A Shortest 2-Basis for Boolean Algebra in Terms of the Sheffer Stroke *

Robert Veroff
University of New Mexico, Albuquerque, NM 87131, U.S.A.

Abstract. In this article, we present a short 2-basis for Boolean algebra in terms of the Sheffer stroke and prove that no such 2-basis can be shorter. We also prove that the new 2-basis is unique (for its length) up to applications of commutativity. Our proof of the 2-basis was found by using the method of *proof sketches* and relied on the use of an automated reasoning program.

Keywords: Boolean algebra, Sheffer stroke

1. Introduction

There is an ongoing interest in finding "simple" axiom systems for various algebras and logics, where simplicity is characterized by the number of axioms, by the lengths of the axioms, and by the number of distinct variable symbols appearing in the axioms of the system [8]. These measures often conflict with each other in the sense that we may be able to improve one, but only at the expense of the others.

We have been developing automated reasoning techniques to search for and prove simple axiom systems for Boolean algebra and related algebras and logics in terms of various sets of operators. See, for example, [4] and [5]. In this article, we show that the pair of equations

$$(x \mid y) \mid (x \mid (y \mid z)) \;=\; x \qquad \text{(26a)}$$
$$x \mid y \;=\; y \mid x \qquad \text{(Commutativity)}$$

is a 2-axiom system (2-basis) for Boolean algebra in terms of the Sheffer stroke '|'.[1] We also show that no 2-basis can be shorter and that the new 2-basis is unique (for its length) up to applications of commutativity. This basis is especially simple in that it has a total length of only 18 variable and operator symbols (including equality) and requires only three distinct variable symbols. To the best of our knowledge, this system is strictly shorter than any other previously known 2-basis.

* This work was supported in part by National Science Foundation grant no. CDA-9503064.

[1] An application $x|y$ of the Sheffer stroke typically is interpreted as a **nand** operation, but it also can be interpreted as **nor**.

Our proof of the 2-basis, which was found with the automated reasoning program Otter [2], was the result of a successful application of the method of proof sketches [10]. After presenting the new 2-basis and its properties, we briefly describe the method and the search.

We note that knowledge of the 2-basis presented in this article led directly to the proof of previously unknown *shortest single axioms* for Boolean algebra in terms of the Sheffer stroke [4]. Specifically, the single axioms were first proved by deriving equations 26a and Commutativity.

2. Background

In 1913, Sheffer [7] presented the following 3-basis for Boolean algebra in terms of the Sheffer stroke.

$$(x \mid x) \mid (x \mid x) = x \qquad \text{(Sheffer 1)}$$
$$x \mid (y \mid (y \mid y)) = (x \mid x) \qquad \text{(Sheffer 2)}$$
$$(x \mid (y \mid z)) \mid (x \mid (y \mid z)) = ((y \mid y) \mid x) \mid ((z \mid z) \mid x) \qquad \text{(Sheffer 3)}$$

More recently, a number of equivalent simplifications ("abridgements") of Sheffer's system have been presented. These include, for example, five systems presented by Meredith [6]. The simplest of these five systems is a 2-basis that has a total length of 24 and requires three distinct variable symbols.

$$(x \mid x) \mid (y \mid x) = x \qquad \text{(Meredith 1)}$$
$$x \mid (y \mid (x \mid z)) = ((z \mid y) \mid y) \mid x \qquad \text{(Meredith 2)}$$

The pair of equations {26a, Commutativity} is one of several candidate systems proposed for study by Stephen Wolfram [12]. Wolfram's interest in these equations arose from his research project *A New Kind of Science* [13].

3. A Short 2-Basis

In this section, we prove the correctness of the new 2-basis. In the next section, we prove that no such basis can be shorter and that the new 2-basis is unique (for its length) up to applications of commutativity. OTTER played a crucial role in the search for and discovery of the proof

of correctness of the 2-basis. In Section 4, we describe briefly how we used OTTER and the method of proof sketches to find this proof.

Theorem 1. The pair of equations {26a, Commutativity} forms a 2-basis for Boolean algebra in terms of the Sheffer stroke.

