Journal of Philosophical Logic (2006)
DOI: 10.1007/s10992-005-9024-5

WAYNE AITKEN and JEFFREY A. BARRETT

STABILITY AND PARADOX IN ALGORITHMIC LOGIC

Received on 25 May 2005

ABSTRACT. There is significant interest in type-free systems that allow flexible self-application. Such systems are of interest in property theory, natural language semantics, the theory of truth, theoretical computer science, the theory of classes, and category theory. While there are a variety of proposed type-free systems, there is a particularly natural type-free system that we believe is prototypical: the logic of recursive algorithms. *Algorithmic logic* is the study of basic statements concerning algorithms and the algorithmic rules of inference between such statements. As shown in [1], the threat of paradoxes, such as the Curry paradox, requires care in implementing rules of inference in this context. As in any type-free logic, some traditional rules will fail. The first part of the paper develops a rich collection of inference rules that do not lead to paradox. The second part identifies traditional rules of logic that are paradoxical in algorithmic logic, and so should be viewed with suspicion in type-free logic generally.

KEY WORDS: abstraction, algorithmic logic, curry paradox, type-free logic

1. Introduction

In second-order logic, one distinguishes between two types of objects. *First-order objects* are the basic objects of interest. *Second-order objects* are the properties and classes, the functions and operators for the first-order objects. There are, however, situations in which this division is unnatural. When one wants, for whatever purpose, to mix the first-order and second-order universes, one is reminded of the reason for their original separation: the paradoxes.

This paper is a study in *type-free logic*. The goal of type-free logic is to find consistent, natural, and flexible ways to handle type-free systems where the second-order objects are not separated from the first-order objects, but are in some sense part of the first-order universe. In type-free logic one desires enough flexibility to including meaningful self-application and self-containment: functions and properties should be able to apply to themselves and collections should be able to contain themselves. One also wants natural or 'naïve' comprehension and functional abstraction principles to create collections and functions. One might also want a truth predicate.

Standard ZFC set theory fails these desiderata. Functions may be applied to functions, and collections may contain collections, but within limits: a function cannot be a member of its domain and a collection cannot contain itself. In ZFC only a well-behaved, well-founded part of the second-order universe is allowed inside the first-order universe. In fact, due to well-foundedness, the set-theoretic universe can be regarded as a typed theory where the universe is typed by ordinal rank. Meaningful self-application is blocked: the relation $x \in y$ is automatically false unless x has a lower ordinal rank than y. ZFC does not allow a universal set (and even extensions of ZFC such as GB or MK class theory do not allow a universal class that contains the universal class). A typical example of the limitations of ZFC in this regard concerns the status of the self-composition function T: given a function f with domain and codomain the same class, define Tf to be $f \circ f$. This natural function is not an object in the ZFC universe.[1] Now there is nothing preventing one from studying T in set theory, but the point is that it is external to the set-theoretical universe V, a universe intended to be rich enough for all of mathematics. This example is not atypical. Set theorists work outside of V whenever properties about the intersection $\cap$ or union $\cup$ operators are discussed: the set-theoretical operators $\cap$ or $\cup$ are not themselves objects of the set theoretical universe V.

Whenever a type-free system is considered with more expressive power than ZFC set theory, for example a system with unrestricted comprehension, the threat of paradox emerges anew. So some part or another of traditional logic must be restricted. Nevertheless, there is significant interest in type-free systems due to applications in property theory, natural language semantics, the theory of truth, theoretical computer science, the theory of classes, and category theory. In property theory, it is desirable, indeed arguably essential, that every open formula in a language should determine an associated object called a *property*. In natural language semantics there should be a truth predicate that behaves in a manner similar to the truth predicate in natural language. In class theory there should be nothing preventing a class from containing itself. In fact, any restriction on class comprehension seems artificial.[2] There should be a universal class, and this class should contain itself. And given its role as an organizing principle of contemporary mathematics, there should to be a more satisfying way to develop category theory than by employing the current large/small category distinction.

There are a variety of proposed type-free systems[3] which are provably free from contradictions engendered by paradoxes, and which restrict the traditional logics in one way or another. What remains is the question of which type-free systems are the most compelling. One obvious criterion is that a type-free system should not introduce artificialities worse that the artificiality of separating first-order objects from second-order objects.

In this paper we introduce a promising methodology for developing a natural type-free system. The common strategy is to start with some form of a classical logic with a naïve comprehension principle, then to weaken it until it is consistent. But it is unclear what to weaken. Our strategy, on the other hand, is to begin with a naturally occurring type-free system, then to investigate the logical properties it in fact possesses. The hope is that this naturally occurring type-free system will serve as a fruitful model for type-free systems more generally.

Perhaps the most natural type-free system is ordinary language, but for our purpose this system is hopelessly intractable. The universe of recursive algorithms, however, is both natural and tractable. If we fix a framework for algorithmic description, then self-application of algorithms is possible, indeed commonplace. If we focus on logical operators that can be defined algorithmically, a rich type-free logical structure emerges. *Algorithmic logic* is the study of this type-free system.

This paper is the second of a series designed to introduce and examine algorithmic logic. The first [1], a short and informal introduction to the subject, focussed on the challenge of the Curry paradox. The Curry paradox is the first test of any type-free system containing implication. The present paper builds on the lessons learned from the first, but is independent of it. It begins the formalization and careful study of algorithmic logic. The main task here is to understand which traditional rules of propositional logic are safe and which are problematic in algorithmic logic. A third paper [2] will discuss the principle of unrestricted functional abstraction in algorithmic logic. Fredrick Fitch [7] sought a type-free logic with an unrestricted abstraction principle; he regarded any restriction on abstraction as artificial and undesirable. Since algorithmic logic is both type-free and allows for a strong abstraction principle, our work can be viewed as part of the Fitch-Curry-Myhill tradition.[4]

2. Algorithmic Logic

The basic objects of algorithmic logic are *algorithmic statements*. An algorithmic statement is an assertion of the form *algorithm* α *with input* u *halts with output* v. The assertion $4! = 24$ can be understood as a true algorithmic statement, where the algorithm is one designed to calculate the factorial function, the input is 4, and the output is 24.

Algorithmic statements can be more subtle.[5] Consider Goldbach's conjecture that every even number greater than two is the sum of two prime numbers. The *negation* of Goldbach's conjecture can be understood as the algorithmic statement that GOLDBACH halts with output 0 when run with input 0, where GOLDBACH is the algorithm that checks each even number in turn, beginning with four, and outputs 0 if it ever finds a number that cannot be represented as the sum of two primes. Note that if Goldbach's conjecture is true, then the algorithm GOLDBACH will simply fail to halt regardless of input.

An algorithmic statement can be false in two ways. It can be false because the algorithm halts with an output different from the one specified. Such statements are *directly false*. Or it can be false because the algorithm fails to halt. Such statements are *indirectly false*.

The assertion that a specified algorithm *halts* on a specified input can also be understood as an algorithmic statement. Consider the algorithm HALT that takes as input a pair $[\alpha, u]$ and runs as a subprocess the algorithm α with input u. The algorithm HALT outputs 1 if the subprocess halts; otherwise HALT itself does not halt. So the algorithmic statement asserting that HALT outputs 1 on input $[\alpha, u]$ is true if and only if α halts on input u.

The algorithm HALT is an example of an *algorithmic predicate*, a predicate that can be represented by an algorithm that outputs 1 if and only if the predicate is true of the input. We require that if an algorithmic predicate halts at all, it outputs 0 or 1. Algorithmic predicates are the basic *internal* predicates of algorithmic logic.

There is an internal truth predicate TRUE for algorithmic statements. The algorithm TRUE expects as input a triple $[\alpha, u, v]$ representing an algorithmic statement with specified algorithm α, specified input u, and specified output v. First TRUE runs the subprocess α with input u. If this subprocess halts with output v, then TRUE outputs 1. If the subprocess halts with output not equal to v, then TRUE outputs 0. If the subprocess fails to halt, then TRUE also fails to halt.

Closely related to the truth predicate, is an algorithmic predicate corresponding to directly false. Because of the halting problem, however, there is no algorithmic predicate corresponding to false: the *external* property of being false is one that cannot be expressed internally.

Finally, *algorithmic connectives* can be defined in terms of algorithmic predicates. The algorithmic conjunction $\wedge$ and disjunction $\vee$ behave as expected, but the algorithmic conditional $\overset{\rho}{\Rightarrow}$ requires special care.

Here the conditional is indexed by a library ρ of inference rules. The algorithmic statement $A \overset{\rho}{\Rightarrow} B$ means that the algorithmic statement B can be deduced from the algorithmic statement A using the rules in the library ρ. Because of its definition, the connective $\overset{\rho}{\Rightarrow}$ can be used to define an internal predicate PROVE_ρ. The connective $\overset{\rho}{\Rightarrow}$ is also used to define negation $\overset{\rho}{\neg}$.

Since algorithms can take algorithms as input, as in the case of HALT above, algorithmic logic is inherently self-referential and so is essentially type-free. Consequently, special care must be taken to avoid contradiction: the rules of the library ρ must be carefully evaluated for validity. Indeed, in an earlier paper [1] we show that the rule *modus ponens* for $\overset{\rho}{\Rightarrow}$ cannot be included in a sufficiently rich library ρ without rendering the rule itself invalid. If *modus ponens* is included in such a library, an algorithmic version of the Curry paradox results in a contradiction.

The first part of the present paper introduces rules for algorithmic logic that form a *stable base*: a valid collection of rules that can be safely extended to form stronger valid collections. The second part of the paper presents a list of *paradoxical rules*: traditional rules of logic that can be shown to be invalid when in a sufficiently rich library, usually through arguments akin to those found in the Russell and Curry paradoxes.

3. Conventions for Algorithms

Rather than stipulate a particular theoretical framework for the discussion of algorithms, we require that a suitable framework behaves as follows.

Anything that can be input into an algorithm is called a *datum*. Data include natural numbers and algorithms. In addition, if $a_1 \dots a_k$ are data, the list $[a_1, \dots, a_k]$ is itself a datum. Every algorithm accepts exactly one input datum and either does not halt or halts with exactly one output datum. If an algorithm requires or produces multiple data, the data are packaged in a single input or output list respectively. Any datum is an allowable input whether or not it is consistent with the intended function of the algorithm. Typically, we will not specify what an algorithm does with an unexpected input datum.

Every datum has a positive integer *size*, and there are only a finite number of data of a given size. The size of a list is strictly greater than the sum of the sizes of the items of the list. A *process* is a pair consisting of an algorithm and an input. Every halting process has a positive integer *runtime*. If a parent process runs one or more subprocesses in its execution, then the runtime of the parent process is strictly greater than the sum of the runtimes of the halting subprocesses.

An *algorithmic statement* can be represented as a datum: If α is the specifed algorithm, u the specified input, and v the specified output, then the list $[\alpha, u, v]$ represents the corresponding algorithmic statement.

The identity algorithm IDENTITY simply outputs a copy of its input. The algorithmic statement $[\text{IDENTITY}, 0, 0]$ is denoted $\mathcal{T}$. This algorithmic statement is true. Similarly, the algorithmic statement $[\text{IDENTITY}, 0, 1]$ is denoted $\mathcal{F}$. This statement is directly false since $0 \neq 1$.

4. DEDUCTION

There is an algorithmic predicate for deduction. This deduction predicate depends on a library of rules instantiated by an algorithmic sequence.

DEFINITION 4.1. An *algorithmic sequence* is an algorithm which halts for every positive integer input. If α is an algorithmic sequence, then α_n denotes the output of α applied to the integer n.

Informally, a rule is an algorithm which expects as input a list of algorithmic statements which it treats as hypotheses. It seeks to generate statements which are logically entailed by these hypotheses. It outputs a list consisting of the input list together with the newly generated statements, if any. Some rules will require a resource integer m in order to limit the amount of time that the rule uses. Resource integers are important in order to allow different rules to take turns being applied by a supervising process. Finally, some rules depend on the choice of a library ρ, so in general a library (or at least an algorithmic sequence which the rule treats as a library) must be included in the input.

DEFINITION 4.2. A *rule* is an algorithm α which expects an input of the form $[H, \rho, m]$ where H is a list of algorithmic statements, ρ is an algorithmic sequence, and m is a positive integer. For any such input, α is required to halt with output consisting of a list of algorithmic statements containing H as an initial sublist. Call H the *hypothesis list*, ρ the *nominal library*, and m the *resource integer*. The output of the rule α is the *conclusion list*.

For convenience, we require a *monotonicity property*: If $m' \geq m$ and if every item of H is also an item of H', then every item of the conclusion list for input $[H, \rho, m]$ is also an item of the conclusion list for input $[H', \rho, m']$.

DEFINITION 4.3. A *library* is an algorithmic sequence ρ such that ρ_n is a rule for all positive integers n.

As defined, a library is an infinite sequence of rules, but these rules are not necessarily distinct. In fact, any finite collection of data can be represented as an algorithmic sequence α by defining α_n to be α_N for all $n \geq N$ where N is the size of the collection. Thus the definition does not exclude finite libraries.

