Item 56
git_comments:

1. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
2. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
3. * A list of words used by Kstem
4. KStemData1 ... KStemData8 are created from "head_word_list.txt"
5. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
6. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
7. * A list of words used by Kstem
8. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
9. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names

"Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

10. * A list of words used by Kstem
11. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
12. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
13. * A list of words used by Kstem
14. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
15. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY

OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

16. * A list of words used by Kstem
17. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
18. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
19. * A list of words used by Kstem
20. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
21. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
22. * A list of words used by Kstem
23. This is a java version of Bob Krovetz' KStem. * * Java version by Sergio Guzman-Lara. * CIIR-UMass Amherst http://ciir.cs.umass.edu
24. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain

permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

25. * A list of words used by Kstem
26. * Returns the next, stemmed, input Token. * @return The stemed form of a token. * @throws IOException
27. * * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
28. * A high-performance kstem filter for english. * * See * "Viewing Morphology as an Inference Process" * (Krovetz, R., Proceedings of the Sixteenth Annual International ACM SIGIR * Conference on Research and Development in Information Retrieval, 191-203, 1993). * * All terms must already be lowercased for this filter to work correctly.
29. index of final letter in stem (within word)
30. I don't know any long words that this applies to,
31. ** * private void initializeStemHash() { if (maxCacheSize > 0) cache = new * CharArrayMap<String> (maxCacheSize,false); } **
32. the default is to convert -ical to -ic
33. YCS: extra lookup()s were inserted so we don't need to do an extra wordInDict() here.
34. * * Title: Kstemmer * Description: This is a java version of Bob Krovetz' kstem stemmer * Copyright: Copyright 2008, Luicid Imagination, Inc. * Copyright: Copyright 2003, CIIR University of Massachusetts Amherst (http://ciir.cs.umass.edu)
35. try just removing the "s"
36. restore the word to the way it was
37. * this routine deals with -al endings. Some of the endings from the previous * routine are finished up here.
38. try removing -ity and adding -e
39. allocate enough space so that an expansion is never needed
40. nolookup() ... converting ical to ic means removing "al" which we already tried ERROR
41. handle -able and -ible
42. handle -ency and -ancy
43. try just removing the ending
44. YCS
45. * the -ability and -ibility endings are highly productive, so just accept * them
46. nolookup()
47. `bookselling' -> `booksell' and `mislabelling' -> `mislabell').
48. * this is a very productive ending, so just * accept it
49. if (endsIn(ion)) { we checked for this earlier... just need to set "j" YCS
50. we almost always want to convert -ied to -y, but
51. the default is to leave the consonant doubled
52. handle -ism
53. ensure calories -> calorie
54. no entry matched means result is "word"
55. * This while loop will never be executed more than one time; it is here * only to allow the break statement to be used to escape as soon as a word * is recognized

56. * * Stems the text in the token. Returns true if changed.
57. try turning -ical to -y (e.g., bibliographical)
58. try converting -ncy to -nt
59. this routine deals with -ment endings.
60. will take care of that later)
61. restore original values
62. * always * convert * - * ally * to * - * al
63. try converting -al to -um
64. may be null, which means that "word" is the stem
65. * this is a very productive endings, so * just accept it
66. handle -eer
67. try converting -ly to -le
68. ditto for -ality
69. try removing -a/ible and adding -e
70. lookup();
71. try removing -ical
72. * this routine deals with -ity endings. It accepts -ability, -ibility, and * -ality, even without checking the dictionary because they are so * productive. The first two are mapped to -ble, and the -ity is remove for * the latter
73. try just removing the -ly
74. * the -ize ending is very productive, so simply * accept it as the root
75. nolookup(), we already did it.
76. sometimes -ial endings should be removed
77. handle -ence and -ance
78. convert to -ive
79. Convert plurals to singular form, and '-ies' to 'y'
80. ** * lookups.clear(); lookups.add(word.toString()); **
81. try removing -or and adding -e
82. try removing the "es"
83. the vowelInStem() is necessary so we don't stem acronyms
84. see if the root ends in `e'
85. (sometimes it gets turned into -y, but we
86. yes, so check against the dictionary
87. try converting -e/ance to -e (adherance/adhere)
88. * this routine deals with -er, -or, -ier, and -eer. The -izer ending is * always converted to -ize
89. handle some derivational endings
90. ** * if (!lookups.contains(word.toString())) { throw new * RuntimeException("didn't look up "+word.toString()+" prev="+prevLookup); * } **
91. * always * convert * - * ably * to * - * able
92. lookups.add(word.toString());
93. return TRUE if word ends with a double consonant
94. try removing the -al and adding -e
95. try converting -ic to -e
96. *** * YCS: this was the one place where lookup was not followed by return. * So restructure it. if ((j>0)&&(lookup(word.toString())) && * !((word.charAt(j) == 's') && (word.charAt(j-1) == 's'))) return; ****
97. try converting -ic to -ical
98. * it wasn't found by just removing the `d' or the `ed', so prefer to end * with an `e' (e.g., `microcoded' -> `microcode').
99. try mapping -ive to -ion (e.g., injunctive/injunction)
100. check left to right since the endings have often already matched
101. restore the endings
102. replace old suffix with s
103. try removing -ive and adding -e
104. note that `ing' has also been removed
105. This file was partially derived from the original CIIR University of Massachusetts Amherst version of KStemmer.java (license for the original shown below)
106. ***** * debugging code String thisLookup = word.toString(); boolean added = * lookups.add(thisLookup); if (!added) { * System.out.println("######extra lookup:" + thisLookup); //

