TILBURG UNIVERSITY
KATHOLIEKE UNIVERSITEIT BRABANT
POSTBOX 90153
5000 LE TILBURG
THE NETHERLANDS

[redacted]

DEPARTMENT OF ECONOMICS

RESEARCH MEMORANDUM

[redacted]

CONSTRAINTS IN BINARY SEMANTICAL NETWORKS

Shan-Hwei Nienhuys-Cheng

FEW 360

CONSTRAINTS IN BINARY SEMANTICAL NETWORKS

Shan-Hwei Nienhuys-Cheng
Infolab, Tilburg University
P. O. Box 90153, 5000 LE Tilburg, the Netherlands

ABSTRACT

Constraints in Information Systems are used to check the correctness of data, preventing redundant data. In this way there shall be more structure in the information system. The ideas and examples of constraints are originated from RIDL language[1,2,3]. This article intends to give a more formal approach to the constraints and their language. We use the concept "relation" to interpret roles and paths in semantical networks and to build up constraints. We divide the constraints in two kinds: declarative and non-declarative. The second kind is more powerful than the first.

I INTRODUCTION

1 What is a binary semantical network?

Suppose we want to make some information analysis of certain things in the real world, especially classification, structures and mutual relations. For example, a group of people, some conferences, the people who attend or organize the conferences, etc. are what we are interested in. We have to choose a method to approach this problem, for example, the method of binary semantical networks according to NIAM. In fact, the work of De Troyer et al.[4] is dedicated to the design of such a network. We shall call it CRIS. It is based on the test model for conference organization given by IFIP in 1982. We are also going to use this design for our examples in constraints in this article. We shall give an introduction to this theory here. There are a few important concepts and terminologies in the binary semantical network:
Important concepts are NOLOT (non-lexical object type) and LOT (lexical object type). These concepts are made more clear by considering their difference. We discuss this distinction using the example of an imaginary IFIP conference in June 1992 (one of a number of different conferences), and an attendant of this conference, L. G. Jansen (one of many people). Both these entities are real and unique, even though almost any data about them can change. The whole point of maintaining an information system on this conference is to keep track of the copious amounts of change in data concerning this event.
All data of L. G. Jansen can change as well, for instance her name can turn out to be misspelled, or she can become L. G. Wouters-Jansen. The information system wants to mirror the structure of the real world, and therefore it will contain something that corresponds to these unique entities, abstracted from any properties or characteristics they might have at any given moment.

In the design phase we may think of a real person, with black hair and a smile with a dimple, but when it comes to implementation we make the *system* (or the computer) think of a unique system identifier (surrogate). Sets of such real world entities (or their internal counterparts in the system) we call NOLOTs. Their characteristics like names, dates, amounts of money, addresses, etc. are collected in LOTs.

The system has two kinds of relations: relations between two sets of the real world entities (NOLOTs) and relations between a NOLOT and data about some characteristics (LOTs).

So if Ms Jansen attends the conference mentioned, this will be expressed as part of a relation between two elements, of NOLOTs *person* and *conference*. There certainly will not be any direct relation between Ms Jansen's name-string "Jansen L G" (an element in LOT *person_name*) and the conference identifier "IFIP92june" (an element in LOT *conf_id*).

The distinction between a LOT and a NOLOT is important when we design and implement a semantical network. This article is about constraints which are established after the design. We treat a LOT and a NOLOT almost the same way when we classify and define the constraints. However, in the implementation of a NOLOT, the elements are represented by system identifiers or surrogates. The identifiers are distinct and they are generated by the system. They are not directly changeable by users. We use symbols like #1, #2, etc. to represent these identifiers.

A diagram is often used to express a binary relation. For example, see Fig. 1.

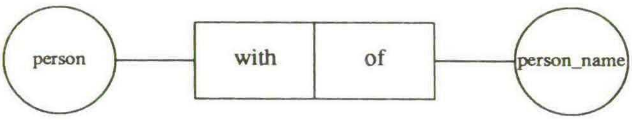

Fig. 1 Typical diagram for a relation.

We can read such a diagram in two ways:

person with person_name
person_name of person

Notice a NOLOT is expressed in solid circle and a LOT is expressed in a dotted circle. The two rectangles are used symbolically to represent a table and this table gives a binary relation. The left column of the table has the name *with* and the right column has the name *of*. We call these columns *roles*. They are also coroles of each other. The table can have for example the following contents at a certain moment:

with	of
#1	Jansen L G
#2	Koffman M
#3	Jansen L G

Note that the same name occurs twice, apparently it belongs to different persons. Although a role means originally a column, we can think of a role as a set of pairs, namely:

of = { (#1, Jansen L G), (#2, Koffman M), (#3, Jansen L G) }

with = { (Jansen L G, #1), (Koffman M, #2), (Jansen L G, #3) }

We can see that an object type contains some elementary data and a role contains pairs of such data.

A design of a binary semantical network can be expressed by a diagram which is the combination of some simpler diagrams like the one above. Such a diagram tells how all object types are related to each other by roles and it is called a *graphical representation of a conceptual scheme* of a semantical network. We say for short *conceptual scheme.* We present such a conceptual scheme in the end of this article. This scheme is called CRIS because it is originated from the scheme in CRIS[4].

2 What are the data types and the permissible operations on the data types?

We distinguish a few data types: strings, numbers and sets. Elementary data are data that are allowed in a LOT or a NOLOT. A LOT consists essentially of numbers only or of strings only. The elements of a NOLOT are essentially different. As we are interested in the abstract theory, we consider elements of NOLOTS temporarily as strings. In a practical situation we can always distinguish more types. Sets are homogeneous: they either consist of strings only, or numbers only, or they consist of pairs of coordinates where the first coordinates (and the second coordinates) are homogeneously all numbers or all strings. Roles are considered as sets of pairs. Operations on numbers, strings and sets are the usual ones: arithmetical and relational operators for numbers, lexicographic comparison for strings, set operations like intersection, union, difference, membership for sets. It is possible to refine "number" into integer, real, also to refine string type by adding a new type "date", but for the time being we keep it simple.

3 What are constraints?

We have to have certain restrictions in inserting, changing or deleting data in a semantical network. We call such restrictions *constraints*. For example, if a conference begins on a date and ends on another date, we expect that the end is not earlier than the beginning. If we give the computer an instruction to check such requirements every time when there is an updating, then a constraint is built into the semantical network. What the computer should do if the check fails, is an implementation detail that does not concern us now. Here we merely classify the constraints.

The same classification and definitions can lead to different designs of constraint languages and a design can be influenced by personal taste and practical use. The grammar (see appendix) and the examples which we give in this article show only one such possibility. In fact, the ideas of the constraints are originated from the original RIDL language[1,2,3]. This article intends to give a more formal approach to the constraints.

Constraints can be represented in three ways: *graphically, declaratively* and *non-declaratively*. These three ways are listed here in order of increasing power. In our discussion we will sometimes mention the graphical representations for illustrative purposes. The CRIS scheme contains also examples of graphical constraints.

