# Computational Investigations of the Lattice of Lattice Varieties

Peter Jipsen

Chapman University, Orange, California

November 18, 2011

# Outline

What is a lattice?

# Outline

What is a lattice?

What is a lattice variety?

# Outline

What is a lattice?

What is a lattice variety?

What is the lattice of lattice varieties?

# Outline

What is a lattice?

What is a lattice variety?

What is the lattice of lattice varieties?

Computing what it looks like

# Outline

What is a lattice?

What is a lattice variety?

What is the lattice of lattice varieties?

Computing what it looks like

Future work

# What is a lattice?

$(P, \leq)$ is a *partially ordered set* (or poset) if $P$ is a set and $\leq$ is a reflexive, anti-symmetric, transitive relation on $P$

# What is a lattice?

$(P, \leq)$ is a *partially ordered set* (or poset) if $P$ is a set and $\leq$ is a reflexive, anti-symmetric, transitive relation on $P$

E.g. $(\mathbb{Z}, \leq)$ or $(\mathbb{N}, |)$ or $(\mathcal{P}(X), \subseteq)$

# What is a lattice?

$(P, \leq)$ is a *partially ordered set* (or poset) if $P$ is a set and $\leq$ is a reflexive, anti-symmetric, transitive relation on $P$

E.g. $(\mathbb{Z}, \leq)$ or $(\mathbb{N}, |)$ or $(\mathcal{P}(X), \subseteq)$

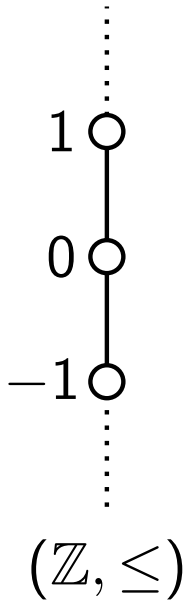
The nice thing about posets is we can draw pictures:

# What is a lattice?

$(P, \leq)$ is a *partially ordered set* (or poset) if $P$ is a set and $\leq$ is a reflexive, anti-symmetric, transitive relation on $P$

E.g. $(\mathbb{Z}, \leq)$ or $(\mathbb{N}, |)$ or $(\mathcal{P}(X), \subseteq)$

The nice thing about posets is we can draw pictures:



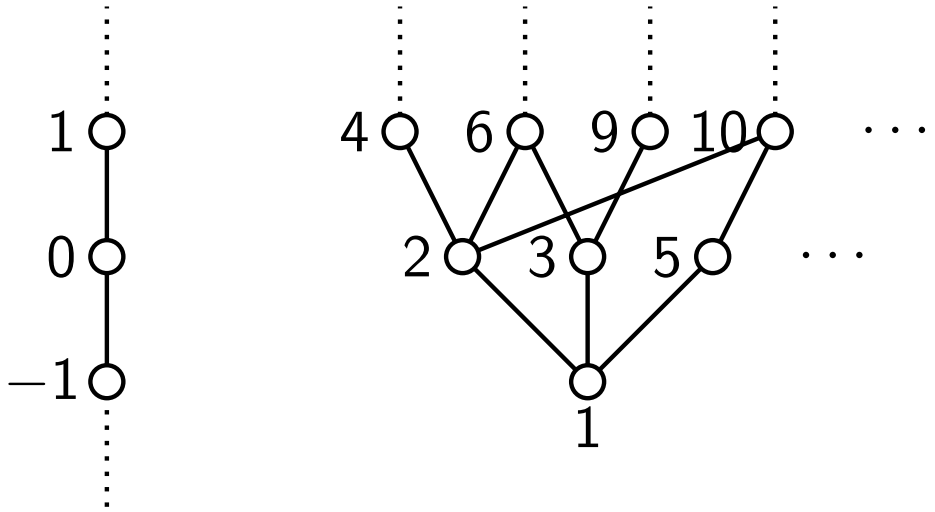

$(\mathbb{Z}, \leq)$

# What is a lattice?

$(P, \leq)$ is a *partially ordered set* (or poset) if $P$ is a set and $\leq$ is a reflexive, anti-symmetric, transitive relation on $P$

E.g. $(\mathbb{Z}, \leq)$ or $(\mathbb{N}, |)$ or $(\mathcal{P}(X), \subseteq)$

The nice thing about posets is we can draw pictures:

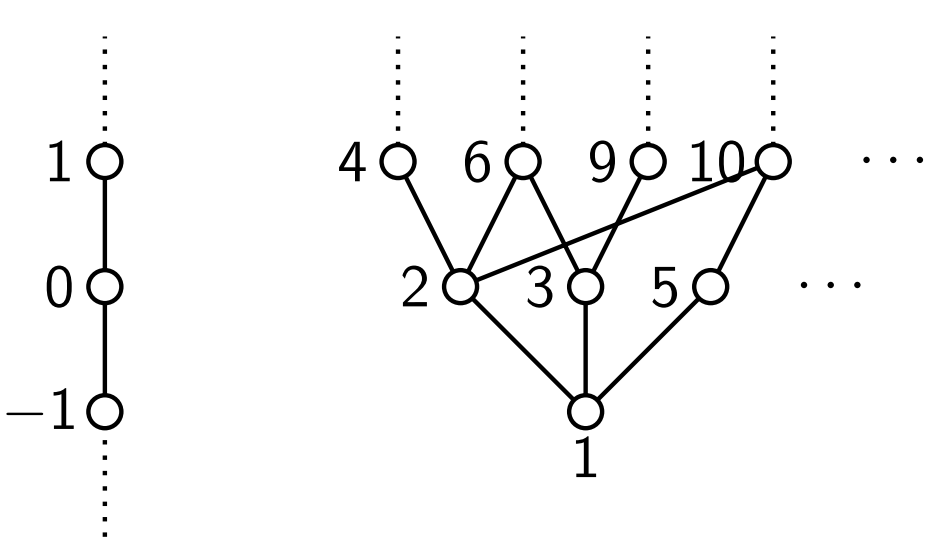


# What is a lattice?

$(P, \leq)$ is a *partially ordered set* (or poset) if $P$ is a set and $\leq$ is a reflexive, anti-symmetric, transitive relation on $P$

E.g. $(\mathbb{Z}, \leq)$ or $(\mathbb{N}, |)$ or $(\mathcal{P}(X), \subseteq)$

The nice thing about posets is we can draw pictures:




$(\mathbb{Z}, \leq)$ $(\mathbb{N}, |)$ $\mathcal{P}(\{1, 2, 3\})$

# What is a lattice?

Lattices are an algebraic version of (special) posets

# What is a lattice?

Lattices are an algebraic version of (special) posets

A *lattice* $L$ is a poset in which all $x, y$ have a *meet* and *join*, i.e.
a *greatest element below or equal to* them, denoted $x \wedge y$ and
a *smallest element above or equal to* them, denoted $x \vee y$

# What is a lattice?

Lattices are an algebraic version of (special) posets

A *lattice* $L$ is a poset in which all $x, y$ have a *meet* and *join*, i.e.
a *greatest element below or equal to* them, denoted $x \wedge y$ and
a *smallest element above or equal to* them, denoted $x \vee y$

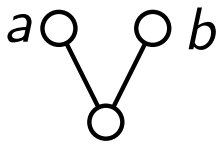
In a lattice $x \leq y$ if and only if $x \wedge y = x$ (iff $x \vee y = y$)

# What is a lattice?

Lattices are an algebraic version of (special) posets

A *lattice* $L$ is a poset in which all $x, y$ have a *meet* and *join*, i.e.
a *greatest element below or equal to* them, denoted $x \wedge y$ and
a *smallest element above or equal to* them, denoted $x \vee y$

In a lattice $x \leq y$ if and only if $x \wedge y = x$ (iff $x \vee y = y$)



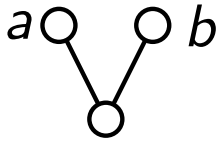

not a lattice

# What is a lattice?

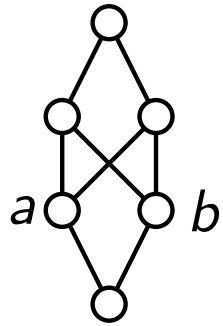
Lattices are an algebraic version of (special) posets

A *lattice* $L$ is a poset in which all $x, y$ have a *meet* and *join*, i.e.
a *greatest element below or equal to* them, denoted $x \wedge y$ and
a *smallest element above or equal to* them, denoted $x \vee y$

In a lattice $x \leq y$ if and only if $x \wedge y = x$ (iff $x \vee y = y$)




not a lattice



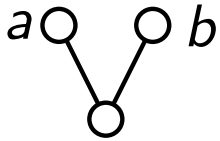

not a lattice

# What is a lattice?

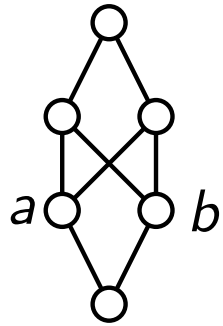
Lattices are an algebraic version of (special) posets

A *lattice* $L$ is a poset in which all $x, y$ have a *meet* and *join*, i.e.
a *greatest element below or equal to* them, denoted $x \wedge y$ and
a *smallest element above or equal to* them, denoted $x \vee y$

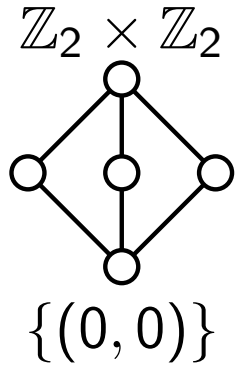
In a lattice $x \leq y$ if and only if $x \wedge y = x$ (iff $x \vee y = y$)




not a lattice




not a lattice




$M_3$=a subgroup lattice

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

A lattice *satisfies* an equation if the equation is true for all elements of the lattice. In that case the lattice is a *model* of the equation.

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

A lattice *satisfies* an equation if the equation is true for all elements of the lattice. In that case the lattice is a *model* of the equation.

Let $E$ be a set of lattice equations, $\mathcal{K}$ a class of lattices

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

A lattice *satisfies* an equation if the equation is true for all elements of the lattice. In that case the lattice is a *model* of the equation.

Let $E$ be a set of lattice equations, $\mathcal{K}$ a class of lattices

$\mathrm{Mod}(E)$ is the class of lattices that satisfies all the equations of $E$

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

A lattice *satisfies* an equation if the equation is true for all elements of the lattice. In that case the lattice is a *model* of the equation.

Let $E$ be a set of lattice equations, $\mathcal{K}$ a class of lattices

$\mathrm{Mod}(E)$ is the class of lattices that satisfies all the equations of $E$

$\mathrm{Eq}(\mathcal{K})$ is the set of equations that are satisfied by all lattices in $\mathcal{K}$

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

A lattice *satisfies* an equation if the equation is true for all elements of the lattice. In that case the lattice is a *model* of the equation.

Let $E$ be a set of lattice equations, $\mathcal{K}$ a class of lattices

$\mathrm{Mod}(E)$ is the class of lattices that satisfies all the equations of $E$

$\mathrm{Eq}(\mathcal{K})$ is the set of equations that are satisfied by all lattices in $\mathcal{K}$

A *lattice variety* is any collection of lattices of the form $\mathrm{Mod}(E)$

# What is a lattice variety?

A *lattice equation* is an expression with variables $x, y, z, \ldots$ and $\wedge, \vee, =$

E.g. $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ is the *distributive law*

A lattice *satisfies* an equation if the equation is true for all elements of the lattice. In that case the lattice is a *model* of the equation.

Let $E$ be a set of lattice equations, $\mathcal{K}$ a class of lattices

$\mathrm{Mod}(E)$ is the class of lattices that satisfies all the equations of $E$

$\mathrm{Eq}(\mathcal{K})$ is the set of equations that are satisfied by all lattices in $\mathcal{K}$

A *lattice variety* is any collection of lattices of the form $\mathrm{Mod}(E)$

E.g. $\mathcal{D} = \mathrm{Mod}(\{x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)\})$ is the variety of distributive lattices

For a class $\mathcal{K}$ of lattices, $\mathrm{Mod}(\mathrm{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

For a class $\mathcal{K}$ of lattices, $\mathrm{Mod}(\mathrm{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathrm{V}(\mathcal{K})$

For a class $\mathcal{K}$ of lattices, $\mathsf{Mod}(\mathsf{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathsf{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathsf{V}(\{$  $\})$

For a class $\mathcal{K}$ of lattices, $\mathsf{Mod}(\mathsf{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathsf{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathsf{V}(\{$  $\})$

The variety $\mathcal{O}$ of one-element lattices is determined by the equation $x = y$

For a class $\mathcal{K}$ of lattices, $\mathsf{Mod}(\mathsf{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathsf{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathsf{V}(\{$  $\})$

The variety $\mathcal{O}$ of one-element lattices is determined by the equation $x = y$

The variety $\mathcal{L}$ of all lattices is determined by the equation $x = x$

For a class $\mathcal{K}$ of lattices, $\mathsf{Mod}(\mathsf{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathsf{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathsf{V}(\{$  $\})$

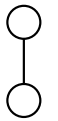
The variety $\mathcal{O}$ of one-element lattices is determined by the equation $x = y$

The variety $\mathcal{L}$ of all lattices is determined by the equation $x = x$

There are only countably many equations, and a lattice variety is given by a set of equations, so there are at most $2^{\mathbb{N}}$ many lattice varieties

For a class $\mathcal{K}$ of lattices, $\mathsf{Mod}(\mathsf{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathsf{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathsf{V}(\{$  $\})$

The variety $\mathcal{O}$ of one-element lattices is determined by the equation $x = y$

The variety $\mathcal{L}$ of all lattices is determined by the equation $x = x$

There are only countably many equations, and a lattice variety is given by a set of equations, so there are at most $2^{\mathbb{N}}$ many lattice varieties

Kirby Baker [1969] proved there are exactly $2^{\mathbb{N}}$ (modular) lattice varieties

For a class $\mathcal{K}$ of lattices, $\mathrm{Mod}(\mathrm{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathrm{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathrm{V}(\{$  $\})$

The variety $\mathcal{O}$ of one-element lattices is determined by the equation $x = y$

The variety $\mathcal{L}$ of all lattices is determined by the equation $x = x$

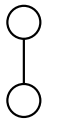
There are only countably many equations, and a lattice variety is given by a set of equations, so there are at most $2^{\mathbb{N}}$ many lattice varieties

Kirby Baker [1969] proved there are exactly $2^{\mathbb{N}}$ (modular) lattice varieties

