Programming Assignment 2: Priority Queues and Disjoint Sets

Revision: August 10, 2019

Introduction

In this programming assignment, you will practice implementing priority queues and disjoint sets and using them to solve algorithmic problems. In some cases you will just implement an algorithm from the lectures, while in others you will need to invent an algorithm to solve the given problem using either a priority queue or a disjoint set union.

Recall that starting from this programming assignment, the grader will show you only the first few tests (see the questions **??** and **??** in the FAQ section).

Learning Outcomes

Upon completing this programming assignment you will be able to:

1. Apply priority queues and disjoint sets to solve the given algorithmic problems.
2. Convert an array into a heap.
3. Simulate a program which processes a list of jobs in parallel.
4. Simulate a sequence of merge operations with tables in a database.

Passing Criteria: 2 out of 3

Passing this programming assignment requires passing at least 2 out of 3 programming challenges from this assignment. In turn, passing a programming challenge requires implementing a solution that passes all the tests for this problem in the grader and does so under the time and memory limits specified in the problem statement.

Contents

1 Convert array into heap

Problem Introduction

In this problem you will convert an array of integers into a heap. This is the crucial step of the sorting algorithm called HeapSort. It has guaranteed worst-case running time of $O(n \log n)$ as opposed to QuickSort's average running time of $O(n \log n)$. QuickSort is usually used in practice, because typically it is faster, but HeapSort is used for external sort when you need to sort huge files that don't fit into memory of your computer.

Problem Description

Task. The first step of the HeapSort algorithm is to create a heap from the array you want to sort. By the way, did you know that algorithms based on Heaps are widely used for external sort, when you need to sort huge files that don't fit into memory of a computer?

Your task is to implement this first step and convert a given array of integers into a heap. You will do that by applying a certain number of swaps to the array. Swap is an operation which exchanges elements a_i and a_j of the array a for some i and j. You will need to convert the array into a heap using only $O(n)$ swaps, as was described in the lectures. Note that you will need to use a min-heap instead of a max-heap in this problem.

Input Format. The first line of the input contains single integer n. The next line contains n space-separated integers a_i.

Constraints. $1 \le n \le 100\,000$; $0 \le i, j \le n - 1$; $0 \le a_0, a_1, \ldots, a_{n-1} \le 10^9$. All a_i are distinct.

Output Format. The first line of the output should contain single integer m — the total number of swaps. m **must satisfy conditions** $0 \le m \le 4n$. The next m lines should contain the swap operations used to convert the array a into a heap. Each swap is described by a pair of integers i, j — the 0-based indices of the elements to be swapped. After applying all the swaps in the specified order the array must become a heap, that is, for each i where $0 \le i \le n - 1$ the following conditions must be true:

1. If $2i + 1 \le n - 1$, then $a_i < a_{2i+1}$.
2. If $2i + 2 \le n - 1$, then $a_i < a_{2i+2}$.

Note that all the elements of the input array are distinct. Note that any sequence of swaps that has length at most $4n$ and after which your initial array becomes a correct heap will be graded as correct.

Time Limits. `C`: 1 sec, `C++`: 1 sec, `Java`: 3 sec, `Python`: 3 sec. `C#`: 1.5 sec, `Haskell`: 2 sec, `JavaScript`: 3 sec, `Ruby`: 3 sec, `Scala`: 3 sec.

Memory Limit. 512MB.

Sample 1.

Input:

```
5
5 4 3 2 1
```

Output:

```
3
1 4
0 1
1 3
```

After swapping elements 4 in position 1 and 1 in position 4 the array becomes 5 1 3 2 4.

After swapping elements 5 in position 0 and 1 in position 1 the array becomes 1 5 3 2 4.
After swapping elements 5 in position 1 and 2 in position 3 the array becomes 1 2 3 5 4, which is already a heap, because $a_0 = 1 < 2 = a_1, a_0 = 1 < 3 = a_2, a_1 = 2 < 5 = a_3, a_1 = 2 < 4 = a_4$.

Sample 2.

