DRAFT

A MATHEMATICA-BASED AUTOMATED DEDUCTION OF THE MARSDEN-HERMAN THEOREM IN QUANTUM LOGIC

Jack K. Horner
jhorner@cybermesa.com

Last modified: 5 November 2018/1050 CT

Abstract

In much the same way that Boolean logic (BL is)the logical structure of the system of propositions describing the results of measurement ("measurement propositions") of classical physical systems and is isomorphic to a Boolean algebra (BA), so also the algebra, C(H), of closed linear subspaces of (equivalently, the system of linear operators on (observables in)) a Hilbert space is a logic of the system of propositions describing the measurement of quantum mechanical systems and is a model of an ortholattice (OL). An OL can thus be thought of as a kind of "quantum logic" (QL). C(H) is also a model of an orthomodular lattice, which is an OL conjoined with the orthomodularity axiom (OMA). A consequence of the existence of non-commuting observables in quantum mechanics is that QL does not satisfy the BL distribution law. Quantum logicians have thus paid much attention to "quasi"-distributive theorems, and one of the better known of which is the Marsden-Herman theorem (MHT). Informally, the MHT states that if there is a cyclic chain of commuting elements in an orthomodular lattice, a strong version of the distributive law hold for those elements. Here I provide an automated deduction of the MHT.

Keywords: *automated deduction, quantum computing, orthomodular lattice, Hilbert space*

1.0 Introduction

In much the same way that Boolean logic (BL,[12]) is the logical structure of a system of propositions describing the results of measurement ("yes/no" propositions, "YNP"s; [13], Chap. 5) of classical physical systems and is isomorphic to a Boolean lattice ([10], [11], [19]), so also the algebra, C(H), of the closed linear subspaces of (equivalently, the system of linear operators on (observables in)) a Hilbert space H ([1], [4], [6], [9], [13]) is a logic of a system of propositions describing the results of measurement of quantum mechanical systems and is a model ([10]) of an ortholattice (OL; [8]). An OL can thus be thought of as a kind of "quantum logic" (QL; [19]). C(H) is also a model of an orthomodular lattice (OML; [7], [8]), which is an OL conjoined with the orthomodularity axiom (OMA; see Figure 1).

Lattice axioms

x = c(c(x)) (AxLat1)
x v y = y v x (AxLat2)
(x v y) v z = x v (y v z) (AxLat3)
(x ^ y) ^ z = x ^ (y ^ z) (AxLat4)
x v (x ^ y) = x (AxLat5)
x ^ (x v y) = x (AxLat6)

Orthogonality axioms
c(x) ^ x = 0 (AxOL1)
c(x) v x = 1 (AxOL2)
x ^ y = c(c(x) v c(y)) (AxOL3)

Orthomodularity axiom
y v (c(y) ^ (x v y)) = x v y (AxOM)

where
x, y are variables ranging over lattice nodes
^ is lattice meet
v is lattice join
c(x) is the orthocomplement of x
= is equivalence ([12])
1 is the maximum lattice element (= x v c(x))
0 is the minimum lattice element (= c(1))

Figure 1. Lattice, orthogonality, and orthomodularity axioms.

In QL, the non-commutativity of (certain) observables implies that the distributive law

(x ^(y v z) = (x ^ y) v (x v z))

does not hold (see [13], Section 5-4). A QL, in fact, can be thought of as a BL in which the distribution law does not hold. Because of the fundamental role that non-commutativity plays in QL, quantum logicians have paid much attention to "quasi"-distributive QL theorems, i.e., QL theorems that (non-trivially) identify conditions under which distributivity holds. Among the better known of the quasi-distributive theorems is the *Marsden-Herman Theorem* (MHT, [8]), shown is in Figure 2

If u, z, w, and x are elements of an orthomodular lattice and

C(u,z) & C(z,w) & C(w,x) & C(x,u)

then

(((u v z) ^ (w v x))) = (((u ^ w) v (u ^ x)) v ((z ^ w) v (z ^ x))).

where C(x,y), "x commutes with y", is defined as

C(x,y) ⟺ (x = ((x ^ y) v (x ^ c(y))))
⟺ means “is defined as”

Figure 2. The Marsden-Herman Theorem.

Informally stated, the MHT says that if there is a four-element cyclic commutative chain of elements in an orthomodular lattice, then a strong distribution law holds for those elements.

2.0 Method

The *Mathematica* ([2]) equational automated deduction framework (ADF) was used to assist the derivation of the Marsden-Herman Theorem.

2.1 Some definitions

Mathematica's ADF is based on a rewriting system that can be formulated in the nomenclature of *universal algebra*. A universal algebra is a pair <A; F>, where A is a non-void set and F is a family of finitary operations defined on A. F is not necessarily finite, and it may be void ([25], p. 8).

I assume the definitions of *term*, *value of a term*, *variable*, and *constant* contained in [23], Chapter 3.

A *rewriting system* is a system of R rules that transforms expressions that satisfy some well-defined set of formation rules to another expression that satisfy those formation rules. For the purposes of this paper, we will restrict our interest in a rewriting system that concerns identities of terms. In an identity of two terms, the values of the terms are equal for all values of variables occurring in them.

A *reduction* of a term T to a term T' is a (typically recursive) rewriting of T to T' using a set of rewriting rules R such that T' is "simpler than" T (given some definition of "simpler than"). A *reduction sequence* of a term T to a term T' is a sequence T0 = T, T1, T2, T3, ..., Tn = T', where each Ti is the result of applying R to Ti-1, i = 1, 2, ..., n.

If "simpler than" is a partial ordering ([16], p. 72) on a reduction sequence that begins with T and ends with T' in a system with a set of rewriting rules R, "simpler than" induces a *reduction order* ([23], p. 102) on the reduction sequence that begins with T and ends with T'.

A term Tn is in *normal form* if no application of R to Tn changes Tn.

A rewriting system is said to be *finitely terminating* if every reduction sequence of any term T produces, in a finite number of iterations, a normal form of T. A rewriting system is said to be *confluent* if the normal forms of all terms in the system are unique.

Some term rewriting systems are both finitely terminating and confluent ([23], esp. Chapter 9). Such rewriting systems have unique normal forms for all expressions. This permits us to use the the output of such a system to determine whether there is an identity between two terms T1 and T2 in the following manner. If T1 and T2 and have the same normal form, then there is an identity between T1 and T2. Otherwise, there is not an identity.

2.2 *Mathematica*'s inference algorithm

The inference algorithm in Mathematica's ADF is the Knuth-Bendix completion algorithm (KBC, [24]). KBC attempts to transform a given finite set of identities (an "input" to KBC) to a finitely terminating, confluent term rewriting system that preserves identity.

At initialization, KBC attempts to "orient" the identities supplied in its input according to the *KnuthBendix reduction order* ([23], Section 5.4.4). This results in an initial set of reduction rules. KBC then attempts to complete this initial set of rules with additional rules, obtaining their normal forms, and adding a new rule for every pair of the normal forms in accordance with the reduction order.
KBC may

1. Terminate with success, yielding a finitely terminating, confluent set of rules, or
2. Terminate with failure, or
3. Loop without terminating.

For further details of KBC, see [24].

2.3 Platform

The OML axiomatizations of Megill, Pavičić, and Horner ([5], [14], [15], [16], [21], [22]) were implemented as described in Section 3.0 of this paper in a *Mathematica* ([2]) script ([3]) configured to derive the MHT.

The notebook was executed under [2] running under Windows 10, on a Dell Inspiron 545 containing an Intel Q8200 quadprocessor clocked at 2.33 GHz and containing 8 GB memory.

A *Mathematica* notebook can, and typically does, contain instructions that execute in the *Mathematica* runtime. Unless a *Mathematica* instruction is terminated with a semicolon, *Mathematica* summarizes the results of attempting to execute that instruction. Those reports are prefixed with an expression of the form "ln[k] = ", followed by an informational message, where k = 1, 2, 3, ... In this script, default reporting is "on".