A lattice is *modular* if it satisfies $x \wedge (y \vee (x \wedge z)) = (x \wedge y) \vee (x \wedge z)$

For a class $\mathcal{K}$ of lattices, $\mathrm{Mod}(\mathrm{Eq}(\mathcal{K}))$ is the smallest variety containing $\mathcal{K}$

This is the variety *generated by* $\mathcal{K}$, denoted $\mathrm{V}(\mathcal{K})$

E.g. $\mathcal{D} = \mathrm{V}(\{$  $\})$

The variety $\mathcal{O}$ of one-element lattices is determined by the equation $x = y$

The variety $\mathcal{L}$ of all lattices is determined by the equation $x = x$

There are only countably many equations, and a lattice variety is given by a set of equations, so there are at most $2^{\mathbb{N}}$ many lattice varieties

Kirby Baker [1969] proved there are exactly $2^{\mathbb{N}}$ (modular) lattice varieties

A lattice is *modular* if it satisfies $x \wedge (y \vee (x \wedge z)) = (x \wedge y) \vee (x \wedge z)$

Lattices of vector subspaces, lattices of normal subgroups, ... all modular

# What is the lattice of lattice varieties?

Lattice varieties are ordered by subset-inclusion

# What is the lattice of lattice varieties?

Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

# What is the lattice of lattice varieties?

Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

The set of all lattice varieties is a complete lattice: $\mathcal{V} \wedge \mathcal{W} = \mathcal{V} \cap \mathcal{W}$

# What is the lattice of lattice varieties?

Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

The set of all lattice varieties is a complete lattice: $\mathcal{V} \wedge \mathcal{W} = \mathcal{V} \cap \mathcal{W}$

$\mathcal{O}$ is the smallest variety, $\mathcal{L}$ is the largest variety

# What is the lattice of lattice varieties?

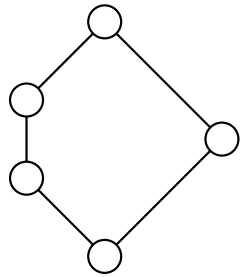
Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

The set of all lattice varieties is a complete lattice: $\mathcal{V} \wedge \mathcal{W} = \mathcal{V} \cap \mathcal{W}$

$\mathcal{O}$ is the smallest variety, $\mathcal{L}$ is the largest variety

$\mathcal{D}$ is the next smallest variety



# What is the lattice of lattice varieties?

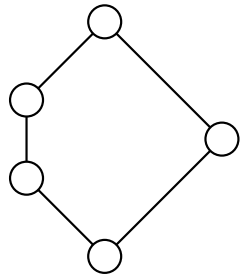
Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

The set of all lattice varieties is a complete lattice: $\mathcal{V} \wedge \mathcal{W} = \mathcal{V} \cap \mathcal{W}$

$\mathcal{O}$ is the smallest variety, $\mathcal{L}$ is the largest variety

$\mathcal{D}$ is the next smallest variety




A *sublattice* of a lattice is a subset that is closed under $\wedge, \vee$

# What is the lattice of lattice varieties?

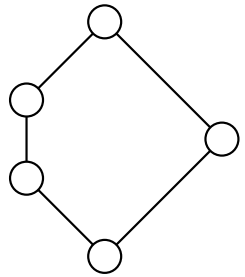
Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

The set of all lattice varieties is a complete lattice: $\mathcal{V} \wedge \mathcal{W} = \mathcal{V} \cap \mathcal{W}$

$\mathcal{O}$ is the smallest variety, $\mathcal{L}$ is the largest variety

$\mathcal{D}$ is the next smallest variety




A *sublattice* of a lattice is a subset that is closed under $\wedge, \vee$

Richard Dedekind [1900]: every nonmodular lattice contains $N_5$ as a sublattice

# What is the lattice of lattice varieties?

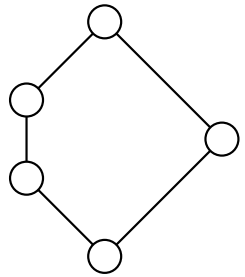
Lattice varieties are ordered by subset-inclusion

$\mathcal{V}$ is a *subvariety* of $\mathcal{W}$ if $\mathcal{V} \subseteq \mathcal{W}$

The set of all lattice varieties is a complete lattice: $\mathcal{V} \wedge \mathcal{W} = \mathcal{V} \cap \mathcal{W}$

$\mathcal{O}$ is the smallest variety, $\mathcal{L}$ is the largest variety

$\mathcal{D}$ is the next smallest variety




A *sublattice* of a lattice is a subset that is closed under $\wedge, \vee$

Richard Dedekind [1900]: every nonmodular lattice contains $N_5$ as a sublattice

Garrett Birkhoff [1935]: every nondistributive modular lattice contains $M_3$ as a sublattice

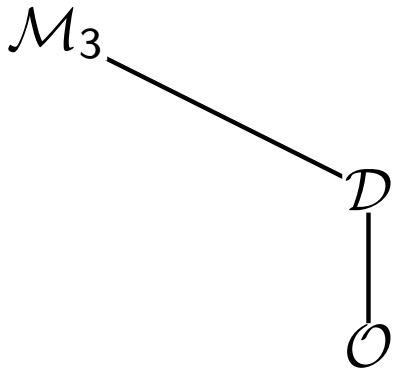
# Computing what the lattice of lattice varieties looks like

The lattice of lattice varieties

# Computing what the lattice of lattice varieties looks like

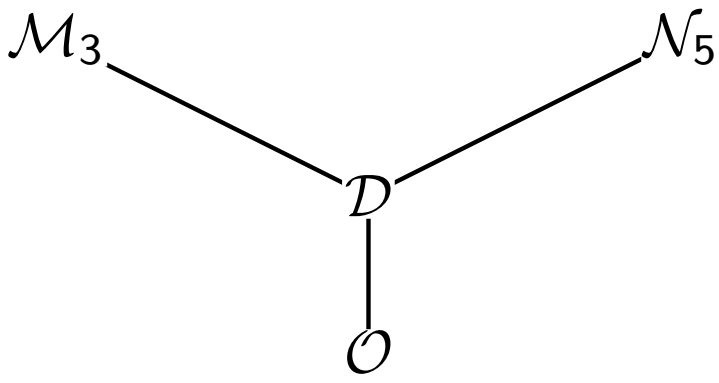

The lattice of lattice varieties

# Computing what the lattice of lattice varieties looks like



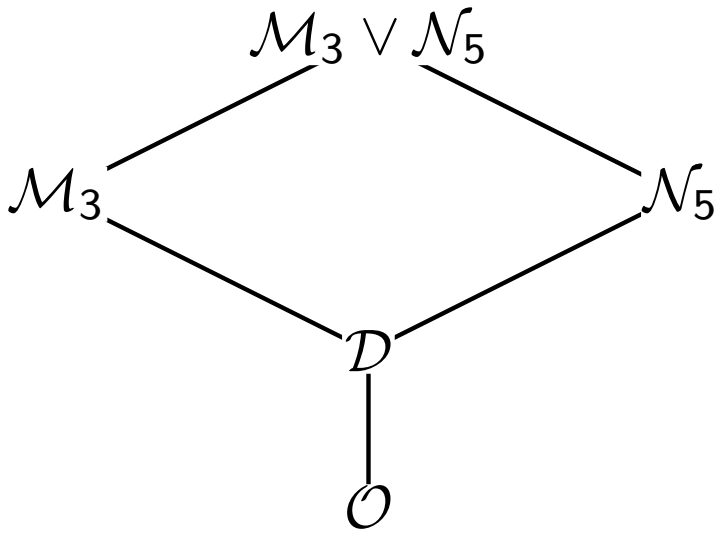

The lattice of lattice varieties

# Computing what the lattice of lattice varieties looks like



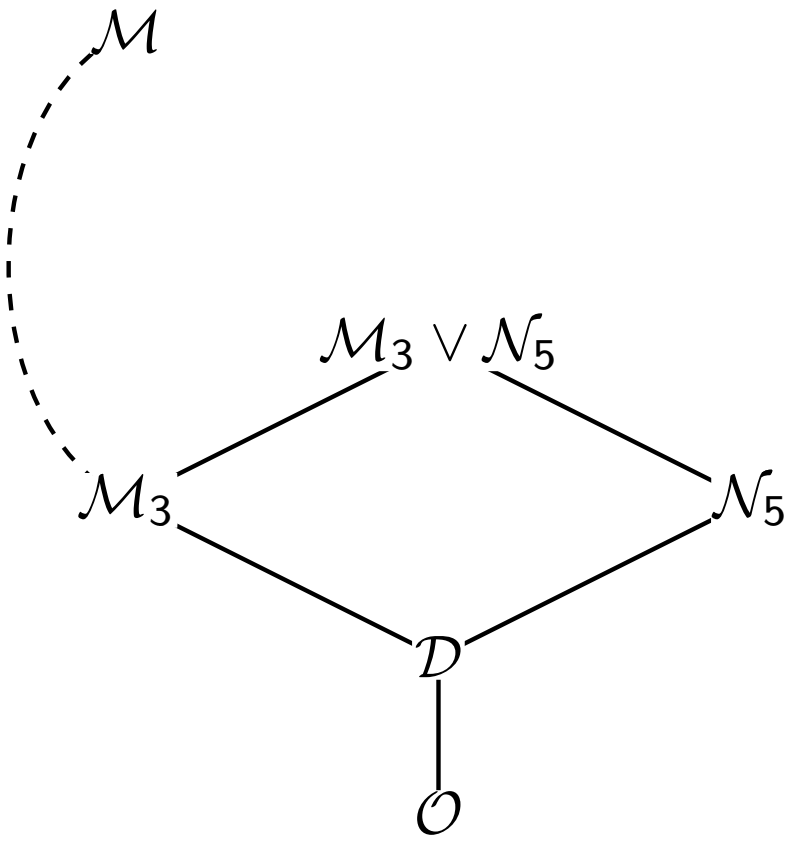

The lattice of lattice varieties

# Computing what the lattice of lattice varieties looks like



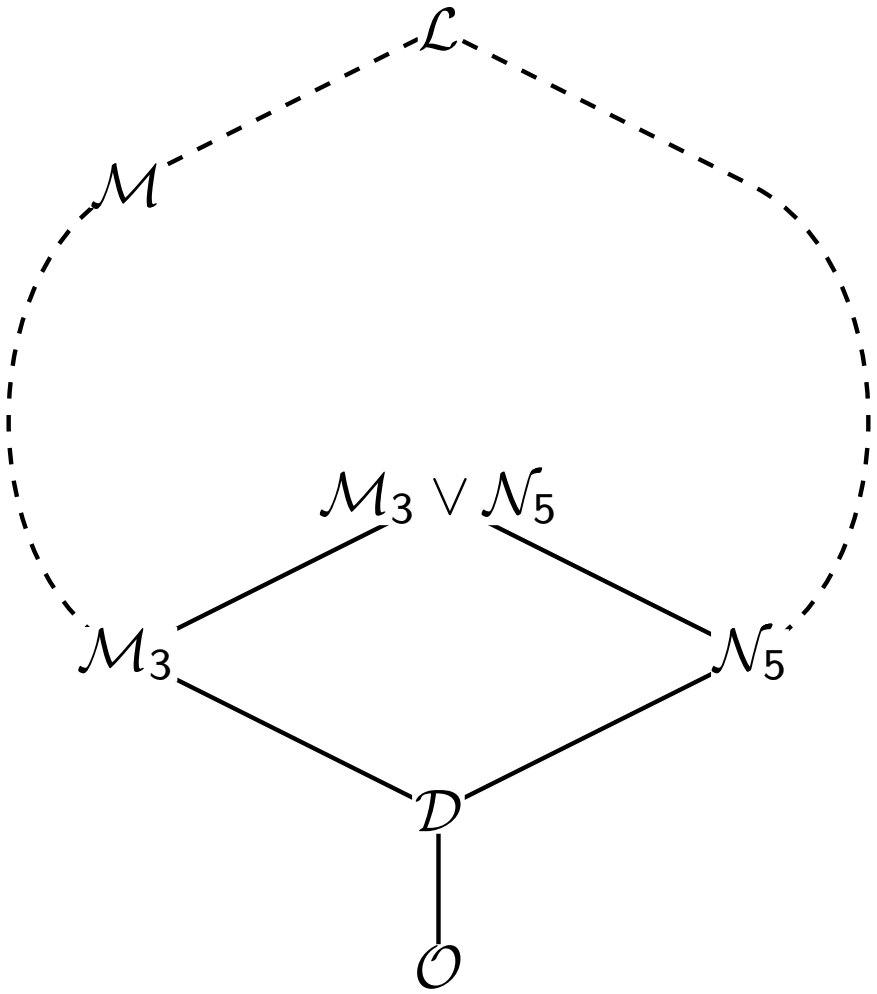

The lattice of lattice varieties

# Computing what the lattice of lattice varieties looks like




The lattice of lattice varieties

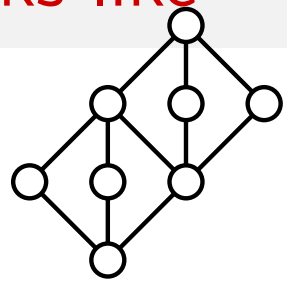
# Computing what the lattice of lattice varieties looks like



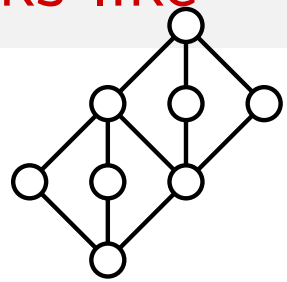

The lattice of lattice varieties

# Computing what the lattice of lattice varieties looks like

George Grätzer [1966]: any finitely generated modular variety strictly above $\mathcal{M}_3$ is above $\mathcal{M}_4$ or $\mathcal{M}_{3^2}$ (or both)



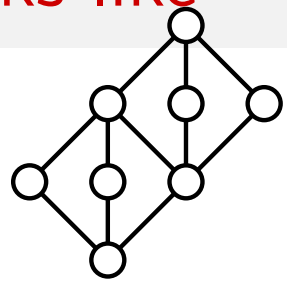
# Computing what the lattice of lattice varieties looks like



George Grätzer [1966]: any finitely generated modular variety strictly above $\mathcal{M}_3$ is above $\mathcal{M}_4$ or $\mathcal{M}_{3^2}$ (or both)

Bjarni Jónsson [1968] removed the restriction that the variety be finitely generated by proving the following result:

