ISE

ISE-TR-90-85

DOCUMENT RETRIEVAL AND IMAGE RETRIEVAL BASED ON FUZZY PROPOSITIONAL INDEX

by

Sadaaki MIYAMOTO, Nobuaki KONISHI, and Teruhisa MIYAKE

September 27, 1990

INSTITUTE
OF
INFORMATION SCIENCES AND ELECTRONICS

UNIVERSITY OF TSUKUBA

Document Retrieval and Image Retrieval Based on Fuzzy Propositional Index

S. Miyamoto, N. Konishi*, and T. Miyake

Institute of Information Sciences and Electronics
University of Tsukuba, Ibaraki 305, Japan

*Master's Program in Sciences and Engineering
University of Tsukuba, Ibaraki 305, Japan

ABSTRACT

The aim of the present paper is to propose fuzzy propositional index and retrieval for documents or images. A set of propositions represents content of a document or an image. A query of the same form of fuzzy propositions is matched with fuzzy propositional indices for a set of documents or images using a matching function, which is a fuzzy relation between two sets of fuzzy propositions. The matching degree by the fuzzy relation is interpreted as the membership value in the retrieved set. Three algorithms are derived for the matching function which is based on a fuzzy set model, and their efficiency is estimated. Illustrative examples for document retrieval and image retrieval are given.

Keywords: fuzzy propositional index, document retrieval, image retrieval, fuzzy retrieval, matching function, algorithms.

1. Introduction

In the study of cognitive psychology and artificial intelligence, propositions or clauses have widely been used for modeling cognitive structure. Since information retrieval in human memory is no doubt important as a part of psychological processes, data retrieval or document retrieval techniques should utilize models studied in human cognition processes. Nevertheless, the present method of document retrieval only discuss matching between a keyword and indexed terms for documents.

This paper discusses the use of propositions for index of documents and images, and accordingly the same form of propositions is used for queries. The propositional index here, which is named after Anderson's propositional network (1980) in human memory, means that contents of a document are summarized into a set of n-tuples as abstraction of a sentence. For example, a sentence "Fuzzy sets are used for information retrieval" is transformed into 3-tuple of (use, fuzzy sets, information retrieval). A set of such n-tuples forms index of a document, and a query of such a set is used for retrieval. The same type of propositions can be used for indexing of images, therefore we consider here document retrieval and image retrieval using the same form of propositional index.

The two sets of propositions in an index and in a query should be matched using some matching functions, and matching degree is considered. When we have a database with such propositional index, the matching degree is interpreted as membership of fuzzy retrieval. We consider a few matching functions based on a fuzzy set model, and develops algorithms for calculating matching degree. Simple examples are discussed to show how the matching degree is calculated.

2. Fuzzy propositional index

In artificial intelligence, predicate forms or clauses are used for inference: a typical example of the expression is *father-of(Zeus,Apollo)* which means that Zeus is father of Apollo. The same type of an abstracted expression for a sentence is used for representing human memory structure: Anderson (1980) uses propositional network for such representation. In the Anderson's representation, a proposition can be interpreted as an n+1-tuple $(p, a_1, a_2, ..., a_n)$, in which p shows relation among the arguments $a_1, a_2, ..., a_n$. For example, the sentence "birds have feathers" is represented as (have, birds, feathers). Since we do not consider inference, but discusses matching between two such n+1-tuples, the word *proposition* is used here. As was noted before, we use a set of the n+1-tuples (propositions) for index of a document or an image. In such indexing, we sometimes require grade of importance or relevance for a proposition, hence the above n+1-tuple is generalized to include the membership value μ, therefore we consider a set of n+2-tuple $(\mu, p, a_1, a_2, ..., a_n)$, which is called a fuzzy propositional index.

Example 1.

Let us consider two documents A,B whose titles are as follows.

A. information retrieval using fuzzy sets.
B. document retrieval based on fuzzy indices.

Ordinary index terms for these documents are

A. information retrieval, fuzzy sets,
B. document retrieval, fuzzy indices.

On the other hand, propositional indices are

A. (1.0, using, information retrieval, fuzzy sets)
B. (1.0, based on, document retrieval, fuzzy indices)

Note that in the latter representation, relationship of the index terms for each document is made clear.

