UNIT 4

STEMMING ALGORITHMS

INTRODUCTION

One technique for improving IR performance is to provide searchers with ways of finding morphological variants of search terms.

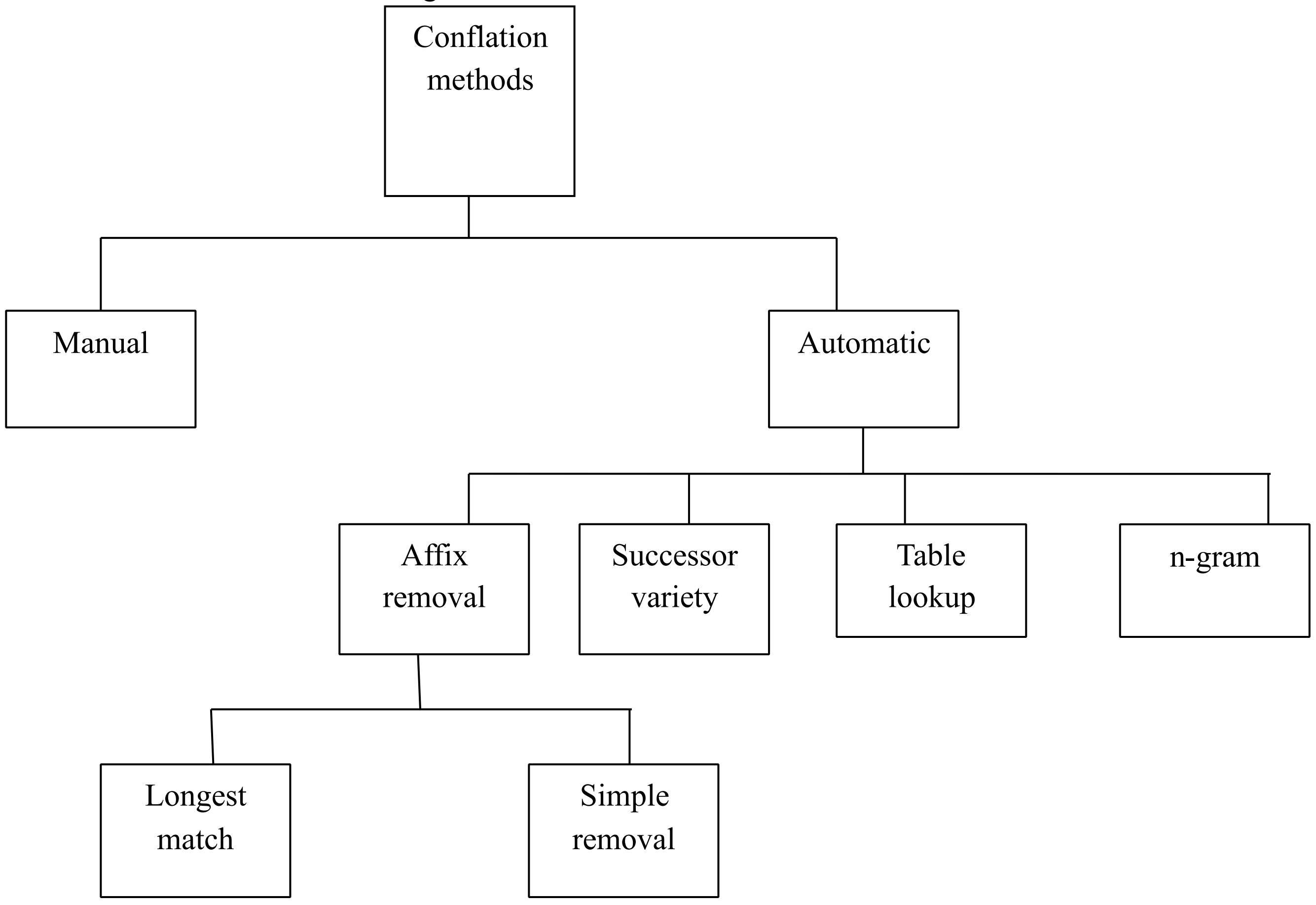
Example: A searcher enters the term stemming as part of a query, it is likely that he or she will also be interested in such variants as stemmed and stem.

Conflation: It is the term frequently used to refer to mapping multiple morphological variants to a single representation. Conflation can be either manual or automatic.

Manual conflation: Using some kind of regular expressions.

Automatic conflation: Using programs (stemmers).

Terms can be stemmed at indexing time or at search time.



The stem carries the meaning of the concept associated with the word and the affixes (endings) introduce subtle modifications to the concept. For example: the stem **"comput"** could associate **"computable, computability, computation, computational, computed, computing, computer, computerize"** to one compressed word.