3.0 Results

The QL axioms shown in Figure 1 and the proof of Marsden-Herman Theorem can be captured in Mathematica as follows. Unless otherwise noted, the letters “a”, “b”, “c”, and “d” are variables ranging over lattice nodes.

```
In[73]:= (* Lattice axioms *)
AxLat1 = ForAll[a, a == comp[comp[a]]]
AxLat2 = ForAll[{a, b}, join[a, b] == join[b, a]]
AxLat3 = ForAll[{a, b, c}, join[join[a, b], c] == join[a, join[b, c]]]
AxLat4 = ForAll[{a, b, c}, meet[meet[a, b], c] == meet[a, meet[b, c]]]
AxLat5 = ForAll[{a, b}, join[a, meet[a, b]] == a]
AxLat6 = ForAll[{a, b}, meet[a, join[a, b]] == a]
LatticeAxioms = {AxLat1, AxLat2, AxLat3, AxLat4, AxLat5, AxLat6}

Out[73]= ∀a a == comp[comp[a]]

Out[74]= ∀{a,b} join[a, b] == join[b, a]

Out[75]= ∀{a,b,c} join[join[a, b], c] == join[a, join[b, c]]

Out[76]= ∀{a,b,c} meet[meet[a, b], c] == meet[a, meet[b, c]]

Out[77]= ∀{a,b} join[a, meet[a, b]] == a

Out[78]= ∀{a,b} meet[a, join[a, b]] == a

Out[79]= {∀a a == comp[comp[a]], ∀{a,b} join[a, b] == join[b, a],
  ∀{a,b,c} join[join[a, b], c] == join[a, join[b, c]],
  ∀{a,b,c} meet[meet[a, b], c] == meet[a, meet[b, c]],
  ∀{a,b} join[a, meet[a, b]] == a, ∀{a,b} meet[a, join[a, b]] == a}

In[80]:= (* Orthogonality axioms *)
AxOL1 = ForAll[a, meet[comp[a], a] == 0]
AxOL2 = ForAll[a, join[comp[a], a] == 1]
AxOL3 = ForAll[{a, b}, meet[a, b] == comp[join[comp[a], comp[b]]]]
OrthogonalityAxioms = {AxOL1, AxOL2, AxOL3}

Out[80]= ∀a meet[comp[a], a] == 0

Out[81]= ∀a join[comp[a], a] == 1

Out[82]= ∀{a,b} meet[a, b] == comp[join[comp[a], comp[b]]]

Out[83]= {∀a meet[comp[a], a] == 0, ∀a join[comp[a], a] == 1,
  ∀{a,b} meet[a, b] == comp[join[comp[a], comp[b]]]}

In[84]:= (* Orthomodularity axiom *)
OrthomodularityAxiom = {ForAll[{a, b}, join[b, meet[comp[b], join[a, b]]] == join[a, b]]}

Out[84]= {∀{a,b} join[b, meet[comp[b], join[a, b]]] == join[a, b]}

In[85]:= (* The QL axioms *)
```

```
In[86]:= QuantumLogicAxioms = Union[LatticeAxioms, OrthogonalityAxioms, OrthomodularityAxiom]

Out[86]= {∀a a == comp[comp[a]], ∀a join[comp[a], a] == 1, ∀a meet[comp[a], a] == 0,
  ∀{a,b} join[a, b] == join[b, a], ∀{a,b} join[a, meet[a, b]] == a,
  ∀{a,b} join[b, meet[comp[b], join[a, b]]] == join[a, b],
  ∀{a,b} meet[a, b] == comp[join[comp[a], comp[b]]], ∀{a,b} meet[a, join[a, b]] == a,
  ∀{a,b,c} join[join[a, b], c] == join[a, join[b, c]],
  ∀{a,b,c} meet[meet[a, b], c] == meet[a, meet[b, c]]}

In[87]:= (* Df. of "commutes with", comm[x,y] *)
comm[a, b] = ForAll[{a, b}, a == join[meet[a, b], meet[a, comp[b]]]]

Out[87]= ∀{a,b} a == join[meet[a, b], meet[a, comp[b]]]

In[88]:= comm[b, c] = ForAll[{b, c}, b == join[meet[b, c], meet[b, comp[c]]]]

Out[88]= ∀{b,c} b == join[meet[b, c], meet[b, comp[c]]]

In[89]:= comm[c, d] = ForAll[{c, d}, c == join[meet[c, d], meet[c, comp[d]]]]

Out[89]= ∀{c,d} c == join[meet[c, d], meet[c, comp[d]]]

In[90]:= comm[d, a] = ForAll[{d, a}, d == join[meet[d, a], meet[d, comp[a]]]]

Out[90]= ∀{d,a} d == join[meet[d, a], meet[d, comp[a]]]

In[91]:= (* consequent of the MHT *)

In[92]:= consequentMHTheorem = meet[join[a, b], join[c, d]] ==
  join[join[meet[a, c], meet[a, d]], join[meet[b, c], meet[b, d]]]

Out[92]= meet[join[a, b], join[c, d]] ==
  join[join[meet[a, c], meet[a, d]], join[meet[b, c], meet[b, d]]]
```

We now invoke Mathematica's FindEquationalProof function. To do this, we conjoin the "commutes with" definitions above (their conjunction is the hypothesis of the MHT) to the QuantumLogicAxioms, and prove the consequent of the MHT by implicitly using the Deduction Theorem ([12], *130).

```
In[93]:= (* summary of the proof of the MHT *)

In[94]:= proofMHTheorem = FindEquationalProof[consequentMHTheorem,
  {comm[a, b], comm[b, c], comm[c, d], comm[d, a], QuantumLogicAxioms}]

Out[94]= ProofObject[
  Logic: EquationalLogic        Steps: 145
  Theorem: meet[join[a, b], join[c, d]] == join[join[meet[a, c], meet[a, d]], join[meet[b, c], meet[b, d]]]
]
```

A listing and graph of this 145-step proof is contained in the Appendix.

The time-to-solution on the platform described in Section 2.0 was approximately five seconds.

4.0 Acknowledgements

This work benefited from discussions with Frank Pecchioni and Tony Pawlicki. I am also indebted to the Tom Oberdan and John K. Prentice, whose passion for foundations of physics inspired those of us privileged to have known them. For any infelicities that remain, I am solely responsible.

APPENDIX. Listing and graph of the proof of the Marsden-Herman Theorem.

In[95]:=
```
proofMHTheorem["ProofNotebook"]
```

Axiom 1

We are given that:

```
x1==meet[x1,join[x1,x2]]
```

Axiom 2

We are given that:

```
x1==join[x1,meet[x1,x2]]
```

Axiom 3

We are given that:

```
x1==join[meet[x1,x2],meet[x1,comp[x2]]]
```

Axiom 4

We are given that:

```
x1==comp[comp[x1]]
```

Axiom 5

We are given that:

```
meet[x1,x2]==comp[join[comp[x1],comp[x2]]]
```

Axiom 6

We are given that:

```
meet[x1,meet[x2,x3]]==meet[meet[x1,x2],x3]
```

Axiom 7

We are given that:

```
join[x1,x2]==join[x2,x1]
```

Axiom 8

We are given that:

```
join[x1,join[x2,x3]]==join[join[x1,x2],x3]
```

Hypothesis 1

We would like to show that:

```
meet[join[a,b],join[c,d]]==join[join[meet[a,c],meet[a,d]],join[meet[b,c],meet[b,d]]]
```

Critical Pair Lemma 1

The following expressions are equivalent:

```
x1==meet[x1,x1]
```

Proof

Note that the input for the rule:

```
meet[x1_,join[x1_,x2_]]→x1
```

contains a subpattern of the form:

```
join[x1_,x2_]
```

which can be unified with the input for the rule:

```
join[x1_,meet[x1_,x2_]]→x1
```

where these rules follow from Axiom 1 and Axiom 2 respectively.

Critical Pair Lemma 2

The following expressions are equivalent:

```
meet[x1,x2]⩵meet[meet[x1,x2],x1]
```

Proof

Note that the input for the rule:

```
meet[x1_,join[x1_,x2_]]→x1
```

contains a subpattern of the form:

```
join[x1_,x2_]
```

which can be unified with the input for the rule:

```
join[meet[x1_,x2_],meet[x1_,comp[x2_]]]→x1
```

where these rules follow from Axiom 1 and Axiom 3 respectively.

Critical Pair Lemma 3

The following expressions are equivalent:

```
meet[comp[x1],x2]⩵comp[join[x1,comp[x2]]]
```

Proof

Note that the input for the rule:

```
comp[join[comp[x1_],comp[x2_]]]→meet[x1,x2]
```

contains a subpattern of the form:

```
comp[x1_]
```

which can be unified with the input for the rule:

```
comp[comp[x1_]]→x1
```

where these rules follow from Axiom 5 and Axiom 4 respectively.

Critical Pair Lemma 4

The following expressions are equivalent:

```
x1⩵meet[x1,join[x2,x1]]
```

Proof

Note that the input for the rule:

```
meet[x1_,join[x1_,x2_]]→x1
```

contains a subpattern of the form:

```
join[x1_,x2_]
```

which can be unified with the input for the rule:

```
join[x1_,x2_]↔join[x2_,x1_]
```

where these rules follow from Axiom 1 and Axiom 7 respectively.

where these rules follow from Axiom 1 and Axiom 7 respectively.

Critical Pair Lemma 5

The following expressions are equivalent:

`meet[x1,x2]==comp[join[comp[x2],comp[x1]]]`

Proof

Note that the input for the rule:

`comp[join[comp[x1_],comp[x2_]]]→meet[x1,x2]`

contains a subpattern of the form:

`join[comp[x1_],comp[x2_]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Axiom 5 and Axiom 7 respectively.

Substitution Lemma 1

It can be shown that:

`meet[x1,x2]==meet[x2,x1]`

Proof

We start by taking Critical Pair Lemma 5, and apply the substitution:

`comp[join[comp[x1_],comp[x2_]]]→meet[x1,x2]`

which follows from Axiom 5.

Critical Pair Lemma 6

The following expressions are equivalent:

`join[meet[x1,x2],join[meet[x1,comp[x2]],x3]]==join[x1,x3]`

Proof

Note that the input for the rule:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[meet[x1_,x2_],meet[x1_,comp[x2_]]]→x1`

where these rules follow from Axiom 8 and Axiom 3 respectively.

Critical Pair Lemma 7

The following expressions are equivalent:

`x1==meet[x1,join[x2,join[x3,x1]]]`

Proof

Note that the input for the rule:

`meet[x1_,join[x2_,x1_]]→x1`

contains a subpattern of the form:

`join[x2_,x1_]`

which can be unified with the input for the rule:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

where these rules follow from Critical Pair Lemma 4 and Axiom 8 respectively.

