Jian Dao 2 (1994): 35-72

A FRAMEWORK FOR THE COMPUTATIONAL ANALYSIS OF TEXT IN BIBLICAL STUDIES

MITSURU SAKITANI

Kyoto University, Japan

4-1-101 Maruhashi-cho, Nishinomiya, Hyogo 662, Japan

Recent developments in biblical studies are so remarkable that one cannot easily comprehend the whole activities, especially in new trends such as structural exegesis, rhetorical criticism or literary criticism. Indeed we have numerous methods of biblical exegesis today, but one may recognize that these methods have their roots in the relatively old theoretical basement in the sixties to the seventies. Furthermore, because of the essential nature of biblical texts, it has been thought to be difficult to apply computational analysis such as artificial intelligence (AI)[1] or logic programming to biblical exegesis. Our aim in the present study is to fill a gap of the availability of scientific technology between the biblical studies and other research fields around them. Certainly we have already obtained in part efficient tools such as word search softwares or digital versions of text database in CD-ROM, but our aim is more advantageous, that is, to proceed from such a level limited to process only character codes of texts toward the artificial intelligence technology in which we can automatically carry out semantic analysis of the religious statements expressed in biblical texts by using mathematical logic and AI-oriented programming language.

In this study, for the first time in the biblical studies, we introduce a new sophisticated method that is based on the mathematical formal system equipped with the computational processing of logical calculation. Our method called the framework for the computational analysis of text

[1]Abbreviations used in this paper are AI: artificial intelligence; KFS: kernel formal system; FCAT: framework for the computational analysis of text; FOL: first-order predicate logic.

(FCAT) consists of three steps. First, we define a minimum formal system called a kernel formal system (KFS), which is easily transported to other formal systems. Second, we try to extend KFS to other two formal systems: logic programming (Prolog) and fuzzy set theory. Third, finally, we apply it to conventional biblical methods: redaction criticism and structural exegesis.

For the FCAT to extend in such a way, first, we can describe the different methods of biblical exegesis in a unified formal language so that scholars who use the different biblical methods can share their expert knowledge with other scholars by the common formal language. Second we can obtain a powerful computational tool by using Prolog. Third, by the fuzzification of information, we can attain theoretical advantage in analyzing the essentially ambiguous biblical texts in a more suitable and flexible way.

Definition of a Kernel Formal System

1.1 *Definition of a Kernel*

Our FCAT contains a *kernel formal system* (KFS) consisting of a minimal set of basic components: a constant, a variable, a relation, a predicate, a universe and a space.

Definition 1: A kernel formal system consists of following components:

a constant (a, b, c, ...), an individual entity;

a variable (x, y, z, ...), a symbol for constants;

a relation (f, g, ...), which relates constants or variables to each others;

a predicate (p, q, ...), which defines attributes of entities;

a universe (u), a domain that contains entities of text;

a space (s), a textual, semantic domain of text.

Either simple objects (constants and variables) or complex objects related to each other by relations or predicates are assigned (or mapped) to domains in their respective universes. Then they are further assigned to respective spaces as their semantic domains. END[2]

An entity is any individual object (either agent or thing). If an entity is a single existent object, we call it a constant, and if an entity represents a common "place-holder"[3] of many existent objects, it is defined as a variable.

Example 1: If an entity "John" is restricted to the "John the baptizer" in its context, this entity is a constant. On the contrary, the expression "John" that is found in the New Testament includes really plural "John's" such as "John the baptizer," "John in the Revelation," "John as the Evangelist of the fourth Gospel," and so on. Thus one should consider the "John" as a variable that can be substituted by other "John's." Therefore the variable "John" can be rewritten by a more general symbol, for example "*x*", to specify a variable, and the variable *x* can be instantiated by any "John's" above mentioned. END

Plural entities are linked to each other in a certain manner. This linkage is defined by a relation. The notation of a relation is usually defined either by a prefix notation 'fxy' or by an infix notation 'xfy', where 'x' and 'y' are entities whereas 'f' is a function symbol. We often use here the prefix notation because of its broad acceptance in logic, linguistics and computer science. In a quite similar manner, the feature of the entities is defined by a predicate. In this case the number of entity is either single or plural.

Example 2: The expresshnion "John is a baptizer" is translated into the prefix notation: `is_a(Jo, baptizer)`, where "`is_a`" is a predicate while "`John`" and "`baptizer`" are two entities linked to each other by "`is_a`". Note that the entities are surrounded by two additional symbols "(" and ")". This notation can be rewritten in a more general form: `is_a(John, baptizer)` $\Leftrightarrow p(a, b)$ where p is a predicate "`is_a`", and a and b are constants "`John`" and "`baptizer`"

[2]The end of a definition is in general indicated by a small square. In this study, however, due to the limited availability of mathematical fonts, we use the symbol "END" in stead of a small square.

[3]V. Sperschneider and G. Antoniou, *Logic: A Foundation for Computer Science* (Reading: Addison Wesley, 1991), 8.

respectively. The new symbol "⇔" designates logical equivalence. END

In general, a notation $p(x_1,...,x_n)$ is called an *atomic formula* or in brief an *atom*, where p is a predicate (or a relation) and $x_1,...,x_n$ are entities with *arity* n. The arity is a number of entities and is expressed by an integer.

In addition, we define a *term* as either a constant or a variable or an atomic formula.

Example 3: The formula `disciple_of(x, y)` is an atom designating that the "disciple of *x*" is "*y*". Given x = Jesus, we obtain y = {Peter, James, Andrew, ...}. END

A number of entities are assigned to a *universe*.

Example 4: In a universe $u_{john4:7\text{-}26}$ of John 4:7-26, there are two entities, Jesus and the Samaritan woman. This is described as follows: $j, s \in u_{john4:7\text{-}26}$, where j and s are a constant "Jesus" and a constant "the Samaritan woman" respectively. The new symbol "∈" designates "is an element of" or "is contained in". END

Any text portion belongs to its respective *space*, in which any entity is assigned to its domain of meaning. It should be noted that spaces are supposed in any part or a whole of text not only in a syntagmatic order but also in a vertically overlaying order.