# Computing what the lattice of lattice varieties looks like



George Grätzer [1966]: any finitely generated modular variety strictly above $\mathcal{M}_3$ is above $\mathcal{M}_4$ or $\mathcal{M}_{3^2}$ (or both)

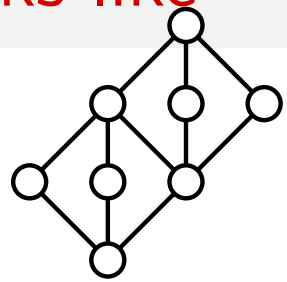
Bjarni Jónsson [1968] removed the restriction that the variety be finitely generated by proving the following result:

**Theorem**

*For a modular variety $\mathcal{V}$ the following conditions are equivalent:*

1. $M_{3^2} \notin \mathcal{V}$,
2. *every subdirectly irreducible member of $\mathcal{V}$ has length $\leq 2$,*
3. *the inequality $x \wedge (y \vee (z \wedge w)) \wedge (z \vee w) \leq y \vee (x \wedge z) \vee (x \wedge w)$ holds in $\mathcal{V}$.*

# Computing what the lattice of lattice varieties looks like



George Grätzer [1966]: any finitely generated modular variety strictly above $\mathcal{M}_3$ is above $\mathcal{M}_4$ or $\mathcal{M}_{3^2}$ (or both)

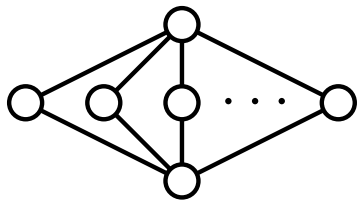
Bjarni Jónsson [1968] removed the restriction that the variety be finitely generated by proving the following result:

**Theorem**

*For a modular variety $\mathcal{V}$ the following conditions are equivalent:*

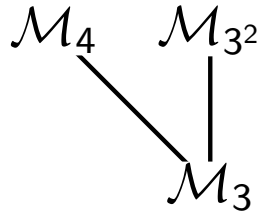
1. $M_{3^2} \notin \mathcal{V}$,
2. *every subdirectly irreducible member of $\mathcal{V}$ has length $\leq 2$,*
3. *the inequality $x \wedge (y \vee (z \wedge w)) \wedge (z \vee w) \leq y \vee (x \wedge z) \vee (x \wedge w)$ holds in $\mathcal{V}$.*

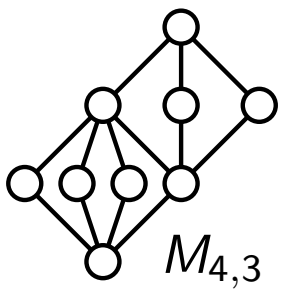
The only lattices of length 2 are $M_n$

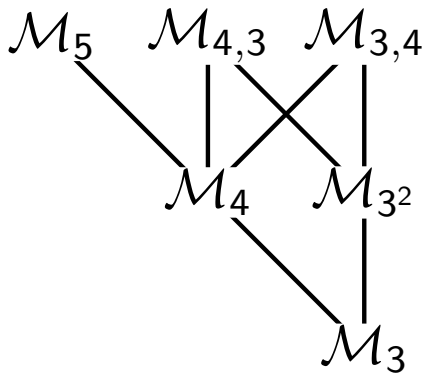


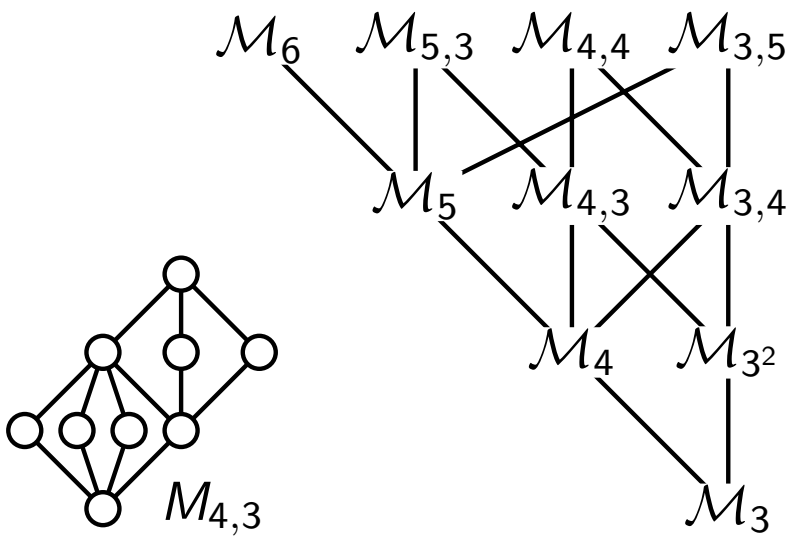
($n+2$ elements)