Several criteria for judging stemmers:

- Correct users
- Retrieval effectiveness
- Compression performance

Two ways stemming can be incorrect:

- Over stemming (too much removal)
- Under stemming (too little removal)

Both have impact on IR performance, such as compression performance.

Example of Stemmer Use in Searching

In CATALOG, terms are stemmed at search time rather than at indexing time. CATALOG prompts for queries with the string "Look for:" At the prompt, the user types in one or more terms of interest. For example: Look for: system users will cause CATALOG to attempt to find documents about system users. CATALOG takes each term in the query, and tries to determine which other terms in the database might have the same stem. If any possibly related terms are found, CATALOG presents them to the user for selection. In the case of the query term "users," for example, CATALOG might respond as follows:

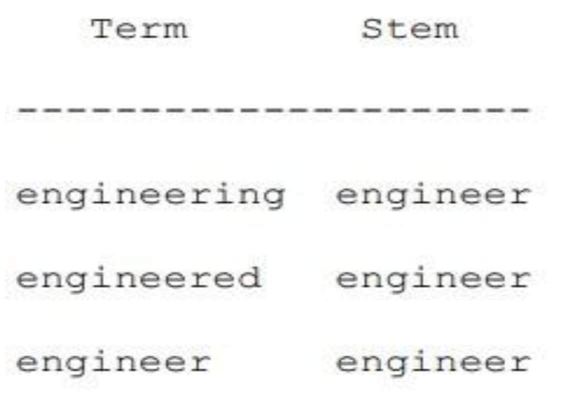
Search Term: users

Term	Occurrences
user	15
users	1
used	3
using	2

Which terms (0 = none, CR = all):

<u>TYPES OF STEMMING ALGORITHMS</u>

There are several approaches to stemming. One way to do stemming is to store a table of all index terms and their stems. For example:



```
   Term         Stem
---------------------
engineering  engineer
engineered   engineer
engineer     engineer
```

Terms from queries and indexes could then be stemmed via table lookup. Using a B-tree or hash table, such lookups would be very fast.

There are problems with this approach are:

- There is no such data for English
- Hence no standard database exists then some other stemming method would be needed.
- Storage overhead for such a table.

Successor Stemmers

Successor variety algorithm: The successor variety of a string is the number of different characters that follow it in words in some body of text. The successor variety of substrings of a term will decrease as more characters are added until a segment boundary is reached. Successor variety stemmers are based on work in structural linguistics. When this process is carried out using a large body of text the successor variety of substrings of a term will decrease as more characters are added until a segment boundary is reached. At this point, the successor variety will sharply increase. This information is used to find the stem. When the successor varieties for a given word have been derived, the information must be used to segment the word.

```
Test Word:  READABLE

Corpus:     ABLE, APE, BEATABLE, FIXABLE, READ, READABLE

            READING, READS, RED, ROPE, RIPE.
```

Prefix	Successor Variety	Letters
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

The following four ways of doing this:

1. **The cutoff method:**
   - Some cutoff value (Threshold) is selected for successor varieties and a boundary is reached.
   - The problem with this method is that if the threshold value selected is too small, incorrect cuts will be made; if it is too large, correct cuts will miss.

     Example: segment when successor variety >= threshold consider threshold = 2

Segment: R|E|AD|ABLE

2. **Peak and plateau method**
    - A segment break is made after a character whose successor variety exceeds that of the character immediately preceding it and the character immediately following it.
    - This method does not suffer from the problem of the cutoff method.
    - It breaks at the character whose successor variety is greater than both its preceding and following character READ|ABLE
3. **Complete word method**
    - A break is made after a segment if the segment is a complete word in the corpus (READ)
4. **Entropy method:**
    - Takes advantage of the distribution of successor variety letters. The method works as follows:
    - Let | Dαi | be the number of words in a text body beginning with the i length sequence of letters α.
    - Let |Dαij | the number of words in |Dαi | with the successor j computed in step one.
    - The probability that a word has the successor j is given by:

The entropy of $|D_{\alpha i}|$ is: $\frac{|D_{\alpha ij}|}{|D_{\alpha i}|}$

$$H_{\alpha i} = \sum_{j=1}^{26} -\frac{|D_{\alpha ij}|}{|D_{\alpha i}|} \cdot \log_2 \frac{|D_{\alpha ij}|}{|D_{\alpha i}|}$$

Entropy method

- for i = 2, α = RE, $|D_{\alpha i}|$ = 5
- for j = 'A', $|D_{\alpha ij}|$ = 4
- for j = 'D', $|D_{\alpha ij}|$ = 1
- *Haij* = - 1/5 * log2 (1/5) - 4/5 * log2 (4/5) = 0.46 + 0.26 = 0.72

This value is low because "REA..." appears four times!