Example 5: According to Z. Kato,[4] the text of Mark 4:1-9 is divided into three parts: the verses 1-2 are attributable to the Evangelist Mark's redaction, a universe u_1; the verses 3-8 is a parable derived from a tradition material, a universe u_2; and the verse 9 may be another tradition material, a universe u_3. Thus different textual, semantic spaces could be postulated as follows: a space s_1: vv.1-2 (redaction); a space s_2: vv.3-8 (tradition 1); a space s_3: v.9 (tradition 2); a space s_4: vv.1-8 (redaction + tradition 1); a space s_5: vv.1-9 (redaction + tradition 1 + tradition 2) and so on. END

---

[4]Zenji Kato, *Die Völkermission im Markusevangelium: Eine redaktionsgeschichtliche Untersuchung* (Bern: Peter Lang, 1986), 36-37.

A space could be postulated in various ways. Thus different spaces can be supposed according to the view points on which interpreters are focusing their interest.

Example 6: The same text, whose entities (objects) are included in a universe u_{text}, may be interpreted in multiple ways: (a) from the point of view of source criticism (its textual space: s_{source}) along with a historical development; (b) from that of redaction criticism ($s_{redaction}$) based on the redactor's intention postulated; (c) from that of a historical reader response (s_{reader}) to the text at the final stage; (d) from that of pragmatics ($s_{pragmatics}$) in a situation of modern readers; or (e) from that of structuralism ($s_{structuralism}$) aiming at a universal, a-historical understanding. The results of such interpretations, that is, the meanings of the same text, are thus highly different because of their different assignments of the universe of the same text, u_{text}, to the different semantic spaces, s_{source}, $s_{redaction}$, s_{reader}, $s_{pragmatics}$ or $s_{structuralism}$. Thus the differences in results of interpretation among scholars who use different exegetical methods are ascribable to the different setting of their own semantic spaces, although its textual universe is common. END

1.2 *Significance of the Kernel Formal System*

Our KFS is characterized by several features as follows.

(1) The components of the KFS are trivial ones except for the newly introduced concept, a *space*. Our purpose to afford a theoretical framework for the computational analysis of biblical, historical texts will be just realized through the manipulation of the concept of space as a theoretical device to ground a whole complexity of historical texts. In general, historical texts such as Bible consist of a number of parts, or subtexts, which may originate from culturally, linguistically, geographically and temporally different environments, and are read even by modern readers in a situation thoroughly detached from the original historical context. Thus one needs a device to access any aspect of complexities of historical texts. Our "space" is one of such a device and even is precisely defined in a mathematical formal language.

(2) Besides historical complexities, we may face the differences in its aim and its domain of meaning among different methods in biblical studies. Since in each method an interpreter assigns entities contained in the universes of the text to the respective spaces that are appropriate

to his or her own purpose or interest, it may be difficult or even impossible for scholars standing on different methods to communicate their scholarly information with each other. Our theoretical device, a semantic space, with a simple formal system (KFS) may offer them a common ground to communicate or to translate one theoretical system to others.

(3) Our formal system is described without being rigidly structured. Thus, along with its formal description, it has rich potentialities to extend itself to other more elaborate formal systems such as first-order predicate logic, logic programming, fuzzy set theory and so on.

In the next section, we try to extend our KFS to other formal systems structured in a more rigid way, and it will be shown that our framework, when KFS is extended to a special method, has rich expressive power and efficiency in the computational analysis of biblical texts.

Theoretical Extension of the Kernel Formal System

First, we extend the KFS to other two formal systems, logic programming and fuzzy set theory. Second, we show that through such extensions our FCAT works well in different applied fields of conventional biblical methods such as redaction criticism or structural exegesis.

2.1. *Logic Programming (Prolog)*

Logic programming has been established, in general, on first-order predicate logic (FOL) by using SLD-resolution mechanism. Its computer language, Prolog,[5] which was first designed in 1973 by A. Colmerauer

[5]Cf. Leon Sterling and Ehud Shapiro, *The Art of Prolog* (Cambridge [Mass.]: MIT Press, 1986); Patrick Saint-Dizier, *An Introduction to Programming in Prolog* (New York: Springer-Verlag, 1989); Ivan Bratko, *Prolog Programming for Artificial Intelligence* (Wokingham: Addison-Wesley, ^{2}1990); Tu Van Le, *Techniques of Prolog Programming with Implementation of Logical Negation and Quantified Goals* (New York: John Wiley & Sons, 1993); George F. Luger and William A. Stubblefield, *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* (Redwood: Benjamin/Commings, ^{2}1993).

and his Gourpe d'Intelligence Artificielle de l'Université d'Aix-Marseille,[6] however, is different from FOL in several points. First, its semantics is based on the *closed world assumption* (CWA) or *negation as failure* (NF), that is, "anything is false if all the opposite cases are not proved to be true."[7] Second, although in Prolog each variable is in general universally quantified, Prolog does not always reflect the exact quantification of FOL. Third, although Prolog works usually in a sound and complete manner by its backtracking mechanism, Prolog often deduces either false conclusions due to its NF mechanism or incomplete results due to its cut mechanism.

In Prolog, logical notations are parallel to those of FOL:

Prolog:	,	;	:-	not
first-order predicate logic:	$\wedge$	$\vee$	$\leftarrow$	$\neg$
English:	and	or	only if	not

Prolog is based on Horn clause but differents notations are used.

Definition 2 (Horn clause): Let $X_1,...,X_n$ be variables, $P_1,...,P_m$ atoms of premise, and C an atom of conclusion, then a Horn clause is one of following formulas.

(H.1) $\forall(X_1,...,X_n)(C \leftarrow P_1 \wedge ... \wedge P_m)$

(H.2) $\forall(X_1,...,X_n)(C \leftarrow \ )$

(H.3) $\forall(X_1,...,X_n)(\ \leftarrow P_1 \wedge ... \wedge P_m)$

In Horn clauses, the conclusion is a single atom and the premises or condition parts consist of conjunctively connected atoms, and variables are universally quantified. END

Definition 3 (Prolog Rule, Fact, Goal): In Prolog, the Horn clauses are

[6]Alain Colmerauer et al., *Un système de communication homme-machine en français*. Research Report, Groupe d'Intelligence Artificielle (Marseilles: Université d'Aix-Marseille II, 1973).

[7]Cf. Saint-Dizier, *Introduction to Prolog*, 72; Luger and Stubblefield, *Artificial Intelligence*, 213.

rewritten as follows.

(P.1) [Rule] C :- $P_1,\dots,P_m$.

(P.2) [Fact] C.

(P.3) [Goal] ?- $P_1,\dots,P_m$.

where C is a conclusion and $P_1,\dots,P_m$ are premises or conditions.