Constraints can be defined by boolean expressions because a constraint is a boolean expression which should be true all the time. According to the kinds of constraints, we distinguish between *declarative boolean expressions* and *non-declarative boolean expressions.* The declarative boolean expressions have standard forms and they are designed to express some declarative constraints. On the other hand, a non-declarative boolean expression uses logical operators, mathematical manipulations, etc.

We divide this article into 11 chapters. The following chapter (II) is about some basic mathematical concepts with respect to relations. The chapters III to X are devoted to the declarative constraints. We discuss the constraints in chapter IV-VII only for the situations of one path or two paths. The readers can generalize them to the general situation of n paths with $n \geq 1$. Chapter XI is about non-declarative constraints.

4 Notes on the appendices and implementation.

To try out the syntax of the language we have written a YACC-program for the UNIX-system[5]. This program consists essentially of a grammar and associated actions. We have given the grammar without actions in the appendix. The YACC program interfaces with a LEX program which scans the input and returns the tokens. For determining the tokens we need also some other programs to interface with the conceptual schemes of a semantical network. The LEX and other programs are left out of the appendices. The graphical conceptual scheme of CRIS is also given in the appendix. The graph takes a few pages (C1-C4) and some pages have things in common. In this way you can look up things from one page to other pages via the common part. This graphical representation is transferred by the program RIDL_G, written by my colleagues, to the usual concept scheme.

II RELATIONS AND PATHS

We introduce a system to discuss states of roles, object types and their constraints. We start with basic mathematical concepts.

1 Definition. A *relation* f from a set A to a set B, denoted by

$$f : A \to B$$

is a subset of the Cartesian product $A \times B$. We can then define the following concepts with respect to f.

$$\begin{aligned}
&\text{source}(f) = A \quad , \\
&\text{target}(f) = B \quad , \\
&\text{support}(f) = \{x \in A \mid (x, y) \in f\} \quad , \\
&\text{range}(f) = \{y \in B \mid (x, y) \in f\} \quad , \\
&f(U) = \{b \in B \mid \exists a \in U \text{ such that } (a, b) \in f\} \quad \text{for } U \subset A \quad , \\
&f^{-1} = \{(x, y) \mid (y, x) \in f\} \quad .
\end{aligned}$$

We call f^{-1} the *inverse* of f. If $(x, y) \in f$, then x is an *original* of y and y is an *image* of x. If y is the only image of x, then we can use $f(x)$ to denote y. So we have in this situation $\{f(x)\} = f(\{x\})$.

2 Interpretation. In a semantical network, a state of a *role* from an object-type A to an object-type B is in fact a relation from A to B. The *corole* is just its inverse. For example, we consider the diagram and the table in section 1 of chapter I. It means that we have the following two relations:

$$\text{of} : \text{person} \to \text{person\_name}$$

$$\text{with} : \text{person\_name} \to \text{person}$$

These two roles are inverses of each other. In this example it is clear which role we mean when we only use the name of the role. In a semantical network there are many roles with the same role name. However, we assume that a role will be uniquely defined if the object types on both sides are given too. Thus we use often the following kind of expressions:

$$\textit{person\_name of person}\ (\{\#1\}) = \{\textit{Jansen L G}\}$$

$$\textit{person with person\_name}\ (\{\textit{Jansen L G}\}) = \{\#1, \#3\}$$

3 Definition. A *path* is a composition of known relations, hence also a relation. Given $f_1 : A_1 \to A_2, f_2 : A_2 \to A_3, \ldots, f_n : A_n \to A_{n+1}$, we can define the *composition* $f_n f_{n-1} \cdots f_1$ of $f_1, f_2, \ldots, f_n$ as the relation f from A_1 to A_{n+1} with the following property:

$(x, y) \in f \Leftrightarrow$

there exist $z_2, z_3, \ldots, z_n$

such that $(x, z_2) \in f_1, (z_2, z_3) \in f_2, \ldots, (z_n, y) \in f_n$.

We denote this composition sometimes in the following way:

$$f_n f_{n-1} \cdots f_1 : A_1 \to A_2 \cdots \to A_n \to A_{n+1} \quad .$$

It is clear that source$(f) = A_1$ and target$(f) = A_{n+1}$. Suppose inverses of $f_1, f_2, \ldots, f_n$ are $g_1, g_2, \ldots, g_n$, respectively. It is easy to see that in this case the inverse g of f is given by $g = g_1 g_2 \cdots g_n$. We say g is the *copath* of f. In this case f is also the copath of g.
Let us consider two paths in CRIS case: one is from *date* to *conf_title* and one is from *conf_title* to *date*. See Fig. 2.

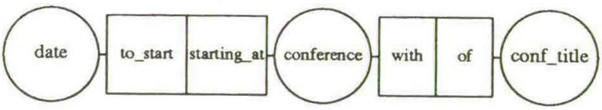

Fig. 2 A path and its inverse in CRIS.

We denote these two examples of paths which are inverse to each other in the following way:

date to_start conference with conf_title

conf_title of conference starting_at date

4 Remark. In a network certain relations, the roles, are given a priori. A "path" is usually a composition of n such roles. Especially, a role is also a path with $n = 1$. In the rest of the article we use often sentences like "f is a path in a semantical network" to mean

$$f = f_n f_{n-1} \cdots f_1 : A_1 \to A_2 \cdots \to A_n \to A_{n+1}$$

where f_i is a role and A_i is an object type for every i and source$(f) = A_1$ and target$(f) = A_{n+1}$.

III CONSTRAINTS ABOUT SUBTYPES

Theoretically, we should discuss the constraints about subtypes after some other declarative constraints. On the other hand we use paths to build up constraints in general, we need to make agreements about paths with respect to subtypes already at beginning. So we begin with constraints about subtypes.

1 Definition. If $A \subset B$ we say A is a *subtype* of B and B is a *supertype* of A. See Fig. 3.

Fig. 3 Subtype and supertype.

We can think of the arrow as a trivial role, namely, the "identity" relation.

2 Path through subtypes. If a role is between a supertype and some object type then we can also consider it as a role between a subtype and that object type. To formulate it more precisely, let us consider A, a subtype of B, $f : B \rightarrow C$ a role. Then define $f_1 : A \rightarrow C$ as

$$f_1 = \{(x, y) \in f \mid x \in A\}$$

Because we always specify the two object types where a role starts and where it ends, we can use f instead of f_1. For example, consider the following path in CRIS (C4):

paper_title of submitted_paper getting paper_ref_assmt on date .

Of is originally a relation from *paper* to *paper_title* but now it is considered as a relation from *submitted_paper* to *paper_title*.
On the other hand, if we have a role between a subtype and some object type, we can also consider it as a role between its supertype and that object type. We only have to consider it as a subset of a bigger Cartesian product.
We can also generalize the idea of subtypes transitively. That means if A is a subtype of B and B is a subtype of C then A is also a subtype of C.

3 Definition. If B has several subtypes $A_1, A_2, \ldots, A_n$, then we say $A_1, A_2, \ldots, A_n$ satisfy the *total constraint for subtypes* with respect to B if $B = A_1 \cup A_2 \cup \cdots \cup A_n$. It shall be clear that the total constraint for subtypes defined here is a special case of the total constraint we shall define later. We often use a diagram to denote the total constraint for subtypes. For example, we have the diagram of Fig. 4 for $n = 3$.