occaasional * extra lookups aren't necessarily errors... could happen by diff * manipulations // throw new RuntimeException("######extra lookup:" + * thisLookup); } else { // System.out.println("new lookup:" + thisLookup); * } *****
107. restore the doubled consonant
108. if we didn't try the "e" ending before
109. handle `-ing' endings
110. try adding an `e' to the stem and check against the dictionary
111. remove the -r ending
112. nolookup() because we already did according to the comment
113. restore the original ending
114. don't stem don't lowercase... it's a requirement that lowercase filter be used before this stemmer.
115. adding on the `e' didn't work, so remove it
116. try just removing -ity
117. (this will sometimes screw up with `under-', but we
118. Returns true if the word is found in the dictionary
119. {"netherlands", "dutch"},
120. restore the ending to the way it was
121. * note: don't check for exceptions here. So, `aides' -> `aide', but * `aided' -> `aid'. The exception for double s is used to prevent * crosses -> crosse. This is actually correct if crosses is a plural * noun (a type of racket used in lacrosse), but the verb is much more * common
122. remove -ion, and if it's found, treat that as the root
123. e.g., militarily -> military
124. unless the word ends in "ous" or a double "s", remove the final "s"
125. * try for a direct mapping (allows for cases like `Italian'->`Italy' and * `Italians'->`Italy')
126. restore the original values
127. the default is to convert it to -nce
128. try just removing the -al
129. ** * caching off is normally faster if (cache == null) initializeStemHash(); * * // now check the cache, before we copy chars to "word" if (cache != null) * { String val = cache.get(term, 0, len); if (val != null) { if (val != * SAME) { result = val; return true; } return false; } } **
130. the default is to remove -ity altogether
131. is the best I can do
132. * just remove -ation (resignation->resign) and * check dictionary
133. (elmination -> eliminate)
134. (e.g., amplification -> amplify)
135. * this routine deals with -ive endings. It normalizes some of the -ative * endings directly, and also maps some -ive endings to -ion.
136. * this routine deals with -ion, -ition, -ation, -ization, and -ication. The * -ization ending is always converted to -ize
137. don't stem
138. ** * if (matchedEntry != null) { if (dict_ht.get(word.getArray(), 0, * word.size()) != matchedEntry) { * System.out.println("Uh oh... cached entry doesn't match"); } return * matchedEntry; } **
139. length of stem within word
140. * * This class implements the Kstem algorithm
141. ** * if (entry == null) { if (!word.toString().equals(new String(term,0,len))) * { System.out.println("CASE:" + word.toString() + "," + new * String(term,0,len)); * * } } **
142. try removing -er/-or
143. this isn't true for short words (died->die)
144. * the word wasn't in the dictionary after removing the stem, and then * checking with and without a final `e'. The default is to add an `e' * unless the word ends in two consonants, so `microcoding' -> * `microcode'. The two consonants restriction wouldn't normally be * necessary, but is needed because we don't try to deal with prefixes and * compounds, and most of the time it is correct (e.g., footstamping -> * footstamp, not footstampe; however, decoupled -> decoupl). We can * prevent almost all of the incorrect stems if we try to do some prefix * analysis first
145. try just removing -ative
146. * remove -ation and add `e', and check against the * dictionary
147. nolookup()... we already tried removing the "ly" variant
148. (e.g., compensable/compensate)
149. try removing the "ed"

