Hill Climbing with Montecarlo

EduHPC'22 Peachy Assignment

1 Introduction

You are provided with a sequential code that uses a probabilistic approach to find the highest point of a two dimensional function (representing for example the height of terrain) in a given rectangular area. The selected area is modelled as a set of discrete and evenly spaced points or positions. A bidimensional array is used to store the value of height provided by the function for the points. As the bidimensional array can be huge and the cost of computing the function for all the points is high, we use a probabilistic method to obtain a solution without computing all the values. The hill-climbing approach is based on selecting a starting random point, and iteratively move the selected point to one of the four neighbors: The one with the highest value of the function. Thus, the search is climbing in the direction that looks more promising to find a higher value. When the selected-point value is higher or equal than the neighbor values, a local maximum has been reached and the search stops. The program uses a probabilistic approach to find the global maximum by starting a given number of hill-climbing searches, with the hope that the highest local maximum found will be the global maximum in the selected area. Launching more searchers implies more computation but probabilities to find the correct global maximum.

In order to avoid computing the costly function for all the control points in advance, a matrix is initialized to non-valid values, and a cell is updated with the corresponding function value only when a searcher checks that point. Also, when a searcher moves to a cell already travelled by a previous searcher, it would follow exactly the same path to the same local maximum, and the searcher can stop. Ancillary structures are used to store which searcher visits each cell for the first time, and which searcher has stopped because it is following the trail of another searcher. After all searches have finished, apart from the global maximum found, the program computes the number of trail steps of all the searchers leading to the same local maximum, the total number of travelled points, and the accumulated values of height in local maximums.

Example Figure 1 shows the output of the program in debug mode for an example with an array of 40x30 cells. The first plot represents the identifiers of the first searchers that travel a cell. The second plot represents the heights computed.

2 Sequential code description

Program arguments The program arguments describe the simulation scenario:

1. `rows, columns:` Number of array positions
2. `x_min, x_max, y_min, y_max:` Limits of the rectangular area explored.
3. `searchers_density:` Ratio of searchers per lattice positions.
4. `short_rnd1, 2 and 3:` Three unsigned short numbers that are used as seeds to initialize random generators to initialize searchers starting positions.