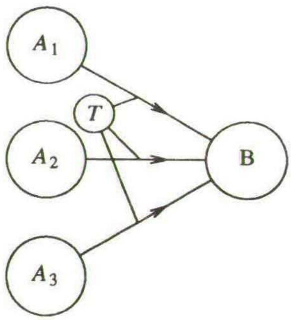

Fig. 4 Total constraint for 3 subtypes.

4 Definition. If $A_1, A_2, \ldots, A_n$ are subtypes of B and $A_i \cap A_j = \varnothing$ for every A_i, A_j where $i \neq j$, then we say these subtypes satisfy *exclusion for subtypes*.

5 Example. If we consider CRIS (C4), we use for example the following way to describe subtype exclusion:

rejected_paper, accepted_paper
SUBTYPE_EXCL submitted_paper

IV UNIQUENESS CONSTRAINTS

1 Convention

In this article we shall assume that roles are sets in the mathematical sense, namely, they do not contain repeated elements. This is considered by others as a special unique constraint.

2 Injective constraints for a path

2.1 Definition. A relation $f : A \to B$ is *injective* if for every $b \in B$ there is at most one $a \in A$ such that $(a, b) \in f$. If we want a path f in a semantical network to be injective then we have an injective constraint for f.

2.2 Diagram. In a diagram the injective constraint for a role f is indicated by a double arrow above or below the box of f.

2.3 Example. Consider the graphical concept scheme of CRIS (C2). We use the following way to express an example of injective constraint:

time IDENTIFIED_BY session_nr of session starting_at time

In other words, if you know the session number of a conference, then you also know the starting time. There can not be two different times for the same session number.

2.4 Proposition. Given

$$f = f_n f_{n-1} \cdots f_1 : A_1 \to A_2 \cdots \to A_n \to A_{n+1}$$

If f_i satisfies the injective constraint for every i, then f satisfies the injective constraint.

3 Injective constraint for more than one path.

3.1 Diagram. The diagram of Fig. 5 represents a uniqueness constraint traditionally with the following two characteristics:

(1) support(f) = support(g).

(2) If (a, b), $(a', b) \in f$ and (a, c), $(a', c) \in g$, then $a = a'$.

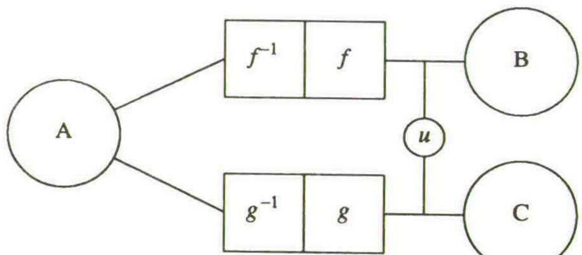

Fig. 5 A uniqueness constraint

3.2 Proposition. Define a relation $h : A \to B \times C$ induced by the paths $f : A \to B$, $g : A \to C$ where support(f) = support(g) as follows:

$$(a, (b, c)) \in h \Leftrightarrow (a, b) \in f \text{ and } (a, c) \in g \quad .$$

Then the h is an injective if and only if f, g have the characteristics (2).

Notice that h is usually denoted by $f \times g$ in mathematics.

3.3 Definition. In a semantical network, a pair (f, g) of paths sharing sources is said to satisfy the *injective constraint* if support(f) = support(g) and $f \times g$ from the same source to the Cartesian product target(f) $\times$ target(g) is injective.

3.4 Example. Consider CRIS (C4). We have an example of an injective constraint of more than one path expressed in the following way:

paper_ref_assmt
IDENTIFIED_BY
person referee_for paper_ref_assmt,
submitted_paper of paper_ref_assmt

This means if a paper and a referee (there can be more than one referee for a paper) are known, then there is at most one paper-referee-assignment which concerns this paper and this referee.

4 Functional constraints

Functionality is the dual concept of injectivity. We mean by this that a functional constraint on a path is equivalent to an injective constraint on its copath.
4.1 Definition. A relation $f : A \to B$ is a *function* if

$$(a, b), (a, b') \in f \Rightarrow b = b' \quad .$$

So if a path f in a semantical network is a function then we say f satisfies the *functional constraint*.
4.2 Diagram. Consider the diagram mentioned in 2.2., where f satisfies the injective constraint. We can also use the same diagram to denote the functional constraint of f^{-1}.
4.3 Remark. If f is a path which satisfies the functional constraint, then the notation $f(a)$ makes sense if the set of all images of a under f is not empty. See also section 1 of chapter II.
4.4 Definition. The pair (f, g) is said to satisfy the functional constraint if the pair (f^{-1}, g^{-1}) satisfies the injective constraint.

V TOTALITY CONSTRAINTS

1 Total constraints

1.1 Definition. Given $f : A \to B$. We say that f is *surjective* if range(f) $= B$. If we have a path f in a semantical network which satisfies the surjective property, we can say f satisfies the *surjective constraint* or the *total constraint.*
1.2 Proposition. Given a path $f = f_n f_{n-1} \cdots f_1$ in a semantical network. If every f_i satisfies total constraint, then f also satisfies the total constraint.
1.3 Diagram. If f is a role, then f satisfies the total constraint is denoted by the following diagram of Fig. 6:

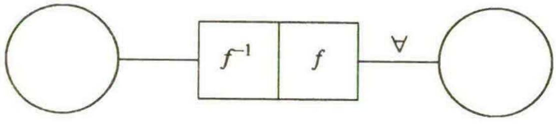

Fig. 6 Total constraint for f

1.4 Definition. Given two paths f and g in a semantical network where target(f) = target(g). Then the pair (f, g) is said to satisfy the *total constraint* if

$$\text{range}(f) \cup \text{range}(g) = \text{target}(f) = \text{target}(g) \quad .$$

1.5 Diagram. In diagram form this is indicated by connecting the boxes for f and g by a “T” in a small circle.

1.6 Proposition. Given two relations $f : A \to C$ and $g : B \to C$. Define $h : A \cup B \to C$, as follows:

$$(x, c) \in h \Leftrightarrow (x, c) \in f \text{ or } (x, c) \in g \quad .$$

So the surjectivity of h is equivalent with the definition of surjectivity of the pair (f, g).
1.7 Example. Consider the concept scheme CRIS (C2). We can give for example the following total constraint with our language:

accepted_paper
TOTAL_IN
accepted_paper presented_in lecture during session,
accepted_paper with abstracts

It means if a paper is accepted, it is either presented in a lecture of a session or it is collected in a bundle of abstracts for the conference.
If you think there are too many "accepted_paper" in this expression, you can change the grammar in the implementation. For clearness we keep this structure for the time being because our definition of a path begins always from an object type.

2 Constraints of total support

2.1 Definition. A relation $f : A \to B$ has *total support* if support$(f) = A$. Thus for a path f in a semantical network, f is said to satisfy the constraint of *total support* if support(f) = source(f). The diagram of Fig. 6 can also be used to denote the constraint of total support of f^{-1}.

