CHAPTER I
STANDARD SECOND ORDER LOGIC.

1.- INTRODUCTION.

1.1. General idea.

Second order logic (**SOL**) is distinguished from first order logic in that it has relational as well as individual variables, and both types of variables can be quantified. Since it was Frege who pioneered the use of relational variables, second order logic is over one hundred years old; but the effective distinction between first and second order logic took the work of a few other people[1]. It was, in fact, implicit in Russell's work, but it was not made explicit until the work of Hilbert & Ackermann [1928]. First order logic was indeed only a fragment of the highly expressive language introduced by Frege [1879] and Russell [1908].

Thus, in second order logic we can say: "for all individuals, φ holds", as in first order logic, and formalize it as $\forall x\varphi$. We can also say: "for all properties, φ holds", unlike first order logic, and write it as $\forall X\varphi$. By $\forall X^2\varphi$ we express: "for all binary relations, φ holds". And so on....

Therefore, second order structures must contain different domains: the domain of individuals **A**, for variables of individual sort to range on; the unary relational domain $\mathbf{A}_1$, as the range for the unary relational variables; the binary relational domain $\mathbf{A}_2$, and so on. When we wish our second order logic to be standard, we want the formula $\forall X\varphi$ to mean: "for all possible subsets of **A**, φ holds". While we are doing that, we are taking the notion of subset from the background set theory we are using as metalanguage. That forces us to include in $\mathbf{A}_1$ all subsets of **A**, even all those ghostly sets we could never describe or define. Consider the situation where **A** is an infinite set, say of cardinality α; $\mathcal{P}$A, the power set of **A**, which is the standard $\mathbf{A}_1$, contains 2^{α} elements, whereas our formal language has just $\aleph_0$ formulas.

[1] See the historical notes in Church [1956], page 288.

We will see that adding to our universe of sets, $\mathbf{A}_1$, all the sets picked up by the so little descriptive quality of being a subset of **A**, which is the standard definition of structure, gives us a nonabsolute logic; ***i.e.***, a logic whose concept of truth depends on the background set theory.

1.2. Expressive power [2].

Anyway, second order logic with standard semantics (based upon standard structures) has great expressive power (too much, we might say).
For instance:

(1) ***Arithmetical induction*** can be formulated and would retain all its expressive power as

$$\forall X(Xc \wedge \forall x(Xx \rightarrow Xox) \rightarrow \forall x\, Xx)$$

This formula says: Any property which holds for zero and for the successor of any number having this property is a property of all numbers.

(2) The ***identity of individuals*** can be defined and not be, as in first order logic, a primitive relation. The most popular definition of identity is the Leibniz one, which in second order logic becomes:

$$\forall xy(x=y \leftrightarrow \forall X(Xx \leftrightarrow Xy))$$

This formula says: Two individuals are equal if and only if they share all their properties.

(3) The intuitive notion "most **R** are **S**" (***i.e.***, most things having property **R** also have property **S**), which is not expressible in a first order language with equality and with two unary relation symbols for **R** and **S**, can be expressed in **SOL** as

$$\neg\exists X^2(\forall x(\exists y\, X^2xy \leftrightarrow Rx \wedge Sx) \wedge \forall x(\exists y\, X^2yx \rightarrow Rx \wedge \neg Sx) \wedge \forall xyz(X^2xy \wedge X^2xz \rightarrow y=z)$$
$$\wedge \forall xyz(X^2xy \wedge X^2zy \rightarrow x=z))$$

This **SOL** formula expresses: there is no one-to-one function from **R**∩**S** into **R**–**S**. Everybody agrees that it captures the intuitive meaning of "most **R** are **S**", since it says that the set **R**∩**S** is "bigger" than the set **R**–**S**.

(4) Both ***finiteness*** and ***infinity***[3] can be formulated by a single formula. For instance,

[2] All the questions raised here are revisited in section 4.8.

[3] For more information about axioms of infinity, see Alonzo Church [1956], page 342. There you will find some

finiteness can be written as

$$\forall F(\forall xy(Fx=Fy \rightarrow x=y) \rightarrow \forall x\exists y\ x=Fy)$$

(every one-to-one function, **f**: **A** $\longrightarrow$ **A**, on the whole universe of individuals, **A**, is also onto).

