

# Knowledge Representation Issues in Information Extraction

Wee Li Kwang Angela<sup>1</sup>, Tong Loong Cheong<sup>1</sup>, Tan Chew Lim<sup>2</sup>

<sup>1</sup>Kent Ridge Digital Labs, 21 Heng Mui Keng Terrace, Singapore 119613  
{angela, tonglc}@krdl.org.sg

<sup>2</sup>School of Computing, National University of Singapore, Kent Ridge, Singapore 119260  
isctancl@nus.edu.sg

**Abstract.** The advent of computing has exacerbated the problem of overwhelming information. Advanced information management strategies such as Information Extraction, Information Filtering, Information Retrieval, and Text Categorization are becoming important to manage the deluge of information. Information Extraction (IE) systems can be used to automatically extract relevant information from free-form text for update to databases or for report generation. This paper describes the major challenge of knowledge representation issues in an information extraction task – representing the meaning of the input text, the knowledge of the field of application (or domain application) and the knowledge about the target information to be extracted. In this research, we have chosen a directed graph structure to represent the input text meaning, a domain ontology to represent the domain application and a frame representation to capture the target information to be extracted. We discuss in this paper how these knowledge structures interplay to perform the task of information extraction.

## 1. Introduction

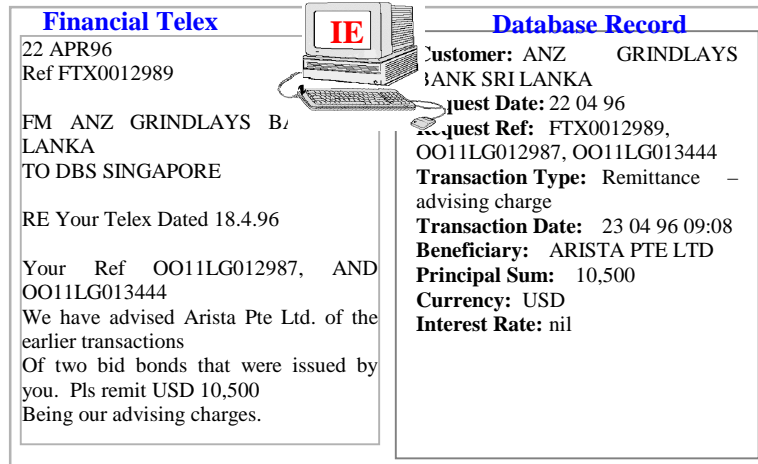
Information is pertinent in today's society. Having the right information at the right time is critical for decision-making. But while information is a valuable resource, overwhelming information can pose a serious problem of time wastage in sorting through the disorganized textual mass for the relevant information.

This information deluge will escalate further with the advent of computers and Internet. It is vital to leverage on advanced information management technology aids for a fast way of sieving through these texts for meaningful information. The most common information management strategies are Text Categorization [13,21,25,27], Information Filtering [6,20], Information Extraction [6,7,8,9,12,15,19], Text Summarization [22,23] and Personalized Information Dissemination [20].

*Information Extraction* is the extraction of relevant information from free-form text (or unrestricted text) based on a set of pre-defined template of what information is to be extracted (or what is relevant for extraction). The template specification constrains the relevancy of the information to be extracted, which could be about pre-specified types of events, entities or relationships. For example, in a financial transaction, an information extraction system could extract the transaction type, date, customer, principal, currency, interest rate, and so on.

The extracted information is usually formatted as a database record suitable for subsequent processing, which may include database update for on-line access or data trend analysis, abstract, summary or report generation, text categorization, and message routing. Fig 1 shows extraction of information from a financial telex to update fields within a database record.

Practical Natural Language Processing applications in IE are exemplified by work on the Tapestry projects [15,19,22] and by publications in the Message Understanding Conferences (MUC) [6,7,8,9].



**Fig 1. Information Extraction (IE) for Database Updating**

A wide range of techniques for IE have been reported, ranging from statistics [14] and lexical pattern matching [1,16,21] to full text understanding [11,15]. Whichever techniques used, the issues of representing knowledge required in IE have to be addressed.

