TOWARDS A LANGUAGE INDEPENDENT ENCODING OF DOCUMENTS

A Novel Approach to Multilingual Question Answering

Pushpraj Shukla, Amitabha Mukherjee, Achla Raina
Indian Institute of Technology, Kanpur, India – 208016
Email: praj@iitk.ac.in , amit@iitk.ac.in , achla@iitk.ac.in

Keywords: Multi-linguality, inter-lingua, semantics, ontology, en-conversion, de-conversion

Abstract: Given source text in several languages, can one answer queries in some other language, without translating any of the sources into the language of the questioner? In this paper we try to address this question as we report our work on a restricted domain, multilingual Question – Answering system, with current implementations for source text in English and questions posed in English and Hindi. The cross - language functionality has been achieved by converting the queries and the documents to an intermediate representation, an inter-lingua called UNL. Built under the premise of a homogeneous language-independent encoding, the UNL can be used as the predicate knowledge base on which inferences can be performed effectively.

1 INTRODUCTION

Natural language question answering systems aim at making information access easier by using natural-language questions directly and effectively as search queries by exploiting their semantic structure, without asking for additional human effort to translate the questions into sets of keywords. Much of the current research in question answering systems focuses on open domains. The availability of large volumes of data (e.g. documents extracted from the World Wide Web) has prompted the development of systems that focus on shallow text processing. But there are many document sets in restricted domains that are potentially valuable as a source for question answering systems. For example, the documentation pages of Unix and Linux systems would make an ideal corpus for Q/A systems targeted at users that want to know how to use these operating systems. There is a wealth of information in other technical documentation such as software manuals, car maintenance manuals, and encyclopaedias of specific areas such as medicine. Users interested in these specific areas would benefit from Q/A systems targeted to their areas of interest.

Multilinguality and cross-linguality have emerged as issues of great interest to the Language Engineering community that deals with Question Answering systems. Many organizations worldwide are grappling with problems like the following: Given source text in several European languages, would it be possible to demonstrate semantic understanding in some other language (like Hindi) without explicitly translating any of the sources into the language of the questioner? This is a difficult question to address at first. We present here a tool in this direction which can be treated as a mechanism for mapping any language into a uniform language-independent predicate structure and thus make the task of multilingual Q/A more realistic.

Traditional approaches classify cross–language information management systems as follows:

1. systems that translate queries into the target language, or
2. systems that do the document collection into the source language, or both,
3. systems that convert queries and documents into an intermediate representation (inter–lingua).

We adopted the last of the above schemes to implement the multilingual features in the Q/A System. The inter-lingua used for the above purpose was the “Universal Networking Language” (UNL) (Uchida, 2001; www.undl.org). Our choice of the inter-lingua depended on the functionality and the language independent features it provided. With its set of “Universal Words”(UWs) having well defined universal interpretations and ontological information embedded in them, a small and simple binary predicate structure and a