Fuzzy thesaurus may be used at different stages of fuzzy retrieval. For example, relation p in a proposition may be replaced by other related term p'. In the above example, the two relation terms may be related by a thesaurus F: F(based on,using)=0.9. Accordingly, the second proposition becomes

(0.9, using, document retrieval, fuzzy indices).

Thus, even when the initial membership values are all unity, they may be replaced by other values using a fuzzy thesaurus.

Example 2.

Let us consider indexing of an image. Examples of the propositional network representing contents of images have been studied by Anderson (1980) and Sakauchi (1988). Sakauchi showed a simple figure in which a person is in front of a house, and a flag is on the top of that house. He used a network which is equivalent to the next two propositions:

(1.0, in-front-of, person, house),
(1.0, on-the-top-of, flag, house).

When the title is "fuzzy sets", or the image shows "Mt. Fuji", what proposition should be used? One way for such an independent argument is to use a formal relation "on". Namely, the indices are (1.0, on, fuzzy sets) and (1.0, on, Mt.Fuji).

3. Matching function and fuzzy retrieval

Queries for such a propositional index may be simple index terms, in which case usual method of matching between query terms and index terms in the propositions may be used. Alternatively, queries may have the same form of propositions; in the latter case matching between a query and an index set becomes much more complicated. In the case of propositions, perfect matching between a query and an index is exceptional. Accordingly, partial matching should be considered and matching degree should be measured.

For considering degree of partial matching, we introduce the following symbols. First, W is a set of terms used for relations or arguments of propositions. Propositions are denoted by $x = (\mu, p, a_1, ..., a_n)$, $y = (\mu', p', a'_1, ..., a'_{n'})$, and so on, where $p, p', a_1, a'_1, ...a_n, a'_{n'} \in W$. The set of all such propositions is denoted by $FP(W)$. Thus, $x, y \in FP(W)$. In general, the number of arguments may be different depending upon propositions: $n \neq n'$ for the above x and y. By abuse of terminology, we assume that x represents the n+1 tuple $(p, a_1, a_2, ..., a_n)$ and its membership μ at the same time, and accordingly set operations can be performed between two propositions which have the same n+1 tuple. For example, when $x = (\mu, p, a_1, ..., a_n)$ and $x' = (\mu', p, a_1, ..., a_n)$, we have

$$x \cup x' = (\max[\mu, \mu'], p, a_1, ..., a_n)$$

From now on we assume that a query and an index are subsets of $FP(W)$, which are denoted by Q and I, respectively. Two types of matching degree between Q and I are measured using two matching functions $s(I, Q)$ and $t(I, Q)$. The measure s is symmetric: $s(I, Q) = s(Q, I)$, whereas t is nonsymmetric in general. Since we propose different forms of the measures s, t, they are distinguished by subscripts s_1, t_1, s_2, t_2 and so on.

Various ideas may be used for defining the functions s, t; one simple way is to use a fuzzy set model for thesaurus (Miyamoto et al. 1983). Let $|A|$ be the cardinality of a fuzzy set A, we may define

$$s(I, Q) = \frac{|I \cap Q|}{\max[|I|, |Q|]}$$

$$t(I, Q) = \frac{|I \cap Q|}{|Q|}$$

$$(|Q| = \sum_{x \in Q} \mu_Q(x))$$

It is easy to note that $s(I, Q) = 1$ if and only if $I = Q$, whereas $t(I, Q) = 1$ if $Q \subseteq I$. Namely the measure t means the degree of inclusion of Q in I. Since

$$s(I, Q) = \min[t(I, Q), t(Q, I)],$$

we consider only the function t hereafter.

Fuzzy thesaurus is a usual tool for improving retrieval effectiveness, therefore we use a reflexive fuzzy relation F on W which implies a thesaurus in defining the matching functions. We consider the following three measures.

$$t_1(I,Q) = \frac{|F(I) \cap Q|}{|Q|}$$
$$t_2(I,Q) = \frac{|I \cap F(Q)|}{|F(Q)|}$$
$$t_3(I,Q) = \frac{|F(I) \cap F(Q)|}{|F(Q)|}$$

$F(I)$ is defined as follows. For $x = (\mu, p, a_1, ..., a_n) \in I$, define

$$F_I(x) = \{y = (\mu', p', a'_1, ..., a'_n) | \mu' = \min[\mu, F(p,p'), F(a_1,a'_1), ..., F(a_n,a'_n)]\} \quad (1)$$
$$F(I) = \bigcup_{x \in I} F_I(x)$$

Among the above three functions, we consider solely t_1 hereafter, since t_1 is simpler than t_3, and t_2 is less interesting in its theoretical property than t_1, as we will see later.