This paper will first present the knowledge representation issues and their proposed knowledge structures in detail. After which, it describes how these knowledge structures interplay to perform the task of information extraction.

## 2. Knowledge Representation Issues

One of the major challenges in Information Extraction is how to represent the 3 different types of domain knowledge that is required in this task, namely:

- a) the meaning of the input text,
- b) the knowledge of the field of application (domain application),
- c) the target information that is to be extracted

### 2.1 Message Intermediate Representation (MIR)

Traditionally, input text is represented as syntax trees where the branching of the trees is governed by the syntax rules of the language grammar. Such syntax representation has an obvious weakness in its syntax-dependency, making it difficult to handle flexibility of language expressions.

The following sentences, S1 to S5, have different grammatical constructs and hence have different syntax tree structures.

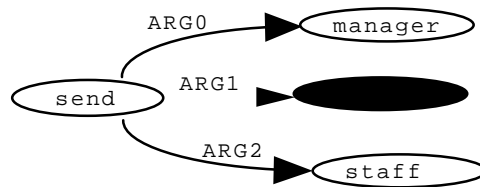
- S1 - "The manager sent the document to the staff."
- S2 - "The manager sent the staff the document."
- S3 - "The document was sent to the staff by the manager."
- S4 - "The staff was sent the document by the manager."
- S5 - "It was the manager who sent the document to the staff."

These sentences are semantically equivalent and state the same facts. The variants of syntax structures made extraction of information difficult. In order to identify the person who sent the document, the variants of the grammatical constructs have to be considered. In some cases, the "manager" appears at the front and sometimes, it appears at the back of the tree structure. To simplify this process of identifying the required concept for information extraction, there needs to be a less syntax-dependent way of representing the input

text that is also able to express the meaning of the text. Here, we have chosen semantic graph structures to represent the text meaning. We refer to this semantic graph structures as *Message Intermediate Representation (MIR)*.

The MIR is a directed acyclic labeled graph with nodes that represent the concepts that occur in the text. Concepts could range from simple keywords, terms, and phrases to more complex patterns, which represent real-world entities that have been discovered during the text analysis. These concepts could be events, activities, transactions, attributes, circumstances, facts, incidents, milestones, occasions, outcome, etc.. The nodes of the graphs (referred to as MIR-nodes) are inter-connected through labeled directed links (referred to as MIR-links), that represent the relationships between these concepts. The MIR is a more neutral representation, independent of syntax or linear ordering, more meaningful semantic representation (sometimes referred to as logical form or deep structures) yet preserving syntactic information for reconstruction back to its syntactic tree structure.

The sentences S1-S5 have different syntax tree structures but will have the same MIR structure as shown in Fig 2. The main concepts that appear in the sentences are “manager”, “send”, “document”, and “staff” and they are represented as nodes of the graph structure. The relationships between these concepts are “ARG0”, “ARG1”, and “ARG2”. With the MIR, identifying the person who sent the document becomes much easier



**Fig 2. Message Intermediate Representation (MIR)**

with a single representation.

The designed and implemented MIR is very similar to the PEGASUS semantic representation [10] which also provides a definitive move from syntax to semantics, based on the definition of case frames or thematic roles (or predicate-argument relations). The PEGASUS structure is also a labeled, directed graph. The MIR and PEGASUS differ in argument assignments from the identification of meaningful relationships between head words of phrases and their modifiers or adjuncts. Hence, both representations also differ in the concepts and the set of relations. PEGASUS is similar to MIR in adopting the notion of “deep” cases or functional roles. For example, deep subject of ARG0 for MIR and DSUB for PEGASUS; ARG1 for MIR and DOBJ for PEGASUS.