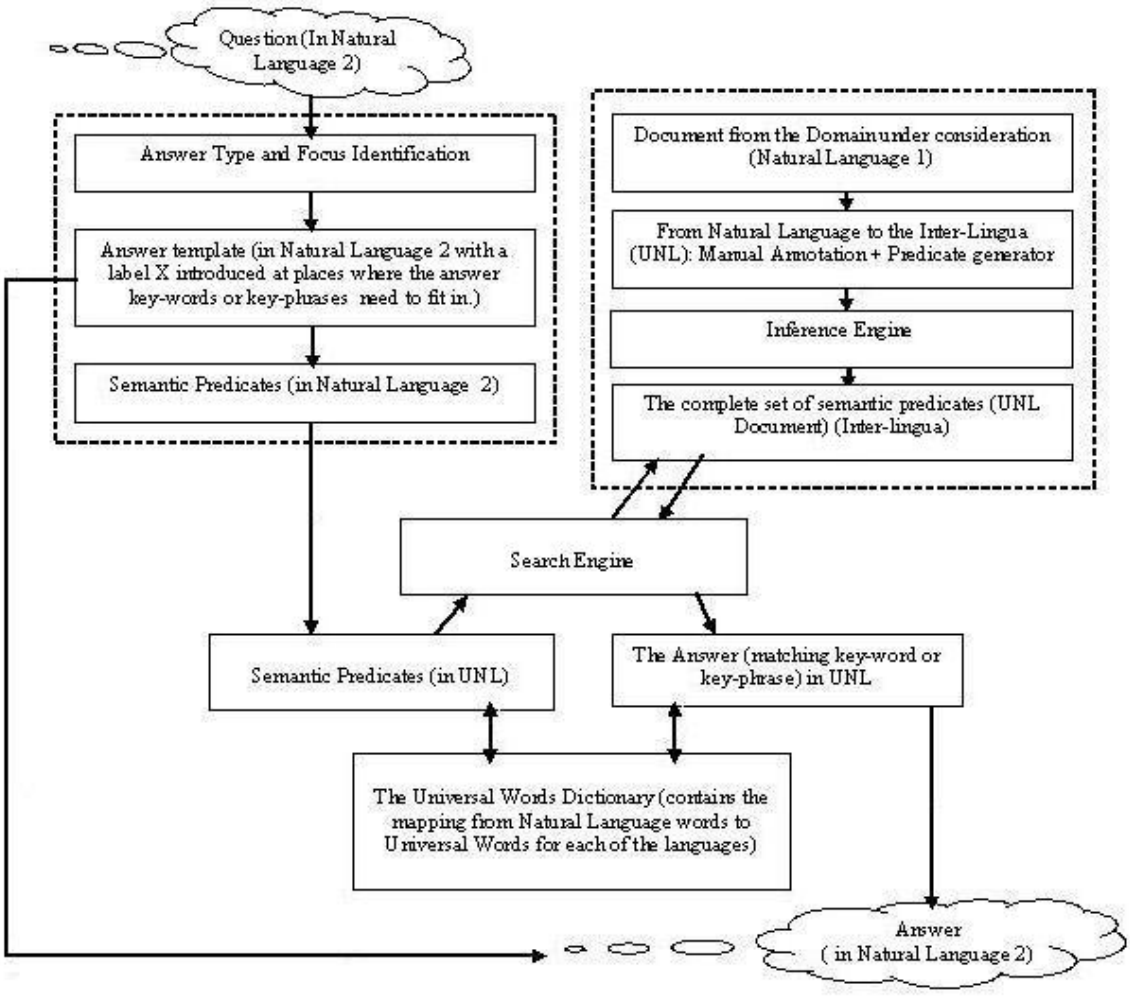

Figure 1: The System Implementation Flowchart

knowledge base connecting the UW's as a weighted graph of relations, UNL proved to be the most fit. Added to it is the world wide effort in developing en-converters and de-converters for converting languages into UNL and vice versa, which if fruitful, would allow us to extended our Q/A model to substantially unrestricted, open – domains.

2 SYSTEM OVERVIEW

Figure 1 above displays the control flow between the major components of the Q/A module. Two major tasks are involved in the system implementation. The first is that of document processing, which is to be done once for each document we wish to be queried. This converts the document into the interlingua, with added inference rules. The other involves query processing and answer extraction and is done once for every query. Our present work is based on an English corpus for safe drinking water built from documents obtained from the official websites of EPA and WHO (Ground Water and Drinking Water, EPA[Online]; WHO:Fact Sheets [Online]).

3 DOCUMENT PROCESSING

The corpus is manually annotated and processed through a Predicate Generator (the UNL Parser) to get the complete set of semantic predicates for the document (the "UNL expression" for the corpus). Some common known facts are also added to provide context information which would be

commonly known to the human reader but is not available from the text itself.
For example, the NL corpus sentence, At some level, minerals are considered contaminants that can make water unsafe, is annotated as follows:

```
<c>At some{<qua,>n} level.n.@pl.
@entry</c> {<man,>p} mineral.@pl {<gol,>p}
are consider.p contaminant{1}
.@pl{<obj,<p} that {<1}{<agt,>p} can
make.p.@ possible
water{<obj,<p}<c>unsafe{<or,>p} or
even{<man ,>p} unsafe.p.@entry</c>{<gol
,<p}.
```

The corpus sentence in its annotated form is input to the UNL parser to generate the UNL parsed graph, represented as a list of relations or predicates.

3.1 Semantic Equivalence

The same information may be expressed in very different ways:

1. Safe water can be obtained through boiling and distillation.
2. We can obtain safe water through boiling and distillation.
3. We can get safe water through boiling and distillation.
4. The methods for making safe water are boiling and distillation.
5. One can make safe water by boiling or distillation.
6. While distilling results in pure water, for practical purposes, boiling is sufficient to make water safe for drinking.