2.2 Definition. If support$(f) \cup$ support(g) = source(f) = source(g) for two paths f and g in a semantical network, then the pair (f, g) is said to satisfy the constraint of *total support*.

VI KEY CONSTRAINTS

Key constraint is a kind of combination of total support, injective and functional constraints. Because the concept of "key" is important for database, we need to know the corresponding concept in semantical networks.
1 Definition. Given a path f, then f satisfies the *key constraint* if f is injective, functional and has the property of total support.
This means that the elements of source(f) and range(f) determine each other uniquely. We say b is the key of a if $(a, b) \in f$. Observe that it is possible that there are elements in target(f) that are not in range(f), and hence not key of an element in source(f).
2 Diagram. If we consider the simplest situation that f is a role, then the key constraint can be presented with the diagram of Fig. 7.

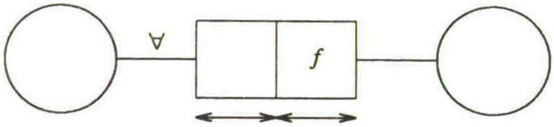

Fig. 7 Key constraint for one role f

3 Definition. Given two paths $f : A \to \cdots \to B$ and $g : A \to \cdots \to C$. The combination (f, g) is said to satisfy the *key constraint* if $f \times g : A \to B \times C$ satisfies the properties of total support, injectivity and functionality.

In other words, if $f \times g : A \to B \times C$ is defined in the same way as theorem 3.2 of chapter IV, then the pair (f, g) satisfies the key constraint if and only if $f \times g$ satisfies the properties of total support, injectivity and functionality.

4 Diagram. Fig. 8 represents the diagram for the key constraint on two roles. It can be proved that the functionality of $f \times g$ is equivalent with the functionality of f and functionality of g together in this situation. That is why we use such diagram to denote the key constraint of (f, g).

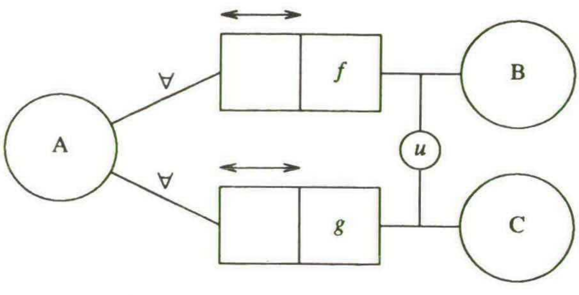

Fig. 8 Key constraint for two roles

5 Example. We can give a simple key constraint in our language for CRIS (C3):

nat_repr_TC
HAVING_KEY
society of nat_repr_TC,
TC of nat_repr_TC,
nat_repr of nat_repr_TC

You can think that a nat_repr_TC is an abstract object which is a combination of three other less abstract objects, namely, a society, a TC, and a national representative. The choice of these three objects is also unique.

VII SUBSET CONSTRAINTS

Subset constraints express a subset relationship between supports, ranges of two paths or a subset relationship between the two paths themselves.

1 Definitions. Let us consider paths f and g in a semantical network. Suppose that $\text{source}(f) = \text{source}(g)$. If $\text{support}(f) \subset \text{support}(g)$, then we say the pair (f, g) satisfies the *subsupport constraint*. Now suppose f and g share the same target, rather than the same source. If $\text{range}(f) \subset \text{range}(g)$, then we say the pair (f, g) satisfies the *subrange constraint*. Now suppose that both $\text{source}(f) = \text{source}(g)$ and $\text{target}(f) = \text{target}(g)$. We say the pair (f, g) satisfies the *subpath constraint* if $f \subset g$. That is to say, every element (x, y) in f is also an element in g.

2 Diagrams. The subsupport constraint on two roles can be denoted by the diagram of Fig. 9. The diagram of Fig. 10 is sometimes used to denote the subpath constraint in the simplest situation, namely, relationship between two roles.

3 Examples. With our language we can express an example of subsupport constraint in CRIS (C2) in the following way:

person presenting acc_paper
SUB_SUPPORT
session comprising lecture about acc_paper

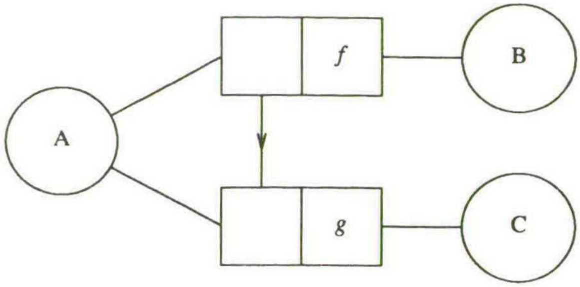

Fig 9 Subsupport constraint

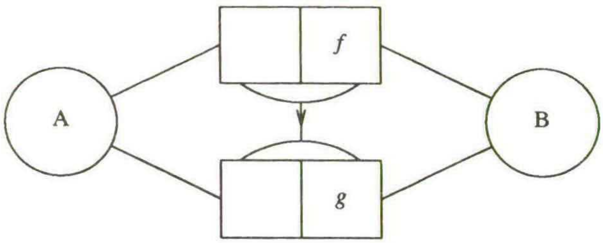

Fig. 10 A simple subpath constraint

This constraint requires that if an accepted paper is known to be presented by someone, then it must be already scheduled for a lecture in a session.
We can give an example of a subpath constraint in CRIS (C1):

member_country holding conference
SUBPATH_OF
member_country location_of body of conference

This constraint requires that the conference must be held in a country where one of the sponsoring bodies of the conference also is located.

VIII EQUIVALENCE CONSTRAINTS

The equivalence constraints expresses the set equality between supports or ranges of two paths or with the set equality between the paths themselves.
1 Definitions. Let us consider two paths f and g in a semantical network. If f and g share the same source, we say they are *support equivalent* if support(f) = support(g). Dually, if they share the same target, we say they are *range equivalent* if range(f) = range(g). If they share the same source and also the same target then we say f, g are *path equivalent* if $f = g$.
If f, g are roles then we can express these properties in diagrams. They are similar to the diagrams for subset constraints. We use =-signs in a small circle in the appropriate places to indicate the equality between sets.
2 Examples. We give now an example of support equivalence for CRIS (C2) with our language.

time starting session SUPPORT_EQ time ending session

This requires that if the starting time of a session is known, then the ending time is known too and vice versa. A path equivalence of two paths in CRIS (C1)can be expressed in the following ways:

org_unit being IFIP_sponsor of conference
PATHEQ
org_unit organizing conference

This constraint requires that an organization unit which is involved in the organization of the conference is automatically an IFIP-sponsor of the conference and vice versa.

IX EXCLUSION CONSTRAINTS

1 Exclusion constraints are always about the disjoint property between supports of two paths, ranges of two paths and the disjoint property between the two paths themselves. They can be formulated just like subset constraints, equivalence constraints. We only have to change support(f) $\subset$ support(g) into support(f) $\cap$ support(g) $= \varnothing$, for example.
2 **Diagrams.** When we consider the simplest path, namely, f, g are roles, then we can use diagrams to express the exclusion constraints. The diagrams look like the diagrams for subset, equivalence constraints. You use an "X" to denote the exclusive property.
3 **Example.** Consider the concept scheme CRIS (C4).
An example of a path exclusion can be expressed in the following way:

person author_of submitted_paper
PATHEXCL
person referee_for paper_ref_assmt of submitted_paper

This example simply tells that the author of a paper is not the referee of the same paper.