DEFINITION 4.4. A rule is *ρ-valid* for a library ρ if, for all hypothesis lists H consisting of only true statements and for all resource integers m, the conclusion list for input $[H, \rho, m]$ consists only of true statements. A library ρ is *valid* if it contains only ρ-valid rules.

DEFINITION 4.5. Let $A_1, \ldots, A_n$ and B be algorithmic statements. Let ρ_k be the kth rule of a library ρ. The statement B is a *direct ρ_k-consequence* of $A_1, \ldots, A_n$ if there is a hypothesis list H and a resource integer m such that (i) every item of H is in $\{A_1, \ldots, A_n\}$ and (ii) B is an item of the conclusion list obtained by running ρ_k with input $[H, \rho, m]$.[6]

DEFINITION 4.6. A set of algorithmic statements S is *ρ-deductively closed* if, for all k, every direct ρ_k-consequence of elements in S is itself in S.

LEMMA 4.7. *If ρ is a valid library then the set S of true algorithmic statements is ρ-deductively closed.*

LEMMA 4.8. *The intersection of ρ-deductive closed sets is ρ-deductively closed.*

DEFINITION 4.9. Let S be a set of algorithmic statements. The *ρ-deductive closure* $\overline{S}$ of S is the intersection of all ρ-deductively closed sets containing S.

LEMMA 4.10. *The ρ-deductive closure $\overline{S}$ of a set of algorithmic statements is the minimal ρ-deductively closed set containing S. Thus $\overline{\overline{S}} = \overline{S}$.*

The ρ-deductive closure of a finite set $\{A_1, \ldots, A_n\}$ of algorithmic statements can be explicitly constructed as follows. Let $f : \mathbb{N}_+ \to \mathbb{N}_+ \times \mathbb{N}_+$ be a recursive bijection. Define $H_0 = [A_1, \ldots, A_n]$. For $i > 0$, define H_i to be the conclusion list obtained by running ρ_k on $[H_{i-1}, \rho, m]$ where $f(i) = (k, m)$.

LEMMA 4.11. *B is in the ρ-deductive closure of $\{A_1, \ldots, A_n\}$ if and only if B is an item of H_i for some i.*

Proof. Let S be the set of statements on $H_0, H_1, \ldots$. The strategy is to show (i) the set of items of any particular H_i is in the ρ-deductive closure (so S is a subset of the ρ-deductive closure), and (ii) S is ρ-deductively closed.

(i) By induction on i. The case $i = 0$ is clear. Assume that every item of H_{i-1} is in the ρ-deductive closure. If B is an item of H_i, and if $f(i) = (k, m)$, then B is a direct ρ_k-consequence of the items of H_{i-1}. So B is in the ρ-deductive closure.

(ii) Suppose B is a direct ρ_k-consequence of $C_1, \ldots, C_r \in S$. We must show that $B \in S$. By definition, B is on the conclusion list obtained by running ρ_k with input $[H, \rho, m]$ for some integer m and some list H where every item of H is in the set $\{C_1, \ldots, C_r\}$. By the mononicity requirement for rules, if H' is any list whose items include each $C_1, \ldots, C_r$ and if $m' \geq m$ then B is on the conclusion list when ρ_k is run with input $[H', \rho, m']$.
Let i_0 be an integer such that $C_1, \ldots, C_r$ are all on H_{i_0}. There are an infinite number of pairs (k, m') with $m' \geq m$, and all but a finite number are of the form $f(i)$ for $i > i_0$. Choose such an i. So B is on the conclusion list when ρ_k is run with the input $[H_{i-1}, \rho, m']$. That is, B is on H_i. Thus $B \in S$. $\square$

DEFINITION 4.12. The algorithm DEDUCE expects an input of the form $[\Gamma, \rho, B]$ where Γ is a list of algorithmic statements, ρ is a library, and B is an algorithmic statement. It computes $H_0, H_1, \ldots$, where $H_0 = \Gamma$ and H_i is defined as above. After computing H_k, DEDUCE checks to see if B is on H_k. If so, DEDUCE outputs 1; otherwise, it calculates H_{k+1}.

DEFINITION 4.13. Let B be an algorithmic statement and Γ a list of algorithmic statements. The algorithmic statement $\left[\text{DEDUCE}, [\Gamma, \rho, B], 1\right]$ is denoted as $\Gamma \vdash_{\rho} B$ (usually ρ is a library, but the definition applies to any datum ρ). If Γ is the list $[A_1, \ldots, A_n]$ one may write $A_1, \ldots, A_n \vdash_{\rho} B$ instead. Likewise, $\Gamma, C_1, \ldots, C_k \vdash_{\rho} B$ denotes $\Gamma' \vdash_{\rho} B$ where Γ' is the list obtained by appending $C_1, \ldots, C_k$ to the list Γ.

PROPOSITION 4.14. *Suppose $\Gamma = [A_1, \ldots, A_n]$ where $A_1, \ldots, A_n$ are algorithmic statements, and suppose ρ is a library. Then $\Gamma \vdash_{\rho} B$ if and only if B is in the ρ-deductive closure of $\{A_1, \ldots, A_n\}$.*

Proof. This follows from Lemma 4.11 and the definition of DEDUCE. $\square$

In particular, if $\Gamma \vdash_\rho B$ then any ρ-deductively closed set containing all the items of Γ contains B.

COROLLARY 4.15. *Let A and B be algorithmic statements, Γ and Γ' lists of algorithmic statements, and ρ a library.*

(i) *If every item of Γ is on Γ' and if $\Gamma \vdash_\rho A$ then $\Gamma' \vdash_\rho A$.*
(ii) *$A \vdash_\rho A$.*
(iii) *If $\Gamma \vdash_\rho A$ and $\Gamma, A \vdash_\rho B$ then $\Gamma \vdash_\rho B$.*
(iv) *If $\Gamma \vdash_\rho A$ and if $\Gamma' \vdash_\rho C_i$ for all items C_i of Γ, then $\Gamma' \vdash_\rho A$.*

Proof. (i) If $S_1 \subseteq S_2$ then $\overline{S_1} \subseteq \overline{S_2}$. (ii) $S \subseteq \overline{S}$. (iii) Let S be the ρ-deductive closure of the items of Γ. So $A \in S$. Since $\Gamma, A \vdash_\rho B$ and S is ρ-deductively closed, $B \in S$. (iv) Let S be the ρ-deductive closure of the items of Γ'. So every item C_i of Γ is in S. Since $\Gamma \vdash_\rho A$ and since S is deductively closed, S must contain A. □

PROPOSITION 4.16 (Soundness). *Suppose every statement on the list Γ is true, ρ is a valid library, and $\Gamma \vdash_\rho B$. Then B is true.*

Proof. The set of true statements S is ρ-deductively closed by Lemma 4.7. The result follows from Proposition 4.14. □

Because DEDUCE is an *internal* predicate representing deduction, one can use $\vdash_\rho$ to define a conditional connective $\stackrel{\rho}{\Rightarrow}$. (A material conditional $\rightarrow$, not dependent on DEDUCE, will be defined in Section 12). The algorithm DEDUCE can also be used to define an internal provability predicate PROVE_ρ.

DEFINITION 4.17. Let $A \stackrel{\rho}{\Rightarrow} B$ denote $A \vdash_\rho B$. Let $\text{PROVE}_\rho(A)$ denote $\mathcal{T} \stackrel{\rho}{\Rightarrow} A$.

The above results, restated in this notation, yield the following.

PROPOSITION 4.18. *Let A, B, and C be algorithmic statements, and ρ a library. Then*

(i) *$A \stackrel{\rho}{\Rightarrow} A$, and*
(ii) *if $A \stackrel{\rho}{\Rightarrow} B$ and $B \stackrel{\rho}{\Rightarrow} C$ then $A \stackrel{\rho}{\Rightarrow} C$.*

Moreover, if ρ is a valid library, then

(iii) *if $A \stackrel{\rho}{\Rightarrow} B$ and A are true, then so is B, and*
(iv) *if $\text{PROVE}_\rho(A)$ is true, then so is A.*

5. Transitivity Rule

In the next several sections 11 inference rules will be introduced. These rules will be used to form a stable base (in the sense of Definition 13.1). The first is an internal implementation of Proposition 4.18(ii).

RULE 1. The *Transitivity Rule* is an algorithm that implements the *rule diagram*

$$\frac{A \overset{\rho}{\Rightarrow} B \qquad B \overset{\rho}{\Rightarrow} C}{A \overset{\rho}{\Rightarrow} C.}$$

In other words, assuming the input is of the expected form $[H, \rho, m]$, the Transitivity Rule first copies the hypothesis list H to a working list Δ. Then it looks for a statement of the form $A \overset{\rho}{\Rightarrow} B$ and a statement of the form $B \overset{\rho}{\Rightarrow} C$ on the hypothesis list H where A, B, C are algorithmic statements. For all such pairs that it finds, the Transitivity Rule appends the statement $A \overset{\rho}{\Rightarrow} C$ to the working list Δ. After processing all such pairs, it outputs the resulting list Δ as its conclusion list.

PROPOSITION 5.1. *The Transitivity Rule is ρ-valid for all libraries ρ.*

Proof. The ρ-validity of this rule follows from Proposition 4.18(ii). □

6. Universal Rules

RULE 2. The *Universal Rule* is an algorithm that generates all true algorithmic statements. More specifically, assuming the input is of the expected form $[H, \rho, m]$, the Universal Rule outputs a list consisting of H appended with all m-true algorithmic statements. An algorithmic statement B is *m-true* if (i) the datum B has size at most m, (ii) the runtime of the associated process is at most m, and (iii) B is true.

PROPOSITION 6.1. *The Universal Rule is ρ-valid for all libraries ρ.*

Proof. The Universal Rule only appends true statements to the input list. □

PROPOSITION 6.2. *Suppose the library ρ contains the Universal Rule. Let Γ be a list of algorithmic statements, and A and B be algorithmic* *statements. If B is true then $\Gamma \vdash_\rho B$. In particular, if B is true, then so is $A \stackrel{\rho}{\Rightarrow} B$ and* $\text{PROVE}_\rho(B)$.

Proof. Every true algorithmic statement is m-true for some m. So the ρ-deductive closure of any set contains all true statements. □

COROLLARY 6.3. *If the library ρ is valid and contains the Universal Rule, then an algorithmic statement A is true if and only if* $\text{PROVE}_\rho(A)$.

Proof. This follows from Proposition 4.18(iv) and Proposition 6.2. □

So, in algorithmic logic, there is a sense in which internal deduction is complete for any valid library containing the Universal Rule. By Proposition 13.7, however, there is also a sense in which algorithmic logic is inherently incomplete.

RULE 3. The *Meta-Universal Rule* is an algorithm that implements the rule diagram

$$\frac{B}{A \stackrel{\rho}{\Rightarrow} B.}$$

More specifically, assuming an input of the expected form $[H, \rho, m]$, the Meta-Universal Rule appends to H all statements of the form $A \stackrel{\rho}{\Rightarrow} B$ where (i) B is on H and (ii) the size of the datum $A \stackrel{\rho}{\Rightarrow} B$ is at most m.

The Meta-Universal Rule is the first rule whose validity is contingent on the *contents* of the library.

PROPOSITION 6.4. *If the library ρ contains the Universal Rule, then the Meta-Universal Rule is ρ-valid.*

Proof. This follows from Proposition 6.2. □

PROPOSITION 6.5. *If the library ρ contains the Transitivity Rule and the Meta-Universal Rule, then*

(i) $A, A \stackrel{\rho}{\Rightarrow} C \vdash_\rho B \stackrel{\rho}{\Rightarrow} C$, *and*
(ii) $A, A \stackrel{\rho}{\Rightarrow} C \vdash_\rho \text{PROVE}_\rho(C)$.

Proof.

(i) Let S be the ρ-deductive closure of (the set consisting of) A and $A \stackrel{\rho}{\Rightarrow} C$. By the Meta-Universal Rule, $B \stackrel{\rho}{\Rightarrow} A$ is in S. By the Transitivity Rule, $B \stackrel{\rho}{\Rightarrow} C$ is in S.
(ii) This is a special case of Part(i) where B is $\mathcal{T}$. □

7. Conjunction

DEFINITION 7.1. The algorithm AND expects as input a list $[A, B]$ where A and B are algorithmic statements. If A and B are true, then AND outputs 1. If either is directly false, then AND outputs 0. Otherwise, AND does not halt. If A and B are algorithmic statements, then $\big[\text{AND}, [A, B], 1\big]$ is denoted by $A \wedge B$.