In Prolog, the conclusion part is called *head* and the condition part is called *body* or *constraint*. END

A constant, a variable, a predicate and a term in KFS are straightforwardly defined in Prolog.

Example 7: The formulas of the KFS shown in the example 3 can be translated into a Prolog language:

```
[fact]  disciple_of(jesus, peter).
        disciple_of(jesus, james).
        disciple_of(jesus, andrew).
[goal (query or question)] ?- disciple_of(jesus, Who).
[answer] Who = peter; Who = james;
         Who = andrew; no                END
```

Prolog provides us a powerful tool to automatically process logical calculi even in a sound and complete way, if programs are well designed. Thus when KFS is extended to Prolog, biblical scholars can obtain a power to mechanically analyze semantics of biblical texts on the basis of the mathematically defined formal logic system that can be not only sound but also complete. The completeness of computational system is really a necessary condition in biblical studies, since a complete program returns us the whole answers by searching all the cases in its knowledge database. Our FCAT, when extended to Prolog, therefore, can afford an invaluable efficient power to biblical scholars. Examples of powerful Prolog programming will be later demonstrated.

2.2. *Fuzzy Set Theory*

Fuzzy set theory, established by L.A. Zadeh in 1965,[8] also provides us another theoretical possibility. Recent development of studies in fuzzy sets has proved that fuzzy set theory is a powerful tool in processing real-world problems whose information is inevitably ambiguous, that is, imprecise and incomplete.[9] Thus fuzzy set theory may well fit our aim to analyze historical texts that are essentially surrounded by ambiguous conditions or constraints that are to be determined often heuristically without any evident proof.

Definition 4 (Fuzzy set): Let x be an element of a domain X of space of points. A *fuzzy set* (*class*) A in X is characterized by a *membership* (*characteristic*) *function* $\mu_A(x)$, which associates with each point in X a real number in the interval [0, 1], with the value of $\mu_A(x)$ at x representing the "grade of membership" of x in A.

$\mu_A(x) \rightarrow [0, 1]$ END

According to Saint-Dizier and Li & Liu,[10] there are several ways to implement fuzzy set theory to Prolog. Probably the easiest way is to add to the last of body a predicate that defines a membership function. Our way to implement fuzzy sets to Prolog is shown in the following example of a toy version of redaction criticism.

Example 8 (Redaction Criticism by Fuzzy Prolog): Let `space_trad` and `universe_trad` be predicates for a space of tradition and a universe of tradition respectively, and `{t1, t3, t4, t5, t7}` be objects of tradition related by a predicate `obj_trad`. Let `space_red` and `universe_red` be predicates for a space of redaction and a universe of redaction respectively, and `{r1, r2, r5, r6, r7}` be objects of redaction related by a predicate `obj_red`. If a membership

[8]Lotfi A. Zadeh, "Fuzzy Sets," *Information and Control* 8 (1965): 338-353.

[9]Cf. Lotfi A. Zadeh, *Fuzzy Sets and Applications*: Selected Papers by L.A. Zadeh. Ed. by R.R. Yager et al. (New York: John Wiley and Sons, 1987); Lotfi A. Zadeh and Janusz Kacprzyk (eds.), *Fuzzy Logic for the Management of Uncertainty* (New York: John Wiley & Sons, 1992).

[10]Saint-Dizier, *Introduction to Prolog*, 136-139; Deyi Li and Dongbo Liu, *A Fuzzy Prolog Database System* (New York: John Wiley & Sons, 1990);

function from the universe of tradition to the space of tradition and that from the universe of redaction to the space of redaction are expressed by `memb_f(trad_trad)` and `memb_f(red_red)` respectively, and if a membership function from the universe of tradition to a space of redaction and that of the association between objects of tradition and of redaction under redactor's intention are expressed by `memb_f(trad_red)` and `memb_f(tradred_red)` respectively, then the grades of membership functions may be supposed to be `red_red ≥ tradred_red ≥ trad_red`. Let associated objects between tradition and redaction under the intention of the redactor be `association(t1,r1)`, `association(t5,r5)` and `association(t7,r7)`. Thus we obtain following Prolog clauses:

```
/* Fact and Rule */

% Fact:

   obj_trad(t1). obj_trad(t3). obj_trad(t4).

   obj_trad(t5). obj_trad(t7).

   obj_red(r1). obj_red(r2). obj_red(r5).

   obj_red(r6). obj_red(r7).

   association([t1,r1]). association([t5,r5]).

   association([t7,r7]).

   memb_f(trad_trad). memb_f(red_red).

   memb_f(tradred_red). memb_f(trad_red).

% Rule:

   universe_trad(X):-obj_trad(X).

   universe_red(Y):-obj_red(Y).

   space_trad(X,trad_trad):-

      universe_trad(X),memb_f(trad_trad).

   space_red(Y,red_red):-
```

```
      universe_red(Y),memb_f(red_red).
   space_red([X,Y],tradred_red):-
      association([X,Y]),
      memb_f(tradred_red).
   space_red(X,trad_red):-
      universe_trad(X),memb_f(trad_red).
   set_of_memb_f(red_red,
                  tradred_red,trad_red).
   max_grade(A):-set_of_memb_f(A,B,C).
   mid_grade(B):-set_of_memb_f(A,B,C).
   min_grade(C):-set_of_memb_f(A,B,C).
/* Query and Answer */
% Query 1:
?- setof(pair(Obj,Red_memb_f),
      space_red(Obj,Red_memb_f),List).
   List =
   [pair(r1, red_red), pair(r2, red_red),
   pair(r5, red_red), pair(r6, red_red),
   pair(r7, red_red),
   pair(t1, trad_red), pair(t3, trad_red),
   pair(t4, trad_red),
   pair(t5, trad_red), pair(t7, trad_red),
   pair([t1, r1], tradred_red),
   pair([t5, r5], tradred_red),
   pair([t7, r7], tradred_red)]
% Query 2:
?- setof(Obj,(space_red(Obj,Memb_f),
      max_grade(Memb_f)),
      Set_of_max_memb_f).
   Set_of_max_memb_f = [r1, r2, r5, r6, r7]
% Query 3:
?- setof(Obj,(space_red(Obj,Memb_f),
      mid_grade(Memb_f)),
      Set_of_mid_memb_f).
   Set_of_mid_memb_f =
      [[t1, r1], [t5, r5], [t7, r7]]
% Query 4:
?- setof(Obj,(space_red(Obj,Memb_f),
      min_grade(Memb_f)),
      Set_of_min_memb_f).
   Set_of_min_memb_f = [t1, t3, t4, t5, t7]
```

By the query 1, all the objects are shown with their respective values of fuzzy membership function to the redactor's space, and by the queries 2, 3 and 4, the respective objects at the maximum, middle and minimum grades of the fuzzy membership functions to the redactor's space are selectively deduced. In these cases, the membership functions that map the objects to the redactor's space, that is, `red_red`, `tradred_red` and `trad_red`, are treated as constants for the sake of simplification. In order to process real-world objects, however, the values of membership functions should be expanded to variables, because each object has its own value of membership function. END

Since in classical fuzzy set theory the conjunction of fuzzy sets are determined by the maximum value of membership functions, the maximum value of the redactor's membership function `red_red` indicates that the whole objects both in traditions and in redactions are controlled under the redactor's intention that is well revealed in the redactional portions with the maximum membership function `red_red`.

