completion_trie

February 27, 2023

1 Complétion

Comparaion de plusieurs algorithmes pour implémenter un système de complétion.

[1]:
```
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

[1]:
```
<IPython.core.display.HTML object>
```

1.1 Tester des idées

1.1.1 Meilleur ordre pour a, ab, abc, abcd

[2]:
```
from mlstatpy.nlp.completion import CompletionTrieNode
import itertools
queries = ['a', 'ab', 'abc', 'abcd']
for per in itertools.permutations(queries):
    trie = CompletionTrieNode.build([(None, w) for w in per])
    gain = sum( len(w) - trie.min_keystroke(w)[0] for w in per)
    print(gain, "ordre", per)
```

```
0 ordre ('a', 'ab', 'abc', 'abcd')
1 ordre ('a', 'ab', 'abcd', 'abc')
1 ordre ('a', 'abc', 'ab', 'abcd')
2 ordre ('a', 'abc', 'abcd', 'ab')
2 ordre ('a', 'abcd', 'ab', 'abc')
2 ordre ('a', 'abcd', 'abc', 'ab')
1 ordre ('ab', 'a', 'abc', 'abcd')
2 ordre ('ab', 'a', 'abcd', 'abc')
2 ordre ('ab', 'abc', 'a', 'abcd')
3 ordre ('ab', 'abc', 'abcd', 'a')
3 ordre ('ab', 'abcd', 'a', 'abc')
3 ordre ('ab', 'abcd', 'abc', 'a')
2 ordre ('abc', 'a', 'ab', 'abcd')
3 ordre ('abc', 'a', 'abcd', 'ab')
2 ordre ('abc', 'ab', 'a', 'abcd')
3 ordre ('abc', 'ab', 'abcd', 'a')
4 ordre ('abc', 'abcd', 'a', 'ab')
4 ordre ('abc', 'abcd', 'ab', 'a')
3 ordre ('abcd', 'a', 'ab', 'abc')
3 ordre ('abcd', 'a', 'abc', 'ab')
```

```
3 ordre ('abcd', 'ab', 'a', 'abc')
3 ordre ('abcd', 'ab', 'abc', 'a')
4 ordre ('abcd', 'abc', 'a', 'ab')
4 ordre ('abcd', 'abc', 'ab', 'a')
```

1.1.2 Meilleur ordre pour a, ab, abc, abcd, edf, edfh

[3]:
```
queries = ['a', 'ab', 'abc', 'abcd', 'edf', 'edfh']
res = []
for per in itertools.permutations(queries):
    trie = CompletionTrieNode.build([(None, w) for w in per])
    gain = sum( len(w) - trie.min_keystroke(w)[0] for w in per)
    res.append((gain, "ordre", per))
res.sort(reverse=True)
for r in res[:30]:
    print(r)
```