In summary, the successor variety stemming process has 3 parts:

- Determine the successor varieties of a word.
- Use this information to segment the word using one of the methods above.
- Select one of the segments as a stem.

N-Gram Stemmers

N-Gram can be viewed as a special technique for conflation (stemming) and as a unique data structure in information systems. N-Grams are a fixed length consecutive series of "n" characters. Unlike stemming that generally tries to determine file stem of a word that represents the semantic meaning of the word, n-grams do not care about semantics.

- For example, the terms statistics and statistical can be broken into digrams as follows.

```
statistics => st ta at ti is st ti ic cs

unique digrams = at cs ic is st ta ti

statistical => st ta at ti is st ti ic ca al

unique digrams = al at ca ic is st ta ti
```

- Thus, "statistics" has nine digrams, seven of which are unique, and "statistical" has ten digrams, eight of which are unique. The two words share six unique digrams: at, ic, is, st, ta, ti.
- Once the unique digrams for the word pair have been identified and counted, a similarity measure based on them is computed.
- The similarity measure used was Dice's coefficient, which is defined as

$$S = \frac{2C}{A + B}$$

- where A is the number of unique digrams in the first word, B the number of unique digrams in the second, and C the number of unique digrams shared by A and B.
- For the example above, Dice's coefficient would equal (2 x 6)/(7 + 8) = .80.
- Such similarity measures are determined for all pairs of terms in the database, forming a similarity matrix.

Examples of bigrams, trigrams and pentagrams are given in below example for the word phrase "sea colony". For n-grams, with n greater than two, some systems allow interword symbols to be part of the n-gram set usually excluding the single character with inter word symbol option. The symbol # is used to represent the inter word symbol which is anyone of a set of symbols (e.g., blank, period, semicolon, colon etc)

se ea co ol lo on ny	Bigrams(no interword symbols)
sea col olo lon ony	Trigrams(no interword symbols)
#se sea ea# #co col olo lon ony ny#	Trigrams(with interword symbol #)
#sea# #colo colon olony lony#	Pentagrams(with interword symbol #)

Bigrams, Trigrams and Pentagrams for "sea colony" N-Gram data structure

N-Gram is a data structure that ignores words and treats the input as a continuous data, optionally limiting its processing by interword symbols. The data structure consists of fixed length overlapping symbol segments that define the searchable processing tokens. These tokens have logical linkages to all the items in which the tokens are found.

The advantage of n-grams is that they place a finite limit on the number of searchable tokens.

$$\text{MaxSeg}_n = (\lambda)^n$$

The maximum number of unique n-grams that can be generated, MaxSeg, can be calculated as a function of n which is the length of the n-grams, and λ which is the number of processable symbols from the alphabet (i.e., non-interword symbols).

Affix Removal Stemmers (Porter stemming Algorithm)

- Affix removal algorithms remove suffixes and/or prefixes from terms leaving a *stem*.
- A simple example of an affix removal stemmer is one that removes the plurals from terms.

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 currently in use are iterative longest match stemmers, a kind of affix removal stemmer first developed by Lovins (1968).
- An iterative longest match stemmer removes the longest possible string of characters from a word according to a set of rules.
- This process is repeated until no more characters can be removed. Even after all characters have been removed, stems may not be correctly conflated.
- The word "skies," for example, may have been reduced to the stem "ski" which will not match "sky." There are two techniques to handle this--recoding or partial matching.
- Recoding is a context sensitive transformation of the form AxC -> AyC where A and C specify the context of the transformation, x is the input string, and y is the transformed string.
- We might, for example, specify that if a stem ends in an "i" following a "k," then i -> y.
- In partial matching, only the n initial characters of stems are used in comparing them.
- Using this approach, we might say that two stems are equivalent if they agree in all but their last characters.
- The Porter algorithm is more compact than Lovins, Salton, and Dawson, and seems, on the basis of experimentation, to give retrieval performance comparable to the larger algorithms.

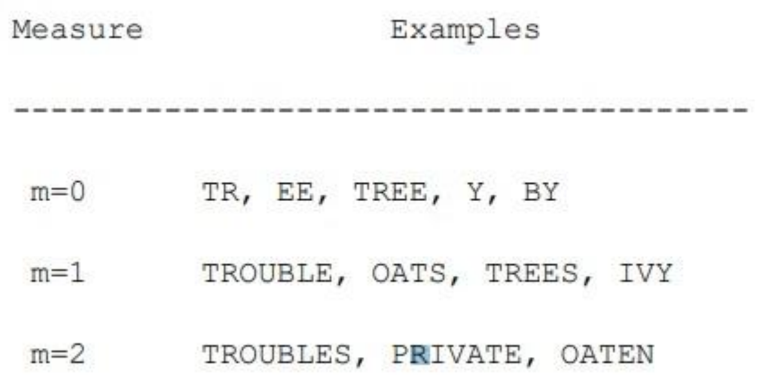
- The Porter algorithm consists of a set of condition/action rules. The conditions fall into three classes: **conditions on the stem, conditions on the suffix, and conditions on the rules**.