Proof. If suffices to show that this pair is equivalent to the original Sheffer system. That 26a and Commutativity are theorems in Boolean algebra follows from a straightforward evaluation, so they necessarily are derivable from the Sheffer axioms. It remains to show then that each of the Sheffer axioms can be derived from 26a and Commutativity.

In the following proof, derivations of the Sheffer axioms are flagged with a * symbol. Applications of paramodulation are indicated by using a vector notation for terms. Specifically, in the vector `E.a1.a2.a3.a4...`, `E` is an equation (clause) number; `a1` refers to an argument of the equation—1 for the left side of the equation or 2 for the right side; `a2` refers to an argument of `E.a1`; `a3` refers to an argument of `E.a1.a2`; and so on. For example, the justification `4.1.2` ← `3.1` given for clause 9 indicates paramodulation from the left side of clause 3 into the subterm `(y|(x|z))` of clause 4. The word "flip" in a justification refers to an application of symmetry of equality.

1.	$(x\|y)\|(x\|(y\|z)) = x$	(26a)
2.	$x\|y = y\|x$	(Commutativity)
3.	$(x\|y)\|(x\|(z\|y)) = x$	[1.1.2.2 ← 2.1]
4.	$(x\|y)\|(y\|(x\|z)) = y$	[1.1.1 ← 2.1]
5.	$x\|((x\|y)\|(z\|(x\|(u\|y)))) = x\|y$	[3.1.1 ← 3.1]
6.	$(x\|y)\|(y\|(z\|x)) = y$	[3.1.1 ← 2.2]
7.	$((x\|y)\|(z\|y))\|x = x\|y$	[3.1.2 ← 3.1]
8.	$(x\|(y\|z))\|(x\|z) = x$	[3.1 ← 2.2]
9.	$(x\|(x\|y))\|x = x\|y$	[4.1.2 ← 3.1]
10.	$(x\|y)\|((x\|z)\|y) = y$	[4.1.2 ← 2.2]
11.	$(x\|(y\|z))\|(y\|x) = x$	[4.1 ← 2.2]
12.	$((x\|y)\|(x\|z))\|z = x\|z$	[6.1.2 ← 4.1]
13.	$(x\|(y\|z))\|(z\|x) = x$	[6.1 ← 2.2]
14.	$x\|((y\|x)\|(z\|y)) = y\|x$	[8.1.1 ← 6.1]
15.	$(x\|(y\|(z\|x)))\|y = y\|(z\|x)$	[6.1.2 ← 8.1]
16.	$((x\|(y\|z))\|z)\|x = x\|(y\|z)$	[3.1.2 ← 8.1]
* 17.	$(x\|x)\|(x\|y) = x$	[10.1.2 ← 9.1]
18.	$x\|((x\|y)\|(z\|(x\|x))) = x\|y$	[6.1.1 ← 17.1]

