Survey Methodology, December 1988
Vol. 14, No. 2, pp. 289–298
Statistics Canada

QUID, A General Automatic Coding Method

JACQUES LORIGNY[1]

ABSTRACT

The QUID system, which was designed and developed by INSEE (Paris) Institut National de la Statistique et des Études Économiques - National Statistics and Economic Studies Institute, is an automatic coding system for survey data collected in the form of literal headings expressed in the terminology of the respondent. The system hinges on the use of a very wide knowledge base made up of real phrases coded by experts. This study deals primarily with the preliminary automatic standardization processing of the phrases, and then with the algorithm used to organize the phrase base into an optimized tree pattern. A sorting example is provided in the form of an illustration. At present, the processing of additional coding variables used to complement the information contained in the phrases presents certain difficulties, and these will be examined in detail. The QUID 2 project, an updated version of the system, will be discussed briefly.

KEY WORDS: Automatic coding; Natural language variables; Phrase matching; N-grams.

1. INTRODUCTION

The QUID (abbreviation of QUestionnaires d'IDentification - Identification Questionnaires) system is an automatic coding system designed and developed by the Institut National de la Statistique et des Études Économiques (INSEE - National Statistics and Economic Studies Institute) in 1979-1980.

Review of the Problem

The problem consists of automatically classifying an individual surveyed into a job defined in accordance with an existing nomenclature (for example, the nomenclature of the professions). In order to do this, the system uses mainly the natural language answer given in response to a direct question (for example, "What is your present profession or trade?"), as well as additional information contained in the survey form, which is assumed to have been previously coded (for example, the Economic Activity code for the firm where the individual works).

In our terminology, a direct answer in natural language is called the "literal heading", or simply "heading". Any additional encoded information is represented by the generic term "additional variables".

In the next section, we will discuss the basic approach of the QUID system and the results of its implementation at INSEE. In section 3, we will describe the present version of the system. Finally, in section 4, we will examine the problems surrounding the processing of additional variables, and will discuss the new version of the system (QUID 2), which should help resolve the difficulties encountered.

[1] Jacques Lorigny, Administrateur à l'Institut National de la Statistique et des Études Économiques 18, Bld Adolphe Pinard 75675 PARIS CEDEX 14 (France).

2. THE PRINCIPLE BEHIND THE METHOD

2.1 The Basic Approach

The basic approach of the QUID system consists of building a very large data base made up of typical respondent headings accompanied by a corresponding code assigned by an expert. The data base is as large as possible in order to make it possible to obtain a high matching rate, and new headings are added to the base as they appear.

In our terminology, the data base is called a "knowledge base" or "knowledge file" (KF), because it has the ordinary structure of a flat file in its raw state. Most often, the knowledge file is set up on the basis of a survey carried out during a previous year, which has already been coded either manually or using an interactive method. Each base heading is accompanied by its code (which is *a priori* assumed to be accurate), and its "frequency of occurrence" in the KF; that is, the number of individuals who responded using this heading.

The management task of the knowledge base (auditing, expansion) is completely separate from the operation of coding the survey under way. It is the responsibility of a central office staffed by expert coders, while the coding operation itself is most often regionally decentralized.

The difficulties of an approach of this type derive from the rapid increase in the time required to search the base as it grows in size. In order to solve this problem, the QUID system uses mathematical results derived from Information Theory (Shannon 1948; C.-F. Picard 1972; B. Bouchon-Meunier 1978; M. Terrenoire 1970; D. Tounissoux 1980), which can be used to minimize search time by organizing the base in the form of an optimized tree structure.

The basic approach of the QUID system also makes it possible to opt for a set of general programs; that is, those that can be used with all semantic fields, for example, professional, food products, or municipal headings.

2.2 Results

The system has been tried for various INSEE tasks and is presently being used to code the CS (socio-professional category) code in order to process DADS (Déclarations annuelles de données sociales - Annual social information) data provided by all firms that employ paid labour. The following figures provide an idea of the orders of magnitude involved.

At present the knowledge file for the DADS application contains 122,000 headings (representing a knowledge base population of 650,000 wage earners). Its optimized organization consists of a tree with about 100,000 nodes (of which 86,000 represent certainty nodes; see section 3.2). It has been used to code a population of 570,000 wage earners with an average effectiveness of 90%, varying between 85% and 95%, depending upon the region. By "effectiveness", we mean the percentage of cases where the system provides a single answer which is accepted on principle under the conditions of this application. At present, since we do not have a precise measurement of the validity of these single answers, we estimate that the error rate is likely to be in the order of 5% to 10%. However, the knowledge base is being audited by the Dijon Expert Centre, according to which a significant proportion of the error rate should normally decrease. Once this has been achieved, we will have more accurate figures to report.