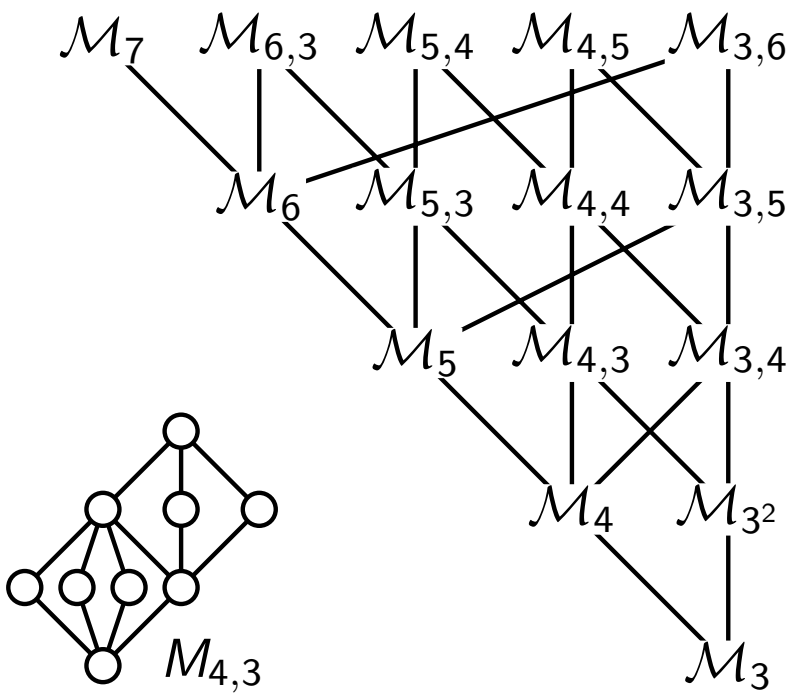
$\mathcal{M}_3$

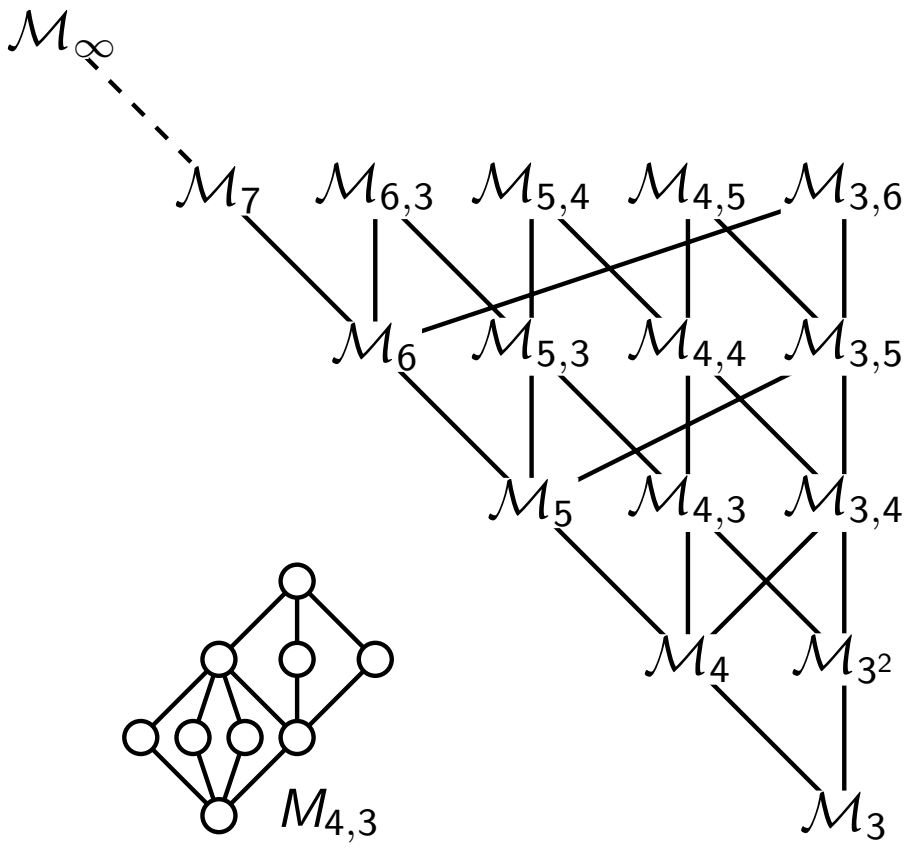

$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$

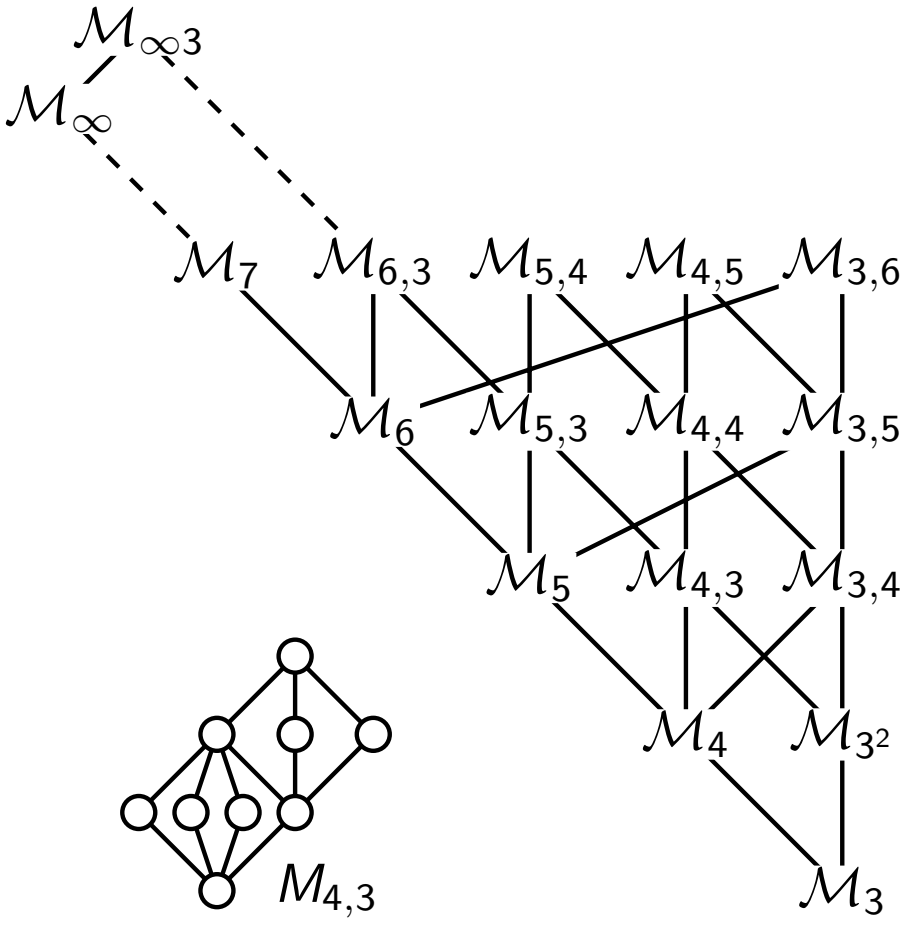

$M_{4,3}$

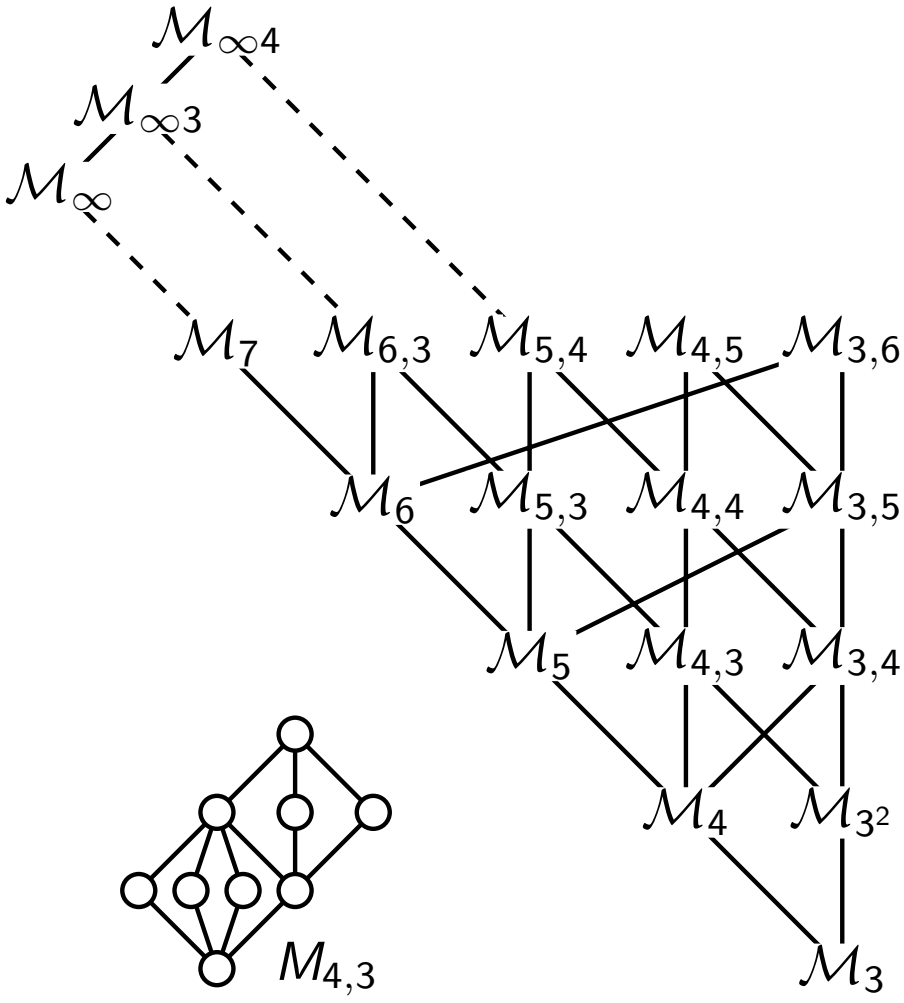

$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$

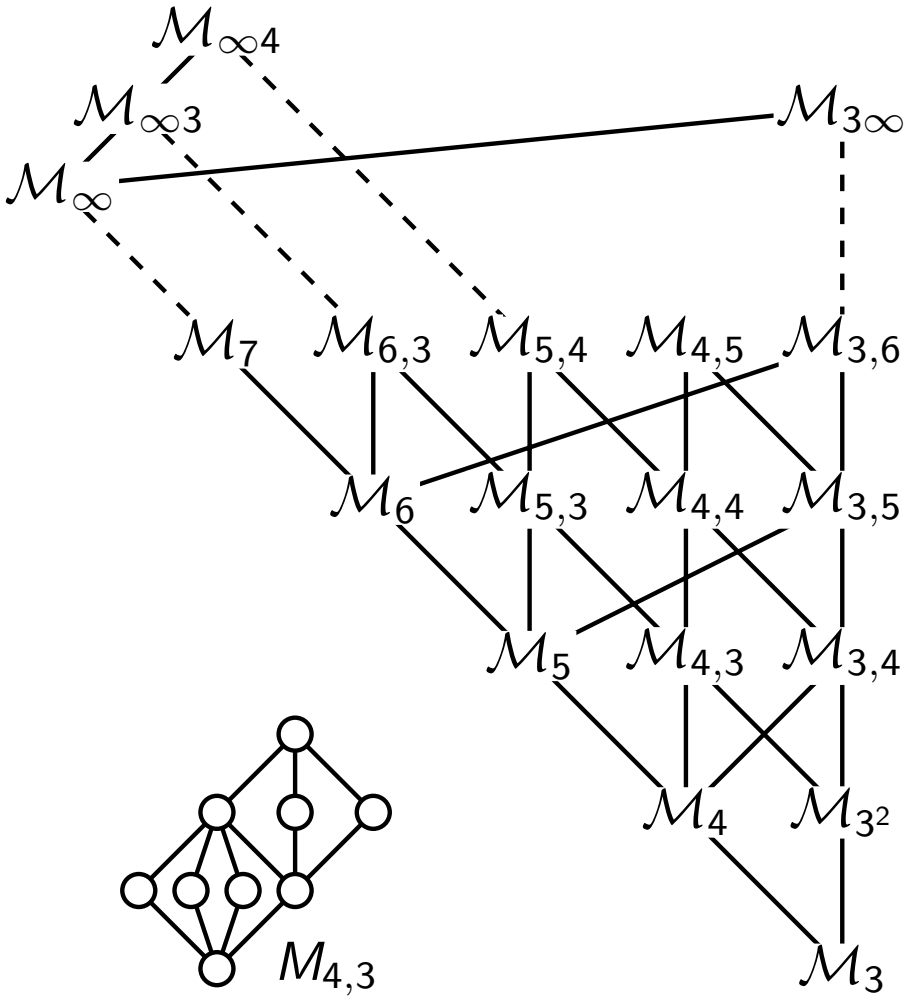

$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

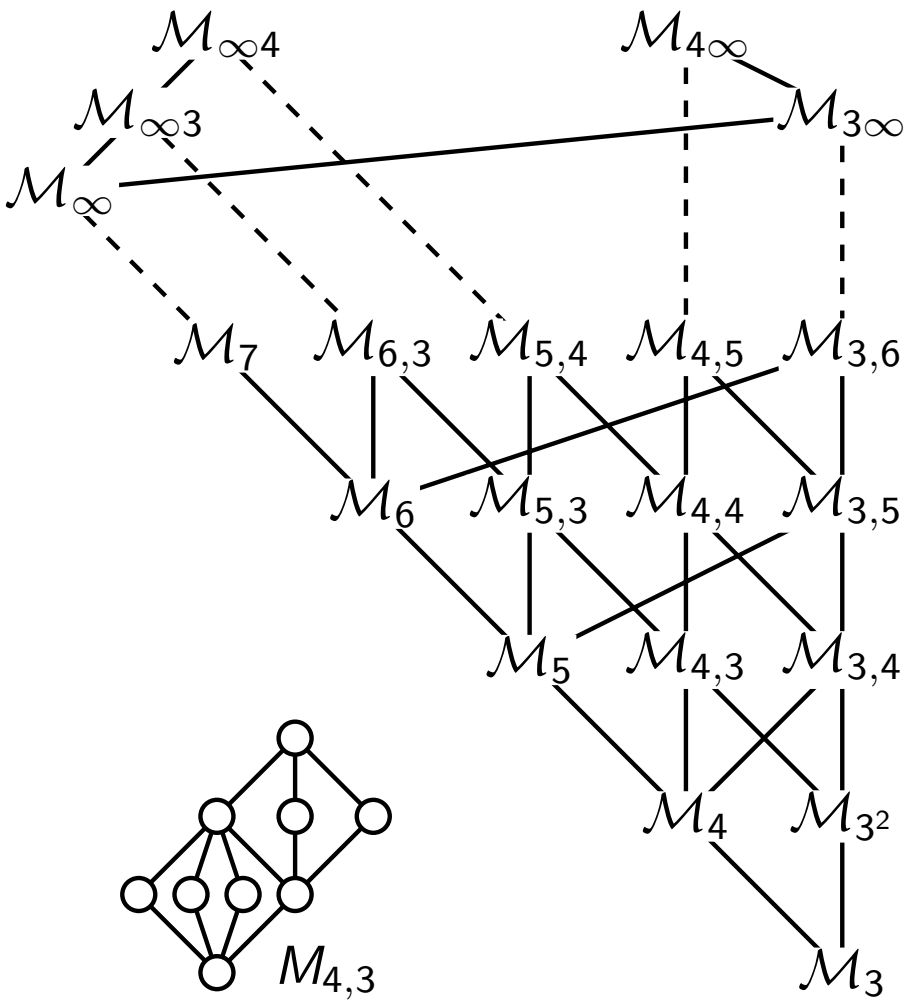

$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

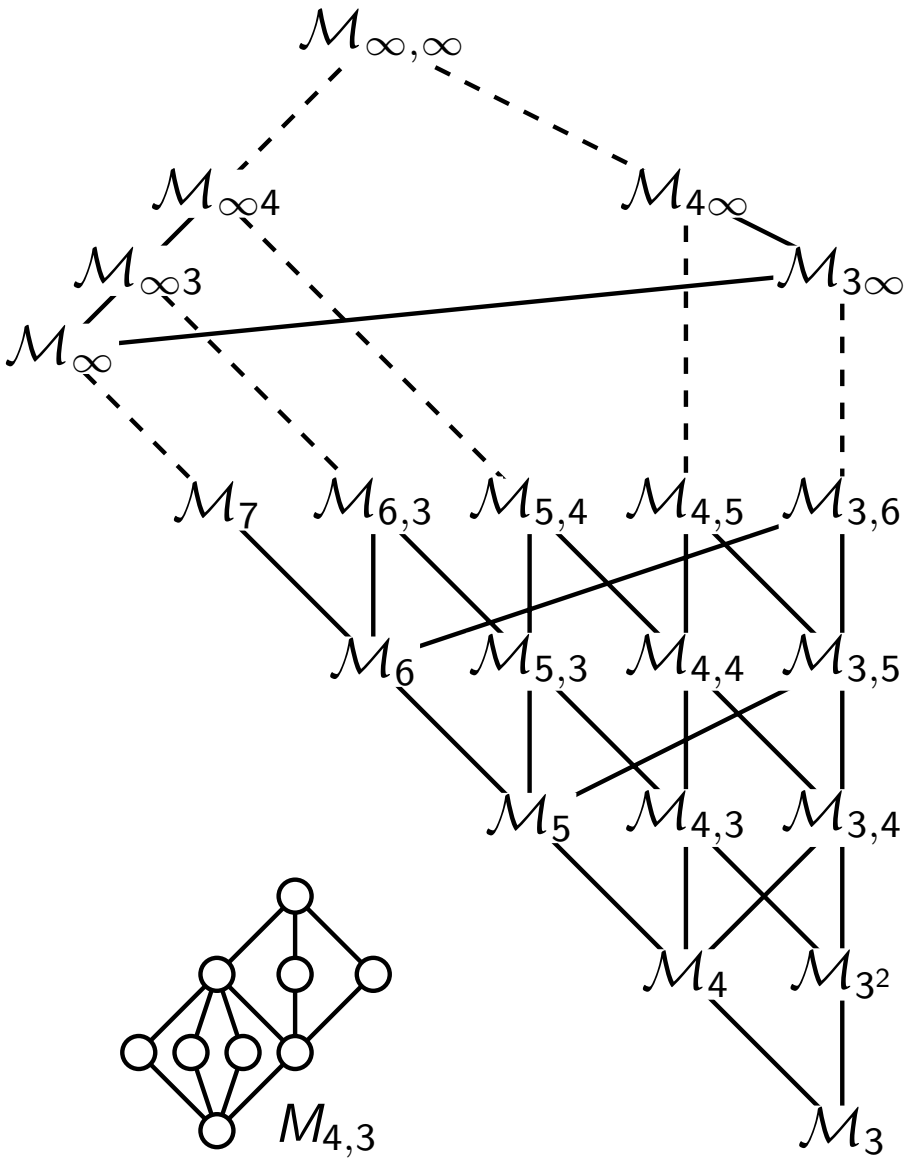

$\mathcal{M}_\infty$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

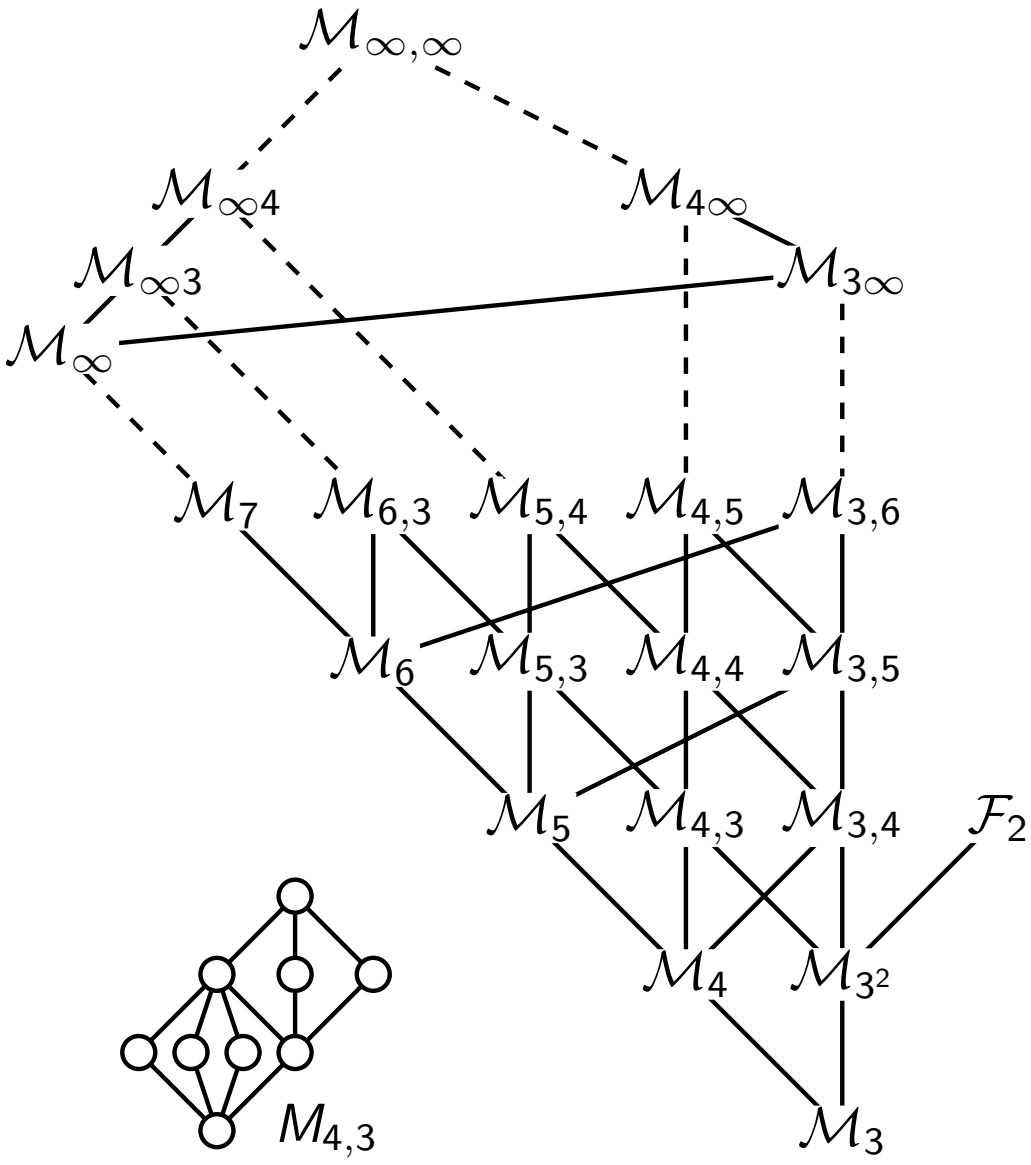

$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

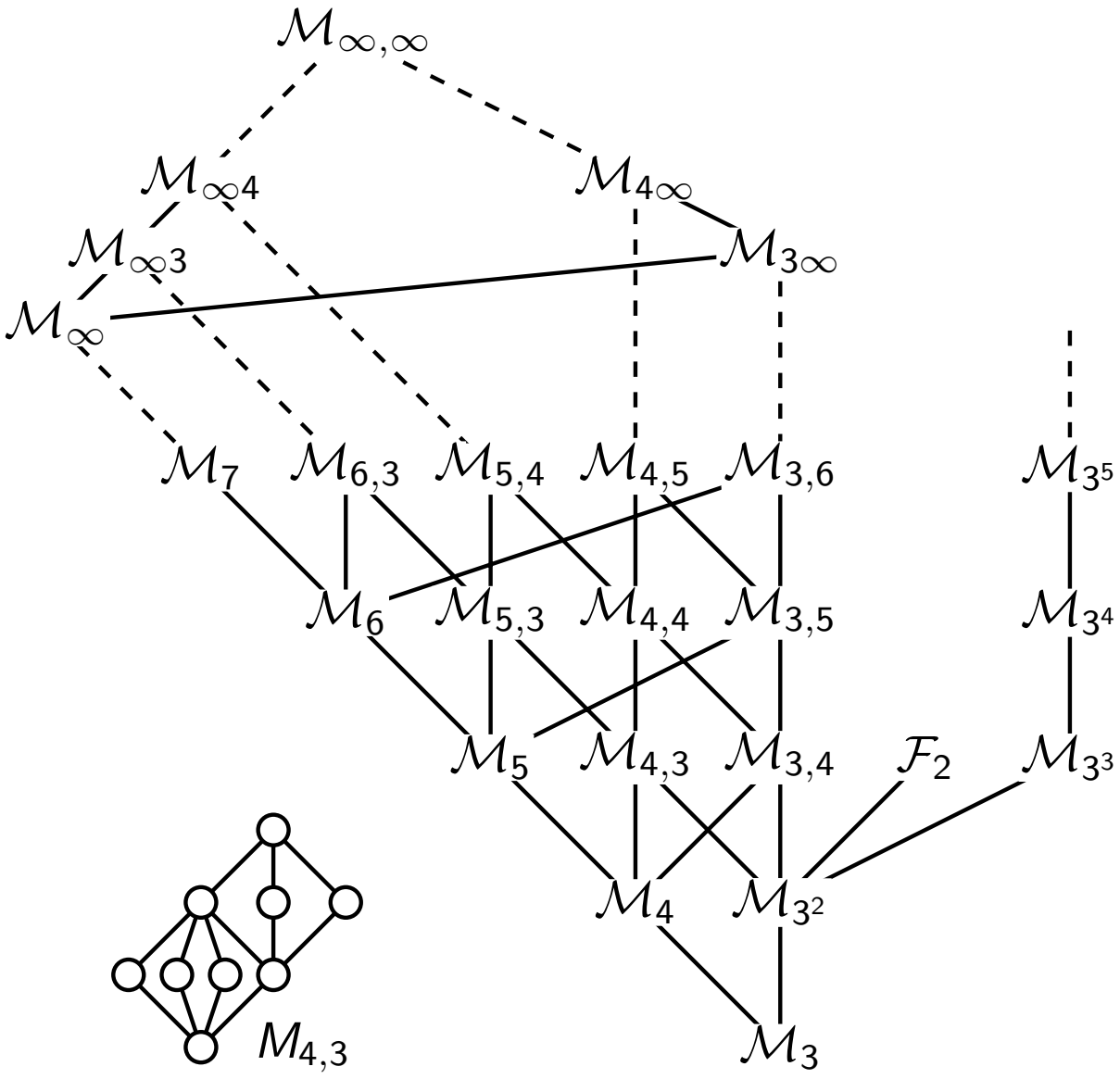

$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

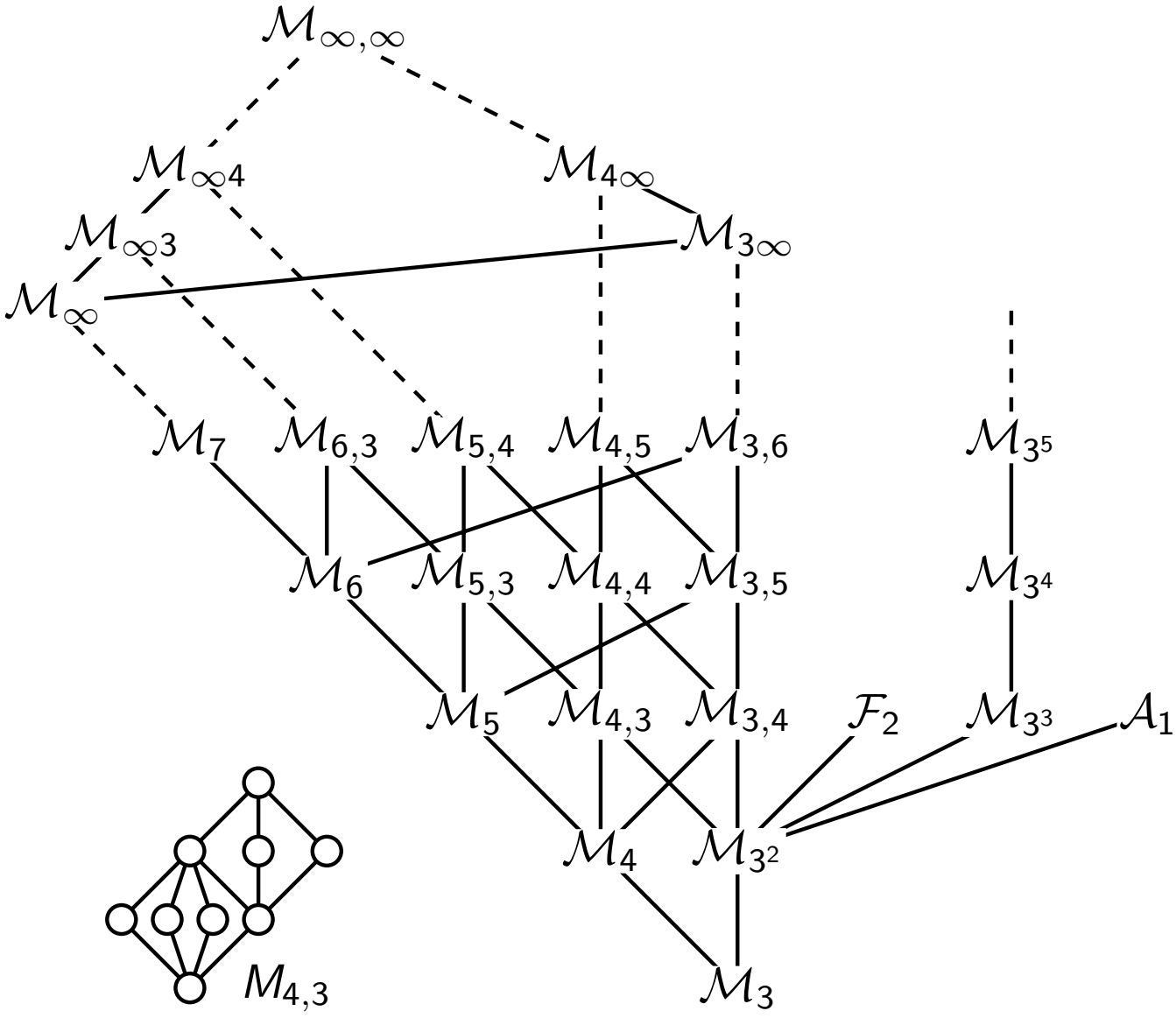

$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

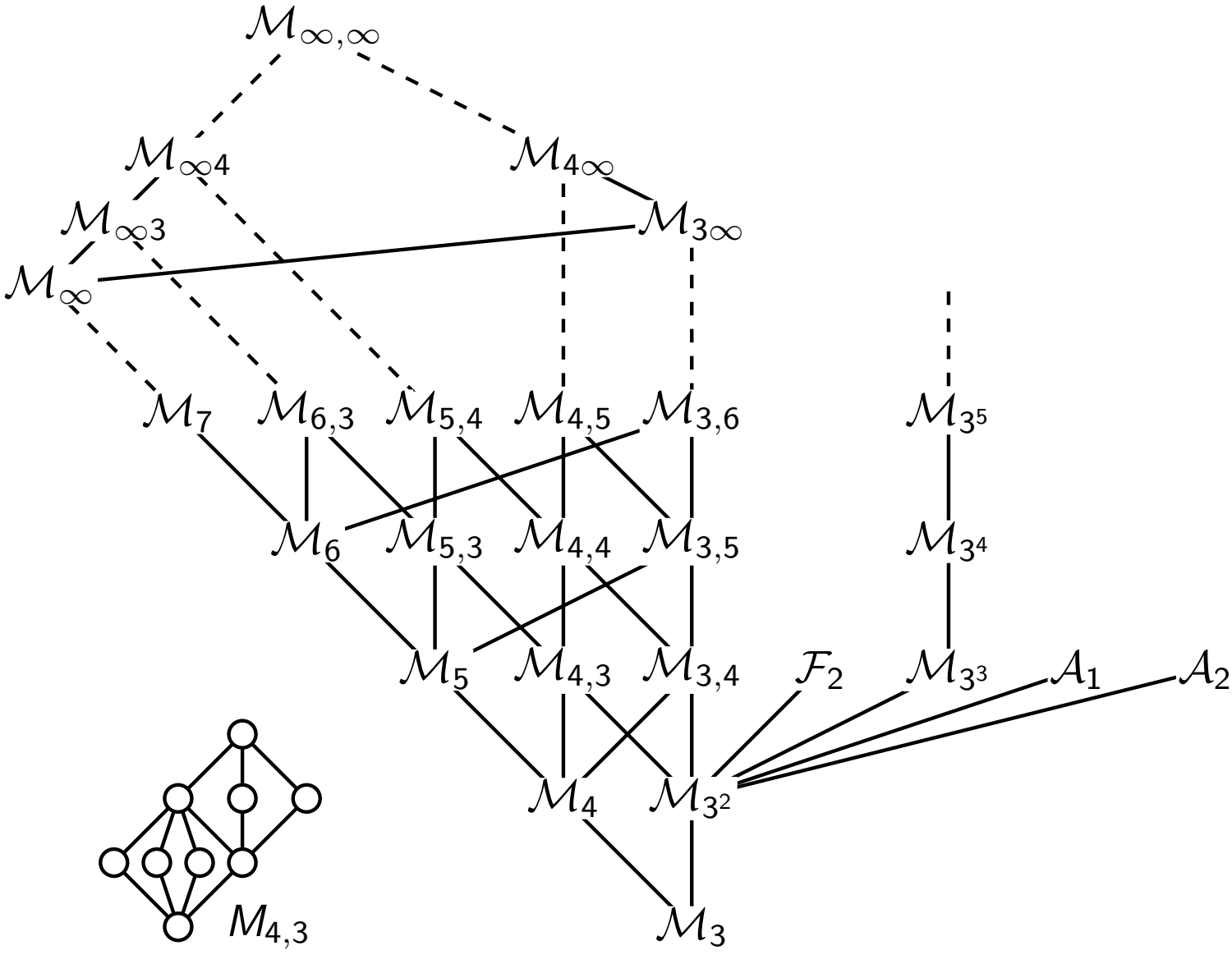

$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