19. $x|(y|(x|(y|z))) = x|(y|z)$ [11.1.1 ← 11.1]
20. $x|(y|(x|x)) = x|x$ [13.1.1 ← 17.1]
21. $x|(y|(x|(z|y))) = x|(z|y)$ [13.1.1 ← 8.1]
* 22. $(x|y)|(y|y) = y$ [4.1.2 ← 20.1]
23. $(x|(y|z))|(x|(u|(y|x))) = (y|x)|(x|(y|z))$ [14.1.2.1 ← 4.1]
24. $x|(y|(y|x)) = x|x$ [15.1.1 ← 4.1 (flip)]
25. $x|(y|(x|y)) = x|x$ [15.1.1 ← 1.1 (flip)]
26. $(x|x)|y = y|(y|x)$ [15.1.1 ← 24.1]
27. $(x|x)|((x|y)|(z|(x|z))) = z|(x|z)$ [10.1.1 ← 25.1]
28. $x|(x|(y|y)) = y|x$ [26.1.1 ← 17.1 (flip)]
29. $x|(y|y) = x|(x|y)$ [26.1 ← 2.2]
30. $((x|y)|(x|y))|(x|x) = (x|x)|x$ [26.2.2 ← 17.1]
31. $x|(x|y) = x|(y|y)$ [2.1 ← 26.1]
32. $x|(x|(y|y)) = x|y$ [28.2 ← 2.2]
33. $(x|y)|y = y|(x|x)$ [9.1.1 ← 28.1]
34. $x|(y|y) = x|(y|x)$ [29.2.2 ← 2.2]
35. $(x|y)|x = x|(y|y)$ [29.2 ← 2.2 (flip)]
36. $x|(y|x) = (y|y)|x$ [2.1 ← 34.1]
37. $((x|y)|(z|y))|(x|x) = (x|y)|((x|y)|(z|y))$ [35.1.1 ← 7.1 (flip)]
38. $x|((x|y)|(z|(x|z))) = x|y$ [18.1.2.2 ← 34.1]
39. $x|(y|(x|(z|y))) = x|(y|z)$ [19.1.2.2.2 ← 2.2]
40. $(x|y)|(x|(x|(z|y))) = (x|y)|(x|y)$ [19.1.2 ← 5.1 (flip)]
41. $x|(y|z) = x|(z|y)$ [39.1 ← 21.1]
42. $(x|y)|z = z|(y|x)$ [41.1 ← 2.2]
43. $x|(y|z) = (z|y)|x$ [41.2 ← 2.2]
44. $(x|y)|(z|u) = (u|z)|(y|x)$ [42.1 ← 41.2]
45. $((x|y)|(x|y))|(x|x) = x|(x|x)$ [30.2 ← 43.2]
46. $((x|x)|(x|x))|((x|y)|(z|(x|z))) = ((x|y)|(z|(x|z)))|(z|(x|z))$ [36.1.2←27.1 (flip)]
47. $(x|(y|(z|y)))|(x|(x|(z|z))) = (x|(y|(z|y)))|(x|(y|(z|y)))$ [40.1.2.2.2 ← 25.1]
48. $(x|(y|z))|(x|(u|(y|x))) = x$ [23.2 ← 4.1]
49. $x|(y|(x|(z|(y|x)))) = x|x$ [16.1.1 ← 48.1 (flip)]
50. $x|(y|(z|(x|y))) = x|(y|y)$ [19.1.2 ← 49.1 (flip)]
51. $x|((y|y)|(z|(x|(x|y)))) = x|((y|y)|(y|y))$ [50.1.2.2.2 ← 31.2]

52. $x|(y|((y|x)|z)) = x|(y|y)$ [50.1.2.2 ← 43.1]
53. $(x|y)|((x|y)|(x|y)) = x|(x|x)$ [37.1 ← 45.1 (flip)]
54. $(x|(y|y))|(((y|x)|x)|((y|x)|x)) = (y|x)|((y|x)|(y|x))$ [53.1.1 ← 33.1]
55. $x|((y|y)|(z|(x|(x|y)))) = x|y$ [51.2.2 ← 22.1]
56. $((x|x)|(x|x))|((x|y)|(z|(x|z))) = (z|(x|z))|((x|y)|(x|y))$ [46.2 ← 33.1]
57. $(x|(y|y))|((x|(y|y))|((y|x)|x)) = (y|x)|((y|x)|(y|x))$ [54.1.2.1 ← 33.1]
58. $(x|(y|y))|((x|(y|y))|(x|(y|y))) = (y|x)|((y|x)|(y|x))$ [57.1.2.2 ← 33.1]
59. $(x|y)|((x|y)|(x|y)) = y|(y|y)$ [58.1 ← 53.1 (flip)]
60. $x|(x|x) = y|(y|y)$ [59.1 ← 53.1]
61. $(x|(y|(z|y)))|(x|(y|(z|y))) = (x|(y|(z|y)))|(x|z)$ [47.1.2 ← 32.1 (flip)]
62. $(x|(y|x))|((y|z)|(y|z)) = y|((y|z)|(x|(y|x)))$ [56.1.1 ← 22.1 (flip)]
63. $(x|(y|x))|((y|z)|(y|z)) = y|z$ [62.2 ← 38.1]
* 64. $x|(y|(y|y)) = x|x$ [18.1.2 ← 60.1]
65. $(x|y)|(x|(z|(z|z))) = x$ [1.1.2.2 ← 60.1]
66. $(x|((y|z)|x))|(y|((y|z)|(y|(u|(u|u))))) = (y|z)|(y|(u|(u|u)))$ [63.1.2.1 ← 65.1]
67. $(x|((y|z)|x))|(y|y) = (y|z)|(y|(u|(u|u)))$ [66.1.2.2 ← 65.1]
68. $(x|((y|z)|x))|(y|y) = y$ [67.2 ← 65.1]
69. $(x|(x|(y|z)))|(z|z) = z$ [68.1.1.2 ← 43.2]
70. $x|((y|(y|(z|x)))|x) = y|(y|(z|x))$ [8.1.1 ← 69.1]
71. $x|(y|(z|(z|(u|(y|x))))) = x|(y|y)$ [52.1.2.2 ← 70.1]
72. $x|(y|(y|(z|(x|y)))) = x|(y|(x|x))$ [19.1.2 ← 71.1 (flip)]
73. $x|(y|(y|(z|(x|y)))) = x|x$ [72.2 ← 20.1]
74. $(x|x)|(y|(y|(z|(y|(x|y))))) = (x|x)|(x|x)$ [73.1.2.2.2.2 ← 36.2]
75. $(x|x)|(y|(y|(z|(y|(x|y))))) = x$ [74.2 ← 22.1]
76. $x|(((y|(x|(z|x)))|(y|(x|(z|x))))|z) = x|(y|(x|(z|x)))$ [55.1.2.2 ← 75.1]
77. $x|(((y|(x|(z|x)))|(y|z))|z) = x|(y|(x|(z|x)))$ [76.1.2.1 ← 61.1]
78. $x|(y|(x|(z|x))) = x|(y|z)$ [77.1.2 ← 12.1 (flip)]
79. $x|(y|(x|(x|z))) = x|(y|z)$ [78.1.2.2.2 ← 2.2]

