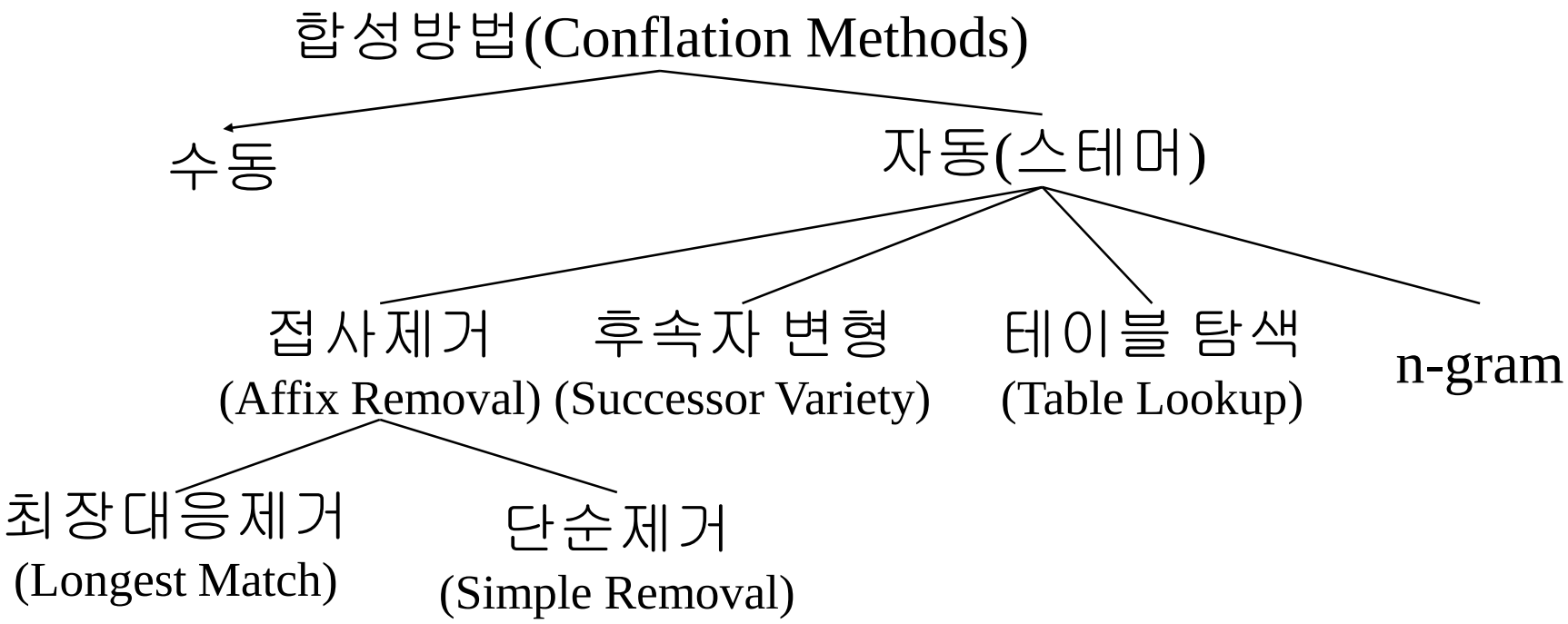
8. Stemming Algorithm

8.1 소개

- Stemming
  - 색인 파일의 크기를 줄이기 위해 정보검색시에 사용
  - 단어(어절) 대신 어간(stem)을 저장 → 50% 이상의 압축비율
  - 색인시간과 탐색시간에 stemming 기능
    - 색인시간
      - 색인어가 어간화되어 효율성과 색인파일 압축성이 증진
      - search time에 이런 연산을 위한 자원 요청이 불필요
    - 탐색시간
      - 시스템과 탐색기술에 대한 지식을 요구하지 않고도 용어 합성
      - 스테머에 의해 찾아진 용어들의 집합에서 용어선택
        → 오결합 가능성 감소

8.1 소개

- 자동합성방법
  - 몇 가지 수식을 사용한 수동이나 stemmer라 불리는 프로그램을 통해 자동처리



8.1 소개

- 접사제거(Affix Removal)
  - 하나의 어간을 남기기 위해 용어들의 접두어와 접미어 제거
- 후속자 변형(Succesor Variety)
  - 본문내의 글자가 연속으로 나타나는 빈도를 사용
- 테이블 탐색(Table Lookup)
  - 용어와 어간을 테이블에 저장하여 테이블을 탐색
- n-gram
  - 용어가 공유할 수 있는 도표나 n-gram 수에 기초한 용어들의 합성

8.2.0 테이블 탐색

- 모든 색인어와 그 어간을 하나의 테이블에 저장하는 방법

색인용어	어 간
engineering	engineer
engineered	engineer
engineer	engineer

- 질의로부터 용어들과 색인어들을 table lookup을 통해 어간화 가능 (B-tree, hash table)
- 문제점
  - 용어와 어간의 관계에 대한 자료부족, 저장 오버헤드

8.2.1 후속자 변형

- 한 문자열의 후속자 변형(successor variety)
  - 본문내의 단어들 중 글자를 후속하는 상이한 글자 수량
- 후속자 변형 스테밍 처리 3단계

  (1) 단어에 대한 후속자 변형을 결정한다.