```
(6, 'ordre', ('edfh', 'edf', 'abcd', 'abc', 'ab', 'a'))
(6, 'ordre', ('edfh', 'edf', 'abcd', 'abc', 'a', 'ab'))
(6, 'ordre', ('edfh', 'edf', 'abcd', 'ab', 'abc', 'a'))
(6, 'ordre', ('edfh', 'edf', 'abcd', 'ab', 'a', 'abc'))
(6, 'ordre', ('edfh', 'edf', 'abcd', 'a', 'abc', 'ab'))
(6, 'ordre', ('edfh', 'edf', 'abcd', 'a', 'ab', 'abc'))
(6, 'ordre', ('edfh', 'edf', 'abc', 'abcd', 'ab', 'a'))
(6, 'ordre', ('edfh', 'edf', 'abc', 'abcd', 'a', 'ab'))
(6, 'ordre', ('edf', 'edfh', 'abcd', 'abc', 'ab', 'a'))
(6, 'ordre', ('edf', 'edfh', 'abcd', 'abc', 'a', 'ab'))
(6, 'ordre', ('edf', 'edfh', 'abcd', 'ab', 'abc', 'a'))
(6, 'ordre', ('edf', 'edfh', 'abcd', 'ab', 'a', 'abc'))
(6, 'ordre', ('edf', 'edfh', 'abcd', 'a', 'abc', 'ab'))
(6, 'ordre', ('edf', 'edfh', 'abcd', 'a', 'ab', 'abc'))
(6, 'ordre', ('edf', 'edfh', 'abc', 'abcd', 'ab', 'a'))
(6, 'ordre', ('edf', 'edfh', 'abc', 'abcd', 'a', 'ab'))
(6, 'ordre', ('abcd', 'abc', 'edfh', 'edf', 'ab', 'a'))
(6, 'ordre', ('abcd', 'abc', 'edfh', 'edf', 'a', 'ab'))
(6, 'ordre', ('abcd', 'abc', 'edfh', 'ab', 'edf', 'a'))
(6, 'ordre', ('abcd', 'abc', 'edfh', 'ab', 'a', 'edf'))
(6, 'ordre', ('abcd', 'abc', 'edfh', 'a', 'edf', 'ab'))
(6, 'ordre', ('abcd', 'abc', 'edfh', 'a', 'ab', 'edf'))
(6, 'ordre', ('abcd', 'abc', 'edf', 'edfh', 'ab', 'a'))
(6, 'ordre', ('abcd', 'abc', 'edf', 'edfh', 'a', 'ab'))
(6, 'ordre', ('abcd', 'abc', 'edf', 'ab', 'edfh', 'a'))
(6, 'ordre', ('abcd', 'abc', 'edf', 'ab', 'a', 'edfh'))
(6, 'ordre', ('abcd', 'abc', 'edf', 'a', 'edfh', 'ab'))
(6, 'ordre', ('abcd', 'abc', 'edf', 'a', 'ab', 'edfh'))
(6, 'ordre', ('abcd', 'abc', 'ab', 'edfh', 'edf', 'a'))
(6, 'ordre', ('abcd', 'abc', 'ab', 'edfh', 'a', 'edf'))
```

1.1.3 Influence du poids

[4]:
```
queries = ['actuellement', 'actualité', 'acte', 'actes']
weights = [1, 1, 1, 2]
total = sum(weights) * 1.0 / len(queries)
```

```
res = []
for per in itertools.permutations(zip(queries, weights)):
    trie = CompletionTrieNode.build([(None, w) for w, p in per])
    wks = [(w, p, len(w)-trie.min_keystroke(w)[0]) for w, p in per]
    gain = sum( g*p/total for w, p, g in wks)
    res.append((gain, wks))
res.sort(reverse=True)
for r in res:
    print("{0:3.4} - {1}".format(r[0], " | ".join("%s p=%1.1f g=%1.1f" % _ for _ in
 ↪r[1])))
```

```
19.2 - actes p=2.0 g=4.0 | actuellement p=1.0 g=10.0 | acte p=1.0 g=1.0 |
actualité p=1.0 g=5.0
19.2 - actes p=2.0 g=4.0 | actualité p=1.0 g=7.0 | acte p=1.0 g=1.0 |
actuellement p=1.0 g=8.0
19.2 - actes p=2.0 g=4.0 | acte p=1.0 g=2.0 | actualité p=1.0 g=6.0 |
actuellement p=1.0 g=8.0
19.2 - actes p=2.0 g=4.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0 |
acte p=1.0 g=0.0
19.2 - actes p=2.0 g=4.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=9.0 |
acte p=1.0 g=0.0
19.2 - actes p=2.0 g=4.0 | acte p=1.0 g=2.0 | actuellement p=1.0 g=9.0 |
actualité p=1.0 g=5.0
18.4 - actuellement p=1.0 g=11.0 | actes p=2.0 g=3.0 | actualité p=1.0 g=6.0 |
acte p=1.0 g=0.0
18.4 - actuellement p=1.0 g=11.0 | actes p=2.0 g=3.0 | acte p=1.0 g=1.0 |
actualité p=1.0 g=5.0
18.4 - actualité p=1.0 g=8.0 | actes p=2.0 g=3.0 | actuellement p=1.0 g=9.0 |
acte p=1.0 g=0.0
18.4 - actualité p=1.0 g=8.0 | actes p=2.0 g=3.0 | acte p=1.0 g=1.0 |
actuellement p=1.0 g=8.0
18.4 - acte p=1.0 g=3.0 | actes p=2.0 g=3.0 | actuellement p=1.0 g=9.0 |
actualité p=1.0 g=5.0
18.4 - acte p=1.0 g=3.0 | actes p=2.0 g=3.0 | actualité p=1.0 g=6.0 |
actuellement p=1.0 g=8.0
17.6 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | actes p=2.0 g=2.0 |
acte p=1.0 g=0.0
17.6 - actuellement p=1.0 g=11.0 | acte p=1.0 g=2.0 | actes p=2.0 g=2.0 |
actualité p=1.0 g=5.0
17.6 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | actes p=2.0 g=2.0 |
acte p=1.0 g=0.0
17.6 - actualité p=1.0 g=8.0 | acte p=1.0 g=2.0 | actes p=2.0 g=2.0 |
actuellement p=1.0 g=8.0
17.6 - acte p=1.0 g=3.0 | actuellement p=1.0 g=10.0 | actes p=2.0 g=2.0 |
actualité p=1.0 g=5.0
17.6 - acte p=1.0 g=3.0 | actualité p=1.0 g=7.0 | actes p=2.0 g=2.0 |
actuellement p=1.0 g=8.0
16.8 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | acte p=1.0 g=1.0 |
actes p=2.0 g=1.0
16.8 - actuellement p=1.0 g=11.0 | acte p=1.0 g=2.0 | actualité p=1.0 g=6.0 |
actes p=2.0 g=1.0
16.8 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | acte p=1.0 g=1.0 |
actes p=2.0 g=1.0
16.8 - actualité p=1.0 g=8.0 | acte p=1.0 g=2.0 | actuellement p=1.0 g=9.0 |
```