80. $x|(y|(x|(z|z))) = x|(y|z)$ [78.1.2.2 ← 34.2]
81. $x|(y|(z|x)) = x|(y|(z|z))$ [79.1.2.2 ← 28.1]
82. $(x|(y|z))|(x|(y|z)) = (x|(y|z))|(x|(z|z))$ [81.1.2 ← 21.1]
83. $(x|(y|(x|(z|z))))|(x|(z|z))$
$= (x|(y|z))|(x|(y|z))$ [82.2.1 ← 80.2 (flip)]
84. $(x|(y|y))|(x|(z|(x|(y|y)))) = (x|(z|y))|(x|(z|y))$ [83.1 ← 2.2]
85. $(x|(y|y))|(x|(z|z)) = (x|(z|y))|(x|(z|y))$ [84.1 ← 81.1]
* 86. $((x|x)|y)|((z|z)|y) = (y|(x|z))|(y|(x|z))$ [85.1 ← 44.2]

Steps 17 and 22 are generalizations of Sheffer 1; 64 is Sheffer 2; and 86 is the flip of Sheffer 3.

4. Properties

Theorem 2. Any 2-basis for Boolean algebra in terms of the Sheffer stroke has a total of at least six applications of the Sheffer stroke operator.

Proof. We show that no pair of equations with a total of fewer than six applications of the Sheffer stroke operator can be a 2-basis. In particular, each pair is ruled out either because at least one of the equations in the pair is not a Boolean identity or because there is a model that satisfies both equations in the pair but does not satisfy some Boolean identity.

By simple evaluation with the standard 2-element model for the Sheffer stroke, it is straightforward to see that the only possible 2-bases with strictly fewer than six applications of the Sheffer stroke operator will consist of Commutativity and either

$$(x \mid x) \mid (x \mid x) = x \qquad \text{(EQ-2.1)}$$

or a commutative variant of

$$(x \mid x) \mid (x \mid y) = x. \qquad \text{(EQ-2.2)}$$

But each of these remaining candidate pairs can be ruled out with the following 3-element model for $|$.

M_1 :

$\mid$	0	1	2
0	2	2	2
1	2	1	1
2	2	1	0

Specifically, this model satisfies each of the candidate 2-bases, but it does not satisfy the Boolean identity $x \mid (x \mid x) = y \mid (y \mid y)$.

All candidate pairs have been ruled out, so the result follows.

