Further Enhancement to the Porter's Stemming Algorithm

Fadi Yamout[1], Rana Demachkieh[1], Ghalia Hamdan[1], Reem Sabra[1]
[1] Faculty of Computer Sciences
C&E American University I., Beirut, Lebanon
Email: fyamout@inco.com.lb

Abstract. Stemming algorithms are used to transform the words in texts into their grammatical root form, and are mainly used to improve the Information Retrieval System's efficiency. Several algorithms exist with different techniques. The most widely used is the Porter Stemming algorithm. However, it still has several drawbacks, although many attempts were made to improve its structure. This paper reveals the inaccuracies encountered during the stemming process and proposes the corresponding solutions.

1. Introduction

Finding information is not the only activity that exists in an Information Retrieval (IR) system. Indexing, for instance, refers to how information and user's requests from the system are represented. We will refer to the information to be indexed as documents. Hence, documents are represented through a set of index terms or keywords. The terms are extracted from the text of the documents. This might be done automatically or generated by a specialist.

It was estimated in Kowalski [1] that for relatively short documents (e.g., 300-500 words) it normally takes a specialist at least five minutes per item to produce the terms, while it takes just a few seconds on a moderate computer. The extracted terms are mainly nouns since they describe better the semantic of the documents while adjectives, adverbs, and connectives (including transitions, conjunctions…) are less useful because they work mainly as complements.

These irrelevant terms are usually placed in a file called Stoplist. A Stoplist algorithm is applied to all the documents in the collection with an objective to eliminate the terms that have little value to the system. In addition, a word, which occurs in 80% of the documents in the collection, is useless [2]. An example of 425 stopwords is shown in a list in Frakes and Baeza-Yates [2]. The remaining terms are stemmed using Porter's algorithm [3], which brings down distinct words to their grammatical root and thus reduces further the number of unique terms.

Many attempts were made to improve the structure of the Porter algorithm [4], however, it still has several drawbacks. In this paper, further improvements are

introduced to overcome these problems in order to enhance the stemming process. We will refer to the existing Porter algorithm as Porter 2002 and the new as Porter 2004.

2. Porter's Algorithm

Porter Stemming Algorithm was developed by Martin Porter at the University of Cambridge in 1980 and was first published in Porter, M.F., [5] and reprinted in Sparck, Karen, and Peter [6]. As described in the publication, "*The Porter stemming algorithm (or 'Porter stemmer') is a process for removing the commoner morphological and inflexional endings from words in English. Its main use is as part of a term normalization process that is usually done when setting up Information Retrieval systems*". Since then it has been very widely used and coded in various programming languages. It is based mainly on stemming operations that remove suffixes from words, such as gerunds (motoring → motor), plurals (cats → cat), and replacing words ending with "ator" for example with "ate" (operator → oper), etc….

These operations are classified into rules where each of these rules deals with a specific suffix and having certain condition(s) to satisfy. A given word's suffix is checked against each rule in a sequential manner until it matches one, and consequently the conditions in the rule are tested on the stem that may result in a suffix removal or modification.

3. Drawbacks of the Porter Algorithm

Natural languages are not completely regular constructs, and therefore stemmers operating on natural words unavoidably make mistakes. For instance, words, which are distinct, may be wrongly conflated to give similar stems (ex: design → design; designate → design, etc...) and affect seriously the retrieval performance of an IR system since the semantic of the word is expressed differently; these are known as over-stemming errors. On the other hand, words which ought to be merged together may remain distinct after stemming (ex: characterizes → character; characteristic → characterist, etc…); these are known as under-stemming errors and do not affect the retrieval performance of an IR system [2]. In this paper we deal with over-stemming errors.