A simple algorithm for calculating t_1 is

1. extend I into $F(I)$ using the above definition.
2. take the intersection $F(I) \cap Q$
3. count $|F(I) \cap Q|$ and $|Q|$ and then calculate $t_1(I,Q)$.

This algorithm is not efficient, in particular, in the case of many document data or image data, since every index of a data unit must be extended by the thesaurus. We therefore consider other algorithms for calculating the matching functions. It is sufficient to consider

$$K(I,Q) = |F(I) \cap Q|$$

because calculation of $|Q|$ is straightforward. Now, we note the following proposition.

Proposition 1. *For $x = (\mu, p, a_1, ..., a_n)$ and $y = (\mu', p', a'_1, ..., a'_n)$, let*

$$L(x,y) = \min[\mu, \mu', F(p,p'), F(a_1,a'_1), ..., F(a_n,a'_n)]. \quad (2)$$

Then,

$$K(I,Q) = \sum_{y \in Q} \max_{x \in I} L(x,y) \quad (3)$$

(Note that if lengths, i.e. the numbers of arguments, of two propositions x and y are different, then $L(x, y) = 0$.)

(Proof) For simplicity, max is denoted by $\vee$ and min is denoted by $\wedge$ in this proof. Moreover, membership values at z for the fuzzy sets Q, $F_I(x)$, and $F(I)$ are denoted by $Q[z]$, $[F_I(x)][z]$, and $[F(I)][z]$, respectively. Now, we have

$$\begin{aligned}
\max_x \min L(x, y) &= \bigvee_x \{Q[y] \bigwedge [F_I(x)][y]\} \\
&= Q[y] \bigwedge \{\bigvee_x [F_I(x)][y]\} \\
&= [Q \bigcap \{\bigcup_x F_I(x)\}][y] \\
&= [Q \cap F(I)][y]
\end{aligned}$$

Hence we have

$$\begin{aligned}
K(I, Q) &= \sum_y \{Q \cap F(I)\}[y] \\
&= |Q \cap F(I)|
\end{aligned}$$

(QED)

Using this proposition, three algorithms can be developed for calculating t_1. In the following description of the algorithms, *for all I in DB do — repeat* means the repeat for all indices in a database. We also assume that a query Q is given.

Algorithm A
for all I in DB do
  for all $y \in Q$ do
    for all $x \in I$ do
      compute $L(x, y)$ using (2)
    repeat
    compute $K(I, Q)$ using (3)
  repeat
  compute $t_1(I, Q) = K(I, Q)/|Q|$
repeat

This algorithm simply calculate $L(x, y)$ for all $x \in I$, $y \in Q$, and then $K(I, Q)$ is calculated by (3). Thesaurus file is referred to for each calculation of $L(x, y)$.

Next algorithm is nothing but the simple calculation of $|F(I) \cap Q|$ noted before. Remark also that $F(I)$ in the next algorithm means a set of records defined by $F_I(y)$ given by (1).

Algorithm B

```
for all I in DB do
  make records F(I)
  L = 0
  for all y ∈ Q do
    L' = 0
    for all x ∈ F(I) do
      compare x = (μ, p, a_1, ..., a_n) and y = (μ', p', a'_1, ..., a'_n), and
      if (p, a_1, ..., a_n) = (p', a'_1, ..., a'_n) then
        L' = max [L', min[μ, μ']]
      endif
    repeat
    L = L + L'
  repeat
  compute t_1(I, Q) = L/|Q|
repeat
```

Two files representing Q and $F(I)$ in which records are the propositions are compared in this algorithm. Note that after the for loop with respect to $y \in Q$, we have $L = K(I, Q)$.

A problem in algorithm B is that every index I must be extended to $F(I)$, which leads to a large amount of computation. Now, let us consider use of $F^{-1}(Q)$ instead of $F(I)$. This idea is realized by Proposition 1 in the next algorithm.