Since the length of an equation corresponds directly to the number of applications of the Sheffer stroke operator, it follows as a trivial corollary to Theorem 2 that the 2-basis {26a, Commutativity} indeed is a shortest 2-basis. The following theorem establishes that this 2-basis is unique up to applications of commutativity.

Theorem 3. The pair of equations {26a, Commutativity} and its commutative variants are the *only* shortest 2-bases for Boolean algebra in terms of the Sheffer stroke.

Proof. We show this by ruling out all of the other possibilities. Since any axiom must be a Boolean identity, we can restrict ourselves to the following two cases.

1. Commutativity together with one Boolean identity having four applications of the Sheffer stroke operator
2. Two Boolean identities, each having three applications of the Sheffer stroke operator

Case 1. By considering *all* well-formed formulas of the appropriate length, and by simple evaluation with the standard 2-element model for the Sheffer stroke, it is straightforward to see that the only identities with exactly four applications of the Sheffer stroke operator are (same-length) instances and commutative variants of the following three equations.

$$(x \mid x) \mid (y \mid (y \mid y)) = x \qquad \text{(EQ-3.1)}$$
$$(x \mid x) \mid (x \mid (y \mid z)) = x \qquad \text{(EQ-3.2)}$$
$$(x \mid y) \mid (x \mid (y \mid z)) = x \qquad \text{(EQ-3.3)}$$

Equation EQ-3.1 (including its instances and commutative variants) can be ruled out with the following 3-element model.

M_2 :

|	0	1	2
0	0	2	1
1	2	2	0
2	1	0	1

Specifically, M_2 is a commutative model for EQ-3.1, but it does not satisfy the Boolean identity EQ-2.2. EQ-3.2 (including its instances and commutative variants) can be ruled out by using the model M_1 from the proof of Theorem 2. EQ-3.3 is equation 26a, but we must rule out its (same-length) proper instances:

$$(x \mid x) \mid (x \mid (x \mid z)) = x \quad \text{(EQ-3.3a)}$$
$$(x \mid y) \mid (x \mid (y \mid x)) = x \quad \text{(EQ-3.3b)}$$
$$(x \mid y) \mid (x \mid (y \mid y)) = x \quad \text{(EQ-3.3c)}$$

EQ-3.3a is ruled out because it is an instance of EQ-3.2, which already has been ruled out by model M_1. EQ-3.3b and EQ-3.3c are ruled out by the following commutative model that does not satisfy the Boolean identity EQ-3.3.

M_3 :

|	0	1	2
0	2	2	1
1	2	2	0
2	1	0	1

Case 2. The candidates to consider are pairs of commutative variants of equation EQ-2.2. These pairs all can be ruled out by using the model M_1 from the proof of Theorem 2.

Since all other cases have been ruled out, the pair {26a, Commutativity} and its commutative variants indeed are the *only* shortest 2-bases for Boolean algebra in terms of the Sheffer stroke.

We note that we used the model-generation program MACE [3] to find some of the models presented in this section.

5. The Search

Our search for a proof of Theorem 1 involved sequences of OTTER experiments and relied heavily on the use of hints [9] and on the method of proof sketches [10]. Under the hints strategy, a generated clause is given special consideration (as defined by the user) if it subsumes or is subsumed by a user-supplied hint clause. The hints strategy is closely related to the weighting strategy [1], in which clauses are assigned weights that are used to help direct the search for a proof. In contrast to weighting, the hints strategy focuses directly on the identification of key clauses rather than on the general calculation of weights. Any generated clause that subsumes or is subsumed by a user-supplied hint clause is identified as being "interesting". The weight of such a clause is adjusted (either positively or negatively) according to user preferences; the cases of subsuming a hint, being subsumed by a hint, or both are controlled separately. Being based on subsumption, the hints strategy adds a semantic or logical component to the evaluation of a clause.

A proof sketch for a theorem T is a sequence of clauses giving a set of conditions *sufficient* to prove T. In the ideal case, a proof sketch

consists of a sequence of lemmas, where each lemma is fairly easy to prove. In any case, the clauses of a proof sketch identify potentially notable milestones on the way to finding a proof. From a strategic standpoint, it is desirable to recognize when we have achieved such milestones and to adapt the continued search for a proof accordingly. In particular, we wish to focus our attention on such milestone results and pursue their consequences sooner rather than later.