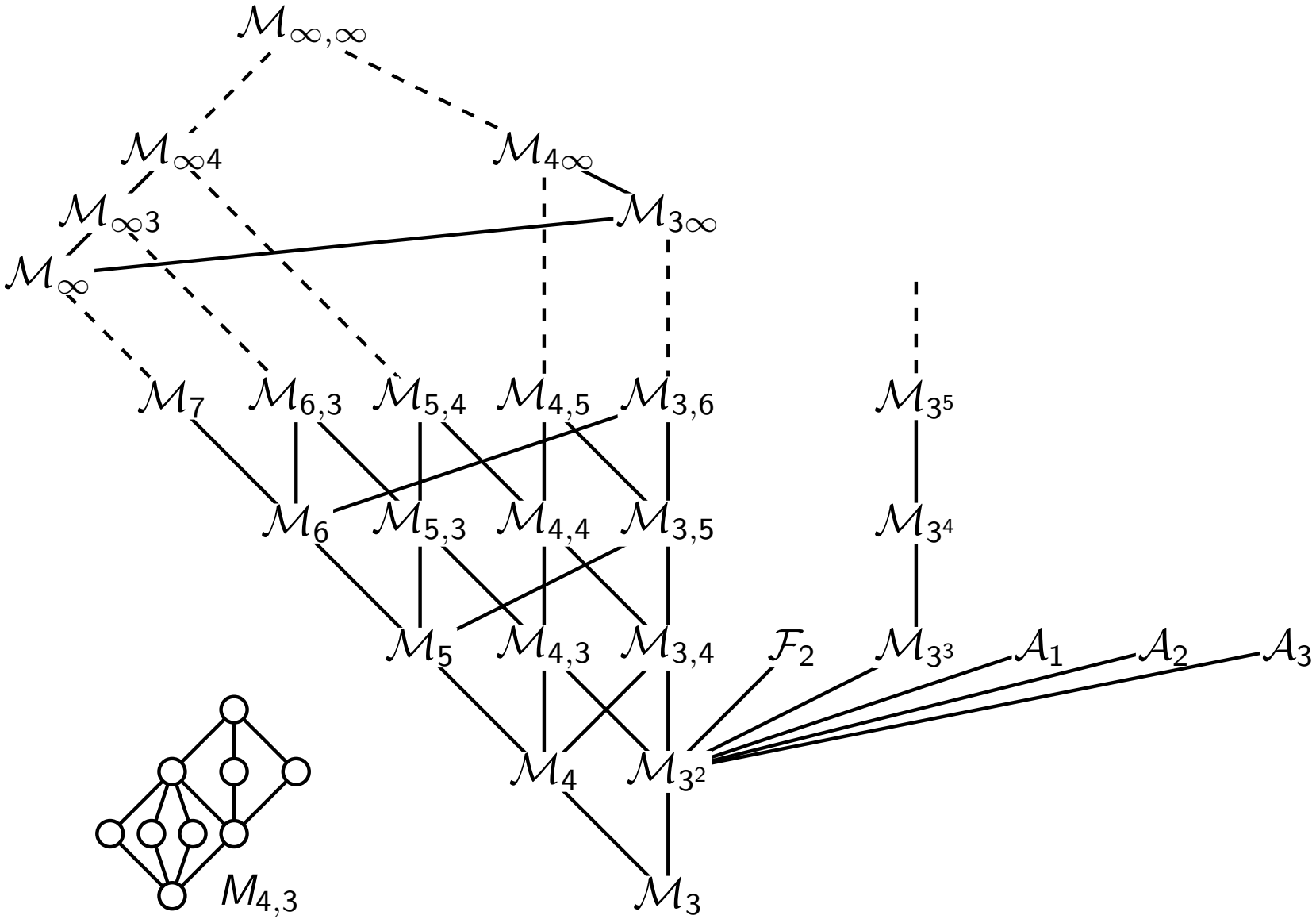

$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

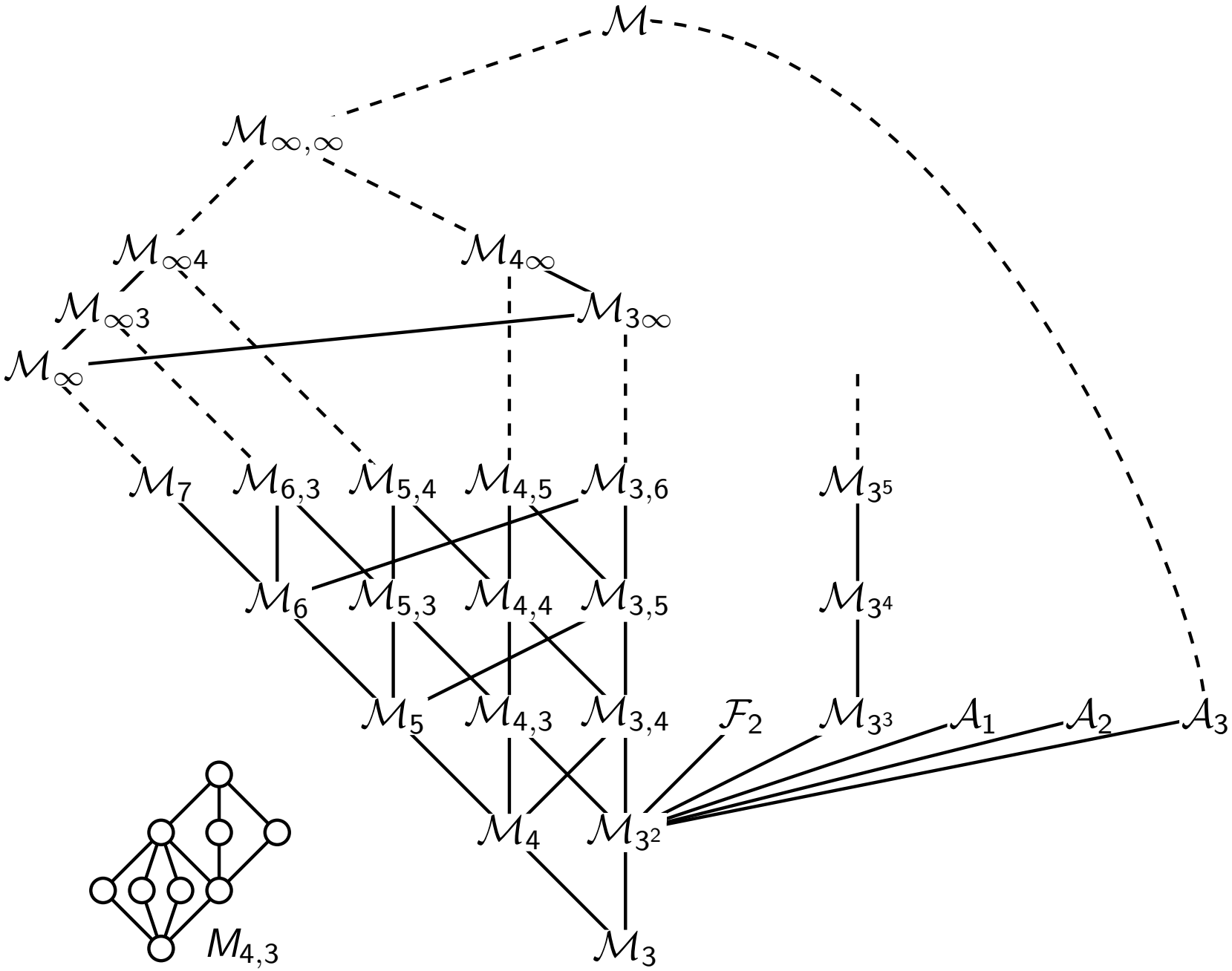

$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{F}_2$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

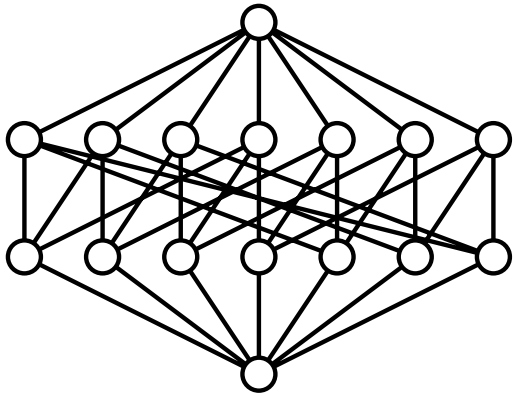

$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_{3^5}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_{3^4}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{F}_2$
$\mathcal{M}_{3^3}$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$


$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_{3^5}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_{3^4}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{F}_2$
$\mathcal{M}_{3^3}$
$\mathcal{A}_1$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

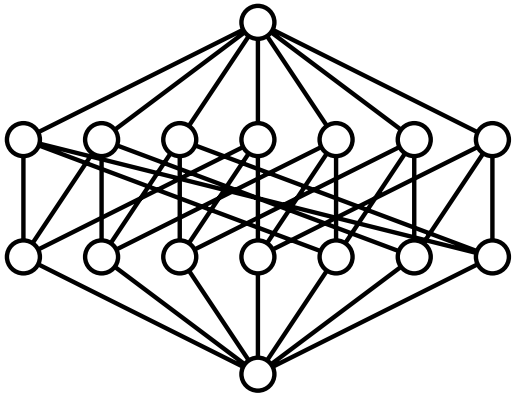

$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_{3^5}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_{3^4}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{F}_2$