Critical Pair Lemma 8

The following expressions are equivalent:

`x1==join[x1,meet[x2,x1]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x1_,x2_]]→x1`

contains a subpattern of the form:

`meet[x1_,x2_]`

which can be unified with the input for the rule:

`meet[x1_,x2_]↔meet[x2_,x1_]`

where these rules follow from Axiom 2 and Substitution Lemma 1 respectively.

Critical Pair Lemma 9

The following expressions are equivalent:

`x1==join[meet[x2,x1],meet[x1,comp[x2]]]`

Proof

Note that the input for the rule:

`join[meet[x1_,x2_],meet[x1_,comp[x2_]]]→x1`

contains a subpattern of the form:

`meet[x1_,x2_]`

which can be unified with the input for the rule:

`meet[x1_,x2_]↔meet[x2_,x1_]`

where these rules follow from Axiom 3 and Substitution Lemma 1 respectively.

Substitution Lemma 2

It can be shown that:

`meet[x1,x2]==meet[x1,meet[x2,x1]]`

Proof

We start by taking Critical Pair Lemma 2, and apply the substitution:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

which follows from Axiom 6.

Critical Pair Lemma 10

The following expressions are equivalent:

`meet[x1,meet[x2,x3]]==meet[x1,meet[x2,meet[x3,x1]]]`

Proof

Note that the input for the rule:

`meet[x1_,meet[x2_,x1_]]→meet[x1,x2]`

`meet[x1_,meet[x2_,x1_]]→meet[x1,x2]`

contains a subpattern of the form:

`meet[x2_,x1_]`

which can be unified with the input for the rule:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

where these rules follow from Substitution Lemma 2 and Axiom 6 respectively.

Critical Pair Lemma 11

The following expressions are equivalent:

`meet[x1,meet[meet[x2,x1],x3]]==meet[meet[x1,x2],x3]`

Proof

Note that the input for the rule:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

contains a subpattern of the form:

`meet[x1_,x2_]`

which can be unified with the input for the rule:

`meet[x1_,meet[x2_,x1_]]→meet[x1,x2]`

where these rules follow from Axiom 6 and Substitution Lemma 2 respectively.

Substitution Lemma 3

It can be shown that:

`meet[x1,meet[x2,meet[x1,x3]]]==meet[meet[x1,x2],x3]`

Proof

We start by taking Critical Pair Lemma 11, and apply the substitution:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

which follows from Axiom 6.

Substitution Lemma 4

It can be shown that:

`meet[x1,meet[x2,meet[x1,x3]]]==meet[x1,meet[x2,x3]]`

Proof

We start by taking Substitution Lemma 3, and apply the substitution:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

which follows from Axiom 6.

Critical Pair Lemma 12

The following expressions are equivalent:

`join[x1,x2]==join[join[x1,x2],x1]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,x1_]]→x1`

contains a subpattern of the form:

`meet[x2_,x1_]`

which can be unified with the input for the rule:

`meet[x1_,join[x1_,x2_]]→x1`

where these rules follow from Critical Pair Lemma 8 and Axiom 1 respectively.

Substitution Lemma 5

It can be shown that:

`join[x1,x2]==join[x1,join[x2,x1]]`

Proof

We start by taking Critical Pair Lemma 12, and apply the substitution:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

which follows from Axiom 8.

Critical Pair Lemma 13

The following expressions are equivalent:

`x1==join[x1,meet[x2,meet[x3,x1]]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,x1_]]→x1`

contains a subpattern of the form:

`meet[x2_,x1_]`

which can be unified with the input for the rule:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

where these rules follow from Critical Pair Lemma 8 and Axiom 6 respectively.

Critical Pair Lemma 14

The following expressions are equivalent:

`join[x1,join[meet[x2,x1],x3]]==join[x1,x3]`

Proof

Note that the input for the rule:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[x1_,meet[x2_,x1_]]→x1`

where these rules follow from Axiom 8 and Critical Pair Lemma 8 respectively.

Critical Pair Lemma 15

The following expressions are equivalent:

`join[x1,join[x2,x3]]==join[x1,join[x2,join[x3,x1]]]`

Proof

Note that the input for the rule:

`join[x1_,join[x2_,x1_]]→join[x1,x2]`

contains a subpattern of the form:

`join[x2_,x1_]`

which can be unified with the input for the rule:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

where these rules follow from Substitution Lemma 5 and Axiom 8 respectively.

Critical Pair Lemma 16

The following expressions are equivalent:

`join[x1,join[join[x2,x1],x3]]==join[join[x1,x2],x3]`

Proof

Note that the input for the rule:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[x1_,join[x2_,x1_]]→join[x1,x2]`

where these rules follow from Axiom 8 and Substitution Lemma 5 respectively.

Substitution Lemma 6

It can be shown that:

`join[x1,join[x2,join[x1,x3]]]==join[join[x1,x2],x3]`

Proof

We start by taking Critical Pair Lemma 16, and apply the substitution:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

which follows from Axiom 8.

Substitution Lemma 7

It can be shown that:

`join[x1,join[x2,join[x1,x3]]]==join[x1,join[x2,x3]]`

Proof

We start by taking Substitution Lemma 6, and apply the substitution:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

which follows from Axiom 8.

Critical Pair Lemma 17

The following expressions are equivalent:

`meet[comp[x1],x2]==comp[join[comp[x2],x1]]`

Proof

Note that the input for the rule:

`comp[join[x1_,comp[x2_]]]→meet[comp[x1],x2]`

contains a subpattern of the form:

contains a subpattern of the form:

`join[x1_,comp[x2_]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Critical Pair Lemma 3 and Axiom 7 respectively.

Critical Pair Lemma 18

The following expressions are equivalent:

`meet[comp[x1],comp[x2]]==comp[join[x1,x2]]`

Proof

Note that the input for the rule:

`comp[join[x1_,comp[x2_]]]→meet[comp[x1],x2]`

contains a subpattern of the form:

`comp[x2_]`

which can be unified with the input for the rule:

`comp[comp[x1_]]→x1`

where these rules follow from Critical Pair Lemma 3 and Axiom 4 respectively.

Critical Pair Lemma 19

The following expressions are equivalent:

`join[x1,comp[x2]]==comp[meet[comp[x1],x2]]`

Proof

Note that the input for the rule:

`comp[comp[x1_]]→x1`

contains a subpattern of the form:

`comp[x1_]`

which can be unified with the input for the rule:

`comp[join[x1_,comp[x2_]]]→meet[comp[x1],x2]`

where these rules follow from Axiom 4 and Critical Pair Lemma 3 respectively.

Critical Pair Lemma 20

The following expressions are equivalent:

`join[x1,comp[x2]]==comp[meet[x2,comp[x1]]]`

Proof

Note that the input for the rule:

`comp[meet[comp[x1_],x2_]]→join[x1,comp[x2]]`

contains a subpattern of the form:

`meet[comp[x1_],x2_]`

which can be unified with the input for the rule:

`meet[x1_,x2_]↔meet[x2_,x1_]`

where these rules follow from Critical Pair Lemma 19 and Substitution Lemma 1 respectively.

Critical Pair Lemma 21

Critical Pair Lemma 21

The following expressions are equivalent:

`join[comp[x1],comp[x2]]==comp[meet[x1,x2]]`

Proof

Note that the input for the rule:

`comp[meet[comp[x1_],x2_]]→join[x1,comp[x2]]`

contains a subpattern of the form:

`comp[x1_]`

which can be unified with the input for the rule:

`comp[comp[x1_]]→x1`

where these rules follow from Critical Pair Lemma 19 and Axiom 4 respectively.

Critical Pair Lemma 22

The following expressions are equivalent:

`comp[meet[x1,join[comp[x2],x3]]]==join[comp[x1],meet[comp[x3],x2]]`

Proof

Note that the input for the rule:

`join[comp[x1_],comp[x2_]]→comp[meet[x1,x2]]`

contains a subpattern of the form:

`comp[x2_]`

which can be unified with the input for the rule:

`comp[join[comp[x1_],x2_]]→meet[comp[x2],x1]`

where these rules follow from Critical Pair Lemma 21 and Critical Pair Lemma 17 respectively.

Critical Pair Lemma 23

The following expressions are equivalent:

`x1==meet[x1,join[join[x2,x1],x3]]`

Proof

Note that the input for the rule:

`meet[x1_,join[x2_,join[x3_,x1_]]]→x1`

contains a subpattern of the form:

`join[x2_,join[x3_,x1_]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Critical Pair Lemma 7 and Axiom 7 respectively.

Substitution Lemma 8

It can be shown that:

`x1==meet[x1,join[x2,join[x1,x3]]]`

Proof

We start by taking Critical Pair Lemma 23, and apply the substitution:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]` `join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

which follows from Axiom 8.

Critical Pair Lemma 24

The following expressions are equivalent:

`join[x1,x2]==join[join[x1,x2],meet[x3,x1]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,meet[x3_,x1_]]]→x1`

contains a subpattern of the form:

`meet[x3_,x1_]`

which can be unified with the input for the rule:

`meet[x1_,join[x1_,x2_]]→x1`

where these rules follow from Critical Pair Lemma 13 and Axiom 1 respectively.

