

Lambda-Conversion

We have seen in § 0C that the theory of lambda-conversion occupies a position intermediate between ordinary logic and the theory of combinators, and that it is convenient to begin the study of combinatory logic with the former. Accordingly we shall devote the present chapter to it. There are several forms of lambda-conversion which we shall treat currently; hence we shall speak somewhat indifferently of a calculus or of calculuses of lambda-conversion.

We shall begin (§ A) with an intuitive discussion of variables in mathematics (which will amplify somewhat the discussion in § 0A and § 2C). Then we shall introduce the idea of functional abstraction (§ B) and the rules for it; after that the operation of application. Proceeding in the direction from the intuitive to the formal, we arrive at the calculus formulated as a formal system in §§ C-D. A full formulation of the calculus as formal system will include, in § E, a rigorous theory of the prefixes of substitution, such that substitution is always defined; since any sort of bound variables can be defined in terms of functional abstraction, this will simplify the statement of substitution rules in several theories.

Examples of the technique of λ -conversion will be found in Chapter 5. We leave for Chapter 4 the formulation and proof of the consistency theorem of Church and Rosser.

A. VARIABLES AND FUNCTIONS IN MATHEMATICS

In this section we discuss intuitively the role of formal variables in mathematics. The discussion may be compared with the introductory portions of various papers by Church, with the treatment of the same topic in Rosser [LMth], and with the writings of Menger.

1. Elementary examples

From the discussions of Chapter 2 and the Introduction it is evident that formal variables are employed in logical systems to indicate substitution processes. Combinatory logic is thus concerned with the analysis of formal variables and their eventual elimination.

Since formal variables correspond to the intuitive uses of variables in ordinary mathematics, it will clarify the statements in § 2C somewhat if we consider some uses of variables of that sort. Let us take as examples the statements:

$$(1a) \quad (x + 1)^2 = x^2 + 2x + 1,$$

(1b) x^2 is a function of x ,

$$(1c) \quad \frac{d}{dx} x^2 = 2x,$$

$$(1d) \quad \int_0^9 x^2 dx = 9.$$

In these statements it is ordinarily said that x is a variable. But we cannot say, except perhaps in the first case, that the symbol ' x ' is an intuitive variable. On the other hand, these statements do not say anything either about the symbol ' x ' or about the object x , whatever it may be. Variables are thus means of enunciating theorems about some other things. A little thought will show that these other things are functions. Indeed, each of the propositions (1) expresses, when interpreted, a property of a special function, the square.

2. Need for a functional notation

Curiously a systematic notation for functions is lacking in ordinary mathematics. The usual notation ' $f(x)$ ' does not distinguish between the function itself and the value of this function for an undetermined value of the argument. This defect is especially striking in theories which employ functional operations, i.e., functions which admit other functions as arguments. For special operations such as differentiation and integration we have notations having a unique sense, but not for operations in general.

For example, in theories using abstract operators P , Q , application of the operator P to the function $f(x)$ is often expressed by ' $P[f(x)]$ '. What then does ' $P[f(x + 1)]$ ' mean? Does $g(x) = f(x + 1)$ have to be formulated first, and then $P[g(x)]$; or is $h(x) = P[f(x)]$ to be formed first and then $h(x + 1)$? Although both these procedures lead to the same result for several important operators, we do not have the same result in the following case. Let

$$(2) \quad P[f(x)] = \begin{cases} \frac{f(x) - f(0)}{x} & \text{for } x \neq 0, \\ f'(0) & \text{for } x = 0. \end{cases}$$

Then, if $f(x) = x^2$:

$$\begin{aligned} P[g(x)] &= P[x^2 + 2x + 1] = x + 2, \\ h(x) &= P[f(x)] = x, \\ h(x + 1) &= x + 1 \neq P[g(x)].^1 \end{aligned}$$

3. The λ -notation

To have a systematic notation for the function in itself we must remember that a function is a law of correspondence, i.e., a class of ordered couples, and that to indicate the function we must indicate both elements of each couple. If we abbreviate by M an expression containing ' x ' which indicates the value of a function when the argument has the value x , we write ²

$$(3) \quad \lambda x(M)$$

to designate the function in itself. Thus $\lambda x(x^2)$ is the square function, i.e. the function having x^2 for value if x is the value of the argument. Obvious abbreviations may be used: ' $\lambda x x^2$ ' may be written instead of ' $\lambda x(x^2)$ ' as there is no ambiguity. Or we may use a dot instead of the round brackets, the scope of the dot extending to the end of the functional expression; hence ' $\lambda x.x + 2$ ' is equivalent to ' $\lambda x(x + 2)$ '.³

The fact that the statements (1) concern the square function, $\lambda x.x^2$, can now be made explicit. Suppose we denote differentiation by ' \mathcal{D} ' and integration by ' \mathcal{J} ', then we can define

$$\begin{aligned} \frac{d}{dx} M &\equiv D(\lambda x.M), \\ \int_a^b M dx &\equiv J(a, b, \lambda x.M). \end{aligned}$$

Then the statements (1) become:

$$\begin{aligned} (4a) \quad & \lambda x(x + 1)^2 = \lambda x(x^2 + 2x + 1), \\ (4b) \quad & \lambda x.x^2 \text{ is a function,} \\ (4c) \quad & D(\lambda x.x^2) = \lambda x.2x, \\ (4d) \quad & J(0, 3, \lambda x.x^2) = 9. \end{aligned}$$

1. An error resulting from this ambiguity actually occurs in the published literature. See [ADOJ].

2. Other notations are in use, e.g., ' λx ' \mathcal{M} ' used later, or ' $\lambda x M$ ' [PM], ' $x \rightarrow M$ ' [H. T. Davis], ' $[M]_x$ ' [ADOJ].

3. For the rules governing the use of dots see § 2B5a.

As for the example in connection with (2), if we let E be an operator such that

$$E(\lambda x.f(x)) = \lambda x.f(x + 1),$$

then the first of the two interpretations of ' $P[f(x + 1)]$ ' is $P[E(\lambda x.f(x))]$, the second is $E[P(\lambda x.f(x))]$. If we use ' \mathcal{J} ' for $\lambda x.f(x)$, these are $P(E\mathcal{J})$ and $E(P\mathcal{J})$ respectively.

B. FUNCTIONAL ABSTRACTION

1. The idea of functional abstraction

The examples of § A allow a certain generalization. The idea of function, as a law of correspondence, can be extended to quite arbitrary objects, and in particular to the obs of any system. A certain generalization of this sort is implied in the treatment of D , J , P and E as functions;⁴ they are functions whose arguments are other functions; except for J , their values are functions also. Such a situation can occur in any system in which functions are obs (even if they belong to a separate category).

As stated in § A3, we use ' $\lambda x M$ ' to denote a function in itself. The formation of $\lambda x M$ from x and M will be called *functional abstraction*. It is evidently an operation in the intuitive sense. If it is actually an operation (primitive or derived) of the system, so that $\lambda x M$ is an ob whenever x is a variable and M is an ob, then the system will be combinatorially complete in the sense of the Introduction; indeed, we can regard § A (3) as a more exact statement of what we mean by "any function which we can define intuitively by means of a variable".

2. Functions of several arguments

For functions of several arguments we might similarly define an n -fold functional abstraction

$$(1) \quad \lambda^n x_1 x_2 \dots x_n.M$$

with the intuitive sense of "the function whose value is M when the arguments are x_1, x_2, \dots, x_n ". However, a little thought will show that this operation can be defined by iteration of the operation of § A (3).⁵ Take, for instance, addition, whose value, for the arguments x and y , is $x + y$. If we regard x as a fixed value, the function $\lambda y(x + y)$ will represent, according to our conventions, the operation of adding the argument to x . If we use the generalized concept of a function, this can be regarded as itself the value of a function of x , viz., the correspondence which associates to each x the operation of adding the argument to x . This correspondence,

4. This is done in Church [CLC] Chapter I. Cf. also the papers by Menger.

5. This is the principle of the Schönfinkel function concept.

which according to our conventions is $\lambda x(\lambda y(x + y))$ or $\lambda x : \lambda y. x + y$, is then the intuitive equivalent of addition; hence we can adopt the definition:

$$\lambda^2 xy. x + y : \equiv : \lambda x : \lambda y. x + y.$$

In the general case of (1) we adopt the recursive ⁶ definition

$$(2) \quad \begin{aligned} \lambda^1 x. M &: \equiv : \lambda x. M, \\ \lambda^{2n+1} x_1 \dots x_n y. M &: \equiv : \lambda^{2n} x_1 \dots x_n (\lambda y. M). \end{aligned}$$

The exponent of the λ is usually omitted.

