Automated Construction Of Domain Ontologies From Lecture Notes

Neelamadhav Gantayat

under the guidance of

Prof. Sridhar Iyer

Department of Computer Science and Engineering,
Indian Institute of Technology, Bombay
Powai, Mumbai - 400 076

June 28, 2011

Outline

Motivation

- Courseware Repositories
  - MIT's OCW[1]
  - NPTEL [2]
  - CDEEP[3]
- Searching in Repositories

[1] http://ocw.mit.edu/
[2] http://www.nptel.iitm.ac.in/
[3] http://www.cdeep.iitb.ac.in/

Repositories

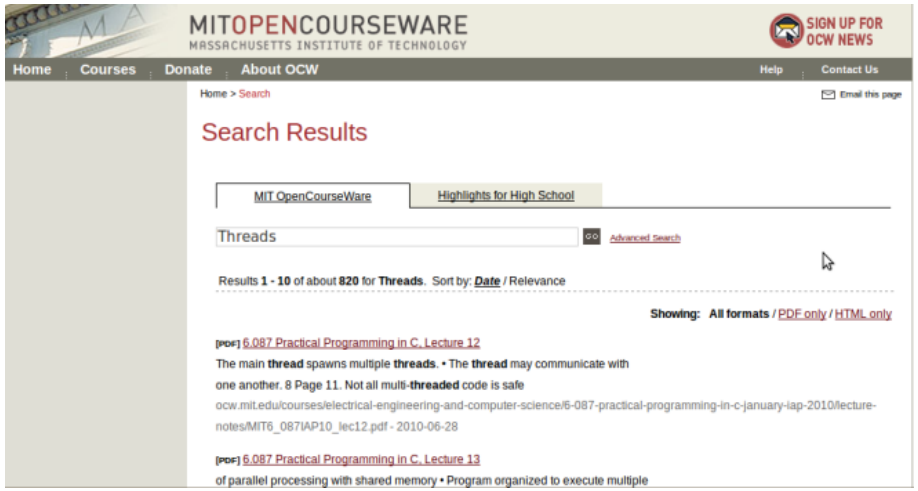


Figure: MIT's OCW search for "Threads"

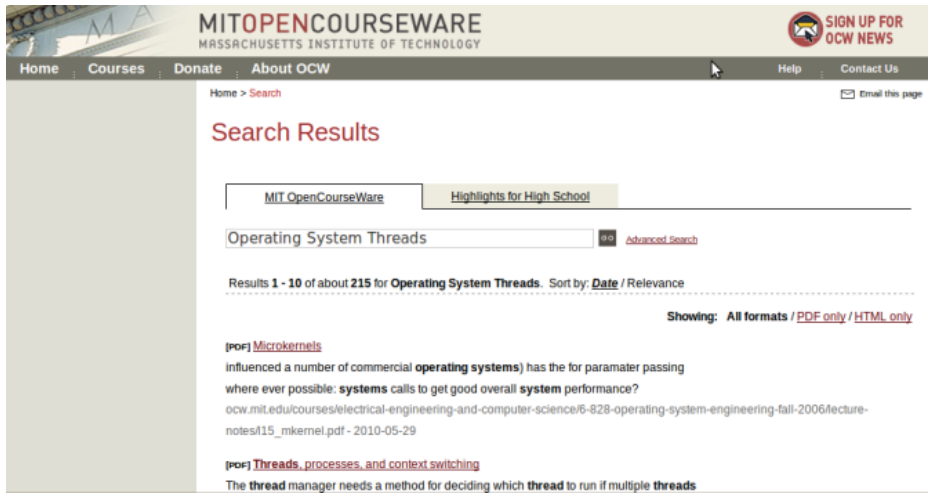


Figure: MIT's OCW search for "Operating system Threads"

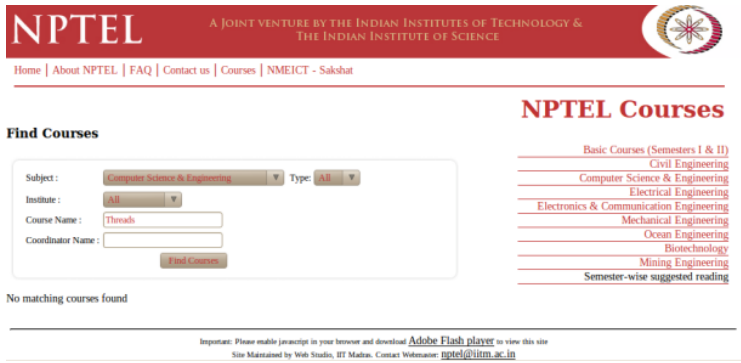


Figure: NPTEL search for "Threads" under course OS

Searching Tools

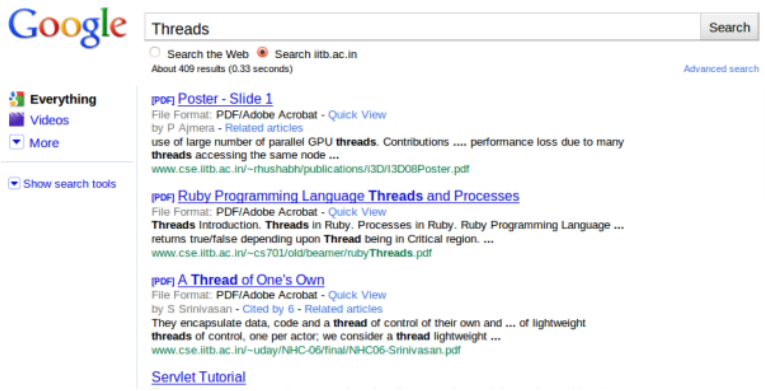


Figure: Google tool search for "Threads"

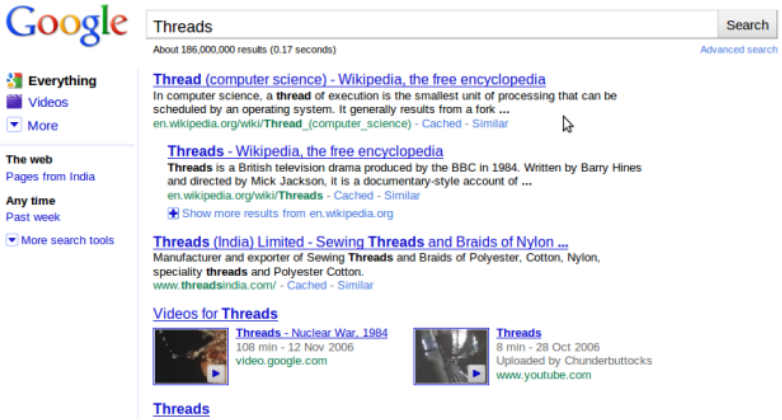


Figure: simple google search for "Threads"

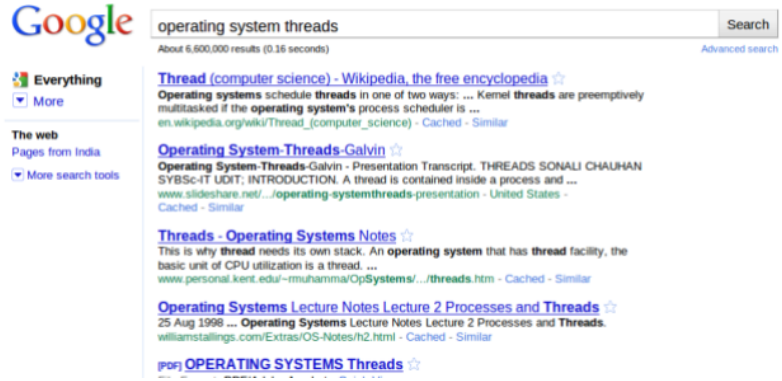


Figure: Advanced google search for "Operating system Threads"

Outline

Problem Statement

Given a set of lecture notes (pdf files) or PDF of a textbook from a course-ware repository,

- Provide user with the reading material, suggest some Basic and Advanced Topics.