There are several types of stem conditions:

- The measure, denoted m, of a stem is based on its alternate vowel-consonant sequences. Vowels are (a, e, i, o, u) and y if preceded by a consonant. Consonants are all letters that are not vowels. Let C stand for a sequence of consonants, and V for a sequence of vowels. The measure m, then, is defined as:

$$[C](VC)^{m}[V]$$

The superscript m in the equation, which is the measure, indicates the number of VC sequences. Square brackets indicate an optional occurrence. Some examples of measures for terms follow.



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

- < X >--the stem ends with a given letter X
- *v*--the stem contains a vowel
- *d--the stem ends in a double consonant
- *o--the stem ends with a consonant-vowel-consonant, sequence, where the final consonant is not w, x, or y.

Suffix conditions take the form: (current_suffix == pattern).

Actions are rewrite rules of the form:

old_suffix -> new_suffix

The rules are divided into steps to define the order of applying the rules. The following are some examples of the rules:

The rules for the steps of the stemmer are as follows.

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

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

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

Step 3 Rules

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

Step 4 Rules

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

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

- The rules in a step are examined in sequence, and only one rule from a step can apply. The longest possible suffix is always removed because of the ordering of the rules within a step. The algorithm is as follows.

```
{
step1a(word);
step1b(stem);
if (the second or third rule of step 1b was used)
step1b1(stem);
step1c(stem);
step2(stem);
step3(stem);
step4(stem);
step5a(stem);
step5b(stem);
}
```

EXAMPLE 1:

Let's trace the algorithm with **word = "relational"**:

1. **Step 1a:** no plural → stays *relational*
2. **Step 1b:** no *ed/ing* → unchanged
3. **Step 1c:** no final *y* → unchanged
4. **Step 2:** *ational* → *ate* → *relate*
5. **Step 3:** no further suffix match
6. **Step 4:** no removal
7. **Step 5a/5b:** final *e* not removed → final stem = **relate**

Example 2:

word = "Controllably"

Step	**Action**	**Result**
Step 1a	no plural suffix	controllably
Step 1b	no *ed/ing*	controllably
Step 1c	no *y* → *i*	controllably
Step 2	*ably* → *able*	controllable
Step 3	remove *able* → *NULL* (m>0)	controll
Step 4	check for other suffixes (none)	controll
Step 5a	no final e	controll
Step 5b	ends with *ll*, m>1 → remove one	**control**

EXPERIMENTAL EVALUATION OF STEMMING TO COMPRESS INVERTED FILES

There have been many experimental evaluations of stemmers:

Salton (1968) examined the relative retrieval performance of fully stemmed terms against terms with only the suffix "s" removed. Three document collections were used in these studies: the IRE-3 collection consisting of 780 computer science abstracts and 34 search requests, the ADI collection consisting of 82 papers and 35 search requests, and the Cranfield-1 collection consisting of 200 aerodynamics abstracts and 42 search requests. Differences between the two conflation methods, fully stemmed and "s" stemmed, were calculated on 14 dependent variables for each query: rank recall, log precision, normalized recall, normalized precision, and precision for ten recall levels. As Salton points out, these measures are probably inter correlated, and since the inferential tests used require independence of the dependent variables, the results reported must be viewed with caution. This data was analyzed using both related group t tests and sign tests. Both of these statistical methods yielded significant results at the .05 probability level, but since none of the t values are reported, precluding their use in an estimate of effect size, and since sufficient data is provided for the estimate of effect size from the sign tests.

Note: The effect size for a sign test can only take on values in the range 0 to .5.

Hafer and Weiss (1974) tested their stemmer against other stemming methods using the ADI collection, and the Carolina Population Center (CPC) Collection consisting of 75 documents and five queries. Comparisons were made on the basis of recall-precision plots. No statistical testing was reported.

Van Rijsbergen et al. (1980) tested their stemmer (Porter 1980) against the stemmer described by Dawson (1974) using the Cranfield-1 test collection. Both of these stemmers are of the longest match type, though the Dawson stemmer is more complex. They report that the performance of their stemmer was slightly better across ten paired recall-precision levels, but that the observed differences are probably not meaningful. No statistical results of any kind are reported.

Katzer et al. (1982) examined the performance of stemmed title-abstract terms against six other document representations: unstemmed title-abstract terms, unstemmed abstract terms, unstemmed title terms, descriptors, identifiers, and descriptors and identifiers combined. The stemmer used was a simple suffix removal stemmer employing 20 endings.