The modified Porter algorithm was tested on 23,531[1] words provided by Porter and compared to an already existing output provided from the same site. In addition, it was tested on 45,000 words extracted from the Oxford's dictionary, and the following over-stemming errors were observed:

[1] http://www.tartarus.org/~martin/PorterStemmer/index.html

Error #1:
The non-existence of "e" at the end of the words that have m=1 and begin with a consonant, and end with two consonants; for ex: paste, loathe…:

Paste → past
Past → past

Error #2:
The removal of "s" in step1 from words ending with "is" such as his and appendicitis:

Appendicitis → append
Append → append

Error #3:
Words ending with "yed" and "ying" and having different meanings may end up with the same stem:

Dying → dy (impregnate with dye)
Dyed → dy (passes away)

Error #4:
The removal of "ic" or "ical" from words having m=2 and ending with a series of consonant, vowel, consonant, vowel, such as generic, politic…:

Political → polit
Politic → polit
Polite → polit

Error #5:
The removal of the suffix "ative" from all words ending with it and having m=1 or m=2, the thing that leads to serious conflicts:

Combative → comb Generative → gener
Comb → comb General → gener

Error #6:
The removal of the suffix "ness" from all words where m=1 and end with consonant, vowel, consonant (cvc) such as witness:

Witness → wit
Wit → wit

Error #7:
The suffix "al" is removed from all words where m=2 e.g. admiral, animal…:

Admiral → admir
Admire → admir

Error #8:
The elimination of the suffix "eer" from words with m=2 such as engineer:

Engineer → engin
Engine → engin

Error #9:
The exclusion of the suffix “ible” from all words where m=2 starting by a consonant and not ending with a series of consonant, vowel, consonant, vowel, such as responsible:

Responsible → respons
Response → respons

Error #10:
The exclusion of the suffix “ance” from words with m=2 ending with a series of consonant, vowel, consonant, vowel:

Severance → sever
Several → sever

Error #11:
The removal of the suffix “ment” from all words even those ending with “iment” having m=2 and not ending with a series of consonant, vowel, consonant, vowel; e.g. experiment:

Experiment → experi
Experience → experi

Error #12:
The elimination of “ion” from all words where m=2 and not consonant, vowel, consonant, vowel, without replacement:

Secretion → secret
Secret → secret

Error #13:
The removal of the suffix “ate” or “nate” from all words where m=2 and ending with a series of consonant, vowel, consonant, vowel:

Designate → design
Design → design

Error #14:
The elimination of the suffix “ize” from all words having m=2 and starting by a consonant, and ending with a series of consonant, vowel, consonant, vowel:

Colonize → colon
Colon → colon

Error #15:
The exclusion of “itive” from words with m= 1 and starting by consonant, and ending with a series of consonant, vowel, consonant, vowel:

Positive → posit
Position → posit

Error #16:
The removal of “iti” from all words where m=2 starting by a vowel and ending with a series of consonant, vowel, consonant, vowel:

Ameniti → amen
Amen → amen

The removal of “iti” from all words where m=3 starting by a vowel and not ending with a series of consonant, vowel, consonant, vowel:

Universiti → univers
Universe → univers

4. Modifications

The following section describes the corresponding solutions for each of the errors revealed previously (Table 1 describes the symbols used).

k	: Pointer to the last letter in the word
m()	: Counts how many consecutive vowel, consonant exist in a word
cons()	: Checks whether the letter at a certain position is a consonant or not
ends()	: Determines if the word ends with the variable sent and consequently truncates this variable from the original word

Table1: Symbol’s Intuitions

Solution #1:

To solve the problem ending with “e” a function is created to keep the “e” at the end of the word by returning false
If m=1:
Starts with a consonant and ends with two consonants
Paste, loathe, and bottle.

While adding this method, another problem arises for the words such as beaches, bushes…, so an additional statement is added to step1: If the word ends with “ches” or with “shes” the program will remove the “es" since in step6 the cvd method is used.

Beaches → beach
Bushes → bush

The cvd method is as follow:

```
function cvd (int d)
  if cons(d) then
    d := d-1;
    if !cons(d) and d!=0 then
       while !cons(d) and d>0 do
          d := d-1;
       if cons(d) then return true;
    return false;
  return true;
```

Step1:

```
  if ends("ches") or ends("shes") then   k := k-2;
```

Solution #2:

If the word ends with “is”, the “s” is not deleted

Appendicitis → appendicitis

The statement is:

```
if ends("is")
```

Solution #3:

To prevent words ending with “ying” and “yed”, and having different meanings, from producing the same stem, the “ying” will be set to “i” if it has m=0, starting with consonant and vowel.