Substitution Lemma 9

It can be shown that:

`join[x1,x2]==join[x1,join[x2,meet[x3,x1]]]`

Proof

We start by taking Critical Pair Lemma 24, and apply the substitution:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

which follows from Axiom 8.

Critical Pair Lemma 25

The following expressions are equivalent:

`join[x1,meet[x2,comp[x1]]]==join[x1,x2]`

Proof

Note that the input for the rule:

`join[x1_,join[meet[x2_,x1_],x3_]]→join[x1,x3]`

contains a subpattern of the form:

`join[meet[x2_,x1_],x3_]`

which can be unified with the input for the rule:

`join[meet[x1_,x2_],meet[x1_,comp[x2_]]]→x1`

where these rules follow from Critical Pair Lemma 14 and Axiom 3 respectively.

Critical Pair Lemma 26

The following expressions are equivalent:

`join[x1,comp[x2]]==join[x1,comp[join[x2,x1]]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]`

contains a subpattern of the form:

`meet[x2_,comp[x1_]]`

which can be unified with the input for the rule:

`meet[comp[x1_],comp[x2_]]→comp[join[x1,x2]]`

where these rules follow from Critical Pair Lemma 25 and Critical Pair Lemma 18 respectively.

Critical Pair Lemma 27

The following expressions are equivalent:

`join[x1,meet[x2,x3]]==join[x1,meet[x2,meet[x3,comp[x1]]]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]`

contains a subpattern of the form:

`meet[x2_,comp[x1_]]`

which can be unified with the input for the rule:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

where these rules follow from Critical Pair Lemma 25 and Axiom 6 respectively.

Critical Pair Lemma 28

The following expressions are equivalent:

`join[x1,x2]==join[x1,meet[comp[x1],x2]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]`

contains a subpattern of the form:

`meet[x2_,comp[x1_]]`

which can be unified with the input for the rule:

`meet[x1_,x2_]↔meet[x2_,x1_]`

where these rules follow from Critical Pair Lemma 25 and Substitution Lemma 1 respectively.

Critical Pair Lemma 29

The following expressions are equivalent:

`join[x1,join[meet[x2,comp[x1]],x3]]==join[join[x1,x2],x3]`

Proof

Note that the input for the rule:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]`

where these rules follow from Axiom 8 and Critical Pair Lemma 25 respectively.

Substitution Lemma 10

It can be shown that:

`join[x1,join[meet[x2,comp[x1]],x3]]==join[x1,join[x2,x3]]`

PROOF

We start by taking Critical Pair Lemma 29, and apply the substitution:

`join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]`

which follows from Axiom 8.

Critical Pair Lemma 30

The following expressions are equivalent:

`meet[comp[comp[join[x1,comp[x2]]]],x2]==comp[join[comp[x2],comp[x1]]]`

PROOF

Note that the input for the rule:

`comp[join[comp[x1_],x2_]]→meet[comp[x2],x1]`

contains a subpattern of the form:

`join[comp[x1_],x2_]`

which can be unified with the input for the rule:

`join[x1_,comp[join[x2_,x1_]]]→join[x1,comp[x2]]`

where these rules follow from Critical Pair Lemma 17 and Critical Pair Lemma 26 respectively.

Substitution Lemma 11

It can be shown that:

`meet[join[x1,comp[x2]],x2]==comp[join[comp[x2],comp[x1]]]`

PROOF

We start by taking Critical Pair Lemma 30, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

Substitution Lemma 12

It can be shown that:

`meet[join[x1,comp[x2]],x2]==meet[comp[comp[x1]],x2]`

PROOF

We start by taking Substitution Lemma 11, and apply the substitution:

`comp[join[comp[x1_],x2_]]→meet[comp[x2],x1]`

which follows from Critical Pair Lemma 17.

Substitution Lemma 13

It can be shown that:

`meet[join[x1,comp[x2]],x2]==meet[x1,x2]`

PROOF

We start by taking Substitution Lemma 12, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

which follows from Axiom 4.

Substitution Lemma 14

It can be shown that:

`meet[x1,join[x2,comp[x1]]]==meet[x2,x1]`

Proof

We start by taking Substitution Lemma 13, and apply the substitution:

`meet[x1_,x2_]→meet[x2,x1]`

which follows from Substitution Lemma 1.

Critical Pair Lemma 31

The following expressions are equivalent:

`meet[comp[x1],x2]==meet[x2,comp[meet[x1,x2]]]`

Proof

Note that the input for the rule:

`meet[x1_,join[x2_,comp[x1_]]]→meet[x2,x1]`

contains a subpattern of the form:

`join[x2_,comp[x1_]]`

which can be unified with the input for the rule:

`join[comp[x1_],comp[x2_]]→comp[meet[x1,x2]]`

where these rules follow from Substitution Lemma 14 and Critical Pair Lemma 21 respectively.

Critical Pair Lemma 32

The following expressions are equivalent:

`meet[x1,x2]==meet[x2,join[comp[x2],x1]]`

Proof

Note that the input for the rule:

`meet[x1_,join[x2_,comp[x1_]]]→meet[x2,x1]`

contains a subpattern of the form:

`join[x2_,comp[x1_]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Substitution Lemma 14 and Axiom 7 respectively.

Critical Pair Lemma 33

The following expressions are equivalent:

`meet[x1,comp[x2]]==meet[comp[x2],join[x2,x1]]`

Proof

Note that the input for the rule:

`meet[x1_,join[comp[x1_],x2_]]→meet[x2,x1]`

contains a subpattern of the form:

`comp[x1_]`

which can be unified with the input for the rule:

```
comp[comp[x1_]]→x1
```

where these rules follow from Critical Pair Lemma 32 and Axiom 4 respectively.

Critical Pair Lemma 34

The following expressions are equivalent:

```
meet[join[x1,meet[x2,x3]],comp[x3]]==meet[comp[x3],join[x3,x1]]
```

Proof

Note that the input for the rule:

```
meet[comp[x1_],join[x1_,x2_]]→meet[x2,comp[x1]]
```

contains a subpattern of the form:

```
join[x1_,x2_]
```

which can be unified with the input for the rule:

```
join[x1_,join[x2_,meet[x3_,x1_]]]→join[x1,x2]
```

where these rules follow from Critical Pair Lemma 33 and Substitution Lemma 9 respectively.

Substitution Lemma 15

It can be shown that:

```
meet[join[x1,meet[x2,x3]],comp[x3]]==meet[x1,comp[x3]]
```

Proof

We start by taking Critical Pair Lemma 34, and apply the substitution:

```
meet[comp[x1_],join[x1_,x2_]]→meet[x2,comp[x1]]
```

which follows from Critical Pair Lemma 33.

Critical Pair Lemma 35

The following expressions are equivalent:

```
join[x1,join[x2,x3]]==join[x3,meet[join[x1,join[x2,x3]],comp[x3]]]
```

Proof

Note that the input for the rule:

```
join[meet[x1_,x2_],meet[x2_,comp[x1_]]]→x2
```

contains a subpattern of the form:

```
meet[x1_,x2_]
```

which can be unified with the input for the rule:

```
meet[x1_,join[x2_,join[x3_,x1_]]]→x1
```

where these rules follow from Critical Pair Lemma 9 and Critical Pair Lemma 7 respectively.

Substitution Lemma 16

It can be shown that:

```
join[x1,join[x2,x3]]==join[x3,join[x1,join[x2,x3]]]
```

Proof

We start by taking Critical Pair Lemma 35, and apply the substitution:

```
join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]
```

```
join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]
```

which follows from Critical Pair Lemma 25.

Critical Pair Lemma 36

The following expressions are equivalent:

```
join[x1,join[x2,x3]]⩵join[x2,meet[join[x1,join[x2,x3]],comp[x2]]]
```

Proof

Note that the input for the rule:

```
join[meet[x1_,x2_],meet[x2_,comp[x1_]]]→x2
```

contains a subpattern of the form:

```
meet[x1_,x2_]
```

which can be unified with the input for the rule:

```
meet[x1_,join[x2_,join[x1_,x3_]]]→x1
```

where these rules follow from Critical Pair Lemma 9 and Substitution Lemma 8 respectively.

Substitution Lemma 17

It can be shown that:

```
join[x1,join[x2,x3]]⩵join[x2,join[x1,join[x2,x3]]]
```

Proof

We start by taking Critical Pair Lemma 36, and apply the substitution:

```
join[x1_,meet[x2_,comp[x1_]]]→join[x1,x2]
```

which follows from Critical Pair Lemma 25.

Critical Pair Lemma 37

The following expressions are equivalent:

```
x1⩵join[meet[meet[x2,x1],x1],meet[comp[x2],x1]]
```

Proof

Note that the input for the rule:

```
join[meet[x1_,x2_],meet[x2_,comp[x1_]]]→x2
```

contains a subpattern of the form:

```
meet[x2_,comp[x1_]]
```

which can be unified with the input for the rule:

```
meet[x1_,comp[meet[x2_,x1_]]]→meet[comp[x2],x1]
```

where these rules follow from Critical Pair Lemma 9 and Critical Pair Lemma 31 respectively.

Substitution Lemma 18

It can be shown that:

```
x1⩵join[meet[x2,meet[x1,x1]],meet[comp[x2],x1]]
```

Proof

We start by taking Critical Pair Lemma 37, and apply the substitution:

```
meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]
```

which follows from Axiom 6.