$\mathcal{M}_{3^3}$
$\mathcal{A}_1$
$\mathcal{A}_2$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$


$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_{3^5}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_{3^4}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{F}_2$
$\mathcal{M}_{3^3}$
$\mathcal{A}_1$
$\mathcal{A}_2$
$\mathcal{A}_3$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$

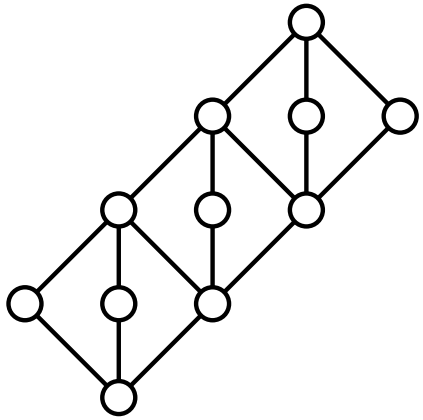

$\mathcal{M}$
$\mathcal{M}_{\infty,\infty}$
$\mathcal{M}_{\infty 4}$
$\mathcal{M}_{4\infty}$
$\mathcal{M}_{\infty 3}$
$\mathcal{M}_{3\infty}$
$\mathcal{M}_{\infty}$
$\mathcal{M}_7$
$\mathcal{M}_{6,3}$
$\mathcal{M}_{5,4}$
$\mathcal{M}_{4,5}$
$\mathcal{M}_{3,6}$
$\mathcal{M}_{3^5}$
$\mathcal{M}_6$
$\mathcal{M}_{5,3}$
$\mathcal{M}_{4,4}$
$\mathcal{M}_{3,5}$
$\mathcal{M}_{3^4}$
$\mathcal{M}_5$
$\mathcal{M}_{4,3}$
$\mathcal{M}_{3,4}$
$\mathcal{F}_2$
$\mathcal{M}_{3^3}$
$\mathcal{A}_1$
$\mathcal{A}_2$
$\mathcal{A}_3$
$\mathcal{M}_4$
$\mathcal{M}_{3^2}$
$\mathcal{M}_3$
$M_{4,3}$