The MIR is also similar to Schank’s conceptual dependency (CD) notion [17]. However there are subtle differences. The basic entity in a CD is the event, or conceptualization whereas a basic entity in a MIR is the node, which could be an event or an entity. The CD has only a limited number of cases such as ACTOR which is similar to ARG0 of MIR. It also has a limited number of action types or primitives based on the abstract notion of transfer: ATRANS, abstract transfer (such as of possession of an object); PTRANS, physical transfer; and MTRANS, mental transfer (such as talking). It also includes primitive actions of bodily activity such as PROPEL (apply force), MOVE (move a body part), GRASP, INGEST, and EXPEL, as well as a few mental actions such as CONC (conceptualize or think) and MBUILD (perform inference).

## 2.2 Domain Ontology

The second important piece of knowledge required in an Information Extraction system is the knowledge of the domain application, i.e. the domain ontology. A *domain ontology* defines the set of basic terms comprising the vocabulary of the domain area and the relationships that bind these terms. For example, in an automobile domain, the set of vocabulary includes motorcar, motorist, engine, chassis, body, wheel, steering, light, brake, fuel system, gearbox, wiper, etc. These terms have some relationships to one another,

example, a motorist is a person operating a motorcar; the glove compartment and the boot are reserved spaces in an automobile for storage of things.

In this research, for pragmatic reason, we have selected to model a single restrictive domain, using a simple yet process-able ontology. The domain knowledge is organized in object-centered hierarchies with inheritance, with the objects representing the terms and concepts in the domain, and the relationships of these concepts through the hierarchy. In this hierarchy, the subordinate concepts, in addition to having their own attributes, will inherit characteristics and features from the super-ordinate concepts. However, if the subordinate concept has an attribute that has a different value from that inherited from its super-ordinate parent, its attribute value takes precedence.

Fig 3 shows an ontology on the concept of delivery mode of documents. The term “delivery-mode” comprises of all the various concepts which are lower level in the hierarchy, such as, mail, airmail, surface mail, registered mail, express mail, courier, Federal Express, DHL, local urgent mail, and LUM. All these subordinate concepts will inherit all characteristics of the term “delivery-mode”.

Such an ontology facilitates reasoning. For example, the system through its inference reasoning “understands” that the concept “DHL” is a kind “courier”, which is in turn a kind of “express mail” of “mail”, and a type of “delivery-mode”. From the larger ontology that encompasses this ontology, when “DHL” is being used in a different sense, such as a verb in “Pls DHL the documents to the issuing bank”, the concept of “send” is implied.

This ontology representation also facilitates the knowledge specification of the target information to be extracted. As super-ordinate concepts of a concept could be automatically inferred from the ontology, there is no need to specify all possible concepts in the knowledge specification of the target information to be

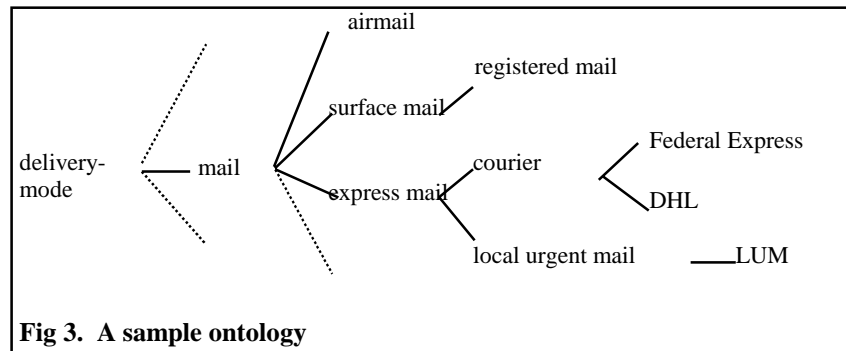


Fig 3. A sample ontology

extracted.

### 2.3 Frame for Extracting Information from Messages (FEIM)

After having represented the meaning of the input text and the application domain field, the third piece of information that needs to be represented in an Information Extraction application is the target information to be extracted (refer to interchangeably as *target-information*).

In this research, a frame approach to represent the target information to be extracted has been chosen. The frame system is called *Frame for Extracting Information from Messages (FEIM)*.