If $\Gamma = [C_1, \ldots, C_k]$ is a list of algorithmic statements, then the *conjunction* $C_1 \wedge \ldots \wedge C_k$ of Γ is defined to be $(C_1 \wedge \ldots \wedge C_{k-1}) \wedge C_k$. If $k = 1$ then the conjunction is simply defined to be C_1, and if $k = 0$ (so Γ is the empty list) then the conjunction is defined to be $\mathcal{T}$. Observe that the conjunction $C_1 \wedge \cdots \wedge C_k$ is true if and only if each C_i is true. Similarly, the conjunction is directly false if and only if some C_i is directly false.

RULE 4. The *Conjunction Rule* is an algorithm that simultaneously implements the following three rule diagrams:

$$\frac{\begin{array}{c}A\\ B\end{array}}{A \wedge B} \qquad \frac{A \wedge B}{A} \qquad \frac{A \wedge B}{B}.$$

More specifically, for any statements A and B on H, the Conjunction Rule appends $A \wedge B$ to H. In addition, for any statement $A \wedge B$ on the given H, the Conjunction Rule appends A and B to H.

PROPOSITION 7.2. *The Conjunction Rule is ρ-valid for all libraries ρ.*

PROPOSITION 7.3. *Let S be a ρ-deductively closed set of algorithmic statements where ρ is a library containing the Conjunction Rule. Let $A_1, \ldots, A_k$ be algorithmic statements where $k \geq 1$. The conjunction $A_1 \wedge \cdots \wedge A_k$ is in S if and only if each A_i is in S. (If $k = 0$ assume that ρ contains the Universal Rule instead of the Conjunction Rule).*

COROLLARY 7.4. *Let ρ be a library containing the Conjunction Rule. The logical connective $\wedge$ satisfies both the symmetry and associativity laws:*

(i) $A \wedge B \vdash_\rho B \wedge A$.
(ii) $(A \wedge B) \wedge C \vdash_\rho A \wedge (B \wedge C)$ *and* $A \wedge (B \wedge C) \vdash_\rho (A \wedge B) \wedge C$.

COROLLARY 7.5. *Let $\Gamma = [C_1, \ldots, C_k]$ be a list of algorithmic statements, $A_1, \ldots, A_n, B$ be algorithmic statements, and $C = C_1 \wedge \cdots \wedge C_k$.*

Assume that ρ contains the Conjunction Rule and the Universal Rule (for the case $k = 0$ or $n = 0$). Then

(i) $\Gamma \vdash_{\rho} A_1 \wedge \cdots \wedge A_n$ *if and only if* $\Gamma \vdash_{\rho} A_i$ *for each* A_i, *and*
(ii) $\Gamma \vdash_{\rho} B$ *if and only if* $C \overset{\rho}{\Rightarrow} B$.

RULE 5. The *Meta-Conjunction Rule* is an algorithm that implements the rule diagram

$$\frac{\begin{array}{c} A \overset{\rho}{\Rightarrow} B \\ A \overset{\rho}{\Rightarrow} C \end{array}}{A \overset{\rho}{\Rightarrow} (B \wedge C).}$$

PROPOSITION 7.6. *The Meta-Conjunction Rule is ρ-valid for all libraries ρ containing the Conjunction Rule.*

Proof. Assume $A \overset{\rho}{\Rightarrow} B$ and $A \overset{\rho}{\Rightarrow} C$. Let S be the ρ-deductive closure of A. By assumption B and C are in S. By the Conjunction Rule $B \wedge C$ is also in S. Therefore, $A \overset{\rho}{\Rightarrow} B \wedge C$. □

Several laws can be deduced from the above rules.

PROPOSITION 7.7. *If ρ contains all the above rules, then*

(i) $A \overset{\rho}{\Rightarrow} B \;\vdash_{\rho}\; A \overset{\rho}{\Rightarrow} B \wedge A$,
(ii) $A \overset{\rho}{\Rightarrow} B,\; B \wedge A \overset{\rho}{\Rightarrow} C \;\vdash_{\rho}\; A \overset{\rho}{\Rightarrow} C$, *and*
(iii) $A \overset{\rho}{\Rightarrow} B \;\vdash_{\rho}\; C \wedge A \overset{\rho}{\Rightarrow} C \wedge B, \quad A \overset{\rho}{\Rightarrow} B \;\vdash_{\rho}\; A \wedge C \overset{\rho}{\Rightarrow} B \wedge C$.

Proof.

(i) Let S be the ρ-deductive closure of $A \overset{\rho}{\Rightarrow} B$. The statement $A \overset{\rho}{\Rightarrow} A$ is true by Proposition 4.18(i). By the Universal Rule, $A \overset{\rho}{\Rightarrow} A$ is in S. So by the Meta-Conjunction Rule $A \overset{\rho}{\Rightarrow} B \wedge A$ is in S.
(ii) Let S be the ρ-deductive closure of $A \overset{\rho}{\Rightarrow} B$ and $B \wedge A \overset{\rho}{\Rightarrow} C$. By the first part, $A \overset{\rho}{\Rightarrow} B \wedge A$ is in S. So, by the Transitivity Rule, $A \overset{\rho}{\Rightarrow} C$ is in S.
(iii) This follows by a similar argument. □

PROPOSITION 7.8. *Suppose ρ contains all the above rules. If $B \wedge A \overset{\rho}{\Rightarrow} C$ then $B \overset{\rho}{\Rightarrow} (A \overset{\rho}{\Rightarrow} C)$.*

Proof. Let S be the ρ-deductive closure of B. By supposition $B \wedge A \overset{\rho}{\Rightarrow} C$ holds, so is in S by the Universal Rule. By the Meta-

Universal Rule, $A \overset{\rho}{\Rightarrow} B$ is in S. Finally, by Proposition 7.7(ii), $A \overset{\rho}{\Rightarrow} C$ is in S. □

THEOREM 7.9 *Suppose ρ contains all the above rules. Let Γ be a list of algorithmic statements, and let A and C be algorithmic statements. If $\Gamma, A \vdash_\rho C$ then $\Gamma \vdash_\rho A \overset{\rho}{\Rightarrow} C$.*

Proof. Let $\Gamma = [B_1, \ldots, B_k]$ and $B = B_1 \wedge \cdots \wedge B_k$. If $\Gamma, A \vdash_\rho C$, then $B \wedge A \overset{\rho}{\Rightarrow} C$ by Corollary 7.5(ii). By Proposition 7.8, $B \overset{\rho}{\Rightarrow} (A \overset{\rho}{\Rightarrow} C)$ holds. By Corollary 7.5(ii) again, $\Gamma \vdash_\rho A \overset{\rho}{\Rightarrow} C$. □

COROLLARY 7.10 *If ρ contains all the above rules, then*

(i) $A \ \vdash_\rho \ B \overset{\rho}{\Rightarrow} A \wedge B$,
(ii) $A \overset{\rho}{\Rightarrow} B \ \vdash_\rho \ (B \overset{\rho}{\Rightarrow} C) \overset{\rho}{\Rightarrow} (A \overset{\rho}{\Rightarrow} C)$,
(iii) $A \overset{\rho}{\Rightarrow} B \ \vdash_\rho \ (C \overset{\rho}{\Rightarrow} A) \overset{\rho}{\Rightarrow} (C \overset{\rho}{\Rightarrow} B)$, *and*
(iv) $B \wedge A \overset{\rho}{\Rightarrow} C \ \vdash_\rho \ B \overset{\rho}{\Rightarrow} (A \overset{\rho}{\Rightarrow} C)$.

Proof.

(i) By the Conjunction Rule, $A, B \vdash_\rho A \wedge B$. Now use Theorem 7.9.
(ii) By the Transitivity Rule, $A \overset{\rho}{\Rightarrow} B, B \overset{\rho}{\Rightarrow} C \vdash_\rho A \overset{\rho}{\Rightarrow} C$. Now use Theorem 7.9. Part(iii) is similar.
(iii) Let S be the ρ-deductive closure of $B \wedge A \overset{\rho}{\Rightarrow} C$ and B. By the Meta-Universal Rule, $A \overset{\rho}{\Rightarrow} B$ is in S. By Proposition 7.7(ii), $A \overset{\rho}{\Rightarrow} C$ is in S. Thus $B \wedge A \overset{\rho}{\Rightarrow} B \vdash_\rho A \overset{\rho}{\Rightarrow} C$. Now use Theorem 7.9. □

8. BICONDITIONAL

DEFINITION 8.1. Let $A \overset{\rho}{\Longleftrightarrow} B$ denote $(A \overset{\rho}{\Rightarrow} B) \wedge (B \overset{\rho}{\Rightarrow} A)$.

PROPOSITION 8.2. *The following laws hold for any library ρ:*

(i) $A \overset{\rho}{\Longleftrightarrow} A$,
(ii) *If* $A \overset{\rho}{\Longleftrightarrow} B$ *then* $B \overset{\rho}{\Longleftrightarrow} A$, *and*
(iii) *If* $A \overset{\rho}{\Longleftrightarrow} B$ *and* $B \overset{\rho}{\Longleftrightarrow} C$, *then* $A \overset{\rho}{\Longleftrightarrow} C$.

Some results concerning conjunction can be conveniently expressed with the biconditional.

PROPOSITION 8.3. *Suppose ρ is a library containing the Conjunction and Universal Rules. Then*

(i) $A \overset{\rho}{\Longleftrightarrow} A \wedge A$,
(ii) $A \overset{\rho}{\Longleftrightarrow} A \wedge \mathcal{T}$,

(iii) $A \wedge B \overset{\rho}{\Longleftrightarrow} B \wedge A$, *and*
(iv) $A \wedge (B \wedge C) \overset{\rho}{\Longleftrightarrow} (A \wedge B) \wedge C$.

9. Disjunction

DEFINITION 9.1. The algorithm OR expects as input a list $[A, B]$ where A and B are algorithmic statements. If either A or B are true, then OR outputs 1. If both are directly false, then OR outputs 0. Otherwise, OR does not halt.

If A and B are algorithmic statements, then we denote $\big[\text{OR}, [A, B], 1\big]$ by $A \vee B$. The statement $A \vee B$ is true if and only if either A is true or B is true. Similarly, $A \vee B$ is directly false if and only if both A and B are directly false.

RULE 6. The *Disjunction Introduction Rule* is an algorithm that simultaneously implements the following two rule diagrams:

$$\frac{A}{A \vee B} \qquad \frac{B}{A \vee B}.$$

More specifically, assuming an input in the expected form $[H, \rho, m]$, the Disjunction Introduction Rule appends to H all statements of the form $A \vee B$ where (i) either A or B is on H and (ii) the size of $A \vee B$ is at most m.

PROPOSITION 9.2. *The Disjunction Introduction Rule is ρ-valid for all libraries ρ.*

At this point one might expect an algorithmic disjunction elimination rule allowing the deduction of C from $A \overset{\rho}{\Rightarrow} C$, $B \overset{\rho}{\Rightarrow} C$, and $A \vee B$. The difficulties of such a rule will be discussed in Section 14. An unproblematic but weaker version of this rule can be produced by requiring a sort of verification for the hypotheses $A \overset{\rho}{\Rightarrow} C$ and $B \overset{\rho}{\Rightarrow} C$. The following rule implements this idea.[7]

RULE 7. The *Disjunction Elimination Rule* is an algorithm, denoted D-ELIM, that implements the rule diagram

$$\begin{array}{l} G \wedge A \overset{\rho}{\Rightarrow} C \;\; * \\ G \wedge B \overset{\rho}{\Rightarrow} C \;\; * \\ G \\ \underline{A \vee B \qquad\quad} \\ C \end{array}$$

where $*$ indicates that the corresponding statement must be *verified*. More specifically, assuming an input of the expected form $[H, \rho, m]$, whenever D-ELIM finds four statements on H of the form of the premises of the rule diagram, it determines if the runtimes of the processes associated with the first two statements in the diagram are less than m. If the runtimes are both less than m and if both statement are *true*, then D-ELIM appends the statement represented by C to the conclusion list.

Proposition 4.18(iii) and the algorithmic definition of $\vee$ gives validity:

PROPOSITION 9.3. *The Disjunction Elimination Rule is ρ-valid for all valid libraries ρ.*

This is the first rule we have considered where the validity of the rule is contingent on the validity of the library. If the library ρ is valid, then this rule is ρ-valid, but in Section 14 we shall see several examples of valid rules that cannot themselves be contained in a stable base (in the sense of Definition 13.1). Since the goal is to form a stable base of inference rules, we need something stronger than the above proposition. Theorem 9.5 is sufficient.

LEMMA 9.4. *If a library ρ is not valid, but contains the Conjunction Rule, then there are algorithmic statements A and B such that A and $A \overset{\rho}{\Rightarrow} B$ are true, but B is false.*

Proof. Since ρ is not valid, there is a rule ρ_k in ρ which is not ρ-valid. In other words, there is a list H of true statements and an integer m such that when the list $[H, \rho, m]$ is given as input to ρ_k, the rule generates an output list containing at least one false statement B. Let $H = [A_1, \ldots, A_n]$ and let $A = A_1 \wedge \cdots \wedge A_n$. Note that A is true. Let S be the ρ-deductive closure of A. Since ρ contains the Conjunction Rule, each A_i is in S. So B is in S by the definition of the deductive closure. Thus $A \overset{\rho}{\Rightarrow} B$ is true. $\square$

THEOREM 9.5. *Let ρ be a library that contains at least the Conjunction Rule and the Disjunction Elimination Rule. Suppose that every rule in ρ other than the Disjunction Elimination Rule is ρ-valid. Then ρ is valid.*

Proof. Suppose to the contrary that ρ is not valid. By the previous lemma there are statements D and E such that D is true, $D \overset{\rho}{\Rightarrow} E$ is true, but E is false. Choose D and E so that the runtime r of the process associated with $D \overset{\rho}{\Rightarrow} E$ is minimal.