The hints strategy provides a natural and effective way to take full advantage of a proof sketch in the search for a proof. Including each clause from the proof sketch as a hint clause and making an OTTER assignment such as

```
% decrease by 100 the weight of any derived
% clause that back subsumes a hint clause
assign(bsub_hint_add_wt, -100).
```

virtually ensure that when a clause is derived that back subsumes a hint clause—in particular, one of the key milestone clauses of a proof sketch—the newly generated clause will become the focus of attention (that is, chosen as the "given" clause) as soon as possible.

The use of hints is additive in the sense that hints from multiple proof sketches or from sketches for different parts of a proof can all be included at the same time. For this reason, hints are particularly valuable for "gluing" subproofs together and completing partial proofs, even when wildly different search strategies were used to find the individual subproofs.

In [10], we consider how the generation and use of proof sketches, together with the sophisticated strategies and procedures supported by an automated reasoning program such as OTTER, can be used to find proofs to challenging theorems, including open questions. The general approach used in the search for simple axiom systems is to derive a known axiom system from some sufficient set of formulas—for example, a target axiom system with extra assumptions included—and then successively eliminate formulas from the input set, using all previous proofs as hints. For the Boolean algebra problem, we started with Wolfram's full set of candidate equations [11] and systematically eliminated them until only equations 26a and Commutativity remained. Because the elimination of equations was not strictly monotonic—at each step we considered the elimination of different candidates—we have a large number of proofs of intermediate results for various sets of equations. Rather than being a detriment to our search, this set of results served as a rich set of proof sketches (hints) that ultimately led us to the final result.

Although the proofs we initially found generally were proofs by contradiction and often relied on the use of demodulation, we were

able to use the techniques described in [10] to convert these proofs into strictly forward derivations of the desired theorems from the axioms. We find that strictly forward, demodulation-free proofs tend to make better proof sketches.

The intermediate proofs used in this study were not all found with a single, uniform strategy. The elimination of an equation as an assumption generally required a number of different tries with varying demodulation and weighting strategies. Our current work includes both the automation of the systematic derivation and use of proof sketches as well as the general improvement of the strategies for searching for individual proofs.

References

1. McCharen, J., Overbeek, R. and Wos, L.: Complexity and related enhancements for automated theorem-proving programs, *Computers and Mathematics with Applications* **2** (1976), 1–16.
2. McCune, W.: OTTER 3.0 reference manual and guide, Technical Report ANL-94/6, Argonne National Laboratory, Argonne, Illinois, 1994.
3. McCune, W.: MACE 2.0 reference manual and guide, Technical Memorandum ANL/MCS-TM-249, Argonne National Laboratory, Argonne, Illinois, 2001.
4. McCune, W., Veroff, R., Fitelson, B., Harris, K., Feist, A. and Wos, L.: Short single axioms for Boolean algebra, *J. Automated Reasoning* **29**(1) (2002), 1–16.
5. McCune, W., Padmanabhan, R. and Veroff, R.: Yet another single law for lattices, *Algebra Universalis*, to appear.
6. Meredith, C.: Equational postulates for the Sheffer stroke, *Notre Dame J. Formal Logic* **10**(3) (1969), 266–270.
7. Sheffer, H.: A set of five independent postulates for Boolean algebras, with application to logical constants, *Trans. Amer. Math. Soc.* **14**(4) (1913), 481–488.
8. Ulrich, D.: A legacy recalled and a tradition continued, *J. Automated Reasoning* **27**(2) (2001), 97–122.
9. Veroff, R.: Using hints to increase the effectiveness of an automated reasoning program: case studies, *J. Automated Reasoning* **16**(3) (1996), 223–239.
10. Veroff, R.: Solving open questions and other challenge problems using proof sketches, *J. Automated Reasoning* **27**(2) (2001), 157–174.
11. Veroff, R.: `http://www.cs.unm.edu/~veroff/BA/candidates.html`, 2001.
12. Wolfram, S.: Correspondence by electronic mail, 2000.
13. Wolfram, S.: A new kind of science. `http://wolframscience.com`, 2000.