Parallel programming

Homework 1 – Data storage optimization

CIIRC CTU
Industrial Informatics Department

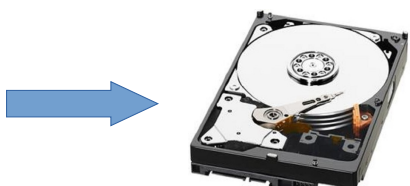


ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Storing lot of data

- We are given *n* records
  - A *record* is a sequence of integers, e.g.,
    - `1, 6, 3, 5, 1, 4`
- Our goal is to store them in disk in the most memory-efficient way possible

```
1, 6, 3, 5, 1, 4
8, 2, 3, 1
2, 3, 1, 0
3, 0, 9
```



Storing records efficiently

- How to store the records in memory-efficient way?
  - Find the *edit difference* between the records and store only the differences
    - Edit difference = Levenshtein distance
  - The original records then can be restored by re-applying the differences

```
1, 6, 3, 5, 1, 4
8, 2, 3, 1
2, 3, 1, 0
3, 0, 9
```



```
1, 6, 3, 5, 1, 4
8, 2, 3,  , 1,

1, 6, 3, 5, 1, 4
 , 2, 3,  , 1, 0

1, 6, 3, 5, 1, 4
 ,  , 3,  , 0, 9
```

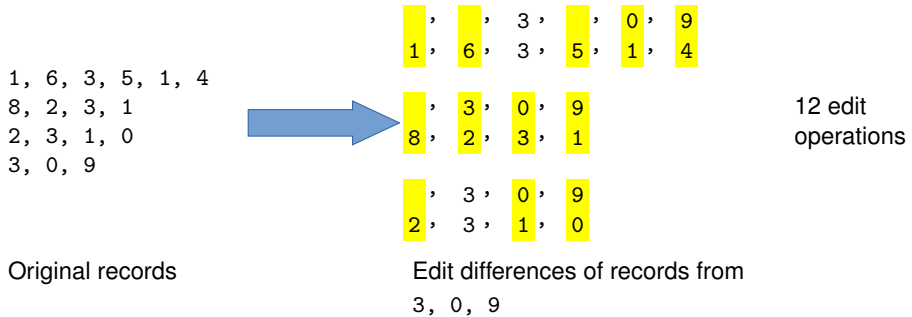
13 edit operations

Original records

Edit differences of records from `1, 6, 3, 5, 1, 4`

Which differences??!

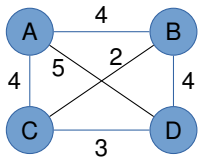
- Clearly, the number of stored edit differences depends on the record from which the difference is computed



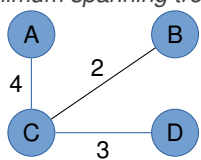
Tree of distances

- Better approach
  - Compute the edit difference between every pair of records

```
A: 1, 6, 3, 5, 1, 4
B: 8, 2, 3, 1
C: 2, 3, 1, 0
D: 3, 0, 9
```



  - Find the *minimum spanning tree* on the complete graph (e.g., using Prim's algorithm)



9 edit operations

Your assignment

- Implement **parallel** version of the data storage optimization using **OpenMP**
- Create a **complete graph** by computing the **edit difference** between **every pair** of records
- Find the minimum spanning tree on the complete graph – return the **cost** of the **minimum spanning tree**
- Use the prepared template `Storage.cpp`
- The instance generator is available for you `InstanceGenerator.cpp`