Let $H_0 = [D]$. Since $D \overset{\rho}{\Rightarrow} E$, when $[H_0, \rho, E]$ is input to DEDUCE the output is 1. Recall that DEDUCE generates a monotonic sequence $H_0, H_1, \ldots$ of lists, and since it outputs 1, it eventually generates a list

H_k containing E. Thus, since H_0 contains only true statements but E is false, there is a unique $i \geq 1$ such that H_{i-1} contains only true statements and H_i contains at least one false statement C. Let the function f be as in the definition of DEDUCE, and let $f(i) = (k, m)$. Thus H_i is obtained by running ρ_k with input $[H_{i-1}, \rho, m]$. Note that ρ_k cannot be ρ-valid, so ρ_k must be D-ELIM (since we assumed that all other rules are ρ-valid). Since D-ELIM generates C, H_{i-1} must contain statements of the form (i) $G \wedge A \overset{\rho}{\Rightarrow} C$, (ii) $G \wedge B \overset{\rho}{\Rightarrow} C$, (iii) G, and (iv) $A \vee B$. These four statements are true since they are on H_{i-1}. So either A or B is true, and it is enough to consider the case where A is true. In this case $G \wedge A$ is true. Since D-ELIM generates the statement C, it must first run the process associated with $G \wedge A \overset{\rho}{\Rightarrow} C$ and determine that the statement is true. The runtime r of the global process associated with $D \overset{\rho}{\Rightarrow} E$ must be strictly larger than the runtime r' associated with $G \wedge A \overset{\rho}{\Rightarrow} C$ (since r' is the runtime a subprocess of a subprocess of the global process associated with $D \overset{\rho}{\Rightarrow} E$). Since $r' < r$ and since both $G \wedge A$ and $G \wedge A \overset{\rho}{\Rightarrow} C$ are true, it follows from the definition of r that C must be true, a contradiction. □

PROPOSITION 9.6. *Let ρ be a library containing the Universal, Conjunction, and Disjunction Elimination Rules.*

(i) *If $G \wedge A \overset{\rho}{\Rightarrow} C$ and $G \wedge B \overset{\rho}{\Rightarrow} C$ then $G \wedge (A \vee B) \overset{\rho}{\Rightarrow} C$.*
(ii) *If $\Gamma, A \vdash_{\rho} C$ and $\Gamma, B \vdash_{\rho} C$, then $\Gamma, A \vee B \vdash_{\rho} C$.*

Proof.

(i) Let S be the ρ-deductive closure of $G \wedge (A \vee B)$. We must show that C is in S. By the Conjunction Rule, G and $A \vee B$ are in S. By the Universal Rule, $G \wedge A \overset{\rho}{\Rightarrow} C$ and $G \wedge B \overset{\rho}{\Rightarrow} C$ are also in S. So by the Disjunction Elimination Rule, C is in S (where D-ELIM needs a resource number m larger than the runtimes associated with $G \wedge A \overset{\rho}{\Rightarrow} C$ and $G \wedge B \overset{\rho}{\Rightarrow} C$).
(ii) This follows from Part(i) and Corollary 7.5(ii). □

PROPOSITION 9.7. *If ρ contains all the above rules, then*

(i) $A \vee \mathcal{T} \overset{\rho}{\Longleftrightarrow} \mathcal{T}$,
(ii) $A \overset{\rho}{\Longleftrightarrow} A \vee A$,
(iii) $A \vee B \overset{\rho}{\Longleftrightarrow} B \vee A$, *and*
(iv) $A \overset{\rho}{\Longleftrightarrow} A \wedge (A \vee B)$.
(v) $A \overset{\rho}{\Longleftrightarrow} A \vee (A \wedge B)$.
(vi) $A \vee (B \vee C) \overset{\rho}{\Longleftrightarrow} (A \vee B) \vee C$.

Proof.

(i) to (v) These are similar to and easier than Part (vi).

(vi) The Disjunction Introduction Rule (twice) gives $B \vdash_\rho (A \vee B) \vee C$. Likewise, $C \vdash_\rho (A \vee B) \vee C$. Proposition 9.6(ii) gives $B \vee C \vdash_\rho (A \vee B) \vee C$. The Disjunction Introduction Rule (twice) gives $A \vdash_\rho (A \vee B) \vee C$. Finally, Proposition 9.6(ii) gives $A \vee (B \vee C) \vdash_\rho (A \vee B) \vee C$. This gives one direction. The other direction follows from a similar argument. □

PROPOSITION 9.8. *If ρ contains all the above rules, then*

(i) $A \wedge (B \vee C) \overset{\rho}{\Longleftrightarrow} (A \wedge B) \vee (A \wedge C)$, *and*
(ii) $A \vee (B \wedge C) \overset{\rho}{\Longleftrightarrow} (A \vee B) \wedge (A \vee C)$.

Proof.

(i) By the Disjunction Introduction Rule, $A \wedge B \vdash_\rho (A \wedge B) \vee (A \wedge C)$ and $A \wedge C \vdash_\rho (A \wedge B) \vee (A \wedge C)$. Now use Proposition 9.6(i) to show $A \wedge (B \vee C) \vdash_\rho (A \wedge B) \vee (A \wedge C)$. The other direction is similar.

(ii) Showing $A \vee (B \wedge C) \vdash_\rho (A \vee B) \wedge (A \vee C)$ is similar to Part(i). For the other direction, first show $C, A \vdash_\rho A \vee (B \wedge C)$ and $C, B \vdash_\rho A \vee (B \wedge C)$ using the Disjunction Introduction and Conjunction Rules. Use Proposition 9.6(ii) to get $C, A \vee B \vdash_\rho A \vee (B \wedge C)$. In other words, $A \vee B, C \vdash_\rho A \vee (B \wedge C)$. Use the Disjunction Introduction Rule to get $A \vee B, A \vdash_\rho A \vee (B \wedge C)$. Use Proposition 9.6(ii) again to get $A \vee B, A \vee C \vdash_\rho A \vee (B \wedge C)$. Finally, use Proposition 7.5(ii) to get the conclusion. □

RULE 8. The *Meta-Disjunction Rule* is an algorithm that implements the rule diagram

$$\frac{\begin{array}{c} G \wedge A \overset{\rho}{\Rightarrow} C \\ G \wedge B \overset{\rho}{\Rightarrow} C \end{array}}{G \wedge (A \vee B) \overset{\rho}{\Rightarrow} C.}$$

PROPOSITION 9.9. *If the library ρ contains the Universal, Conjunction, and Disjunction Elimination Rules, then the Meta-Disjunction Rule is ρ-valid.*

Proof. This follows from Proposition 9.6(i). □

PROPOSITION 9.10. *If ρ contains all the above rules, then*

$$A \overset{\rho}{\Rightarrow} C, B \overset{\rho}{\Rightarrow} C \vdash_{\rho} A \vee B \overset{\rho}{\Rightarrow} C.$$

Proof. Let S be the ρ-deductive closure of the two hypotheses. By the Conjunction, Universal, and Transitivity Rules, $\mathcal{T} \wedge A \overset{\rho}{\Rightarrow} C$ and $\mathcal{T} \wedge B \overset{\rho}{\Rightarrow} C$ are in S. By the Meta-Disjunction Rule, $\mathcal{T} \wedge (A \vee B) \overset{\rho}{\Rightarrow} C$ is in S. By the Universal Rule, $\mathcal{T}$ is in S. So, by Corollary 7.10(i), $A \vee B \overset{\rho}{\Rightarrow} \mathcal{T} \wedge (A \vee B)$ is in S. Finally, by the Transitivity Rule, $A \vee B \overset{\rho}{\Rightarrow} C$ is in S. □

PROPOSITION 9.11. *Assume that ρ contains all of the rules defined above. Then $A \overset{\rho}{\Rightarrow} B \vdash_{\rho} C \vee A \overset{\rho}{\Rightarrow} C \vee B$ and $A \overset{\rho}{\Rightarrow} B \vdash_{\rho} A \vee C \overset{\rho}{\Rightarrow} B \vee C$.*

Proof. Let S be the deductive closure of $A \overset{\rho}{\Rightarrow} B$. By the Disjunction Introduction, Universal, and the Transitivity Rules, $A \overset{\rho}{\Rightarrow} C \vee B$ and $C \overset{\rho}{\Rightarrow} C \vee B$ are in S. By Proposition 9.10, $C \vee A \overset{\rho}{\Rightarrow} C \vee B$ is in S. Similarly, $A \vee C \overset{\rho}{\Rightarrow} B \vee C$ is in S. □

10. NEGATION

DEFINITION 10.1. Let A be an algorithmic statement. The statement $\overset{\rho}{\neg} A$ is defined to be $A \overset{\rho}{\Rightarrow} \mathcal{F}$.

PROPOSITION 10.2. *If ρ contains all of the above rules, then*

(i) $A \overset{\rho}{\Rightarrow} B,\ \overset{\rho}{\neg} B \vdash_{\rho} \overset{\rho}{\neg} A$,
(ii) $A \overset{\rho}{\Rightarrow} B \vdash_{\rho} \overset{\rho}{\neg} B \overset{\rho}{\Rightarrow} \overset{\rho}{\neg} A$, *and*
(iii) $A,\ \overset{\rho}{\neg} A \vdash_{\rho} \overset{\rho}{\neg} B$.

Proof.

(i) Use the Transitivity Rule.
(ii) Use Part(i) and Theorem 7.9.
(iii) Use the Meta-Universal Rule to form $B \overset{\rho}{\Rightarrow} A$. Then use Part(i). □

One might expect the law $A, \overset{\rho}{\neg} A \vdash_{\rho} B$ to hold. Unfortunately it often fails. The instability of the corresponding rule will be discussed in Section 14.

PROPOSITION 10.3 (De Morgan). *If ρ contains all of the above rules, then*

(i) $\overset{\rho}{\neg}(A \vee B) \overset{\rho}{\Longleftrightarrow} \overset{\rho}{\neg}A \wedge \overset{\rho}{\neg}B$, *and*
(ii) $\overset{\rho}{\neg}A \vee \overset{\rho}{\neg}B \vdash_{\rho} \overset{\rho}{\neg}(A \wedge B)$.

Proof.

(i) Let S be the ρ-deductive closure of $\overset{\rho}{\neg}(A \vee B)$. In other words, $A \vee B \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S. Use the Disjunction Introduction Rule to get $A \overset{\rho}{\Rightarrow} A \vee B$ and the Universal Rule to show that it is in S. So, $A \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S by the Transitivity Rule. In other words, $\overset{\rho}{\neg}A$ is in S. Likewise, $\overset{\rho}{\neg}B$ is in S. The Conjunction Rule gives that $\overset{\rho}{\neg}A \wedge \overset{\rho}{\neg}B$ is in S. So $\overset{\rho}{\neg}(A \vee B) \vdash_{\rho} \overset{\rho}{\neg}A \wedge \overset{\rho}{\neg}B$.
For the other direction, let S be the deductive closure of $\overset{\rho}{\neg}A \wedge \overset{\rho}{\neg}B$. By the Conjunction Rule $A \overset{\rho}{\Rightarrow} \mathcal{F}$ and $B \overset{\rho}{\Rightarrow} \mathcal{F}$ are in S. By Proposition 9.10, $A \vee B \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S. So $\overset{\rho}{\neg}A \wedge \overset{\rho}{\neg}B \vdash_{\rho} \overset{\rho}{\neg}(A \vee B)$.

(ii) Let S be the ρ-deductive closure of $A \overset{\rho}{\Rightarrow} \mathcal{F}$. Use the Conjunction Rule to get $A \wedge B \overset{\rho}{\Rightarrow} A$ and the Universal Rule to show that it is in S. So, by the Transitivity Rule, $A \wedge B \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S.
Therefore, $\overset{\rho}{\neg}A \vdash_{\rho} \overset{\rho}{\neg}(A \wedge B)$. Similarly, $\overset{\rho}{\neg}B \vdash_{\rho} \overset{\rho}{\neg}(A \wedge B)$. So, by Proposition 9.6(ii), $\overset{\rho}{\neg}A \vee \overset{\rho}{\neg}B \vdash_{\rho} \overset{\rho}{\neg}(A \wedge B)$. □

The problems with the full converse $\overset{\rho}{\neg}(A \wedge B) \overset{\rho}{\Rightarrow} \overset{\rho}{\neg}A \vee \overset{\rho}{\neg}B$ of the second part of De Morgan will be addressed in Theorem 14.12. Part(ii) of the following gives a partial version.

