EDA Final

周昀、劉芸欣、高紹芳

Outline

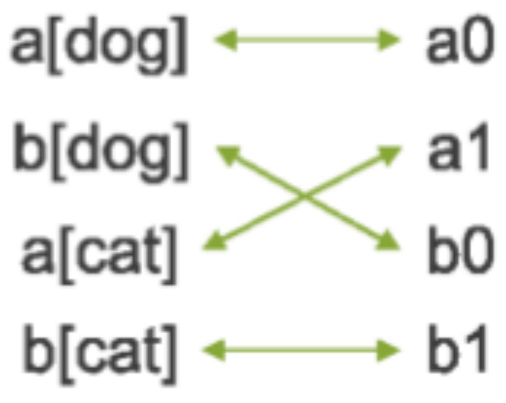
- Problem define
- Reference and possible approach
- Our approach
- Current progress

Outline

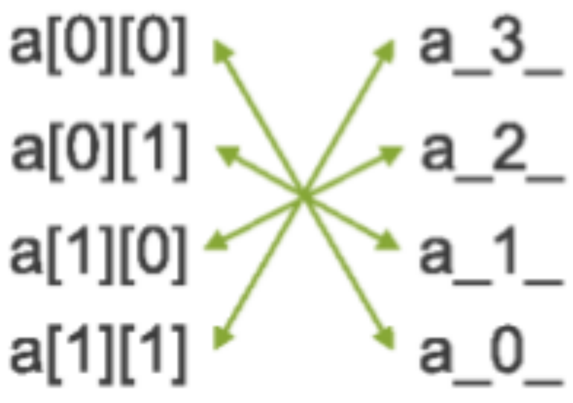
- Problem define
- Reference and possible approach
- Our approach
- Current progress

2018 ICCAD Problem A

- The name mapping problem



Example.1



Example.2

Outline

- Problem define
- Reference and possible approach
- Our approach
- Current progress

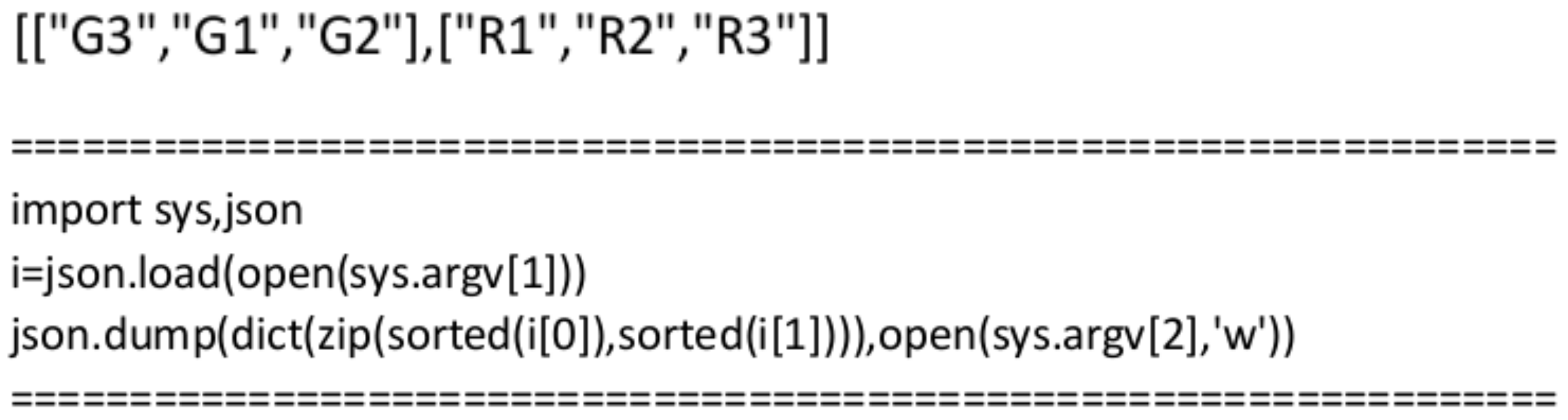
Possible approach

- Find the mathematical function of input and output
  - The solution space is too big. Related papers usually focus on binary output.
- Machine learning model
  - It’s really hard to use a well developed model and tune the function for 100% match.
- Regular expression
  - Might help in some level.