Thus we can express functions of any number of arguments by means of simple functional abstraction; and in a combinatorially complete system functions of any number of variables will be obs.

3. λ -applicative systems

We have noted that we can use the reduction of § IE3 to define all ordinary operators in terms of special obs and application. If this is carried out systematically we shall be left with a system in which the only operations are application and functional abstraction. ⁷ We may call such a system *λ -applicative*. ⁸ The preceding argument shows that such a system can be adequate for any logical purpose. It has a special importance for combinatory logic.

In view of this importance, and also in view of the fact that the notation of application differs from that current in ordinary mathematics, we shall discuss the meaning attached to the notation a little further.

In mathematics the application of a unary function is written in various ways: we write ' x^2 ', ' $\log x$ ', ' x' ', ' $f(x)$ ', ' $|x|$ ', ' e^x '. For binary functions we write ' $f(x, y)$ ' and use, in some cases, infixes, e.g. the '+' in ' $x + y$ '. Here application is taken as a unique binary operation, such that, if the first argument is a function f , and the second one is a value a , then the result of the operation is the value which f makes correspond to a . We indicate application by simple juxtaposition, with association to the left to avoid too many brackets. Thus, fab is $(fa)b$, $fabc$ is $((fa)b)c$ and so on. In this way values of functions of several variables are constructed, so to speak, progressively. Thus, if A is the addition function, Aa is the function of adding the argument to a , and $(Aa)b$, or Aab , is the sum of a and b .

We shall show by means of a table the intuitive meaning of certain

6. It is recursive as definition of a function from the natural numbers to obs. This is a natural generalization of the notions considered in § 2E2.

7. Of course we can use the method of § IE3 to eliminate functional abstraction. But with reference to the intended interpretation the transformation in the text preserves an element of meaning which the other procedure would not.

8. It is a special case of a quasi-applicative system as defined in § IE3.

functions applied to arguments; we write the technical notation in the left column and the usual mathematical notation (supplemented if necessary by the use of functional abstraction) in the right column.

1) If f is a unary function:

$$\begin{array}{l} f \\ fa \end{array} \qquad \begin{array}{l} \lambda x. f(x) \\ f(a) \end{array}$$

2) If f is a ternary function:

$$\begin{array}{l} f \\ fa \\ fabc \\ f(ga)b \\ f(hab) \end{array} \qquad \begin{array}{l} \lambda xyz. f(x, y, z) \\ \lambda xy. f(a, x, y) \\ f(a, b, c) \\ \lambda x. f(g(a), b, x) \\ \lambda xy. f(h(a, b), x, y) \end{array}$$

3) If we define:

$$\begin{array}{l} A \equiv \lambda xy. x + y \\ M \equiv \lambda xy. x \cdot y \\ N \equiv \lambda x. (-x) \end{array} \qquad \begin{array}{l} \text{(mathematical sum),} \\ \text{(mathematical product),} \\ \text{(mathematical negative).} \end{array}$$

Then we have the correspondences:

$$\begin{array}{l} A1 \\ Aab \\ A(Mab)(Mac) \\ N(Mac) \\ A(Mab)(N(Mac)) \end{array} \qquad \begin{array}{l} \lambda x. 1 + x \\ a + b \\ ab + ac \\ -ac \\ ab + (-ac). \end{array}$$

4. Relation to bound variables in general

In § 2C we have said that a system contains bound variables if and only if there is at least one proper operation with respect to which the variables play a special role. Let us call such an operation a *binding operation*; other operations will be called for the moment ordinary operations. Any binding operation will then have a certain number of arguments which are restricted to be variables, and a number which are not so restricted; let us call these the binding and the ordinary arguments respectively. Then functional abstraction has one binding argument and one ordinary argument.

We shall now show that any binding operation can in principle be defined in terms of functional abstraction and an ordinary operation. (This is to be understood in the sense that a primitive binding operation, other than functional abstraction, can be replaced by one so defined.) For let f be a primitive binding operation with m binding arguments and n ordinary arguments. When the binding arguments are x_1, \dots, x_m and the ordinary arguments are M_1, \dots, M_n , let the corresponding value be

(3) $f(x_1, \dots, x_m; M_1, \dots, M_n).$

Let $M_k^* \equiv \lambda x_1 \dots x_m. M_k$

$k = 1, 2, \dots, n,$

and let F be a new ordinary operation of n arguments. Then if we replace (3) throughout by

(4) $F(M_1^*, M_2^*, \dots, M_n^*),$

we shall have eliminated f as a primitive idea.^{8a} It follows that the *theory of functional abstraction is tantamount to the theory of bound variables in general*. Further, the essential purpose of bound variables is to enable us to use functions as arguments.

The definitions of $(d/dx)M$ and the integral above are special cases of this reduction. Other special cases may be found in logistic systems. In many such systems one uses notations, such as those given at the left of the following table, to designate notions whose interpretations are stated briefly at the right (X and Y are values of functions for the argument x):⁹

$(x)X$ or $(Vx)X$	X for all x
$(\exists x)X$	X for some x
$X \supset_x Y$	Y for all x such that X
$(\theta x)X$	the x such that X
$\{x \ni X\}$	the (class of) x 's such that X
$\{X \ni_x Y\}$	the X such that Y . ¹⁰

It follows from the preceding paragraph that if one introduced operators $\Pi, \Sigma, \mathcal{E}, \mathcal{A}$, one could define the first five of them thus:

$$\begin{aligned} (\pi x) X &\equiv \Pi (\lambda x. X), \\ (\exists x) X &\equiv \Sigma (\lambda x. X), \\ X \supset_x Y &\equiv \mathcal{E} (\lambda x. X) (\lambda x. Y), \end{aligned}$$

8a. (Added in proof). If a variable x_i is omitted from the prefix (§ C3a) of M_k^* , we can take care of the case where x_i is to be free in its occurrences in M_k , for example,

$$\int_{M_i}^{M_i} M_i dx_i \equiv J (M_i, M_i, \lambda x_i. M_i).$$

Still more general situations are conceivable, but of little use. (Cf. Church, A. [MTL₂], § 06.)

9. Note that the expressions on the left are nouns; the interpretations for the first three may be thought of as interpretations for $\vdash (x)X$, etc.

10. I.e., the x 's which are values of X for some x such that Y ; e.g., $\{x^2 \ni x \ 0 \leq x \leq 3\}$ is the set of squares of the integers from 0 to 3 inclusive, viz. the set consisting of 0, 1, 4, 9. This is of some importance for mathematics, and notations for it are beginning to be used, although none of them is universally accepted. Rosser [Mth] p. 291, writes ' $\{X | Y\}$ ', leaving the bound variable (or variables) to be determined by inspection. The notation suggested here is an adaptation of that used for the preceding notion (viz. that where X is x) by the Peano school. This was introduced in Peano [FMT], vol. II, no. 2, to replace another notation (ibid. vol. II, no. 1); cf. also Burali-Forti [LMT₂] § I 11, Rosenbloom [EMT] p. 98. Moore [TFG] uses a notation akin to that here proposed. (All these latter notations omit the braces.)

$$\begin{aligned} (\theta x) X &\equiv \mathcal{O} (\lambda x. X), \\ (x \ni X) &\equiv \mathcal{A} (\lambda x. X),^{11} \end{aligned}$$

where in the third case we have written the two arguments immediately to the right of the operator without parentheses. We can, of course, use the method of reduction of § IE3 and of § C below to define the operations in terms of obs and the operation of application; in that case the notation is consistent with that of § IE3.¹² The ob Π may then be interpreted as the attribute of universal generality, Σ that of satisfiability, \mathcal{E} as a relation of restricted generality, \mathcal{O} as the definite article, and \mathcal{A} as an abstraction operator for forming classes.

