Алгоритмын шинжилгээ ба зохиомж хичээлийн лабораторийн ажил

(F.CS301)

Д. Батмөнх

2024/12/05

Агуулга

1 1-р хичээл

- Өөрийн сонгосон программчлалын 2 хэл дээр өгөгдсөн текст файлыг унших программ бич. Жич бичсэн программ болзошгүй алдаа бүрийг мэдээлдэг байна (error handling).
- `https://leetcode.com/problems/two-sum/`
- `https://leetcode.com/problems/3sum/`

2 2-р хичээл

Энэ лабораторийн ажлын хүрээнд өмнөх лекцэд үзсэн алгоритмын хийсвэр кодыг өөрийн сонгосон 2 хэл дээр бичиж хэсэгчлэх аргыг/зохиомжийг (divide and conquer method) бататгаж, туршилтын өгөгдлийг бэлтгэсэн энгийн текст файлаас дуудаж unit test хийнэ:

- Insertion sort implementation
- Merge sort implementation
- Хоёртын хайлтыг өөрийгөө дахин дуудах аргаар бич (Recursive Binary search)
- Өгөгдсөн жагсаалтаас хамгийн их утгыг буцаах алгоритмыг бич (divide and conquer method ашиглана)

3 3-р хичээл

Өмнөх хэсэгчлэх аргаа бататгаж дараах бодлогуудыг бодно:

- `https://leetcode.com/problems/majority-element/`
- `https://leetcode.com/problems/longest-nice-substring/`
- `https://leetcode.com/problems/sort-an-array/`
- `https://leetcode.com/problems/convert-sorted-array-to-binary-search-tree/`

4 4-р хичээл

Энэхүү лабораторийн ажлаар динамик программчлалын дараах бодлогуудыг бодно:

- `https://leetcode.com/problems/climbing-stairs/`
- `https://leetcode.com/problems/min-cost-climbing-stairs/`
- `https://leetcode.com/problems/coin-change/`
- `https://leetcode.com/problems/house-robber/`

5 5-р хичээл

Энэхүү лабораторийн ажлаар динамик программчлалын аргаа бататган дараах бодлогуудыг 2 хэл дээр сөөлжлөн (нэг хэл дээр 1 бодлого, нөгөө хэл дээр 2 бодлого) бодож unit test хийнэ:

- Математикийн хувьд $F(0) = 0$ ба $F(1) = 1$, харин $n > 1$ тохиолдолд $F(n) = F(n-1) + F(n-2)$ байх Фибоначчийн тоог динамик программчлалын аргаар (cache ашиглан) тооцоол.
- Хулгайч дэлгүүрийн n тооны бараанаас, W фунтийн багтаамжтай үүргэвчээ дүүртэл авахыг хүснэ. Тэгвэл, i -р барааны үнэ нь v_i доллар ба жин нь w_i, үүнд, v_i болон w_i нь бүтэл тоонууд бол хулгайч чухам аль бараанаас авах ёстой вэ? (Энэ бодлогыг 0-1 үүргэвчтэй бодлого 0-1 knapsack problem гэдэг)
- `https://leetcode.com/problems/longest-common-subsequence/`

6 6-р хичээл

Энэхүү лабораторийн ажлаар хоёртын оновчтой хайлтын мод (OBST) болон хомхойлох (greedy) аргачлалыг бататган дараах бодлогуудыг бодно.

- `https://leetcode.com/problems/unique-binary-search-trees/`
- Эрэмбэлэгдсэн n ширхэг хайлтын түлхүүр агуулсан k жагсаалт болон түлхүүр бүрийн хайгдсан тоог агуулсан f жагсаалт өгөгджээ. Тэгвэл нийт түлхүүрийн хувьд хайлтын хамгийн бага өртгийг ол.

Жишээлбэл, $n = 2,\ k = \{5, 6\},\ f = \{17, 25\}$ бол хамгийн бага өртөг нь 59. Учир нь оройн утга нь 5 (5–6) бол $c = 17 * 1 + 25 * 2 = 67$ болно. Харин 6 бол (6–5) $c = 25 * 1 + 17 * 2 = 59$ болно.

- `https://leetcode.com/problems/best-time-to-buy-and-sell-stock-ii/`

7 7-р хичээл

Энэхүү лабораторийн ажлаар хомхойлох аргачлалыг бататган дараах бодлогуудыг бодно.

- n тооны бараа тус бүрийн үнэ болон жин өгөгджээ. Тэгвэл тэдгээр бараанаас w багтаамжтай үүргэвчинд цуглуулахдаа нийлбэр нь хамгийн их байхаар сонго. 5-р хичээлийн хоёрдугаар бодлогоос ялгаатай нь хэрэв тухайн бараа үүргэвчинд багтахгүй бол хувааж болно. (Үүнийг fractional knapsack гэдэг.)