X NUMERIC CONSTRAINTS

This is a very simple declarative constraint. There are two kinds of object types. One kind is a set of strings. The other kind is a set of numbers. We need to know which ones are of number types, then we also know which ones are not. This constraint tells that an object type is a number type. We can give the following example to illustrate how it is defined in our language:

NUMERIC conf_fee, hotel_rate, fee, expense, days, years

XI NON-DECLARATIVE CONSTRAINTS

1 Basic building bricks for non-declarative constraints

"Path" is an important concept to build up the declarative constraints as well as non-declarative constraints (see also Meersman[2] for the original examples). Let us take an example of a path in CRIS (C1),

expense of conference starting_at date with year

Consider some simple data in these roles:

	with	starting_at	of
year	date	conference	expense
88	880101	#1	20000
88	880304	#2	30000
88	880501	#3	20000
89	890405	#4	40000

If we have a constraint which requires the expense of a conference in 1988 not to exceed 40000 dollars, then we have to do with the image set of 88, namely $\{20000, 30000\}$.

$$\{(88, 20000), (88, 30000)\}$$

The second coordinate may not exceed 40000. On the other hand, if we require the sum of the expenses of conferences in 1988 not to exceed 100000 dollars, then we should break this path in two pieces with conferences as boundary.

$$\text{conference starting\_at date with year}(88) = \{\#1, \#2, \#3\}$$

$$\{(\#1, 20000), (\#2, 30000), (\#3, 20000)\} \subset \text{expense of conference}$$

The sum of expenses is the sum of the second coordinates of the three elements.
We notice what we need for the constraints are concepts of paths, subsets of paths, sum of a number set, sums of coordinates of a set of pairs, maximum of a set, etc. So we try to define some standard functions for these concepts. We use such functions and paths as the building bricks of number expressions or set expressions. These expressions are again used to build up boolean expressions. The boolean expressions are the basis for constraints.

2 Standard set functions

Our sets are sets of strings, sets of numbers, sets of pairs where the first coordinates (and the second coordinates) are homogeneously either all numbers or all strings.
2.1 Definition. Given a role $f : A \to B$ in a semantical network. We have

$$f(\{a\}) = \{b \in B \mid (a, b) \in f\} \quad .$$

The way to find such a set for a given role and an element is a *standard function*. With this as basis we can define another two standard functions:

$$f(U) = \cup \{f(\{a\}) \mid a \in U\} \quad .$$

$$f \mid U = \{(a, b) \in f \mid a \in U \text{ and } b \in f(\{a\})\} \quad .$$

The last formula is the *restriction* of f to U.
We can generalize the functions above to $f = f_n f_{n-1} \cdots f_1$, because

$$f(U) = f_n(f_{n-1} \cdots (f_1(U) \cdots))$$

Evidently, the type (string set or number set) of $f(U)$ is the same as the type of the last object type of the path. The type of $f \mid U$ is the same as the type of f.
2.3 Definition. If g is a set of pairs with the first (and the second) coordinates homogeneously either numbers or strings, then

SET1 $(g) =$ the set of the first coordinates of g ;

SET2 $(g) =$ the set of the second coordinates of g ;

$\text{INV}(g) = \{(y, x) \mid (x, y) \in g\}$.

A special case is when $g = f \mid U$. In fact, $f(U) = \mathrm{SET2}(f \mid U)$.

3 Standard number and string functions

3.1 Definition. For a set S of numbers or strings or a set of pairs S in a semantical network we define the following function:

$$CARD\,(S) = \#\{x \mid x \in S\} \quad .$$

So $CARD\,(f)$ is the number of elements in the path f; $CARD\,(f\,(U))$ is the number of elements of image set of U under f, etc.
3.2 Definition. If we have a set of strings or a set of numbers, then we can compare two such elements (strings lexicographically and numbers in the usual way). Thus there exist a maximum and a minimum of such a set. So if S is a set of numbers, then

$$\mathrm{MAX}\,(S) = x, \text{ where } x \in S \text{ and } x \geq y \text{ for every } y \in S\,;$$
$$\mathrm{MIN}\,(S) = x, \text{ where } x \in S \text{ and } x \leq y \text{ for every } y \in S \quad .$$

If S is a set of strings, then we use MAXSTR, MINSTR instead of MAX, MIN, respectively.
3.3 Definition. Let us consider a set of numbers $S = \{a_1, a_2, \cdots, a_n\}$, where $n \geq 1$, then

$$\mathrm{SUM}(S) = a_1 + a_2 + \cdots + a_n\,;$$

$$\mathrm{AVG}(S) = \mathrm{SUM}(S)/\mathrm{CARD}(S)$$

3.4 Definition. We want to pay special attention to sum and average of the coordinates of a subset of a relation. (See section 1 for motivation.) Let $K = \{(a_1, b_1), (a_2, b_2), \cdots, (a_n, b_n)\}$:

$$\mathrm{SUM1}\,(K) = a_1 + a_2 \cdots + a_n, \text{ if the } a_i \text{ are numbers} \;;$$

$$\mathrm{SUM2}\,(K) = b_1 + b_2 \cdots + b_n, \text{ if the } b_i \text{ are numbers} \;;$$

AVG1, AVG2 are defined analogously to AVG.

4 Expressions and operators.

There are essentially three kinds of expressions: number, set and boolean. Constraints are just boolean expressions with special meanings, namely, they should always get *true* as value. This article concentrates on constraints, so numbers and sets are only the basis for constructing boolean expressions.

4.1 Operators. If we have number expressions, we can do mathematical computations with them. The operations are the binary +, –, *, / and unary operation minus –.
From two sets of the same type we can construct a third set by using the set operators: UNION, INTERSECT, MINUS. The mathematical meanings of such operations are well known.
We can compare two numbers by the relational operators: $<, \leq, >, \geq, \neq, =$. We can also compare two sets with the relational operators: $\subseteq, \supseteq, =, \neq$. We can also compare two strings lexicographically with the same relational operators as for numbers. There is a special relational operator IN ($\in$) which establishes membership of a set.
To construct a new boolean expression from simpler boolean expressions, we can use the boolean operators like AND, OR, NOT, IFF, IMPLY.
4.2 Number expressions. Number expressions are essentially made of number constants, number identifiers and results of number expressions. Furthermore, we can call a standard function which yields a number as the result.
4.3 Set expressions. The simplest set expression are empty set, an object type, a role, a path, an explicit expression of a set of numbers or strings and an explicit expression of a set of pairs where

the first coordinates (and the second coordinates) are all numbers or are all strings. One can also call a standard function which deliver a set as value. Furthermore one can use set operators to combine two set expressions of the same type in order to get a new set expressions.