In fact, we are using functional variables in this formulation, but they can be easily removed in favor of relational ones by explicitly saying that our binary relation is a function with the whole universe as its domain. Can you write it with relation variables only?

(5) The axioms of ***well ordering***. When $\leq$ is an ordering, the formula

$$\forall X(\exists y\ Xy \rightarrow \exists u(Xu \wedge \forall z(Xz \rightarrow u\leq z)))$$

expresses that all non-empty subsets have a least element.

(6) The ***comprehension axioms***, stating that all definable relations exist.

$$\exists X^n \forall x_1 ... x_n (X^n x_1 ... x_n \leftrightarrow \varphi)$$

where X^n shall not be free in φ.

(7) The property of ***being countable*** can also be formulated within second order logic by just expressing: A set is countable iff there is a linear ordering relation on it such that every element has only finitely many predecessors.

(8) Even the ***continuum hypothesis***, CH, can be formulated in second order logic[4]. The formula φ_{CH} says: If the domain is of the same cardinality as $\mathbb{R}$, then every subset of the domain is either countable (finite or infinite) or else of the same cardinality as the whole domain. Thus, φ_{CH} is valid iff CH holds.

(9) Also the ***generalized continuum hypothesis***, GCH, can be expressed in second order logic; that is, the formula

$$\forall XY(\textbf{\textit{inf}}(X) \wedge Y \sim \mathcal{P}X \rightarrow \forall Z(Z \subseteq Y \rightarrow Z \preceq X \vee Z \sim Y))$$

can be written completely in second order, as we shall see in section 4.8 below. This formula says: "Every subset **Z** of a set **Y** ($\mathbf{Z} \subseteq \mathbf{Y}$) that is equipollent to the power set of an infinite set **X** ($\mathbf{Y} \sim \mathcal{P}\mathbf{X}$) is either equipollent to that power set ($\mathbf{Z} \sim \mathbf{Y}$) or of equal or less power than the infinite set ($\mathbf{Z} \preceq \mathbf{X}$)."

Again, the formula φ_{GCH} is valid iff GCH holds. In fact, it has been known from early times that when using second order logic, the border with set theory has been

historical references.

[4]See section 4.8.

trespassed over. To set up the set of validities we need to specify which set theory will be used in the metalanguage in much more detail than in first order logic. Church, Henkin, Kreisel and Quine were aware of the situation long ago. Nowadays, one can find comments on it in almost any textbook[5].

We do not have to go into mathematics to find examples of thoughts needing second order logic to be expressed. Here are some colloquial examples:

(a) "Hay gente para todo" (there are all kinds of people). This can be formulated as:

$$\forall X \exists y\, Xy$$

(b) "There is at least one characteristic shared by all authoritarian regimes - either leftist or rightist." This can be formulated as:

$$\exists X \forall z (Az \wedge (Lz \vee Rz) \rightarrow Xz)$$

(c) "There are certain women who are able to love different men who don't share any quality." We can select the formalization:

$$\exists x (Wx \wedge \exists z \exists y (Mz \wedge My \wedge z \neq y \wedge Lxz \wedge Lxy \wedge \neg \exists X (Xz \wedge Xy)))$$

The problem is that most of them are trivially true or obviously false because the intended meaning is a bit more subtle.

(d) When we choose: "Mathematicians and philosophers share at least one quality", and we formalize it as

$$\exists Z \forall xy (Mx \wedge Py \rightarrow (Zx \wedge Zy))$$

we should not be satisfied either, since it is trivially true. (Think of the quality of being either a mathematician or a philosopher.)

1.3. Model-theoretic counterparts of expressiveness.

As a by-product of the expressive power of second order logic with standard semantics we obtain the following model-theoretic counterparts:

(1) The Peano axioms are categorical: any two second order models of the Peano axioms are isomorphic.

[5]Ebbinghaus, Flum & Thomas [1984], page 135 or van Benthem & Doets [1983], page 275.

(The proof of this was already in Dedekind. Chapter III will be devoted to this subject.)

(2) Second order logic is not a compact logic, that is: the compactness theorem fails.
(This result is a direct consequence of finiteness being expressable in the language. Think of the infinite set of formulas $\{\varphi_n / n \geq 2\}$, saying that there are at least n elements in the universe, and the formula expressing that the universe is finite. A detailed proof of the non-compactness is in 4.9.1 of this chapter.)