Fortunately, a part of this problem (eg. active vs passive voice) is resolved by the UNL encoding process – thus (1) and (2) will result in the same UNL structure:

agt(get(agt>thing,obj>thing,src>thing).@possible, we);
obj(get(agt>thing,obj>thing,src>thing), water(icl>liquid));
mod(water(icl>liquid), safe(mod<thing));
man(get(agt>thing,obj>thing,src>thing),
through(icl>how(obj>thing)));
obj(through(icl>how(obj>thing)), :01);
and :01(boiling(icl>act), distillation(icl>act));

Even (3) which uses the word "get" which is used here in the same sense as "obtain" results in the same universal word get(agt>thing,obj>thing,src> thing) and thus result in the same UNL structure. However sentences (4 and 5) use "make" which has a different UW, and these are handled in the inference engine by using rules for unifying similar UWs when used in the context of water. Very wide variations such as (6), which requires added pragmatic knowledge such as "pure water is safe", and also results in a set of two conjunctive UNL expressions (one for the "while" clause, and the other for the main clause) can be handled but since the set of such constructs is very large, they are not handled in the current version.

3.2 The Inference Engine

Although UNL structure provides a first order logic encoding of natural language, it is not designed for making semantic inferences, and an inference engine needs to be built for this purpose. The rules embedded in the process implement the First Order Logic in order to obtain new inferences. For example, given the facts "*Water-borne diseases are caused by ingestion of contaminated water. and cholera is a water-borne disease"*, one may infer that "cholera is caused by ingestion of contaminated water."

A meta-rule for this situation, incorporated as part of the inference rule base is that, given:
agt(cause(icl>abstract thing).@entry:1);
obj(cause(icl>abstract thing).@entry:2);
nam(2:3);
which says that variable 1 causes 2, and 2 is a type of 3. Given this set of UNL relations, the meta-rule says that one can infer:
agt(cause(icl>abstract thing).@entry:,1:);
obj(cause(icl>abstract thing).@entry:,3:);
i.e. 3 is caused by 1. The current system is designed to be tested only on a simple Question and Answer mechanism. We use single-tiered inferences, and construct a complete set of all possible inferences that can be made from the given text and the pragmatic rules.

4 THE QUESTION – ANSWER MODULE

Every query is processed to get an answer template that represents the form of the potential answer corresponding to the question. This template is input to an HPSG Parser (Sharma et al, 2002) which outputs a pseudo UNL expression corresponding to the proposed answer template. The pseudo UNL expression is subsequently subject to a structure matching with the UNL document. The multilinguality of the system comes from the multilingual characteristic of the parser and the language – independent semantic structure provided by the inter-lingua (UNL).

4.1 Question Analysis and Processing – Generating the Answer Template

The goal of the question analysis phase is to determine the focus of the question and the expected type of answer for it. We then generate an answer template that represents the form of the answer corresponding to a question, with a label "X" introduced at the "answer slot" (the position where the answer key-word or key-phrase needs to fit on). "X" represents the entity to be searched for as an answer. In this work we use a set of transformational rules to arrive at the answer template. These rules have been categorized on the basis of different question types, like the "wh" questions, the "yes/no" questions, the "how" questions etc. The rules in each category check different features of the question. In particular the rule may detect the presence of a particular word or phrase occurrence, like words of a given part of speech or those of a particular semantic category. The rules introduce the label "X" at the location of the keyword or the key phrase in the answer pattern. For example, the rule, *HOW:AUX:1:V(PPL) > 1:AUX:V(PPL):BY:X*, works upon a question such *as How is water contaminated?* which is transformed to its corresponding answer template *Water is contaminated by X*.

Most of these rules are language dependent. We have used a rule base of around 50 transformation rules for English and around 20 rules for Hindi, which offers a relatively easier answer type identification and template generation. An example could be – 1:kha:V(pre) > 1:X:ma:V(pre) which matches questions like, "jala kha bahta h?"(Where does water flow?) and have a easy transformation to the answer "jala naidyaa ma bahta h", while its English counterpart will have to follow from a rule like where:does:1:V(base):2 > 1:V(pres):Prep(pos):X:2 to produce "water flows in rivers." which requires a verb-form transformation also.