Substitution Lemma 19

It can be shown that:

`x1==join[meet[x2,x1],meet[comp[x2],x1]]`

Proof

We start by taking Substitution Lemma 18, and apply the substitution:

`meet[x1_,x1_]→x1`

which follows from Critical Pair Lemma 1.

Critical Pair Lemma 38

The following expressions are equivalent:

`meet[x1,meet[x2,join[comp[x1],x3]]]==meet[x1,meet[x2,meet[x3,x1]]]`

Proof

Note that the input for the rule:

`meet[x1_,meet[x2_,meet[x1_,x3_]]]→meet[x1,meet[x2,x3]]`

contains a subpattern of the form:

`meet[x1_,x3_]`

which can be unified with the input for the rule:

`meet[x1_,join[comp[x1_],x2_]]→meet[x2,x1]`

where these rules follow from Substitution Lemma 4 and Critical Pair Lemma 32 respectively.

Substitution Lemma 20

It can be shown that:

`meet[x1,meet[x2,join[comp[x1],x3]]]==meet[x1,meet[x2,x3]]`

Proof

We start by taking Critical Pair Lemma 38, and apply the substitution:

`meet[x1_,meet[x2_,meet[x3_,x1_]]]→meet[x1,meet[x2,x3]]`

which follows from Critical Pair Lemma 10.

Substitution Lemma 21

It can be shown that:

`join[x1,join[x2,x3]]==join[x3,join[x1,x2]]`

Proof

We start by taking Substitution Lemma 16, and apply the substitution:

`join[x1_,join[x2_,join[x3_,x1_]]]→join[x1,join[x2,x3]]`

which follows from Critical Pair Lemma 15.

Substitution Lemma 22

It can be shown that:

`join[x1,join[x2,x3]]==join[x2,join[x1,x3]]`

Proof

We start by taking Substitution Lemma 17, and apply the substitution:

`join[x1_,join[x2_,join[x1_,x3_]]]→join[x1,join[x2,x3]]`

which follows from Substitution Lemma 7.

Substitution Lemma 23

It can be shown that:

`meet[comp[x1],join[x2,meet[x3,x1]]]==meet[x2,comp[x1]]`

Proof

We start by taking Substitution Lemma 15, and apply the substitution:

`meet[x1_,x2_]→meet[x2,x1]`

which follows from Substitution Lemma 1.

Critical Pair Lemma 39

The following expressions are equivalent:

`meet[x1,comp[meet[x2,comp[x1]]]]==meet[comp[meet[x2,comp[x1]]],join[x1,meet[x3,x2]]]`

Proof

Out[95]=

Note that the input for the rule:

`meet[comp[x1_],join[x2_,meet[x3_,x1_]]]→meet[x2,comp[x1]]`

contains a subpattern of the form:

`join[x2_,meet[x3_,x1_]]`

which can be unified with the input for the rule:

`join[x1_,meet[x2_,meet[x3_,comp[x1_]]]]→join[x1,meet[x2,x3]]`

where these rules follow from Substitution Lemma 23 and Critical Pair Lemma 27 respectively.

Substitution Lemma 24

It can be shown that:

`meet[x1,join[x1,comp[x2]]]==meet[comp[meet[x2,comp[x1]]],join[x1,meet[x3,x2]]]`

Proof

We start by taking Critical Pair Lemma 39, and apply the substitution:

`comp[meet[x1_,comp[x2_]]]→join[x2,comp[x1]]`

which follows from Critical Pair Lemma 20.

Substitution Lemma 25

It can be shown that:

`x1==meet[comp[meet[x2,comp[x1]]],join[x1,meet[x3,x2]]]`

Proof

We start by taking Substitution Lemma 24, and apply the substitution:

`meet[x1_,join[x1_,x2_]]→x1`

which follows from Axiom 1.

Substitution Lemma 26

It can be shown that:

`x1==meet[join[x1,comp[x2]],join[x1,meet[x3,x2]]]`

PROOF

We start by taking Substitution Lemma 25, and apply the substitution:

```
comp[meet[x1_,comp[x2_]]]→join[x2,comp[x1]]
```

which follows from Critical Pair Lemma 20.

Critical Pair Lemma 40

The following expressions are equivalent:

```
meet[join[meet[x1,comp[comp[x2]]],x3],x2]==meet[x2,join[comp[x2],join[x1,x3]]]
```

PROOF

Note that the input for the rule:

```
meet[x1_,join[comp[x1_],x2_]]→meet[x2,x1]
```

contains a subpattern of the form:

```
join[comp[x1_],x2_]
```

which can be unified with the input for the rule:

```
join[x1_,join[meet[x2_,comp[x1_]],x3_]]→join[x1,join[x2,x3]]
```

where these rules follow from Critical Pair Lemma 32 and Substitution Lemma 10 respectively.

Substitution Lemma 27

It can be shown that:

```
meet[join[meet[x1,x2],x3],x2]==meet[x2,join[comp[x2],join[x1,x3]]]
```

PROOF

We start by taking Critical Pair Lemma 40, and apply the substitution:

```
comp[comp[x1_]]→x1
```

which follows from Axiom 4.

Substitution Lemma 28

It can be shown that:

```
meet[join[meet[x1,x2],x3],x2]==meet[join[x1,x3],x2]
```

PROOF

We start by taking Substitution Lemma 27, and apply the substitution:

```
meet[x1_,join[comp[x1_],x2_]]→meet[x2,x1]
```

which follows from Critical Pair Lemma 32.

Substitution Lemma 29

It can be shown that:

```
meet[x1,join[meet[x2,x1],x3]]==meet[join[x2,x3],x1]
```

PROOF

We start by taking Substitution Lemma 28, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Critical Pair Lemma 41

The following expressions are equivalent:

`meet[join[x1,x2],x3]⩵meet[x3,join[x2,meet[x1,x3]]]`

Proof

Note that the input for the rule:

`meet[x1_,join[meet[x2_,x1_],x3_]]→meet[join[x2,x3],x1]`

contains a subpattern of the form:

`join[meet[x2_,x1_],x3_]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Substitution Lemma 29 and Axiom 7 respectively.

Critical Pair Lemma 42

The following expressions are equivalent:

`meet[join[x1,x2],x3]⩵meet[x3,join[meet[x3,x1],x2]]`

Proof

Note that the input for the rule:

`meet[x1_,join[meet[x2_,x1_],x3_]]→meet[join[x2,x3],x1]`

contains a subpattern of the form:

`meet[x2_,x1_]`

which can be unified with the input for the rule:

`meet[x1_,x2_]↔meet[x2_,x1_]`

where these rules follow from Substitution Lemma 29 and Substitution Lemma 1 respectively.

Critical Pair Lemma 43

The following expressions are equivalent:

`meet[join[x1,x2],x3]⩵meet[x3,join[x2,meet[x3,x1]]]`

Proof

Note that the input for the rule:

`meet[x1_,join[x2_,meet[x3_,x1_]]]→meet[join[x3,x2],x1]`

contains a subpattern of the form:

`meet[x3_,x1_]`

which can be unified with the input for the rule:

`meet[x1_,x2_]↔meet[x2_,x1_]`

where these rules follow from Critical Pair Lemma 41 and Substitution Lemma 1 respectively.

Critical Pair Lemma 44

The following expressions are equivalent:

`join[x1,meet[x2,join[comp[comp[x1]],x3]]]⩵join[x1,meet[comp[x1],meet[x2,x3]]]`

Proof

Note that the input for the rule:

`join[x1_,meet[comp[x1_],x2_]]→join[x1,x2]`

contains a subpattern of the form:

`meet[comp[x1_],x2_]`

which can be unified with the input for the rule:

`meet[x1_,meet[x2_,join[comp[x1_],x3_]]]→meet[x1,meet[x2,x3]]`

where these rules follow from Critical Pair Lemma 28 and Substitution Lemma 20 respectively.

Substitution Lemma 30

It can be shown that:

`join[x1,meet[x2,join[x1,x3]]]==join[x1,meet[comp[x1],meet[x2,x3]]]`

Proof

We start by taking Critical Pair Lemma 44, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

Substitution Lemma 31

It can be shown that:

`join[x1,meet[x2,join[x1,x3]]]==join[x1,meet[x2,x3]]`

Proof

We start by taking Substitution Lemma 30, and apply the substitution:

`join[x1_,meet[comp[x1_],x2_]]→join[x1,x2]`

which follows from Critical Pair Lemma 28.

Critical Pair Lemma 45

The following expressions are equivalent:

`join[x1,meet[x2,join[x3,x1]]]==join[x1,meet[x2,join[x1,x3]]]`

Proof

Note that the input for the rule:

`join[x1_,meet[x2_,join[x1_,x3_]]]→join[x1,meet[x2,x3]]`

contains a subpattern of the form:

`join[x1_,x3_]`

which can be unified with the input for the rule:

`join[x1_,join[x2_,x1_]]→join[x1,x2]`

where these rules follow from Substitution Lemma 31 and Substitution Lemma 5 respectively.

Substitution Lemma 32

It can be shown that:

`join[x1,meet[x2,join[x3,x1]]]==join[x1,meet[x2,x3]]`

Proof

We start by taking Critical Pair Lemma 45, and apply the substitution:

`join[x1_,meet[x2_,join[x1_,x3_]]]→join[x1,meet[x2,x3]]`

which follows from Substitution Lemma 31.