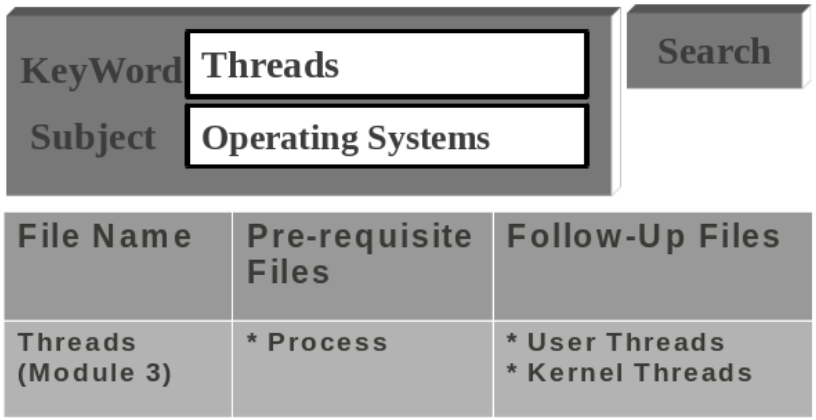


Figure: Expected System

Problem Statement

- Dependency graph (Ontology) for a course.

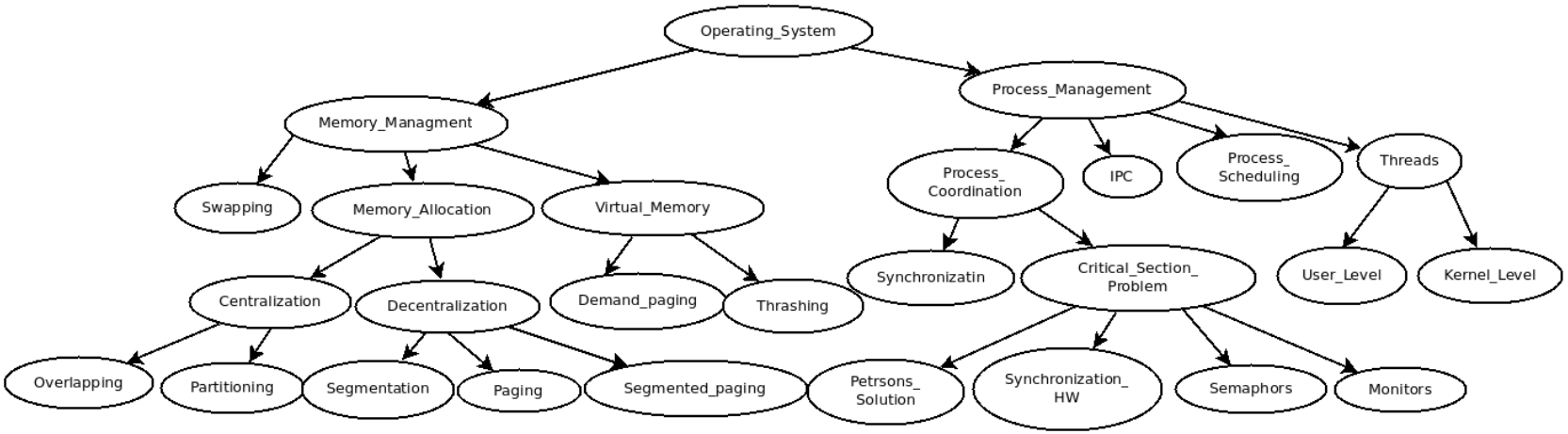


Figure: Dependency Graph (Ontology) for Operating System

Outline

Ontology

It is borrowed from philosophy - the study of "The nature of being".

It "consists of concepts, hierarchical organization of concepts".

Domain Ontology

- Model which provides definitions and relationships of the concepts, and major theories, principles and activities in the domain.
- Domain ontologies provide shared and common understanding of a specific domain.

Applications of Ontology

- **Knowledge management:** Acquiring, maintaining, and accessing an organization's data.
  - "What is the birthplace of Gandhi?"
- **Web commerce:** On-line market places and auction houses.
- **E-learning:** Dependencies between the keywords of a topic in the repository.

Outline

Ontology for Operating System

- **Scope and Domain:** To find out the dependencies between the course ware repositories for operating system.
- **Reuse existing Ontology**
  - `www.ksl.stanford.edu/software/ontolingua`
  - `www.daml.org/ontologies/`
  - `www.unspc.org`, `www.dmoz.org`
  - `www.roselternet.org`
- **Important Keywords:**

Table: Keywords

Types of computing	Types of Systems
Memory Management	Process Management
Secondary Management	File Management
Memory Allocation	Virtual Memory
Disk Scheduling	Threads