C. MORPHOLOGY OF A FORMAL λ -APPLICATIVE SYSTEM

1. Preliminary

For the purposes of combinatory logic we need a system which satisfies the conditions (a)-(d) of § 0B. A lambda-applicative system will evidently satisfy the conditions (b) and (d); the condition (c) requires an equality relation which will concern us later. The condition (a) requires that there be only one type of ob and one type of variable. This entails that any ob can be applied to any other ob, and that there be no formal distinction between functions and other sorts of obs. From the standpoint of interpretation there will therefore be many combinations to which no interpretation can be assigned, nevertheless these combinations are "significant" as obs of the system. Thus, if x, y, z are variables, $xy, x(yz), xz(yz)$ are obs.

For pure combinatory logic, furthermore—if we suppose that equality is formulated as a predicate—we shall have in principle no need of primitive obs which are not variables. For if such primitive constants were to occur they would be indeterminates, and therefore formal variables of a different sort. Since only a finite number of variables can actually be substituted for, or used as bound arguments, in any one context, there will always be variables left over which are in effect indeterminates. These can be interpreted as constants when the theory is extended. Thus the theory with no constants is compatible with the addition of constants later.¹³

11. The symbols ' \mathcal{T} ', ' \mathcal{Z} ', and ' \mathcal{E} ' have been used in approximately these senses by Curry (see, e.g. [RFR], [CFM]). The others are tentative, but it seems best to reserve them for this purpose for the time being. There seems no point in reserving a symbol for the sixth operation.

12. The symbols ' \mathcal{T} ', ' \mathcal{Z} ', ' \mathcal{E} ', ' \mathcal{O} ', ' \mathcal{A} ' then designate obs, with application indicated by juxtaposition.

13. Kleene [PCF], in studying a system of illative combinatory logic due to Church, admits explicitly the possibility of functional abstraction with respect to any "proper symbol" (i.e. atom). For a case where we need constants, see § D6.

2. Statement of the morphology

From these considerations we arrive at a morphology as follows:

- a. The atoms are an infinite sequence of variables

$$(5) \quad e_1, e_2, e_3, \dots$$

b. There is a binary operation, called *application*, indicated by juxtaposition, such that if X and Y are obs, then XY is an ob.

c. There is a binary operation designated by a prefixed ' γ ' as in § A(3), such that λxM is an ob whenever x is a variable and M is an ob.

d. There is a binary predicate, indicated by infixed ' $=$ ', such that $X = Y$ is an elementary statement wherever X and Y are obs. When we do not wish to postulate symmetry for this predicate we use the infix ' \geq ' (see § D3).

3. Auxiliary conventions

In connection with this morphology we shall use the following symbolic conventions.

a. The letters ' x ', ' y ', ' z ', and lower case italic letters generally, will be intuitive variables for unspecified primitive obs. Capitals, italic or German, will be used for unspecified obs.

We make use of the notation for iterated functional abstraction specified in § B2. Dots may be used according to § 2B5a.

Given an ob $\lambda x_1 \dots x_n M$, we call M the *base*; the operation $\lambda x_1 \dots x_n$ the *prefix*, and the variables x_1, \dots, x_n the *prefixed sequence*. These terms may also be applied to the corresponding expressions of the A-language. We admit association to the left for the application operation, as stated in § B3.

A *component* is here defined as a component, in the sense of § 2B2, with respect to application and prefixing of λx , where x is any variable; only a variable can be a first argument of λ .

b. We shall say that a variable x *occurs free* in an ob X if and only if x is a component of X with respect to application and functional abstraction with respect to variables distinct from x . By the argument of § 2D2 this is equivalent to the following recursive definition:

- (b1) x occurs free in x (but not in any $y \neq x$);
 (b2) x occurs free in XY if (and only if) x occurs free in either X or Y ;
 (b3) x occurs free in $\lambda y.X$ if (and only if) x is distinct from y and occurs free in X .

Here the words in parentheses are superfluous if we use the general understanding in regard to inductive definitions. Since the construction of X is a finite structure, "to occur free" is a decidable relation.

c. We shall say that x is *bound in* X if and only if x is the bound

3D] RULES OF THE CALCULUSES OF LAMBDA-CONVERSION

argument in (X or) a component of X . This again is equivalent to the recursive definition:

- (c1) no variable is bound in any primitive ob;
 (c2) x is bound in XY if (and only if) x is bound in X or Y ;
 (c3) x is bound in $\lambda y.X$ if (and only if) $x \equiv y$ or x is bound in X .

4. Modifications of the morphology

The morphology as stated is that for what we call *λK -conversion* below.

In the theories of Church λxM is not accepted as an ob¹⁴ unless x occurs free in M . This will be called *λI -conversion*. It excludes functions with fictitious arguments defined over all obs. Such functions can, however, be defined over limited ranges, and it is conceivable that this suffices for the purposes of ordinary mathematics.¹⁵

Another restriction, in which repeated occurrences of x are not allowed in M , is of some interest in the theory of functionality.¹⁷ But apparently one can do little with it. One needs multiple occurrences, at least over restricted ranges.

D. THEORETICAL RULES OF THE CALCULUSES OF LAMBDA-CONVERSION

1. Preliminaries

We now consider how to formulate equality in the system.

Evidently equality must satisfy the properties (θ), (σ), (τ), (μ), (ν) of § 2D. In order to have the replacement theorem we must have (since the system is quasi-applicative) also the rule

$$(5) \quad A = B \Rightarrow \lambda xA = \lambda xB.$$

2. Rules of β -conversion

These properties, however, do not exhaust the properties of equality if it is to signify equivalence in meaning. We shall need certain other principles, as follows:

a. From the standpoint of meaning it is evident that the bound variables are irrelevant—the correspondence is the same no matter what variable is used to indicate it. Thus we should like to have the axiom scheme,

$$\lambda x.X \equiv \lambda y.[y/x]X$$

where ' $[y/x]$ ' indicates the substitution of y for x .¹⁸

¹⁴ I.e., in his terminology, as a "well-formed formula". Cf. § S2.

¹⁵ Cf. § S3.

¹⁶ Cf. Fitch [SFL].

¹⁷ See § 10C.

¹⁸ This is defined by § 2C2. For revised definition see § E (cf. § b).

However, this scheme cannot be accepted without restriction. For if y should occur free in X , then an instance of y free in X ¹⁹ would become bound after substitution. For example, if X were xy the above equation would be

$$\lambda x.yx = \lambda y.yy,$$

where the two sides certainly do not have the same meaning. This phenomenon is called *confusion of bound variables*. As another example of it note that

$$\int_0^3 6xy \, dx = 27y;$$

if we change the bound variable on the left to y we should have

$$\int_0^3 6y^2 \, dy = 27y,$$

which is false.

To exclude this confusion of variables we define $[y/x]X$ as in §b and state the axiom scheme as follows:

(a) *If y is not free in X ,*

$$\lambda x.X = \lambda y.[y/x]X.$$

b. Next, if $\lambda x.M$ is the function whose unspecified value is M , then its application to any N must be the same as the result of substituting N for x in M , i.e.,

$$(\beta) \quad (\lambda x.M)N = [N/x]M.$$

Here again there is a possibility of confusion of variables to be guarded against. Thus suppose $M \equiv \lambda y.xy$ and $N \equiv y$. Then substitution of N for x in M without regard to the bound variables would lead to $\lambda y.yy$. But if we first transform M to $\lambda z.xz$ by (a) and then substitute, the result is $\lambda z.yz$.