Dying → di;

Dyed → dy;

The statements are:

```
if ends("ying) then
  if m()=0 and cons(0) and !cons(1) then setto("i");
```

Solution #4:

Usually the words that end by “ic” in step3 or “ical” in step4 must be removed but in other cases it must not. Therefore, if the word is of size m = 2 and consists of a series of consonant, vowel, consonant, vowel, it is replaced by “ica*” rather than being removed, then in step5 it is transformed to “ic”.

polite→ polit,

political→ politic,

political→ politic

The statements are:

Step3:

```
case 'i':  if ends("ic") and m()=2 then
             while k > 0 do
               if cons(k) and !cons(k-1) then k  :=  k – 2;
               else j  :=  j + 2; k  :=  j; break;
             if k <= 0 then r("ica*");
             break;
           else break;
```

Step4:

```
case 'l':  if ends("ical") then
             if m() = 2 then
               while k > 0 do
                 if cons(k) and !cons(k-1) then k  :=  k – 2;
                 else k  :=  j + 2; r("ic"); break;
             if  k <= 0 then r("ica*"); break ;
             else r("ic"); break;
```

Step5:

```
if ends ("ica*") then r ("ic"); j  :=  j + 2; break;
else j  :=  k; break;
```

Solution #5:

If the word ends by “ative” and m = 2, it is replaced by “ate”.

Generative→ generate

Or if it is m > 2 it is removed.

Authoritative→ authorit

Or if m = 1 it is replaced by “at”.

Combative→ combat

The statements are:

```
if ends("ative") then
  if  m() = 2 then r("ate");
  else if  m() =1 then r("at");
  else if  m() > 2  then r("");
```

Solution #6:

If the word ends with “ness”, m = 1, and ends with consonant, vowel, and a consonant, it is kept as it is.

Witness→witness

Else it will be removed.

The statements are:

```
case 's': if ends("ness") then
            if m() == 1 and cvc(k-4) then break;
            else r("");
            break;
          break;
```

Solution #7:

If it ends by “iral” and m = 2 it is left as it is.

Admiral→ admiral.

Or if it ends by “al”, m = 2, and it consists of a series of consonant, vowel, consonant, vowel, it is removed

General→ gener

Admiral→ admiral

Else if m >1 it is removed

The statements are:

```
case 'a': if  ends("al") then
            if  m() = 2 then
              if ends ("iral") then j  :=  j + 4; break;
              p  :=  p – 2;
              while ( p > 0 )  do
                 if cons (p) and !cons(p-1) then p  :=  p – 2;
                 else k  :=  j; break;
              if  p <= 0 then j  :=  j + 2;
            else if  m() > 1  then k  :=  j; break;
```

Solution #8:

If it ends with “eer” and m = 2, then only the “r” is removed in step4 and consequently the last “e” is removed in step6

Engineer→ engine

The statements are:

```
case 'e': if ends("er") then
            if m()=2 and ends ("eer")  then j  :=  j + 2; break;
            else break;
          return
```

Solution #9:
If it ends with “ible”, m = 2, and starts with a consonant and not ending with a series of consonant vowel consonant vowel, then it is kept as it is.

Responsible → responsible.

Reducible → reduc

Or if m> 1 it is removed.

Reprehensible → reprehens

The statements are:

```
if ends("ible") then
   if m()=2 and cons(0) then
      p  :=  p – 4;
      while  p > 0 do
         if cons (p) and !cons (p-1) then p  :=  p – 2;
         else j  :=  j + 3; break;
      if  p <= 0  then k  :=  j; break;
      else k  :=  j;
   else if   m() > 1 then k  :=  j; break;
```

Solution #10:
If it ends with “ance”, m = 2, and consist of a series of consonant, vowel, consonant, vowel, therefore, it is replaced by “e”.

Severance→severe,

If not, it is removed.

Importance→import

The statements are:

```
case 'c': if ends ("ance") then
                if  m() = 2  then
                   p  :=  p – 4;
                   while  p > 0 do
                      if cons (p) and !cons(p-1) then p  :=  p – 2;
                      else k  :=  j; break;
                   if  p <= 0 and cons(0) then b[j := j+1]='e'; k  :=  j; break;
                   else  k  :=  j; break;
                if  m() > 1 then k  :=  j; break;
```

Solution #11:
If it ends with “iment”, m = 2, and not ending with a series of consonant vowel consonant vowel, therefore, it is left as it is.