Outline

- Problem define
- Reference and possible approach
- Our approach
- Current progress

Sorting



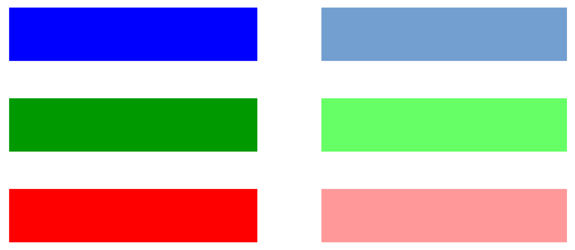
```
[["G3","G1","G2"],["R1","R2","R3"]]

============================================================
import sys,json
i=json.load(open(sys.argv[1]))
json.dump(dict(zip(sorted(i[0]),sorted(i[1]))),open(sys.argv[2],'w'))
============================================================
```

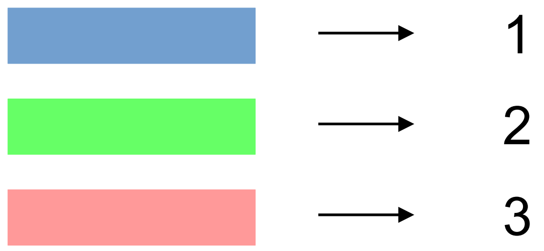
- If we can find and record a certain sequence of the string, then we can simply match the strings by sorting them according to that sequence.

Basic Idea

- First we sort the string pairs according to the first strings.

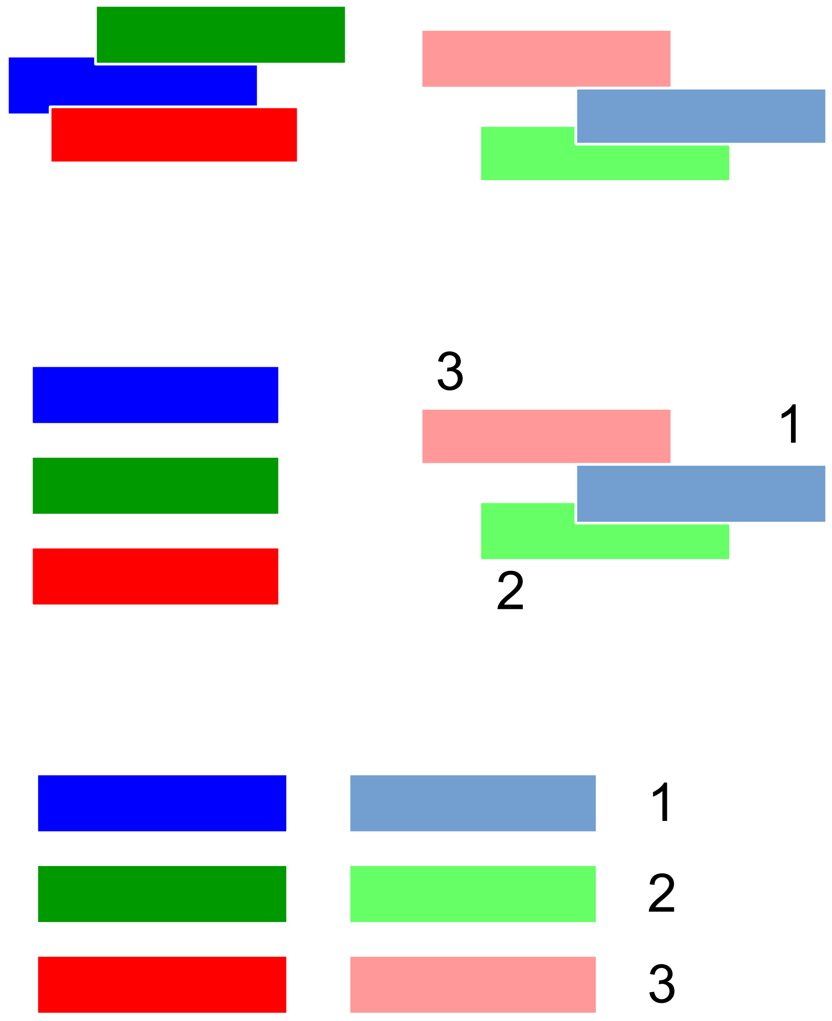


- We try to build a hash function for the second strings where the hash number of a string reflects its order.