There will be cases where more than one transformation rule applies for a given question. In such cases, we let all the possible answer templates pass on to the next phase. This is very common in case of complex query structures, for example the "how" queries in English and the "ksa" queries in Hindi. No answer selection is done at this stage.

Thus the queries in one language are converted into the corresponding answer templates in the same language and passed on to the next phase – the semantic parser.

4.2 From the Natural Language (NL) Answer Template to the NL semantic predicates

The answer template is converted into a pseudo UNL representation by a multilingual HPSG Parser (Sharma et al, 2002) which operates on lexicons specifying the semantic selection (as against the categorical selection) properties of phrasal heads. We have build separate semantic lexicons for English and Hindi with shallow syntactic rules. Semantic relation attributes are mostly used in the lexicon instead of syntactic subcategory features since the parsed answer form needs to be unified with a database that is in the UNL semantic predicate format, i.e. the UNL Document. The UNL structure uses relations that are defined in terms of semantic features such as agency, place, etc. Therefore, these relations need to be identified in the parsed answer form for structure matching to be possible. To take examples of a lexical entries the said semantic feature information, let us look at the verb "makes" and its corresponding verb "banaata" in Hindi:

<makes>@V(pre,sg){agt|!|obj|~gol|~plc}
< banaata >@V(pre,sg,male){/agt|~plc|~obj|~gol\|!}

So, for the word "makes", we can capture the syntactic information of it being a singular verb(sg) and used in present tense(pre) while the semantic description "{agt|!|obj|~gol}" captures the fact that "makes" heads a phrase which takes the form of an Agent relation followed by the verb itself and then an Object relation, an optional Goal relation and an optional Place(location) relation". Note that agt, obj ,gol and plc are all UNL relations. "banaata" has similar semantic subcategorization though the positioning of the various sub-phrases in the phrase that "banaata" heads is different and is specific to the Hindi language. The parser thus needs different lexicons for parsing answer templates from different languages .

The Nominal heads which take on the roles agent, object and goal are entered in the two lexicon were as follows:

<impurities>@agt(pl){~qua|~mod|!|~plc}
<water>@obj(sg){~qua|~mod|!|~plc}
<unsafe>@gol{!}

<klTaNa >@agt(pl){~qua|~mod|!|~prep}
<jala>@obj(sg){~qua|~mod|!|~prep}
<Asarixat >@gol{!}

The HPSG parser reads the lexicons and states relations as given in lexical entries. From the parsed tree thus obtained, we can get the relation

between two nodes, which would essentially be the label attached with the child node. To take an example of how the pseudo-enconverter works, given the questions in Hindi(and English), *"jala ka kana Asarixat banaata h ?"(What makes water unsafe*?), we generate the answer templates *jala ka X Asarixat banaata h (X makes water unsafe)*, with the transformational rules 1: *kana*:2 > 1:X:2 (and *what:V:1 > X:V:1)*

The parsed outputs are as follows:

((X agt(sg)) makes V(pre,sg) (water obj(sg)) (unsafe gol))
+-X makes water unsafe
|+-X_agt(sg)
|+-makes water unsafe
| | +-makes_V(pre,sg)
| | +-water_obj(sg)
| | +-unsafe_gol

(((*jala _agt(sg){}* (* ka _prep{}*)) (*X_obj(){}*)(*Asarixat_gol{}*)*banaata V(pre,sg,male){}*)*h _aux{}*)
+- jala ka X Asarixat banaata h
+- jala ka X Asarixat banaata
| +-jala ka
| | +-jala _agt(sg)
| | +- ka _prep
| +-X_obj(sg)
| +- Asarixat _gol
| +- banaata _ V(pre,sg,male)
+- h _aux

The list of semantic predicates produced is –

agt(makes,X)	agt(*banaata*,X)
obj(makes,water)	obj(*banaata*, *jala*)
gol(makes,unsafe)	gol(*banaata*, *Asarixat*)

4.3 From NL semantic predicates to inter-lingua

The set of predicates obtained thus far have the attributes in the Query Language. In this phase these NL words are mapped to Universal Words (UW's of UNL) to produce the UNL semantic predicates(the UNL expression), very similar to the form in which the Document to be queried had been converted using manual annotation and predicate generation (Section 2).