When we extend our KFS to fuzzy sets, the newly introduced textual, semantic space may be well grounded by fuzzy set theory. In our fuzzy Prolog version of redaction criticism, three groups of fuzzy membership function `red_red`, `tradred_red` and `trad_red` can reflect in detail the redactional tasks along with the redactor's intention. On the contrary, in conventional redaction criticism, these three redactional aspects are not fully evaluated because of the limited ability of discrimination due to its incomplete, intuitive method. Thus our FCAT, when extended to fuzzy Prolog, could enrich the potentiality of redaction criticism not only with the expressive power to analyze in detail the redaction through its mathematically defined formal system but also with a powerful automated processing of the objects under the redactor's intention by the computational analysis.

Application to the Biblical Methods

Our final task is to apply the FCAT to conventional biblical methods. We have already shown a toy version of redaction criticism by fuzzy Prolog. Thus in the following section, first, we want to concentrate our task on the application of FCAT to structural exegesis, and then we will discuss in short about possibilities of FCAT application to other methods.

3.1. *Structural Exegesis*

There are two kinds of structural exegesis in biblical studies.[11] The one is the well-known method of structural exegesis in biblical

[11]Cf. Daniel Patte, *Structural Exegesis for New Testament Critics* (Minneapolis: Fortress Press, 1990), 3-4; id., *The Religious Dimensions of Biblical Texts: Greimas's Structural Semiotics and Biblical Exegesis* (Atlanta: Scholars Press, 1990), 25-72.

studies such as X. Léon-Dufour, R. Barthes, D. Patte, Groupe d'Entrevernes, A. Gueuret and M.W.G. Stibbe,[12] established on Greimas's early work,[13] while the other is now under a process of establishment by D. Patte[14] and is mainly based on Greimas's latest work.[15] Our task is, of course, not the introduction of Patte's recent ideas on structural exegesis, but the implementation of our KFS to structural exegesis from the perspective of FCAT.

(3.1.1) Early Version of Structural Exegesis

Strictly speaking, the actantial model is merely a part of structural exegesis that has been based on the early version of Greimas's semiotics. Greimas seems to have adopted various resources from F. de Saussure, C. Lévi-Strauss, V. Propp, R. Jacobson and N. Chomsky,[16] while semiotic

---

[12]Xavier Léon-Dufour (ed.), *Exégèse et herméneutique* (Paris: Seuil, 1971); Roland Barthes et al., *Structural Analysis and Biblical Exegesis* (Pittsburgh: Pickwick Press, 1976); Daniel Patte, *What is Structural Exegesis?* (Philadelphia: Fortress Press, 1976); Daniel Patte (ed.), *Semiology and Parables: Exploration of the Possibilities Offered by Structuralism for Exegesis* (Pittsburgh: Pickwick Press, 1976); Groupe d'Entrevernes, *Signes et paroles: Sémiotique et texte évangélique (Paris: Seuil, 1977); Agnès Gueuret, La mise en discours: Recherches sémiotiques à propos de l'Evangile de Luc* (Paris: Cerf, 1987); Mark W.G. Stibbe, "'Return to Sender': A Structuralist Approach to John's Gospel," *Biblical Interpretation* 1 (1993): 189-206.

[13]Algirdas J. Greimas, *Sémantique structurale* (Paris: Larousse, 1966).

[14]Cf. Patte's two books, *Structural Exegesis for NT Critics* and *Religious Dimensions.*

[15]Algirdas J. Greimas and Joseph Courtés, *Sémiotique: Dictionnaire raisonné de la théorie du langage* I & II (Paris: Hachette, 1979 [I], 1986 [II]). The volume one was translated into English: tr. D. Patte and M Rengstorf, *Semiotics and Language* (Bloomington: Indiana University Press, 1982).

[16]Ferdinand de Saussure, *Cours de linguistique générale* [new edition] (Paris: Payot, 1972); Claude Lévi-Strauss, *Anthropologie structurale* (Paris: Plon, 1958); Vladimir Propp, *Morphologie du conte* (Paris: Seuil, 1965); R. Jacobson, *Essais de linguistique générale* (Paris: Minuit, 1963); Noam Chomsky, *Syntactic Structures* (The Hague: Mouton, 1957).

studies in North-American biblical exegesis has been strongly affected by the actantial model as a part of Greimas's semiotics.

In order to implement FCAT to structural exegesis, we choose two theoretical models: the one is the model of structural exegesis precisely presented by Patte,[17] and the other is the frame that is one of AI technology of knowledge representation by M. Minsky,[18] and we show an example in which the result of Patte's structural exegesis on the parable of the good Samaritan in Luke 10:30-15 is rewritten by a structured knowledge representation technique, that is, the frame written in Prolog.

Example 9 (Structural Exegesis by Frame): First, the model of Patte's early version of structural exegesis is summarized in a prototype. Then the result of Patte's analysis of Luke 10:30-35 is represented by frames in a form of an expert knowledge database.

```
/* Prototype */

%  frame(name(kind_of_sequence),

%     sort(sequence_number),

%     lexie('verse_of_the_luke_chapter_10'),

%     upper_sequence([correlated_sequence,

%                         topical_sequence])

%     syntagm([contract_syntagm1,contract_syntagm2,

%                  disjunction_conjunction_syntagm,

%                  performance_syntagm1,

%                  performance_syntagm2,

%                  performance_syntagm3]),
```

[17]Patte, *What is Structural Exegesis?* 35-52.