4.4 Boolean expressions. The simplest boolean expressions are the constants TRUE and FALSE. One can also apply all kinds of relational operators in sets, numbers and strings to obtain a boolean expression. One can combine boolean expressions with operators like IFF, OR, IMPLY to obtain new boolean expressions. Furthermore we have the special boolean expressions as follows:

```
IF boolean expression THEN boolean expression
IF boolean expression THEN boolean expression
  ELSE boolean expression
FOR EACH identifier FROM set expression,
              identifier FROM set expression, ... :
   boolean expression
```

The first and the second constructions look like a standard *if* statement. We can also use IMPLY instead of *if* and *then* in the first expression. The third construction can be explained by the following example:

```
FOR EACH x FROM conference:
   expense of conference (x) <= 200000
```

means no conference should spend more than 200000 dollars. It delivers *false* as value if some conference spends more than this amount. Notice the type of the identifier is determined by the set expression after FROM. We are allowed to use the notation *expense of conference* (*x*) because of the functional constraint of the path. (See chapter II).

5 Examples of non-declarative constraints

All the examples here are originated from CRIS. The constraint language uses sometimes a notation that differs from the mathematical notation, for example <> instead of $\neq$, and $[x]$ instead of $\{x\}$.

5.1 Example. We want the expense of every conference which starts in 1988 not to exceed 40000 dollars (CRIS C1).

```
for each x from conference:
if year of date to_start conference (x) = 88
then expense of conference (x) <= 40000
```

5.2 Example. We want to construct a constraint which requires that the sum of all expenses of conferences in 1988 be less than or equal to 100000 dollars. We have in section 1 of this chapter explained why we need to break the path *expense of conference starting_at date with year* in two pieces at the point of *conference*. The essential idea is that the sum has to range over all conferences starting at 1988. Thus we should formulate this constraint in the following way (CRIS C1):

```
SUM2(expense of conference
      | conference starting_at
        date with year ([88])) <= 100000
```

where *expense of conference* is a relation. This relation is restricted to a subset in *conference*, namely, the set of images of 88 under the relation *conference starting_at date with year*.

5.3 Examples. We require that the same person may not submit more than 3 papers per conference. This constraint can be formulated as follows (CRIS C4):

```
for each x from conference:
  for each y from
      person author_of
      submitted_paper for conference([x]):
        CARD (submitted_paper for
              conference([x]) intersect
              submitted_paper written_by
              person([y])) <= 3
```

Notice that from the second FOR EACH until the end is the boolean expression needed for after "conference:" in the first FOR EACH expression.
We can also formulate the same constraint in a shorter way:

```
for each x from conference:
  for each y from person:
    CARD( submitted_paper written_by
          person([y]) intersect
          submitted_paper for
          conference([x])) <=3
```

Acknowledgements. Hereby I thank Professor R. Meersman for his advices. Furthermore, I have also used the RIDL_G^3 developed by my colleagues for drawing the graphical conceptual scheme and mapping the graphical representation to the usual conceptual scheme.

REFERENCES

1. Wintraecken, J.J.V.R., *Informatie-analyse volgens NIAM*. Academic Service, The Hague (1985). English translation to appear, published by Reidel Company, Holland.
2. Meersman, R., *The RIDL conceptual language*. Control Data DMRL Report (1982).
3. Verheijen, G.M.A., J. van Beekum, *NIAM: An information analysis method*. Proceedings of IFIP WG 8.1 Working Conference on Comparative Review of Information Systems Design Methodologies (1982).
4. De Troyer, O., R. Meersman, P. Verlinden, *RIDL* on the CRIS case: A workbench for NIAM*. Infolab, Tilburg University (1988).
5. Kernighan, B.W., R. Pike, *UNIX Programming Environment*. Prentice-Hall, Inc., USA (1984).

APPENDIX: GRAMMAR FOR YACC

Remarks. In the following ε stands for empty, not to be confused with EMPTY which corresponds to the empty set (of numbers, strings, etc.). The following production rules are almost identical to those of the YACC source. We use *be*, *ne* and *se* for boolean expression, numerical expression and string expression, respectively.

/* constrform is a collection of constraints */

constrform	:: CONSTRAINTS BEGCONS constraints ENDCONS
constraints	:: ε | constraints constraint | constraints error ';'
constraint	:: be ';'
be	:: declarative | nondeclarative

```
declarative      :: subtype | uniqueness | totality | key | subset | equivalence | exclusion |
                    numeric
subtype          :: subtypedef | subtypetotal | subtypeexcl
uniqueness       :: ot IDENTIFIED paths
totality         :: ot TOTAL paths
key              :: ot KEY paths
subset           :: subsupport | subpath
equivalence      :: suppeq | patheq
exclusion        :: suppexcl | pathexcl
subtypedef       :: SUBTYPE ots OF STROT
subtypetotal     :: ots SUBTYPETOTAL ot
subtypeexcl      :: ots SUBTYPEEXCL ot
subsupport       :: path SUBSUPPORT path
subpath          :: path SUBPATH path
suppeq           :: path SUPPEQ path
patheq           :: path PATHEQ path
suppexcl         :: path SUPPEXCL path
pathexcl         :: path PATHEXCL path
numeric          :: NUMERIC ots
 /* ots, paths */
ots              :: ot ots1
ots1             :: ε | ',' ot ots1
ot               :: NUMOT | STROT
paths            :: path paths1
paths1           :: ε | ',' path paths1
path             :: strtonumpath | strtostrpath | numtostrpath | numtonumpath
strtonumpath     :: NUMOT refersstr
strtostrpath     :: STROT refersstr
numtostrpath     :: STROT refersnum
numtonumpath     :: NUMOT refersnum
refersstr        :: refers1 ROLE STROT
refersnum        :: refers1 ROLE NUMOT
refers1          :: ε | refers1 refer
refer            :: ROLE ot
/* nondeclarative boolean expressions are defined here */
 /* the other expressions can be sets, numbers, strings, set of pairs */
nondeclarative   :: simplebe | NOT be | be AND be | be OR be | be IFF be | be IMPLY be | IF
                    be THEN be ELSE be | IF be THEN be | foreach be
simplebe         :: TRUE | FALSE | '(' be ')' | ne rel ne | se rel se | strnum rel strnum | numstr
                    rel numstr | numnum rel numnum | strstr rel strstr | numset rel numset | strset
                    rel strset | numnumset rel numnumset | numstrset rel numstrset | strnumset rel
                    strnumset | ne IN numset | se IN strset | strnums IN strnumset | strstrs IN
                    strstrset | numnums IN numnumset | numstrs IN numstrset | empty1 | empty2 |
                    empty3 | empty4 | empty5 | empty6
rel              :: GEQ /* >= */ | GR /* > */ | EQ /* = */ | UNEQ /* <> */ | LEQ /* <= */ | LE
                    /* < */
empty1           :: EMPTY rel numset | numset rel EMPTY
empty2           :: EMPTY rel strset | strset rel EMPTY
empty3           :: EMPTY rel numnumset | numnumset rel EMPTY
empty4           :: EMPTY rel strstrset | strstrset rel EMPTY
```