So it is in this phase that the query (or the Answer template) representations, which had so far been in different languages, are brought to a homogeneous language-independent encoding suitable for search.

The UW Dictionaries give us the mapping form NL words to the UWs . Mainly built to cater the needs of the machine translation groups, the UW Dictionaries are expected to be huge and comprehensive. But since we did not have any formal product release yet, we implemented our own dictionaries to cover our corpus needs. The typical entries in the UW Dictionaries are like –

The English dictionary –
[makes] {} "make(agt>thing,obj>thing,src>thing)" (VER,PRE,SG) <ENGLISH>
[water] {} "water(icl>liquid)" (NOUN,SG) <ENGLISH>
[unsafe] {} "unsafe(aoj>thing)" (ADJ) <ENGLISH>

The Hindi Dictionary -
[banaata] {} "make(agt>thing,obj>thing,src>thing)" (VER,SG,MALE,TAA) <HINDI>
[jala] {} "water(icl>liquid)" (NOUN,SG) <HINDI>
[*Asarixat*] {} "unsafe(aoj>thing)" (ADJ) <HINDI>

As can be seen, along with the translation of the words , the UW Dictionaries also give us additional information regarding features of a word. By using the two dictionaries, and processing some additional information that was carried over from the last phase (for example the tense information we carried over for the Hindi answer in a previous example), we can thus obtain the UNL predicates for the answer template which prove to be same for similar queries in different languages. Thus for the two answer templates *jala ka X Asarixat banaata h* and *X makes water unsafe* ,corresponding to the two similar queries in Hindi and English respectively , we obtain the same UNL Expressions -

agt(make(agt>thing, obj>thing, src>thing). @pre, X)
obj(make(agt>thing, obj>thing, src>thing).@pre , water(icl>liquid))
gol(make(agt>thing, obj>thing, src>thing),unsafe (aoj>thing))

This set of predicates is now searched for and matched with the UNL Document of the corpus to give us the entities which are contenders for fitting in place of label X in the Answer Template.

4.4 The Search Engine

Given the answer form of the question in UNL predicates, it has to be matched with the UNL document (the complete set of UNL predicates for the entire corpus) to see if an answer can be provided. First, each sentence in the UNL Document (as generated in section 2) is converted into a UNL graph, with two arguments as nodes, connected by a link with the label of the relation. Next, we convert the answer template present as UNL predicates into the UNL graph. Continuing with our example question , *jala ka kana Asarixat banaata h*

(What makes water unsafe), the UNL predicates for the answer template described in the previous section produce a graph like the following –

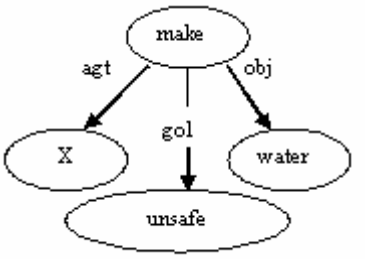

Figure2: The Graph for the template "X makes water unsafe".

Finding an answer involves matching this Query UNL graph with a UNL sub graph from the UNL Document, If one of the nodes in the query has a variable X then the match returns the value of this variable. For "yes/no" question the matching of the whole graph returns "yes" else a "no". For such questions, there is no label X in the answer template.

4.5 Conversion of the Answer from the inter-lingua back to NL (De-conversion process)

The search returns a possible entity (key-word or a key-phrase) in UNL which can serve as the answer (and a replacement for X) in the answer template. If the answer returned is a word, then the de-conversion from the Universal Word to the NL word can be done directly by finding the reverse mapping from the UW Dictionary. In case of phrases as answers, a formal de-conversion process has to be followed. We adopted two strategies to deal with phrasal de-conversion. The first one makes use of the "scope" functionality in UNL description wherein a complete phrase is taken within a "scope" and is said to form relations with other parts of the sentence as a single word. For example, in the question "How is cholera caused" which we looked up in the search engine the phrase "ingestion of contaminated water" is treated as a scope and the predicates generated for the document are –

obj(cause(agt>thing,obj>thing),cholera(icl>disease))
agt(cause(agt>thing,obj>thing),SCOPE1)
label(SCOPE1, ingestion of contaminated water)

The search is thus made to return the complete SCOPE instead of the individual word "contamination". For each of the SCOPES then the UW Dictionary also has complete phrasal translations entered manually. For example –

[sakiMat jala k gahNa] {} "ingestion of contaminated water" (SCOPE) <HINDI>

The second strategy uses shallow de-conversion rules. These rules cover common phrasal constructs, particularly the noun phrases. Some examples are –

mod(X,Y) = X|Y which means that a modifier relationship between X and Y would be "the word Y followed by the word X" in natural language. This rule for example works well for both English and Hindi. For example mod(water,contaminated) can is de-converted to "contaminated water" and to "sakiMat jala" in English and Hindi respectively, which are valid constructs for both languages.