- `https://leetcode.com/problems/task-scheduler/`

- `https://www.hackerrank.com/challenges/tree-huffman-decoding/`

8 8-р хичээл

Энэхүү лабораторийн ажлаар өмнө үзсэн аргуудаа бататгаж дараах бодлогуудыг 2 хэл дээр сөөлжлөн бодож unit test хийнэ:

- Ялгаатай нэгж бүхий зоос жагсаалтаар өгөгджээ. Тэгвэл өгөгдсөн бүхэл тоон мөнгийг, хамгийн багадаа хэчнээн зоосоор сольж болохыг динамик программчлалын дээрээс доош аргачлалаар бод. Жишээлбэл: $coins = [1, 2, 5],\ amount = 11$ гэвэл гаралт нь 3 болно. Учир нь $11 = 5 + 5 + 1$.

- Нэгээс их, өөрөөсөө бусад тоонд үл хуваагдах дэс тоог анхны тоо гэдэг. Тэгвэл өгөгдсөн тоо хүртэл нийт хэчнээн анхны тоо байгааг динамик программчлалын аргаар бодож ол. Жишээ нь, оролтод 18 гэсэн утга өгвөл гаралтад 7 гэсэн утга буцаана. Учир нь 18 хүртэлх анхны тоо нь: $2, 3, 5, 7, 11, 13, 17$.

- n оюутан, m дугуй ($n \leq m$) бүхий сургуулийн хотхон 2 хэмжээст солбицолд өгөгджээ. 1 оюутан өөрт хамгийн ойр байрлах 1 дугуйг унах ёстой бөгөөд хэрэв 1 дугуйг олон оюутан сонгоход хүрвэл (хамгийн ойр зай нь ижил тул) хамгийн бага эрэмбэтэй (индекстэй)

оюутанд нь, хэрэв 1 оюутан олон дугуйг сонгоход хүрвэл хамгийн бага эрэмбэтэй дугуйг сонго. Өгөгдсөн $d_1,\ d_2$ цэгүүдийн хувьд хамгийн ойр зайг тооцохдоо $|d_1.x-d_2.x|+|d_1.y-d_2.y|$ томьёог ашиглана. Жишээлбэл, оюутнуудын солбицол: $[(0,0),(1,1)]$, үүнд $(0,0)$ нь 0-р оюутны солбицол, $(1,1)$ нь 1-р оюутны солбицол болно. Харин дугуйн хувьд: $[(0,1),(4,3),(2,1)]$ гэдэг нь 0 дэх дугуй $(0,1)$, 1 дэх дугуй $(4,3)$, 2 дахь дугуй $(2,1)$ солбицол дээр тус тус байрлахыг заах бөгөөд бодлогын хариу нь $[0,2]$ гэдэг нь, 0-р оюутан 0 дэх дугуйг, 1-р оюутан 2 дахь дугуйг унана гэсэн утгатай. Алгоритмынхаа ажиллах хугацааг мөн тодорхойл.

9 9-р хичээл

Энэхүү лабораторийн ажлаар нэгжийн шинжилгээг (amortized analysis) бататган ойлгож дараах бодлогыг бодно:

- `https://leetcode.com/problems/implement-queue-using-stacks/`
- `https://leetcode.com/problems/implement-stack-using-queues/`
- `https://leetcode.com/problems/linked-list-cycle/`

10 10-р хичээл

Энэхүү лабораторийн ажлаар нэгжийн шинжилгээг бататган дараах дасгал ажлыг гүйцэтгэнэ:

- Нэгтгэсэн шинжилгээ (aggregate analysis) ашиглан n үйлдэл бүхий, хэрэв i дүгээр үйлдлийн хувьд i нь хоёрын зэрэгтэй тэнцэх тохиолдолд i өртөг, бусад тохиолдолд 1 байх дарааллын үйлдлийн нэгжийн өртгийг тооцоол.
- k тооноос үл хэтрэх эгэх дарааллын (стек) хувьд Push болон Pop үйлдлүүдийг гүйцэтгэхдээ k үйлдэл бүрийн дараа дарааллаа автоматаар нөөцөлж хуулбарлана гэвэл санхүүгийн шинжилгээний аргаар, n үйлдлийн дараа нэгж өртгийг ялгаатай үйлдлүүдэд хуваарилахад өртөг нь $O(n)$ байна гэж батал.
- Дурын i тооны хувьд $\Phi(D_0) \neq 0,\ \Phi(D_i) \geq \Phi(D_0)$ байх Φ боломжит функц (potential function) өгөгджээ. Тэгвэл $i \geq 1$ байх i тооны хувьд $\Phi'(D_0) = 0,\ \Phi'(D_i) \geq 0$ байх Φ' функц оршин байгаад Φ'

функцийн нэгжийн өртөг нь Φ функцийн нэгжийн өртөгтэй тэнцүү байна гэдгийг батал.