From the point of view of data processing limitations, the optimized tree is loaded into 3,300 kilobytes of central (virtual) memory and the automatic coding time for an individual case is in the order of 40 ms in an IBM 4341 central processing unit.

For the last few months, we have had available a variant of the coding program itself. This has been designed for use with mini-computers and can load the tree by sections, depending upon available memory space.

In applications other than DADS data, effectiveness is not as high, no more than 75%. It all depends upon the quality and comprehensiveness of the knowledge base.

3. THE PRESENT VERSION OF THE QUID SYSTEM (QUID 1)

3.1 Preliminary Standardization of Headings

Before constructing the optimized tree, the raw headings are first standardized in accordance with a set of external parameters chosen by the user for his application.

The words are separated and fitted into predetermined zones whose length (a single one for all words) and maximum number (a single one for all headings) are parametrized. It is advisable to choose a larger value on the basis of these two parameters, and allow the optimization algorithm itself to select the significant elements of the heading (see section 3.2). For example, the DADS application (see section 2.2) chose 4 zones of 12 characters each.

"Empty words" are eliminated. The list of empty words is an external parameter provided by the user for his application. Most often, it includes articles, prepositions, *etc.*, and is significantly dependent upon the application.

Initials are standardized (I.N.S.E.E. becomes INSEE, S N C F becomes SNCF).

Finally, the user may process the table of separate words in any way he wants (in the form of a subprogram in the PL/1 language). In fact, this is rarely necessary and seldom used (except to code municipal codes from municipal headings).

Once word processing has been completed, the words are divided into bigrams (blocks of two consecutive letters) or trigrams (blocks of three consecutive letters), *etc.* Choosing the type of blocking is parametrized (however, a single parameter is used for the entire application). In practice, blocking into bigrams is the only type that has been used until now; however, the idea of blocking into trigrams should be tested. For the purposes of this study, we will only consider blocking into bigrams.

3.2 The Algorithm Used to Set Up the Optimized Tree Pattern

Let $T = (t_1, t_2, \ldots, t_j, \ldots, t_n)$ the code to be coded, for example all the modalities of the Profession code.

$Q = (q_1, q_2, \ldots, q_i, \ldots, q_m)$ all the bigrams resulting from the standardization of the headings (for example, m = 24 when the number 4 has been chosen as the "number of words" parameter, and 12 characters as the "word length" parameter).

$X =$ the tree pattern to be constructed, which we call a "QUID" (questionnaire d'identification - identification questionnaire).

The algorithm constructs X by parsing down from the root node x_0 (which by convention is placed at "level 0") to the nodes in levels 1, 2, *etc.*

At root node x_0 it links the entire KF, and searches for the best bigram to query first; that is, that which can discriminate best for the desired code T in the entire KF.

$N(x_0)$ represents the total frequency of occurrence associated with the entire KF; that is, the sum of frequencies accompanying the base headings,

$N(x_0, j)$ is the frequency of occurrence of code t_j in the entire KF.

We assume that the knowledge population is statistically representative of the population to be coded (we should recall that, in practice, the KF is often the survey file for a previous year).

Thus, we can estimate the probability of finding code t_j the population to be coded on the basis of the following formula:

$$\Pr(t_j \mid x_0) = N(x_0,j)/N(x_0).$$

The *a priori* ambiguity for T is measured on the basis of Shannon's entropy:

$$H(T/x_0) = \sum_j \Pr(t_j \mid x_0) \log 1/\Pr(t_j \mid x_0).$$

Let us assume that a bigram, for example q_i is allocated to node x_0. To each of its modalities in the KF, we associate the sub-base made up of the headings that have this modality.

Let $(a_i^1, a_i^2, \ldots, a_i^k, \ldots)$ represent the modalities captured by bigram q_i in the KF. For each of these modalities, thus, for each of the sub-bases generated, we create a node y, which follows immediately after x and is located at level 1 of the tree.

The information provided by bigram q_i (which is assumed to be assigned to root node x_0) is measured by the average reduction in the ambiguity of T when we go from x_0 to one of the y nodes.