[18]Marvin Minsky, “A Framework for Representing Knowledge,” in P. H. Winston (ed.), *Psychology of Computer Vision* (Cambridge (Mass.): MIT Press, 1975).

```
%     function([[movement;arrival;
%                     departure;return],
%                 [disjunction;conjunction],
%                 [mandating;acceptance;refusal],
%                 [confrontation;association],
%                 [domination;submission],
%                 [communication;reception],
%                 [attribution;deprivation]]),
%     actant([subject(_),object(_),
%                 sender(_),receiver(_),
%                 helper(_),opponent(_)]),
%     result(_),
%     others(_)).
/* Knowledge Base (Fact) */
   frame(name(initial_correlated_sequence),
      sort(sequence1),
      lexie('30a'),
      upper_sequence(correlated_sequence),
      syntagm(disjunction_conjunction_syntagm),
      function(movement),
      actant([subject(man),_,_,_,_,
            opponent(robbers)]),
      result([movement(not_completed),
            sequence(interrupted)]),
      others(place(from_jerusalem,to_jericho))).
   frame(name(subsequence),
      sort(sequence2a),
      lexie('30b'),
      upper_sequence(correlated_sequence),
      syntagm(performance_syntagm1),
      function(confrontation),
      actant([subject(robbers),
            object(belongings_of_the_man),
            _,receiver(robbers),
            helper(number_of_robbers),
            opponent([man,vigor])]),
      result([confrontation(man,robbers),
            sequence(from_initial_
                  correlated_sequence,
            to_performance_syntagm2)]),
      others([])).
   frame(name(subsequence),
      sort(sequence2b),
      lexie('30b'),
      upper_sequence(correlated_sequence),
      syntagm(performance_syntagm2),
      function(domination_submission),
      actant([subject(robbers),
            object(belongings_of_the_man),
            _,receiver(robbers),
            helper(number_of_robbers),
            opponent([man,vigor])]),
      result([domination(robbers,
                  beating_of_the_man),
            sequence(from_performance_syntagm2,
                  to_performance_syntagm3)]),
      others([])).
   frame(name(subsequence),
      sort(sequence2c),
      lexie('30c'),
      upper_sequence(correlated_sequence),
      syntagm(performance_syntagm3),
      function(attribution),
      actant([subject(robbers),
            object(belongings_of_the_man),
            _,receiver(robbers),_,_]),
      result([attribution(stripping_of_the_man),
            sequence(end_of_
                  performance_syntagms)]),
      others([])).
   frame(name(topical_sequence1),
      sort(sequence3),
    lexie('31'),
    upper_sequence(topical_sequence),
    syntagm(contract_syntagm1),
    function(mandating),
    actant([subject(priest),
          object([health,vigor]),
          _,receiver(man),
          _,opponent([robbers,
               the_effects_
                    of_their_actions])]),
    result([volition(not_established),
          sequence(interrupted)]),
    others([])).
 frame(name(topical_sequence2),
    sort(sequence4),
    lexie('32'),
    upper_sequence(topical_sequence),
    syntagm(contract_syntagm1),
    function(mandating),
    actant([subject(priest),
          object([health,vigor]),
          _,receiver(man),
          _,opponent([robbers,
               the_effects_
```

```
                        of_their_actions])]),
      result([volition(not_established),
            sequence(interrupted)]),
      others([])).
   frame(name(topical_sequence3),
      sort(sequence5a),
      lexie('33'),
      upper_sequence(topical_sequence),
      syntagm(contract_syntagm1),
      function(mandating),
      actant([subject(samaritan),
            object([health,vigor]),
            _,receiver(man),
            _,opponent([robbers,
                  the_effects_
                        of_their_actions])]),
      result([volition(established),
            sequence(to_disjunction_
                  conjunction_syntagm)]),
      others([])).
   frame(name(topical_sequence3),
      sort(sequence5b),
      lexie('34a'),
      upper_sequence(topical_sequence),
    syntagm(disjunction_conjuction_syntagm),
    function(movement),
    actant([subject(samaritan),
        object([health,vigor]),
        _,receiver(man),
        _,opponent([robbers,
            the_effects_
                of_their_actions])]),
    result([movement(completed),
        sequence(to_performance_syntagm1)]),
    others(place(from_road,to_the_man))).
  frame(name(topical_sequence3),
    sort(sequence5c),
    lexie('34b-35'),
    upper_sequence(topical_sequence),
    syntagm(performance_syntagm1),
    function(confrontation),
    actant([subject(samaritan),
        object([health,vigor]),
        _,receiver(man),
        helper([know_how,oil,wine,donkey,
            money,innkeeper]),
        opponent([robbers,
            the_effects_
```

```
                         of_their_actions])]),
      result([confrontation(accepted),
            sequence(to_performance_syntagm2)]),
      others([])).
   frame(name(topical_sequence3),
      sort(sequence5d),
      lexie('34b-35'),
      upper_sequence(topical_sequence),
      syntagm(performance_syntagm2),
      function(domination_submission),
      actant([subject(samaritan),
            object([health,vigor]),
            _,receiver(man),
            helper([know_how,oil,wine,donkey,
                  money,innkeeper]),
            opponent([robbers,
                  the_effects_
                        of_their_actions])]),
      result([sequence(end_
            of_all_the_sequences)]),
      others([])).
/* Queries and Answers */
% Query 1:
?- setof((A,B,C,D),
          frame(name(A),sort(B),lexie(C),
          upper_sequence(D),_,_,_,_,_),
          All_sequences).
All_sequences =
[(initial_correlated_sequence,sequence1,
  '30a',correlated_sequence)];
[(subseqeunce,sequence2a,'30b',
  correlated_sequence)];
[(subsequence,sequence2b,'30b',
  correlated_sequence)];
[(subseqeunce,sequence2c,'30c',
  correlated_sequence)];
[(topical_sequence1,sequence3,'31',
  topical_sequence)];
[(topical_sequence2,sequence4,'32',
  topical_sequence)];
[(topical_sequence3,sequence5a,'33',
  topical_sequence)];
[(topical_sequence3,sequence5b,'34a',
  topical_sequence)];
[(topical_sequence3,sequence5c,'34b-35',
  topical_sequence)];
[(topical_sequence3,sequence5d,'34b-35',
  topical_sequence)]; no
```

```
% Query 2:
?- apply(frame,[_,_,_,_,_,_,
     actant([subject(E),_,_,_,_,_]),_,_]),
     write(E),nl,fail.
% Answer
     man robbers robbers robbers priest priest
     samaritan samaritan samaritan samaritan
% Query 3:
?-bagof((sort(A),lexie(B),syntagm(C),function(D),
          object(E),helper(F)),
     frame(_,sort(A),lexie(B),_,syntagm(C),
          function(D),
          actant([subject(samaritan),
          object(E),_,_,helper(F),_]),_,_),
          Samaritan).
   Samaritan =
   [(sort(sequence5a),lexie('33'),
     syntagm(contract_syntagm1),
     function(mandating),
     object([health,vigor]),helper(_))];
   [(sort(sequence5b),lexie('34a'),
     syntagm(disjunction_conjuction_syntagm),
     function(movement),
     object([health,vigor]),helper(_))];
```

```
[(sort(sequence5c),lexie('34b-35'),
   syntagm(performance_syntagm1),
   function(confrontation),
   object([health,vigor]),
   helper([know_how,oil,wine,donkey,
          money,innkeeper]))];
[(sort(sequence5d),lexie('34b-35'),
   syntagm(performance_syntagm2),
   function(domination_submission),
   object([health,vigor]),
   helper([know_how,oil,wine,donkey,
          money,innkeeper]))]; no
```