Frames or templates are common knowledge representation tools in AI applications [2,4,5,13]. However the FEIM system employs templates for the specific purpose of extracting information from input messages. The design of FEIM is very much influenced by the Framekit [13], GUS [3] and KRL [2] frame representations. It has similar basic features such as slot, demon and inference as well as similar operators such as creation, access and update of slot values. In particular, the FEIM structures have been designed to handle practical issues in an Information Extraction application. FEIM provides convenient means for the

knowledge coder (or the user) to specify common phenomena in Information Extraction applications like relationships between target-information.

Knowledge on what constitutes relevant information that is to be extracted from the input text is stored as a set of FEIM specifications. A FEIM specification can be viewed as a template of slots with each slot corresponding to a piece of relevant information that is to be extracted. Associated with each slot is a set of attributes with values of one of the following types: *descriptive* that describes the slot; *status* that contains characteristics of the target-information; *constraint* that spells the type specifications of the slot value; and *demon* that contains the function and macro calls that manipulate the slot value.

The following table summarizes the attributes of the FEIM system.

| Attributes       | Explanations   |
|------------------|--|
| Slot-Name        | Name of the slot   |
| Slot-Description | Description of the slot  |
| Slot-Content     | Content value of the slot  |
| Default          | Default value to Slot-Content  |
| Invalid-Content  | The value of Slot-Content will be transferred here and the Slot-Content will be reset to nil if its value fails the validation test. |
| Discrepancy      | Records the failed validation test   |
| Single           | Determines if the Slot-Content is single-valued or multi-valued  |
| Compulsory       | Determines if the Slot-Content value is optional or compulsory   |
| Validate         | Tests to validate the Slot-Content value   |
| To-Fill          | Demon on how to fill the Slot-Content value  |
| Normalize        | Demon on how to manipulate the Slot-Content value  |
| When-Filled      | Demon on what instructions to perform after the Slot-Content value is determined   |
| Modifiable       | Determines if the Slot-Content value can be manually modified  |

Fig 4 shows a sample of a FEIM slot specification. The FEIM slot named *Dispatch-of-Document-Slot* specifies that the target-information to be extracted could correspond to a MIR subgraph of the input text, if this MIR subgraph is identified and matched by a GML-rule *document-to-send-rule*. GML is a graph manipulation language cum graph pattern matcher developed for matching specified conditions against graph structures such as the MIR structure. The FEIM slot also specifies that the information may be absent in the input text, but a warning is to be reported if the GML rule identifies more than one such information in the input text. Also specified is the relationship between the *Dispatch-of-Document-Slot* and the other slots, *Available-Document-Slot* and *Delivery-Mode-Slot* in the When-Filled demon. The information for these two slots is to be inferred from the extracted information of *Dispatch-of-Document-Slot*.

```

(FEIM-SLOT Despatch-of-Document-Slot
  (content nil)
  (to-fill (gml-rule document-to-send-rule
            (a r1 b r2 c r3 d r4 e)
            ((a despatch)
             (r1 object->)
             (b document)
             (r2 beneficiary->)
             (c (or person organization))
             (r3 means->)
             (d delivery-mode)
             (r4 agent->)
             (e (or person organization))))
            (extract)))
  (when-filled (infer FEIM-SLOT
                    (Available-Document-Slot
                     Delivery-Mode-Slot)))
  optional
  single
  modifiable)

```

**Fig 4. A FEIM Slot**

The design of FEIM has incorporated practical operational requirements of *Inferencing*, *Cross-Checking*, *Discrepancy Highlighting*, and *Change Propagation*.

Cross-checking is a special type of inferencing but with a different purpose. The aim of cross-checking is for verification of consistency across slots. A special slot, known as the *Cross-Checking-Slot* (similar to Inferer slot) is created to verify consistency of one or more of the *To-Be-Cross-Checked-Slots* (similar to Inferer slots). If the consistency check fails, the *Discrepancy* attribute value will be filled. For example, in a loan application domain, a Cross-Checking-Slot “Check-Credit-Limit” is an Inferer slot derives its value of whether credit limit extended to customer is exceeded by cross-checking that the Inferer slot “Credit-Limit-Extended” is less than or equal to 2 times the Inferer slot “Monthly-Salary”. If the cross-checking finds that the check fails, the “Check-Credit-Limit” will have its *Discrepancy* attribute value filled.