150. (e.g., determinative -> determine)
151. try removing -ize and adding -e
152. try removing -ize entirely
153. the vowelinstem() is necessary so we don't stem acronyms
154. * this routine deals with -ly endings. The -ally ending is always converted * to -al Sometimes this will temporarily leave us with a non-word (e.g., * heuristically maps to heuristical), but then the -al is removed in the next * step.
155. try removing -ive entirely
156. ** * caching off is normally faster if (cache != null && cache.size() < * maxCacheSize) { char[] key = new char[len]; System.arraycopy(term, 0, * key, 0, len); if (result != null) { cache.put(key, result); } else { * cache.put(key, word.toString()); } } **
157. * * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
158. if it's in the dictionary and not an exception
159. (e.g.,`fingerspelling' -> `fingerspell'). Unfortunately
160. * if the root isn't in the dictionary, and the variant *is* there, then * use the variant. This allows `immunity'->`immune', but prevents * `capacity'->`capac'. If neither the variant nor the root form are in * the dictionary, then remove the ending as a default
161. convert past tense (-ed) to present, and `-ied' to `y'
162. Without making the algorithm significantly more complicated, this
163. handle -ness
164. * try removing a doubled consonant. if the root isn't found in the * dictionary, the default is to leave it doubled. This will correctly * capture `backfilled' -> `backfill' instead of `backfill' -> * `backfille', and seems correct most of the time
165. this routine deals with -ize endings.
166. (e.g., optimal - > optimum)
167. * test -ication after -ation is attempted (e.g., `complication->complicate' * rather than `complication->comply')
168. try removing -ic altogether
169. remove -ion and add `e', and check against the dictionary
170. allow for a doubled consonant
171. almost all uses of lookup() return immediately and are followed by another lookup in the dict. Store the match to avoid this double lookup.
172. nolookup(); all of the other paths restored original values
173. * * Returns the result of the stem (assuming the word was changed) as a String.
174. terms must be lowercased already
175. if I can remove a doubled consonant and get a word, then do so
176. try converting -ic to -y
177. (e.g., definition->define, opposition->oppose)
178. * handle short words (aging -> age) via a direct mapping. This prevents * (thing -> the) in the version of this routine that ignores inflectional * variants that are mentioned in the dictionary (when the root is also * present)
179. Copyright © 2003, Center for Intelligent Information Retrieval, University of Massachusetts, Amherst. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. 3. The names "Center for Intelligent Information Retrieval" and "University of Massachusetts" must not be used to endorse or promote products derived from this software without prior written permission. To obtain permission, contact info@ciir.cs.umass.edu. THIS SOFTWARE IS PROVIDED BY UNIVERSITY OF MASSACHUSETTS AND OTHER CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

180. do we have a -ier ending?
181. the default is to remove -ly
182. nolookup() - we already tried the "e" ending
183. only cache if it's not an exception.
184. but just in case...
185. * INDEX of final letter in word. You must add 1 to k to get * the current length of word. When you want the length of * word, use the method wordLength, which returns (k+1).
186. * Handle words less than 5 letters with a direct mapping This prevents * (fled -> fl).
187. * if it's in the dictionary and * not an exception
188. so we can remember if it was -er or -or
189. ditto for -ivity
190. try removing -able and adding -ate
191. * remove -ication and add `y', and check against the * dictionary
192. aren't dealing with that case for now)
193. nolookup() because we restored the original ending
194. * handle -ic endings. This is fairly straightforward, but this is also the * only place we try *expanding* an ending, -ic -> -ical. This is to handle * cases like `canonic' -> `canonical'
195. * -ize is very productive, so accept it * as the root
196. * try removing -e/ance altogether * (disappearance/disappear)
197. first check the stemmer dictionaries, and avoid using the cache if it's in there.
198. * remove -ition and add `e', and check against the * dictionary
199. ** * caching off private int maxCacheSize; private CharArrayMap<String> cache = * null; private static final String SAME = "SAME"; // use if stemmed form is * the same **
200. try removing -ative and adding -e
201. the default is to retain the "e"
202. Set<String> lookups = new HashSet<String>();
203. nolookup(); we already tried an 'e' ending
204. nolookup
205. index of the character BEFORE the posfix
206. convert to -ble
207. length of word before this suffix
208. if we have a `un-' prefix, then leave the word alone
209. todo
210. * * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
211. * * A StringBuilder that allows one to access the array.
212. * * test the kstemmer optimizations against a bunch of words * that were stemmed with the original java kstemmer (generated from * testCreateMap, commented out below).
213. * * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, *