Input:

```
5
1 2 3 4 5
```

Output:

```
0
```

The input array is already a heap, because it is sorted in increasing order.

Starter Files

There are starter solutions only for C++, Java and Python3, and if you use other languages, you need to implement solution from scratch. Starter solutions read the array from the input, use a quadratic time algorithm to convert it to a heap and use $\Theta(n^2)$ swaps to do that, then write the output. You need to replace the $\Theta(n^2)$ implementation with an $O(n)$ implementation using no more than $4n$ swaps to convert the array into heap.

What to Do

Change the `BuildHeap` algorithm from the lecture to account for min-heap instead of max-heap and for 0-based indexing.

Need Help?

Ask a question or see the questions asked by other learners at this forum thread.

2 Parallel processing

Problem Introduction

In this problem you will simulate a program that processes a list of jobs in parallel. Operating systems such as Linux, MacOS or Windows all have special programs in them called schedulers which do exactly this with the programs on your computer.

Problem Description

Task. You have a program which is parallelized and uses n independent threads to process the given list of m jobs. Threads take jobs in the order they are given in the input. If there is a free thread, it immediately takes the next job from the list. If a thread has started processing a job, it doesn't interrupt or stop until it finishes processing the job. If several threads try to take jobs from the list simultaneously, the thread with smaller index takes the job. For each job you know exactly how long will it take any thread to process this job, and this time is the same for all the threads. You need to determine for each job which thread will process it and when will it start processing.

Input Format. The first line of the input contains integers n and m.
The second line contains m integers t_i — the times in seconds it takes any thread to process i-th job. The times are given in the same order as they are in the list from which threads take jobs.
Threads are indexed starting from 0.

Constraints. $1 \le n \le 10^5$; $1 \le m \le 10^5$; $0 \le t_i \le 10^9$.

Output Format. Output exactly m lines. i-th line (0-based index is used) should contain two space-separated integers — the 0-based index of the thread which will process the i-th job and the time in seconds when it will start processing that job.

Time Limits. `C`: 1 sec, `C++`: 1 sec, `Java`: 4 sec, `Python`: 6 sec. `C#`: 1.5 sec, `Haskell`: 2 sec, `JavaScript`: 6 sec, `Ruby`: 6 sec, `Scala`: 6 sec.

Memory Limit. 512MB.

Sample 1.

Input:

```
2 5
1 2 3 4 5
```

Output:

```
0 0
1 0
0 1
1 2
0 4
```

1. The two threads try to simultaneously take jobs from the list, so thread with index 0 actually takes the first job and starts working on it at the moment 0.
2. The thread with index 1 takes the second job and starts working on it also at the moment 0.
3. After 1 second, thread 0 is done with the first job and takes the third job from the list, and starts processing it immediately at time 1.
4. One second later, thread 1 is done with the second job and takes the fourth job from the list, and starts processing it immediately at time 2.
5. Finally, after 2 more seconds, thread 0 is done with the third job and takes the fifth job from the list, and starts processing it immediately at time 4.

Sample 2.

Input:

```
4 20
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Output:

```
0 0
1 0
2 0
3 0
0 1
1 1
2 1
3 1
0 2
1 2
2 2
3 2
0 3
1 3
2 3
3 3
0 4
1 4
2 4
3 4
```

Explanation:
Jobs are taken by 4 threads in packs of 4, processed in 1 second, and then the next pack comes. This happens 5 times starting at moments 0, 1, 2, 3 and 4. After that all the $5 \times 4 = 20$ jobs are processed.

Starter Files

The starter solutions for C++, Java and Python3 in this problem read the input, apply an $\Theta(n^2)$ algorithm to solve the problem and write the output. You need to replace the $\Theta(n^2)$ algorithm with a faster one. If you use other languages, you need to implement the solution from scratch.

What to Do

Think about the sequence of events when one of the threads becomes free (at the start and later after completing some job). How to apply priority queue to simulate processing of these events in the required order? Remember to consider the case when several threads become free simultaneously.

Beware of integer overflow in this problem: use type `long long` in C++ and type `long` in Java wherever the regular type `int` can overflow given the restrictions in the problem statement.

Need Help?

Ask a question or see the questions asked by other learners at this forum thread.

3 Merging tables

Problem Introduction

In this problem, your goal is to simulate a sequence of merge operations with tables in a database.

Problem Description

Task. There are n tables stored in some database. The tables are numbered from 1 to n. All tables share the same set of columns. Each table contains either several rows with real data or a symbolic link to another table. Initially, all tables contain data, and i-th table has r_i rows. You need to perform m of the following operations:

1. Consider table number $destination_i$. Traverse the path of symbolic links to get to the data. That is,

   while $destination_i$ contains a symbolic link instead of real data do

   $destination_i \leftarrow \texttt{symlink}(destination_i)$

2. Consider the table number $source_i$ and traverse the path of symbolic links from it in the same manner as for $destination_i$.
3. Now, $destination_i$ and $source_i$ are the numbers of two tables with real data. If $destination_i \neq source_i$, copy all the rows from table $source_i$ to table $destination_i$, then clear the table $source_i$ and instead of real data put a symbolic link to $destination_i$ into it.
4. Print the maximum size among all n tables (recall that size is the number of rows in the table). If the table contains only a symbolic link, its size is considered to be 0.

See examples and explanations for further clarifications.

Input Format. The first line of the input contains two integers n and m — the number of tables in the database and the number of merge queries to perform, respectively.
The second line of the input contains n integers r_i — the number of rows in the i-th table.
Then follow m lines describing merge queries. Each of them contains two integers $destination_i$ and $source_i$ — the numbers of the tables to merge.

Constraints. $1 \leq n, m \leq 100\,000$; $0 \leq r_i \leq 10\,000$; $1 \leq destination_i, source_i \leq n$.

Output Format. For each query print a line containing a single integer — the maximum of the sizes of all tables (in terms of the number of rows) after the corresponding operation.

Time Limits. `C`: 2 sec, `C++`: 2 sec, `Java`: 14 sec, `Python`: 6 sec. `C#`: 3 sec, `Haskell`: 4 sec, `JavaScript`: 6 sec, `Ruby`: 6 sec, `Scala`: 14 sec.

Memory Limit. 512MB.

Sample 1.

Input:

```
5 5
1 1 1 1 1
3 5
2 4
1 4
5 4
5 3
```

Output:

```
2
2
3
5
5
```

In this sample, all the tables initially have exactly 1 row of data. Consider the merging operations:

1. All the data from the table 5 is copied to table number 3. Table 5 now contains only a symbolic link to table 3, while table 3 has 2 rows. 2 becomes the new maximum size.
2. 2 and 4 are merged in the same way as 3 and 5.
3. We are trying to merge 1 and 4, but 4 has a symbolic link pointing to 2, so we actually copy all the data from the table number 2 to the table number 1, clear the table number 2 and put a symbolic link to the table number 1 in it. Table 1 now has 3 rows of data, and 3 becomes the new maximum size.
4. Traversing the path of symbolic links from 4 we have $4 \to 2 \to 1$, and the path from 5 is $5 \to 3$. So we are actually merging tables 3 and 1. We copy all the rows from the table number 1 into the table number 3, and now the table number 3 has 5 rows of data, which is the new maximum.
5. All tables now directly or indirectly point to table 3, so all other merges won't change anything.

Sample 2.

Input:

```
6 4
10 0 5 0 3 3
6 6
6 5
5 4
4 3
```

Output:

```
10
10
10
11
```

Explanation:
In this example tables have different sizes. Let us consider the operations:

1. Merging the table number 6 with itself doesn't change anything, and the maximum size is 10 (table number 1).

2. After merging the table number 5 into the table number 6, the table number 5 is cleared and has size 0, while the table number 6 has size 6. Still, the maximum size is 10.
3. By merging the table number 4 into the table number 5, we actually merge the table number 4 into the table number 6 (table 5 now contains just a symbolic link to table 6), so the table number 4 is cleared and has size 0, while the table number 6 has size 6. Still, the maximum size is 10.
4. By merging the table number 3 into the table number 4, we actually merge the table number 3 into the table number 6 (table 4 now contains just a symbolic link to table 6), so the table number 3 is cleared and has size 0, while the table number 6 has size 11, which is the new maximum size.

Starter Files

The starter solutions in C++, Java and Python3 read the description of tables and operations from the input, declare and partially implement disjoint set union, and write the output. You need to complete the implementation of disjoint set union for this problem. If you use other languages, you will have to implement the solution from scratch.

What to Do

Think how to use disjoint set union with path compression and union by rank heuristics to solve this problem. In particular, you should separate in your thinking the data structure that performs union/find operations from the merges of tables. If you're asked to merge first table into second, but the rank of the second table is smaller than the rank of the first table, you can ignore the requested order while merging in the Disjoint Set Union data structure and join the node corresponding to the second table to the node corresponding to the first table instead in you Disjoint Set Union. However, you will need to store the number of the actual second table to which you were requested to merge the first table in the parent node of the corresponding Disjoint Set, and you will need an additional field in the nodes of Disjoint Set Union to store it.

Need Help?

Ask a question or see the questions asked by other learners at this forum thread.

4 Appendix

4.1 Compiler Flags

C (`gcc 5.2.1`). File extensions: `.c`. Flags:

```
gcc -pipe -O2 -std=c11 <filename> -lm
```

C++ (`g++ 5.2.1`). File extensions: `.cc`, `.cpp`. Flags:

```
g++ -pipe -O2 -std=c++14 <filename> -lm
```

If your `C/C++` compiler does not recognize `-std=c++14` flag, try replacing it with `-std=c++0x` flag or compiling without this flag at all (all starter solutions can be compiled without it). On Linux and MacOS, you most probably have the required compiler. On Windows, you may use your favorite compiler or install, e.g., `cygwin`.

C# (`mono 3.2.8`). File extensions: `.cs`. Flags:

```
mcs
```

Go (`golang 1.12`). File extensions: `.go`. Flags

```
go
```

Haskell (`ghc 7.8.4`). File extensions: `.hs`. Flags:

```
ghc -O2
```

Java (`Open JDK 8`). File extensions: `.java`. Flags:

```
javac -encoding UTF-8
java -Xmx1024m
```

JavaScript (`Node v10.15.3`). File extensions: `.js`. No flags:

```
nodejs
```

Kotlin (`Kotlin 1.2.21`). File extensions: `.kt`. Flags:

```
kotlinc
java -Xmx1024m
```

Python 2 (`CPython 2.7`). File extensions: `.py2` or `.py` (a file ending in `.py` needs to have a first line which is a comment containing "python2"). No flags:

```
python2
```

Python 3 (`CPython 3.4`). File extensions: `.py3` or `.py` (a file ending in `.py` needs to have a first line which is a comment containing "python3"). No flags:

```
python3
```

Ruby (`Ruby 2.1.5`). File extensions: `.rb`.

```
ruby
```

Rust (`Rust 1.28.0`). File extensions: `.rs`.

```
rustc
```

Scala (`Scala 2.11.6`). File extensions: `.scala`.