(3) The Löwenheim-Skolem theorem also fails.
(This result follows from the fact that being uncountable is expressible in the language: the formula expressing that the universe is uncountable has no countable model, as required for the Löwenheim-Skolem theorem.)

1.4. Incompleteness.

Therefore, in second order logic with standard semantics we will never find a strongly complete deductive calculus (*i.e.*, satisfying: if $\Gamma \models \varphi$, then $\Gamma \vdash \varphi$). The reason is that compactness, which could be proven from strong completeness, fails. We know even more: the set of validities is so unmanageable that we will never get a complete calculus, not even in the weak sense (*i.e.*, satisfying: if $\models \varphi$, then $\vdash \varphi$). This result follows from Gödel's incompleteness theorem together with item (1) above[6]. (In Chapter III we will sketch this incompleteness proof.)

As was pointed out by Németi and others, following ideas of Sain[7], we don't need the Gödel theorem to realize that a complete calculus can never be obtained. The observation was made with formulas such as CH in mind, formulas whose validity is based upon the background set theory we choose to have. (In Chapter II we will sketch this incompleteness proof.) Roughly posed, how can we define a calculus to generate as theorems the formulas in the unstable set of validities[8]?

[6]See van Heijenoort ed [1967], page 592 for the original proof. See Ebbinghaus, Flum & Thomas [1984], page 162 for a proof of the incompleteness of second order logic based on Trahtenbrot's theorem, which says that the set of sentences valid in all finite structures is not enumerable.

[7]Sain [1979] has important applications to computer science logics, philosophical logic and theory of semantics of natural languages. These applications also appear in Pasztor [1986], [1988] and Sain [1987]. See as well Barwise & Feferman eds [1985], page 600.

[8]In our Occidental tradition this has been maintained ever since the Heraclitean philosophers: "It is impossible to say anything true about things which change".

One might think that, perhaps, adding CH to the axioms of our background set theory will fix the situation. But, from Gödel's incompleteness, we know that this is not the case. It is not possible to give explicitly a complete axiom system for set theory; that is, a set of axioms such that every formula φ of the language of set theory or its negation $\neg\varphi$ is provable from the set of axioms. In fact, there is an inexhaustible supply of independent formulas like CH. However, even knowing that we can never achieve completeness, no one would stop us from defining a sound calculus. For instance, we can define a calculus just as an extension of the first order one, where the rules dealing with quantifiers also cover the relational quantification. Or we can extend the calculus a bit further by adding the comprehension schema to the calculus mentioned. The latter is the one commonly accepted as second order calculus. After that, we may or may not decide to add the axiom of choice or the axiom of extensionality or any other axiom we feel necessary. Any of these calculi is incomplete in the class of standard structures where the notion of subset is taken from the metalanguage (set theory). But if we leave open to interpretation in the structure what sets and relations are - ***i.e.***, if we accept non-standard structures - the situation changes.

This is exactly what Henkin did when in 1949 he proved the completeness theorem for type theory. In Chapter IV we will introduce the general structures invented by Henkin and will experience the dramatic changes they operate in second order logic. The changes are of such a nature that for many people you are no longer in the premises of second order logic. In particular, second order logic with general semantics is quite a manageable logic - since it is compact, strongly complete and enjoys the Löwenheim-Skolem property -, but you pay for it with the loss of a great deal of the expressive power.

2.- SECOND ORDER GRAMMAR.

A specific second order language is defined by giving its alphabet and the rules for its calculus of formulas. The alphabet contains enough logical symbols, quantifiable variables of several types and a possibly empty set of operation constants. Between two second order languages the differences always lay in the alphabet and they may affect any of these sets, but while some of these differences can be considered just minor ones (for example, when they only affect the operation constants), others can have greater relevance (for instance, when they affect the quantifiable relation variables by restricting them to unary relation variables). The set of logical symbols of a second order language always contains enough connectives and

quantifiers, but it might or might not include equality. As in first order logic, we say that the set of connectives is complete when all the truth functions can be defined in terms of connectives in the set and similarly for the quantifiers. The language is second order in so far as it contains quantifiable individual and relation variables.

Restricted SOL.

Some second order languages only have a finite number of kinds of relational variables, plus the individual variables which all of them have. For instance, monadic second order language[9] only contains individual and unary relation variables, both kinds being quantifiable. Binary second order language contains individuals, unary and binary relation variables. In general, n-ary second order language ($n \geq 1$) contains individual and i-ary relation variables for $1 \leq i \leq n$, where the number n represents the greatest degree of admissible quantification.