empty5 :: EMPTY rel numstrset | numstrset rel EMPTY

empty6 :: EMPTY rel strnumset | strnumset rel EMPTY

foreach :: FOREACH idsfromexprs ':'

idsfromexprs :: idsfromexprs1 idsfromexpr

idsfromexprs1 :: ε | idsfromexprs1 ',' idsfromexpr

idsfromexpr :: ids FROM numset | ids FROM strset | idpairs FROM numnumset | idpairs FROM numstrset | idpairs FROM strstrset | idpairs FROM strnumset

ids :: ids1 IDENTIFIER

ids1 :: ε | ids1 IDENTIFIER ','

idpairs :: idpairs1 idpair

idpairs1 :: ε | idpairs1 idpair ','

idpair :: '(' IDENTIFIER ',' IDENTIFIER ')'

ne :: simplene | '-' ne %prec UMINUS | ne numop ne

simplene :: NUMBER | NUMVAR | '(' ne ')' | strtonumpath '(' se ')' | numtonumpath '(' ne ')' | standardnumfunc

numop :: '+' | '-' | '*' | '/'

standardnumfunc :: CARD '(' setexpr ')' | SUM '(' numset ')' | MAX '(' numset ')' | MIN '(' numset ')' | AVG '(' numset ')' | SUM1 '(' numstrset ')' | SUM1 '(' numnumset ')' | SUM2 '(' strnumset ')' | SUM2 '(' numnumset ')' | AVG1 '(' numstrset ')' | AVG1 '(' numnumset ')' | AVG2 '(' strnumset ')' | AVG2 '(' numnumset ')'

se :: strconst | STRVAR | numtostrpath '(' ne ')' | strtostrpath '(' se ')' | standardstrfunc

strconst :: '"' IDENTIFIER '"' | '"' NUMBER '"'

standardstrfunc :: MAXSTR '(' strset ')' | MINSTR '(' strset ')'

setexpr :: numset | strset | numnumset | strnumset | strstrset | numstrset

numset :: NUMOT | strtonumpath '(' strset ')' | numtonumpath '(' numset ')' | '[' nes ']' | numset setop numset | standardnumsetfunc | '(' numset ')'

nes :: ne nes1

nes1 :: ε | ',' ne nes1

ses :: se ses1

ses1 :: ε | ',' se ses1

standardnumsetfunc:: SET1 '(' numstrset ')' | SET1 '(' numnumset ')' | SET2 '(' strnumset ')' | SET2 '(' numnumset ')'

setop :: INTERSECT | UNION | MINUS

strset :: STROT | numtostrpath '(' numset ')' | strtostrpath '(' strset ')' | '[' ses ']' | strset setop strset | standardstrsetfunc | '(' strset ')'

standardstrsetfunc :: SET1 '(' strnumset ')' | SET1 '(' strstrset ')' | SET2 '(' strstrset ')' | SET2 '(' numstrset ')'

numnumset :: NUMOT 'X' NUMOT | numtonumpath | numtonumpath '|' numset | '[' numnums ']' | numnumset setop numnumset | INV '(' numnumset ')'

numnums :: numnum numnums1

numnums1 :: ε | ',' numnum numnums1

numnum :: '(' ne ',' ne ')'

strstrset :: STROT 'X' STROT | strtostrpath | strtostrpath '|' strset | '[' strstrs ']' | strstrset setop strstrset | INV '(' strstrset ')' | '(' strstrset ')'

strstrs :: strstr strstrs1

strstrs1 :: ε | ',' strstr strstrs1

strstr :: '(' se ',' se ')'

numstrset :: NUMOT 'X' STROT | numtostrpath | numtostrpath '|' numset | '[' numstrs

	']' | INV '(' strnumset ')' | '(' numstrset ')'
numstrs	:: numstr numstrs1
numstrs1	:: ε | ',' numstr numstrs1
numstr	:: '(' ne ',' se ')'
strnumset	:: STROT 'X' NUMOT | strtonumpath | strtonumpath '|' strset | '[' strnums ']' | INV '(' numstrset ')' | '(' strnumset ')'
strnums	:: strnum strnums1
strnums1	:: ε | ',' strnum strnums1
strnum	:: '(' se ',' ne ')'

IN 1987 REEDS VERSCHENEN

242 Gerard van den Berg
Nonstationarity in job search theory

243 Annie Cuyt, Brigitte Verdonk
Block-tridiagonal linear systems and branched continued fractions

244 J.C. de Vos, W. Vervaat
Local Times of Bernoulli Walk

245 Arie Kapteyn, Peter Kooreman, Rob Willemse
Some methodological issues in the implementation of subjective poverty definitions

246 J.P.C. Kleijnen, J. Kriens, M.C.H.M. Lafleur, J.H.F. Pardoel
Sampling for Quality Inspection and Correction: AOQL Performance Criteria

247 D.B.J. Schouten
Algemene theorie van de internationale conjuncturele en strukturele afhankelijkheden

248 F.C. Bussemaker, W.H. Haemers, J.J. Seidel, E. Spence
On (v,k,λ) graphs and designs with trivial automorphism group

249 Peter M. Kort
The Influence of a Stochastic Environment on the Firm's Optimal Dynamic Investment Policy

250 R.H.J.M. Gradus
Preliminary version
The reaction of the firm on governmental policy: a game-theoretical approach

251 J.G. de Gooijer, R.M.J. Heuts
Higher order moments of bilinear time series processes with symmetrically distributed errors

252 P.H. Stevers, P.A.M. Versteijne
Evaluatie van marketing-activiteiten

253 H.P.A. Mulders, A.J. van Reeken
DATAAL - een hulpmiddel voor onderhoud van gegevensverzamelingen

254 P. Kooreman, A. Kapteyn
On the identifiability of household production functions with joint products: A comment

255 B. van Riel
Was er een profit-squeeze in de Nederlandse industrie?

256 R.P. Gilles
Economies with coalitional structures and core-like equilibrium concepts

257 P.H.M. Ruys, G. van der Laan
Computation of an industrial equilibrium

258 W.H. Haemers, A.E. Brouwer
Association schemes

259 G.J.M. van den Boom
Some modifications and applications of Rubinstein's perfect equilibrium model of bargaining

260 A.W.A. Boot, A.V. Thakor, G.F. Udell
Competition, Risk Neutrality and Loan Commitments

261 A.W.A. Boot, A.V. Thakor, G.F. Udell
Collateral and Borrower Risk

262 A. Kapteyn, I. Woittiez
Preference Interdependence and Habit Formation in Family Labor Supply

263 B. Bettonvil
A formal description of discrete event dynamic systems including perturbation analysis

264 Sylvester C.W. Eijffinger
A monthly model for the monetary policy in the Netherlands

265 F. van der Ploeg, A.J. de Zeeuw
Conflict over arms accumulation in market and command economies

266 F. van der Ploeg, A.J. de Zeeuw
Perfect equilibrium in a model of competitive arms accumulation

267 Aart de Zeeuw
Inflation and reputation: comment

268 A.J. de Zeeuw, F. van der Ploeg
Difference games and policy evaluation: a conceptual framework

269 Frederick van der Ploeg
Rationing in open economy and dynamic macroeconomics: a survey

270 G. van der Laan and A.J.J. Talman
Computing economic equilibria by variable dimension algorithms: state of the art

271 C.A.J.M. Dirven and A.J.J. Talman
A simplicial algorithm for finding equilibria in economies with linear production technologies