```
actes p=2.0 g=1.0
16.8 - acte p=1.0 g=3.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0 |
actes p=2.0 g=1.0
16.8 - acte p=1.0 g=3.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=9.0 |
actes p=2.0 g=1.0
```

1.2 Nouvelle métrique

1.2.1 Intuition

[5]:
```
def gain_moyen_par_mot(queries, weights):
    total = sum(weights) * 1.0
    res = []
    for per in itertools.permutations(zip(queries, weights)):
        trie = CompletionTrieNode.build([(None, w) for w, p in per])
        wks = [(w, p, len(w)-trie.min_keystroke(w)[0]) for w, p in per]
        gain = sum( g*p/total for w, p, g in wks)
        res.append((gain, wks))
    res.sort(reverse=True)
    for i, r in enumerate(res):
        print("{0:3.4} - {1}".format(r[0], " | ".join("%s p=%1.1f g=%1.1f" % _ for _␣
 ↪in r[1])))
        if i > 10:
            print("...")
            break
```

[6]:
```
queries = ['actuellement', 'actualité', 'actuel']
weights = [1, 1, 1]
gain_moyen_par_mot(queries, weights)
```

```
7.0 - actuellement p=1.0 g=11.0 | actuel p=1.0 g=4.0 | actualité p=1.0 g=6.0
7.0 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | actuel p=1.0 g=3.0
7.0 - actuel p=1.0 g=5.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0
7.0 - actuel p=1.0 g=5.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=9.0
7.0 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | actuel p=1.0 g=3.0
7.0 - actualité p=1.0 g=8.0 | actuel p=1.0 g=4.0 | actuellement p=1.0 g=9.0
```

[7]:
```
queries = ['actuellement', 'actualité', 'actuel']
weights = [1, 1, 0]
gain_moyen_par_mot(queries, weights)
```

```
9.0 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | actuel p=0.0 g=3.0
9.0 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | actuel p=0.0 g=3.0
8.5 - actuellement p=1.0 g=11.0 | actuel p=0.0 g=4.0 | actualité p=1.0 g=6.0
8.5 - actualité p=1.0 g=8.0 | actuel p=0.0 g=4.0 | actuellement p=1.0 g=9.0
8.0 - actuel p=0.0 g=5.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0
8.0 - actuel p=0.0 g=5.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=9.0
```

[8]:
```
queries = ['actuellement', 'actualité']
weights = [1, 1]
gain_moyen_par_mot(queries, weights)
```

