Axiomatizing real representability

yes. . . meaning no

Mike Newman

Joint work with Dillon Mayhew and Geoff Whittle

Workshop on structure in graphs and matroids
Princeton, July 2014

motivation

Whitney (1935):

> *Let us call a system obeying (a) and (b) a "matroid". The present paper is devoted to a study of the elementary properties of matroids. The fundamental question of completely characterizing systems which represent matrices is left unsolved.*

motivation

Whitney (1935):

> *Let us call a system obeying (a) and (b) a "matroid". The present paper is devoted to a study of the elementary properties of matroids. The fundamental question of completely characterizing systems which represent matrices is left unsolved.*

Vámos (1972): The missing axiom of matroid theory is lost forever.

motivation

Whitney (1935):

> *Let us call a system obeying (a) and (b) a "matroid". The present paper is devoted to a study of the elementary properties of matroids. The fundamental question of completely characterizing systems which represent matrices is left unsolved.*

Vámos (1972): The missing axiom of matroid theory is lost forever.

MNW (2014): Is the missing axiom of matroid theory lost forever?

motivation

Whitney (1935):

> *Let us call a system obeying (a) and (b) a "matroid". The present paper is devoted to a study of the elementary properties of matroids. The fundamental question of completely characterizing systems which represent matrices is left unsolved.*

Vámos (1972): The missing axiom of matroid theory is lost forever.

MNW (2014): Is the missing axiom of matroid theory lost forever?

MNW (2014+): yes

characterizing

What does “characterize” mean?

- matroids: axioms

characterizing

What does “characterize” mean?

- matroids: axioms
- minor-closed classes: excluded minors

characterizing

What does "characterize" mean?

- matroids: axioms
- minor-closed classes: excluded minors
- some easily-stated "property" . . . ??

characterizing real-representability

Theorems about characterizing real-representable matroids:

- There are infinitely many excluded minors. (Geelen and Whittle ; Lazarson?)

characterizing real-representability

Theorems about characterizing real-representable matroids:

- There are infinitely many excluded minors. (Geelen and Whittle ; Lazarson?)
- Every real-representable matroid is contained in an excluded minor. (MNW 2009)

characterizing real-representability

Theorems about characterizing real-representable matroids:

- There are infinitely many excluded minors. (Geelen and Whittle ; Lazarson?)
- Every real-representable matroid is contained in an excluded minor. (MNW 2009)
- There is no finite sentence in first order logic that characterizes V-matroids. (Vámos 1972)

characterizing real-representability

Theorems about characterizing real-representable matroids:

- There are infinitely many excluded minors. (Geelen and Whittle ; Lazarson?)
- Every real-representable matroid is contained in an excluded minor. (MNW 2009)
- There is no finite sentence in first order logic that characterizes V-matroids. (Vámos 1972)
- Any axiomatization of real-representability cannot be of the form where it has at most one alternation of quantifier with only element variables after the quantifier. (MNW 2014)

describing matroids

A matroid is a finite ground set E and a collection $\mathcal{I}$ of subsets of E such that:

- $\emptyset \in \mathcal{I}$
- $\forall X, Y \subseteq E \left\{ X \in \mathcal{I}, Y \subseteq X \implies Y \in \mathcal{I} \right\}$
- $\forall X, Y \in \mathcal{I} \left\{ |X| < |Y| \implies \exists y \in Y \setminus X \left\{ X \cup y \in \mathcal{I} \right\} \right\}$

$\mathbb{F}_2$

How to capture binary representability?

$\mathbb{F}_2$

How to capture binary representability?

- There exists a matrix over $\mathbb{F}_2$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{F}_2$.

$\mathbb{F}_2$

How to capture binary representability?

- There exists a matrix over $\mathbb{F}_2$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{F}_2$.
- There exists no $U_{2,4}$ minor.

$\mathbb{F}_2$

How to capture binary representability?

- There exists a matrix over $\mathbb{F}_2$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{F}_2$.
- There exists no $U_{2,4}$ minor.
- It is not the case that there exists an independent set Y and four points a, b, c, d not in Y such that $Y \cup T$ is independent if $|T| \leq 2$ and dependent if $|T| \geq 3$ for $T \subseteq \{a, b, c, d\}$.

$\mathbb{F}_2$

How to capture binary representability?

- There exists a matrix over $\mathbb{F}_2$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{F}_2$.
- There exists no $U_{2,4}$ minor.
- It is not the case that there exists an independent set Y and four points a, b, c, d not in Y such that $Y \cup T$ is independent if $|T| \leq 2$ and dependent if $|T| \geq 3$ for $T \subseteq \{a, b, c, d\}$.

$\mathbb{F}_2$

How to capture binary representability?

- There exists a matrix over $\mathbb{F}_2$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{F}_2$.
- There exists no $U_{2,4}$ minor.
- It is not the case that there exists an independent set Y and four points a, b, c, d not in Y such that $Y \cup T$ is independent if $|T| \leq 2$ and dependent if $|T| \geq 3$ for $T \subseteq \{a, b, c, d\}$.

Those who Believe know that similar statements apply to $\mathbb{F}_q$.

$\mathbb{R}$

How to capture real representability?

$\mathbb{R}$

How to capture real representability?

- There exists a matrix over $\mathbb{R}$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{R}$.

$\mathbb{R}$

How to capture real representability?