This will serve to highlight to the user to do a check on the credit limit that is extended to the customer.

Similar to the Inferer slots, the Cross-Checking-Slot does not have a TO-FILL attribute. Instead, it has a special attribute of *Cross-Check* and the TO-FILL attribute will be automatically generated during compilation of the FEIM specification. The following shows a Cross-Checking-Slot “type-of-tenor” which cross-checks the To-Be-Cross-Checked-Slots “type-of-tenor” and “tenor” through a predicate function “type-tenor-agree-p”.

```

(FEIM-SLOT type-of-tenor
  (cross-check type-tenor-agree-p type-of-tenor tenor))

```

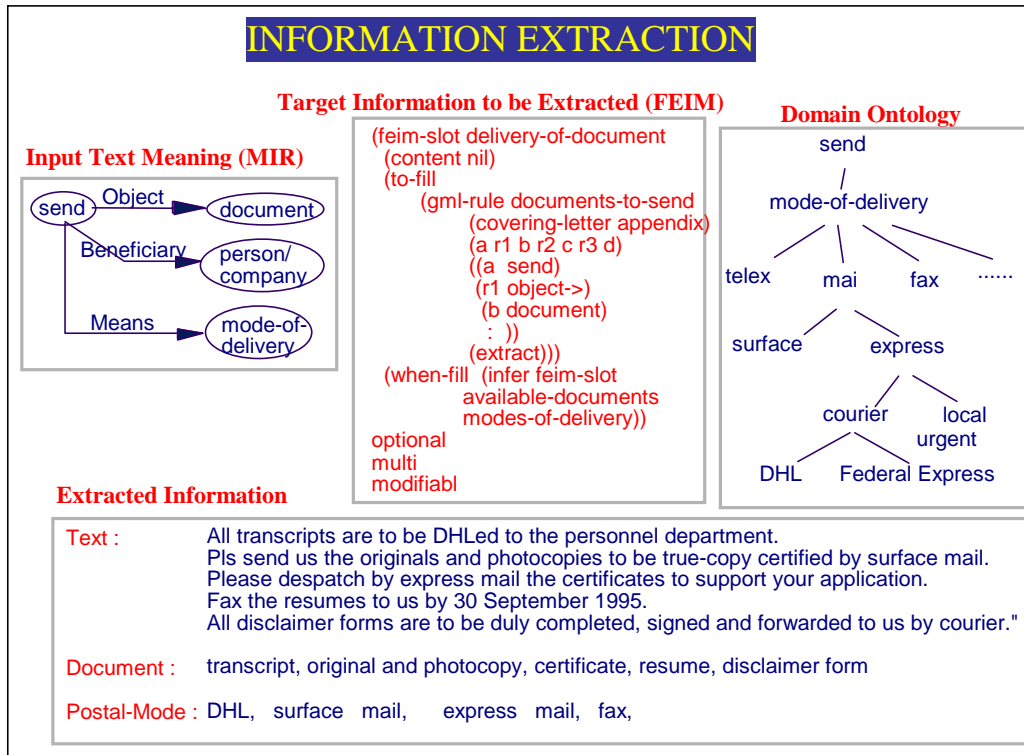
FEIM highlights any discovered discrepancies in the Slot-Content of slots through its *Discrepancy* attribute of the slots. Three types of discrepancies can be discovered through the FEIM system. They are *missing information* (through the Compulsory and Slot-Content attributes), *excess information* (through the Single and Slot-Content attributes) and *wrong information type* (through the Invalid-Content attribute). This discrepancy feature of FEIM conveniently highlights to the user of potential incorrectness in the financial transaction or inaccuracy in the information that has been automatically extracted by the system.