Students are encouraged to create their own scenarios, to test different situations and problems. Several examples are provided along with the code.

```
./climb_seq 40 30 -6.28 -0.2 -6.28 -0.2 0.05 9342 234 73323

Arguments, Rows: 40, Columns: 30
Arguments, x_range: ( 0, 0 ), y_range( 32, 0 )
Arguments, searchers_density: 0.050000
Arguments, Init Random Sequence: 9342,234,7787

Trails:
+------------------------------------------------------------------------------------------------------------------------+
|  12  12  12  12  12  12  12  12  12  12  12  12  26                                  18                      40        |
|          41      57                              12  26                                  18              35      40    22|
|          41      57                                  26                                  18              35   8  40    22|
|          41      57                                                                      18              35   8  40    22|
|          41      57                                                                  15  18              35   8  40    22|
|          41      57                                                                  15  18              35   8  40    22|
|          41      57                                                                  15  15              35   8  40    22|
|                  57                                                                      15  15          35   8  40    22|
|                                   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3  35   8  40    22|
|                                                      31  11  11  11  11  11  11  11  11  11  11   3   3   8  40  22  22|
|                                                                                                       3   2   2  22    |
|                                                                                                       2   2  24  25    |
|          20                                                                                   2   2   2      24  25    |
|          20                                                                               2   2   9  53      24  25    |
|          20                                         6   6   6   6   6   6   6   2   2   2       9  53      24  25    |
|   1      20                                                                     2   2               9  53      24  25    |
|   1      20                                                              33  33                   9  53      24  25    |
|   1      20                                                          33  33                       9  53      24  25    |
|   1      20                                                      33  33                           9  53      24  25    |
|   1      20                                                                                       9  53      24  25    |
|   1      20                                              52  52  52                               9          24  25    |
|   1      20              27              55          52  52                                       9          24  37    |
|   1      20              27              55      52  52                                           9          24  37    |
|   1      20              27              55  52  52                                               9          24  37    |
|   1      20              27              52  52  59  59  59  59  59  59  59                       9          24  37    |
|   1      20   7          27          49  49  49  49  49                                          13          24        |
|   1      20   7          27  49  49  49                                                          13                    |
|   1      20   7           0   5   5   5   5   5   5   5   5   5   5  43  43                                            |
|   1      20   7   0   0   0                                                                                            |
|   1       0   0   0   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4                    |
|   0   0   0                                                                                                            |
|   0  23  23  23  23  23  23  23  23  23  23  23  39  39  39  39  39  39  39  39  39  39                                |
|  14  14  14  45  45  45  45  45  45  47  47  47  47  47  47  47  47  47  47  47  47  47  47  47  58                    |
|  48      14  14  14  16  16  16  16  16  16  30  30  30  30  30  30  30  30  30  30  30  30  30  30  30  30  30  30  30|
|  48              14  14  14                                                                                          32|
|                      50  14  14                                                                                      32|
|                      50      14  14  14                                                                              32|
|                      50              14  14  21  21  21  21  21  21  21  21  21  21  21      46                      32|
|                                          14  14  19  19  19  19  19  19  19  19  19          46  14                  32|
|                                              14  14  14  14  14  14  14  14  14  14  14  14  14  14  32  32  32  32  32|
+------------------------------------------------------------------------------------------------------------------------+

Heights:
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------  ...
|  20542  20281  20013  19739  19457  19168  18871  18564  18249  17923  17586  17237  16876  16501                                            13371  12828  12253                ...
|  17514  17269  17049  16852  16677  16523  16388  16270  16166  16073  15988  15908  15829  15747                                            14708  14433  14110         13290  ...
|         14329  14154  14032  13962  13941                                     14610  14806  15010                                            16014  16001  15923         15536  ...
|         11527  11395  11346  11375  11480                                            13832                                                   17258  17494  17650         17675  ...
|          8928   8837   8855   8976   9198                                                                                              17864  18412  18879  19252         19660  ...
|          6593   6538   6616   6820   7147                                                                                              18683  19449  20124  20692         21444  ...
|          4575   4552   4681   4957   5374                                                                                              19390  20345  21199  21936         22985  ...
|                 2923   3094   3429   3921                 6243   7260   8376   9575  10839  12152  13494  14847  16190  17506  18772  19970  21080  22081  22956  23684  24248  ...
|                               2273                 4385   5377   6491   7712   9022  10403  11838  13305  14785  16256  17698  19090  20409  21636  22749  23728  24553  25205  ...
|                                                           4808   5986   7275   8659  10117  11632  13181  14744  16299  17824  19298  20697  22001  23187  24235  25124  25833  ...
|                                                                                            11538  13125  14726  16319  17882  19392  20828  22166  23386  24465  25382  26118  ...
|                  598                                                                                                                                          25323  26052  ...
|          1100   1131   1350                                                                                                                           24078  24947  25638  ...
|          2086   2102   2295                                                                11943  13368  14805  16234  17634  18983  20262  21450  22526  23470  24263  24885  ...
|   3665   3493   3487   3644                                                         11038  12295  13580  14875  16160  17418  18627  19769  20825  21776  22603  23287  23811  ...
|   5470   5288   5254   5365                                                                12745  13851  14964  16066  17142  18172  19140  20028  20819  21496  22042  22440  ...
|   7624   7431   7363   7419                                                                                            16812  17629  18389  19077                20555  20804  ...
|  10077   9871   9766   9759                                                                                     15826  16437  17011  17534                       18863  18940  ...
|  12773  12553  12406  12330                                                                              15323  15686  16025  16331                              17002  16892  ...
|  15650  15414  15223  15073                                                                       15378  15464  15536  15586                                     15018  14706  ...
|  18641  18390  18151  17926                17312                16583                16027  15826  15611  15380  15128                                            12954  12434  ...
|  21677  21410  21125  20821         20160  19801  19424         18616  18185  17736  17269  16784  16281  15761  15222                                                   10859  10128  ...
|  24689  24405  24074  23693         22791  22270  21706         20456  19774  19058  18311  17535  16733                                                                  8781   7840  ...
|  27606  27307  26930  26476         25339  24662  23917         22238  21314  20339  19320  18262  17171  16052  14913  13758  12593                                      6768   5624  ...
|  30362  30049  29629  29104         27747  26922  26006         23922  22768  21549  20273  18949  17584  16188  14769  13337  11898  10461                               4866   3530  ...
|  32893  32567  32108  31518  30800  29958  28997  27923  26744  25468  24103  22660  21149  19579  17964  16312  14637  12950  11260                                      3120   1607  ...
|  35141  34802  34309  33661  32864  31922  30840  29627  28291  26841  25290  23647  21926  20139  18301  16423  14520  12606  10693                                      1569    -99  ...
|  37053  36705  36181  35485  34620  33592  32408  31076  29606  28010  26299  24487  22588  20616  18587  16517  14421  12314  10211   8128                              -1552  ...
|  38587  38230  37683  36947  36028  34932  33665  32238  30661  28946  27108  25160  23118  20998  18817  16593  14341  12079   9825   7594   5401   3261   1188   -808  -2717  ...
|  39705  39343  38778  38014  37056  35909  34582  33085  31430  29630  27698  25651  23505  21277  18985  16648  14283  11908   9543   7204   4907   2668    502  -1579  -3567  ...
|  40383  40017  39442  38661  37679  36501  35138  33599  31897  30044  28056  25949  23739  21446  19087  16681  14247  11805   9372   6967   4608   2309     86  -2047  -4082  ...
|  40606  40238  39660  38873  37883  36695  35320  33768  32050  30180  28173  26046  23816  21501  19120  16692  14236  11771   9316   6890   4509   2191    -49  -2201  -4251  ...
|  40366  40000  39426  38645  37663  36487  35124  33587  31885  30034  28047  25941  23734  21442  19084  16680  14248  11807   9376   6973   4615   2318     96  -2036  -4069  ...
|  39672  39309  38745  37982  37025  35880  34555  33060  31407  29609  27680  25636  23493  21269  18980  16646  14284  11913   9551   7215   4922   2686    522  -1556  -3541  ...
|  38537  38181  37635  36901  35983  34889  33625  32200  30627  28916  27082  25138  23101  20986  18810  16590  14343  12087   9837   7611   5423   3288   1218   -774  -2680  ...
|  36989                       34562  33536  32356  31028  29562  27971                                                                                                         ...
|                              32793  31854  30777  29568  28238  26794  25249  23613  21899  20120  18289  16419  14524  12618  10713   8822   6957   5129           1622         ...
|                              30719  29881  28925  27856  26684  25414  24057  22621  21118  19557  17950  16308  14642  12963  11282   9610   7955   6327   4733   3181   1674  ...
|                                     27661                       23862  22716  21506  20239  18924  17569  16183  14774  13352  11923  10495   9077   7674   6291   4934   3604  ...
|                                                                        21258  20293  19283  18236  17155  16047  14918  13774  12619  11459  10297   9139   7986   6840   5703  ...
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------  ...

Time: 0.109113
Result: 3, 40606, 285, 438, 14012463
```