SOL.

Nevertheless, what is usually presented as second order language contains n-ary relation variables of any degree n (for $n \geq 1$, any positive integer). That is, it contains individual variables, unary relation variables, binary relation variables, etc. Unless otherwise explicitly stated, the second order languages used in this book, which belong to the class we are naming **SOL**, allow quantification for n-ary relation variables for any $n \geq 1$. These languages contain a first order basis, **FOL**, upon which we build the new second order features. **SOL** includes extended equality, for both individual and relation symbols. Equality is added as a logical symbol because, as we take it as primitive, it will have its genuine and fixed meaning; ***i.e.***, independent of the standard/non-standard semantics issue. (In Chapter IV we discuss it extensively.) In general, second order language does not have equality for relations, but we will have it. There are several good reasons for this choice, to be discussed in section 4.8 of this chapter.

Extended SOL.

Sometimes, we also have function variables that can be quantified. As before, we might allow function quantification up to a certain n, or for any n. As long as we have relation variables, having function variables or not is only a matter of convenience. This variation is inessential because we can always rewrite the formula using only relation variables.

[9]Monadic second order logic has very special properties, as can be seen in Gurevich [1985].

λ-SOL.
On the other hand, some second order languages contain the abstractor λ[10]. λ-abstraction will allow us to built predicates from formulas. The use of the λ-abstractor, while very convenient, is not essential since it adds no expressive power to **SOL** as presented here. We have decided to include it because it will give a preliminary glimpse of a typed λ-calculus, to be presented later on. λ-abstraction could also be used to build functions from terms, but we are not using it in this sense.

Equality-free SOL.
In the literature of standard second order logic the equality sign for individuals is defined by Leibniz's indiscernibility principle. Therefore, there is no real need of having it as primitive and no difference between **SOL** and **Equality-free SOL.** (This is no longer true when we shift to non-standard semantics and so, to make things easier, we have decided to include primitive equality even in this standard chapter.)

The set of operation constants.
Besides variables and logical symbols, in each second order language there is a set of relation and/or function constants. We call them operation constants and they are in a set **OPER.CONS** of our choice. Every operation constant in **OPER.CONS** must be different from the rest of the symbols in the language and none is a string of other operation symbols. In the classical presentation of second order language the symbols in **OPER.CONS** are all of them first order; that is, they are symbols for functions and relations among individuals obtained from **FOL**. Following a quite standard procedure, individual constants are identified with zero-ary function constants and propositional symbols, if any, are zero-ary relation symbols. Nowadays it is also common to include symbols for functions and relations among relations or for functions or relations between individuals and relations, but the essential feature of **SOL** is the quantification of relations. In **SOL** the only proper second order relation symbols we are having are the logical symbols of equality for relations.

Pure SOL.
We can also have a second order language with no operation constants; *i.e.*, **OPER.CONS** $= \emptyset$. In this language we have individual and relation variables[11]. This language is the natural one to

[10]The lambda abstractor was first introduced by Alonzo Church, see Church [1940] and [1941].

[11]Denyer [1992] has proved that second order logic without individual quantification, which he terms "Pure second order logic", is decidable. Do not get confused, **Pure SOL** has individual variables that can be quantified.

express the properties of size of the domain.

2.1. Definitions (signature and alphabet).

To specify a particular second order language we will give its signature. From the signature we learn the kinds of quantifiable variables and we also learn the relation and function constants we are having and their types. The only thing we specify separately is whether we are having λ-abstraction or not.

A signature Σ is a pair $\langle$**VAR**, **FUNC**$\rangle$ where **VAR** is the set containing all the kinds of quantifiable variables (in higher order logic the kinds are types, in many-sorted logic they are sorts) and **FUNC** is a function whose domain is the set **OPER.CONS** of operation constants of the language and it gives types as values; *i.e.*, finite sequences of members of **VAR**.

In 2.1.1 we will present the signature of a second order language in a very general case; *i.e.*, it might have function variables as well as relation variables and all of them are quantified. Moreover, if we have functions of degree n, then we have functions of degree 1,..., n-1 as well; so we can continually go from unary variables to n-ary variables.