```
scalac
```

4.2 Frequently Asked Questions

Why My Submission Is Not Graded?

You need to create a submission and upload the *source file* (rather than the executable file) of your solution. Make sure that after uploading the file with your solution you press the blue "Submit" button at the bottom. After that, the grading starts, and the submission being graded is enclosed in an orange rectangle. After the testing is finished, the rectangle disappears, and the results of the testing of all problems are shown.

What Are the Possible Grading Outcomes?

There are only two outcomes: "pass" or "no pass." To pass, your program must return a correct answer on all the test cases we prepared for you, and do so under the time and memory constraints specified in the problem statement. If your solution passes, you get the corresponding feedback "Good job!" and get a point for the problem. Your solution fails if it either crashes, returns an incorrect answer, works for too long, or uses too much memory for some test case. The feedback will contain the index of the first test case on which your solution failed and the total number of test cases in the system. The tests for the problem are numbered from 1 to the total number of test cases for the problem, and the program is always tested on all the tests in the order from the first test to the test with the largest number.

Here are the possible outcomes:

- `Good job!` Hurrah! Your solution passed, and you get a point!
- `Wrong answer.` Your solution outputs incorrect answer for some test case. Check that you consider all the cases correctly, avoid integer overflow, output the required white spaces, output the floating point numbers with the required precision, don't output anything in addition to what you are asked to output in the output specification of the problem statement.
- `Time limit exceeded.` Your solution worked longer than the allowed time limit for some test case. Check again the running time of your implementation. Test your program locally on the test of maximum size specified in the problem statement and check how long it works. Check that your program doesn't wait for some input from the user which makes it to wait forever.
- `Memory limit exceeded.` Your solution used more than the allowed memory limit for some test case. Estimate the amount of memory that your program is going to use in the worst case and check that it does not exceed the memory limit. Check that your data structures fit into the memory limit. Check that you don't create large arrays or lists or vectors consisting of empty arrays or empty strings, since those in some cases still eat up memory. Test your program locally on the tests of maximum size specified in the problem statement and look at its memory consumption in the system.
- `Cannot check answer. Perhaps the output format is wrong.` This happens when you output something different than expected. For example, when you are required to output either "Yes" or "No", but instead output 1 or 0. Or your program has empty output. Or your program outputs not only the correct answer, but also some additional information (please follow the exact output format specified in the problem statement). Maybe your program doesn't output anything, because it crashes.
- `Unknown signal 6 (or 7, or 8, or 11, or some other).` This happens when your program crashes. It can be because of a division by zero, accessing memory outside of the array bounds, using uninitialized variables, overly deep recursion that triggers a stack overflow, sorting with a contradictory comparator, removing elements from an empty data structure, trying to allocate too much memory, and many other reasons. Look at your code and think about all those possibilities. Make sure that you use the same compiler and the same compiler flags as we do.
- `Internal error: exception...` Most probably, you submitted a compiled program instead of a source code.
- `Grading failed.` Something wrong happened with the system. Report this through Coursera or edX Help Center.

May I Post My Solution at the Forum?

Please do not post any solutions at the forum or anywhere on the web, even if a solution does not pass the tests (as in this case you are still revealing parts of a correct solution). Our students follow the Honor Code: "I will not make solutions to homework, quizzes, exams, projects, and other assignments available to anyone else (except to the extent an assignment explicitly permits sharing solutions)."

Do I Learn by Trying to Fix My Solution?

My implementation always fails in the grader, though I already tested and stress tested it a lot. Would not it be better if you gave me a solution to this problem or at least the test cases that you use? I will then be able to fix my code and will learn how to avoid making mistakes. Otherwise, I do not feel that I learn anything from solving this problem. I am just stuck.

First of all, learning from your mistakes is one of the best ways to learn.

The process of trying to invent new test cases that might fail your program is difficult but is often enlightening. Thinking about properties of your program makes you understand what happens inside your program and in the general algorithm you're studying much more.

Also, it is important to be able to find a bug in your implementation without knowing a test case and without having a reference solution, just like in real life. Assume that you designed an application and an annoyed user reports that it crashed. Most probably, the user will not tell you the exact sequence of operations that led to a crash. Moreover, there will be no reference application. Hence, it is important to learn how to find a bug in your implementation yourself, without a magic oracle giving you either a test case that your program fails or a reference solution. We encourage you to use programming assignments in this class as a way of practicing this important skill.

If you have already tested your program on all corner cases you can imagine, constructed a set of manual test cases, applied stress testing, etc, but your program still fails, try to ask for help on the forum. We encourage you to do this by first explaining what kind of corner cases you have already considered (it may happen that by writing such a post you will realize that you missed some corner cases!), and only afterwards asking other learners to give you more ideas for tests cases.