Experiment→ experiment

Or if m> 1 it is removed.

Accompaniment→ accompani

The statements are:

```
if ends("iment") and m() = 2 then
   p  :=  p - 5
   while  p > 0 do
      if cons(p) and !cons(p-1) then p  :=  p – 2;
      else break;
   if p>0 then j  :=  j + 5; break;
if ends ("ement") then break;
if ends ("ment") then break;
```

Solution #12:

If it ends with "tion", m = 2, and not ending with a series of consonant vowel consonant vowel…, it is replaced with an "e".

Secretion→secrete

Sedition→sedit

Or if m> 1 it is removed. The statements are:

```
if ends("ion") and j >= 0 then
   if b[j] = 't' then
      if m()= 2 then
      p  :=  p – 3;
      while  p > 0 do
         if cons (p) and !cons (p-1) then  p  :=  p – 2;
         else b[j := j+1]  :=  'e'; k  :=  j; break;
      if p <= 0 then k  :=  j; break;
   else if  m() > 1 then k  :=  j; break;
```

Solution #13:

If it ends with "nate" or "ate", m = 2, and ends with a series of consonant vowel consonant vowel…, it is not replaced.

Designate→designate

Or if m> 1, then it is removed.

Collaborate→ collabor

Or if m = 1, then the "at" is kept

Situate→ situat

Or if m = 0, then it is left as it is.

Ate→ate

The statements are:

```
case 't': if ends("nate") and m() = 2 then
            p := p - 4
            while p > 0 do
              if cons (p) and !cons (p-1) p := p – 2;
              else k := j + 1; break;
            if p <= 0 and cons(0) then j := j + 4; break;
          else if ends("ate") then
            if m() = 2 then
              p := p - 3
              while p > 0 do
                if cons (p) and !cons(p-1) then p := p – 2;
                else k := j; break;
              if p<=0 and cons(0) then j := j + 3; break;
              else   k := j; break;
            else if m() > 1 then k := j; break;
            else if m() = 1 then j := j + 2; k := j; break;
            else j := j + 3; break;
```

Solution #14:

If it ends with “ize”, m = 2, and starts with a consonant, and ends with a series of consonant, vowel, consonant, vowel…, it is kept as it is:

Colonize→colonize

Or if m> 1 it is removed.

Aerosolize→ aerosol

The statements are:

```
case 'z': if ends("ize") then
            if m() = 2 then
              p := p – 3;
              while p > 0 do
                if cons(p) and !cons(p-1) then p := p – 2;
                else k := j; break;
              if p <= 0 and cons(0) then
                j := j + 3; break;
              else k := j; break;
            else if m() > 1 then k := j; break;
```

Solution #15:

If it ends with “itive”, m = 1, starts with a consonant, and ends with a series of consonant, vowel, consonant, vowel…, it is kept as it is:

Positive→ positive

Or if m> 1 it is removed.

Acquisitive→ acquisit

The statements are:

```
case 'v': if ends("itive") and m() = 1 and cons(0) Then
              p  :=  p – 5;
              while p > 0 do
                 if cons(p) and !cons(p-1) then p  :=  p – 2;
                 else j  :=  j + 2; k  :=  j; break;
              if p <= 0 and cons(0) then j  :=  j + 5; break;
              else    k  :=  j; break;
           else if ends("ive") break;
return;
```

Solution #16:

If it ends with "iti", m = 2, starts with a vowel, and ends with a series of consonant, vowel, consonant, vowel…, it is kept as it is:

amenity→ ameniti

If it ends with "iti", m = 3, starts with a vowel, and ends with a series of consonant, vowel, consonant, vowel…, it is kept as it is:

