Experimental Repeatability Instructions

Welcome to **Faerie**, an efficient filtering method for approximate dictionary-based entity extraction. All the codes were implemented in C++, and all the algorithms can be run on either Windows or Linux platforms. Here we provide the binary code on Linux (In our experiments, we used Ubuntu 8.0.4.) and introduce how to run our code and redraw the figures in our paper.

➢ Experimental Requirements

1. Software: GCC (4.2.4 or above): Running the algorithms.

   Gnuplot (4.2 http://www.gnuplot.info/): Redrawing the figures.

2. Hardware: 1G RAM (2G for NGPP)
3. Datasets:
   - DBLP
     - Dictionary: 100k author names (./dataset/dblp.dict)
     - Document: 10k paper records (./dataset/dblp.doc)
   - WebPage
     - Dictionary: 100k paper titles (./dataset/webpage.dict)
     - Document: 1k publication pages (./dataset/webpage.doc)
   - Pubmed
     - Dictionary: 100k paper titles (./dataset/pubmed.dict)
     - Document: 10k paper records (./dataset/ pubmed.doc)

➢ Algorithms

1 **Faerie:** ./bin/faerie 〈Similarity Function〉 [q-gram] 〈Threshold〉 〈Dictionary〉 〈Document 〉 〈Multi|Single|Lazy|Bucket|Binary|Best|Scale〉

Parameters:

- 〈Similarity Function〉:
  EditDistance/EditSimilarity/Jaccard/Cosine/Dice
- [q-gram]: the length of a gram. Valid for Edit Distance and Edit Similarity.
- 〈Threshold〉: Threshold of a similarity function. Integer for EditDistance; float in (0,1] for other functions.
- 〈Dictionary〉 : The file path and name of the Dictionary.
- 〈Document〉 : The file path and name of the Document.
- 〈Multi|Single|Lazy|Bucket|Binary|Best|Scale〉:

  Multi: Our method using multiple heaps

  Single: Our method using a single heap without pruning

  Lazy: Our method using a single heap with lazy-update pruning

  Bucket: Our method using a single heap with bucket pruning

  Binary: Our method using a single heap with a binary search

  Best: Our best method with all pruning techniques

  Scale: Test scalability

Example: To evaluate faerie using the binary search method on the dblp dataset with edit-distance threshold 2 and 5-gram, one can use the following command.

```
./bin/faerie EditDistance 5 2 dataset/dblp.dict dataset/dblp.doc Binary
```

2 **ISH**: ./bin/ish ⟨Similarity Function⟩ [q-gram] ⟨Threshold⟩ ⟨Dictionary⟩ ⟨Document⟩

Parameters:

- ⟨Similarity Function ⟩: EditSimilarity/Jaccard
- [q-gram]: The length of a gram. Valid for Edit Similarity.
- ⟨Threshold⟩: Threshold in (0,1] of a similarity function.
- ⟨Dictionary⟩ : The file path and name of the Dictionary.
- ⟨Document⟩ : The file path and name of the Document.

Example: To evaluate ISH on the webpage dataset with Jaccard similarity threshold 1, one can use the following command.

```
./bin/ish Jaccard   1   ./dataset/webpage.dict   ./dataset/webpage.doc
```

3 **NGPP**: ./bin/ngpp ⟨Threshold⟩ ⟨lp⟩ ⟨Dictionary⟩ ⟨Document⟩

Parameters:

- ⟨Threshold⟩: Threshold of a similarity function.
- ⟨lp⟩: Prefix length
- ⟨Dictionary⟩ : The file path and name of the Dictionary.
- ⟨Document⟩ : The file path and name of the Document.

Example: To evaluate NGPP on the dblp dataset, using edit distance with threshold 1 and prefix length 7, one can use the following command.

```
./bin/ngpp 1 7 dataset/dblp.dict dataset/dblp.doc
```

How to Run

Run the following commands to get all experimental results.

- Unzip **faerie.tar.gz** to **faerie/**

  -
    ```
    tar -zxvf   faerie.tar.gz
    ```
- Go to the **faerie** directory

  -
    ```
    cd faerie
    ```
- Add execution privileges

  -
    ```
    chmod 777 ./bin/*
    ```

  -
    ```
    chmod 777 *.sh
    ```
- Get results of Figure 13: It took about 23 hours.

  -
    ```
    nohup   ./runFig13.sh   >   ./results/fig13.rst
    ```
- Get results of Figures 14&15: It took about 5 hours.

  -
    ```
    nohup ./runFig1415.sh   >   ./results/fig1415.rst
    ```
- Get results of Figure 16: It took about 7 hours.

  -
    ```
    nohup ./runFig16.sh > ./results/fig16.rst
    ```
- Get results of Figure 17: It took about 30 minutes.

  -
    ```
    nohup ./runFig17.sh > ./results/fig17.rst
    ```

Note that it will print out “FINISHED” when the program finishes. You can run the above four commands in parallel.

How to Redraw The Figures

Run the following command to redraw the figures.

- Go to the **faerie** directory
- Redraw Figure 13:
  - `./bin/getFigureData ./results/fig13.rst   13`
  - `cd ./figs`
  - `gnuplot fig13.plt`
- Redraw Figure 14 and Figure 15:
  - `./bin/getFigureData   ./results/fig1415.rst   1415`
  - `cd ./figs`
  - `gnuplot fig14.plt`
  - `gnuplot fig15.plt`
- Redraw Figure 16:
  - `./bin/getFigureData ./results/fig16.rst 16`
  - `cd ./figs`
  - `gnuplot   fig16.plt`
- Redraw Figure 17:
  - `./bin/getFigureData ./results/fig17.rst   17`
  - `cd ./figs`
  - `gnuplot   fig17.plt`
- Finally you can find all figures(.esp files) in **faerie/figs/** directory.