Basic Idea

- As we get the two groups of strings.
- Sort the first group, and find a hash number for each string in the second group according to our hash function.
- Sort the second group according to those numbers.



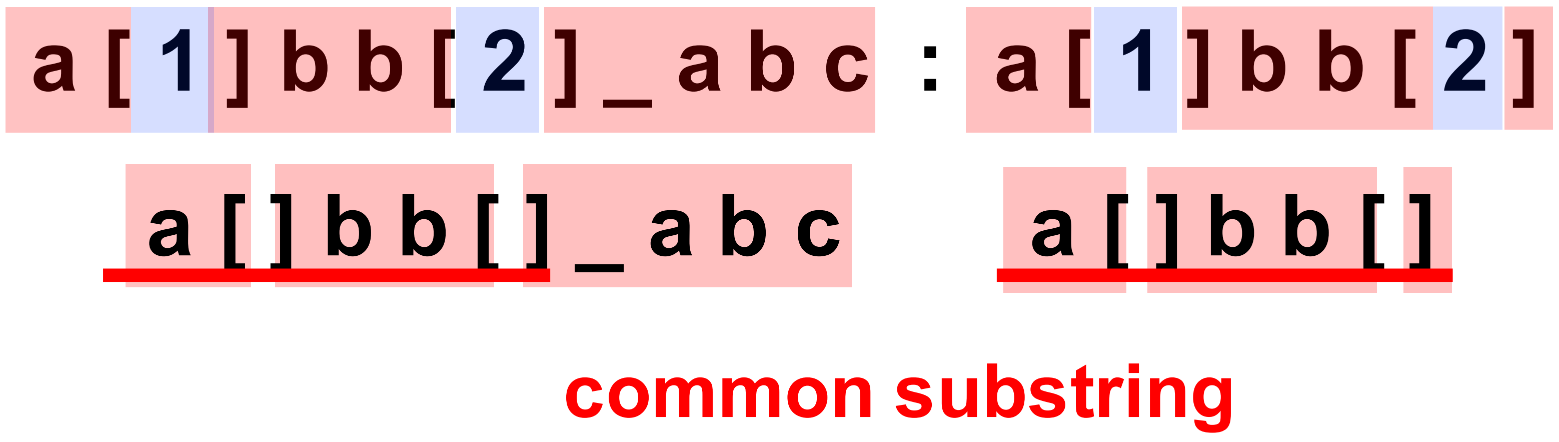
Name Group

- Critical observation from testcases:
  the given name pairs usually have a long common substring.
- We only need to consider the different part.
- Group some name pairs to reduce the complexity of the hash functions.

Name Group

- Cut the name into numbers and substring without number.
- Group the name pair together with the same longest common substring.

a [1] b b [2] _ a b c : a [1] b b [2]