Such a confusion may occur if there is a variable free in N which is bound in M . We could exclude this possibility by adding a restriction to (β), as we did in the case of (a).²⁰ However, the confusion resulted from the naive conception of substitution. If we change the definition of substitution in such a way that bound variables are shifted automatically so as to avoid confusion, then (β) may be accepted without restriction. We have so accepted it here; the revised definition of substitution is in §E.

c. Beside the confusion of variables, objection is often made to con-
19. An instance of y is free in X if every operation leading from it to X is either application or prefixing of λz with $z \neq y$; the instance is bound if prefixing by λy occurs.

20. This is done by Church. The scheme (β) corresponds to his Rules II and III.

structs such as $\lambda x(\lambda y(\lambda z.xz))$. Such obs are said to involve *collision of bound variables*. But as a matter of fact such obs can be interpreted in only one way; therefore the objection to them is practical rather than theoretical. Consequently our rules are so formulated that collisions of bound variables are not excluded theoretically.

We shall show later, by means of the synthetic theory of combinators, that the rules we have adopted are in fact acceptable—a fact which, assertions to the contrary notwithstanding, is not self-evident.

3. Generalization

We have stated the rules (a), (β), and (ξ) in terms of a relation of equality; but in reality we wish to state these rules as properties of an unspecified relation R , parallel to the properties (θ), (σ), (τ), (μ), (ν), postulated in §2D4. In this form the rules become:

(a) *If y does not occur free in X , $\lambda y[y/x]X R \lambda xX$.*

(β) $(\lambda xM)N R [N/x]M$.

(ξ) $X R Y \Rightarrow \lambda xX R \lambda xY$.

As in §2D we shall say that a special relation has one of the above properties if the property holds when R is specialized to that relation. We note that the rule (ξ) is a part of the requirement that R be monotone; while the rules (a) and (β) are axiom schemes.

The monotone equivalence generated by (a) and (β) will be called *β-convertibility*,²¹ that generated by (a) alone will be called *α-convertibility*. For either of these kinds we may have to make a further distinction between the two sorts of morphology considered in §C3; thus we shall have *βK-convertibility* or *βI-convertibility* according to which morphology is used. We symbolize any of these kinds of convertibility—and also other kinds to be considered later—by the usual sign of equality; the special infix 'cnv', employed by Church, may be used when there is possibility of confusion with other uses of the equality sign. The different sorts of convertibility may be distinguished by subscripts, but that is not usually necessary. A *conversion* is a transformation of an ob into one to which it is convertible.

Besides this equivalence we shall use also the monotone quasi-ordering. This will be called *reducibility*, and symbolized by the infix ' \geq '. A reduction is a transformation of an ob into one to which it is reducible. The converse transformation is called an *expansion*. There are, of course, different species of reduction as in the case of conversion. Since (a) is symmetric, *α-reducibility* and *α-convertibility* are the same.

21. Strictly, we should say "α-β-convertibility"; but there seems to be no interest in conversion based on (β) alone.

4. Rules of η -conversion

Beside the properties (α) and (β), a third property of a relation is that represented by the following rule:

(η) *If x is not free in M , then $\lambda x(Mx) R M$.*

This rule is intuitively acceptable for convertibility, because both sides of the relation represent the function whose value for the argument X is MX . On the other hand there are purposes for which the rule is not acceptable, because the left side is a function, while the right side may not be. ²² Church and his students have been especially interested in such interpretations.

The rule (ξ) of § 1 and the rule (η) taken together are equivalent to the following rule, which is a form of the principle of extensionality:

(ζ) *If x is not free in either M or N , then*

$$Mx = Nx \Rightarrow M = N. \text{ }^{23}$$

The rule (ζ) follows from (ξ) and (η) thus:

$$\begin{aligned} Mx = Nx &\Rightarrow \lambda x(Mx) = \lambda x(Nx) && \text{by } (\xi), \\ &\Rightarrow M = N && \text{by } (\eta). \end{aligned}$$

Conversely (η) and (ξ) follow from (ζ) together with (β) thus:

$$\begin{aligned} (\lambda x.Mx) x = Mx &&& \text{by } (\beta), \\ \lambda x.Mx = M &&& \text{by } (\zeta). \end{aligned}$$

This proves (η). To prove (ξ) we have

$$\begin{aligned} M = N &\Rightarrow (\lambda xM)x = (\lambda xN)x && \text{by } (\beta), \\ &\Rightarrow \lambda xM = \lambda xN. && \text{by } (\zeta). \end{aligned}$$

We call the lambda-conversion calculus with the rule (η) the $\beta\eta$ -calculus (there can be a $\beta\eta K$ -calculus and a $\beta\eta I$ -calculus).

5. Redexes

We introduce here a terminology which simplifies many of the succeeding formulations.

²² This is, of course, a matter of interpretation; because from the formal standpoint, as noted above, every ob is a function.

²³ In (ζ) it is not sufficient that $MX = NX$ for every ob X ; the x must be a variable which does not appear in M or N . Church [CLC] p. 64 has given an example of two obs M and N of his theory of δ -conversion (see § 6) such that

$$MXY = NXY$$

holds for any obs X and Y which are in normal form (defined below) and contain no free variables, but $M = N$ is not true. This example is discussed further in § 3.

We call an ob which can form the left side of an instance of one of the rules (β), (η), or (δ) (introduced later) a *redex* of the corresponding type; the right side of the same instance will then be called the *contractum* of the redex. A replacement of a redex by its contractum will then be called a *contraction* of the type of the rule. Thus a redex of type (β), or simply a β -redex, is an ob of the form $(\lambda x.M)N$, its contractum is $[N/x]M$; and a replacement of an instance of $(\lambda x.M)N$ by $[N/x]M$ is a β -contraction. Similarly an η -redex is an ob of the form $\lambda x.M'x$ where x is not free in M' ; its contractum is M' ; etc.

In a β -redex $(\lambda x.M)N$ we shall call M the *base* and N the *argument*. An ob will be said to be in *normal form* of a certain λ -calculus if it contains no redex appropriate to that calculus. Thus an ob is in β -normal form if it contains no β -redex; ²⁴ it is in $\beta\eta$ -normal form if it contains neither a β - nor an η -redex. Generally we can omit the type and say the ob is in normal form.

6. δ -conversion

We now formulate a third kind of reduction and a third type of lambda-calculus, viz., that of a calculus admitting a rule of reduction (or conversion) of the following kind (there being certain unspecified further restrictions on M and M^*):

(δ) *Let M be an ob which is not a β -redex and not of the form $\lambda x.N$, and let M contain no free variables nor any proper components which are redexes of any kind. Let M^* be an ob such that no constituent of M^* is a free variable and M^* is not a redex of the same kind as M . Then M is convertible into M^* .*

An ob M to which such a rule may be applied is called a δ -redex. Its contractum and a δ -reduction step are defined in analogy with § 5. It is clear that a δ -redex is of the form

$$aM_1M_2\dots M_n,$$

where a is a primitive constant and M_1, \dots, M_n are in normal form and contain no free variables. It is to be understood that an ob can be a δ -redex in at most one way. Note that primitive constants, which were excluded in § C1, become necessary when δ -conversion is admitted.

A λ -calculus which admits a form of the rule (δ) along with (α) and (β) will be called a $\beta\delta$ -calculus; if it admits also the rule (η), it will be called a $\beta\eta\delta$ -calculus, or simply a *full λ -calculus*. In Chapter 4 we shall prove the Church-Rosser theorem for an arbitrary full λ -calculus. A $\beta\delta$ -normal form is one which contains no redex admissible in a $\beta\delta$ -calculus.

Of course the rule (δ) is not, like the rules (β) and (η), a specific rule,

²⁴ Note that there is no α -redex; if we were to define one in analogy to the above it must not be taken into account in the definition of normal form.

but a type or form of rule. Therefore, it is appropriate that we speak of a rule (δ) but the rule (β), etc. Specific forms of rule (δ) will be indicated by subscripts.

One special form of $\beta\delta$ -calculus has been studied by Church, who calls it the $\lambda\delta$ -calculus. We shall call this special case the $\beta\delta_1$ -calculus.