Lennon et al. (1981) examined the performance of various stemmers both in terms of retrieval effectiveness and inverted file compression. For the retrieval comparisons, the Cranfield-1400 collection was used. This collection contains 1,396 documents, and also 225 queries. The titles of documents and the queries were matched, and the documents ranked in order of decreasing match value.

Frakes (1982) did two studies of stemming. The first tested the hypothesis that the closeness of stems to root morphemes will predict improved retrieval effectiveness. Fifty-three search queries were solicited from users and searched by four professional searchers under four representations: title, abstract, descriptors, and identifiers.

Walker and Jones (1987) did a thorough review and study of stemming algorithms. They used Porter's stemming algorithm in the study. The database used was an on-line book catalog (called RCL) in a library.

Harman (1991) used three stemmers--Porter, Lovins, and S removal--on three databases-Cranfield 1400, Medlars, and CACM--and found that none of them significantly improved retrieval effectiveness in a ranking IR system called IRX. The dependent variable measure was again E, at b levels of .5, 1.0, and 2.0.

STEMMING TO COMPRESS INVERTED FILES

Since a stem is usually shorter than the words to which it corresponds, storing stems instead of full words can decrease the size of index files. The following compression percentages for various stemmers and databases.

Stemmer	Cranfield	National physical laboratory	INSPEC	Brown Corpus
INSPEC	32.1	40.7	40.5	47.5
Lovins	30.9	39.2	39.5	45.8
RADCOL	32.1	41.8	41.8	49.1
Porter	26.2	34.6	33.8	38.8
Frequency	30.7	39.9	40.1	50.5

For a database of 50 megabytes, the index was reduced from 6.7 to 5.8 megabytes--a savings of 13.5 percent.

THESAURUS CONSTRUCTION

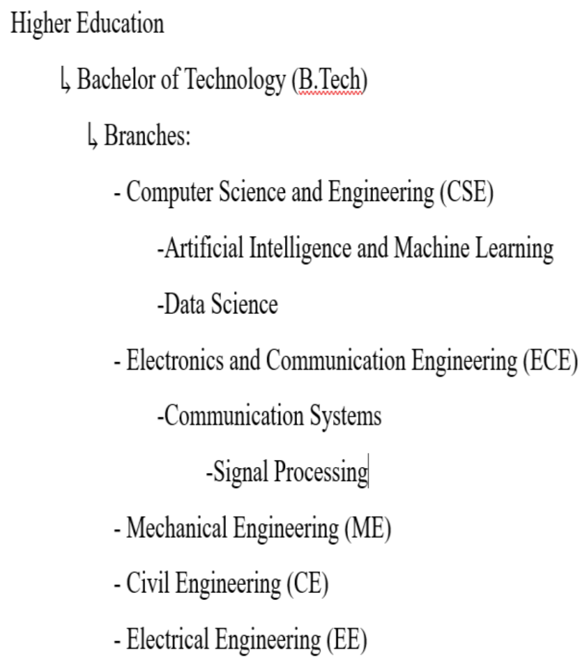
INTRODUCTION:

- Thesauri are valuable structures for Information Retrieval systems.
- A thesaurus provides a precise and controlled vocabulary which serves to coordinate document indexing and document retrieval.
- In both indexing and retrieval, a thesaurus may be used to select the most appropriate terms.
- Additionally, the thesaurus can assist the searcher in reformulating search strategies if required.
- Two major approaches for automatic thesaurus construction have been selected for detailed examination.
- The first is on thesaurus construction from collections of documents, and the second, on thesaurus construction by merging existing thesauri.
- In IR systems, used to retrieve potentially relevant documents from large collections, the thesaurus serves to coordinate the basic processes of indexing and document retrieval.
- In indexing, a succinct representation of the document is derived, while retrieval refers to the search process by which relevant items are identified.
- The IR thesaurus typically contains a list of terms, where a term is either a single word or a phrase, along with the relationships between them.
- It provides a common, precise, and controlled vocabulary which assists in coordinating indexing and retrieval.

Symbol	Meaning
BT	Broader Term(More general)
NT	Narrower Term(More specific)
RT	Related Term
UF	Use For (non-preferred term)
USE	Preferred term to use
TT	Top Term

```
cesium
          USE caesium
computer-aided instruction
          see also education
          UF   teaching machinse
          BT   educational computing
          TT   computer applications
          RT   education
               teaching
          CC   C7810C
          FC   7810Cf
```