Ontology Development for Operating System

- **Identify the classes:**

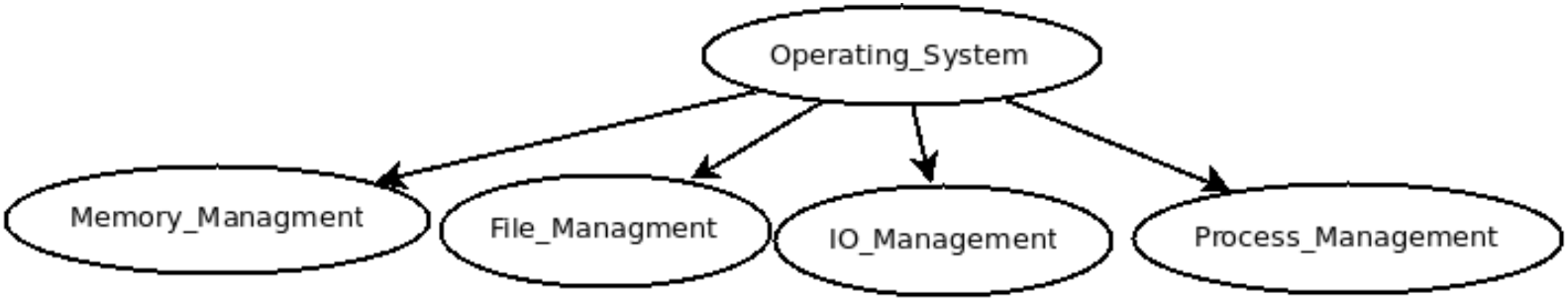


Figure: Classes

- **Define Properties:**

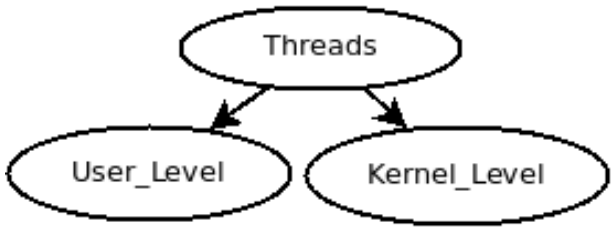


Figure: Types of Thread

Final Ontology using DOT[4] Language

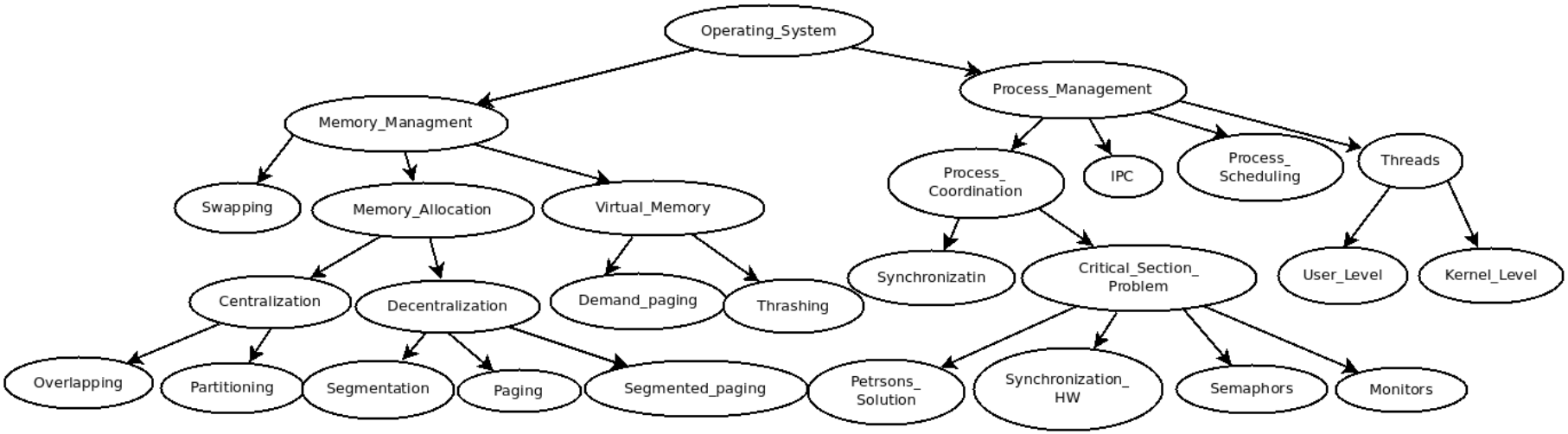


Figure: Ontology for Operating System

[4] http://en.wikipedia.org/wiki/DOT_language

Protégé[5] Ontology

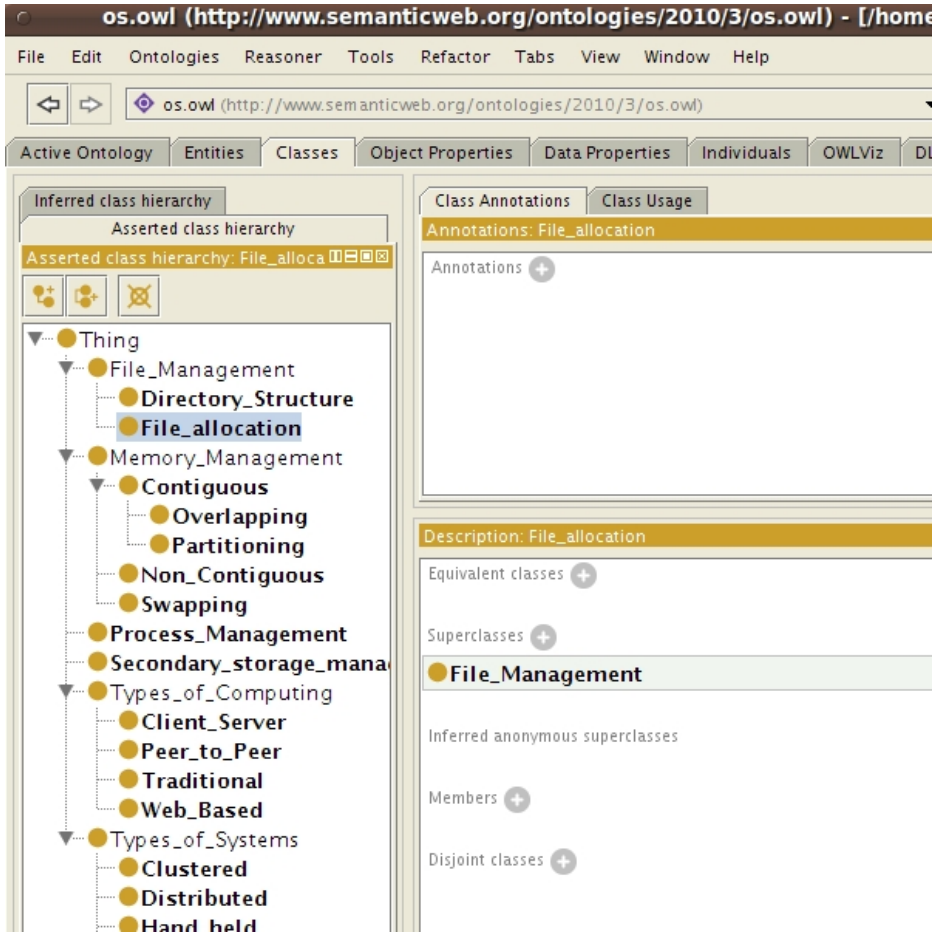


Figure: Protégé Ontology

[5] http://protege.stanford.edu/

Outline

Difficulties with current systems[1]

- Requirement of an Expert.
- Manual processing of the data
- Markup languages and code fragments
- Assumption of More general Ontology.
- Availability of WordNet
- How terms are extracted from Text?

Outline

System - 1

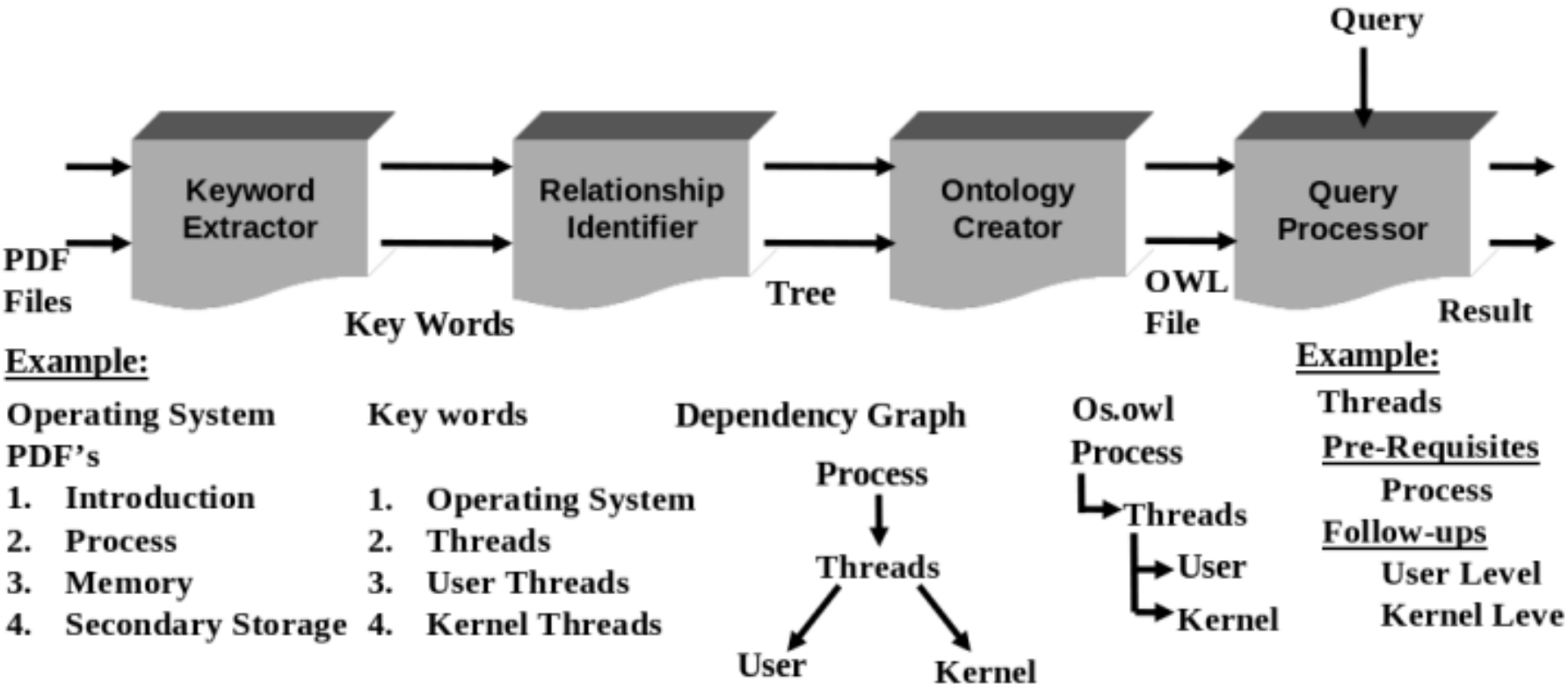
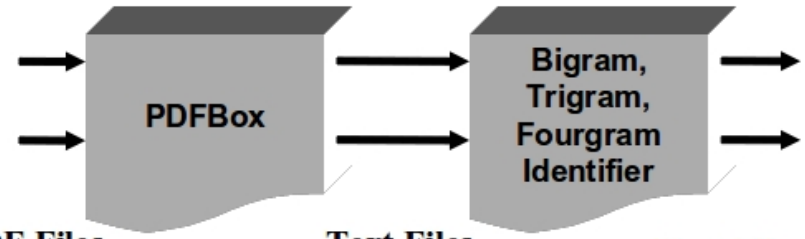


Figure: System overview of System - 1

"ngram" Identification

Example:
Operating System



PDF's
1. M1L1.pdf
2. M2L1.pdf
3. M3L1.pdf

Text
1. M1L1.txt
2. M2L1.txt
3. M3L1.txt

For
1. Bigrams
2. Trigrams
3. Fourgrams

M1L1.pdf

The central part of the computer system is a processing engine called CPU.