PROPOSITION 10.4. *If ρ contains all the above rules, then*

(i) $\overset{\rho}{\neg}(A \wedge B), B \vdash_{\rho} \overset{\rho}{\neg}A$, *and*
(ii) $\overset{\rho}{\neg}(A \wedge B), B \vee \overset{\rho}{\neg}B \vdash_{\rho} \overset{\rho}{\neg}A \vee \overset{\rho}{\neg}B$.

Proof.

(i) Let S be the ρ-deductive closure of $A \wedge B \overset{\rho}{\Rightarrow} \mathcal{F}$ and B. By Corollary 7.10(i), $A \overset{\rho}{\Rightarrow} B \wedge A$ is in S. By Corollary 7.4(i), $B \wedge A \overset{\rho}{\Rightarrow} A \wedge B$ holds so is in S by the Universal Rule. By applying the Transitivity Rule twice, $A \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S.

(ii) Both $\overset{\rho}{\neg}(A \wedge B), B \vdash_{\rho} \overset{\rho}{\neg}A \vee \overset{\rho}{\neg}B$ and $\overset{\rho}{\neg}(A \wedge B), \overset{\rho}{\neg}B \vdash_{\rho} \overset{\rho}{\neg}A \vee \overset{\rho}{\neg}B$ hold. The first follows by Part(i) and the Disjunction Introduction Rule. The second is a consequence of the Disjunction Introduction Rule. So by Proposition 9.6(ii) the conclusion holds. □

One might expect the law $A \vee B,\ \overset{\rho}{\neg} B \vdash_{\rho} A$ to hold. Problems with this law will be discussed in Section 14. A partial version is given by the following.

PROPOSITION 10.5. *If ρ contains the above rules, then*

$$\overset{\rho}{\neg} A \vee B,\ \overset{\rho}{\neg} B \vdash_{\rho} \overset{\rho}{\neg} A.$$

Proof. Corollary 4.15 gives $\overset{\rho}{\neg} B,\ \overset{\rho}{\neg} A \vdash_{\rho} \overset{\rho}{\neg} A$. Proposition 10.2(iii) gives $\overset{\rho}{\neg} B,\ B \vdash_{\rho} \overset{\rho}{\neg} A$. Finally, Proposition 9.6(ii) gives $\overset{\rho}{\neg} B,\ \overset{\rho}{\neg} A \vee B \vdash_{\rho} \overset{\rho}{\neg} A$. □

The proofs of the propositions above are not contingent on any special properties of the statement $\mathcal{F}$ itself: similar results can be derived if $\overset{\rho}{\neg} U$ is systematically replaced with $U \overset{\rho}{\Rightarrow} E$ where E is *any* fixed statement. The following proposition, however, uses a property specific to $\mathcal{F}$: if ρ contains the Elimination of Case Rule defined below, then $\mathcal{F} \overset{\rho}{\Rightarrow} B$ holds for any B.

PROPOSITION 10.6. *Assume that $\mathcal{F} \overset{\rho}{\Rightarrow} B$ holds for any B and that ρ contains all the above rules.*

(i) *If $\overset{\rho}{\neg} A$ then $A \overset{\rho}{\Rightarrow} B$.*
(ii) $\overset{\rho}{\neg} A \vdash_{\rho} A \overset{\rho}{\Rightarrow} B$.
(iii) $A,\ \overset{\rho}{\neg} A \vdash_{\rho} B \overset{\rho}{\Rightarrow} C$.
(iv) *If $\overset{\rho}{\neg} A$ then $A \vee B \vdash_{\rho} B$.*
(iv) $\overset{\rho}{\neg} A \vdash_{\rho} A \vee B \overset{\rho}{\Rightarrow} B$.
(vi) $\overset{\rho}{\neg} A \vee B \vdash_{\rho} A \overset{\rho}{\Rightarrow} B$.

Proof.

(i) By assumption, $A \overset{\rho}{\Rightarrow} \mathcal{F}$ and $\mathcal{F} \overset{\rho}{\Rightarrow} B$. So, the conclusion follows from Proposition 4.18(ii).
(ii) Let S be the ρ-deductive closure of $A \overset{\rho}{\Rightarrow} \mathcal{F}$. Since $\mathcal{F} \overset{\rho}{\Rightarrow} B$ holds, it is in S by the Universal Rule. So, by the Transitivity Rule, $A \overset{\rho}{\Rightarrow} B$ is in S.
(iii) Let S be the ρ-deductive closure of A and $\overset{\rho}{\neg} A$. Use Part(ii) to get $A \overset{\rho}{\Rightarrow} C$ in S. By the Meta-Universal Rule, $B \overset{\rho}{\Rightarrow} A$ is in S. So, by the Transitivity Rule, $B \overset{\rho}{\Rightarrow} C$ is in S.
(iv) Assume $\overset{\rho}{\neg} A$. So by Part(i), $A \vdash_{\rho} B$. Since $B \vdash_{\rho} B$, the conclusion follows from Proposition 9.6(ii).

(v) Let S be the ρ-deductive closure of $\overset{\rho}{\neg}A$. By Part(ii), $A \overset{\rho}{\Rightarrow} B$ is in S. By the Universal Rule, $B \overset{\rho}{\Rightarrow} B$ is in S. By Proposition 9.10, $A \vee B \overset{\rho}{\Rightarrow} B$ is in S.

(vi) By Part(ii) above, $\overset{\rho}{\neg}A \vdash_\rho A \overset{\rho}{\Rightarrow} B$. By the Meta-Universal Rule, $B \vdash_\rho A \overset{\rho}{\Rightarrow} B$. The conclusion follows from Proposition 9.6(ii). □

Part(iii) above is a weak version of the ideal law $A, \overset{\rho}{\neg}A \vdash_\rho B$. Part(iv) and Part(v) are closely related to the ideal law $A \vee B, \overset{\rho}{\neg}A \vdash_\rho B$. And the converse $A \overset{\rho}{\Rightarrow} B \vdash_\rho \overset{\rho}{\neg}A \vee B$ of Part(vi) is another ideal law. The instability of the corresponding rules is addressed in Section 14.

11. Strong Negation

DEFINITION 11.1. The algorithm S-NEG expects as input an algorithmic statement $[\alpha, u, v]$. It runs α as a subprocess with input u. If this subprocess halts with output v, then S-NEG outputs 0. If the subprocess halts with output other than v, then S-NEG outputs 1. Otherwise S-NEG does not halt.

Let A be an algorithmic statement. Then the *strong negation* of A, denoted by $-A$, is the algorithmic statement $[\text{S-NEG}, A, 1]$. Note that $-A$ is true if and only if A is directly false, and $-A$ is directly false if and only if A is true. In particular, $-\mathcal{F}$ is true and $-\mathcal{T}$ is directly false.

RULE 9. The *Elimination of Case Rule* is an algorithm that implements the rule diagram

$$\begin{array}{c} A \vee B \\ -A \\ \hline B \end{array}.$$

PROPOSITION 11.2. *The above rule is ρ-valid for all libraries ρ.*

PROPOSITION 11.3. *If ρ contains all the above rules then*

(i) $A, -A \vdash_\rho B$,
(ii) $\mathcal{F} \vdash_\rho B$,
(iii) $-A \vdash_\rho \overset{\rho}{\neg}A$, *and*
(iv) $\mathcal{F} \vee A \overset{\rho}{\Longleftrightarrow} A$ *and* $\mathcal{F} \wedge A \overset{\rho}{\Longleftrightarrow} \mathcal{F}$.

Proof.

(i) Let S be the ρ-deductive closure of A and $-A$. By the Disjunction Introduction Rule, $A \vee B$ is in S. So, by the Elimination of Case Rule, B is in S.
(ii) By the Universal Rule, $\mathcal{F} \vdash_{\rho} -\mathcal{F}$. By Part(i), $\mathcal{F}, -\mathcal{F} \vdash_{\rho} B$. The conclusion follows by Corollary 4.15(iii).
(iii) By Part(i), $-A, A \vdash_{\rho} \mathcal{F}$. By Theorem 7.9, $-A \vdash_{\rho} A \overset{\rho}{\Rightarrow} \mathcal{F}$.
(iv) The first biconditional follows from the Disjunction Introduction Rule, Part(ii), and Proposition 9.6(ii). The second follows from the Conjunction Rule, Part(ii), and Proposition 7.5(i). □

RULE 10. The *Double Negation Rule* is an algorithm that simultaneously implements the following rule diagrams:

$$\frac{A}{- - A} \qquad \frac{- - A}{A}\ .$$

RULE 11. The *Strong De Morgan Rule* is an algorithm that simultaneously implements the following rule diagrams:

$$\frac{-(A \vee B)}{-A \wedge -B} \quad \frac{-A \wedge -B}{-(A \vee B)} \quad \frac{-(A \wedge B)}{-A \vee -B} \quad \frac{-A \vee -B}{-(A \wedge B)}.$$

PROPOSITION 11.4. *The Double Negation and Strong De Morgan Rules are ρ-valid for all libraries ρ.*

PROPOSITION 11.5. *If ρ contains the Double Negation and Strong De Morgan Rules, then*

(i) $A \overset{\rho}{\Longleftrightarrow} -A$,
(ii) $-(A \vee B) \overset{\rho}{\Longleftrightarrow} -A \wedge -B$, *and* $-(A \wedge B) \overset{\rho}{\Longleftrightarrow} -A \vee -B$.

12. Material Conditional

DEFINITION 12.1. Define the *material conditional* $A \to B$ to be $-A \vee B$. Define $\mathcal{H}(A)$ to be $-A \vee A$.

Note that $\mathcal{H}(A)$ is $A \to A$. Also note that the statement $\mathcal{H}(A)$ is true if and only if the process associated with A halts.

PROPOSITION 12.2. *If ρ contains all the above rules, then*

$$A \rightarrow B \vdash_{\rho} A \stackrel{\rho}{\Rightarrow} B.$$

Proof. Use Proposition 11.3(i) and Theorem 7.9 to get $-A \vdash_{\rho} A \stackrel{\rho}{\Rightarrow} B$. By the Meta-Universal rule, $B \vdash_{\rho} A \stackrel{\rho}{\Rightarrow} B$. Finally, use Proposition 9.6(ii). □

The material conditional $\rightarrow$ has many of the properties one would expect. Indeed, some of its properties are stronger than those of the connective $\stackrel{\rho}{\Rightarrow}$. The connective $\rightarrow$ has, however, several striking weaknesses. The following are true for the material conditional in classical logic:

$$A \rightarrow A,\ A \rightarrow A \vee B,\ A \wedge B \rightarrow A,\ A \rightarrow (B \rightarrow A),\ (A \rightarrow B) \wedge (B \rightarrow C) \rightarrow (A \rightarrow C),$$
$$A \wedge (A \rightarrow B) \rightarrow B, \text{ and } (A \vee B) \wedge (A \rightarrow C) \wedge (B \rightarrow C) \rightarrow C.$$

But if A, B, and C are chosen so that $\mathcal{H}(A)$, $\mathcal{H}(B)$ and $\mathcal{H}(C)$ are false, then these statements are all false for the material conditional of Definition 12.1.

Some such tautologies of classical logic can, however, be interpreted to form corresponding laws of algorithmic logic containing the connective $\stackrel{\rho}{\Rightarrow}$ (or equivalently $\vdash_{\rho}$) or containing a mixture of both $\rightarrow$ and $\stackrel{\rho}{\Rightarrow}$ (or $\vdash_{\rho}$). For example, $A \stackrel{\rho}{\Rightarrow} A$, $A \stackrel{\rho}{\Rightarrow} A \vee B$, and $A \wedge B \stackrel{\rho}{\Rightarrow} A$ hold in general for libraries ρ containing all the above rules. The following give further examples (Parts(i) and (iv) – (vi) correspond directly to the remaining tautologies above).

PROPOSITION 12.3. *If ρ contains all the above rules, then*

(i) $A \vdash_{\rho} B \rightarrow A$,
(ii) $A \rightarrow \mathcal{F} \stackrel{\rho}{\Longleftrightarrow} -A$,
(iii) $A \rightarrow B \stackrel{\rho}{\Longleftrightarrow} -B \rightarrow -A$,
(iv) $A \rightarrow B, B \rightarrow C \vdash_{\rho} A \rightarrow C$,
(v) $A, A \rightarrow B \vdash_{\rho} B$
(vi) $A \vee B, A \rightarrow C, B \rightarrow C \vdash_{\rho} C$, *and*
(vii) *if* $\Gamma \vdash_{\rho} A \rightarrow B$ *then* $\Gamma, A \vdash_{\rho} B$.

Proof.

(i) Use the Disjunction Introduction Rule.
(ii) One direction follows from the Disjunction Introduction Rule. The other direction uses Proposition 11.3(ii) and Proposition 9.6(ii).