The discrepancy feature enables the user to more quickly detect incorrectness and allows him to make the necessary modifications. Only the slots with the Modifiable attribute can be modified by the user. For ensuring integrity of some Slot-Content of slots, the slots are marked intentionally with unmodifiable

attribute and hence the user will not be able to modify these slots. Most of the Inference slots are intentionally marked unmodifiable as their values are inferred. As FEIM caters for the flexibility of inferencing, once the Slot-Content value of a Inference slot is modified, FEIM has a Change-Propagation mechanism to propagate changes to other Inference slots that are affected by this change. The Change-Propagation mechanism will re-infer and re-validate the values of all the Inference slots. Without this mechanism, it will be difficult to ensure the consistency and integrity of the Slot-Content values of all the Inference and Inference slots through manual changes.

### 3. Interplay of the Knowledge Structures in Information Extraction

Fig 5 shows how the various knowledge structures interplay to perform the task of information extraction. For example, the text “All transcripts are to be DHLed to the personnel department.”, it would be pre-processed and analyzed to a MIR with entities of ‘transcript’, ‘DHL’ and ‘personnel department’.



**Fig 5. Interplay of Knowledge Structures in Information Extraction**

Through the general and domain ontology, it is derived that ‘transcript’ is a type of ‘document’, ‘DHL’ is a mode of ‘send’, and ‘personnel department’ is part of a ‘company’.

When the system attempts to fill up the FEIM slot ‘delivery-of-document’, it searches the MIR through the GML-rule and matches successfully the MIR represented by the text. Hence, the relevant portion of the MIR is extracted. The relevant portion of the MIR to be extracted is determined by the text document layout hierarchical structure. However, the user can overwrite the default by specifying in the <return-function> of the GML-rule to return the entity discovered instead of the determined section, paragraph, or sentence of the text document layout hierarchical structure. For example, if the user is only interested to know the documents that were discussed in the text, he can specify in the GML-rule to return the particular node corresponding to the matched document.

The extracted information can then either be updated into databases, or could be used to generate a summary or for subsequent text processing such as Text Categorization.

These knowledge structures have been implemented in the context of a practical IE Architecture - the Generic Information Extraction (GIE). The GIE Architecture was deployed in a few practical IE systems, including a financial application within one of the largest local banks in Singapore - DBS Bank. The Message Formatting Expert (MFE) system [19] has been operational in the bank since early 1997. It was the joint development of the Kent Ridge Digital Labs and DBS Bank. The MFE system processes some finance package application forms from the customers and automatically extract relevant information as database records. The bank officers verify these extracted database records before final update into the bank's central database. The average recall of extraction is 95% and the average precision of extraction is 83%. In the operational MFE system, total recall is ensured with a "miscellaneous" feim-slot that extracts all unextracted information. This feim-slot allows the bank officer to verify that the unextracted information is irrelevant to the extraction.

#### **4. Conclusion**

We discussed in this paper, the major challenge of knowledge representation issues in an information extraction task – representing the meaning of the input text, the knowledge of the field of application (or domain knowledge) and the knowledge about the target information to be extracted. In this research, we have chosen a directed graph structure to represent the input text meaning, a domain ontology to represent the domain knowledge and a frame representation to capture the target information to be extracted. We discuss in this paper how these knowledge structures interplay to perform the task of information extraction. These structures have been implemented in the context of a practical IE Architecture.