That is:

$$I(x_0,T,q_i) = H(T \mid x_0) - \sum_{y \in \Gamma(x_0)} \Pr(y)\, H(T \mid y),$$

where

$\Gamma(x_0)$ represents all the successive y nodes at level 1 below node x_0
$H(T \mid y)$ the conditional entropy of T at node y.
(same formula as above but replacing x_0 by y).
$\Pr(y)$ $= N(x_0,a_i^k)/N(x_0)$ if a_i^k is the modality of bigram q_i which generates node y, and $N(x_0,a_i^k)$ is the frequency of occurrence of modality a_i^k of bigram q_i in the entire KF.

The algorithm carries out this data calculation for all bigrams $q_1, q_2, \ldots, q_m$, because at root node x_0 they are all possible candidates for selection as the first bigram to be queried.

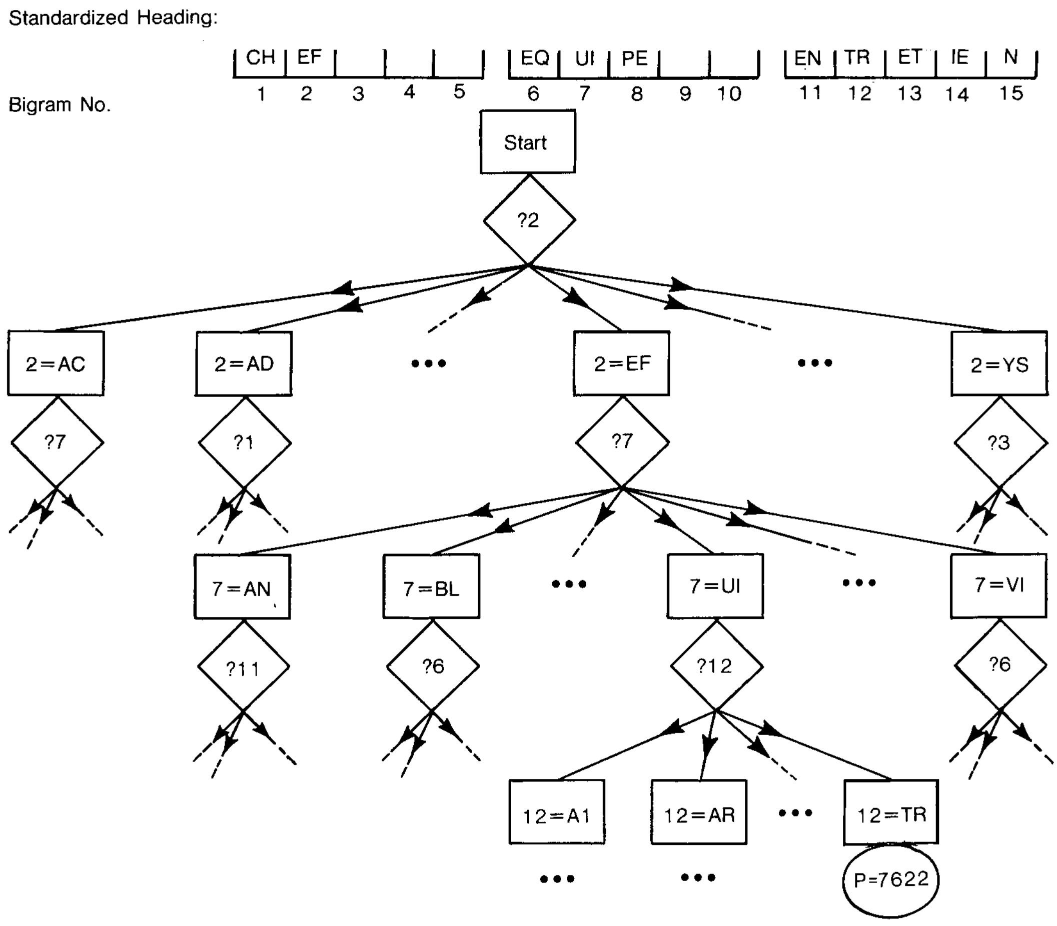
The algorithm chooses the bigram which maximizes $I(x_0,T,q_i)$. For example, in the case of q_{i0}, it effectively divides the base into as many sub-bases as there are modalities of bigram q_{i0} in the base. This effectively creates the y, nodes that follow x_0 at level 1, and the construction of level 1 of X is thus completed.

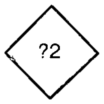
For each sub-base obtained (thus, for each y node), the algorithm carries out exactly the same operation as that which we have just described for root node x_0, and so on.