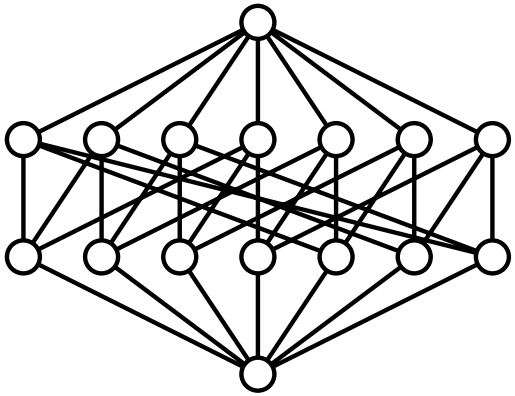
$F_2 = L(\mathbb{F}_2^3, \mathbb{F}_2)$



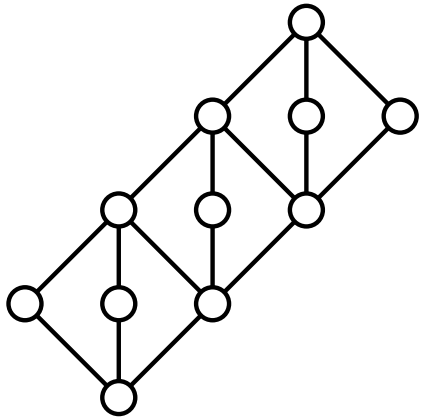
$F_2 = L(\mathbb{F}_2^3, \mathbb{F}_2)$



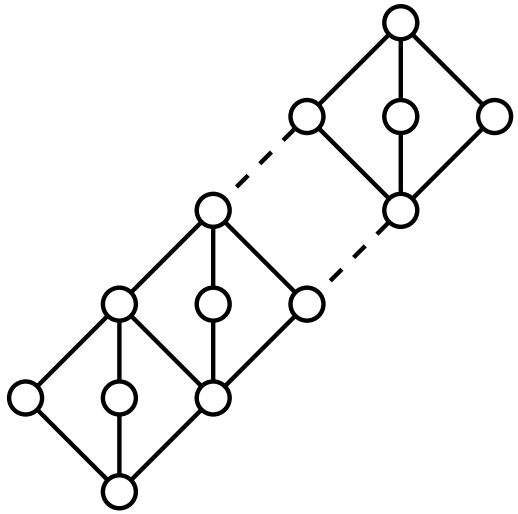
$M_{3^3}$



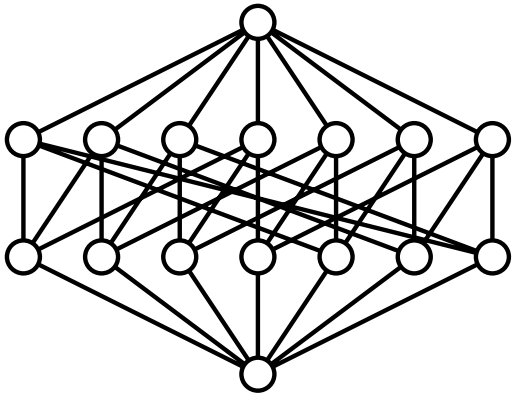
$F_2 = L(\mathbb{F}_2^3, \mathbb{F}_2)$



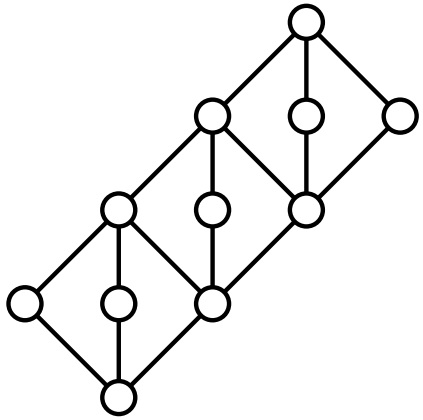
$M_{3^3}$



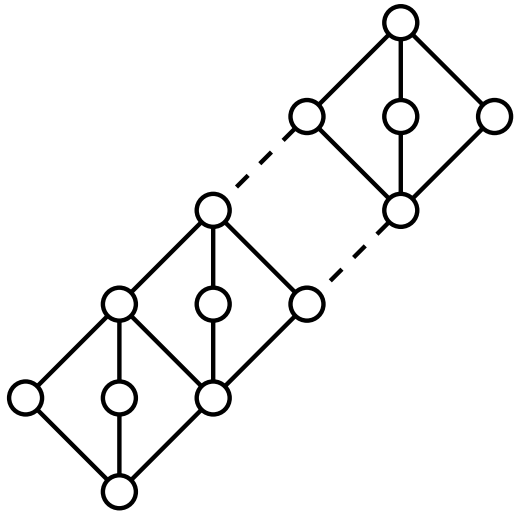
$M_{3^n}$



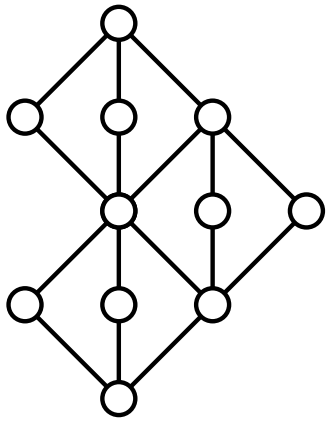
$F_2 = L(\mathbb{F}_2^3, \mathbb{F}_2)$



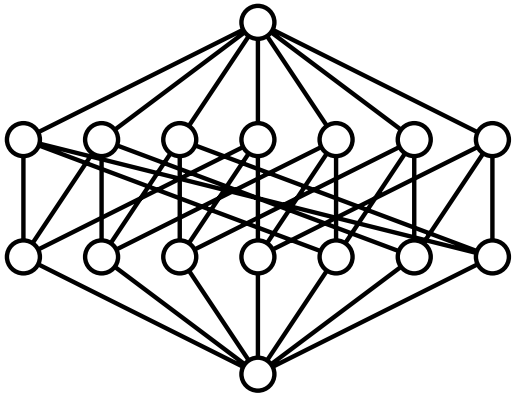
$M_{3^3}$



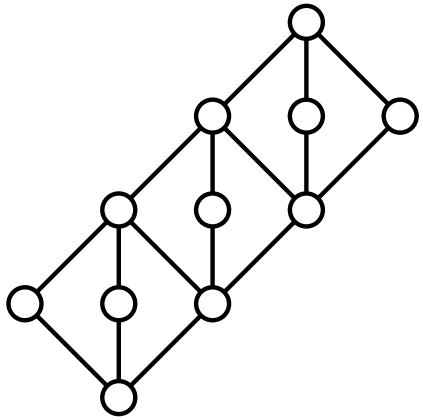
$M_{3^n}$



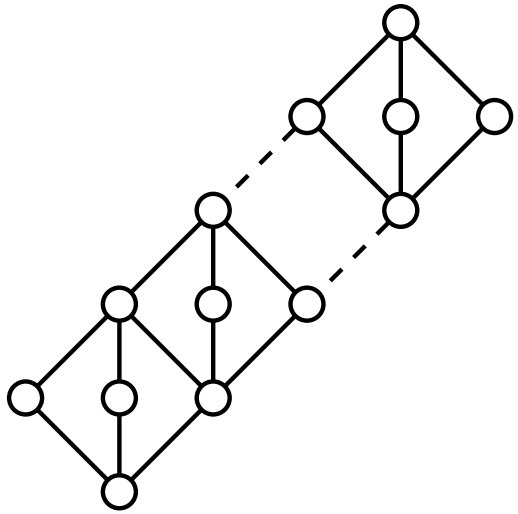
$A_1$



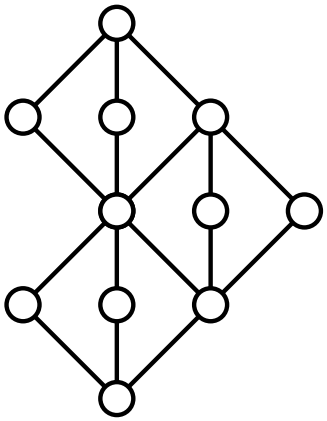
$F_2 = L(\mathbb{F}_2^3, \mathbb{F}_2)$



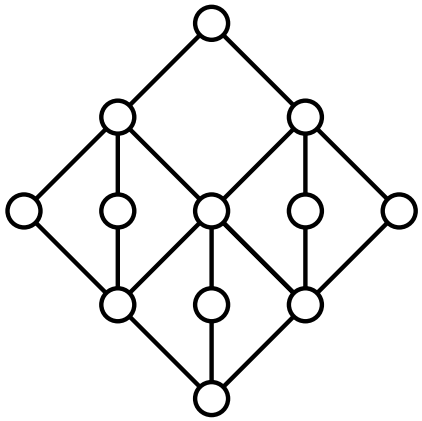
$M_{3^3}$



$M_{3^n}$



$A_1$



$A_2$

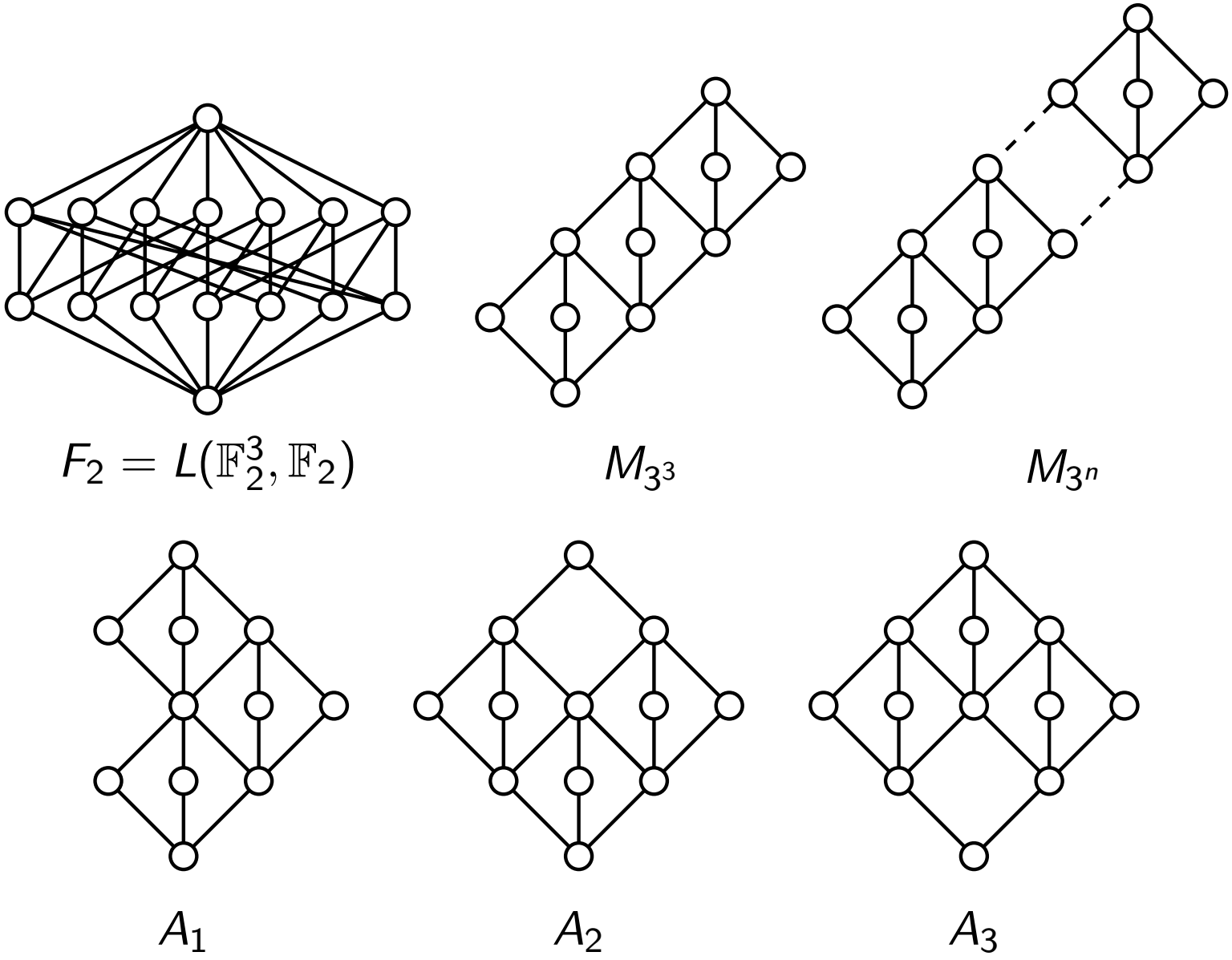



Figure: Modular lattices that generate varieties of finite height

# Varieties above $\mathcal{N}_5$

Ralph McKenzie [1972] found 15 nonmodular varieties $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$ above $\mathcal{N}_5$,

# Varieties above $\mathcal{N}_5$

Ralph McKenzie [1972] found 15 nonmodular varieties $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$ above $\mathcal{N}_5$,

Bjarni Jónsson and Ivan Rival [1979] proved that this list is complete

Ruckelshausen [1978, 1983] found 8 covering both $\mathcal{M}_3$ and $\mathcal{N}_5$

# Varieties above $\mathcal{N}_5$

Ralph McKenzie [1972] found 15 nonmodular varieties $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$ above $\mathcal{N}_5$,

Bjarni Jónsson and Ivan Rival [1979] proved that this list is complete

Ruckelshausen [1978, 1983] found 8 covering both $\mathcal{M}_3$ and $\mathcal{N}_5$

Henry Rose [1984], JG Lee [1985], JB Nation [85,86,90], CY Wong [1989] found all join-irreducible covers of $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$

# Varieties above $\mathcal{N}_5$

Ralph McKenzie [1972] found 15 nonmodular varieties $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$ above $\mathcal{N}_5$,

Bjarni Jónsson and Ivan Rival [1979] proved that this list is complete

Ruckelshausen [1978, 1983] found 8 covering both $\mathcal{M}_3$ and $\mathcal{N}_5$

Henry Rose [1984], JG Lee [1985], JB Nation [85,86,90], CY Wong [1989] found all join-irreducible covers of $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$

But there are many join-reducible covers with join-irreducibles above them

# Varieties above $\mathcal{N}_5$

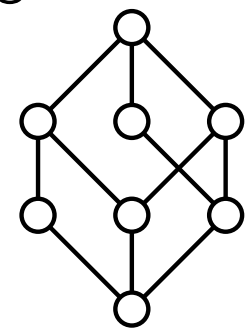
Ralph McKenzie [1972] found 15 nonmodular varieties $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$ above $\mathcal{N}_5$,

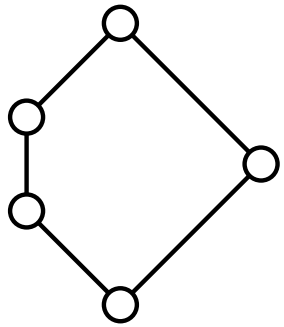
Bjarni Jónsson and Ivan Rival [1979] proved that this list is complete

Ruckelshausen [1978, 1983] found 8 covering both $\mathcal{M}_3$ and $\mathcal{N}_5$

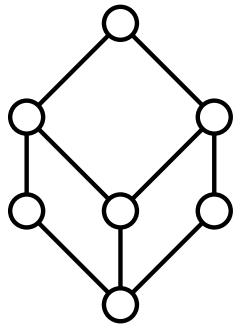
Henry Rose [1984], JG Lee [1985], JB Nation [85,86,90], CY Wong [1989] found all join-irreducible covers of $\mathcal{L}_1, \ldots, \mathcal{L}_{15}$

But there are many join-reducible covers with join-irreducibles above them

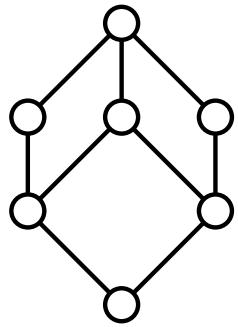
E.g. there is a join-irreducible variety above $L_1, L_2, L_4, L_5$ that is generated by 



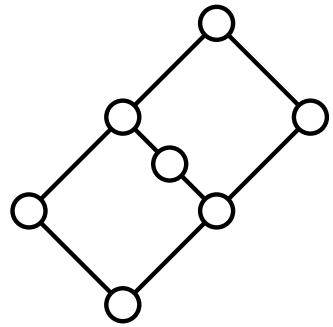
$N_5$



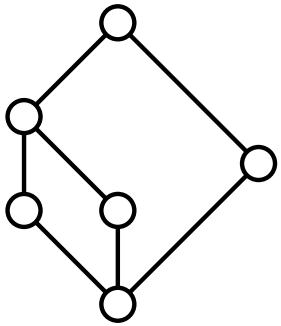
$L_1$



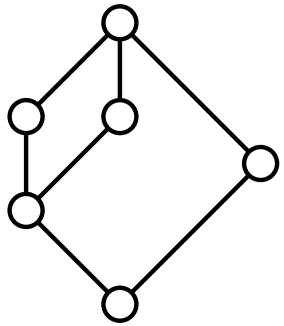
$L_2$



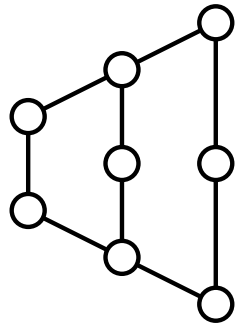
$L_3$



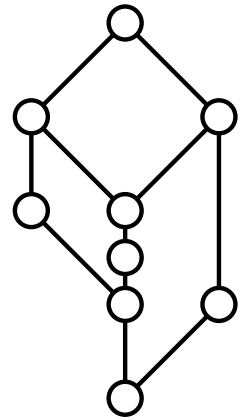
$L_4$



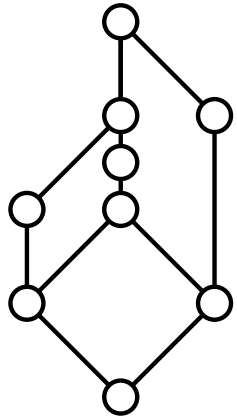
$L_5$



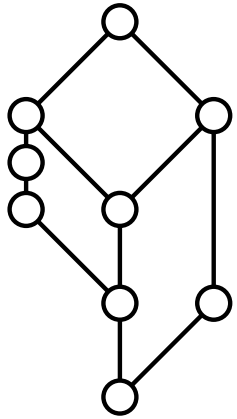
$L_6$



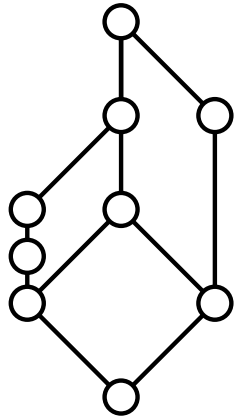
$L_7$



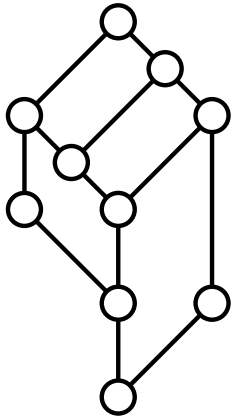
$L_8$



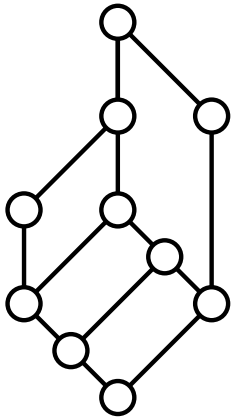
$L_9$



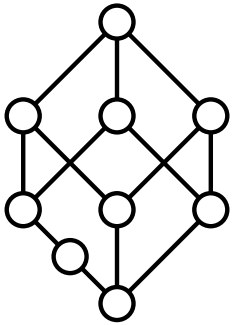
$L_{10}$



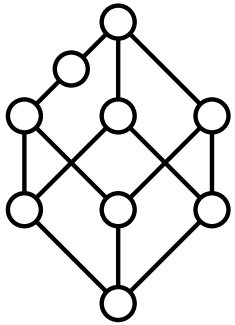
$L_{11}$



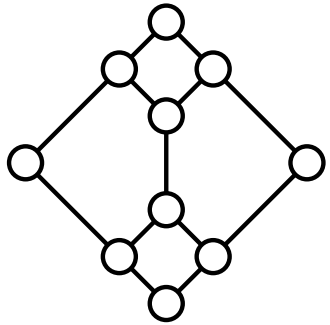
$L_{12}$



$L_{13}$



$L_{14}$



$L_{15}$

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

It combines many research tools with a convenient browser interface

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

It combines many research tools with a convenient browser interface

E.g. includes GAP (for Groups, Algorithms and Programs)

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

It combines many research tools with a convenient browser interface

E.g. includes GAP (for Groups, Algorithms and Programs)

Maxima for symbolic algebra

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

It combines many research tools with a convenient browser interface

E.g. includes GAP (for Groups, Algorithms and Programs)

Maxima for symbolic algebra

Networkx for graph theory

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

It combines many research tools with a convenient browser interface

E.g. includes GAP (for Groups, Algorithms and Programs)

Maxima for symbolic algebra

Networkx for graph theory

R for statistics, etc ...

# Computational approach

Sage (sagemath.org) is a free open-source computer algebra system

It combines many research tools with a convenient browser interface

E.g. includes GAP (for Groups, Algorithms and Programs)

Maxima for symbolic algebra

Networkx for graph theory

R for statistics, etc ...