Universiti→ universiti

Or if m> 1 it is removed

Minority→ minor

The statements are:

```
if ends ("iti")  then
   if  m() = 2 then
      p  :=  p – 3;
      while  p > 0 do
         if cons (p) and !cons(p-1) then p  :=  p – 2;
         else k  :=  j; break;
      if  p <= 0 and !cons(0)  then j  :=  j + 3; break;
      else k  :=  j; break;
   else if m() = 3 and !cons(0)  then
      p  :=  p - 3
      while  p > 0  do
         if  cons(p)  and !cons (p-1) then p  :=  p – 2;
         else j  :=  j + 3; break;
      if  p <= 0 and !cons(0) then k  :=  j; break;
      else k  :=  j; break;
   else if  m() > 1 then k  :=  j; break;
```

Exceptions:

Some of the words are considered as exception to the previously described rules, and therefore are treated separately. The following step contains the words that must keep their "e" while removing the "ing" or the "ed".

Loathing → loathe

Pasted → paste

```
function step0()
      String s1=new String(b);
      String s2=new String("rang secret loath past us butt");
      if s2.regionMatches(s2.indexOf(b[0]),s1,0,j+1) then
         return true;
      else return false;
```

5. Experiments

The previously described solutions produce different results than the existing Porter algorithm. Outputs from both Porter 2002 and 2004 are put alongside in Appendix to demonstrate the dissimilarities.

The two techniques were tested against CISI [7], which is a standards test collection that contains 1460 documents, in an attempt to move more relevant documents (the ones found in the queries' relevance judgments) further up the ranking. The result showed a slight improvement (1.5%) in precision and recall, however for some queries the improvement was 2.5%. The percentage is computed as an average for the precision and recall produced by the 30 queries that come with the collection. The results are illustrated in Figure 1 using the 11-point average curve.

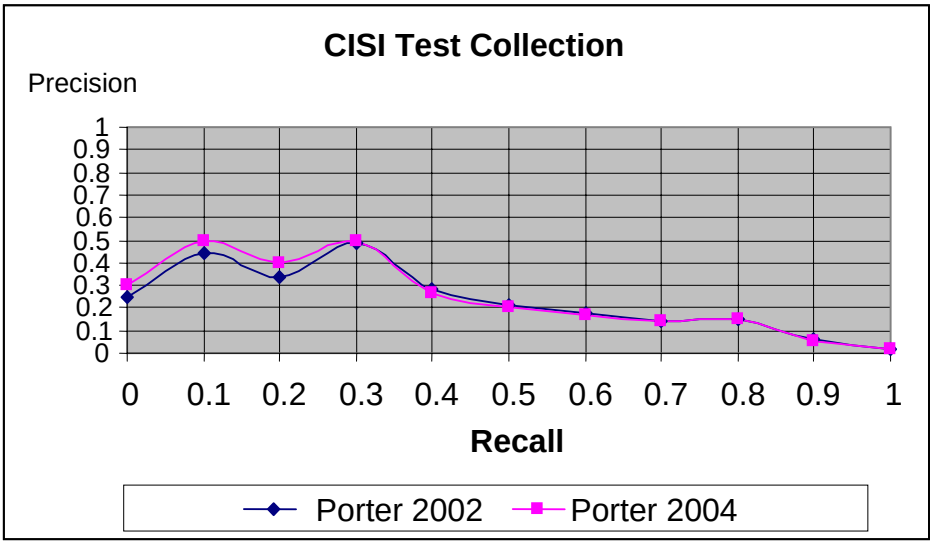

Figure 1: 11 point Average Curve

References

1. Kowalski G. (1997) “Information Retrieval Systems: Theory and Implementation”, Kluwer Academic Publisher, 1997. id387
2. Baeza-Yates R. and Ribeiro-Neto B. (1999) “Modern Information Retrieval”. New York: Addison Wesley
3. Frakes W. B. and Baeza-Yates R. (1992) “Information Retrieval: Data Structures and Algorithms”. Englewood Cliffs, NJ: Prentice-Hall. id175
4. Porter, M.F., (2002) “Developing the English Stemmer”, http://snowball.tartarus.org/.
5. Porter, M.F., (1980), “An Algorithm for Suffix Stripping”, Program, 14(3) :130-137.
6. Sparck Jones, Karen, and Peter Willet, (1997), “Readings in Information Retrieval”, San Francisco: Morgan Kaufmann, ISBN 1-55860-454-4.
7. CISI-collection. The CISI reference collection for information retrieval. 1460 documents and 30 queries, http://local.dcs.gla.ac.uk/idom/ir_resources/test-collections/cisi/, 1981