M1L1.txt

The central part of the computer system is a processing engine called CPU.

Bigram

Computer_system processing_ engine Engine_called Called_cpu.

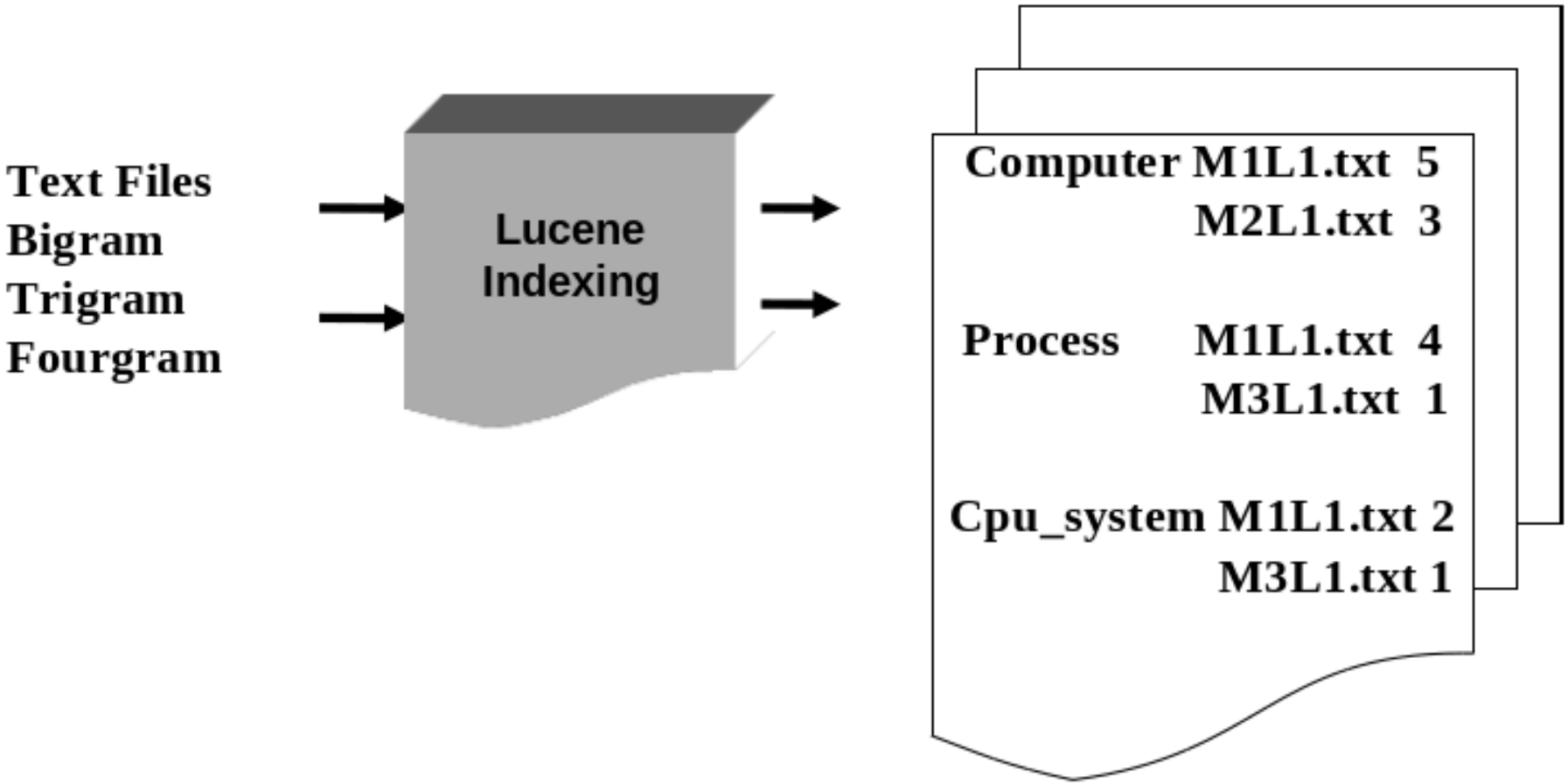
Trigram

Service_the_central Processing_engine _called Engine_called_cpu

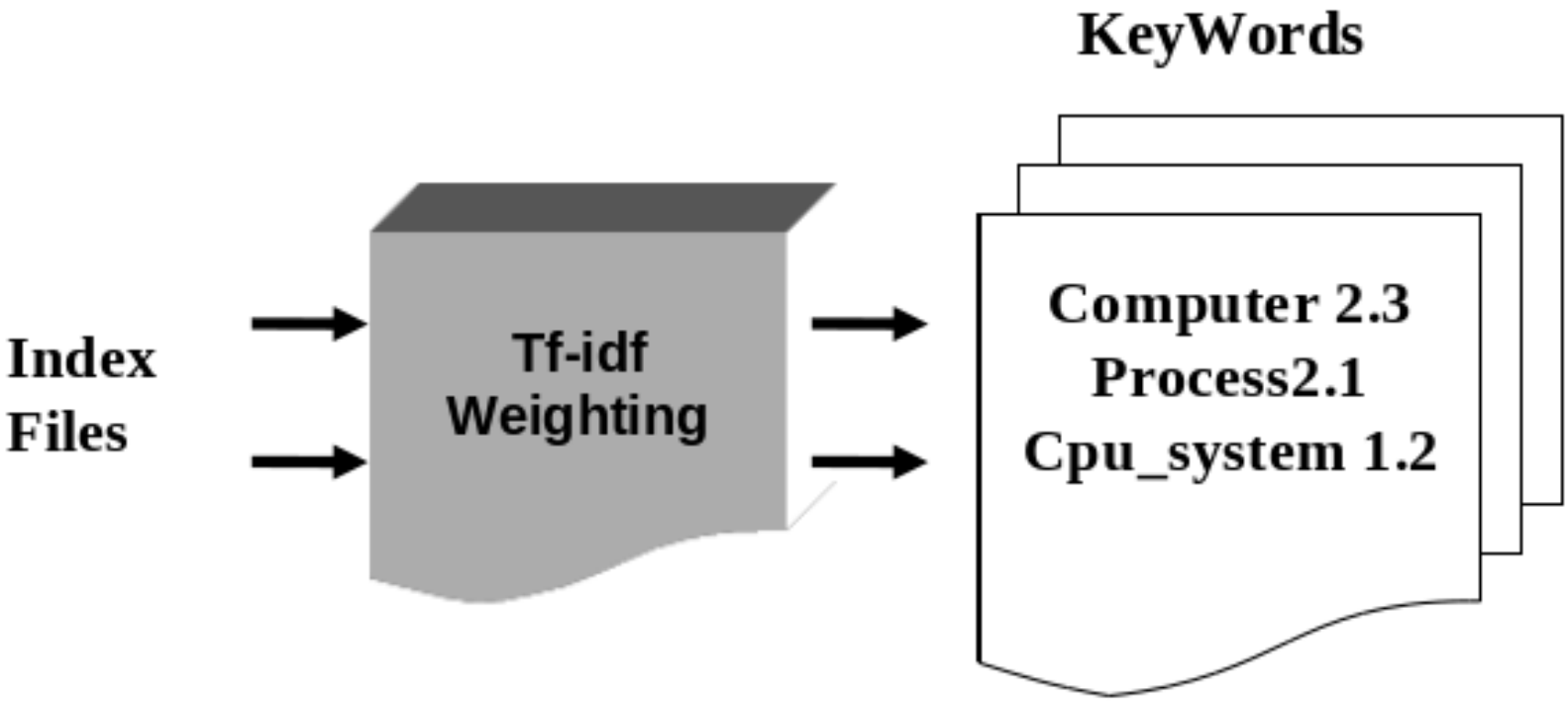
Fourgram

Part_of_the _computer Processing_engine _called_cpu

Indexing


Text Files
Bigram
Trigram
Fourgram
Lucene
Indexing
Computer M1L1.txt 5
M2L1.txt 3
Process M1L1.txt 4
M3L1.txt 1
Cpu_system M1L1.txt 2
M3L1.txt 1