(iii) This follows from Proposition 11.5(i), Proposition 9.11, Proposition 9.7(iii), and Proposition 8.2(iii).
(iv) Use the Disjunction Introduction Rule (twice), Proposition 11.3(i), and Proposition 9.6(ii) (twice).
(v) Use Proposition 11.3(i) to get $A, -A \vdash_\rho B$. Since $A, B \vdash_\rho B$, the result follows from Proposition 9.6(ii).
(vi) Use Part(v) to get $A{\to}C, B{\to}C, A \vdash_\rho C$ and $A{\to}C, B{\to}C, B \vdash_\rho C$. Then the result follows from Proposition 9.6(ii).
(vii) This follows from Part(v). □

The last three parts of Proposition 12.3 show that in some ways the material conditional $\to$ is stronger than the deductive conditional $\overset{\rho}{\Rightarrow}$. Section 14 discusses the corresponding rules obtained by replacing $\to$ with $\overset{\rho}{\Rightarrow}$ in the last three parts of Proposition 12.3.

The converse of Proposition 12.3(vii) does not hold. Choose A equal to B where $\mathcal{H}(A)$ is false and ρ is a valid library. Then $A \vdash_\rho B$ is true, but $\vdash_\rho A{\to}B$ is false. Contrast this with Theorem 7.9. This illustrates a sense in which $\to$ is weaker than $\overset{\rho}{\Rightarrow}$.

Suppose $\mathcal{H}(A)$ and that ρ is valid. Then $\overset{\rho}{\dashv} A$ if and only if $-A$. Similarly, under these conditions, $A \overset{\rho}{\Rightarrow} B$ if and only if $A{\to}B$. The following proposition shows what can be done with a halting assumption but without assuming that ρ is valid.

PROPOSITION 12.4. *If ρ contains all the above rules, then*

(i) *if* $\overset{\rho}{\dashv} A$ *then* $\mathcal{H}(A) \vdash_\rho -A$,
(ii) $\overset{\rho}{\dashv} A \vdash_\rho \mathcal{H}(A) \overset{\rho}{\Rightarrow} -A$,
(iii) *if* $\Gamma, A \vdash_\rho B$ *then* $\Gamma, \mathcal{H}(A) \vdash_\rho A{\to}B$,
(iv) *if* $A \overset{\rho}{\Rightarrow} B$ *then* $\mathcal{H}(A) \vdash_\rho A{\to}B$, *and*
(v) $A \overset{\rho}{\Rightarrow} B \vdash_\rho \mathcal{H}(A) \overset{\rho}{\Rightarrow} (A{\to}B)$.

Proof.

(i) By assumption $A \overset{\rho}{\Rightarrow} \mathcal{F}$, and $\mathcal{F} \overset{\rho}{\Rightarrow} -A$ by Proposition 11.3(ii), so $A \vdash_\rho -A$. This together with $-A \vdash_\rho -A$ gives the result by Proposition 9.6(ii).
(ii) This follows from the Universal Rule, Proposition 11.3(ii), the Transitivity rule, and Proposition 9.10.
(iii) By the Disjunction Introduction Rule, $\Gamma, -A \vdash_\rho A{\to}B$. We have $\Gamma, A \vdash_\rho A{\to}B$ by assumption and the Disjunction Introduction Rule. The result follows from Proposition 9.6(ii).
(iv) This follows from Part(iii).

(v) This follows from the Disjunction Introduction, Universal, and Transitivity Rules, and Proposition 9.10. □

We mentioned above several tautologies of classical logic that do not hold in general for the algorithmic material conditional. When restricted to algorithmic statements that are true or directly false, however, the algorithmic material conditional can be expected to behave precisely as the classical material conditional. The following corollary illustrates this phenomenon.

COROLLARY 12.5. *If ρ contain all the above rules, then*

(i) $\mathcal{H}(A) \vdash_\rho A \rightarrow A$,
(ii) $\mathcal{H}(A) \vdash_\rho A \rightarrow A \vee B$, *and*
(iii) $\mathcal{H}(A), B \vdash_\rho A \rightarrow A \wedge B$.

Proof. These follow directly from Proposition 12.4(iv) and (iii). □

13. Stable Base

The 11 rules developed above are clearly not a complete collection of rules for algorithmic logic. Indeed, as will be seen in Proposition 13.7, one can never have a complete library of rules for algorithmic logic. Rather, the rules discussed so far provides a convenient stable base on which to build more elaborate stable libraries.

DEFINITION 13.1. A *base* is a set $\mathcal{B}$ of rules. We require that a base be finite, or at least arises as the set of terms of a library. A *$\mathcal{B}$-library* is a library containing all the rules of the base $\mathcal{B}$. A $\mathcal{B}$-library ρ is said to be *valid outside* $\mathcal{B}$ if every rule in ρ which is not in $\mathcal{B}$ is ρ-valid. A base $\mathcal{B}$ is *stable* if every $\mathcal{B}$-library that is valid outside of $\mathcal{B}$ is itself valid.

Let $\mathcal{B}_0$ be the set containing Rules 1 to 11 above.

THEOREM 13.2. *The set $\mathcal{B}_0$ is a stable base.*

Proof. Let ρ be a $\mathcal{B}_0$-library that is valid outside $\mathcal{B}_0$. We need to show that ρ is valid.

Rules 1, 2, 4, 6, and 9 are ρ-valid by Propositions 5.1, 6.1, 7.2, 9.2, 11.2, respectively. Rules 10 and 11 are ρ-valid by Proposition 11.4. Rule 3 is ρ-valid by Proposition 6.4 since $\mathcal{B}$ contains the Universal Rule. Rule 5 is ρ-valid by Proposition 7.6 since $\mathcal{B}$ contains the Conjunction Rule. Rule 8 is ρ-valid by Proposition 9.9 since $\mathcal{B}$ contains the Disjunction Elimination, Universal, and Conjunction Rules. Finally, Theorem 9.5 takes care of Rule 7 and shows that ρ is valid. □

DEFINITION 13.3. Let $\mathcal{B}$ be a base. A rule is $\mathcal{B}$-safe if it is ρ-valid for all $\mathcal{B}$-libraries ρ. A *stable extension* $\mathcal{B}'$ of $\mathcal{B}$ is a base containing $\mathcal{B}$ such that every rule in $\mathcal{B}'$ that is not in $\mathcal{B}$ is $\mathcal{B}$-safe.

PROPOSITION 13.4. *A stable extension of a stable base is a stable base.*

Proof. Let $\mathcal{B}$ be a stable base and $\mathcal{B}'$ a stable extension of $\mathcal{B}$. Suppose ρ is a $\mathcal{B}'$-library valid outside of $\mathcal{B}'$. We must show that ρ is valid.

First we show that ρ is actually valid outside $\mathcal{B}$. To that end, let ρ_k be outside $\mathcal{B}$. If ρ_k happens to be in $\mathcal{B}'$ then it is $\mathcal{B}$-safe by the definition of stable extension. In particular, ρ_k is ρ-valid. If ρ_k is outside $\mathcal{B}'$ then it is ρ-valid simply because ρ is valid outside of $\mathcal{B}'$. Thus ρ is valid outside $\mathcal{B}$.

Since $\mathcal{B}$ is stable, and since ρ is valid outside $\mathcal{B}$, the library ρ is valid. □

PROPOSITION 13.5. *If the rules of a library forms a stable base, then the library is valid. Thus, if the rules of a library form a stable extension of $\mathcal{B}_0$, then the library is valid.*

DEFINITION 13.6. Let ρ_1 and ρ_2 be libraries. Then ρ_2 is *stronger than* ρ_1 if $A \stackrel{\rho_1}{\Rightarrow} B$ implies $A \stackrel{\rho_2}{\Rightarrow} B$ for all algorithmic statements A and B. A library ρ_2 is *strictly stronger than* ρ_1 if (i) ρ_2 is stronger than ρ_1, and (ii) there exists A and B such that $A \stackrel{\rho_2}{\Rightarrow} B$ is true but $A \stackrel{\rho_1}{\Rightarrow} B$ is false.

Algorithmic logic is complete in the sense that, for any library ρ containing the Universal Rule, if A is true then $\textsc{prove}_\rho(A)$ is true. But there is also a sense in which the logic is inherently incomplete.

PROPOSITION 13.7. *For every valid library ρ_1 there is a strictly stronger valid library ρ_2. If the set of rules in ρ_1 form a stable base $\mathcal{B}$, then ρ_2 can be taken to be a library whose rules form a stable $\mathcal{B}$-extension.*

Proof. Let $\textsc{mp}(\rho_1)$ be the algorithm implementing

$$\frac{\begin{array}{c} A \stackrel{\rho_1}{\Rightarrow} B \\ A \end{array}}{B}$$

This is a ρ-valid rule for any library ρ since ρ_1 is valid.[8] There is no algorithm that decides whether a statement is false. Thus there is a false algorithmic statement C such that $C \stackrel{\rho_1}{\Rightarrow} \mathcal{F}$ is false. Let $\textsc{deny}(C)$ be the algorithm implementing the diagram $\frac{C}{\mathcal{F}}$. The rule $\textsc{deny}(C)$ is ρ-valid for any library ρ since C is false.

Let ρ_2 be the library containing $\textsc{mp}(\rho_1)$, $\textsc{deny}(C)$, and the Universal Rule. Observe that (i) ρ_2 is valid, (ii) if $A \stackrel{\rho_1}{\Rightarrow} B$ then $A \stackrel{\rho_1}{\Rightarrow} B$, and (iii) C$\stackrel{\rho_1}{\Rightarrow}$

$\mathcal{F}$ is false but $C \overset{\rho_1}{\Rightarrow} \mathcal{F}$ is true. To see (ii), let S be the ρ_2-deductive closure of A. By the Universal Rule, $A \overset{\rho_2}{\Rightarrow} B$ is in S. By $\textsc{mp}(\rho_1)$, B is in S. So ρ_2 is valid and strictly stronger than ρ_1.

Now suppose that the rules of ρ_1 form a stable base $\mathcal{B}$. Let ρ_2 contain all the rules of ρ_1, $\textsc{mp}(\rho_1)$, $\textsc{deny}(C)$, and the Universal Rule. The new rules are $\mathcal{B}$-safe, so the rules of ρ_2 form a stable valid extension of $\mathcal{B}$. Finally, by an argument similar to the one above, ρ_2 is strictly stronger than ρ_1. $\square$

14. Paradoxical Rules

A *paradoxical rule* is an algorithmic counterpart of a traditional rule of logic that cannot be in any stable base.[9] In this section we will show that the following are paradoxical rules:

$$P_1: \begin{array}{c} A \\ \overset{\rho}{\neg} A \\ \hline \mathcal{F} \end{array} \qquad P_2: \begin{array}{c} A \\ \overset{\rho}{\neg} A \\ \hline B \end{array} \qquad P_3: \begin{array}{c} A \overset{\rho}{\Rightarrow} B \\ A \\ \hline B \end{array} \qquad P_4: \begin{array}{c} A \overset{\rho}{\Rightarrow} C \\ B \overset{\rho}{\Rightarrow} C \\ A \vee B \\ \hline C \end{array} \qquad P_5: \begin{array}{c} \overset{\rho}{\neg}\overset{\rho}{\neg} A \\ \hline A \end{array}$$

$$P_6: \begin{array}{c} A \vee B \\ \overset{\rho}{\neg} A \\ \hline B \end{array} \qquad P_7: \begin{array}{c} \overset{\rho}{\neg} A \vee B \\ A \\ \hline B \end{array} \qquad P_8: \begin{array}{c} \emptyset \\ \hline A \vee \overset{\rho}{\neg} A \end{array} \qquad P_9: \begin{array}{c} \emptyset \\ \hline \overset{\rho}{\neg} A \vee \overset{\rho}{\neg}\overset{\rho}{\neg} A \end{array} \qquad P_{10}: \begin{array}{c} A \overset{\rho}{\Rightarrow} B \\ \hline \overset{\rho}{\neg} A \vee B \end{array}$$

$$P_{11}: \begin{array}{c} \overset{\rho}{\neg}(A \wedge B) \\ \hline \overset{\rho}{\neg} A \vee \overset{\rho}{\neg} B \end{array} \qquad P_{12}: \begin{array}{c} A \\ \hline \overset{\rho}{\neg}\overset{\rho}{\neg} A \end{array} \qquad P_{13}: \begin{array}{c} \textsc{prove}_\rho(A) \\ \hline A \end{array} \qquad P_{14}: \begin{array}{c} \textsc{prove}_\rho(\textsc{prove}_\rho(A)) \\ \hline \textsc{prove}_\rho(A) \end{array}$$

Given the expected input $[H, \rho, m]$, all of the rules above use ρ, but only Rules P_2, P_8, and P_9 use the resource integer m in their implementation. The symbol $\emptyset$ in Rules P_8 and P_9 indicates that no premises in H are required. Clearly, some of the paradoxical rules above are interrelated.