By the query 1, we can list up all the sequences, and through it we can grasp the hierarchical order of this parable that consists of two branches, the correlated sequence (one initial correlated sequence and three subsequences) and the topical sequence (three sorts of topical sequences and its total number is six). By the query 2, all the subjects are at once searched out. By the query 3, the role of the good Samaritan is shown in a list disclosing the sequence name, the text, the nature of syntagm, the function and the object with its helper. END

(3.1.2) Multi-Dimensional Model

The early version of structural exegesis is in fact no more than a syntactic analysis, while Patte's recent overall reconsideration of the theory and technique of structural exegesis[19] is perhaps the most remarkable work in biblical exegesis, because his restriction of the semantic domain to the final form of the New Testament text and his precise, practical method (the six-step method) seem to succeed without

[19]Patte, *Structural Exegesis for NT Critics*; *Religious Dimensions.*

difficulty in attaining not only objective but also convincing results. In the previous example, we merely reproduced the theory of the early version of structural exegesis to implement our FCAT to it. In this section, on the contrary, we want to expand further Patte's latest version of structural exegesis in two points in order to make structural exegesis more powerful. First, we introduce fuzzy sets not only to quantify the grade of opposition between the implied author and the implied reader that are postulated within the biblical text, but also to make manipulation of materials more flexible and suitable to the real semantic world of the biblical text. Second, we strengthen Patte's structural exegesis by affording efficiency as well as accuracy of mathematical logic, both of which are obtainable by introducing logic programming by Prolog. We also use the result of Patte's structural exegesis on John 3:1-21 to demonstrate an example of the expanded version of structural exegesis by fuzzy Prolog.

Example 10 (The Latest Version of Structural Exegesis by Fuzzy Prolog): In Prolog, we can distinctively write programs of the knowledge database and of the inference rule. Thus we first translate into the knowledge bases in Prolog language the result of Patte's structural exegesis on the patterns of convictions about Jesus and about believers or religious leaders in the dialogue of Jesus and Nicodemus in John 3:1-21. Then we establish inference rules of the values of fuzzy membership function and of positive and negative rates of example data according to the implied author's patterns of convictions. By these inference rules, we can determine objectively the value of fuzzy membership function.

The implied author's conviction patterns about believers and religious leaders[20] can be rewritten as follows:

```
/* Knowledge Base 1 */

% author's positive convictions

%    about believers or religious leaders

author_convic(be_in_god).
```

[20]Patte, *Structural Exegesis for NT Critic*, 56-59.

```
author_convic(have_some_truth).
author_convic(do_what_is_true).
author_convic(be_willing_to_come_to_the_light).
author_convic(believe_in_jesus_in_the_name_
                 of_the_only_son_of_god).
author_convic(receive_the_testimony_of_jesus).
author_convic(have_a_secondhand_knowledge).
author_convic(be_born_of_the_spirit).
author_convic(have_a_firsthand_knowledge).
author_convic(bear_witness_to_the_role_
                 of_the_spirit).
author_convic(enter_the_kingdom_of_god).
author_convic(have_eternal_life).
```

The author's conviction patterns about Jesus[21] can be rewritten as follows:

```
/* Knowledge Base 2 */
% author's positive convictions about Jesus
author_convic(be_in_heaven).
author_convic(have_knowledge_
                 of_heavenly_things).
author_convic(be_the_only_son_of_god).
author_convic(be_sent_by_god).
author_convic(descend_from_heaven).
```

[21]Patte, *Structural Exegesis for NT Critic*, 54-56.

```
author_convic(bear_witness_to_earthly_things).

author_convic(bear_witness_to_heavenly_things).

author_convic(be_light_of_the_world).

author_convic(be_a_true_teacher).

author_convic(ascend_to_heaven).

author_convic(bring_salvation_to_the_world).

author_convic(give_eternal_life).
```

With respect to the author's positive convictions about believers and religious leaders, Nicodemus's patterns of convictions about believers and religious leaders, which are either positive or negative, are summarized as follows:

```
/* Knowledge Base 3 */

% example: the case of Nicodemus

convic(be_in_god).

convic(have_some_truth).

convic(come_in_darkness).

convic(believe_in_jesus_in_the_name_

                  of_the_only_son_of_god).

convic(receive_the_testimony_of_jesus).

convic(have_a_secondhand_knowledge).

convic(not_be_born_of_the_spirit).

convic(not_enter_the_kingdom_of_god).
```

The inference rules about fuzzy membership function of the conviction in the sample data of knowledge base (Nicodemus's one in this case) with respect to the implied author's conviction are writtern as follows:

```
/* Rule Base 1 */
```

```
% author's conviction list
author_convic_number(AuthorList,ALN):-
      setof(A,author_convic(A),AuthorList),
      length(AuthorList,ALN).
% data conviction list
data_convic(DataList,DN):-
      setof(D,convic(D),DataList),
      length(DataList,DN).
% positive and negative conviction lists
pos_convic(X):-
      convic(X),
      setof(A,author_convic(A),AuthorList),
      member(X,AuthorList).
neg_convic(Y):-
      convic(Y),
      setof(A,author_convic(A),AuthorList),
      not member(Y,AuthorList).
pos_convic_list(PosList,PN):-
      setof(X,pos_convic(X),PosList),
      length(PosList,PN).
neg_convic_list(NegList,NN):-
      setof(Y,neg_convic(Y),NegList),
      length(NegList,NN).
/* Rule Base 2 */
```

```
% the value of fuzzy membership function
%          that maps convic(X) to author_convic(A)
memb_f(M):-
     setof(A,author_convic(A),AuthorList),
     length(AuthorList,ALN),
     setof(X,pos_convic(X),PosList),
     length(PosList,PN),
     M is PN/ALN.
% positive conviction rate
pos_convic_rate(PR):-
     setof(D,convic(D),DataList),
     length(DataList,DN),
     setof(X,pos_convic(X),PosList),
     length(PosList,PN),
     PR is PN/DN.
% negative conviction rate
neg_convic_rate(NR):-
     setof(D,convic(D),DataList),
     length(DataList,DN),
     setof(Y,neg_convic(Y),NegList),
     length(NegList,NN),
     NR is NN/DN.
```