Keyword Extraction



Tf-Idf[6]

Given a document collection D, a word w, and an individual document $d \epsilon D$

$$w_d = f_{w,d} * log(|D|/f_{w,D})$$

$$Tf - Idfweight = \sum_{d=1}^{|D|} w_d$$

where

- $f_{w,d}$ Number of times w appears in the current document d,
- $|D|$ Number of documents, and
- $f_{w,D}$ Number of documents in which w appears

[6] http://en.wikipedia.org/wiki/Tf%E2%80%93idf

Importance of Tf-Idf

- Highest when w occurs many times within a small number of documents;
- Lower when the term occurs fewer times in a document, or occurs in many documents;
- Lowest when the term occurs in virtually all documents.

Apriori Algorithm[5]

Apriori Algorithm is an algorithm for finding association rules.

1. Find Keywords (Tf-idf weights)
2. Find out the *frequent wordsets* with the given *support* and *confidence*, for all pairs of keywords.

Terminology

Association Rule $i \rightarrow j$ means "if a document contains i then it is likely to contain j".

Support The number of documents containing the words in w.

Confidence of this association rule is the probability of j given i.

Ontology for CN

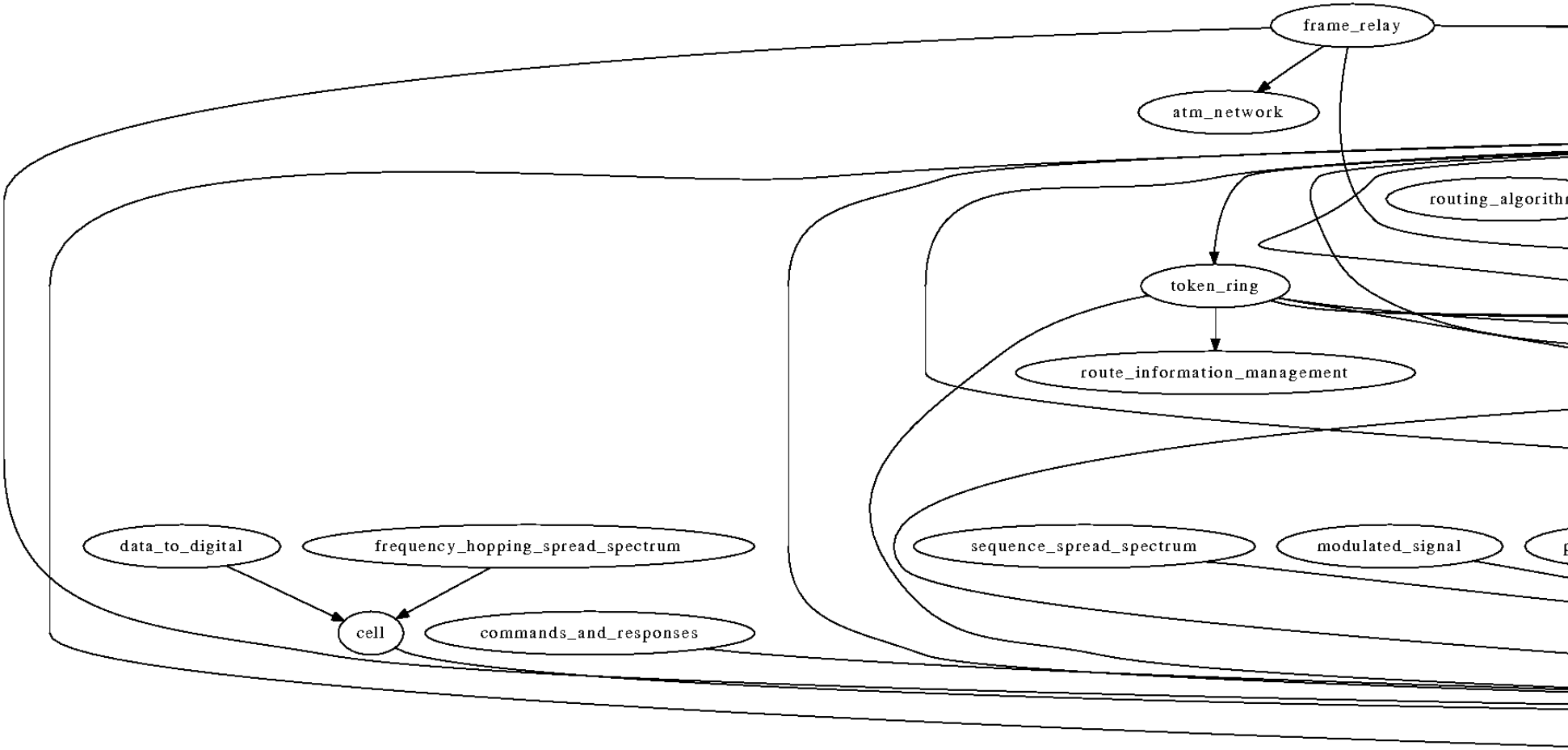


Figure: System output of Computer Networks

System -2

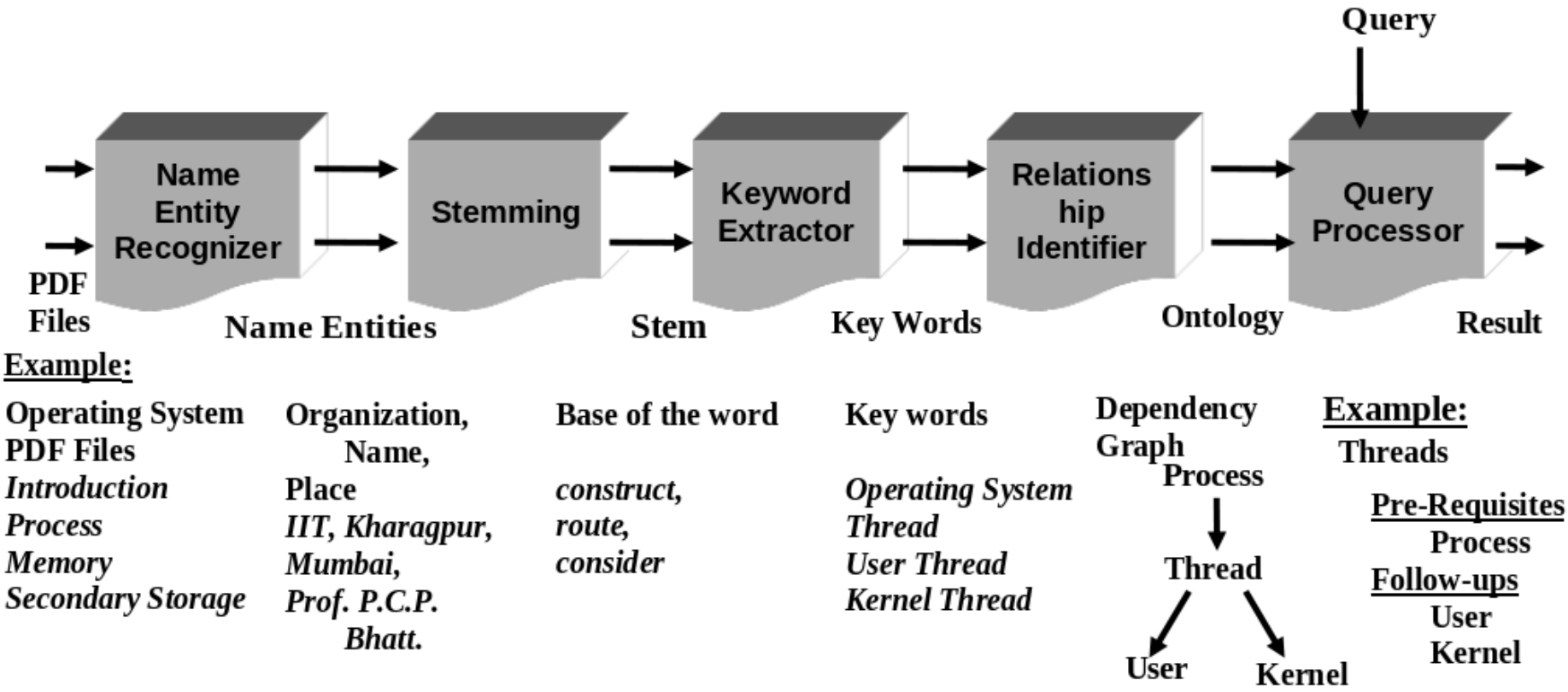


Figure: System overview of System - 2

Name Entity Recognition

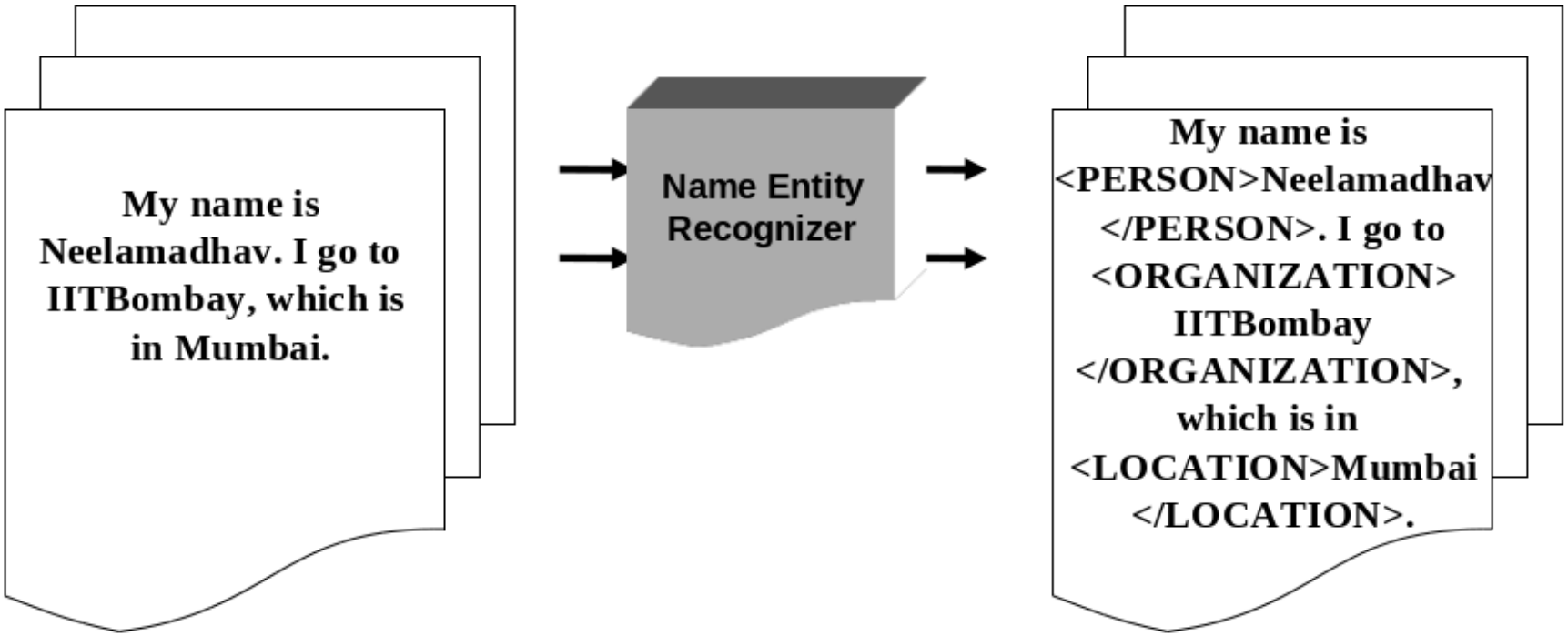


Figure: Name entity recognition

CRF-based NER [6]

We used Stanford NER mainly because of the following reasons

- The NER is trained on CoNLL, MUC and ACE English training data
- By default it recognizes the entities: Person, Location, Organization, Which we need for our experiment
- Finally the NER is trained on both British and American newswire, so robust across both domains

Stemming [7]

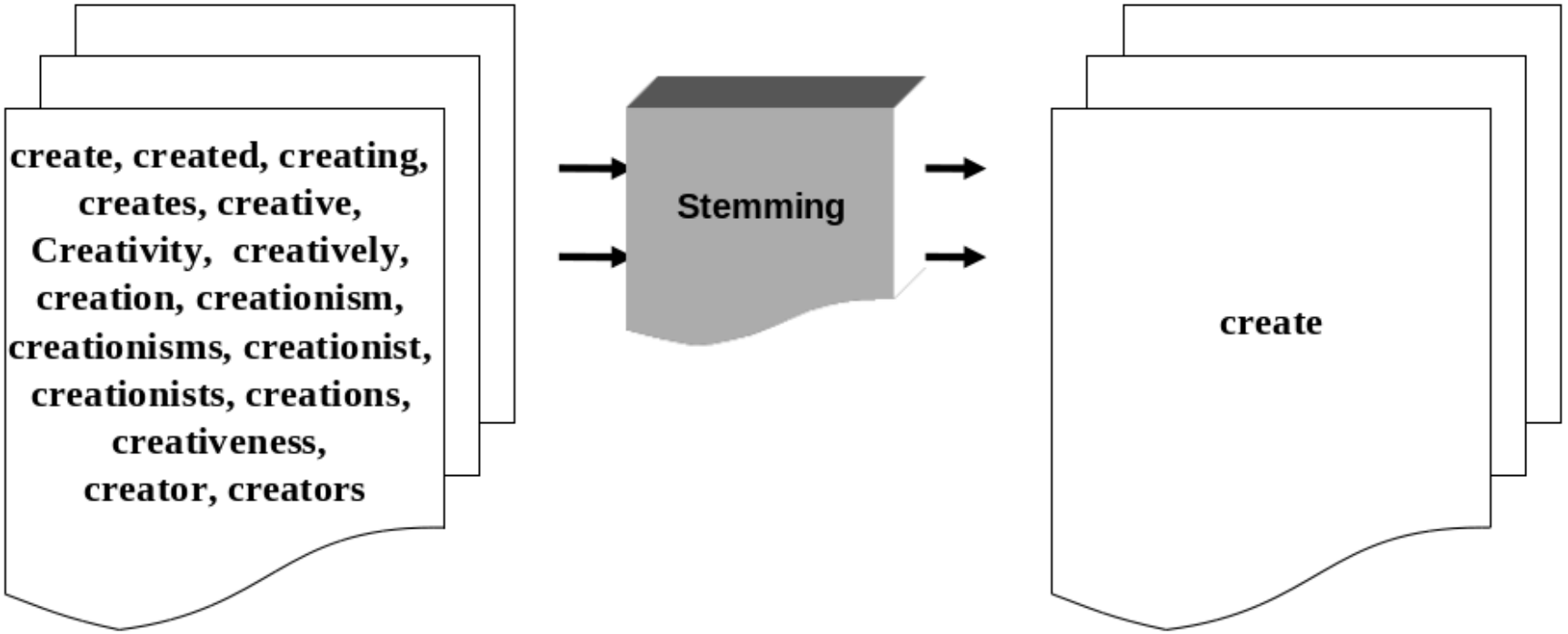


Figure: Stemming

Ontology Development

- We modified our Apriori algorithm in order to get good results.
- We considered a line as a dataset instead of the whole document.

Outline

Recall & Precision

We have compared results generated by our system with those of the expert generated results.

Recall (R) Ratio of the relevant results retrieved to the results suggested by the expert.