272 Th.E. Nijman and F.C. Palm
Consistent estimation of regression models with incompletely observed exogenous variables

273 Th.E. Nijman and F.C. Palm
Predictive accuracy gain from disaggregate sampling in arima - models

289 M.H.C. Paardekooper
A Quadratically convergent parallel Jacobi process for almost diagonal matrices with distinct eigenvalues

290 Pieter H.M. Ruys
Industries with private and public enterprises

291 J.J.A. Moors & J.C. van Houwelingen
Estimation of linear models with inequality restrictions

292 Arthur van Soest, Peter Kooreman
Vakantiebestemming en -bestedingen

293 Rob Alessie, Raymond Gradus, Bertrand Melenberg
The problem of not observing small expenditures in a consumer expenditure survey

294 F. Boekema, L. Oerlemans, A.J. Hendriks
Kansrijkheid en economische potentie: Top-down en bottom-up analyses

295 Rob Alessie, Bertrand Melenberg, Guglielmo Weber
Consumption, Leisure and Earnings-Related Liquidity Constraints: A Note

296 Arthur van Soest, Peter Kooreman
Estimation of the indirect translog demand system with binding nonnegativity constraints

IN 1988 REEDS VERSCHENEN

297 Bert Bettonvil
Factor screening by sequential bifurcation

298 Robert P. Gilles
On perfect competition in an economy with a coalitional structure

299 Willem Selen, Ruud M. Heuts
Capacitated Lot-Size Production Planning in Process Industry

300 J. Kriens, J.Th. van Lieshout
Notes on the Markowitz portfolio selection method

301 Bert Bettonvil, Jack P.C. Kleijnen
Measurement scales and resolution IV designs: a note

302 Theo Nijman, Marno Verbeek
Estimation of time dependent parameters in lineair models
using cross sections, panels or both

303 Raymond H.J.M. Gradus
A differential game between government and firms: a non-cooperative
approach

304 Leo W.G. Strijbosch, Ronald J.M.M. Does
Comparison of bias-reducing methods for estimating the parameter in
dilution series

305 Drs. W.J. Reijnders, Drs. W.F. Verstappen
Strategische bespiegelingen betreffende het Nederlandse kwaliteits-
concept

306 J.P.C. Kleijnen, J. Kriens, H. Timmermans and H. Van den Wildenberg
Regression sampling in statistical auditing

307 Isolde Woittiez, Arie Kapteyn
A Model of Job Choice, Labour Supply and Wages

308 Jack P.C. Kleijnen
Simulation and optimization in production planning: A case study

309 Robert P. Gilles and Pieter H.M. Ruys
Relational constraints in coalition formation

310 Drs. H. Leo Theuns
Determinanten van de vraag naar vakantiereizen: een verkenning van
materiële en immateriële factoren

311 Peter M. Kort
Dynamic Firm Behaviour within an Uncertain Environment

312 J.P.C. Blanc
A numerical approach to cyclic-service queueing models

313 Drs. N.J. de Beer, Drs. A.M. van Nunen, Drs. M.O. Nijkamp
Does Morkmon Matter?

314 Th. van de Klundert
Wage differentials and employment in a two-sector model with a dual labour market

315 Aart de Zeeuw, Fons Groot, Cees Withagen
On Credible Optimal Tax Rate Policies

316 Christian B. Mulder
Wage moderating effects of corporatism
Decentralized versus centralized wage setting in a union, firm, government context

317 Jörg Glombowski, Michael Krüger
A short-period Goodwin growth cycle

318 Theo Nijman, Marno Verbeek, Arthur van Soest
The optimal design of rotating panels in a simple analysis of variance model

319 Drs. S.V. Hannema, Drs. P.A.M. Versteijne
De toepassing en toekomst van public private partnership's bij de grote en middelgrote Nederlandse gemeenten

320 Th. van de Klundert
Wage Rigidity, Capital Accumulation and Unemployment in a Small Open Economy

321 M.H.C. Paardekooper
An upper and a lower bound for the distance of a manifold to a nearby point

322 Th. ten Raa, F. van der Ploeg
A statistical approach to the problem of negatives in input-output analysis

323 P. Kooreman
Household Labor Force Participation as a Cooperative Game; an Empirical Model

324 A.B.T.M. van Schaik
Persistent Unemployment and Long Run Growth

325 Dr. F.W.M. Boekema, Drs. L.A.G. Oerlemans
De lokale produktiestructuur doorgelicht.
Bedrijfstakverkenningen ten behoeve van regionaal-economisch onderzoek

326 J.P.C. Kleijnen, J. Kriens, M.C.H.M. Lafleur, J.H.F. Pardoel
Sampling for quality inspection and correction: AOQL performance criteria

342 W.J. Selen and R.M. Heuts
A modified priority index for Günther's lot-sizing heuristic under capacitated single stage production

343 Linda J. Mittermaier, Willem J. Selen, Jeri B. Waggoner, Wallace R. Wood
Accounting estimates as cost inputs to logistics models

344 Remy L. de Jong, Rashid I. Al Layla, Willem J. Selen
Alternative water management scenarios for Saudi Arabia

345 W.J. Selen and R.M. Heuts
Capacitated Single Stage Production Planning with Storage Constraints and Sequence-Dependent Setup Times

346 Peter Kort
The Flexible Accelerator Mechanism in a Financial Adjustment Cost Model

347 W.J. Reijnders en W.F. Verstappen
De toenemende importantie van het verticale marketing systeem

348 P.C. van Batenburg en J. Kriens
E.O.Q.L. - A revised and improved version of A.O.Q.L.

349 Drs. W.P.C. van den Nieuwenhof
Multinationalisatie en coördinatie
De internationale strategie van Nederlandse ondernemingen nader beschouwd

350 K.A. Bubshait, W.J. Selen
Estimation of the relationship between project attributes and the implementation of engineering management tools

351 M.P. Tummers, I. Woittiez
A simultaneous wage and labour supply model with hours restrictions

352 Marco Versteijne
Measuring the effectiveness of advertising in a positioning context with multi dimensional scaling techniques

353 Dr. F. Boekema, Drs. L. Oerlemans
Innovatie en stedelijke economische ontwikkeling

354 J.M. Schumacher
Discrete events: perspectives from system theory

355 F.C. Bussemaker, W.H. Haemers, R. Mathon and H.A. Wilbrink
A (49,16,3,6) strongly regular graph does not exist

356 Drs. J.C. Caanen
Tien jaar inflatieneutrale belastingheffing door middel van vermogensaftrek en voorraadaftrek: een kwantitatieve benadering

357 R.M. Heuts, M. Bronckers
A modified coordinated reorder procedure under aggregate investment and service constraints using optimal policy surfaces

358 B.B. van der Genugten
Linear time-invariant filters of infinite order for non-stationary processes

359 J.C. Engwerda
LQ-problem: the discrete-time time-varying case