HIERARCHY

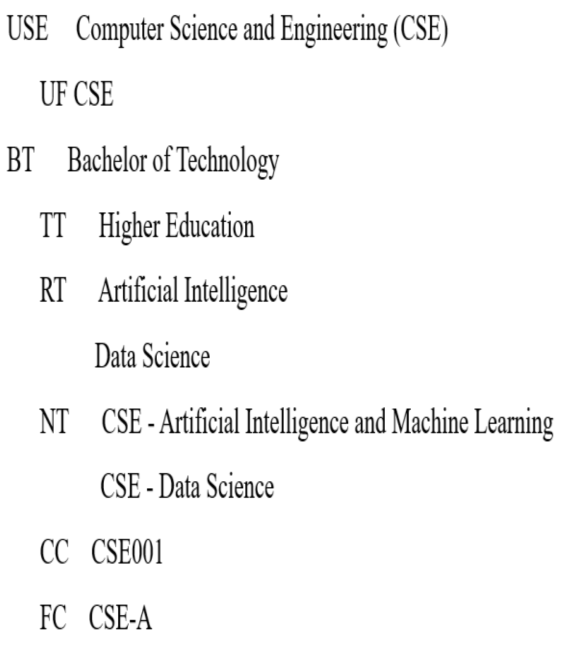


Higher Education

↳ Bachelor of Technology (B.Tech)

↳ Branches:

- Computer Science and Engineering (CSE)
  - -Artificial Intelligence and Machine Learning
  - -Data Science
- Electronics and Communication Engineering (ECE)
  - -Communication Systems
    - -Signal Processing
- Mechanical Engineering (ME)
- Civil Engineering (CE)
- Electrical Engineering (EE)



USE Computer Science and Engineering (CSE)

UF CSE

BT Bachelor of Technology

TT Higher Education

RT Artificial Intelligence

Data Science

NT CSE - Artificial Intelligence and Machine Learning

CSE - Data Science

CC CSE001

FC CSE-A

- **CC (Classification Code)** indicates the **subject category** or **broad area** where a term belongs.
- **FC (Facet Code)** indicates the **facet** or **contextual category** in which the term is used within that subject area.

FEATURES OF THESAURI

Some important features of thesauri will be highlighted here.

1. **Coordination Level**
   - Coordination refers to the construction of phrases from individual terms. Two distinct coordination options are recognized in thesauri: *precoordination* and *post-coordination*. A pre coordinated thesaurus is one that can contain phrases. Consequently, phrases are available for indexing and retrieval. A post coordinated thesaurus does not allow phrases. Instead, phrases are constructed while searching. The choice between the two options is difficult. The advantage in pre coordination is that the vocabulary is very precise, thus reducing ambiguity in indexing and in searching. Also, commonly accepted phrases become part of the vocabulary. However, the disadvantage is that the searcher has to be aware of the phrase construction rules employed.
   - Pre coordination is more common in manually constructed thesauri. Automatic phrase construction is still quite difficult and therefore automatic thesaurus construction usually implies post-coordination.

2. **Term Relationships**

    - Term relationships are the most important aspect of thesauri since the vocabulary connections they provide are most valuable for retrieval. Many kinds of relationships are expressed in a manual thesaurus. These are semantic in nature and reflect the underlying conceptual interactions between terms. Three categories of term relationships:

(1) equivalence relationships,
(2) hierarchical relationships, and
(3) non-hierarchical relationships.

- *Equivalence relations* include both synonymy and quasi-synonymy.. Quasi synonyms are terms which for the purpose of retrieval can be regarded as synonymous, for example, "genetics" and "heredity," which have significant overlap in meaning. Also, the terms "harshness" and "tenderness," which represent different viewpoints of the same property continuum.
- A typical example of a *hierarchical relation* is genus-species, such as "dog" and "german shepherd."
- *Non-hierarchical relationships* also identify conceptually Related terms. There are many examples including: thing--part such as "bus" and "seat"; thing--attribute such as "rose" and "fragrance."

3. **Number of Entries for Each Term**

    - It is in general preferable to have a single entry for each thesaurus term. However, this is seldom achieved due to the presence of homographs--words with multiple meanings. Also, the semantics of each instance of a homograph can only be contextually deciphered. Therefore, it is more realistic to have a unique representation or entry for each meaning of a homograph. This also allows each homograph entry to be associated with its own set of relations.

4. **Specificity of Vocabulary**

    - Specificity of the thesaurus vocabulary is a function of the precision associated with the component terms. A highly specific vocabulary is able to express the subject in great depth and detail. This promotes precision in retrieval. The concomitant disadvantage is that the size of the vocabulary grows since a large number of terms are required to cover the concepts in the domain.

5. **Control on Term Frequency of Class Members**

    - This has relevance mainly for statistical thesaurus construction methods which work by partitioning the vocabulary into a set of classes where each class contains a collection of equivalent terms and have stated that in order to maintain a good match between documents and queries, it is necessary to ensure that terms included in the same thesaurus class have roughly equal frequencies. Further, the total frequency in each class should also be roughly similar.