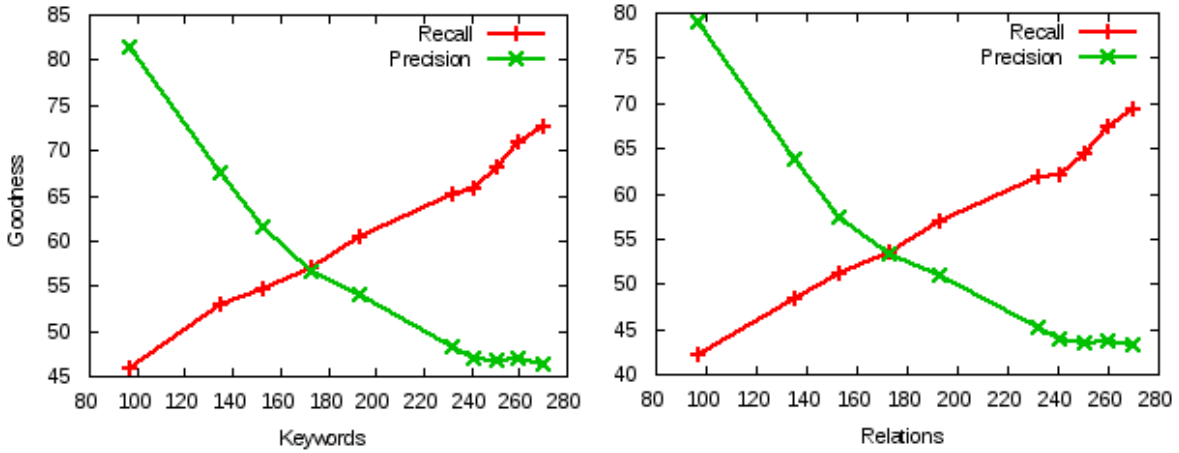
$$R_w = \frac{W_c}{W_e} * 100\% \qquad R_r = \frac{R_c}{R_e} * 100\%$$

Precision (P) Ratio of the relevant results retrieved to the total results identified by the system.

$$P_{pw} = \frac{W_c}{W_s} * 100\% \qquad P_r = \frac{R_c}{R_s} * 100\%$$

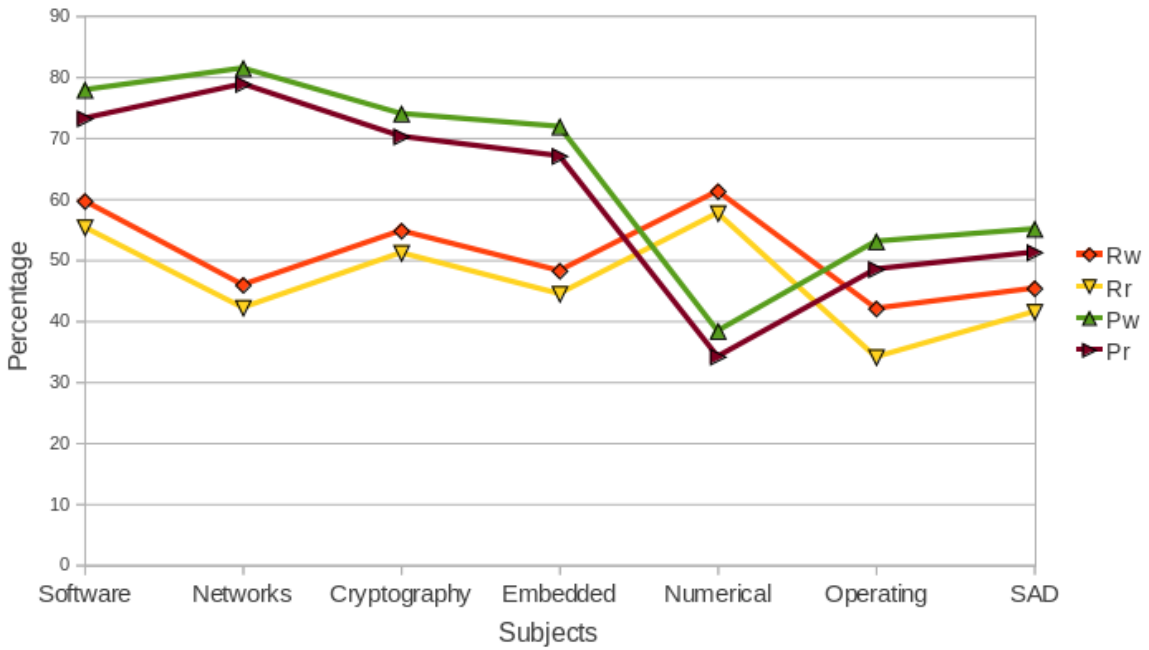
Results of Computer Networks

keywords	Recall(%)		Precision(%)	
	R_w	R_r	P_w	P_r
97	45.93	42.19	81.44	78.86
135	52.91	48.26	67.41	63.84
153	54.65	51.09	61.44	57.32
173	56.98	53.46	56.65	53.21
193	60.47	56.94	53.89	50.82
232	65.12	61.77	48.28	45.19
241	65.70	61.98	46.89	43.88
251	68.02	64.28	46.61	43.43
260	70.93	67.35	46.92	43.56
270	72.67	69.33	46.30	43.16



Recall & Precision for all the subjects

Subjects	keywords	Recall(%)		Precision(%)	
		R_w	R_r	P_w	P_r
Software	95	59.68	55.37	77.89	73.22
Networks	97	45.93	42.19	81.44	78.86
Cryptography	100	54.81	51.19	74	70.26
Embedded	96	48.25	44.47	71.88	67.07
Numerical	99	61.29	57.69	38.38	34.18
Operating	98	42.06	34.1	53.06	48.52
SAD	98	45.38	41.52	55.1	51.28



Confusion Matrix

		System Results: Positive	System Results: Negative
Expert Results	True	True Positive	False Negative
	False	False Positive	True Negative

Where

True Positive: Number of correct results that were correctly identified,

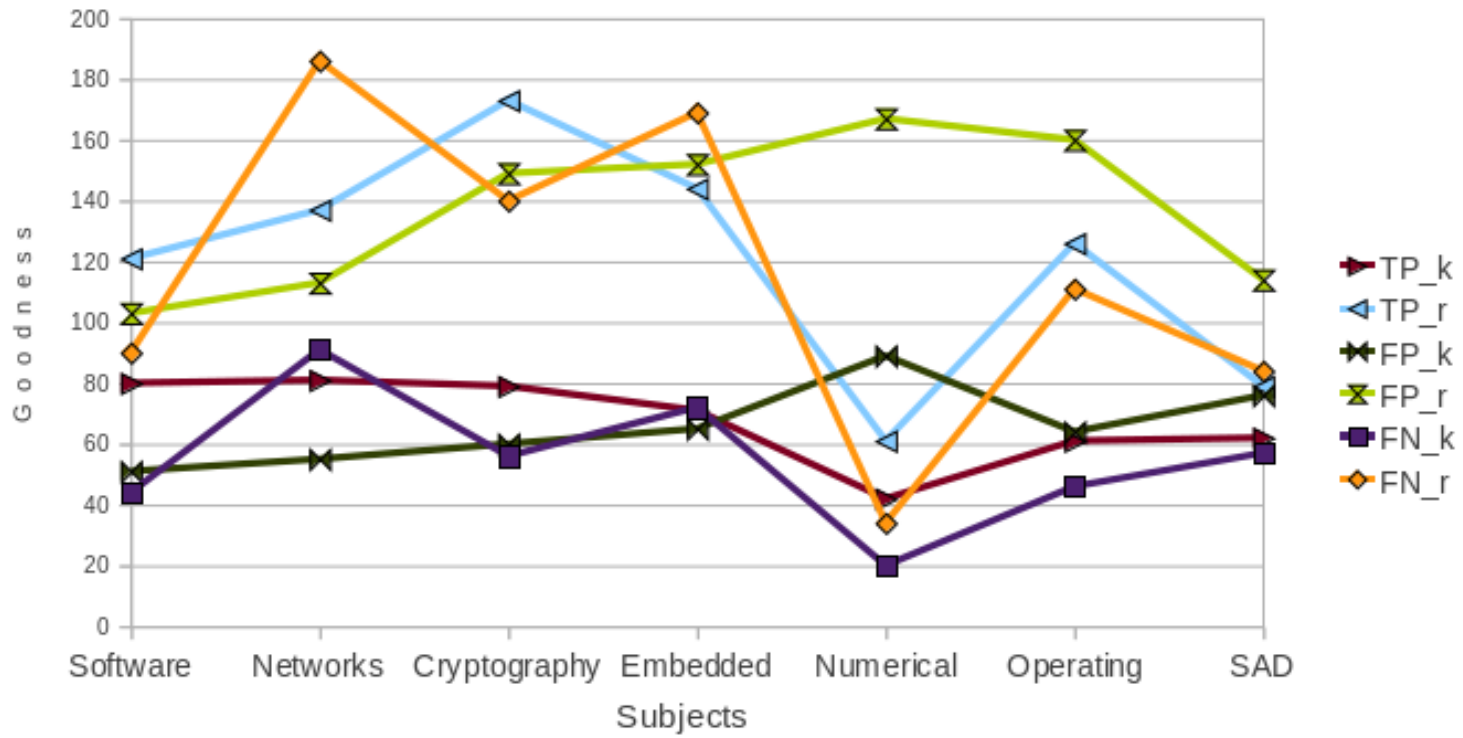
False Positive: Number of incorrect results that were incorrectly classified as positive,