Before describing algorithm C, let us define, for $x = (\mu, p, a_1, ..., a_n)$,

$$G(x) = \{z = (\mu', p', a'_1, ..., a'_n, x) | \mu' = \min[\mu, F(p', p), F(a'_1, a_1), ..., F(a'_n, a_n)]\}$$
$$G(Q) = \bigcup_{x \in Q} G(x)$$

Since the order of the arguments in F is reversed in $G(x)$, $G(Q)$ actually uses the inverse relation F^{-1}. Note also that an element $z = (\mu', p', a'_1, ..., a'_n, x) \in G(Q)$ is indexed by x. In the case of $F(I)$, two records with the same n+1-tuple $(p, a_1, ..., a_n)$ and with different values μ and μ' induced from different x and x' are merged into one record $(\max[\mu, \mu'], p, a_1, ..., a_n)$, whereas in the case of $G(Q)$ they are not merged, as indexed by x and x'.

```
Algorithm C
make records G(Q)
  for all I in DB do
  L = 0
  for all x ∈ Q do L'(x) = 0 repeat
  for all z ∈ G(Q) and for all y ∈ I do
    compare z = (μ', p', a'_1, ..., a'_n, x) and y = (μ, p, a_1, ..., a_n), and
    if (p, a_1, ..., a_n) = (p', a'_1, ..., a'_n) then
```

$L'(x) = \max\ [L'(x), \min[\mu, \mu']]$
endif
repeat
for all $x \in Q$ do $L = L + L'(x)$ repeat
compute $t_1(I, Q) = L/|Q|$
repeat

In this algorithm $L'(x)$ is a table where the value $\max_y L(y, x)$ is stored for each $x \in Q$. After Q has been extended into $G(Q)$, reference to thesaurus F is no longer required.

For the function t_2, it is sufficient to consider algorithms A and B. For applying algorithm C to t_3, it is necessary to consider $G(F(Q))$. We omit the detail.

Now we estimate amount of computation for the three algorithms. For this, let D be the number of data units such as documents or images. Average numbers of propositions in I and Q are denoted by N_I and N_q, respectively. One term in W is extended to M associated terms on average by the thesaurus F. Moreover, an n+1-tuple $(p, a_1, ..., a_n)$ is regarded as a sequence of characters, and accordingly it is called a character sequence $(p, a_1, ..., a_n)$.

Simplification in estimating amount of computation is made here, since we do not concern efficiency of an algorithm in general sense, but comparison of the three algorithms. For example, although various methods such as the hashing can be used for matching a query term to a set of index terms, simple linear search in a sequential file for the matching is assumed. Moreover all data units are referred for calculating matching degree, and accordingly the outermost loop *for all I — repeat* is repeated D times. In algorithm B, when files $F(I)$ and Q are compared, the two sets of records are assumed to be sequential files which have been sorted by the key $(p, a_1, ..., a_n)$. The same method of the comparison is assumed in algorithm C.

Remark. *In general, when two sequential files A of N_1 records and B of N_2 records are sorted by a key, and the key of each record in A is examined against that key of every record in B, the computation time is $Const(n_1 + n_2)$. Here, the constant $Const$ is neglected for simplicity.*

In algorithm A, $N_I N_Q$ times of comparison is performed for a data unit, and in each comparison the thesaurus is used. We assume that one reference to thesaurus F requires C_1 time units, while simple comparison of data needs single time unit. Then the total time is $C_1 D N_I N_Q$.

In algorithm B, extension to $F(I)$ is required for each I. Since the thesaurus is used for each of $p, a_1, ..., a_n$, then the number of propositions in $F(I)$ becomes $M^{n+1}N_I$. The number of comparison between Q and $F(I)$ is $M^{n+1}N_I + N_Q$. Assuming that the time for the sorting k data is $C_0 k \log k$, and that access to the thesaurus requires C_2 time units for obtaining $F(I)$, then the total amount of computation is $D(M^{n+1}N_I + N_Q + C_0 M^{n+1} N_I \log(M^{n+1}N_I) + C_2 N_I)$.

The number of comparison for $G(Q)$ and I in algorithm C is $M^{n+1}N_Q + N_I$. The total amount of computation is $D(M^{n+1}N_Q + N_I) + (C_2 N_Q + C_0 M^{n+1} N_Q \log(M^{n+1}N_Q))$, where the terms in the second parentheses can be neglected in practice, since they are not multiplied by D.