6. **Normalization of Vocabulary**

- Normalization of vocabulary terms is given considerable emphasis in manual thesauri. There are extensive rules which guide the form of the thesaural entries. A simple rule is that terms should be in noun form. A second rule is that noun phrases should avoid prepositions unless they are commonly known. Also, a limited number of adjectives should be used. There are other rules to direct issues such as the singularity of terms, the ordering of terms within phrases, spelling, capitalization, transliteration, abbreviations, initials, acronyms, and punctuation.

THESAURUS CONSTRUCTION

1. **Manual Thesaurus Construction**

The process of manually constructing a thesaurus is both an art and a science. We present here only a brief overview of this complex process. First, one has to define the boundaries of the subject area. (In automatic construction, this step is simple, since the boundaries are taken to be those defined by the area covered by the document database.) Boundary definition includes identifying central subject areas and peripheral ones since it is unlikely that all topics included are of equal importance. Once this is completed, the domain is generally partitioned into divisions or subareas.

Since manual thesauri are more complex structurally than automatic ones, as the previous section has shown, there are more decisions to be made. Now, the collection of terms for each subarea may begin. A variety of sources may be used for this including indexes, encyclopedias, handbooks, textbooks, journal titles and abstracts, catalogues, as well as any existing and relevant thesauri or vocabulary systems. Subject experts and potential users of the thesaurus should also be included in this step. Once the initial organization has been completed, the entire thesaurus will have to be reviewed (and refined) to check for consistency such as in phrase form and word form.

2. **Automatic Thesaurus Construction**

In selecting automatic thesaurus construction approaches for discussion here, the criteria used are that they should be quite different from each other in addition to being interesting. Also, they should use purely statistical techniques. (The alternative is to use linguistic methods.) Consequently, the two major approaches selected here have not necessarily received equal attention in the literature. The first approach, on designing thesauri from document collections, is a standard one. The second, on merging existing thesauri, is better known using manual methods.

(i) From a Collection of Document Items

Here the idea is to use a collection of documents as the source for thesaurus construction. This assumes that a representative body of text is available. The idea is to apply statistical procedures to identify important terms as well as their significant relationships. It is reiterated here that the central thesis in applying statistical methods is to use computationally simpler methods to identify the more important semantic knowledge for thesauri. It is semantic knowledge that is used by both indexer and searcher. Until more direct methods are known, statistical methods will continue to be used.

(ii) By Merging Existing Thesauri

This second approach is appropriate when two or more thesauri for a given subject exist that need to be merged into a single unit. If a new database can indeed be served by merging two or more existing thesauri, then a merger perhaps is likely to be more efficient than producing the thesaurus from scratch.

3. User Generated Thesaurus

In this third alternative, the idea is that users of IR systems are aware of and use many term relationships in their search strategies long before these find their way into thesauri. The objective is to capture this knowledge from the user's search.

THESAURUS CONSTRUCTION FROM TEXTS

The overall process may be divided into three stages:

1. Construction of vocabulary: This involves normalization and selection of terms. It also includes phrase construction depending on the coordination level desired.

2. Similarity computations between terms: This step identifies the significant statistical associations between terms.

3. Organization of vocabulary: Here the selected vocabulary is organized, generally into a hierarchy.

1. Construction of vocabulary

The objective here is to identify the most informative terms (words and phrases) for the thesaurus vocabulary from document collections. The first step is to identify an appropriate document collection. The only loosely stated criterion is that the collection should be sizable and representative of the subject area. The next step is to determine the required specificity for the thesaurus.

Stem evaluation and selection

There are a number of methods for statistically evaluating the worth of a term. The ones we discuss here are:

(1) selection of terms based on frequency of occurrence,

(2) selection of terms based on Discrimination Value,

(3) selection of terms based on the Poisson model.

(1) Selection by Frequency of Occurrence

The basic idea is that each term may be placed in one of three different frequency categories with respect to a collection of documents: high, medium, and low frequency. Terms in the midfrequency range are the best for indexing and searching. Terms in the low-frequency range have minimal impact on retrieval, while high-frequency terms are too general and negatively impact search precision. Salton recommends creating term classes for the low-frequency terms.

(2) Selection by Discrimination Value (DV)

DV measures the degree to which a term is able to discriminate or distinguish between the documents of the collection. The more discriminating a term, the higher its value as an index term. The overall procedure is to compute the average inter document similarity in the collection, using some appropriate similarity function. Next, the term k being evaluated is removed from the indexing vocabulary and the same average similarity is recomputed. The discrimination value (DV) for the term is then computed as:

DV(k) = (Average similarity without k) - (Average similarity with k)

(3) Selection by the Poisson Method

The Poisson distribution is a discrete random distribution that can be used to model a variety of random phenomena including the number of typographical errors in a page of writing and the number of red cars on a highway per hour. In all the research that has been performed on the family of Poisson models, the one significant result is that trivial words have a single Poisson distribution, while the distribution of nontrivial words deviates significantly from that of a Poisson distribution.