In the $\beta\delta_1$ -calculus the rule (δ) is specialized as follows:

(δ_1) If M and N are in $\beta\delta_1$ -normal form and contain no free variables, then

$$\begin{aligned} \delta MN &= \lambda xy. x(xy) \text{ if } M \text{ is } \alpha\text{-convertible to } N, \\ \delta MN &= \lambda xy. xy \text{ if } M \text{ is not } \alpha\text{-convertible to } N. \end{aligned}$$

Here $\lambda xy. xy$ and $\lambda xy. x(xy)$ have been chosen since they are the numerical combinators corresponding respectively to 1 and 2. The $\beta\delta_1$ -calculus is intended as an approximate equivalent of the truth-value algebra with two values, any proposition being considered as equal either to 1 (false) or 2 (true). The equivalence is only approximate, inasmuch as the relation of convertibility is stricter than the ordinary equivalence between propositions.

E. SUBSTITUTION PREFIXES

The subject of this section is the definition of the substitution operation, whose closure is $[M/x]X$, in such a manner as to be complete (i.e. defined for all X, M, x , cf. § 2E2). This is a technical matter, and is presented here as a new result whose interest is rather special. The definition is stated in § 1; the theorems to be proved and some general comment in § 2. The rest of the section is devoted to the proof, except that some corollaries are stated in §§ 4 and 7.

1. The basic definition

DEFINITION 1. $[M/x]X$ is the ob X^* defined as follows:

Case 1. X is a variable.

- (a) If $X \equiv x$, then $X^* \equiv M$.
- (b) If $X \equiv y \neq x$, then $X^* \equiv X$.

Case 2. $X \equiv YZ$. Then $X^* \equiv Y^*Z^*$.

Case 3. $X \equiv \lambda yY$.

- (a) If $y \equiv x$, then $X^* \equiv X$.
- (b) If $y \neq x$, then $X^* \equiv \lambda z [M/x] [z/y]Y$,

where z is the variable defined as follows:

- (i) If x does not occur free in Y , or if y is not free in M , then $z \equiv y$;
- (ii) if x is free in Y and y is free in M , then z is the first variable in the list e_1, e_2, \dots such that $z \not\equiv x$ and z does not occur free in either M or Y .

This definition is proper in the sense of § 2E1. Hence it is legitimate to use the sign ' \equiv ' in connection with it (Theorem 2E1). Its completeness can be shown by induction on the rank, using Lemma 1 (see § 3).

2. Statement of the theorems

THEOREM 1. The ob $[M/x]X$ has the properties:

- (a) $[x/x]X \equiv X$.
- (b) If x does not occur free in X ,
 $[M/x]X \equiv X$.
- (c) If no variables which occur free in M or N are bound in X , and if $N' \equiv [M/x]N$; then

$$(1) \quad [M/x] [N/y] X \equiv [N'/y] [M/x] X$$

holds under either of the following conditions:

- (c1) y does not occur free in M and $x \neq y$,
- (c2) x does not occur free in X .

THEOREM 2. If equality is α -convertibility, then:

- (a) the relation
$$(2) \quad X = Y \rightarrow [M/x] X = [M/x] Y$$
 holds for all obs X, Y, M , and all variables x .
- (b) If X is an ob and v a given finite set of variables; then there exists an ob Y such that
$$X = Y,$$
 and such that no variable of v is bound in Y , whereas the bound variables of X not in v are bound in the corresponding occurrences in Y .
- (c) If N' , (c1), and (c2) are as in Theorem 1c; then

$$(3) \quad [M/x] [N/y] X = [N'/y] [M/x] X$$

holds under either of the conditions (c1), (c2) without further restriction.

THEOREM 3. If the reducibility relation of any of the theories of λ -conversion is indicated by ' \geq ', then

$$(4) \quad X \geq Y \rightarrow [M/x] X \geq [M/x] Y.$$

The proofs of these theorems by direct methods appear to be considerably more difficult than those obtained (§ 6D) by the synthetic theory of combinators. However, the latter course gives a result considerably weaker than that stated here, in that the identity of Theorem 1 and the α -convertibility of Theorem 2 are replaced by a weaker form of convertibility. Presumably the weaker result is sufficient for most purposes.

Thus the proof which follows has a rather technical significance: it shows that the result can be derived without recourse to the synthetic theory; it reveals some of the complications which the synthetic theory avoids, and thus illustrates the motivation for the latter; and it gives the stronger result just stated.

We shall prove Theorem 1, and some lemmas connected with it, by induction on the rank of X (see below), using the same classification into cases as in Definition 1. In considering Cases 2 and 3 it will be supposed the property being considered is true for all X of lower rank; this will be called the hypothesis of the induction. We shall abbreviate the left side of the equation to be proved, regarded as function of X , as X^* ; the right side, when it is distinct from X , as $X^\#$. In the lemmas X^* is $[M/x]X$, where M may be specialized.

3. Preliminary lemmas

We first define *rank*, as follows:

- (i) A variable is of rank 0.
- (ii) If X is of rank m and Y is of rank n , then XY is of rank $m + n + 1$.
- (iii) If X is of rank m , λxX is of rank $m + 1$.

LEMMA 1. ²⁵ *The rank of $[u/x]X$ is the same as that of X .*

Proof. In Case 1, X and X^* are both variables, and so they both have rank 0.

In Case 2, $X^* = Y^*Z^*$, and since Y^* , Z^* have the same rank as Y , Z respectively, X^* has the same rank as X .

In Case 3a, X and X^* have the same rank because they are identical. In Case 3b

$$X^* \equiv \lambda z [u/x] [z/y] Y;$$

by the hypothesis of induction, applied twice, $[u/x] [z/y] Y$ has the same rank as Y , and hence X^* has the same rank as X .

LEMMA 2. *A necessary and sufficient condition that u be free in $[M/x]X$ is that either (19) $u \not\equiv x$ and u occur free in X , or (20) x occur free in X and u occur free in M .*

Proof. In Case 1a, $X^* \equiv M$ and the variables free in X^* are precisely those free in M . This agrees with the lemma because x is free in X and no $u \not\equiv x$ is free in X . In Cases 1b and 3a $X^* \equiv X$, hence X and X^* have the same free variables. This, again, agrees with the lemma because x is not free in X .

In Case 2, $X^* \equiv Y^*Z^*$. The variables $u \not\equiv x$ which are free in X are

²⁵ The proof of Lemma 1 shows that $[u/x]X$ is always defined, although this point is not emphasized.

the same as the $v \not\equiv x$ free in Y together with the $w \not\equiv x$ free in Z . If x is free in X it is free in Y or Z and vice versa. These considerations show that the lemma is true.

In Case 3b we have

$$X \equiv \lambda y Y, \quad X^* \equiv \lambda z [M/x] [z/y] Y,$$

where z is the variable specified in Definition 1. Then Y is of lower rank than X . By the hypothesis of the induction the v 's free in $[z/y] Y$ are the same, except possibly $v \equiv z$, as those free in X , viz., those distinct from y and free in Y . By § C3b the u 's free in X^* are precisely those such that $u \not\equiv z$ and u is free in $[M/x] [z/y] Y$. By Lemma 1 $[z/y] Y$ has a lower rank than X . Hence, again by the hypothesis of the induction, the u 's free in $[M/x] [z/y] Y$ are precisely those such that either u is a v and $u \not\equiv x$, or x is a v and u is free in M . If u is free in X^* , then $u \not\equiv z$ (§ C3b) and u is free in $[M/x] [z/y] Y$; therefore, either $u \not\equiv x$ and u is a $v \not\equiv z$, hence free in X , or x is a v distinct from z (Definition 1), hence free in X , and u is free in M . This shows that the condition of the lemma is necessary. If $u \not\equiv x$ and u is free in X , then, since $u \equiv z$ is excluded by Definition 1, u is a v distinct from x and hence free in X^* ; condition (19) is therefore sufficient. Finally, if u is free in M and x is free in X , then x is a v , $u \not\equiv z$ (Definition 1), and hence u is free in X^* ; this proves the sufficiency of condition (20) and thus completes the proof of the lemma.

4. Proof of Theorem 1

This is a proof by induction on the rank of X ; we have to prove each property for Case 1, then for Cases 2 and 3.