  (2) 단어를 분할하기 위해 이 정보를 사용한다.

  (3) 어간으로서 분할 중 1개를 선택한다.
- 분할 방법
  - *Cutoff method, peak and plateau method, complete word method, entropy method*

8.2.1 후속자 변형

예) READABLE이라는 단어를 어간으로 결정하는 작업

- Test Word : READABLE → READ + ABLE
- Corpus : ABLE, APE, BEATABLE, FIXABLE, READ, READABLE, READING, READS, RED, ROPE, RIPE

접두사	후속자 변형	문 자
R	3	E, I, O
RE	2	A, D
REA	1	D
READ	3	A, I, S
READA	1	B
READAB	1	L
READABL	1	E
READABLE	1	BLANK

8.2.2 n-gram stemmers

- shared bigram method
  - 용어합성 방법으로 어간이 생성되지 않음
  - bigram : 한 쌍의 연속된 글자
- shared unique bigram에 기초하여 한 쌍의 용어들 사이에 관련성 척도 계산

  예) statistics와 statistical
  - ***statistics*** => st ta st ti is st ti ic cs => 9개의 bigram
  - unique bigrams => at cs ic is st ta ti → 7개
  - ***statistical*** => st ta at ti is st ti ic ca al => 10개의 bigram
  - unique bigrams => al at ca ic is st ta ti → 8개

  → 6개의 unique bigram 공유 : at, ic, is, st, ta, ti

  → Dice's coefficient : (2 x 6) / (7 + 8) = 0.8

8.2.2 n-gram Stemmers

- Similarity measure
  - 단어 한 쌍에 대한 unique bigram을 기초로 유사도 계산
  - Dice 상관계수 : S = 2C / (A+B)
    - A, B : 각각 첫번째, 두번째 단어의 unique bigram 개수
    - C : 공유한 unique bigram 개수
  - 유사도 : 데이터베이스내에 있는 용어의 모든 쌍에 대해 결정하며 유사성 행렬을 형성

8.2.3 Affix Removal Stemmers

- 어간을 남겨두기 위해 용어로부터 접미/접두어를 제거

  예) 명사의 복수형에 대한 스테머 규칙

  - If a word ends in ”ies” but not "eies” or ”aies" then "ies" → "y”
  - If a word ends in ”es” but not "aes", "ees" or "oes” then "es" → "e”
  - If a word ends in ”s” but not “us” or ”ss” then ”s” → NULL

- Most stemmers are “*iterative longest match stemmers*”

  1. 규칙에 따라 한 단어로부터 가능한 가장 긴 스트링 제거
  2. 더 이상의 문자들이 제거되지 않을 때 까지 계속 반복

Porter's Algorithm(1)

- Consists of a set of condition/action rules

- Three kind of conditions
  1. Stem conditions
  2. Suffix conditions
  3. Rule conditions

Porter's Algorithm(2)

- Stem conditions
  1. Measure *m* indicates the number of VC sequences.
     $[C](VC)^m[V]$ → C: consonant, V: vowel

Measure	Examples
m=0	TR, EE, TREE, Y, BY
m=1	TROUBLE, OATS, TREES, IVY
m=2	TROUBLES, PRIVATE, OATEN

  2. *<X> : the stem ends with a given letter X
  3. *v* : the stem contains a vowel
  4. *d : the stem ends in a double consonant
  5. *o : the stem ends with a consonant-vowel-consonant, where the final consonant is not w, x, and y.

Porter's Algorithm(3)

- Suffix conditions:

  current_suffix == pattern
- Rule conditions: (rule was used)
- Actions are rewrite rules:

  old_suffix → new_suffix
- Stemming rules
  1. The rules in a step are examined in sequence.
  2. Only one rule from a step can apply.
  3. The longest possible suffix is always removed because of the ordering of the rules within a step.