Rules P_1, P_3, P_4, P_6, P_7, P_{13}, and P_{14} have the remarkable property of being ρ-valid for any valid library ρ but, due to their instability, not being in any sufficiently rich valid ρ. Rule P_2 has a similar status, at least for any $\mathcal{B}_0$-library ρ.

REMARK. As one might expect, many of these correspond to rules that have aroused suspicion in the past and have been excluded from weaker logics such as intuitionistic or minimal logic. The long list of paradoxical rules to be avoided in algorithmic logic might make algorithmic logic seem weak. However, in algorithmic logic one always has the option of going to a stronger library ρ, often compensating for not having the above rules.

LEMMA 14.1. *Every stable base $\mathcal{B}$ has a stable extension $\mathcal{B}'$ with the following property: For every $\mathcal{B}'$-library ρ there is an algorithmic statement Q_ρ such that $Q_\rho \overset{\rho}{\Longleftrightarrow} \overset{\rho}{\dashv} Q_\rho$.*

Proof. The proof requires an algorithm CURRY that expects as input a list $[\alpha, \rho]$ where α is an algorithm. If the algorithmic statement $\overset{\rho}{\dashv} \big[\alpha, [\alpha, \rho], 1\big]$ is true, then CURRY outputs 1. Otherwise CURRY does not halt.[10] Observe that if α is an algorithm, then $\big[\text{CURRY}, [\alpha, \rho], 1\big]$ if and only if $\overset{\rho}{\dashv} \big[\alpha, [\alpha, \rho], 1\big]$.

Let β be the rule that simultaneously implements the two rule diagrams:

$$\frac{\big[\text{CURRY}, [\alpha, \rho], 1\big]}{\overset{\rho}{\dashv} \big[\alpha, [\alpha, \rho], 1\big]} \qquad \frac{\overset{\rho}{\dashv} \big[\alpha, [\alpha, \rho], 1\big]}{\big[\text{CURRY}, [\alpha, \rho], 1\big]}.$$

More specifically, assuming an input of the expected form $[H, \rho, m]$, the rule β looks for all statements of the form of the first line of either of the above diagrams, where α is required to be an algorithm. For each such statement it finds, it appends the appropriate statement to H.

Clearly β is $\mathcal{B}$-safe where $\mathcal{B}$ is the given stable base. Let $\mathcal{B}'$ be the stable extension of $\mathcal{B}$ obtained by simply adding the rule β to $\mathcal{B}$. Given a $\mathcal{B}'$-library ρ, let Q_ρ be $\big[\text{CURRY}, [\text{CURRY}, \rho], 1\big]$. So $Q_\rho \overset{\rho}{\Longleftrightarrow} \overset{\rho}{\dashv} Q_\rho$ since ρ contains β. □

While the rule β used in the above proof is not a rule of elementary logic, and may thus seem *ad hoc*, it is a consequence of general, more natural rules concerning the basic properties of algorithms. This is discussed in [2].

THEOREM 14.2. *There is no stable base $\mathcal{B}$ such that the law $A, \overset{\rho}{\dashv} A \vdash_\rho \mathcal{F}$ holds for all valid $\mathcal{B}$-libraries ρ.*

Proof. Suppose that there is such a $\mathcal{B}$, and let $\mathcal{B}'$ be as in Lemma 14.1. Let ρ be the library consisting of the rules of $\mathcal{B}'$. The validity of ρ follows from Proposition 13.5. By Lemma 14.1, there is a statement Q_ρ such that $Q_\rho \overset{\rho}{\Longleftrightarrow} \overset{\rho}{\dashv} Q_\rho$. By validity, Q_ρ holds if and only if $\overset{\rho}{\dashv} Q_\rho$ holds.

Let S be the ρ-deductive closure of Q_ρ. Since, $Q_\rho \vdash_\rho \overset{\rho}{\dashv} Q_\rho$, the set S contains $\overset{\rho}{\dashv} Q_\rho$. By assumption $Q_\rho, \overset{\rho}{\dashv} Q_\rho \vdash_\rho \mathcal{F}$ holds, so S contains $\mathcal{F}$. Thus $Q_\rho \overset{\rho}{\Rightarrow} \mathcal{F}$ holds; that is, $\overset{\rho}{\dashv} Q_\rho$ is true. As mentioned above, this implies that Q_ρ is true. Since Q_ρ, $\overset{\rho}{\neg} Q_\rho \vdash_\rho \mathcal{F}$ and since ρ is valid, $\mathcal{F}$ is true. $\square$

COROLLARY 14.3. *No stable base contains Rule P_1 or Rule P_2 (defined at the beginning of this section).*

COROLLARY 14.4. *If $\mathcal{B}$ is a stable base, then the assertion that*

$$\Gamma \vdash_\rho A \overset{\rho}{\Rightarrow} B \ \ \textit{implies} \ \ \Gamma, A \vdash_\rho B$$

fails for some valid $\mathcal{B}$-library ρ.

Proof. Let ρ be a valid $\mathcal{B}$-library for which the assertion holds. By Corollary 4.15(ii), $\overset{\rho}{\neg} A \vdash_\rho \overset{\rho}{\neg} A$. In other words, $\overset{\rho}{\neg} A \vdash_\rho A \overset{\rho}{\Rightarrow} \mathcal{F}$. So $\overset{\rho}{\neg} A$, $A \vdash_\rho \mathcal{F}$ by the assertion. By Theorem 14.2 this cannot hold for all such ρ. $\square$

COROLLARY 14.5. *There is no stable base $\mathcal{B}$ such that $A \overset{\rho}{\Rightarrow} B, A \vdash_\rho B$ holds for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains Rule P_3.*

Proof. Suppose otherwise. If ρ is a valid $\mathcal{B}$-library, then $A, A \overset{\rho}{\Rightarrow} \mathcal{F} \vdash_\rho \mathcal{F}$ for all A. In other words, A, $\overset{\rho}{\neg} A \vdash_\rho \mathcal{F}$ holds, contradicting Theorem 14.2. $\square$

COROLLARY 14.6 *There is no stable base $\mathcal{B}$ such that the law*

$$A \overset{\rho}{\Rightarrow} C,\ B \overset{\rho}{\Rightarrow} C,\ A \vee B \ \vdash_\rho \ C$$

holds for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains Rule P_4.

Proof. Suppose that there is such a $\mathcal{B}$. The Disjunction Introduction Rule is $\mathcal{B}$-safe, so the base $\mathcal{B}'$ that results from adding this rule to $\mathcal{B}$ is also stable. Let ρ be a valid $\mathcal{B}'$-library. Let S be the deductive closure of A and $\overset{\rho}{\neg} A$. By the Disjunction Introduction Rule, $A \vee A$ is in S. Since $A \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S, so is $\mathcal{F}$. Thus A, $\overset{\rho}{\neg} A \vdash_\rho \mathcal{F}$ for all A and all such ρ, contradicting Theorem 14.2 for the stable base $\mathcal{B}'$. $\square$

COROLLARY 14.7. *There is no stable base $\mathcal{B}$ such that the law $\overset{\rho}{\neg}\overset{\rho}{\neg} A \vdash_\rho A$ holds for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains Rule P_5.*

Proof. Suppose that there is such a stable base $\mathcal{B}$. The Universal and Transitivity Rules are $\mathcal{B}$-safe, so the base $\mathcal{B}'$ that results from adding these rules to $\mathcal{B}$ is also stable. The Meta-Universal Rule is $\mathcal{B}'$-safe since $\mathcal{B}'$ contains the Universal Rule, so the base $\mathcal{B}''$ that results from adding the Meta-Universal Rule to $\mathcal{B}'$ is also stable.

Let ρ be a valid $\mathcal{B}''$-library and A a statement. Let S be the deductive closure of A and $\overset{\rho}{\neg}A$. By the Meta-Universal rule $\overset{\rho}{\neg}\mathcal{F}\overset{\rho}{\Rightarrow}A$ is in S. Since $\overset{\rho}{\neg}A$ is $A\overset{\rho}{\Rightarrow}\mathcal{F}$, which is in S, $\overset{\rho}{\neg}\mathcal{F}\overset{\rho}{\Rightarrow}\mathcal{F}$ is in S by the Transitivity Rule. In other words, $\overset{\rho}{\neg}\overset{\rho}{\neg}\mathcal{F}$ is in S. Thus $\mathcal{F}$ is in S by hypothesis. So $A, \overset{\rho}{\neg}A \vdash_{\rho} \mathcal{F}$, contradicting Theorem 14.2 for the base $\mathcal{B}''$. □

COROLLARY 14.8. *There is no stable base $\mathcal{B}$ where the law $A \vee B, \overset{\rho}{\neg}A \vdash_{\rho} B$ holds for all $\mathcal{B}$-libraries ρ. Likewise, there is no stable base $\mathcal{B}$ where the law $\overset{\rho}{\neg}A \vee B, A \vdash_{\rho} B$ holds for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains either Rule P_6 or Rule P_7.*

Proof. Suppose that there is a $\mathcal{B}$ where the first of these laws holds. The Disjunction Introduction Rule is $\mathcal{B}$-safe, so the extension $\mathcal{B}'$ obtained by adding this Rule to $\mathcal{B}$ is stable. Let ρ be any valid $\mathcal{B}'$-library.

Let S be the deductive closure of A and $\overset{\rho}{\neg}A$. By the Disjunction Introduction Rule, $A \vee \mathcal{F}$ is in S. By hypothesis, $\mathcal{F}$ is in S. We have established that $\overset{\rho}{\neg}A, A \vdash_{\rho} \mathcal{F}$ holds for every statement A and valid $\mathcal{B}'$-library ρ, contradicting Theorem 14.2.

The second part of the theorem follows by a similar argument. □

LEMMA 14.9. *Suppose A is an algorithmic statement where $A \overset{\rho}{\Longleftrightarrow} \overset{\rho}{\neg}A$ with ρ a valid library. Then A, $\overset{\rho}{\neg}A$, and $\overset{\rho}{\neg}\overset{\rho}{\neg}A$ are all false.*

Proof. Suppose A is true. By hypothesis, $A\overset{\rho}{\Rightarrow}\overset{\rho}{\neg}A$. So, by Proposition 41.8 (iii) and the validity of ρ, the statement $\overset{\rho}{\neg}A$ holds. Thus A and $A\overset{\rho}{\Rightarrow}\mathcal{F}$ hold. Again, by Proposition 4.18 (iii), $\mathcal{F}$ is true.

Suppose $\overset{\rho}{\neg}A$. By hypothesis, $\overset{\rho}{\neg}A\overset{\rho}{\Rightarrow}A$. So A is true, contradicting the above.

Suppose $\overset{\rho}{\neg}\overset{\rho}{\neg}A$; in other words, $\overset{\rho}{\neg}A\overset{\rho}{\Rightarrow}\mathcal{F}$. By hypothesis, $A\overset{\rho}{\Rightarrow}\overset{\rho}{\neg}A$. By Proposition 4.18 (ii), $A\overset{\rho}{\Rightarrow}\mathcal{F}$. In other words, $\overset{\rho}{\neg}A$ which contradicts the above. □

COROLLARY 14.10. *Let $\mathcal{B}$ be a stable base. There is a valid $\mathcal{B}$-library ρ and an algorithmic statement Q_ρ such that $Q_\rho \vee \overset{\rho}{\neg}Q_\rho$ and $\overset{\rho}{\neg}Q_\rho \vee \overset{\rho}{\neg}\overset{\rho}{\neg}Q_\rho$ are both false. In particular, Rules P_8 and P_9 are not ρ-valid.*

Therefore, there is no stable base containing Rules P_8 or P_9.

Proof. Let $\mathcal{B}'$ be as in Lemma 14.1. Let ρ be a library consisting of the rules in $\mathcal{B}'$, and let Q_ρ be as in Lemma 14.1. The result follows from Lemma 14.9. □

PROPOSITION 14.11. There is no stable base $\mathcal{B}$ where $A\overset{\rho}{\Rightarrow}B \vdash_{\rho} \overset{\rho}{\neg}A \vee B$ holds for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains Rule P_{10}.

Proof. Suppose that there is such a $\mathcal{B}$. Let ρ be a library consisting of the rules in $\mathcal{B}'$ as defined in Lemma 14.1. Let Q_ρ be as in Lemma 14.1. The library ρ is valid since $\mathcal{B}'$ is a stable base.

By hypothesis $Q_\rho \overset{\rho}{\Rightarrow} \overset{\rho}{\neg} Q_\rho \vdash_\rho \overset{\rho}{\neg} Q_\rho \vee \overset{\rho}{\neg} Q_\rho$, so by Proposition 41.6 and the validity of ρ, the statement $\overset{\rho}{\neg} Q_\rho \vee \overset{\rho}{\neg} Q_\rho$ is true. So $\overset{\rho}{\neg} Q_\rho$ is true contradicting Lemma 14.9. $\square$

THEOREM 14.12. *There is no stable base $\mathcal{B}$ where the law*

$$\overset{\rho_1}{\neg}(A \wedge B) \vdash_\rho \overset{\rho_1}{\neg} A \vee \overset{\rho_1}{\neg} B$$

holds for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains Rule P_{11}.