Proof of (a). In Case 1a $X^* \equiv x \equiv X$. In Case 1b and 3a $X^* \equiv X$ by definition.

In Case 2, $X^* \equiv Y^*Z^* \equiv YZ \equiv X$.

In Case 3b, since y is not free in x , $z \equiv y$ and

$$X^* \equiv \lambda y [x/x] [y/y] Y.$$

By the hypothesis of induction applied twice

$$[x/x] [y/y] Y \equiv Y.$$

Hence $X^* \equiv X$, q.e.d.

COROLLARY 1.1. *If $y \not\equiv x$ and x is not free in Y or y is not free in M ,*

$$[M/x] \lambda y Y \equiv \lambda y [M/x] Y.$$

Proof of (b). The statement is trivial in Cases 1b and 3a. In Case 1a the premise is not satisfied.

In Case 2 we have, as before $X^* \equiv Y^*Z^* \equiv YZ \equiv X$.

In Case 3b we have $x \not\equiv y$ and hence x is not free in Y . By Corollary 1.1 and the hypothesis of the induction

$$X^* \equiv \lambda y Y^* \equiv \lambda y Y \equiv X.$$

Proof of (c₁). We dispose first of two trivial cases (in which the statement is true without the restriction on the bound variables of X). If y does not occur free in X , then it does not occur free in $[M/x]X$ either, and (1) follows by (b). If x does not occur in either X or N , we get a second trivial case which is symmetric to the first. We proceed with the induction supposing that these trivial cases are excluded.

In Case 1 the only nontrivial case is $X \equiv y$. Then

$$X^* \equiv [M/x]N \equiv N', \quad X\# \equiv N'.$$

In Case 2 we have by the hypothesis of induction $Y^* \equiv Y\#, Z^* \equiv Z\#$, hence

$$X^* \equiv Y^*Z^* \equiv Y\#Z\# \equiv X\#.$$

In Case 3 suppose $X \equiv \lambda z Y$. If $z \equiv y$ we have the first trivial case, if $z \equiv x$ the second.²⁶ If $z \not\equiv x$, $z \not\equiv y$, we can apply Corollary 1.1 and obtain

$$X^* \equiv \lambda z Y^*, \quad X\# \equiv \lambda z Y\#.$$

By the hypothesis of the induction $Y^* \equiv Y\#$, hence

$$X^* \equiv X\#, \quad \text{q.e.d.}$$

Proof of (c₂). By (b) the formula to be proved is

$$(5) \quad [M/x] [N/y] X \equiv [N'/y] X.$$

This statement is trivial if y does not occur in X or if x does not occur in N . We therefore exclude these cases.

In Case 1, $X \equiv y$, $X^* \equiv [M/x] N \equiv N' \equiv X\#$.

In Case 2, $X^* \equiv Y^*Z^* \equiv Y\#Z\# \equiv X\#$.

In Case 3, let $X \equiv \lambda z Y$. Then $z \not\equiv x$ since x , which occurs free in N , is not bound in X . Also $z \not\equiv y$ since y occurs free in X . Hence by Corollary 1.1

$$X^* \equiv \lambda z Y^* = \lambda z Y\# = X\#.$$

REMARK. The following shows that the property (c) is not generally true without the restrictions on the bound variables.

Let

$$X \equiv \lambda x.fxy,$$

$$M \equiv a \text{ (a variable not otherwise used),}$$

$$N \equiv x.$$

²⁶ Since then x is bound in X and hence not free in N .

Then

$$N' \equiv a,$$

$$X^* \equiv \lambda z.fza,$$

$$X\# \equiv \lambda x.fxa.$$

This is a counterexample for both of the alternatives in (c).

5. Proof of Theorem 2

In this the crucial property is (a). We cannot derive this from Theorem 2E7, because the hypotheses of that theorem are not satisfied in Case 3b. The proof of (a) is taken up in § 6 below; here we shall derive (b) and show that (c) is a consequence of (a).

a. If we have the properties (a) and (b), then the property (c) is an immediate consequence of the corresponding property under Theorem 1. For let Y be as in (b) with \mathfrak{b} specialized to the class of variables free in M, N , or X . Then

$$Y^* \equiv Y\#$$

by Theorem 1. The equations

$$X^* = Y^*, \quad X\# = Y\#,$$

follow by property (a) of this theorem from the equation of property (b). b. Before we take up the property (b) we need a lemma as follows:

LEMMA 3. *If M contains no free variables which are bound in X , then a necessary and sufficient condition that u be bound in $[M/x]X$ is that either u be bound in X or x be free in X and u bound in M .*

Proof. The lemma is trivial in the cases where x is not free in X and $X^* \equiv X$. This takes care of the Cases 1b and 3a. Likewise the induction is clear in Case 2.

In Case 1a, $X^* \equiv M$. Since x is free in X and no variables are bound in X , the lemma is verified.

In Case 3b, y is not free in M , $z \equiv y$ and hence (Corollary 1.1)

$$X^* \equiv \lambda y [M/x] Y.$$

If u is bound in X^* , then either $u \equiv y$, or else u is bound in $[M/x] Y$; in the latter case u is bound in Y or x is free in Y and u is bound in M . If $u \equiv y$, or u is bound in Y , then u is bound in X . If x is free in Y , then (since $x \not\equiv y$ by the conditions of Case 3b) x is free in X . Hence any u bound in X^* is either bound in X or is bound in M with x free in X . Conversely if u is bound in X , then either $u \equiv y$ or u is bound in Y , and

in either case u is bound in X^* . If x is free in Y and u is bound in M , then x is free in X and hence u is bound in X^* .

c. *Proof of (b)*. Consider a component of X of the form λxZ , where x is in \mathfrak{b} . Let y be a variable not in \mathfrak{b} and not occurring, bound or free, in Z . By Lemma 3, $[y/x]Z$ has the same bound variables as Z . Hence, if we replace λxZ by $\lambda y[y/x]Z$, we shall have replaced the bound variable x (in that instance) by y and have made no other changes in the bound variables.²⁷ If X is transformed into X_1 by this process then $X = X_1$ by (a) and the replacement theorem.²⁸ We can continue this way until all the bound variables in \mathfrak{b} have been eliminated.

6. Proof of (a)

a. We proceed by a modified deductive induction. We define

$$X^* \equiv [M/x]X, \quad Y^* \equiv [M/x]Y,$$

and we shall call $X^* = Y^*$ the $*$ -form of $X = Y$. Likewise we use the asterisk as applied to the name of a rule to indicate the modified rule obtained by replacing each premise and conclusion by its $*$ -form. Thus $(\mu)^*$ is the rule

$$X^* = Y^* \rightarrow (ZX)^* = (ZY)^*.$$

The typical procedure of a proof by deductive induction would be to show that each instance of $(\varrho)^*$ and $(a)^*$ is valid, and that $(\sigma)^*$, $(\tau)^*$, $(\mu)^*$, $(\eta)^*$, $(\xi)^*$ are valid derived rules. This is easy in the case of $(\varrho)^*$, $(\sigma)^*$, $(\tau)^*$, $(\mu)^*$, $(\eta)^*$. In the first three cases the rule is a specialization of the corresponding unstarred rule; thus $(\sigma)^*$, viz.

$$X^* = Y^* \rightarrow Y^* = X^*,$$

is a specialization of (σ) . In the case of (μ) and (η) we only need in addition to use Definition 1, Case 2; thus the $(\mu)^*$ just stated is equivalent to

$$X^* = Y^* \rightarrow Z^*X^* = Z^*Y^*,$$

which is a special case of (μ) .

In the other cases the procedure is modified. We observe that if $X = Y$ then X and Y have the same rank; this can be seen by a deductive induction and Lemma 1. They also have the same free variables and the same structure, differing only in their bound variables. We shall define the rank of the equation as their common rank, and shall also define the

²⁷ In particular we shall not have introduced any new bound variables other than y . Note that Z is of lower rank than X .