Critical Pair Lemma 46

The following expressions are equivalent:

`x1==meet[join[comp[x2],x1],join[x1,meet[x3,x2]]]`

Proof

Note that the input for the rule:

`meet[join[x1_,comp[x2_]],join[x1_,meet[x3_,x2_]]]→x1`

contains a subpattern of the form:

`join[x1_,comp[x2_]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Substitution Lemma 26 and Axiom 7 respectively.

Critical Pair Lemma 47

The following expressions are equivalent:

`x1==meet[join[x1,comp[x2]],join[meet[x3,x2],x1]]`

Proof

Note that the input for the rule:

`meet[join[x1_,comp[x2_]],join[x1_,meet[x3_,x2_]]]→x1`

contains a subpattern of the form:

`join[x1_,meet[x3_,x2_]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Substitution Lemma 26 and Axiom 7 respectively.

Critical Pair Lemma 48

The following expressions are equivalent:

`join[comp[join[comp[x1],comp[x2]]],meet[comp[meet[x3,x2]],x1]]==comp[comp[x1]]`

Proof

Note that the input for the rule:

`comp[meet[x1_,join[comp[x2_],x3_]]]→join[comp[x1],meet[comp[x3],x2]]`

contains a subpattern of the form:

`meet[x1_,join[comp[x2_],x3_]]`

which can be unified with the input for the rule:

`meet[join[x1_,comp[x2_]],join[x1_,meet[x3_,x2_]]]→x1`

where these rules follow from Critical Pair Lemma 22 and Substitution Lemma 26 respectively.

Substitution Lemma 33

It can be shown that:

`join[meet[comp[comp[x1]],x2],meet[comp[meet[x3,x1]],x2]]==comp[comp[x2]]`

Proof

We start by taking Critical Pair Lemma 48, and apply the substitution:

`comp[join[comp[x1_],x2_]]→meet[comp[x2],x1]`

which follows from Critical Pair Lemma 17.

Substitution Lemma 34

It can be shown that:

`join[meet[x1,x2],meet[comp[meet[x3,x1]],x2]]==comp[comp[x2]]`

Proof

We start by taking Substitution Lemma 33, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

Substitution Lemma 35

It can be shown that:

`join[meet[x1,x2],meet[comp[meet[x3,x1]],x2]]==x2`

Proof

We start by taking Substitution Lemma 34, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

Critical Pair Lemma 49

The following expressions are equivalent:

`x1==meet[join[x2,x1],join[x1,meet[x3,comp[x2]]]]`

Proof

Note that the input for the rule:

`meet[join[comp[x1_],x2_],join[x2_,meet[x3_,x1_]]]→x2`

contains a subpattern of the form:

`comp[x1_]`

which can be unified with the input for the rule:

`comp[comp[x1_]]→x1`

where these rules follow from Critical Pair Lemma 46 and Axiom 4 respectively.

Critical Pair Lemma 50

The following expressions are equivalent:

`join[meet[x1,x2],x3]==join[x3,meet[comp[join[x3,comp[x2]]],join[meet[x1,x2],x3]]]`

Proof

Note that the input for the rule:

`join[meet[x1_,x2_],meet[comp[x1_],x2_]]→x2`

contains a subpattern of the form:

`meet[x1_,x2_]`

which can be unified with the input for the rule:

`meet[join[x1_,comp[x2_]],join[meet[x3_,x2_],x1_]]→x1`

where these rules follow from Substitution Lemma 19 and Critical Pair Lemma 47 respectively.

where these rules follow from Substitution Lemma 19 and Critical Pair Lemma 47 respectively.

Substitution Lemma 36

It can be shown that:

`join[meet[x1,x2],x3]==join[x3,meet[comp[join[x3,comp[x2]]],meet[x1,x2]]]`

Proof

We start by taking Critical Pair Lemma 50, and apply the substitution:

`join[x1_,meet[x2_,join[x3_,x1_]]]→join[x1,meet[x2,x3]]`

which follows from Substitution Lemma 32.

Substitution Lemma 37

It can be shown that:

`join[meet[x1,x2],x3]==join[x3,meet[meet[comp[x3],x2],meet[x1,x2]]]`

Proof

We start by taking Substitution Lemma 36, and apply the substitution:

`comp[join[x1_,comp[x2_]]]→meet[comp[x1],x2]`

which follows from Critical Pair Lemma 3.

Substitution Lemma 38

It can be shown that:

`join[meet[x1,x2],x3]==join[x3,meet[comp[x3],meet[x2,meet[x1,x2]]]]`

Proof

We start by taking Substitution Lemma 37, and apply the substitution:

`meet[meet[x1_,x2_],x3_]→meet[x1,meet[x2,x3]]`

which follows from Axiom 6.

Substitution Lemma 39

It can be shown that:

`join[meet[x1,x2],x3]==join[x3,meet[x2,meet[x1,x2]]]`

Proof

We start by taking Substitution Lemma 38, and apply the substitution:

`join[x1_,meet[comp[x1_],x2_]]→join[x1,x2]`

which follows from Critical Pair Lemma 28.

Substitution Lemma 40

It can be shown that:

`join[meet[x1,x2],x3]==join[x3,meet[x2,x1]]`

Proof

We start by taking Substitution Lemma 39, and apply the substitution:

`meet[x1_,meet[x2_,x1_]]→meet[x1,x2]`

which follows from Substitution Lemma 2.

Critical Pair Lemma 51

The following expressions are equivalent:

`x1==meet[join[comp[x2],x1],join[meet[x2,x3],x1]]`

PROOF

Note that the input for the rule:

`meet[join[comp[x1_],x2_],join[x2_,meet[x3_,x1_]]]→x2`

contains a subpattern of the form:

`join[x2_,meet[x3_,x1_]]`

which can be unified with the input for the rule:

`join[meet[x1_,x2_],x3_]↔join[x3_,meet[x2_,x1_]]`

where these rules follow from Critical Pair Lemma 46 and Substitution Lemma 40 respectively.

Critical Pair Lemma 52

The following expressions are equivalent:

`x1==join[meet[join[x2,x3],x1],meet[comp[x2],x1]]`

PROOF

Note that the input for the rule:

`join[meet[x1_,x2_],meet[comp[meet[x3_,x1_]],x2_]]→x2`

contains a subpattern of the form:

`meet[x3_,x1_]`

which can be unified with the input for the rule:

`meet[x1_,join[x1_,x2_]]→x1`

where these rules follow from Substitution Lemma 35 and Axiom 1 respectively.

Critical Pair Lemma 53

The following expressions are equivalent:

`x1==meet[join[x2,x1],join[meet[x3,comp[x2]],x1]]`

PROOF

Note that the input for the rule:

`meet[join[x1_,x2_],join[x2_,meet[x3_,comp[x1_]]]]→x2`

contains a subpattern of the form:

`join[x2_,meet[x3_,comp[x1_]]]`

which can be unified with the input for the rule:

`join[x1_,x2_]↔join[x2_,x1_]`

where these rules follow from Critical Pair Lemma 49 and Axiom 7 respectively.

Substitution Lemma 41

It can be shown that:

`x1==join[meet[comp[x2],x1],meet[x1,join[x2,x3]]]`

PROOF

We start by taking Critical Pair Lemma 52, and apply the substitution:

`join[meet[x1_,x2_],x3_]→join[x3,meet[x2,x1]]`

which follows from Substitution Lemma 40.

Critical Pair Lemma 54

The following expressions are equivalent:

```
meet[x1,join[x2,x3]]==meet[join[comp[comp[x2]],meet[x1,join[x2,x3]]],x1]
```

Proof

Note that the input for the rule:

```
meet[join[comp[x1_],x2_],join[meet[x1_,x3_],x2_]]→x2
```

contains a subpattern of the form:

```
join[meet[x1_,x3_],x2_]
```

which can be unified with the input for the rule:

```
join[meet[comp[x1_],x2_],meet[x2_,join[x1_,x3_]]]→x2
```

where these rules follow from Critical Pair Lemma 51 and Substitution Lemma 41 respectively.

Substitution Lemma 42

It can be shown that:

```
meet[x1,join[x2,x3]]==meet[join[x2,meet[x1,join[x2,x3]]],x1]
```

Proof

We start by taking Critical Pair Lemma 54, and apply the substitution:

```
comp[comp[x1_]]→x1
```

which follows from Axiom 4.

Substitution Lemma 43

It can be shown that:

```
meet[x1,join[x2,x3]]==meet[join[x2,meet[x1,x3]],x1]
```

Proof

We start by taking Substitution Lemma 42, and apply the substitution:

```
join[x1_,meet[x2_,join[x1_,x3_]]]→join[x1,meet[x2,x3]]
```

which follows from Substitution Lemma 31.

Substitution Lemma 44

It can be shown that:

```
meet[x1,join[x2,x3]]==meet[x1,join[x2,meet[x1,x3]]]
```

Proof

We start by taking Substitution Lemma 43, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Substitution Lemma 45

It can be shown that:

```
meet[x1,join[x2,x3]]==meet[join[x3,x2],x1]
```

Proof

We start by taking Substitution Lemma 44, and apply the substitution:

`meet[x1_,join[x2_,meet[x1_,x3_]]]→meet[join[x3,x2],x1]`

which follows from Critical Pair Lemma 43.