1. **step1a(word);**
2. **step1b(stem);**
3. **If (the second or third rule of step 1b was used) step1b1(stem);**
4. **step1c(stem);**
5. **step2(stem);**
6. **step3(stem);**
7. **step4(stem);**
8. **step5a(stem);**
9. **step5b(stem);**

Porter's Algorithm(4)

Step 1a Rules

Conditions	Suffix	Replacement	Examples
NULL	sses	ss	caresses -> caress
NULL	ies	i	ponies -> poni
			ties -> tie
NULL	ss	ss	carress -> carress
NULL	s	NULL	cats -> cat

Step 1b Rules

Conditions	Suffix	Replacement	Examples
(m>0)	eed	ee	feed -> feed
			agreed -> agree
(*v*)	ed	NULL	plastered -> plaster
			bled -> bled
(*v*)	ing	NULL	motoring -> motor
			sing -> sing

Porter's Algorithm(5)

Step 1b1 Rules

Conditions	Suffix	Replacement	Examples
NULL	at	ate	conflat(ed) -> conflate
NULL	bl	ble	troubl(ing) -> trouble
NULL	iz	ize	siz(ed) -> size
(*d and not (*<L> or *<S> or *<Z>))	NULL	single letter	hopp(ing) -> hop tann(ed) -> tan fall(ing) -> fall hiss(ing) -> hiss fizz(ed) -> fizz
(m=1 and *o)	NULL	e	fail(ing) -> fail fil(ing) -> file

Step 1c Rules

Conditions	Suffix	Replacement	Examples
(*v*)	y	i	happy -> happi sky -> sky

Porter's Algorithm(6)

Step 2 Rules

Conditions	Suffix	Replacement	Examples
(m>0)	ational	ate	relational -> relate
(m>0)	tional	tion	conditional -> condition
			rational -> rational
(m>0)	enci	ence	valenci -> valence
(m>0)	anci	ance	hesitanci -> hesitance
(m>0)	izer	ize	digitizer -> digitize
(m>0)	abli	able	conformabli -> conformable
(m>0)	alli	al	radicalli -> radical
(m>0)	entli	ent	differentli -> different
(m>0)	eli	e	vileli -> vile
(m>0)	ousli	ous	analogousli -> analogous
(m>0)	ization	ize	vietnamization -> vietnamize
(m>0)	ation	ate	predication -> predicate
(m>0)	ator	ate	operator -> operate
(m>0)	alism	al	feudalism -> feudal
(m>0)	iveness	ive	decisiveness -> decisive
(m>0)	fulness	ful	hopefulness -> hopeful
(m>0)	ousness	ous	callousness -> callous
(m>0)	aliti	al	formaliti -> formal
(m>0)	iviti	ive	sensitiviti -> sensitive
(m>0)	biliti	ble	sensibiliti -> sensible

Porter's Algorithm(7)

Conditions	Suffix	Replacement	Examples
(m>0)	icate	ic	triplicate -> triplic
(m>0)	ative	NULL	formative -> form
(m>0)	alize	al	formalize -> formal
(m>0)	iciti	ic	electriciti -> electric
(m>0)	ical	ic	electrical -> electric
(m>0)	ful	NULL	hopeful -> hope
(m>0)	ness	NULL	goodness -> good

Conditions	Suffix	Replacement	Examples
(m>1)	al	NULL	revival -> reviv
(m>1)	ance	NULL	allowance -> allow
(m>1)	ence	NULL	inference -> infer
(m>1)	er	NULL	airliner -> airlin
(m>1)	ic	NULL	gyroscopic -> gyroscop
(m>1)	able	NULL	adjustable -> adjust
(m>1)	ible	NULL	defensible -> defens
(m>1)	ant	NULL	irritant -> irrit
(m>1)	ement	NULL	replacement -> replac
(m>1)	ment	NULL	adjustment -> adjust
(m>1)	ent	NULL	dependent -> depend
(m>1 and (*<S> or *<T>))	ion	NULL	adoption->adopt
(m>1)	ou	NULL	homologou->homolog
(m>1)	ism	NULL	communism->commun
(m>1)	ate	NULL	activate->activ
(m>1)	iti	NULL	angulariti->angular
(m>1)	ous	NULL	homologous ->homolog
(m>1)	ive	NULL	effective->effect
(m>1)	ize	NULL	bowdlerize->bowdler

Porter's Algorithm(8)

Step 5a Rules

Conditions	Suffix	Replacement	Examples
(m>1)	e	NULL	probate -> probat rate- > rate
(m=1 and not *o)	e	NULL	cease- > ceas

Step 5b Rules

Conditions	Suffix	Replacement	Examples
(m>1 and *d and *<L>)	NULL	single letter	controll -> control roll -> roll

8.3 스테밍의 실험 평가

- 스테머 평가기준
  - 정확성, 검색효과, 압축성능

- 예) 사용자 질의의 예

  Look for : users

  Search term : users

  Term Occurences

  1. user 15
  2. users 1
  3. used 3
  4. using 2

  which terms(O=one, CR=all) → 찾고자 하는 용어 번호 입력