During our search , we incorporate both the strategies though the first one is given preference owing to the unavailability of robust UNL-Hindi deconverters. The answer obtained from this phase can directly be inserted in place of the label X in the answer template and reported. In case of multiple answers matching a given set of predicates, all the results are reported. There is no word sense disambiguation or answer selection applied in this process.

5 RESULTS

The present system has given results upto 60% accuracy to the questions which have answers directly taken from the corpus, both for Hindi and English. Failure analysis of also suggests that upto 50% of the failures in obtaining a satisfactory answer in the language of the query were due to the inabilities of the system to de-convert complex English phrases into query language. The conversion of the question to answer template resulted in another 27% of the failures. Also, the short answer questions(one word and two word) are answered much more successfully(upto 70% success rate) than the phrasal ones and the performance keeps on decreasing with the complexity of the answer. The analysis of the system was done on a database of questions given by students in the project group and other volunteers.

6 CONCLUSION

Being our first attempt at developing a Question Answer module, the focus was more on system implementation than on system tuning. This may

be the first attempt toward developing a cross-language Hindi/English Question Answer Module, and though the system is not restricted in any form to any language, we have reported our current implementations for English and Hindi. The module is being tested on short stories for children and the results are encouraging.

The performance of the system, for any language, will critically depend on the effectiveness of the En-conversion (form NL to UNL) and the De-conversion(for UNL to NL) engines developed for that language. Although, we have been able to achieve reasonable accuracy on the En-conversion front, by using a self developed parser for the purpose, the de-conversion task is still far from good. This imposes a heavy restriction on the complexity of the answers that can be extracted in different languages. While the UNL structure is a First Order Predicate form, there are remarkable differences with normal logical models. For one, UNL structures do not provide for an implication connective, and also use the disjunction relation "or" rather sparingly. A rigorous mapping to more traditional logical structures is needed for more extensive UNL based logical inferencing. Efforts are on in this direction.

Also, the manual process of designing the pragmatic knowledge-base is expensive – it needs to be seen if further synergies can be gained by unifying this effort with parts of the UNL KB. Despite these shortcomings, we hope the present work will provide a start towards this difficult yet important problem.

REFERENCES

Bittner, M. (1998) 'Cross-Linguistic Semantics for Questions', *Linguistics and Philosophy*, vol. 21, pp. 1-82.

Uchida, H., 2001. UNL Beyond Machine Translation. *International Symposium on Language in Cyberspace, Seoul, Korea Systems*. ICEIS Press.

Ground Water and Drinking Water, EPA. [online]. Available from <http://www.epa.gov/safewater/dwh/contams.html > [Accessed 6 July 2003].

WHO:Fact Sheets [online]. Available from http://www.who.int/inf-fs/en/fact112.html [Accessed 6 July, 2004].

Sharma, D., Vikram ,K., Mittal, M. R., Mukerjee, A., Raina, A., M. (2002) 'Saarthaka-A generalized HPSG parser for English and Hindi', *Recent Advances in Natural Language Processing-Proceedings* ICON-2002, Mumbai, India, pp. 18-21.

Martins, R. (2002) 'UNL as a linguistic point of view' ICUKL, Goa, India

Moldovan, D., Pasca, M., Harabagiu, S., Surdeanu, M. (2002) 'Performance Issues and Error Analysis in an Open-domain Question-Answering System.' *40th Annual Meeting of the Association for Computational Linguistics(ACL)*, Philadelphia, pp. 33- 40.
.

Moldovan, D., Clark, C., Harabagiu, S., and Maiorana, S., (2003) 'COGEX: A Logic Prover for Question Answering.' *Proceedings of HLT-NAACL Main Papers* ,Edmonton, pp. 87-93.