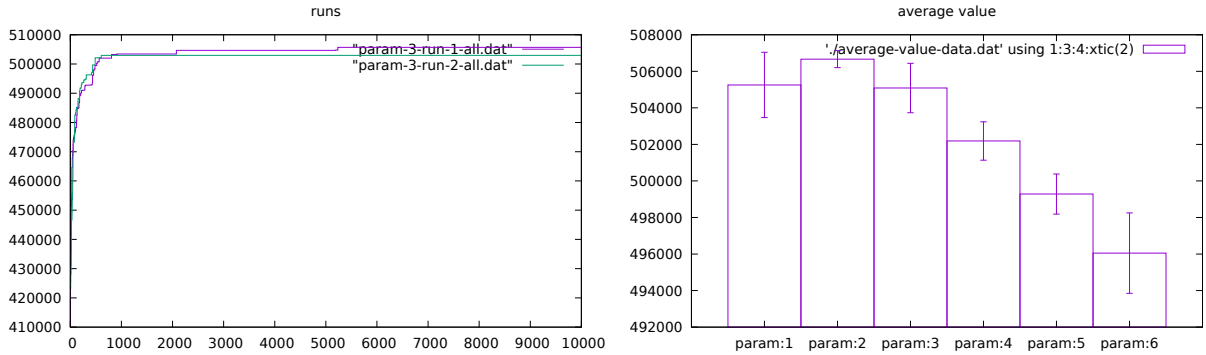
- nr_iterations: the number of iterations we run the algorithm
- nr_to_unpack: the number of items to unpack in each iteration

For "nr_iterations" we choose 10000 so that we can run many experiments in a short time. It is unclear what value for "nr_to_unpack" works best. Therefore, we will try different values to see which works best. As this is an random algorithm we will compare the average value over multiple runs with the same "nr_to_unpack" value. For this purpose write the following Bash scripts:

3.1 hc_experiments.sh

That generates 5 different runs with the "hill_climbing" algorithm for each of the values 1, 2, 3, 4, 5 and 6 for "nr_to_unpack" and saves each run in a separate file in the "hc-logs" directory. Use the following parameters:

parameter	***value(s)***
verbose	1
nr_items	100
dimensions	2
solver	hill_climbing
nr_iterations	10000
nr_to_unpack	{1, 2, 3, 4, 5, 6}



(a) hc-unpack3-runs.pdf shows two runs with nr_to_unpack=3

(b) hc-average-values.pdf shows the average final value over all runs for each nr_to_unpack value

Figure 1: Examples of the two different types of plots

3.2 hc_statistics.sh

That prepares the data for visualization. For each separate run create a file where you store the value of the knapsack at each iteration. And for each "nr_to_unpack" value create a temporary file with the final value of each separate run. Afterwards compute the average value and standard deviation of each of these temporary files and store the result in a single file.

3.3 hc_plots.sh

That makes 2 different types of plots with the data extracted by the hc_statistics.sh script using gnuplot:

- **hc-unpack<nr_to_unpack>-runs.pdf**, that shows for all runs with a specific "nr_to_unpack" value the knapsack value at each iteration. See figure 1a as example.
- **hc-average-values.pdf**, that shows the average final value and standard deviation over all the runs for each "nr_to_unpack" value. We can use this graph to see what "nr_to_unpack" value gives the best results. See figure 1b as example.

3.4 examples

In the "team/examples" directory you'll find some examples of data and how you can process and plot this data. See the "process_data.sh" script that generates figure 1a and 1b. There is no need to make the plots look pretty, they are just to analyze the results of our experiments.

4 Simulated Annealing

Now that our "hill_climbing" experiments show what the best "nr_to_unpack" value is, we can use that value also for the "simulated_annealing" algorithm. But the "simulated_annealing" algorithm has an additional parameter "max_temperature" that should be chosen. We will again try different values to see which works best. For this purpose write the following Bash scripts:

4.1 sa_experiments.sh

Analogous to hc_experiments.sh, generate 5 different runs with the "simulated_annealing" algorithm for each of the values 0, 1, 10, 100, 1000 and 10000 of "max_temperature" and saves each run in a separate file in a "sa-logs" directory. Use the following parameters:

parameter	***value(s)***
verbose	1
nr_items	100
dimensions	2
solver	simulated_annealing
nr_iterations	10000
nr_to_unpack	select best value in found hc-average-value.pdf
max_temperature	{0, 1, 10, 100, 1000, 10000}

4.2 sa_statistics.sh

Analogous to hc_statistics.sh, prepares the data for visualization. For each separate run create a file where you store the value of the knapsack at each iteration. And for each "max_temperature" value create a temporary file with the final value of each separate run. Afterwards compute the average value and standard deviation of each of these temporary files and store the result in a single file.

4.3 sa_plots.sh

Analogous to hc_plots.sh, makes 2 different types of plots with the data extracted by the hc_statistics.sh script using gnuplot:

- **sa-temp<max_temperature>-runs.pdf**, that shows for all runs with a specific "max_temperature" value the knapsack value at each iteration.
- **sa-average-values.pdf**, that shows the average value and standard deviation over all the runs for each "max_temperature" value.

5 Makefile

Add to the **"Makefile"** in the root directory so that:

- **"make hc"** takes all steps to produce the above plots of the "hill_climbing" algorithm
- **"make sa"** takes all steps to produce the above plots of the "simulated_annealing" algorithm
- **"make clean"** also removes the files generated by the scripts above (be very careful not to remove your own scripts by accident!)

6 Optionally, solve the Knapsack

Now that we know good parameters for the "simulated_annealing" algorithm we can optionally write a script to find the best possible solution to the two dimensional Knapsack problem with 100 items. Or use any other parameters or algorithms to do so. Store the string representation of your knapsack with the highest value you found in file "best_2d100_knapsack.txt" in the root directory of this repository.

7 Submission

Submit your solutions on Canvas after cleaning up the repository (make clean) and zipping it as a whole to knapsack.zip. Double check that all your solution files are included.