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
214. ***** requires original java kstem source code to create map public void testCreateMap() throws Exception { String input = getBigDoc(); Reader r = new StringReader(input); TokenFilter tf = new LowerCaseFilter(new LetterTokenizer(r)); // tf = new KStemFilter(tf); KStemmer kstem = new KStemmer(); Map<String,String> map = new TreeMap<String,String>(); for(;;) { Token t = tf.next(); if (t==null) break; String s = t.termText(); if (map.containsKey(s)) continue; map.put(s, kstem.stem(s)); } Writer out = new BufferedWriter(new FileWriter("kstem_examples.txt")); for (String key : map.keySet()) { out.write(key); out.write('\t'); out.write(map.get(key)); out.write('\n'); } out.close(); } *****
215. * * Tests for {@link KStemmer}
216. * blast some random strings through the analyzer
217. * * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
218. * * Factory for {@link KStemFilter}
219. * * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
220. * * Simple tests to ensure the kstem filter factory is working.

git_commits:

1. **summary:** LUCENE-152: add KStem
   message: LUCENE-152: add KStem git-svn-id: https://svn.apache.org/repos/asf/lucene/dev/branches/branch_3x@1130532 13f79535-47bb-0310-9956-ffa450edef68

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** [PATCH] KStem for Lucene
   description: September 10th 2003 contributionn from "Sergio Guzman-Lara" <guzman@cs.umass.edu> Original email: Hi all, I have ported the kstem stemmer to Java and incorporated it to Lucene. You can get the source code (Kstem.jar) from the following website: http://ciir.cs.umass.edu/downloads/ Just click on "KStem Java Implementation" (you will need to register your e-mail, for free of course, with the CIIR -- Center for Intelligent Information Retrieval, UMass -- and get an access code). Content of Kstem.jar: java/org/apache/lucene/analysis/KStemData1.java java/org/apache/lucene/analysis/KStemData2.java java/org/apache/lucene/analysis/KStemData3.java java/org/apache/lucene/analysis/KStemData4.java java/org/apache/lucene/analysis/KStemData5.java java/org/apache/lucene/analysis/KStemData6.java java/org/apache/lucene/analysis/KStemData7.java java/org/apache/lucene/analysis/KStemData8.java java/org/apache/lucene/analysis/KStemFilter.java java/org/apache/lucene/analysis/KStemmer.java

KStemData1.java, ..., KStemData8.java Contain several lists of words used by Kstem KStemmer.java Implements the Kstem algorithm KStemFilter.java Extends TokenFilter applying Kstem To compile unjar the file Kstem.jar to Lucene's "src" directory, and compile it there. What is Kstem? A stemmer designed by Bob Krovetz (for more information see http://ciir.cs.umass.edu/pubfiles/ir-35.pdf). Copyright issues This is open source. The actual license agreement is included at the top of every source file. Any comments/questions/suggestions are welcome, Sergio Guzman-Lara Senior Research Fellow CIIR UMass

jira_issues_comments:

1. KStem would be nice to have in the contrib/Analysis package, but I don't see that UMass is donating the code to the ASF. It should be sufficient that people can go download and incorporate it quite easily into Lucene (I have done this in the past). Thus, I think it is fine to mark this as won't fix.
2. If the original sources are BSD licensed, is a software grant required to incorporate the sources into the Lucene/Solr source tree?
3. The general rule is that if its a fair amount of code, and it was developed outside of the Apache system, we want a software grant - even if its Apache 2 licensed code.
4. bq. even if its Apache 2 licensed code. Uh... that may be a stretch. More specifically: compile time dependencies on compiled BSD libraries are fine, but actually incorporating and *releasing* code that is under a BSD license is something we're aren't suppose to do (last time i checked)
5. bq. Uh... that may be a stretch. It's what the incubator seems to recommend, and the side have err'd on in the past. http://incubator.apache.org/ip-clearance/index.html If it was developed outside of Apache, we don't really know it's IP history, and that's something we want to take seriously.
6. **body:** To extract a bit for clarity: {quote} This form is not for new projects. This is for projects and PMCs that have already been created and are receiving a code donation into an existing codebase. Any code that was developed outside of the ASF SVN repository and our public mailing lists must be processed like this, even if the external developer is already an ASF committer. {quote}
   label: code-design
7. bq. More specifically: compile time dependencies on compiled BSD libraries are fine, but actually incorporating and releasing code that is under a BSD license is something we're aren't suppose to do (last time i checked) Code is fine to afaik: http://www.apache.org/legal/3party.html
8. bq. Code is fine to afaik: http://www.apache.org/legal/3party.html My interpretation of this is that we can directly include the KStem source code in Lucene/Solr's source tree, and then modify it at will, since its license (BSD style) is in Category A (authorized licenses). Thoughts?
9. I think that's right.
10. heh - I had heard enough times that the license wouldn't permit it that I never looked into it myself. http://markmail.org/message/zlett7y3dj76xa2f Anyway, I did a bunch of optimizations for Lucid's version way back when. It makes sense for those to be contributed back here... I'll see what I can do (but it might be delayed a week by everyone being busy at Lucene Revolution).
11. Reopening - license issues appear to have disappeared.
12. Setting fix version... seems like there's alot of interest for this stemmer and Lucid's improvements to it.
13. +1
14. +1
15. OK folks, here's Lucid's optimized version of kstemmer. Changes by Lucid to the original kstemmer are being contributed under the ASL. This is not a patch, but simply a tarball of Lucid's version. Not sure what we want to do with some of the stuff (like the biggish test files). IIRC, there were two types of optimizations... one type was efficiency (i.e. using CharArrMap, directly using a char[] in the stemmer, etc). Other optimizations actually changed the logic and code paths though, which is one reason I tested it over a whole document to ensure it still matched the original.
16. {quote} This is not a patch, but simply a tarball of Lucid's version. Not sure what we want to do with some of the stuff (like the biggish test files) {quote} I don't think biggish test files are a problem personally, we already have these for the snowball stemmers for example.
17. I'm fine with the 1.2MB history_of_the_united_states.txt in the tests
18. we can zip it anyway, the existing stemmer tests use zipped files for this exact purpose. zipped: all the test data is about 500KB our snowball test data currently in src/test is zipped 3.1MB... so I think 500kb is ok.
19. patch file, testdata zip goes in src/test/org/apache/lucene/analysis/en The testdata was converted over to tab-separated to use VocabularyAssert. The big HistoryOfUnitedStates file was actually only used in the little benchmarker, not in tests.
20. +1 Looks good! Should OpenStringBuilder and CharsRef be combined? If not, is OpenStringBuilder usful outside of analysis? Should it be in org.apache.lucene.util?

21. Ryan: maybe, I thought of this too myself looking at the patch. Then again there are probably other kinds of refactoring improvements we could make... honestly I didn't dig deep enough into this one to see if it can be solved just by 'add Appendable interface to CharsRef' or to even think if thats the right thing to do. I don't think we should move it out of the analysis package for now (maybe i shouldn't have put it in util even in the patch) unless there's something else that actually wants to use it: I think this would be premature.
22. its also worth mentioning if the class is just used for appending (not sure if it is), we might be able to just append to the CharTermAttribute directly instead, it implements Appendable already.
23. great. Another thing that jumps out is {code} CharArrayMap<DictEntry> d = new CharArrayMap<DictEntry>(Version.LUCENE_31, 1000, false); {code} Looks like we need to refactor: {code} private static final CharArrayMap<DictEntry> dict_ht = initializeDictHash(); {code} so it can be passed the Lucene Version. I'm not sure we need it to be static either... I can take a look at that if you are not already on it
24. Personally I don't think we should do this: if you look thru the analyzers you will find other similar code. Plus, there's no need to support the 'broken' pre-3.1 behavior here, since this thing isn't planned to be released until 3.3.
25. ok -- just making sure there is nothing lost with Version.LUCENE_40 +1 to commit
26. No, nothing will be lost... and actually since 'false' is passed here for ignoreCase, the constant does nothing... just looks wierd.
27. Thanks for reviewing Ryan... i found some @authors just doing another scan, i'll nuke those before committing.
28. Committed revision 1130527 (trunk), 1130532 (branch_3x). Thanks Yonik!
29. wow, closing a ticket from 2003! Thanks Robert, Yonik, etc
30. **body:** very minor optimization to avoid a char[] allocation per stemmed word.
    label: code-design
31. why create strings either?
32. bq. why create strings either? Good point. I assume you mean something like this patch?
33. it looks good... i think its the same as the patch i uploaded (_alt.patch)... only i used the .append syntactic sugar
34. bq. i think its the same as the patch i uploaded D'oh! I hate that the "All" tab in JIRA isn't selected by default (and hence one doesn't see stuff like file uploads ;-)
35. bulk close for 3.3
36. sorry for the reopen, but why is the constructor of KStemmer not public?