The list of Nicodemus's convictions about believers and religious leaders can be easily classified into positive and negative lists with respect to the author's convictions by using the knowledge base 1, the knowledge base 3 and the rule base 1:

```
% Query 1: positive conviction list
%                 and its number
?- pos_convic_list(PosList,PN).
      PosList =
      [be_in_god,
      believe_in_jesus_in_the_name_
            of_the_only_son_of_god,
      have_a_secondhand_knowledge,
      have_some_truth,
      receive_the_testimony_of_jesus],
      PN = 5 ;
% Query 2: negative conviction list
% and its number
?- neg_convic_list(NegList,NN).
      NegList =
      [come_in_darkness,
      not_be_born_of_the_spirit,
      not_enter_the_kingdom_of_god],
      NN = 3
```

From the author's point of view, the grade of conviction as a believer or a religious leader and the positive and negative rates of convictions in the case of Nicodemus is judged from the knowledge base 1, the knowledge base 3 and the rule base 2:

```
% Query 3: value of fuzzy membership function
%                    about Nicodemus's belief

?- memb_f(M).

       M = 0.416667

% Query 4: positive rate of convictions

%                    of Nicodemus's belief

?- pos_convic_rate(PR).

       PR = 0.625

% Query 5: negative rate of convictions

%                    of Nicodemus's belief

?- neg_convic_rate(NR).

       NR = 0.375
```

The grade of the appropriateness of Nicodemus's belief to the author's conviction world is expressed by the value of fuzzy membership function, 0.42, which was deduced objectively and automatically.

Next, we examined another example of the convictions about the mediator (Jesus in the case of John 3:1-21). We show a counter-example of Gnostic malevolent angels or principalities as found in several treatises of the Nag Hammadi Library such as the *Apocryphon of John*, the *Hypostasis of Archon* or *On the Origin of the World.* It should be notes that our presentation is a toy example of a highly simplified case:

```
/* Knowledge Base 4 */

% example: Gnostic malevolent angels

%                    or principalities

convic(be_in_heaven).

convic(be_darkness_of_the_world).

convic(be_sent_by_evil_deity).

convic(descend_from_heaven).
convic(ascend_to_heaven).

convic(give_earthly_painful_life).
```

From the knowledge base 2, the knowledge base 4 and the rule base 1, we can determine which description is judged as being positive or negative in the world of theological convictions of the fourth Gospel's author:

```
% Query 6: positive description list

%                  and its number

?- pos_convic_list(PosList,PN).

      PosList =

      [ascend_to_heaven,

      be_in_heaven,

      descend_from_heaven],

      PN = 3

% Query 7: negative description list

%                  and its number

?- neg_convic_list(NegList,NN).

      NegList =

      [be_darkness_of_the_world,

      be_sent_by_evil_deity,

      give_earthly_painful_life],

      NN = 3
```

Accordingly, the values of fuzzy membership function, positive and negative rates are obtained from knowledge base 2, knowledge base 4 and rule base 2 as follows:

```
% Query 8: value of fuzzy membership function
%                    about Gnosticism

?- memb_f(M).

     M = 0.25

% Query 9: positive rate of descriptions

%                    about Gnosticism

?- pos_convic_rate(PR).

     PR = 0.5

% Query 10: negative rate of descriptions

%                    about Gnosticism

?- neg_convic_rate(NR).

     NR = 0.5         END
```

The latter example may be interesting because it discloses in part overlapping features between the implied author of the Gospel of John and Gnosticism. However, its value of membership function, which assigns the description of Gnosticism to the author's space of semantic domain or world, is relatively low (0.25). Thus by fuzzy Prolog, we can attain the quantification of the grade of conviction or theological belief, which is in turn highly significant in scholarly tasks about understanding of historical, religious texts.

As shown in the example 10, the distinction of knowledge base and of rule base could provide us efficiency and easy application of the method of automated text analysis. When we prepare new knowledge bases around these themes, we can easily evaluate its distance from or affinity with the theological world of the fourth Gospel by determining the value of fuzzy membership function to the semantic space of the fourth Gospel's author.

3.2. *Comment on Other Methods*

In the implementation of our FCAT to redaction criticism, as demonstrated in the example 8 of the toy version of redaction criticism in fuzzy Prolog, our method itself is objective, rigorous and powerful.

The most difficult matter in developing automated text analysis in redaction criticism can be rather ascribed to the heuristic discrimination of traditional or redactional parts of biblical texts. This means the subjective determination of tradition or redaction that can be easily biased by interpreter's presumptions. If the expert knowledge base itself is predetermined by interpreter's presumptions, the results of its computational analysis may be merely a direct reflection or reproduction of such presumptions. This may fall into highly subjective results in the guise of a scientific method.

In literary criticism, rather biblical exegetes positively use their pre-understanding or their own conviction or feeling in order to grasp semantic values of biblical texts. On the contrary to redaction criticism in which we can easily implement FCAT, it may be somewhat difficult to translate such subjective values in modern interpreters' semantic world into objective, computational programming language. The interaction between the semantic space of the implied author found in the latest form of Bible and the semantic space of modern interpreters could be in part evaluated in FCAT by fuzzy set theory with Prolog language, but its result may be not so impressive.

In structural exegesis, on the contrary, as demonstrated in examples 9 and 10, the FCAT becomes a powerful tool, because the nature of structural exegesis itself is objective and even it can be easily extended to other methods. Thus the expanded version of structural exegesis under FCAT by using fuzzy sets and Prolog will provide for exegetes both a highly efficient technology and a broad application research field.