```
9.0 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0
9.0 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0
```

1.3 Vérification

[9]:
```
def gain_dynamique_moyen_par_mot(queries, weights, permutation=True):
    total = sum(weights) * 1.0
    res = []
    for per in itertools.permutations(zip(queries, weights)):
        trie = CompletionTrieNode.build([(None, w) for w, p in per])
        trie.precompute_stat()
        trie.update_stat_dynamic()
        wks = [(w, p, len(w)-trie.min_dynamic_keystroke(w)[0]) for w, p in per]
        gain = sum( g*p/total for w, p, g in wks)
        res.append((gain, wks))
        if not permutation:
            break
    res.sort(reverse=True)
    for i, r in enumerate(res):
        print("{0:3.4} - {1}".format(r[0], " | ".join("%s p=%1.1f g=%1.1f" % _ for _␣
 ↪in r[1])))
        if i > 10:
            print("...")
            break
```

Pas de changement :

[10]:
```
queries = ['actuellement', 'actualité', 'actuel']
weights = [1, 1, 0]
gain_dynamique_moyen_par_mot(queries, weights)
```

```
9.0 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | actuel p=0.0 g=3.0
9.0 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | actuel p=0.0 g=3.0
8.5 - actuellement p=1.0 g=11.0 | actuel p=0.0 g=4.0 | actualité p=1.0 g=6.0
8.5 - actuel p=0.0 g=5.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=10.0
8.5 - actualité p=1.0 g=8.0 | actuel p=0.0 g=4.0 | actuellement p=1.0 g=9.0
8.0 - actuel p=0.0 g=5.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0
```

Changements :

[11]:
```
queries = ['actuellement', 'actualité', 'actuel']
weights = [1, 1, 1]
gain_dynamique_moyen_par_mot(queries, weights)
```

```
7.333 - actuel p=1.0 g=5.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=10.0
7.0 - actuellement p=1.0 g=11.0 | actuel p=1.0 g=4.0 | actualité p=1.0 g=6.0
7.0 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | actuel p=1.0 g=3.0
7.0 - actuel p=1.0 g=5.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0
7.0 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | actuel p=1.0 g=3.0
7.0 - actualité p=1.0 g=8.0 | actuel p=1.0 g=4.0 | actuellement p=1.0 g=9.0
```

[12]:
```
gain_moyen_par_mot(queries, weights)
```

```
7.0 - actuellement p=1.0 g=11.0 | actuel p=1.0 g=4.0 | actualité p=1.0 g=6.0
7.0 - actuellement p=1.0 g=11.0 | actualité p=1.0 g=7.0 | actuel p=1.0 g=3.0
7.0 - actuel p=1.0 g=5.0 | actuellement p=1.0 g=10.0 | actualité p=1.0 g=6.0
7.0 - actuel p=1.0 g=5.0 | actualité p=1.0 g=7.0 | actuellement p=1.0 g=9.0
7.0 - actualité p=1.0 g=8.0 | actuellement p=1.0 g=10.0 | actuel p=1.0 g=3.0
7.0 - actualité p=1.0 g=8.0 | actuel p=1.0 g=4.0 | actuellement p=1.0 g=9.0
```

1.4 Ajouter une complétion

[13]:
```
queries = ['macérer', 'maline', 'machinerie', 'machinerie infernale', 'machinerie␣
 ↪infernalissime',
           'machine artistique', 'machine automatique',
           'machine chaplin', 'machine intelligente', 'machine learning']
weights = [1] * len(queries)
gain_dynamique_moyen_par_mot(queries, weights, permutation=False)
```

```
10.1 - macérer p=1.0 g=6.0 | maline p=1.0 g=4.0 | machinerie p=1.0 g=7.0 |
machinerie infernale p=1.0 g=16.0 | machinerie infernalissime p=1.0 g=20.0 |
machine artistique p=1.0 g=12.0 | machine automatique p=1.0 g=12.0 | machine
chaplin p=1.0 g=7.0 | machine intelligente p=1.0 g=11.0 | machine learning p=1.0
g=6.0
```

[14]:
```
queries = ['machine'] + queries
weights = [1] * len(queries)
weights[queries.index('machine')] = 0.0
gain_dynamique_moyen_par_mot(queries, weights, permutation=False)
```

```
12.3 - machine p=0.0 g=6.0 | macérer p=1.0 g=5.0 | maline p=1.0 g=3.0 |
machinerie p=1.0 g=8.0 | machinerie infernale p=1.0 g=17.0 | machinerie
infernalissime p=1.0 g=21.0 | machine artistique p=1.0 g=15.0 | machine
automatique p=1.0 g=15.0 | machine chaplin p=1.0 g=11.0 | machine intelligente
p=1.0 g=16.0 | machine learning p=1.0 g=12.0
```

1.5 Wikipedia

- PageCount
- dump

[15]:
```

```