Critical Pair Lemma 55

The following expressions are equivalent:

`join[meet[x1,comp[comp[x2]]],x3]==meet[join[x2,join[meet[x1,comp[comp[x2]]],x3]],join[x1`

Proof

Note that the input for the rule:

`meet[join[x1_,x2_],join[meet[x3_,comp[x1_]],x2_]]→x2`

contains a subpattern of the form:

`join[meet[x3_,comp[x1_]],x2_]`

which can be unified with the input for the rule:

`join[meet[x1_,x2_],join[meet[x1_,comp[x2_]],x3_]]→join[x1,x3]`

where these rules follow from Critical Pair Lemma 53 and Critical Pair Lemma 6 respectively.

Substitution Lemma 46

It can be shown that:

`join[meet[x1,x2],x3]==meet[join[x2,join[meet[x1,comp[comp[x2]]],x3]],join[x1,x3]]`

Proof

We start by taking Critical Pair Lemma 55, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

Substitution Lemma 47

It can be shown that:

`join[meet[x1,x2],x3]==meet[join[x2,join[meet[x1,x2],x3]],join[x1,x3]]`

Proof

We start by taking Substitution Lemma 46, and apply the substitution:

`comp[comp[x1_]]→x1`

which follows from Axiom 4.

Substitution Lemma 48

It can be shown that:

`join[meet[x1,x2],x3]==meet[join[x2,x3],join[x1,x3]]`

Proof

We start by taking Substitution Lemma 47, and apply the substitution:

`join[x1_,join[meet[x2_,x1_],x3_]]→join[x1,x3]`

which follows from Critical Pair Lemma 14.

Critical Pair Lemma 56

The following expressions are equivalent:

The following expressions are equivalent:

`join[meet[x1,x2],x3]==meet[join[x1,x3],join[x3,x2]]`

PROOF

Note that the input for the rule:

`meet[join[x1_,x2_],join[x3_,x2_]]→join[meet[x3,x1],x2]`

contains a subpattern of the form:

`meet[join[x1_,x2_],join[x3_,x2_]]`

which can be unified with the input for the rule:

`meet[x1_,join[x2_,x3_]]↔meet[join[x3_,x2_],x1_]`

where these rules follow from Substitution Lemma 48 and Substitution Lemma 45 respectively.

Critical Pair Lemma 57

The following expressions are equivalent:

`join[meet[x1,x2],meet[x2,x3]]==meet[x2,join[x1,meet[x2,x3]]]`

PROOF

Note that the input for the rule:

`meet[join[x1_,x2_],join[x3_,x2_]]→join[meet[x3,x1],x2]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[x1_,meet[x1_,x2_]]→x1`

where these rules follow from Substitution Lemma 48 and Axiom 2 respectively.

Substitution Lemma 49

It can be shown that:

`join[meet[x1,x2],meet[x2,x3]]==meet[join[x3,x1],x2]`

PROOF

We start by taking Critical Pair Lemma 57, and apply the substitution:

`meet[x1_,join[x2_,meet[x1_,x3_]]]→meet[join[x3,x2],x1]`

which follows from Critical Pair Lemma 43.

Critical Pair Lemma 58

The following expressions are equivalent:

`join[meet[x1,x2],meet[x3,x1]]==meet[x1,join[meet[x3,x1],x2]]`

PROOF

Note that the input for the rule:

`meet[join[x1_,x2_],join[x2_,x3_]]→join[meet[x1,x3],x2]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[x1_,meet[x2_,x1_]]→x1`

where these rules follow from Critical Pair Lemma 56 and Critical Pair Lemma 8 respectively.

Substitution Lemma 50

It can be shown that:

`join[meet[x1,x2],meet[x3,x1]]==meet[join[x3,x2],x1]`

Proof

We start by taking Critical Pair Lemma 58, and apply the substitution:

`meet[x1_,join[meet[x2_,x1_],x3_]]→meet[join[x2,x3],x1]`

which follows from Substitution Lemma 29.

Critical Pair Lemma 59

The following expressions are equivalent:

`join[meet[x1,x2],meet[x1,x3]]==meet[x1,join[meet[x1,x3],x2]]`

Proof

Note that the input for the rule:

`meet[join[x1_,x2_],join[x2_,x3_]]→join[meet[x1,x3],x2]`

contains a subpattern of the form:

`join[x1_,x2_]`

which can be unified with the input for the rule:

`join[x1_,meet[x1_,x2_]]→x1`

where these rules follow from Critical Pair Lemma 56 and Axiom 2 respectively.

Substitution Lemma 51

It can be shown that:

`join[meet[x1,x2],meet[x1,x3]]==meet[join[x3,x2],x1]`

Proof

We start by taking Critical Pair Lemma 59, and apply the substitution:

`meet[x1_,join[meet[x1_,x2_],x3_]]→meet[join[x2,x3],x1]`

which follows from Critical Pair Lemma 42.

Substitution Lemma 52

It can be shown that:

`meet[join[a,b],join[d,c]]==join[join[meet[a,c],meet[a,d]],join[meet[b,c],meet[b,d]]]`

Proof

We start by taking Hypothesis 1, and apply the substitution:

`join[x1_,x2_]→join[x2,x1]`

which follows from Axiom 7.

Substitution Lemma 53

It can be shown that:

`meet[join[a,b],join[d,c]]==join[join[meet[a,d],meet[a,c]],join[meet[b,c],meet[b,d]]]`

Proof

We start by taking Substitution Lemma 52, and apply the substitution:

We start by taking Substitution Lemma 52, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 54

It can be shown that:

```
meet[join[b,a],join[d,c]]==join[join[meet[a,d],meet[a,c]],join[meet[b,c],meet[b,d]]]
```

Proof

We start by taking Substitution Lemma 53, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 55

It can be shown that:

```
meet[join[b,a],join[d,c]]==join[join[meet[b,c],meet[b,d]],join[meet[a,d],meet[a,c]]]
```

Proof

We start by taking Substitution Lemma 54, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 56

It can be shown that:

```
meet[join[b,a],join[d,c]]==join[meet[b,c],join[meet[b,d],join[meet[a,d],meet[a,c]]]]
```

Proof

We start by taking Substitution Lemma 55, and apply the substitution:

```
join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]
```

which follows from Axiom 8.

Substitution Lemma 57

It can be shown that:

```
meet[join[b,a],join[d,c]]==join[meet[b,c],join[join[meet[a,d],meet[a,c]],meet[b,d]]]
```

Proof

We start by taking Substitution Lemma 56, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 58

It can be shown that:

```
meet[join[b,a],join[d,c]]==join[join[join[meet[a,d],meet[a,c]],meet[b,d]],meet[b,c]]
```

Proof

We start by taking Substitution Lemma 57, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 59

It can be shown that:

```
meet[join[b,a],join[d,c]]⩵join[join[meet[a,d],join[meet[a,c],meet[b,d]]],meet[b,c]]
```

Proof

We start by taking Substitution Lemma 58, and apply the substitution:

```
join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]
```

which follows from Axiom 8.

Substitution Lemma 60

It can be shown that:

```
meet[join[d,c],join[b,a]]⩵join[join[meet[a,d],join[meet[a,c],meet[b,d]]],meet[b,c]]
```

Proof

We start by taking Substitution Lemma 59, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Substitution Lemma 61

It can be shown that:

```
meet[join[d,c],join[b,a]]⩵join[join[meet[a,d],join[meet[a,c],meet[b,d]]],meet[c,b]]
```

Proof

We start by taking Substitution Lemma 60, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Substitution Lemma 62

It can be shown that:

```
meet[join[d,c],join[b,a]]⩵join[join[meet[a,d],join[meet[a,c],meet[d,b]]],meet[c,b]]
```

Proof

We start by taking Substitution Lemma 61, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Substitution Lemma 63

It can be shown that:

```
meet[join[d,c],join[b,a]]⩵join[join[meet[a,d],join[meet[c,a],meet[d,b]]],meet[c,b]]
```

Proof

We start by taking Substitution Lemma 62, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Substitution Lemma 64

Substitution Lemma 64

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[join[meet[a,d],join[meet[d,b],meet[c,a]]],meet[c,b]]
```

Proof

We start by taking Substitution Lemma 63, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 65

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[join[meet[d,a],join[meet[d,b],meet[c,a]]],meet[c,b]]
```

Proof

We start by taking Substitution Lemma 64, and apply the substitution:

```
meet[x1_,x2_]→meet[x2,x1]
```

which follows from Substitution Lemma 1.