Figure 1: Example of the program output in debug mode for the indicated argument values. Only the left part of the second matrix is shown in the figure due to limited space.

Results The program shows at the end of the simulation the following results. The execution time of the simulation (without initialization times) and a list of values that is used to verify the correctness of the simulation.

Debug mode If the program is compiled with the flag -DDEBUG (option included in the provided makefile) a piece of code is activated to print with ASCII art a representation of the results at the end of the program (as in the previous figure). In the sequential program it prints the results after each step of a searcher, to better understand the sequential program behaviour. The arguments and starting data are also shown to have a record of the arguments used to create the scenario.

3 Project goal

Use the parallel programming model proposed by the teacher to parallelize this program without changing the algorithm. Optimize the code and obtain the best possible performance. Always check that the results are correct and equivalent to the sequential execution with the same input arguments.

A note about random numbers and parallelism The generation of a pseudo-random sequence of numbers with the classical C library functions is inherently sequential. Our program uses library functions that compute the next random value using a small input array that stores the state of a random sequence after each call. Take care to avoid changing the order in which random numbers from a given sequence are obtained, or the program results will change and will be non correct.

Code modifications allowed Students can modify the sequential code provided as long as they observe the following restrictions:

- Exploit parallelism using only the parallel programming model proposed.
- The argument processing section, array memory allocations, time measurement points, and output of results, should not be changed. The section of code that the students should target and modify is clearly identified in the main function. This section is found between the points where the time measurement is started and ended. Functions defined in the program that are called from the target section of the code can also be modified, substituted or eliminated.
- Any change in the algorithm or data structures must be discussed with the teachers in advance, in order to avoid modifications that significantly alter the parallelism exploitation with the proposed parallel programming model, which is the purpose of the assignment.

To measure execution times, compile the program with the maximum optimization level of the compiler (for example gcc -O3). We want to focus on program changes that do not interfere, and even facilitate, compiler optimizations.