#### **References**

1. Appelt D E, J Bear, J R Hobbs, D Israel and M Tyson (1992). "SRI International FASTUS System" Proc. MUC-4, Morgan Kaufmann : 143-147.
2. Bobrow G. Daniel and Winograd Terry (1977). "An Overview of KRL, a Knowledge Representation Language". Cognitive Science 1(1), 1977,3-46.
3. Bobrow G. Daniel, R M Kaplan, M Kay, D A Norman, H Thompson and T Winograd (1977). "GUS, A Frame-Driven Dialog System" Artificial Intelligence, North-Holland Publishing Company 1977: 155-173.
4. Brachman R.J. and Schmolze J.G. (1985). "An Overview of the KL-ONE Knowledge Representation System". Cognitive Science 9 : 171-216.
5. Charniak Eugene (1978). "On the use of framed knowledge in language comprehension". Artificial Intelligence 11 :225-265
6. DARPA (1991). Proc. of Third Message Understanding Conference (MUC-3). Morgan Kaufmann Publishers Inc.
7. DARPA (1992). Proc. of Fourth Message Understanding Conference (MUC-4). Morgan Kaufmann Publishers Inc.
8. DARPA (1993). Proc. of Fifth Message Understanding Conference (MUC-5). Morgan Kaufmann Publishers Inc.
9. DARPA (1995). Proc. of Sixth Message Understanding Conference (MUC-6). Morgan Kaufmann Publishers Inc.
10. Jensen Karen, Heidorn E. George, Richardson D. Stephen 1993 Natural Language Processing : The PLNLP Approach. Kluwer Academic Publishers, Boston/Dordrecht/London. Chapter 16 : 203-214. Chapter 21 : 273-283.
11. Krupka G, P Jacobs, L Rau, and L Iwanska (1991). "The GE NLToolset System" Proc. MUC-3, Morgan Kaufmann.

12. Marco Costantino, Richard G. Morgan, Russell J. Collingham, Roberto Garigliano (1997). "Natural Language Processing and Information Extraction: Qualitative Analysis of Financial News Articles" Proc. of Conference on Computational Intelligence for Financial Engineering (CIFER '97), New York City, March 23-25, 1997.
13. Nyberg H. Eric (1988). "The FrameKit User's Guide Version 2.0", Carnegie Mellon University, 1988.
14. Tan Sian Lip, Tong Loong Cheong (1993). "A statistical approach to automatic text extraction." Asian Libraries, Vol 3 No 1, Mar 1993: 46-54.
15. Tan Sian Lip, Aw Ai Ti (1993). "Domain specific information Extraction - a NLP-Enable application." Proc. of the First Symposium on Intelligent Systems Applications (SISA '93), Singapore, Nov 1993.
16. Tong Loong Cheong, Wee Li Kwang, Goh Ann Loo, Lee Chee Qwun, and Teo Pit Koon (1992). "A Telex Destination Identification System." Proc. First Singapore Int. Conf. on Intelligent Systems (SPICIS 92), Sep 1992: 281-287.
17. Allen James (1987). "Natural Language Understanding". University of Rochester. Menlo Park: The Benjamin/Cummings Publishing Company, Inc..
18. Wan Kwee Ngim, Tong Loong Cheong, Lynda Ang Seok Lay (1993). "Automatic Categorisation of Cargo Descriptions." Proc. of the First Symposium on Intelligent Systems Applications (SISA '93), Singapore, Nov 1993.
19. Tong Loong Cheong, Angela Wee Li Kwang, Augustina Gunawan, Goh Ann Loo, Lee Chee Qwun, and Shu Huey Leng (1994). "A Pragmatic Information Extraction Architecture for the Message Formatting Expert (MFE) System". Proc. Second Singapore Int. Conf. on Intelligent Systems (SPICIS 94), Nov 1994: B371-377.
20. Wee Li Kwang Angela, Tong Loong Cheong, Chng Tiak Jung (1997). "DeNews - A Personalized News System." Journal of Expert Systems with Applications, Vol. 13, 1997, Elsevier Science Ltd, UK 0957-4174/97.
21. Dolan C P, S R Goldman, T V Cuda and A M Nakamura (1991). "Hughes Trainable Text Skimmer" Proc. MUC-3, Morgan Kaufmann : 155-162.
22. Tong Loong Cheong, Low Poh Lian (1991). "Automatic Text Abstraction - Prospects and a Proposed R&D Plan" Information Technology, Journal of SCS, Vol 4 No 2, Sep 1991: 85-94.
23. Julian Kupiec, Jan O. Pedersen, Francine Chen (1995). "A Trainable Document Summarizer". Proc. of the 18<sup>th</sup> ACM/SIGIR Conference, 1995: 68-73.