Appendix: Dissimilarities between existing and new algorithm

Word	Porter02	Porter04
abl	abl	abl
able	abl	able
ach	ach	ach
ached	ach	ach
aches	ach	ach
aching	ach	ach
ache	ach	ache
ad	ad	ad
add	add	add
added	ad	add
adding	ad	add
adds	add	add
abl	abl	abl
able	abl	able
ach	ach	ach
ached	ach	ach
aches	ach	ach
aching	ach	ach
ache	ach	ache
ad	ad	ad
add	add	add
added	ad	add
adding	ad	add
adds	add	add
admirable	admir	admir
admirably	admir	admir
admiration	admir	admir
admire	admir	admir
admired	admir	admir
admirer	admir	admir
admirers	admir	admir
admires	admir	admir
admiring	admir	admir
admiringly	admir	admir
admiral	admir	admiral
amen	amen	amen
amenable	amen	amen
amenities	(none)	amen
amenity	(none)	amen
and	and	and
ande	and	ande
andes	andes	ande
animate	anim	anim
animated	anim	anim
animates	anim	anim
animating	anim	anim
animation	anim	anim
animal	anim	animal
animalized	anim	animal
animals	anim	animal
Ann	ann	ann
anne	ann	anne

Word	Porter02	Porter04
bath	bath	bath
bathed	bath	bath
bathing	bath	bath
baths	bath	bath
bathe	bath	bathe
bathes	bath	bathe
bell	bell	bell
belled	bell	bell
belling	bell	bell
bells	bell	bell
belle	bell	belle
bonn	bonn	bonn
bonne	bonn	bonne
born	born	born
borne	born	borne
brown	brown	brown
browning	brown	brown
browns	brown	brown
browne	brown	browne
bush	bush	bush
bushes	bush	bush
bushe	bush	bushe
call	call	call
called	call	call
calling	call	call
calls	call	call
calle	call	calle
cloth	cloth	cloth
clothed	cloth	cloth
clothing	cloth	cloth
cloths	cloth	cloth
clothe	cloth	clothe
clothes	cloth	clothe
cross	cross	cross
crossed	cross	cross
crosses	cross	cross
crossing	cross	cross
crosse	cross	crosse
dank	dank	dank
danke	dank	danke
design	design	design
designed	design	design
designer	design	design
designing	design	design
designs	design	design
designates	design	designat
designation	design	designat
ear	ear	ear
eared	ear	ear
earings	ear	earing
ell	ell	ell
elle	ell	elle

Word	Porter02	Porter04
elles	ell	elle
engine	engin	engin
engines	engin	engin
engineer	engin	engine
engineering	engin	engine
even	even	even
evening	even	even
evenly	even	even
evenness	even	even
evenings	even	evening
fill	fill	fill
filled	fill	fill
filling	fill	fill
fills	fill	fill
fille	fill	fille
fort	fort	fort
forts	fort	fort
forte	fort	forte
forty	forti	forti
fortis	forti	fortis
front	front	front
fronted	front	front
fronting	front	front
fronts	front	front
fronte	front	fronte
funeral	funer	funeral
funerals	funer	funeral
funereal	funer	funere
futur	futur	futur
future	futur	future
futures	futur	future
gang	gang	gang
ganging	gang	gang
gangs	gang	gang
ganges	gang	gange
generous	generous	gener
generousness	(none)	gener
generously	generous	gener
general	general	general
generalities	general	general
generality	general	general
generalization	general	general
generally	general	general
generality	(none)	general
generalizations	(none)	general
generalize	(none)	general
generalized	(none)	general
generalizer	(none)	general
generalizers	(none)	general
generalizes	(none)	general
generalizing	(none)	general
generals	general	general