2.1.1. Signature of any second-order language.

By a second order signature Σ we mean an ordered pair $\Sigma = \langle$**VAR**, **FUNC**$\rangle$ where:

(i) **VAR** is a set such that: (1) $1 \in$ **VAR**, $\langle 0,1\rangle \in$ **VAR** and (2) whenever $\alpha \in$ **VAR** then $\alpha = \langle 0,1,\overset{n}{\ldots},1\rangle$ with $n \geq 1$ or $\alpha = \langle 1,\overset{n+1}{\ldots},1\rangle$.
Besides that, whenever $\alpha \in$ **VAR** and $\alpha = \langle 0,1,\overset{n}{\ldots},1\rangle$ with $n>1$ then also $\langle 0,1,\overset{n-1}{\ldots},1\rangle \in$ **VAR**. And whenever $\beta \in$ **VAR** and $\beta = \langle 1,\overset{n}{\ldots},1\rangle$ with $n>1$, then also $\langle 1,\overset{n-1}{\ldots},1\rangle \in$ **VAR**.

(ii) **FUNC** = **FUNC**(Σ) is a function whose values are of these forms: $\langle 0,1,\overset{n}{\ldots},1\rangle$ with $n \geq 1$ or $\langle 1,\overset{n+1}{\ldots},1\rangle$ with $n \geq 0$. We are using **OPER.CONS** as domain of **FUNC** and call its elements operation constants. ▨

Explanation.

The set **VAR** contains the kinds of quantifiable variables, while in **FUNC** we obtain the type of each operation constant. 1 is the type of individuals, $\langle 0,1,\overset{n}{\ldots},1\rangle$ is the type of n-ary

relations and $\langle 1,\overset{n+1}{\ldots},1\rangle$ is the type of n-ary functions. (So zero-ary functions are safely identified with individuals.) Since in second order logic our variables are for sets and relations, the kinds of variables are also types. In the classical presentation of second order logic the function and relation constants are always among individuals and so the types of relation constants are types of certain variables as well.

2.1.2. Signature of the classes of languages SOL and λ-SOL.

(i) **VAR** = $\{1, \langle 0,1\rangle, \langle 0,1,1\rangle, \langle 0,1,1,1\rangle,\ldots\}$

(ii) **FUNC** = **FUNC**(Σ) is a function defined as in 2.1.1 ▨

Remark.

The set **VAR** of any language in **SOL** is formed by two disjoint sets corresponding to individual and relation types. Since we want to keep the formulas easy to read, we are not using the types as superscripts, instead we will use the more conventional treatment of first order logic: we will use just a number indicating the arity of the variable or relation constant.

2.1.3. Alphabet of the λ-SOL language λ-L_2.

The alphabet of a second order language of signature Σ contains all the operation constants in **OPER.CONS**, logical symbols and an infinite number of variables for each type $\alpha \in$ **VAR**. Besides that, it may contain the symbol λ.

In particular, our λ-language (λ-L_2) will have:

(1) Connectives: $\neg, \vee, \wedge, \rightarrow, \longmapsto$.

(2) Quantifiers: $\forall, \exists$.

(3) Abstractor: λ.

(4) Parentheses:),(.

(5) Equality symbols: $E, E_1, E_2,\ldots$ (for individuals and relations).

They have types: $\langle 0,1,1\rangle$, $\langle 0,\langle 0,1\rangle,\langle 0,1\rangle\rangle,\ldots,\langle 0,\langle 0,1,\overset{n}{\ldots},1\rangle,\langle 0,1,\overset{n}{\ldots},1\rangle\rangle$, etc.

(6) Falsity: $\perp$ (its type is 0).

(7) A set $\mathcal{V}$ of individual variables: $x, y, z, x_1, x_2, x_n,\ldots$ (their type is 1)
A set $\mathcal{V}_1$ of unary relation variables: $X^1, Y^1, Z^1, X^1_1, X^1_2, X^1_3,\ldots$ (their type is $\langle 0,1\rangle$.)
A set $\mathcal{V}_2$ of binary relation variables: $X^2, Y^2, Z^2, X^2_1, X^2_2, X^2_3,\ldots$ (their type is $\langle 0,1,1\rangle$)
and so on.
We will consider a countable infinite set, **OPER.CONS**, including:

(8) Zero-ary function constants: $a, b, c, c_1, c_2, c_3,\ldots$ (their type is 1, as the type of