²⁸ This does not require (a). We are replacing the component Z in X , and no substitution is involved.

rank of an instance of (a) to be that of the premise. We also observe that the rules (σ) and (τ) do not change the rank, and the rules (μ) , (η) , (ξ) increase it. Accordingly we shall supplement the deductive induction by an induction on the rank. In proving instances of (a) where the premise is derived by (a) or (ξ) we shall assume that (a) is true for all cases of lower rank. We call this assumption the hypothesis of the induction. The effect of the whole argument is to reduce the proof of any special case of (a) to a finite number of cases which either correspond to preceding steps in the proof of $X = Y$ or are of lower rank.

We proceed to the analysis of the remaining cases, $(\xi)^*$ and $(a)^*$. In the former case we reduce the problem to $(a)^*$ for the same rank. In both cases we suppose the rank of $X = Y$ is m and assume (a) for ranks less than m .

b. *Analysis of $(\xi)^*$* . Suppose that

$$(6) \quad X \equiv \lambda yU, \quad Y \equiv \lambda yV,$$

and that $X = Y$ is derived from $U = V$ by (ξ) . If y is not free²⁹ in M , then by Corollary 1.1

$$X^* \equiv \lambda yU^*, \quad Y^* \equiv \lambda yV^*.$$

By the hypothesis of the induction $U^* = V^*$; hence $X^* = Y^*$ follows by (ξ) .

In the general case let z be a variable which is not free in U, V , or M .

Let

$$(7) \quad U_1 \equiv [z/\eta]U, \quad V_1 \equiv [z/\eta]V.$$

Then by the hypothesis of the induction $U_1 = V_1$. Since U_1 and V_1 are still of rank less than m , we can apply the hypothesis of the induction again and obtain $U_1^* = V_1^*$.

By the special case just treated we then have

$$(8) \quad [M/x]\lambda zU_1 = [M/x]\lambda zV_1.$$

On the other hand we have by (a) and (7)

$$X = \lambda zU_1, \quad Y = \lambda zV_1,$$

and hence, by (a)* for the rank m ,

$$X^* = [M/x]\lambda zU_1, \quad Y^* = [M/x]\lambda zV_1.$$

From this and (8) we have

$$X^* = Y^*.$$

This argument reduces $(\xi)^*$ to $(a)^*$ of the same rank.

²⁹ We assume $y \neq x$. If $y \equiv x$, the case is trivial.

c. *Proof of (a)*.* Suppose that

$$X \equiv \lambda y Z, \quad Y \equiv \lambda z [z/y] Z,$$

where z is not free in Z . We wish to prove that

$$(9) \quad X^* = Y^*.$$

If $y \equiv z$ this conclusion is trivial. Likewise, if x is not free in X , $X^* \equiv X$; since X and Y have the same free variables $Y^* \equiv Y$ and (9) is trivial. This includes the cases $x \equiv y$, $x \equiv z$. We therefore suppose that $x \not\equiv y$, $x \not\equiv z$, $y \not\equiv z$ and x is free in Z .

By Definition 1, Case 3b, we have

$$(10) \quad X^* \equiv \lambda u. [M/x] [u/y] Z,$$

where 1) if y is not free in M , $u \equiv y$, and 2) if y is free in M , $u \not\equiv x$, u is not free in M , and u is not free in Z . By the same definition

$$(11) \quad Y^* \equiv \lambda v. [M/x] [v/z] [z/y] Z,$$

where 1) if z is not free in M , $v \equiv z$, and 2) if z is free in M , $v \not\equiv x$, v is not free in M , and v is not free in $[z/y] Z$, and hence not free in Z unless $v \equiv y$. But if $v \equiv y$, then y is not free in M and $u \equiv y$, hence $v \equiv u$. From (10) we have, by (a),

$$X^* = \lambda v. [v/u] [M/x] [u/y] Z,$$

provided either $v \equiv u$ (Theorem 1a) or v is not free in $[M/x] [u/y] Z$. The v in (11) satisfies these conditions. Hence, by (5), a sufficient condition for (9) is

$$(12) \quad [v/u] [M/x] [u/y] Z = [M/x] [v/z] [z/y] Z.$$

Now let V be an ob, constructed according to (b), such that

$$(13) \quad Z = V,$$

and neither any of the variables u , v , z , nor any variable free in M is bound in V . Then, by Theorem 1c, we have

$$\begin{aligned} [v/u] [M/x] [u/y] V &\equiv [M/x] [v/u] [u/y] V \\ &\equiv [M/x] [v/y] V \quad (\text{Th. 1c}_2); \end{aligned}$$

whereas by Theorem 1c₂ alone

$$[M/x] [v/z] [z/y] V \equiv [M/x] [v/y] V.$$

Therefore

$$(14) \quad [v/u] [M/x] [u/y] V \equiv [M/x] [v/z] [z/y] V.$$

On the other hand, applying the hypothesis of the induction to (13), we have

$$[z/y] Z = [z/y] V.$$

Since both sides are still of rank less than m , we can proceed again and have

$$[v/z] [z/y] Z = [v/z] [z/y] V,$$

$$(15) \quad [M/x] [v/z] [z/y] Z = [M/x] [v/z] [z/y] V.$$

Again

$$(16) \quad [u/y] Z = [u/y] V.$$

If M is a variable we can proceed still further to

$$(17) \quad [v/u] [M/x] [u/y] Z = [v/u] [M/x] [u/y] V.$$

From (17), (14), and (15) we then have (12). This completes the proof of Theorem 2a for the special case where M is an atom. But with this special case we can pass from (16) to (17) in the general case. Theorem 2 is therefore proved.

7. Corollaries

COROLLARY 2.1. $[M/x] [N/x] X = [N'/x] X$,

where $N' \equiv [M/x] N$.

Proof. By Theorem 2c, if z is not free in X or M ,

$$\begin{aligned} [M/x] [N/x] X &= [M/x] [N/z] [z/x] X && (\text{Th. 2c}), \\ &= [N'/z] [M/x] [z/x] X && \\ &= [N'/x] X && (\text{Th. 2c}). \end{aligned}$$

It is well known that multiple substitution can be defined in terms of simple substitution. In fact, given X , x_1, \dots, x_n , M_1, \dots, M_n , if we let X^* be

$$[M_n/z_n] \dots [M_1/z_1] [z_n/x_n] \dots [z_1/x_1] X,$$

where z_1, \dots, z_n are variables chosen so that z_k is distinct from x_1, \dots, x_n , z_1, \dots, z_{k-1} and all variables free in X, M_1, \dots, M_n , then X^* can represent the result of substituting the M_k for the x_k in X simultaneously. In particular X^* becomes M_i when X becomes x_i ; it becomes X if X does not contain any of the x_i ; etc. These properties will not be considered further.

8. Proof of Theorem 3

We show simply that if one of the rules (β), (η), (θ) hold, the corresponding *-rule (cf. § 6) is derivable. We adopt the convention

$$X^* \equiv [M/x] X,$$

where M , x are fixed but X is arbitrary. To avoid confusion with (2) we change the bound variable in the rule from x to y ; and we replace ' M ', ' N ' in the statement of the rule respectively by ' X ', ' Y '. We denote the left side of the rule by ' L ', the right side by ' R '. We have to consider separately the cases $x \equiv y$ and $x \not\equiv y$; in the former case we allow x to stand in L and R .

For $(\beta)^*$ with $x \equiv y$ we have

$$L \equiv (\lambda x X) Y, \quad R \equiv [Y/x] X.$$

Then

$$R^* \equiv [M/x] [Y/x] X$$

$$= [Y^*/x] X \quad (\text{Cor. 2.1}).$$

$$L^* \equiv (\lambda x X) Y^* \quad (\text{Def. 1.2, 1.3a}),$$

$$\geq [Y^*/x] X \quad (\text{by } (\beta)),$$

$$= R^*$$

(as just shown).