Besides these three methods, redaction criticism, literary criticism and structural exegesis, the most fruitful field in biblical studies through the implementation of FCAT would be a historical approach, in which numerous extra-biblical texts are massively referred in form of knowledge database and new hypotheses are automatically induced by the inductive machine learning from examples of such extra-biblical texts. There are several advantageous circumstances. First, recent collaborative works to prepare digital versions of classical texts such as *Thesaurus Linguae Graecae* CD-ROM or others offer us a powerful tool to easily access to massive original classical texts in Greek language or others. These CD-ROM databases are now available for our AI version of expert knowledge base in FCAT. Second, recent trends in the AI community have disclosed the importance of the inductive machine learning in knowledge acquisition, as indicated by several international conferences and symposium.[22] By adopting such recent AI technology, our historical approach in FCAT will become more rigid and efficient.

In addition, by using technique of metaprogramming in Prolog, we can easily modify our FCAT to other environments. For example, we may implement FCAT to rhetorical criticism as an automated text processing. This becomes our future task.

Together with our theoretical elaboration as FCAT, we are now at the starting point of the new scientific research field, that is, the computational analysis of historical text equipped with digital massive databases and powerful AI technology.

Conclusion and Future Works

4.1 *Conclusion*

(1) In the present introductory study, we established our theory and technology in biblical exegesis as a framework for the computational analysis of text (FCAT), which consists of three steps: definition of a kernel formal system (KFS), extension to other formal systems, and application to conventional biblical methods. We first defined KFS as a minimum formal system that includes a constant, a variable, a relation, a predicate, a universe and a space; then we extended KFS to other two formal systems, logic programming by Prolog and fuzzy set theory; and finally we applied it to redaction criticism and structural exegesis.

(2) Our extension path reached finally fuzzy Prolog. By fuzzy sets, we can quantify the grade of appropriateness of the semantic world of an object to other ones. By Prolog, we can obtain an efficient tool to automatically process logic calculation of the statements expressed in numerous biblical texts as well as historical texts.

---

[22]F. Bergadano et al. (eds.), *Machine Learning: An Integrated Framework and its Applications* (New York: Ellis Horwood, 1991); Stephen Muggleton (ed.), *Inductive Logic Programming* (London: Academic Press, 1992); Pavel B. Brazdil (ed.), *Machine Learning*: ECML-93 (Berlin: Springer-Verlag, 1993).

(3) Our KFS is a primitive formal system. Under FCAT, we can cover a wide range of biblical methods by rewriting them in a formal language as in a KFS and by extending them to the computational environments as in Prolog. By our KFS and FCAT, we can thus treat diverse biblical methods in a unified manner.

4.2 *Future Works*

(1) We can choose extension paths different from our extension to fuzzy sets and Prolog, either by adopting other mathematical systems such as non-monotonic logic in place of fuzzy sets or Horn clauses, or by using other computer languages such as object-oriented programming language in place of Prolog language.

(2) In order to make our FCAT more suitable to the computational analysis of religious text, we should take into consideration the implementation of FCAT to auto-epistemic logic (logic of belief)[23] or to deontic logic (logic of obligation).[24]

(3) In order to apply FCAT to other methods in biblical exegesis, the implementation of FCAT to literary criticism and rhetorical criticism may become interesting works. However, our FCAT seems very suitable to a historical approach by using massive extra-biblical database and advanced AI technology. We are now preparing new studies.

ABSTRACT

In this introductory study, the framework for the computational analysis of text (FCAT) in bibical studies was presented. FCAT consists of three continuous steps: first, definition of a kernel formal system (KFS) that includes minimum components, a constant, a variable, a relation, a predicate, a universe

[23]Cf. Anil Nerode et al. (eds.), *Logic Programming and Non-monotonic Reasoning*: Proceedings of the First International Workshop (Cambridge [Mass.]: MIT Press, 1991).

[24]Cf. John-Jules Ch. Meyer and Roel J. Wieringa (eds.), *Deontic Logic in Computer Science: Normative System Specification* (Chichester: John Wiley & Sons, 1993).

and a space; second, extension of KFS to other advanced formal systems such as logic programming by Prolog and fuzzy set theory, which thereafter provides a simplified version of fuzzy Prolog; third, application of KFS to conventional biblical methods such as redaction criticism and structural exegesis, in which several programs written in Prolog were demonstrated. By FCAT, biblical scholars could obtain (a) a common tool to share information between diverse biblical methods, (b) a powerful computational tool, and (c) theoretical advantage suitable to biblical texts that are essentially ambiguous in nature.

要約

この予備的研究において、テキストのコンピュータ分析のためのフレームワーク (Framework for the Computational Analysis of Text [FCAT]) が提示された。FCAT は次の三つの連続的な段階によって成っている。第一に最小限の構成要素からなるカーネルを形づくる形式的システム (Kernel Formal System [KFS]) の定義。KFS の構成要素は定数、変数、関係、述語、ユニバース、空間から成る。第二にこの KFS を他のより優れた形式的システムへ拡張すること。ここでは Prolog による論埋プログラミングとファジイ集合論への拡張が行われた。第三に従来の聖書学方法論への KFS の適用。ここでは編集史と構造主義批評とについてなされ、それらについての Prolog によるプログラムが提示された。この FCAT によって聖書学研究者は以トの三つを得ることが可能となる。第一に相異なる聖書学方法論に分断された研究領域間での情報を交換するための共通の手法。第二に強力なコンピュータ分析のための手法。第三に本質的に曖昧さが伴う聖書のテキストに合致した埋論的な優位性。これにより従来には想定できなかった全く新しい研究領域が開拓されるであろう。

撮要

作者在本篇文章中討論用電腦分析經文的架構（Framework for the Computational Analysis of Text [FCAT]) 來研究聖經。FCAT包括三個連續的步驟：一、替 KFS (Kernel Formal System) 下定義，它包括最小組成部分、常數、變數、關係式、述語、universe、及字元；二、將KFS接駁至其他先進的正規（formal）系統，例如使用Prolog及快思集論（Fuzzy Set Theory）寫成的邏輯程式（製造商根據該程式製成Fuzzy Prolog 的簡化版）；三、運用 KFS 來研經，而研經法則是傳統的編修鑑別及結構式研經法。作者在本文化展示一些用 Prolog 寫成的程式。運用FCAT，聖經學者：一、能夠有共同工具來使相異之研經法所得之成果得以相互沛通；二、得到一個强力的資料處理工具；三、面對那些意思不清晰的經文，則可在理論上佔優勢。