The process stops for a given node:

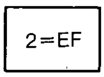
(1) when there is only one heading at the node; in this case, the conditional entropy is zero; or
(2) when there is only a restricted number of headings that differ in terms of the remaining bigrams, but which have all the same code; or
(3) when there are two headings or more, but they have different and not distinguishable codes.

Cases (1) and (2) are known as "certainty nodes", and case (3) is known as the "uncertainty node". Together, they represent the "terminal nodes".

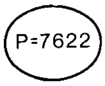




Query the content of bigram no. 2 in this node of the tree.



The content of bigram no. 2 is EF.



In this node of the tree, we may determine the profession code: its value is 7622 (1975 Trade Nomenclature).

In this example, the raw heading is that of the profession entered by the individual surveyed. The objective of the system is to find the corresponding profession code in the 1975 Trade Nomenclature.

Initially, we extract the first ten characters of the three most significant words. In this way, we obtain the standardized heading, which is then blocked into pairs of letters (these are called bigrams and are numbered from 1 to 15). Then, we query the system. This operates in accordance with a chain of questions and answers optimized by a mathematical algorithm based on information theory. This calculation takes place during the course of a preliminary phase which determines the first bigram queried as a function of a given knowledge file, and then the following sequence of questions depending upon the answer obtained each time. At this point, the computer queries first bigram no. 12, which contains TR. At this stage, it ascertains that it can without ambiguity determine that this repesents the Profession 7622 code (Technical Staff and Technicians). On the average, processing time takes a total of 41 milliseconds of computer time in an IBM 370/148, and the amount of central memory used is 380 Kbytes.

Raw Heading: head, maintenance team.

Figure 1. Example of Classification of a Heading in the Tree.

The construction of the tree *X* continues from level to level until the KF has been exhausted. In fact, we have never gone beyond level 15, but there is no set limit for the system itself. An example of classification in the tree is shown in Figure 1.

3.3 The Use of the Coding Itself

In order to code a heading in the current survey, we start by standardizing the information in accordance with section 3.1. Then, the bigrams obtained are matched against those of the Quid loaded into the computer. The exploration leads to three possible results.

3.3.1 Certainty Node

The system provides a single code but this may well be wrong if the knowledge base is not comprehensive enough. For example, during one of our first tests in 1979, we obtained a certainty node for level 1 on the basis of bigram 2 = *CC*, since the only heading obtained had been VACCINEUR VOLAILLES (POULTRY VACCINATOR).

When we later had to code the heading RACCOMMODEUR VÊTEMENTS (GARMENT MENDER) the single code obtained was that representing agricultural service professions, and the error was obvious.

Thus, we added to the system a control procedure based on single echoes. This process is known as "redundancy control" and consists of verifying, after the detection of a single echo, the content of the first three bigrams of each word. A single echo (obtained on the basis of the vector leading to a certainty node) is said to be non-ambiguous, when the cluster of headings in the certainty node contains at least one heading that has the same redundancy bigrams as those of the heading to be coded. Otherwise, the echo is said to be ambiguous, and consequently treated as an anomaly of the automatic system. Experience has shown that this arrangement tends to consolidate significantly the reliability of the system without appreciably overburdening the tables in memory or increasing processing time (even in large applications, the number of redundancy formulas per certainty node is, on the average, in the order of one, and rarely goes beyond ten).

In order to be thorough, we should add that this redundancy control is not rigidly set once and for all. The user has two external parameters: the list of bigrams over which he intends to exercise control, and the (maximum) number of bigrams retained. In this way, he can keep in check the severity of the matching control, depending upon his objectives in terms of the quality and "effectiveness" of automatic coding.

3.3.2 The Uncertainty Node

The system provides various possible codes (most often two codes), and displays their respective frequencies of occurrence at the node under consideration. In this case, the officer who has the file of the survey being processed will then manually reject one of the two.

3.3.3 The Case of an Unknown Response

If, during the course of exploring the Quid, the modality sought is not found in the modalities captured by the bigram queried, the search will fail and this also represents a case of rejection that must be processed manually.

New cases encountered during the course of processing will be stored in memory, centralized in the expert centre, verified, and then incorporated into the KF in order to produce a new expanded version of the Quid.

At present, for purposes of convenience, the knowledge iteration takes place once a year, but nothing prevents it from being organized so that it takes place more often so that applications can progress faster, for example in the case of population surveys.