Therefore,

$$L^* \geq R^*.$$

If $x \not\equiv y$, we have

$$L \equiv (\lambda y X) Y, \quad R \equiv [Y/y] X.$$

$$L^* \equiv (\lambda z [M/x] [z/y] X) Y^* \quad (\text{Def. 1}),$$

$$\geq [Y^*/z] [M/x] [z/y] X \quad (\text{by } (\beta)),$$

$$= [M/x] [Y/z] [z/y] X \quad (\text{Th. 2c}),$$

$$= [M/x] [Y/y] X \quad (\text{by } (5)),$$

$$\equiv R^*.$$

For $(\eta)^*$ with $x \equiv y$ we have

$$L^* \equiv L \geq R \equiv R^*.$$

On the other hand if $x \not\equiv y$,

$$L \equiv \lambda y. Xy, \quad R \equiv X.$$

$$L^* \equiv \lambda z. [M/x] [z/y] Xy \quad (\text{Def. 1}),$$

$$\equiv \lambda z. [M/x] Xz \quad (\text{Def. 1}),$$

$$\equiv \lambda z. X^*z \quad (\text{Def. 1}),$$

$$\geq X^* \equiv R^* \quad ((\eta)).$$

The case of $(\delta)^*$ is trivial, since the two sides contain no free variables.

S. SUPPLEMENTARY TOPICS

In this section all bibliographic citations are to papers by Church unless otherwise indicated. Also we regard all symbols of the system studied as being their own names. ³⁰

30. I.e. we use the "anonymous mode of speech" of Carnap [LSL]. There is no harm in this as long as the nominal symbols actually have no other meaning (cf. the remark at the end of § 1D6).

1. Historical comment

The idea of functional abstraction is a very old one in logic, and notations for it and related notions were introduced by both Frege and Peano. For this remote history see Church [Abs], Feys [PBF]; we shall not be concerned with it further.

The ideas of variable and function in mathematics have been the subject of study by Menger. His papers are listed in the Bibliography. The study is rather differently motivated than those considered here, and we have not attempted to take account of it. It appears to be quite independent.

The calculuses of lambda-conversion are the creation of Church. In this he was aided considerably by his students Rosser and Kleene. The current systematic treatment is [CLC], which contains references to all previous papers.

In the present chapter we treat only the fundamentals of the system; in fact, the consistency proof is postponed to Chapter 4, the equivalence with the theory of combinators to Chapter 6, and the combinatory arithmetic, which takes up more than half of the space in [CLC] and includes some of the most important results of the study, to the proposed second volume.

The calculus first appeared, in combination with certain illative notions, in [SPF]. There followed a series of papers by Church, Rosser, and Kleene, culminating in Kleene and Rosser [FEL], which proved that the system of [SPF] was inconsistent. This led to dropping of the illative concepts except for certain vestiges inherent in the notion δ and restrictions to obs in normal form. The resulting system was proved consistent in Church and Rosser [PCn] (see Chapter 4). A "system of logic" based on it was proposed (1934-5) in [RPx], [PFC]; this was written up systematically in Church's Princeton lecture notes [MLg] of 1936. In that year appeared the important papers [UPE] and [NEP] showing the undecidability of the λ -calculus and of the classical predicate calculus respectively. These belong to combinatory arithmetic, and [PFC] does also in large part; so they cannot be discussed here. The monograph [CLC], which appeared first in 1941, made an important change in notation and other improvements; but omitted treatment of the system proposed in [PFC].

2. Notation

As presented by Church the theory is syntactical. ³¹ One starts with primitive symbols consisting of variables, ³² parentheses, and λ . Any expression in the primitive symbols constitutes a "formula". Among the formulas a definite subclass, the "well-formed formulas" is singled out. These well-formed formulas represent the obs of the formal system that is presented in this chapter. We regard this as purely a notational difference.

As mentioned, Church used two different notations. In principle juxtaposition is used to indicate application, and a prefixed λ to denote the λ -operation, in both systems; but the conventions in regard to parentheses are different. In [CLC] these conventions are the same as in § B3. In the earlier papers there were three kinds of parentheses—braces, round parentheses, and square brackets. It was provided that in well-formed formulas the first argument of an application should be enclosed in braces, the second in parentheses, whereas the second argument of a λ -closure was enclosed in brackets. Abbreviations were then allowed so that the application of a function to several arguments successively would have a notation resembling that found in mathematics. Various possibilities are shown in Table 1.

31. It is syntactical in the broad sense, but not in the narrow sense (§ 1S2), because it is relational rather than logistic.

32. In the earlier theories there were also primitive constants. These and the variables together were called "proper symbols" in Kleene [PCF].

TABLE 1

[CLC]	[SPF]
Full	Full
(FA)	$\{F\} (A)$
$((FA) B)$	$\{\{F\} (A)\} (B)$
$(((FA) B) C)$	$\{\{\{F\} (A)\} (B)\} (C)$
$\lambda x.M$	$\lambda x.M$
Short	Short
FA	$F (A)$
FAB	$F (A, B)$
$FABC \dots$	$F (A, B, C)$
	$\lambda x.M$

Here the short forms for [SPF] suppose that F is a simple symbol; if not we replace F by $\{F\}$ and still get some shortening. Evidently Church used the Schönfinkel concept (he cited Schönfinkel), but refused to use the notation for it until 1941.

Since λ is a binary prefix, a notation without parentheses can be obtained by using a binary prefix for the application operator. Rosenbloom [EML] p. 123 uses such a notation with '!' as the prefix.³³

3. Motivation of the restricted systems

The theories of Church exhibit a curious allergy to both the λK -system and the principle of extensionality (§). We shall make here a few remarks about what seems to us to be the motivation back of these restrictions.

As to the λK -system, Church has advanced at various times three arguments: 1) that he can do anything he wants to do without it; 2) that it enables obs to be constructed which have a normal form although they have components which do not; 3) that it may lead to inconsistency. The first of these arguments is probably sound, and therefore there is some interest in seeing what can be done in the λI -system; but we find that the theory has then some difficulties and complexities, such as, for example, the fact that his numerals include a zero only at cost of admitting certain exceptions for it.³⁴ The second argument is connected with the fact that Church is thinking of an interpretation in which only obs with a normal form are significant. For us significance is an illative concept; when we come to illative combinatory logic we shall find that, although the presence of the combinator K (§ 5A) does cause some formal difficulties, there are points of view from which the thesis that only obs with a normal form are significant is not acceptable.³⁵ The third argument we believe is mistaken, although that is not quite certain; at least no inconsistency is known for a λK -type system which is not obtainable for the corresponding λI -type, and we know of no consistency proof which is essentially complicated by the greater freedom of the λK -system.

As for the extensionality principle, the argument of Church to the effect that for his theory of $\lambda\delta$ -conversion an extensional interpretation is impossible seems to us to be a fallacy. It depends on the example (see his [CLC], p. 64) $X \equiv \lambda xy. xy$, $Y \equiv \lambda xy. \delta_1 xy X xy$, which, he says, "correspond to the same function in extension". However, X and Y do not satisfy the hypothesis of (2). On the other hand there are, of course, many interpretations for which we may wish to distinguish between two combinators which correspond to the same combination with different orders, and for these (2) is not acceptable. We shall run across such situations in the theory of functionality.

These restricted systems have thus an interest from certain points of view. In this work we devote primary attention to the stronger systems; but we do not ignore the others completely.

33. Cf. § 1E3.

34. Cf. § 5E5.

35. See §§ 4E3, 8B, 8S1.

4. The δ_1 -system

As we saw in § 1, Church did not include his studies of the system of [PFC] in [CLC]. The only source of detailed information about this system is [ML δ_1] which uses the old notation. The system, however, is of some interest, and we therefore make some remarks about it.

The philosophy back of the system is expounded in [RPx]. There is a hierarchy of quantifiers akin to the \mathcal{E} of § B4. The system has thus the appearance of being illative. However, when the system is interpreted, these quantifiers correspond to an enumeration of the objects corresponding to obs constructed by certain processes within the system, rather than to generalization with respect to a category of objects existing independently of the system. The illative appearance is therefore illusory. But the system gives an interesting possibility of interpreting an illative system.

The system has another peculiarity in that truth is identified with the number 2. This is highly artificial. It is an instance of the reduction, mentioned in § 1E1, of a logistic system to relational form.