This estimation shows that if N_I and N_Q are not very different, algorithm C is more efficient than algorithm B. In comparison between algorithms A and C, algorithm A is better when reference to F does not require much time (i.e., C_1 is small); otherwise algorithm C should be used.

Remark. *Null values frequently occur in the propositions, such as (use, fuzzy index, $*$), where the character $*$ stands for the null value. This null value means that $*$ can be matched with any word in W, and the above method can be used in the presence of null values.*

4. Illustrative examples

Let us consider two examples: one from document retrieval and the other from image retrieval.

Example 3.

Three documents whose indices are the following I_1, I_2, I_3 are assumed to be given.

I_1: y_1 =(1.0, for, fuzzy clustering, document retrieval)
I_2: y_2 =(1.0, using, information retrieval, fuzzy sets)
I_3: y_3 =(1.0, based on, content analysis, fuzzy graph)

for which the query is

Q: x =(1.0, based on, document retrieval, fuzzy indices).

The thesaurus F which is reflexive and symmetric is given by the following.

F(content analysis, document retrieval) = 0.4
F(content analysis, information retrieval) = 0.4
F(document retrieval, information retrieval) = 0.9
F(fuzzy sets, fuzzy indices) = 0.7
F(fuzzy sets, fuzzy clustering) =0.7
F(fuzzy sets, fuzzy graph) = 0.7
F(fuzzy indices, fuzzy clustering) = 0.5
F(fuzzy indices, fuzzy graph) = 0.6
F(fuzzy clustering, fuzzy graph) = 0.8
F(using, based on) = 0.8
F(using, for) =0.7
F(based on, for) = 0.7

where $F(a, a) = 1$, $a \in W$ is omitted. Zero values of the relation is also omitted.

Now let us apply algorithms A, B, and C.

Algorithm A:

$$L(x, y_1) = \min[F(\text{based on}, \text{for}), F(\text{document retrieval}, \text{fuzzy clustering}),$$
$$F(\text{fuzzy indices}, \text{document retrieval})]$$
$$= \min[0.7, 0, 0] = 0.$$
$$t_1(I_1, Q) = L(x, y_1)/1.0 = 0.$$
$$L(x, y_2) = \min[F(\text{using}, \text{based on}),$$
$$F(\text{information retrieval}, \text{document retrieval}),$$
$$F(\text{fuzzy sets}, \text{fuzzy indices})]$$
$$= \min[0.9, 0.9, 0.7] = 0.7$$
$$t_1(I_2, Q) = L(x, y_2)/1.0 = 0.7$$
$$L(x, y_3) = \min[F(\text{based on}, \text{based on}),$$
$$F(\text{content analysis}, \text{document retrieval}),$$
$$F(\text{fuzzy graph}, \text{fuzzy indices})]$$
$$= \min[1.0, 0.4, 0.6] = 0.4$$
$$t_1(I_3, Q) = L(x, y_3)/1.0 = 0.4$$

Contents of the three documents are more or less related to the query. Nevertheless, the matching degree between I_1 and Q is zero, since between the relation *for* and *based on*, the orders of the arguments are different. In the case of *for*, the method to be used is the second argument and the subject for application is the third, whereas *based on* has the subject for the second argument and the method for the third. To avoid this failure in matching, I_1 should be extended to

I_1': y_1 =(1.0, for, fuzzy clustering, document retrieval)

y_1' =(0.9, using, document retrieval, fuzzy clustering)

where the second record y_1' is obtained using fuzzy thesaurus. Note that $t_1(I_1', Q) = 0.5$.

In the case of algorithm B, $F(I_1)$ has 36 records, which are omitted here. Since no common character sequence exists between Q and $F(I_1)$, we have $t_1(I_1, Q) = 0$. The record in $F(I_2)$ having the same character sequence with the record in Q is (0.7, based on, document retrieval, fuzzy indices), hence $t_1(I_2, Q) = 0.7$. For $F(I_3)$, the record of the same character sequence with the record in Q has $\mu = 0.4$, and therefore $t_1(I_3, Q) = 0.4$.