4. THE PROBLEM OF PROCESSING ADDITIONAL VARIABLES

In the present version, QUID 1, additional variables are simply structured into bigrams and processed in the same way as literal data. This leads to certain difficulties and problems that made it necessary to develop a new version, QUID 2, which operates in two stages:

- in the first stage, QUID 1, which is reserved for processing the literal heading and producing either the final code (when this is totally determined by the heading), or an internal code designating a rule or decision table that can be applied to the additional variables to achieve the calculation;
- in the second stage, the rules or decision tables achieve the determination of the final code.

Detailed Examination of the Difficulties Encountered

At times, certain nomenclatures that are particularly complex, such as the PCS Code (Nomenclature of Professions and Socio-Professional Categories) call upon a combination of the literal heading and various additional variables.

For example, the coding of the PCS code uses the Professional Category additional variable (which is abbreviated to CPF). The following is the question such as it appears in the 1982 Population Census Individual Form:

Indicate the professional category of your present job:

	- unskilled or semi-skilled labourer	1
- labourer	- semi-skilled labourer (OS, O1, O2, O3, ...)	2
	- skilled labourer (P1, P2, P3, TA, OP, OQ ...)	3
- clerk		4
- technician, draftsman		5
	- supervising workers or clerks	6
- foreman	- supervising other foremen or technicians	7
- engineer or professional staff		8

The additional question was made necessary by the fact that the heading alone is not always enough to classify the individual in accordance with PCS nomenclature.

For example, a LUMBER COMPANY WORKER

- must be classified into 6916 (lumber company or forestry worker) if his CPF is 1, 2, 3, or 4
- and into 4801 (Managerial and supervisory staff of agricultural or lumber operations) if his CPF is 5, 6, 7, or 8.

The present system considers these additional variables as if they were literal data. They are placed at the end of the heading and structured into bigrams in the same way (for example, the CPF variable with the addition of a blank space is placed into the $(m + 1)$th bigram). However, this solution is not satisfactory and leads to various errors:

Error No. 1. When there is not enough information in the KF, this may lead to many cases of unknown responses.

For example, if the KF has only one LUMBER COMPANY EMPLOYEE with a CPF = 2 and another with a CPF = 7 the file will be unable to find a LUMBER COMPANY WORKER with a CPF other than 2 or 7 (that is, *a priori* in 6 cases out of 8). This error is made worse when the additional variable is very diluted, for example, in the case of the variable representing the Economic Activity of the undertaking (which is abbreviated as additional variable AE).

Error No. 2. When there is not enough information in the KF, this may lead to miscodings.

For example, if the KF has only one LUMBER COMPANY WORKER with a CPF = 2, the CPF bigram will not discriminate or appear in the search key, so that a LUMBER COMPANY WORKER with a CPF = 7, will be classified into PCS = 6916 instead of 4801. This is a case of miscoding

In order to correct this defect in the present system, the only measure we can take is to apply the redundancy control to the additional variables (and thus obtain an ambiguous or questionable case which is rejected or corrected manually, instead of allowing the error to remain undetected). However, here again, this is only a last resort. In fact, the additional variables lead to an unchecked expansion of the KF. Each KF reference has its own cross combination of modalities of additional variables, and it is not very likely that we would find the same combination for a new individual to be coded. Thus, this will lead to many uncertain cases and automatic coding rejections, which will reduce the practical benefits of mass exploitation.

The two errors, no. 1 and no. 2, are related to the relative incompleteness of the KF. For example, it would be enough to enter into the KF eight LUMBER COMPANY WORKER titles and add in each case one of the possible CPF modalities (1 to 8), in order for the two errors to disappear. However, in the case of real applications, we find that the relative incompleteness of the KF decreases quite slowly, as it grows to reach its operating pace. Contrary to the lexicographic space of literal headings, which tend to become dense rather quickly, the cross checked space of the additional variables remains a vast frontier for a long time, and goes very slowly from a density of occupation of 0 to a density of 1 (one individual).

Error No. 3. There is a third category of errors that are not caused by the incompleteness of the KF but by the excessive sensitivity of the QUID in relation to errors inevitably contained in the file (and this always in relation to the additional variables).