Proof. Suppose that there is such a stable base $\mathcal{B}$. The rule represented by the diagram $\frac{A \wedge -A}{\mathcal{F}}$ is $\mathcal{B}$-safe. Let $\mathcal{B}'$ be the stable extension obtained by adding this rule to $\mathcal{B}$. Let ρ be a library consisting of the rules in $\mathcal{B}'$. The library ρ is valid since $\mathcal{B}'$ is a stable base. The statement $\overset{\rho}{\neg}(A \wedge -A)$ holds for any A because of the new rule added to the library. So, by hypothesis and the validity of ρ, the statement $\overset{\rho}{\neg} A \vee \overset{\rho}{\neg} -A$ is true for all A.

Let β be an algorithm that expects an algorithm α as input. The algorithm β finds the smallest m such that $\overset{\rho}{\neg}[\alpha, \alpha, 1]$ or $\overset{\rho}{\neg} -[\alpha, \alpha, 1]$ is m-true. There will be such an m since $\overset{\rho}{\neg} A \vee \overset{\rho}{\neg} -A$ holds for all A. If $\overset{\rho}{\neg}[\alpha, \alpha, 1]$ is m-true for this value of m, then β outputs *1*. If $\overset{\rho}{\neg}[\alpha, \alpha, 1]$ is not m-true, but $\overset{\rho}{\neg} -[\alpha, \alpha, 1]$ is m-true for this value of m, then β outputs *0*. The notion of m-true here is as in the definition of the Universal Rule. Observe that if α is an algorithm then β halts for input α.

Let B be the statement $[\beta, \beta, 1]$. If B is true, then $\overset{\rho}{\neg} B$ is true. If $-B$ is true, then $\overset{\rho}{\neg} -B$. Since ρ is valid, it is not possible for a statement A and its negation $\overset{\rho}{\neg} A$ to both be true. So neither B nor $-B$ is true. In other words, β does not halt for input β, a contradiction. $\square$

PROPOSITION 14.13. *Let $\mathcal{B}$ be a stable base. Then there is a valid $\mathcal{B}$-library ρ such that $\overset{\rho}{\neg}\overset{\rho}{\neg}\mathcal{T}$ is false. Furthermore, the law $A \vdash_\rho \overset{\rho}{\neg}\overset{\rho}{\neg} A$ does not hold for all $\mathcal{B}$-libraries ρ. In particular, no stable base contains Rule P_{12}.*

Proof. As in the proof of Corollary 14.7, there is a stable base $\mathcal{B}''$ containing $\mathcal{B}$ together with the Universal, the Meta-Universal, and the Transitivity Rules. Let ρ be any valid $\mathcal{B}''$-library. Suppose, $\overset{\rho}{\neg}\overset{\rho}{\neg}\mathcal{T}$ holds. Thus $\overset{\rho}{\neg}\mathcal{T} \vdash_\rho \mathcal{F}$. Let S be the deductive closure of A and $\overset{\rho}{\neg} A$. By the Meta-Universal Rule, $\mathcal{T} \overset{\rho}{\Rightarrow} A$ is in S. Note that $A \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S, so, by the Transitivity Rule, $\mathcal{T} \overset{\rho}{\Rightarrow} \mathcal{F}$ is in S. In other words, $\overset{\rho}{\neg}\mathcal{T}$ is in S. Since $\overset{\rho}{\neg}\mathcal{T} \vdash_\rho \mathcal{F}$ is true, $\mathcal{F}$ must be in S.

We have established that if $\overset{\rho}{\neg}\overset{\rho}{\neg}\mathcal{T}$ holds then $\overset{\rho}{\neg}A, A \vdash_{\rho} \mathcal{F}$ holds for all A. Therefore, by Theorem 14.2, there must be a valid $\mathcal{B}''$-library ρ such that $\overset{\rho}{\neg}\overset{\rho}{\neg}\mathcal{T}$ is false. The law $A \vdash_{\rho} \overset{\rho}{\neg}\overset{\rho}{\neg}A$ does not hold for such ρ. To see this, consider the case where A is $\mathcal{T}$. □

THEOREM 14.14. *There is no stable base $\mathcal{B}$ where the law*

$$\textsc{prove}_{\rho}(\textsc{prove}_{\rho}(A)) \vdash_{\rho} \textsc{prove}_{\rho}(A)$$

holds for all $\mathcal{B}$-libraries ρ. In particular, there is no stable base $\mathcal{B}$ where $\textsc{prove}_{\rho}(A) \vdash_{\rho} A$ holds for all $\mathcal{B}$-libraries ρ. So no stable base contains Rule P_{13} or Rule P_{14}.

Proof. Suppose otherwise that there is such a stable base $\mathcal{B}$. As in the proof of Corollary 14.7, there is a stable base $\mathcal{B}''$ containing $\mathcal{B}$ together with the Universal, the Meta-Universal, and the Transitivity Rules.

Consider an algorithm β that expects as input $[\alpha, \rho]$ where α is an algorithm. The algorithm β checks the truth of $\big[\alpha, [\alpha, \rho], 1\big] \overset{\rho}{\Rightarrow} \textsc{prove}_{\rho}(\mathcal{F})$. If the statement is true, β outputs 1. Otherwise, β does not halt. Let R_{ρ} be the statement $\big[\beta, [\beta, \rho], 1\big]$. Observe that R_{ρ} is true if and only if $R_{\rho} \overset{\rho}{\Rightarrow} \textsc{prove}_{\rho}(\mathcal{F})$ is true.

The rule implementing

$$\frac{R_{\rho}}{R_{\rho} \overset{\rho}{\Rightarrow} \textsc{prove}_{\rho}(\mathcal{F})}$$

is ρ-valid for all libraries ρ. Let ρ be the library consisting of this rule together with all the rules of $\mathcal{B}''$. The $\mathcal{B}''$-library ρ is valid since $\mathcal{B}''$ is stable.

Let S be the deductive closure of R_{ρ}. So $R_{\rho} \overset{\rho}{\Rightarrow} \textsc{prove}_{\rho}(\mathcal{F})$ is in S. By Proposition 6.5(ii), $\textsc{prove}_{\rho}(\textsc{prove}_{\rho}(\mathcal{F}))$ is in S. Finally, by supposition, $\textsc{prove}_{\rho}(\mathcal{F})$ is also in S. We have shown that $R_{\rho} \overset{\rho}{\Rightarrow} \textsc{prove}_{\rho}(\mathcal{F})$ is true. Therefore, R_{ρ} is true. Since ρ is valid, Proposition 4.18(iii) implies that $\textsc{prove}_{\rho}(\mathcal{F})$ is true. So by Proposition 4.18(iv) and the validity of ρ, $\mathcal{F}$ is true. □

15. Conclusion

In [2] we introduce additional rules to algorithmic logic which do not concern logical connectives as do the rules in the current paper. Instead, these new rules relate to the basic structure of algorithms themselves. These structural rules will lead to a strong internal abstraction principle making algorithmic logic more flexible and powerful.

In particular, for bases $\mathcal{B}$ containing these structural rules, Lemma 14.1 can be strengthened to apply to all $\mathcal{B}$-libraries ρ. Consequently, the main results of Section 14 can be significantly strengthened. More precisely, let $\mathcal{B}_1$ be the stable base consisting of $\mathcal{B}_0$ together with the structural rules of the promised future paper. Many of the results of Section 14 refer to laws which do not hold for all $\mathcal{B}$-libraries. In other words there exists *some* $\mathcal{B}$-library where the law fails. For the base $\mathcal{B}_1$, however, these results can be strengthened to assert that the given law fails for *all* valid $\mathcal{B}_1$-libraries.

In Section 14 above we mention that several of the paradoxical rules are ρ-valid as long as ρ is valid. The other rules, with one exception, cannot be expected to be ρ-valid. More specifically, if ρ is a valid $\mathcal{B}_1$-library, then all the other rules, with the exception of Rule P_5, are not ρ-valid. This can be seen with arguments similar to those of Section 14. Rule P_5 is ρ-valid for such ρ, however, because of the striking fact that $\overset{\rho}{\neg}\overset{\rho}{\neg}A$ is false for *all* A. This fact can be shown with an argument similar to that of Proposition 14.13.[11]

NOTES

[1] Of course, *restricted* versions of T can be defined to apply to functions on a fixed domain.

[2] This is in contrast to *sets* where good conceptual reasons have been given to restrict the comprehension principle.

[3] Recent examples include [3, 6, 11]. Recent examples from the substructural tradition include [4, 8, 12, 14, 15]. See the bibliographies of these works for earlier examples. The articles in [9], especially those by A. Cantini, S. Feferman, H. Field, H. Friedman, H. Sturm, and K. Wehmeier, show the contemporary interest in type-free systems and in strong forms of comprehension and abstraction.

[4] We recommend [3, 5, 10] as interesting introductions to type-free logic. We have found [13] to be a helpful introduction to the substructural tradition.

[5] In this paper *algorithms* will be limited to recursive algorithms. With this restriction, the collection of algorithmic statements can be seen to be in some sense equivalent to the collection of Σ_1-statements in first-order arithmetic.

[6] This definition depends on ρ as well as the particular rule ρ_k.

[7] To see the usual disjunction elimination rule, think of G as $\mathcal{T}$. Allowing general G is important in the proof of Proposition 9.6.

[8] This rule differs essentially from P_3 discussed in Section 14 in that, given input $[H, \rho, m]$, the rule $\textsc{mp}(\rho_1)$ does not use the input ρ, but rather uses the fixed library ρ_1. Rule P_3, on the other hand, does use the input ρ.

[9] The term *paradoxical* is used since many of the arguments related to such rules are akin to those occurring in the Russell and Curry paradoxes.

[10] This algorithm is called Curry due to the resemblance of the proof of Theorem 14.2 to a common version of the Curry Paradox.

[11] We would like to thank our colleagues for many useful discussions, and the referees for several good suggestions including the suggestion to use the term *strong negation* in honor of David Nelson. One referee asked an interesting question concerning the status of $(A \overset{\rho}{\Rightarrow} \overset{\rho}{\neg}B) \vdash_{\rho} (B \overset{\rho}{\Rightarrow} \overset{\rho}{\neg}A)$, a contrapositive law whose analogue

holds in intuitionistic and even minimal logic. For sufficiently rich ρ, the statement $\overset{\rho}{\neg}\overset{\rho}{\neg}\mathcal{T}$ is false (Proposition 14.13). For such valid ρ the law fails: consider the case where A is $\overset{\rho}{\neg}\mathcal{T}$ and B is $\mathcal{T}$.

REFERENCES

1. Aitken, W. and Barrett J. A. (2004): Computer implication and the curry paradox, *J. Philos. Logic* **33**, 631–637.
2. Aitken, W. and Barrett, J. A.: *Abstraction in Algorithmic Logic* (preprint).
3. Cantini, A. (1996): Logical frameworks for truth and abstraction: An axiomatic study, North-Holland. ISBN: 0-444-82306-9.
4. Cantini, A. (2003): The undecidability of Grišin's set theory, *Stud. Log.* **74**(3), 345–368.
5. Feferman, S. (1984): Toward useful type-free theories, *J. Symb. Log.* **49**, 75–111.
6. Field, H. (2004): The consistency of the naïve theory of properties, *Philos. Q.* **54**(214), 78–104.
7. Fitch, F. B. (1969): A method for avoiding the Curry paradox, in N. Rescher (ed.), *Essays in Honor* of Carl G. Hempel, pp. 255–265.
8. Girard, J.-Y. (1998): Light linear logic, *Inform. Comput.* **143**(2), 175–204.
9. Link, G. (ed.) (2004): One hundred years of Russell's paradox: Mathematics, logic, philosophy, de Gruyter. ISBN 3-11-017438-3.
10. Myhill, J. (1984) Paradoxes, *Synthese* **60**, 129–143.
11. Orilia, F. (2000): Property theory and the revision theory of definitions, *J. Symb. Log.* **65**(1), 212–246.
12. Petersen, U. (2000): Logic without contraction as based on inclusion and unrestricted abstraction, *Stud. Log.* **64**(3), 365–403.
13. Restall, G. (1994): On logics without contraction, doctoral dissertation, University of Queensland.
14. Terui, K. (2004) Light affine set theory: A naive set theory of polynomial time, *Stud. Log.* **77**(1), 9–40.
15. Weir, A. (1998): Naïve set theory, paraconsistency and indeterminacy. I., *Log. Anal. (N.S.)* **41**(161–163), 219–266.

WAYNE AITKEN
California State University,
San Marcos, CA 92096,
USA
E-mail: waitken@csusm.edu

JEFFREY A. BARRETT
UC Irvine,
Irvine, CA 92697,
USA
E-mail: jabarret@uci.edu