Word	Porter02	Porter04
generate	generat	generat
generated	generat	generat
generation	generat	generat
generates	(none)	generat
generating	(none)	generat
generative	(none)	generat
generator	(none)	generat
generators	(none)	generat
generations	generat	generat
generic	generic	generic
generically	(none)	generic
goeth	Goeth	goeth
goethe	goeth	goethe
grande	grand	grande
grandee	grande	grande
grandees	grande	grande
hand	hand	hand
handed	hand	hand
handful	hand	hand
handfuls	hand	hand
handing	hand	hand
hands	hand	hand
hande	hand	hande
hast	hast	hast
haste	hast	haste
her	her	her
hers	her	her
herrings	her	herring
hing	hing	hing
hinges	hing	hinge
host	host	host
hosts	host	host
hoste	host	hoste
however	howev	howev
howeve	howev	howeve
iron	iron	iron
ironed	iron	iron
ironing	iron	iron
irons	iron	iron
irony	ironi	ironi
ironical	iron	ironic
ironically	iron	ironic
later	later	later
lateral	later	lateral
laterally	later	lateral
loath	loath	loath
loathe	loath	loathe
loathed	loath	loathe
loathing	loath	loathe
lungs	lung	lung
lunge	lung	lunge
missy	missi	missi
missis	missi	missis
mond	mond	mond
monde	mond	monde

Word	Porter02	Porter04
mont	mont	mont
monte	mont	monte
montes	mont	monte
numerous	numer	numer
numerical	numer	numeric
of	of	of
off	off	off
offing	of	off
offe	off	offe
past	past	past
pasted	past	paste
moral	moral	moral
morality	moral	moral
moralities	moral	moral
morale	moral	morale
morally	moral	moral
morals	moral	moral
petulance	petul	petule
petulant	petul	petul
petulantly	petul	petul
petulance	petul	petule
petulant	petul	petul
petulantly	petul	petul
picture	pictur	picture
pictured	pictur	pictur
pictures	pictur	picture
picturing	pictur	pictur
pierce	pierc	pierce
pierced	pierc	pierc
pierces	pierc	pierce
piercing	pierc	pierc
piercingly	pierc	pierc
position	posit	posit
positions	posit	posit
positive	posit	positiv
positively	posit	positiv
positiveness	posit	positiv
private	privat	privat
privateer	privat	private
privately	privat	privat
privation	privat	privat
privations	privat	privat
proceed	proceed	proce
proceeds	proce	proceed
rang	rang	rang
range	rang	range
ranged	rang	range
rangees	range	range
ranges	rang	range
ranging	rang	range
regal	regal	regal
regale	regal	regale
regaled	regal	regal
regaling	regal	regal
response	respons	respons

Word	Porter02	Porter04
responsibilities	respons	responsibl
responsibility	respons	responsibl
responsible	respons	responsibl
responsive	respons	respons
roll	roll	roll
rolle	roll	rolle
rolled	roll	roll
rolling	roll	roll
rollings	roll	rolling
rolls	roll	roll
round	round	round
rounde	round	rounde
rounded	round	round
rounding	round	round
roundly	round	round
roundness	round	round
rounds	round	round
relax	relax	relax
relaxe	relax	relaxe
relaxes	relax	relaxe
remain	remain	remain
remaine	remain	remaine
singeing	sing	singe
singing	sing	sing
sings	sing	sing
scienc	scienc	scienc
science	scienc	science
sciences	scienc	science
secrete	secret	secrete
secreted	secret	secrete
secretes	secret	secrete
secreting	secret	secrete
secretion	secret	secrete
secretly	secret	secret
secrets	secret	secret
sever	sever	sever
severa	severa	severa
several	sever	several
severally	sever	several
severe	sever	severe
severed	sever	sever
severely	sever	severe
severer	sever	sever
severity	sever	sever
sooth	sooth	sooth
soothe	sooth	soothe
soothed	sooth	sooth
soothing	sooth	sooth
soothingly	sooth	sooth
start	start	start
starte	start	starte
started	start	start
starting	start	start
startings	start	starting
starts	start	start

Word	Porter02	Porter04
stern	stern	stern
sterne	stern	sterne
sternly	stern	stern

Word	Porter02	Porter04
sternness	stern	stern
wit	wit	wit
witness	wit	witness
witnessed	wit	witness

Word	Porter02	Porter04
witnesses	wit	witness
witnessing	wit	witness
wits	wit	wit
witted	wit	wit