Algorithm C extends Q into 36 records in $G(Q)$, which includes (0.7, using, information retrieval, fuzzy sets, x) and (0.4, based on, content analysis, fuzzy graph, x). Each of these two records has the common character sequence with a record in $F(I_2)$ and $F(I_3)$, respectively. Accordingly we have $t_1(I_2, Q) = 0.7$ and

$t_1(I_3, Q) = 0.4$. No common character sequence exists between records in $G(Q)$ and those in I_1, and therefore $t_1(I_1, Q) = 0$.

Example 4.
We consider Example 2 and let I: { y_1 =(1.0, in-front-of, person, house), y_2 =(1.0, on-the-top-of, flag, house) }. Let us calculate matching degree between I and Q: x =(1.0, side, woman, house). Assuming that F(side,in-front-of)=0.6 and F(person,woman)=0.9, we have $L(x, y_1) = 0.6$ and $L(x, y_2) = 0$, hence

$$t_1(I, Q) = \max[0.6, 0]/1.0 = 0.6$$

5. Conclusion

A considerable part of subjects of scientific papers in engineering can be summarized simply: use of some method to a specific application. For example, the proposition of the subject of the present paper is *(based on, information retrieval, fuzzy propositional index)*. Thus, it is not only natural to describe the content of a document by a set of such propositions, but also useful in categorizing scientific documents by the use of a *relation* between a *method* and an *application*.

There are different techniques which deal with partial matching degree encountered in information retrieval. Here the degree is regarded as membership of fuzzy retrieval. In general, fuzzy retrieval is the superior method of such techniques, since fuzzy retrieval solely admits logical operations and quantitative information.

The present technique not only shows matching between a user query and an index, but also matching between two indices. Thus, when we have a document and wish to find associated documents of similar subjects to the former, we can perform *similarity retrieval* using the index of the former document as Q.

We have also introduced a method of image retrieval. The propositional index for images implies that results in recognition of images can be used as the index, and similarity retrieval of images using such automatically generated indices can be performed.

Such a set of propositions indicates the use of a fuzzy database system for storing and retrieving data. Although we do not discuss fuzzy database systems here, relation between the fuzzy propositional index and fuzzy database systems is an interesting subject for further research. As noted in Example 3, inference is required for retrieval of propositions. Thus, fuzzy deductive database is adequate for the present retrieval technique.

We acknowledge that this work has been partially supported by the Grant in Aid for Scientific Research (01850086,01550321).

References

J. R. Anderson, Cognitive Psychology and Its Implications, Freeman, New York, 1980.

S. Miyamoto, T. Miyake, and K, Nakayama, Generation of a pseudothesaurus for informaion retrieval based on cooccurrences and fuzzy set operations, IEEE Trans., on Syst. Man, and Cybern., Vol.13, No.1, pp.62-70, 1983.

M. Sakauchi, Image retrieval techniques, The Journal of the Institute of Electronics, Information, and Communication Engineers, Japan, Vol.71, No.9, pp.911-914, 1988 (in Japanese).

INSTITUTE OF INFORMATION SCIENCES AND ELECTRONICS
UNIVERSITY OF TSUKUBA
TSUKUBA-SHI, IBARAKI 305 JAPAN

REPORT DOCUMENTATION PAGE	REPORT NUMBER ISE-TR-90-85

TITLE

Document Retrieval and Image Retrieval Based on Fuzzy Propositional Index

AUTHOR(S)

S. Miyamoto, *N. Konishi, and T. Miyake

Institute of Information Sciences and Electronics
University of Tsukuba, Ibaraki 305, Japan

*Master's Program in Sciences and Engineering
University of Tsukuba, Ibaraki 305, Japan

REPORT DATE 1990-9-27	NUMBER OF PAGES 12
MAIN CATEGORY Information Search and Retrieval	CR CATEGORIES H.3.3

KEY WORDS fuzzy propositional index, document retrieval, image retrieval, fuzzy retrieval, matching function, algorithms.

ABSTRACT

The aim of the present paper is to propose fuzzy propositional index and retrieval for documents or images. A set of propositions represents content of a document or an image. A query of the same form of fuzzy propositions is matched with fuzzy propositional indices for a set of documents or images using a matching function, which is a fuzzy relation between two sets of fuzzy propositions. The matching degree by the fuzzy relation is interpreted as the membership value in the retrieved set. Three algorithms are derived for the matching function which is based on a fuzzy set model, and their efficiency is estimated. Illustrative examples for document retrieval and image retrieval are given.

SUPPLEMENTARY NOTES