Python is used to interact with and program in Sage

But Sage does not have direct support for Universal Algebra

But Sage does not have direct support for Universal Algebra

The UA Calculator (uacalc.org) by E. Kiss, R. Freese and M. Valeriote contains highly optimized algorithms for computations in a finite algebra

But Sage does not have direct support for Universal Algebra

The UA Calculator (uacalc.org) by E. Kiss, R. Freese and M. Valeriote contains highly optimized algorithms for computations in a finite algebra

Has a Java graphical interface for displaying (congruence) lattices

But algebras have to be entered one at a time

But algebras have to be entered one at a time

Computed results are not easy to export

But algebras have to be entered one at a time

Computed results are not easy to export

Difficult to program additional algorithms or extend interface

But algebras have to be entered one at a time

Computed results are not easy to export

Difficult to program additional algorithms or extend interface

Instead wrote some small Java routines that allow Sage to access the UA Calculator

But algebras have to be entered one at a time

Computed results are not easy to export

Difficult to program additional algorithms or extend interface

Instead wrote some small Java routines that allow Sage to access the UA Calculator

Sage can handle lists of algebras, e.g. compute Con($A$) for each of them

Heitzig and Reinhold [2002] describe a method for enumerating finite lattices

Heitzig and Reinhold [2002] describe a method for enumerating finite lattices

I implemented it in Python and used it to compute all lattices with up to 12 elements

Heitzig and Reinhold [2002] describe a method for enumerating finite lattices

I implemented it in Python and used it to compute all lattices with up to 12 elements

Filtered the list to get only subdirectly irreducible lattices (building blocks)

Heitzig and Reinhold [2002] describe a method for enumerating finite lattices

I implemented it in Python and used it to compute all lattices with up to 12 elements

Filtered the list to get only subdirectly irreducible lattices (building blocks)

Task: compute (the bottom of) the poset of join-irreducible lattice varieties $V(A)$

Heitzig and Reinhold [2002] describe a method for enumerating finite lattices

I implemented it in Python and used it to compute all lattices with up to 12 elements

Filtered the list to get only subdirectly irreducible lattices (building blocks)

Task: compute (the bottom of) the poset of join-irreducible lattice varieties $V(A)$

The bottom lattice of lattice varieties is obtained from the downsets of this poset

For finite s.i. lattices $A, B$ we have $V(A) \subseteq V(B)$ iff $A \in HS(B)$

For finite s.i. lattices $A, B$ we have $V(A) \subseteq V(B)$ iff $A \in HS(B)$

So how does one compute $HS(B)$?

For finite s.i. lattices $A, B$ we have $V(A) \subseteq V(B)$ iff $A \in HS(B)$

So how does one compute $HS(B)$?

Use the UA Calculator to find $\mathrm{Sub}(B) =$ all subalgebras of $B$

For finite s.i. lattices $A, B$ we have $V(A) \subseteq V(B)$ iff $A \in HS(B)$

So how does one compute $HS(B)$?

Use the UA Calculator to find $\mathrm{Sub}(B)$ = all subalgebras of $B$

Eliminate isomorphic copies to get $S(B)$

For finite s.i. lattices $A, B$ we have $V(A) \subseteq V(B)$ iff $A \in HS(B)$

So how does one compute $HS(B)$?

Use the UA Calculator to find $\mathrm{Sub}(B)$ = all subalgebras of $B$

Eliminate isomorphic copies to get $S(B)$

For each $C \in S(B)$ test if $A \in H(C)$, but how?

For finite s.i. lattices $A, B$ we have $V(A) \subseteq V(B)$ iff $A \in HS(B)$

So how does one compute $HS(B)$?

Use the UA Calculator to find $\mathrm{Sub}(B)$ = all subalgebras of $B$

Eliminate isomorphic copies to get $S(B)$

For each $C \in S(B)$ test if $A \in H(C)$, but how?

Checking all maps from $C$ to $A$ is not feasible

Instead use a Constraint Satisfaction solver, e.g. Minion (minion.sourceforge.net)

Instead use a Constraint Satisfaction solver, e.g. Minion (minion.sourceforge.net)

A *constraint satisfaction problem* (CSP) is traditionally given as a triple $(X, D, \mathcal{C})$ where

Instead use a Constraint Satisfaction solver, e.g. Minion (minion.sourceforge.net)

A *constraint satisfaction problem* (CSP) is traditionally given as a triple $(X, D, \mathcal{C})$ where

- $X = \{x_0, \ldots, x_{n-1}\}$ is a set of variables

Instead use a Constraint Satisfaction solver, e.g. Minion (minion.sourceforge.net)

A *constraint satisfaction problem* (CSP) is traditionally given as a triple $(X, D, \mathcal{C})$ where

- $X = \{x_0, \ldots, x_{n-1}\}$ is a set of variables
- $D$ is a finite set, the *domain* for the variables

Instead use a Constraint Satisfaction solver, e.g. Minion (minion.sourceforge.net)

A *constraint satisfaction problem* (CSP) is traditionally given as a triple $(X, D, \mathcal{C})$ where

- $X = \{x_0, \ldots, x_{n-1}\}$ is a set of variables
- $D$ is a finite set, the *domain* for the variables
- $\mathcal{C} = \{(v_0, R_0), \ldots, (v_{n-1}, R_{n-1})\}$ is a set of *constraints*, i.e. $v_i \in X^{k_i}$ and $R_i \subseteq D^{k_i}$ for some $k_i > 0$

Instead use a Constraint Satisfaction solver, e.g. Minion (minion.sourceforge.net)

A *constraint satisfaction problem* (CSP) is traditionally given as a triple $(X, D, \mathcal{C})$ where

- $X = \{x_0, \ldots, x_{n-1}\}$ is a set of variables
- $D$ is a finite set, the *domain* for the variables
- $\mathcal{C} = \{(v_0, R_0), \ldots, (v_{n-1}, R_{n-1})\}$ is a set of *constraints*, i.e. $v_i \in X^{k_i}$ and $R_i \subseteq D^{k_i}$ for some $k_i > 0$

A *solution* of the CSP $(X, D, \mathcal{C})$ is a function $h : X \to D$ such that $\bar{h}(v_i) \in R_i$, where $\bar{h}(x_0, \ldots, x_{k-1}) = (h(x_0), \ldots, h(x_{k-1}))$

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

**Lemma**

*The CSP and the generalized CSP are equivalent*

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

**Lemma**

*The CSP and the generalized CSP are equivalent*

**Proof.**

Given $(X, D, \mathcal{C})$, let $\mathbf{U} = (X, \{v_0\}, \ldots, \{v_{m-1}\})$, $\mathbf{V} = (D, R_0, \ldots, R_{m-1})$

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

**Lemma**

*The CSP and the generalized CSP are equivalent*

**Proof.**

Given $(X, D, \mathcal{C})$, let $\mathbf{U} = (X, \{v_0\}, \ldots, \{v_{m-1}\})$, $\mathbf{V} = (D, R_0, \ldots, R_{m-1})$

Then a solution of $(X, D, \mathcal{C})$ is a homomorphism $\mathbf{U} \to \mathbf{V}$

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

**Lemma**

*The CSP and the generalized CSP are equivalent*

**Proof.**

Given $(X, D, \mathcal{C})$, let $\mathbf{U} = (X, \{v_0\}, \ldots, \{v_{m-1}\})$, $\mathbf{V} = (D, R_0, \ldots, R_{m-1})$

Then a solution of $(X, D, \mathcal{C})$ is a homomorphism $\mathbf{U} \to \mathbf{V}$

Conversely, given a pair $(\mathbf{U}, \mathbf{V})$, both of type $R_0, \ldots, R_{t-1}$

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

**Lemma**

*The CSP and the generalized CSP are equivalent*

**Proof.**

Given $(X, D, \mathcal{C})$, let $\mathbf{U} = (X, \{v_0\}, \ldots, \{v_{m-1}\})$, $\mathbf{V} = (D, R_0, \ldots, R_{m-1})$

Then a solution of $(X, D, \mathcal{C})$ is a homomorphism $\mathbf{U} \to \mathbf{V}$

Conversely, given a pair $(\mathbf{U}, \mathbf{V})$, both of type $R_0, \ldots, R_{t-1}$

let $X = U$, $D = V$ and $\mathcal{C} = \{(v, R_i^{\mathbf{V}}) | v \in R_i^{\mathbf{U}}, i = 0, \ldots, t-1\}$

The *generalized CSP* is a pair $(\mathbf{U}, \mathbf{V})$ of finite relational structures of the same type with finitely many relation symbols

A solution of the generalized CSP is a homomorphism $h : \mathbf{U} \to \mathbf{V}$

**Lemma**

*The CSP and the generalized CSP are equivalent*

**Proof.**

Given $(X, D, \mathcal{C})$, let $\mathbf{U} = (X, \{v_0\}, \ldots, \{v_{m-1}\})$, $\mathbf{V} = (D, R_0, \ldots, R_{m-1})$

Then a solution of $(X, D, \mathcal{C})$ is a homomorphism $\mathbf{U} \to \mathbf{V}$

Conversely, given a pair $(\mathbf{U}, \mathbf{V})$, both of type $R_0, \ldots, R_{t-1}$

let $X = U$, $D = V$ and $\mathcal{C} = \{(v, R_i^{\mathbf{V}}) | v \in R_i^{\mathbf{U}}, i = 0, \ldots, t-1\}$

Then a homomorphism $\mathbf{U} \to \mathbf{V}$ is a solution of $(X, D, \mathcal{C})$ □

CSP solvers have been developed for the past 30 years

CSP solvers have been developed for the past 30 years

Minion is a recent open-source CSP solver

CSP solvers have been developed for the past 30 years

Minion is a recent open-source CSP solver

Input syntax is similar to the traditional $(X, D, \mathcal{C})$ version

CSP solvers have been developed for the past 30 years

Minion is a recent open-source CSP solver

Input syntax is similar to the traditional $(X, D, \mathcal{C})$ version

Wrote short Sage/Python routines that convert a pair of finite algebras into a Minion file, call Minion and read the output file back into Sage

CSP solvers have been developed for the past 30 years

Minion is a recent open-source CSP solver

Input syntax is similar to the traditional $(X, D, \mathcal{C})$ version

Wrote short Sage/Python routines that convert a pair of finite algebras into a Minion file, call Minion and read the output file back into Sage

Implemented commands: Hom($\mathbf{A}, \mathbf{B}$), End($\mathbf{A}$),

CSP solvers have been developed for the past 30 years

Minion is a recent open-source CSP solver

Input syntax is similar to the traditional $(X, D, \mathcal{C})$ version

Wrote short Sage/Python routines that convert a pair of finite algebras into a Minion file, call Minion and read the output file back into Sage

Implemented commands: $\mathrm{Hom}(\mathbf{A}, \mathbf{B})$, $\mathrm{End}(\mathbf{A})$,

$\mathrm{Emb}(\mathbf{A}, \mathbf{B})$, $\mathrm{Aut}(\mathbf{A})$, $\mathrm{Pol}_1(\mathbf{A})$

CSP solvers have been developed for the past 30 years

Minion is a recent open-source CSP solver

Input syntax is similar to the traditional $(X, D, \mathcal{C})$ version

Wrote short Sage/Python routines that convert a pair of finite algebras into a Minion file, call Minion and read the output file back into Sage

Implemented commands: $\mathrm{Hom}(\mathbf{A}, \mathbf{B})$, $\mathrm{End}(\mathbf{A})$,

$\mathrm{Emb}(\mathbf{A}, \mathbf{B})$, $\mathrm{Aut}(\mathbf{A})$, $\mathrm{Pol}_1(\mathbf{A})$

is_hom_image$(\mathbf{A}, \mathbf{B})$, is_subalgebra$(\mathbf{A}, \mathbf{B})$, is_isomorphic$(\mathbf{A}, \mathbf{B})$

The number of subdirectly irreducible lattices of size $n$:

The number of subdirectly irreducible lattices of size $n$:

| $n$ | All | SI |
|---|---|---|
| 1 | 1 | 0 |
| 2 | 1 | 1 |
| 3 | 1 | 0 |
| 4 | 2 | 0 |
| 5 | 5 | 2 |
| 6 | 15 | 4 |
| 7 | 53 | 16 |
| 8 | 222 | 69 |
| 9 | 1078 | 360 |
| 10 | 5994 | 2103 |
| 11 | 37622 | 13867 |
| 12 | 262776 | 100853 |

The number of subdirectly irreducible lattices of size $n$:

| $n$ | All | SI |
|---|---|---|
| 1 | 1 | 0 |
| 2 | 1 | 1 |
| 3 | 1 | 0 |
| 4 | 2 | 0 |
| 5 | 5 | 2 |
| 6 | 15 | 4 |
| 7 | 53 | 16 |
| 8 | 222 | 69 |
| 9 | 1078 | 360 |
| 10 | 5994 | 2103 |
| 11 | 37622 | 13867 |
| 12 | 262776 | 100853 |

Heitzig and Reinhold [2002] enumerated all lattices up to $n = 18$

There are 2555 subdirectly irreducible lattices of size $\leq 10$

There are 2555 subdirectly irreducible lattices of size $\leq 10$

Each lattice generates a join-irreducible variety

There are 2555 subdirectly irreducible lattices of size $\leq 10$

Each lattice generates a join-irreducible variety

Computing the poset of subvarieties took about 2 weeks

There are 2555 subdirectly irreducible lattices of size $\leq 10$

Each lattice generates a join-irreducible variety

Computing the poset of subvarieties took about 2 weeks

The poset has height 7 (so very flat and wide)

There are 2555 subdirectly irreducible lattices of size $\leq 10$

Each lattice generates a join-irreducible variety

Computing the poset of subvarieties took about 2 weeks

The poset has height 7 (so very flat and wide)

Levels 0, 1, 2 were discussed before (known) and confirmed by the calculations

# Future Work

Find a good way to view the data from these calculations

Apply some data mining techniques

Develop calculations that can prove the list of varieties above a given variety is complete

Find sets of equations that determine these varieties

# References

R. Freese, E. Kiss, M. Valeriote, Universal Algebra Calculator, http://uacalc.org

P. Jipsen and H. Rose, *Varieties of Lattices*, Lecture Notes in Mathematics, Springer Verlag, 1992, pp x+162

Minion Constraint Satisfier, http://minion.sourceforge.net

W. Stein, Sage Computer Algebra System, http://sagemath.org

# References

R. Freese, E. Kiss, M. Valeriote, Universal Algebra Calculator, http://uacalc.org

P. Jipsen and H. Rose, *Varieties of Lattices*, Lecture Notes in Mathematics, Springer Verlag, 1992, pp x+162

Minion Constraint Satisfier, http://minion.sourceforge.net

W. Stein, Sage Computer Algebra System, http://sagemath.org

Thank You