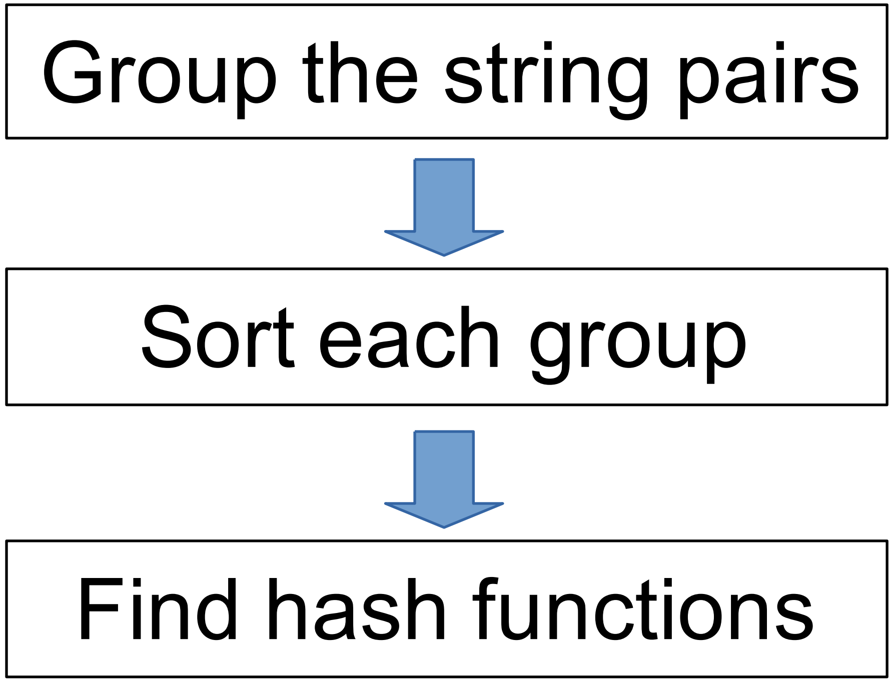
Find the Hash Functions

- Slice the string into array of strings and numbers. We see each character as one number, so the input becomes a vector.
- Objective:

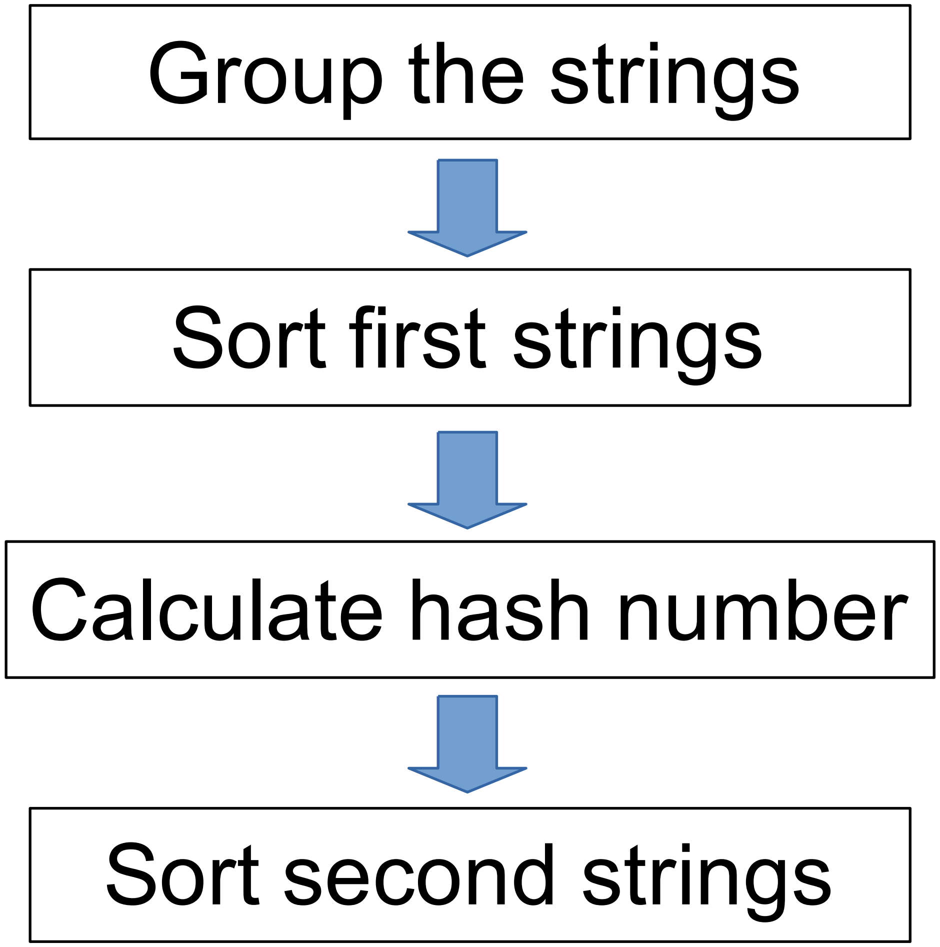
  Find a vector f such that for any two vector x_1 and x_2, if $x_1 > x_2$, then $x_1^T f > x_2^T f$.
- We can solve it by linear programming!

Workflow

Main Program :



Python Script :



Outline

- Problem define
- Reference and possible approach
- Our approach
- Current progress

Progress & Goal

- Parsing and data structure defining finished.
- Finish the implementation (that give the correct answers for the testcases) before alpha test.

Thank You