Let us take a simple example. Let us assume that the SENIOR SECRETARY heading must be coded PCS = 4615 (senior secretarial staff), regardless of the value of all the additional variables. Let us consider the following KF, in which an error has slipped by (for example, the failure to assign the PCS code):

Heading	**CPF a.v.**	**AE a.v.**	**PCS Code**
Senior Secretary	$\boxed{7}$	$\boxed{49}\boxed{11}$ (fashion design, haute couture)	4615
Senior Secretary	$\boxed{7}$	$\boxed{83}\boxed{43}$ (loan cooperative)	4616 $\uparrow$ error

Even though the AE additional variable should not be used to code the PCS code, the QUID algorithm uses it to separate the two certainty nodes.

- One in favour of 4615 in view of bigram AE1 = 49.
- And the other in favour of 4616, in view of bigram AE1 = 83.

The result is that, during the coding stage itself, all senior secretaries belonging to economic sectors other than those starting by 49 or 83 will appear as "unknown cases". Moreover, those in all sectors starting by 83 will obviously produce errors. However, it is mainly the first phenomenon that interferes with accuracy, because it affects an area that is much larger than that affected by the initial error.

Error No. 4. Finally, the present QUID algorithm is excessively rigid in terms of choosing the optimal question. Most often, this results in a simple inversion of the order of the questions in the course of the search, in relation to the order that would have been preferred by the designer. Thus, the effect is secondary, since the final results are identical. However, this may also lead to more serious distortions.

Let us take the following (partly fictitious) example. Let us assume that, according to the nomenclature, the SENIOR SECRETARY heading should be coded either PCS = 4615 as above if the CPF additional variable CPF = 1 to 7, and PCS = 3726 (current managerial staff in other administrative business services), if CPF equals 8.

Let us examine the KF containing the following two references:

Heading	**AE a.v.**	**CPF a.v.**	**PCS Code**
Senior Secretary	49 | 11	8	3726
Senior Secretary	83 | 43	7	4615

Thus, the two references are correctly coded. When the QUID algorithm arrives at a node where it has examined all the possible bigrams of the literal heading, it must now choose one bigram in the additional variables, in order to separate the two final results: PCS = 3726 and PCS = 4615. In this simple but not altogether unrealistic example, the three possible bigrams: AE1, AE2, and CPF, provide the same quantity of information (one bit). In our algorithm, the arbitrary convention is that in cases of equality, the program should choose the first question in the order in which the additional variables were presented in the form. However, in this example, this will be deceiving, since we would encounter the aberration discussed above (error no. 3). However, it is not possible to determine an order of additional variables that would prevent this type of error in all important cases. We can only seek an order of questions that will be statistically the least invalid, by groping our way on the basis of the order of conceptual splits, the negentropic capacity of each additional variable, *etc.*

5. CONCLUSION

In its QUID 1 version, the present QUID system provides very valuable services to INSEE. Nevertheless, it still has certain weak points regarding the processing of additional variables.

The new QUID 2 version should improve processing while remaining faithful to our "basic approach" to the automatic coding problem, which could be summarized in two points:

1. Separation of the knowledge base (in this case, a base of rules and decision tables that are written in natural language, are independent of each other, and are audited and managed by an autonomous expert centre), and the use of automatic coding programs (in this case, loading and table exploration programs).
2. Construction of general programs; that is, programs that are independent of the semantic field processed.

At least, these are the objectives that we try to attain.

ACKNOWLEDGEMENTS

I would like to thank the reviewers for their invaluable help during the preparation of this paper.

REFERENCES

BOUCHON-MEUNIER, B. (1978). Sur la réalisaiton de questionnnaires. Doctoral thesis. Paris.

KNAUS, R. (1987). Methods and problems in coding natural language survey data, *Journal of Official Statistics*, 3, 45-67.

LORIGNY, J. (1982). Mesures d'entropie et d'information pour les systèmes ouverts complexes. Doctoral thesis, Paris.

LORIGNY, J. (1985). Manuel d'utilisation du système QUID. Institut National de la Statistique et des Études Économiques, Direction de la production, Paris.

PICARD, C.-F. (1972). *Graphes et Questionnaires.* Paris: Gauthier-Villars.

SHANNON, C.E. (1948). A Mathematical Theory of Communication. *Bell Systems Technical Journal*, 27, 379-423, 623-656.

TERRENOIRE, M. (1970). Un modèle mathématique de processus d'interrogation: les pseudo-questionnaires. Doctoral Thesis, Grenoble.

TOUNISSOUX, D. (1980). Processus séquentiels adaptifs de reconnaissance de formes pour l'aide au diagnostic. Doctoral Thesis, Lyon.