True Negative: Number incorrect results that were identified as negative,

False Negative: Number of correct results that were incorrectly classified as negative.

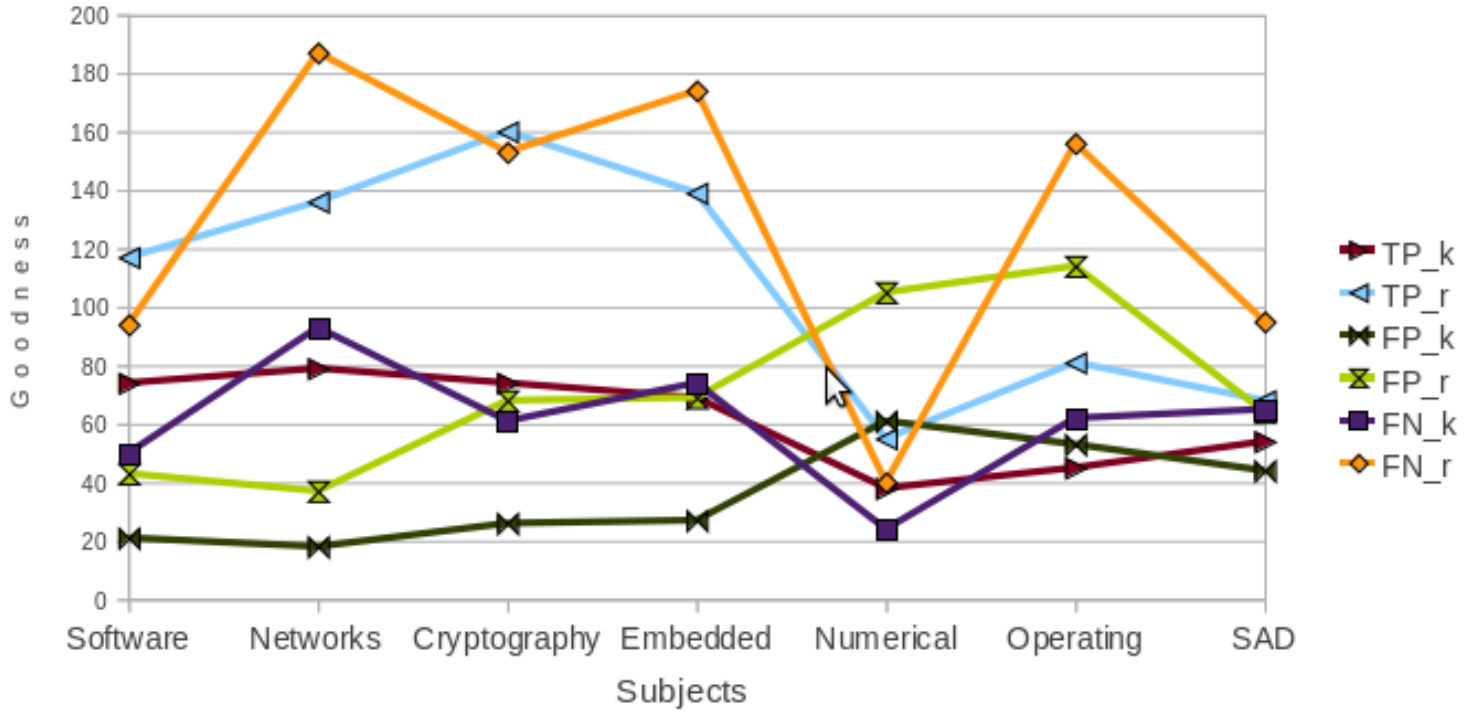
Confusion Matrix for System-1

Subject	System Generated		Expert identified		True Positive		False Positive		False Negative	
	Keywords	Relations	Keywords	Relations	Keywords	Relations	Keywords	Relations	Keywords	Relations
Software	131	224	124	211	80	121	51	103	44	90
Networks	136	250	172	323	81	137	55	113	91	186
Cryptography	139	322	135	313	79	173	60	149	56	140
Embedded	136	296	143	313	71	144	65	152	72	169
Numerical	131	228	62	95	42	61	89	167	20	34
Operating	125	286	107	237	61	126	64	160	46	111
SAD	138	193	119	163	62	79	76	114	57	84



Confusion Matrix for System-2

Subject	System Generated		Expert identified		True Positive		False Positive		False Negative	
	Keywords	Relations	Keywords	Relations	Keywords	Relations	Keywords	Relations	Keywords	Relations
Software	95	160	124	211	74	117	21	43	50	94
Networks	97	173	172	323	79	136	18	37	93	187
Cryptography	100	228	135	313	74	160	26	68	61	153
Embedded	96	208	143	313	69	139	27	69	74	174
Numerical	99	160	62	95	38	55	61	105	24	40
Operating	98	195	107	237	45	81	53	114	62	156
SAD	98	132	119	163	54	68	44	64	65	95



System-1 Vs. System-2

- *System-1* generated more number of `False Positives` than *System-2*
- Processing time of *System-2* is lesser than *System-1*
- Named Entity Recognition

Outline

Future Work

- We observed that `Recall` was maximum when the number of keywords was nearly equivalent to 95.
- The optimum value of `Recall` was obtained when the number of unigrams was 40, number of bigrams was 30, number of trigrams was 20, and number of fourgrams was 10.
- Efficiency of the system is mainly dependent on number of PDF files in case of multiple PDF files.
- And dependent on number of pages in case of a single pdf file.
- Relationship identification and keyword extraction algorithms can be modified for better results.

Outline

References

[1] Ivan Bedini and Benjamin Nguyen. *"Automatic ontology generation: State of the art."* In PRiSM Laboratory Technical Report. University of Versailles, 2007.

[2] Natalya F. Noy and Deborah L. McGuinness, *"Ontology Development 101: A Guide to Creating Your First Ontology"*, Available from Internet:<http://protege.standford.edu/publications/ontology_development/ontolgogy101.html>

[3] Oscar Corcho, Mariano Fernández-López, Asunción Gómez-Pérez, *"Methodologies, tools and languages for building ontologies. Where is their meeting point"*, Data & Knowledge Engineering 46(2003) 41-64.

[4] Deborah L. McGuinness, Frank van Harmelen *"OWL Web Ontology Language Overview"*, http://www.w3.org/TR/owl-features/, W3C Recommendation 10 February 2004.

[5] Rakesh Agrawal and Ramakrishnan Srikant. *"Fast algorithms for mining association rules in large databases."* In Proceedings of the 20th International Conference on Very Large Data Bases, VLDB '94, pages 487-499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[6] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. *"Incorporating non-local information into information extraction systems by gibbs sampling."* In ACL, pages 363-370, 2005.

[7] Martin Porter. *"An algorithm for suffix stripping."* In Workshop on Multimedia Information Systems, 1980.