11 11-р хичээл

Энэхүү лабораторийн ажлаар графын талаарх мэдлэгээ бататгаж дараах бодлогуудыг BFS буюу DFS аргачлал ашиглан бодно:

- `https://leetcode.com/problems/island-perimeter/`
- `https://leetcode.com/problems/valid-parenthesis-string/`
- `https://leetcode.com/problems/open-the-lock/`
- `https://leetcode.com/problems/course-schedule-ii/`

12 12-р хичээл

Энэхүү ажлаар топологи эрэмбэлэлтийн аргаар дараах бодлогыг бодно:

- Өгөгдсөн чиглэлт битүү биш графын хувьд топологи эрэмбэлэлтийг DFS ашиглан хий. Жишээ нь:

  $V = 6,\ E = \{\{2,3\},\{3,1\},\{4,0\},\{4,1\},\{5,0\},\{5,2\}\}$ өгөгдсөн бол буцаах утга нь: 4 5 2 0 3 1
- `https://leetcode.com/problems/course-schedule/`
- `https://leetcode.com/problems/course-schedule-ii/`
- `https://leetcode.com/problems/course-schedule-iv/`

13 13-р хичээл

Энэхүү лабораторийн ажлаар Хамгийн бага үнэлгээт бүрхэлтийн мод (MST) ашиглан дараах бодлогуудыг бодно:

- `https://leetcode.com/problems/min-cost-to-connect-all-points/`
- `https://www.hackerrank.com/challenges/connecting-towns/problem`

14 14-р хичээл

Энэхүү лабораторийн ажлаар Крускал болон Примийн алгоритм ашиглан дараах бодлогуудыг бодно:

- `https://www.hackerrank.com/challenges/kruskalmstrsub/problem`
- `https://www.hackerrank.com/challenges/primsmstsub/problem`

15 Хавсралт: Unit test хийх жишээ

Python хэл дээр:

```
import unittest


def insertion_sort(A, n):
    for i in range(1, n):
        key = A[i]
        j = i - 1
        while j >= 0 and A[j] > key:
            A[j + 1] = A[j]
            j = j - 1
        A[j + 1] = key


class TestInsertionSort(unittest.TestCase):
    def test_insertion_sort(self):
        lst = [12, 3, 7, 9, 14, 6, 11, 2]
        insertion_sort(lst, len(l))
        self.assertEqual([2, 3, 6, 7, 9, 11, 12, 14], lst)


if __name__ == "__main__":
    unittest.main()
```

Java хэл дээр:

```
// InsertionSort.java
public class InsertionSort {
    public void insertionSort(int[] A, int n) {
        for (int i = 1; i < n; ++i) {
            int key = A[i];
            int j = i - 1;
            while (j >= 0 && A[j] > key) {
                A[j + 1] = A[j];
                j--;
            }
            A[j + 1] = key;
        }
    }
}

// InsertionSortTest.java
import org.junit.jupiter.api.Assertions;
import org.junit.jupiter.api.Test;

public class InsertionSortTest {
    @Test
    public void sortArray() {
        int[] array = {12, 3, 7, 9, 14, 6, 11, 2};
        InsertionSort insertionSortTests = new InsertionSort();
        insertionSortTests.insertionSort(array, array.length);
        Assertions.assertArrayEquals(new int[]{2, 3, 6, 7, 9, 11, 12, 14}, array);
    }
}
```

C++ хэл дээр шалгахдаа

```
// CMakeLists.txt
cmake_minimum_required(VERSION 3.25)
project(InsertionSort)

Include(FetchContent)

FetchContent_Declare(
        Catch2
        GIT_REPOSITORY https://github.com/catchorg/Catch2.git
        GIT_TAG        v3.4.0 # or a later release
)

FetchContent_MakeAvailable(Catch2)

add_executable(InsertionSort main.cpp)
target_link_libraries(InsertionSort PRIVATE Catch2::Catch2WithMain)

set(CMAKE_CXX_STANDARD 20)
```

бэлтгээд

```
cmake .
make
```

командаар төсөлд шаардлагатай файлуудаа үүсгэнэ.

```
// main.cpp
#include <catch2/catch_test_macros.hpp>
#include <catch2/matchers/catch_matchers_vector.hpp>
#include <vector>

void insertionSort(std::vector<int> A, int n) {
    for (int i = 1; i < n; ++i) {
        int key = A[i];
        int j = i - 1;
        while (j >= 0 && A[j] > key) {
            A[j + 1] = A[j];
            j = j - 1;
        }
        A[j + 1] = key;
    }
}


TEST_CASE (
"Array sort"
)
 {
    std::vector<int> array = {12, 3, 7};
    insertionSort(array, sizeof(array));
    REQUIRE_THAT(array, Catch::Matchers::UnorderedEquals(std::vector<int>{3, 7, 12}));
}
```