Substitution Lemma 66

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[meet[d,a],join[join[meet[d,b],meet[c,a]],meet[c,b]]]
```

Proof

We start by taking Substitution Lemma 65, and apply the substitution:

```
join[join[x1_,x2_],x3_]→join[x1,join[x2,x3]]
```

which follows from Axiom 8.

Substitution Lemma 67

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[meet[d,a],join[meet[c,b],join[meet[d,b],meet[c,a]]]]
```

Proof

We start by taking Substitution Lemma 66, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Substitution Lemma 68

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[meet[d,a],join[meet[d,b],join[meet[c,b],meet[c,a]]]]
```

Proof

We start by taking Substitution Lemma 67, and apply the substitution:

```
join[x1_,join[x2_,x3_]]→join[x2,join[x1,x3]]
```

which follows from Substitution Lemma 22.

Substitution Lemma 69

It can be shown that:

`meet[join[d,c],join[b,a]]==join[meet[d,b],join[meet[d,a],join[meet[c,b],meet[c,a]]]]`

Proof

We start by taking Substitution Lemma 68, and apply the substitution:

`join[x1_,join[x2_,x3_]]→join[x2,join[x1,x3]]`

which follows from Substitution Lemma 22.

Substitution Lemma 70

It can be shown that:

`meet[join[d,c],join[b,a]]==join[meet[d,b],join[meet[d,a],meet[join[a,b],c]]]`

Proof

We start by taking Substitution Lemma 69, and apply the substitution:

`join[meet[x1_,x2_],meet[x1_,x3_]]→meet[join[x3,x2],x1]`

which follows from Substitution Lemma 51.

Substitution Lemma 71

It can be shown that:

`meet[join[d,c],join[b,a]]==join[meet[d,b],join[meet[d,a],meet[join[b,a],c]]]`

Proof

We start by taking Substitution Lemma 70, and apply the substitution:

`join[x1_,x2_]→join[x2,x1]`

which follows from Axiom 7.

Substitution Lemma 72

It can be shown that:

`meet[join[d,c],join[b,a]]==join[meet[d,b],join[meet[a,d],meet[join[b,a],c]]]`

Proof

We start by taking Substitution Lemma 71, and apply the substitution:

`meet[x1_,x2_]→meet[x2,x1]`

which follows from Substitution Lemma 1.

Substitution Lemma 73

It can be shown that:

`meet[join[d,c],join[b,a]]==join[meet[d,b],join[meet[a,d],meet[c,join[b,a]]]]`

Proof

We start by taking Substitution Lemma 72, and apply the substitution:

`meet[x1_,x2_]→meet[x2,x1]`

which follows from Substitution Lemma 1.

Substitution Lemma 74

It can be shown that:

`meet[join[d,c],join[b,a]]==join[meet[c,join[b,a]],join[meet[d,b],meet[a,d]]]`

Proof

PROOF

We start by taking Substitution Lemma 73, and apply the substitution:

```
join[x1_,join[x2_,x3_]]→join[x3,join[x1,x2]]
```

which follows from Substitution Lemma 21.

Substitution Lemma 75

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[meet[c,join[b,a]],meet[join[a,b],d]]
```

Proof

We start by taking Substitution Lemma 74, and apply the substitution:

```
join[meet[x1_,x2_],meet[x3_,x1_]]→meet[join[x3,x2],x1]
```

which follows from Substitution Lemma 50.

Substitution Lemma 76

It can be shown that:

```
meet[join[d,c],join[b,a]]==join[meet[c,join[b,a]],meet[join[b,a],d]]
```

Proof

We start by taking Substitution Lemma 75, and apply the substitution:

```
join[x1_,x2_]→join[x2,x1]
```

which follows from Axiom 7.

Conclusion 1

We obtain the conclusion:

```
True
```

Proof

Take Substitution Lemma 76, and apply the substitution:

```
join[meet[x1_,x2_],meet[x2_,x3_]]→meet[join[x3,x1],x2]
```

which follows from Substitution Lemma 49.

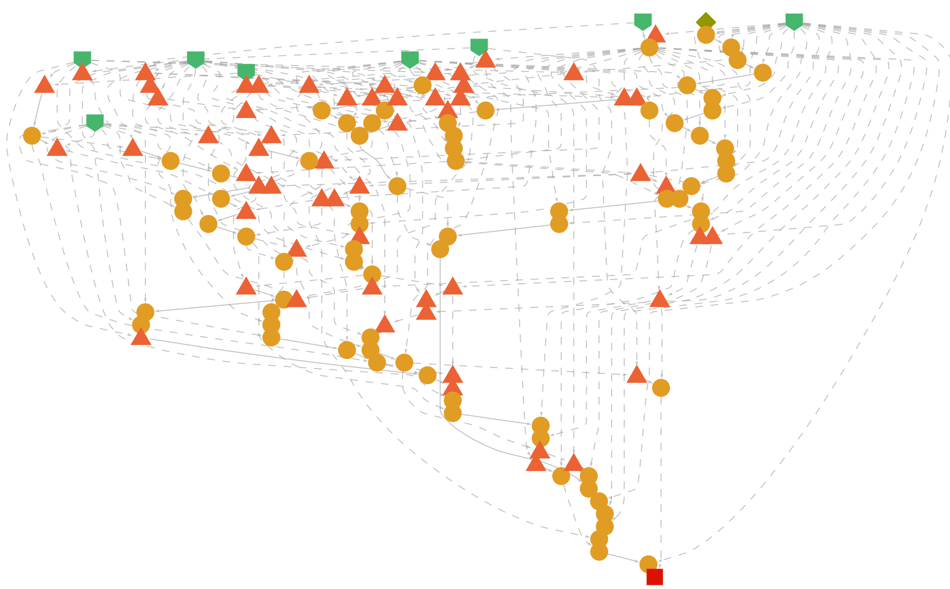
In the Mathematica notebook version of this file, the proposition in the textual representation of the proof that corresponds to a given node in the graph below can be obtained interactively by placing the cursor over the node symbol in the graph. The pdf translation of the Mathematica notebook does not provide this kind of interactive translation.

The symbology of the proof graph is:

a turquoise pentagon represents an axiom
an olive-green diamond represents a hypothesis
a red triangle represents a critical pair lemma
an ochre circle represents a substitution lemma
a line represents a dependency in an inference

In[96]:= `proofMHTheorem["ProofGraph"]`

Out[96]=



5.0 References

[1] von Neumann J. (1936). *Mathematical Foundations of Quantum Mechanics*. Translated by R. T. Beyer, 1983. Princeton.

[2] Wolfram Research. (2018). *Mathematica v11.3 Home Edition*. http://www.wolfram.com/mathematica-home-edition/. Accessed 30 October 2018.

[3] Horner JK. (2018). *MHT Mathematica notebook file.* Available from the author on request.

[4] Dalla Chiara ML and Giuntini R. (2004). *Quantum Logics*. URL http://xxx.lanl.gov/abs/quant-ph/0101028.

[5] Megill ND and Pavičić M. (2001). Orthomodular lattices and quantum algebra. *International Journal of Theoretical Physics* 40, pp. 1387-1410.

[6] Akhiezer NI and Glazman IM. (1961). *Theory of Linear Operators in Hilbert Space.* Volume I. Translated by M. Nestell. Frederick Ungar.

[7] Holland, Jr. SS Orthomodularity in infinite dimensions: a theorem of M. Solèr. (1995). *Bulletin of the American Mathematical Society* 32, pp. 205-234.

[8] Marsden EL and Herman LM. (1974). A condition for distribution in orthomodular lattices. *Kansas State University Technical Report #40*.

[9] Knuth DE and Bendix PB. (1970). Simple word problems in universal algebras. In J. Leech, ed. *Computational Problems in Abstract Algebra*. Pergamon Press. pp. 263-297.

[10] Chang CC and Keisler HJ. (2012). *Model Theory.* Third Edition. North-Holland.

[11] Birkhoff G. (1967). *Lattice Theory*. Third Edition. American Mathematical Society.

[12] Church A. (1956). *Introduction to Mathematical Logic.* Volume I. Princeton.

[13] Jauch J. (1968). *Foundations of Quantum Mechanics*. Addison-Wesley.

[14] Megill ND. (2004). *Metamath.* http://us.metamath.org/qlegif/mmql.html#unify. Accessed 30 October 2018.

[15] Horner JK. (2005). An automated deduction system for orthomodular lattice theory. *Proceedings of the 2005 International Conference on Artificial Intelligence.* CSREA Press. pp. 260-265.

[16] Horner JK. (2007). An automated equational logic deduction of join elimination in orthomodular lattice theory. *Proceedings of the 2007 International Conference on Artificial Intelligence.* CSREA Press. pp. 481-488.

[17] Messiah A. (1958). *Quantum Mechanics.* Dover.

[18] Horner JK. (2002). Using automated theorem-provers to aid the design of efficient compilers for quantum computing. *Los Alamos National Laboratory Quantum Institute Workshop.* December 9-10, 2002.

[19] Birkhoff G and von Neumann J. (1936). The logic of quantum mechanics. *Annals of Mathematics* 37, 823-243.

[20] Nielsen MA and Chuang L. (2000). *Quantum Computation and Quantum Information.* Cambridge.

[21] Pavičić M and Megill N. (1998). Quantum and classical implicational algebras with primitive implication. *International Journal of Theoretical Physics* 37 (1998), 2091-2098. ftp://m3k.grad.hr/pavicic/quantum-logic/1998-int-j-theor-phys-2.ps.gz. Accessed 30 October 2018.

[22] Horner JK. (2009). An automated deduction of the relative strength of orthomodular and weakly orthomodular lattice theory. *Proceedings of the 2009 International Conference on Artificial Intelligence.* CSREA Press. pp. 525-530.

[23] Baader F and Nipkow T. (1999). *Term Rewriting and All That.* Cambridge University Press.

[24] Knuth DE and Bendix PB. (1970). Simple Word Problems in Universal Algebra. In *Computational Problems in Abstract Algebra.* Pergamon Press. pp. 263-297.

[25] Grätzer G. (2008). *Universal Algebra.* Second Edition. Springer.