Phrase construction

This step may be used to build phrases if desired. As mentioned before, this decision is influenced by the coordination level selected. Also, phrase construction can be performed to decrease the frequency of high frequency terms and thereby increase their value for retrieval.

(i) **Salton and McGill Procedure:** This procedure is a statistical alternative to syntactic and/or semantic methods for identifying and constructing phrases. Basically, a couple of general criteria are used. First, the component words of a phrase should occur frequently in a common context, such as the same sentence.

The second general requirement is that the component words should represent broad concepts, and their frequency of occurrence should be sufficiently high. These criteria motivate their algorithm, which is described below:

- ➢ Compute pair wise co-occurrence for high-frequency words. (Any suitable contextual constraint such as the ones above may be applied in selecting pairs of terms.)
- ➢ If this co-occurrence is lower than a threshold, then do not consider the pair any further.
- ➢ For pairs that qualify, compute the cohesion value. Two formulas for computing cohesion are given below. Both ti and tj represent terms, and size-factor is related to the size of the thesaurus vocabulary.

COHESION (ti, tj) = co-occurrence-frequency/sqrt(frequency(ti) * frequency(tj))
COHESION (ti, tj) = size-factor * (co-occurrence-frequency/(total-frequency(ti) * total-frequency(tj)))

If cohesion is above a second threshold, retain the phrase as a valid vocabulary phrase.

(ii) **Choueka Procedure*:*** The second phrase construction method is based on the work by Choueka (1988). He proposes a rather interesting and novel approach for identifying collocational expressions by which he refers to phrases whose meaning cannot be derived in a simple way from that of the component words, for example, "artificial intelligence." The algorithm proposed is statistical and combinatorial and requires a large collection (at least a million items) of documents to be effective. The following are the steps:

1) Select the range of length allowed for each collocational expression.
Example: two to six words.
2) Build a list of all potential expressions from the collection with the prescribed length that have a minimum frequency (again, a preset value).
3) Delete sequences that begin or end with a trivial word. The trivial words include prepositions, pronouns, articles, conjunctions, and so on.
4) Delete expressions that contain high-frequency nontrivial words.
5) Given an expression such as a b c d, evaluate any potential sub expressions such as a b c and b c d for relevance. Discard any that are not sufficiently relevant.
6) Try to merge smaller expressions into larger and more meaningful ones. For example, a b c d and b c d may merge to form a b c d.

2. Similarity computation

Once the appropriate thesaurus vocabulary has been identified, and phrases have been designed if necessary, the next step is to determine the statistical similarity between pairs of terms. It includes two similarity routines:

- **Cosine:** This computes the number of documents associated with both terms divided by the square root of the product of the number of documents associated with the first term and the number of documents associated with the second
- **Dice:** This computes the number of documents associated with both terms divided by the sum of the number of documents associated with one term and the number associated with the other.

3. Vocabulary Organization

Once the statistical term similarities have been computed, the last step is to impose some structure on the vocabulary which usually means a hierarchical arrangement of term classes. For this, any appropriate clustering program can be used. A standard clustering algorithm generally accepts all pair wise similarity values corresponding to a collection of objects and uses these similarity values to partition the objects into clusters or classes such that objects within a cluster are more similar than objects in different clusters. Some clustering algorithms can also generate hierarchies.

MERGING EXISTING THESAURI

- This second approach is appropriate when two or more thesauri for a given subject exist that need to be merged into a single unit.
- If a new database can indeed be served by merging two or more existing thesauri, then a merger perhaps is likely to be more efficient than producing the thesaurus from scratch.
- This approach has been discussed at some length in Forsyth and Rada (1986). The challenge is that the merger should not violate the integrity of any component thesaurus. Rada has experimented with augmenting the MESH thesaurus with selected terms from SNOMED (Forsyth and Rada 1986, 216).
- MESH stands for Medical Subject Headings and is the thesaurus used in MEDLINE, a medical document retrieval system, constructed and maintained by the National Library of Medicine.
- It provides a sophisticated controlled vocabulary for indexing and accessing medical documents. SNOMED, which stands for Systematized Nomenclature of Medicine, is a detailed thesaurus developed by the College of American Pathologists for use in hospital records.
- MESH terms are used to describe documents, while SNOMED terms are for describing patients. Ideally, a patient can be completely described by choosing one or more terms from each of several categories in SNOMED.
- Both MESH and SNOMED follow a hierarchical structure. Rada's focus in his experiments has been on developing suitable algorithms for merging related but separate thesauri such as MESH and SNOMED and also in evaluating the end products.