- There exists a matrix over $\mathbb{R}$ whose columns are indexed by E, such that a set of elements is independent in the matroid if and only if the corresponding set of columns is linearly independent over $\mathbb{R}$.
- ?

language

We define a language with the following ingredients

- variables X, Y, Z, ..., all representing subsets of E
- a unary predicate ind with $\text{ind}(X)$ true if X is independent
- a unary predicate sing with $\text{sing}(X)$ true if $|X| = 1$
- a binary relation $\subseteq$ with $X \subseteq Y$ true if X is a subset of Y
- $\wedge$, $\vee$, $\neg$, $\exists$, $\forall$, $\rightarrow$

language

We define a language with the following ingredients

- variables X, Y, Z, ..., all representing subsets of E
- a unary predicate ind with $\mathrm{ind}(X)$ true if X is independent
- a unary predicate sing with $\mathrm{sing}(X)$ true if $|X| = 1$
- a binary relation $\subseteq$ with $X \subseteq Y$ true if X is a subset of Y
- $\wedge$, $\vee$, $\neg$, $\exists$, $\forall$, $\rightarrow$

We combine atomic formulae using connectives, to obtain a statement about some variables. We may combine these using further connectives. Then we quantify all variables to obtain an "axiom".

language

We define a language with the following ingredients

- variables X, Y, Z, ..., all representing subsets of E
- a unary predicate ind with $\mathrm{ind}(X)$ true if X is independent
- a unary predicate sing with $\mathrm{sing}(X)$ true if $|X| = 1$
- a binary relation $\subseteq$ with $X \subseteq Y$ true if X is a subset of Y
- $\wedge$, $\vee$, $\neg$, $\exists$, $\forall$, $\rightarrow$

We combine atomic formulae using connectives, to obtain a statement about some variables. We may combine these using further connectives. Then we quantify all variables to obtain an "axiom".

Every axiom *can* be written with all variables "in front".

language

We define a language with the following ingredients

- variables X, Y, Z, ..., all representing subsets of E
- a unary predicate ind with $\mathrm{ind}(X)$ true if X is independent
- a unary predicate sing with $\mathrm{sing}(X)$ true if $|X| = 1$
- a binary relation $\subseteq$ with $X \subseteq Y$ true if X is a subset of Y
- $\wedge$, $\vee$, $\neg$, $\exists$, $\forall$, $\rightarrow$

We combine atomic formulae using connectives, to obtain a statement about some variables. We may combine these using further connectives. Then we quantify all variables to obtain an "axiom".

Every axiom *can* be written with all variables "in front".

The language thus obtained is Monadic Second Order (MSO), with the given predicates/relations.

freebies

Various things we didn't include we get for free.
To get the set union $X \cup Y$, use Z where

$$X \subseteq Z \wedge Y \subseteq Z \wedge \forall T \, \{\mathsf{sing}(T) \wedge T \subseteq Z\} \rightarrow \{T \subseteq X \vee T \subseteq Y\}$$

freebies

Various things we didn't include we get for free.
To get the set union $X \cup Y$, use Z where

$$X \subseteq Z \wedge Y \subseteq Z \wedge \forall T \{\mathsf{sing}(T) \wedge T \subseteq Z\} \rightarrow \{T \subseteq X \vee T \subseteq Y\}$$

So $\forall X \exists Y \, \mathsf{ind}(X \cup Y)$ becomes

$$\forall X \exists Y \exists Z \, \mathsf{ind}(Z)$$
$$\wedge \left\{ X \subseteq Z \wedge Y \subseteq Z \wedge \forall T \{\mathsf{sing}(T) \wedge T \subseteq Z\} \rightarrow \{\vee T \subseteq X \vee T \subseteq Y\} \right\}$$

We can use union, intersection, etc, as if they were in our language (subroutines).

matroids

We can define matroids using this language.

matroids

We can define matroids using this language.

We check the basis exchange axiom.
First, a subroutine for "is a basis".

$$\mathsf{bas}(X) := I(X) \wedge \forall T \left\{ \mathsf{sing}(T) \wedge T \not\subseteq X \right\} \rightarrow \neg I(X \cup T)$$

Then, the axiom.

$$\begin{aligned}
&\forall B_1 \forall B_2 \forall T_1 \\
&\left\{ \mathsf{sing}(T_1) \wedge \mathsf{bas}(B_1) \wedge \mathsf{bas}(B_2) \wedge T_1 \subseteq B_1 \wedge T_1 \not\subseteq B_2 \right\} \\
&\rightarrow \\
&\exists T_2 \\
&\left\{ \mathsf{sing}(T_2) \wedge T_2 \not\subseteq B_1 \wedge T_2 \subseteq B_2 \wedge \mathsf{bas}(B_1 - T_1 \cup T_2) \right\}
\end{aligned}$$

minors

We can check whether a matroid has a fixed minor N. For instance, for $U_{2,4}$.

$$\exists Y \exists A \exists B \exists C \exists D$$
$$\mathsf{ind}(Y) \wedge \mathsf{sing}(A) \wedge \mathsf{sing}(B) \wedge \mathsf{sing}(C) \wedge \mathsf{sing}(D)$$
$$\wedge\, A \not\subseteq Y \wedge B \not\subseteq Y \wedge C \not\subseteq Y \wedge D \not\subseteq Y$$
$$\wedge\, \mathsf{ind}(Y \cup A \cup B) \wedge \mathsf{ind}(Y \cup A \cup C) \wedge \mathsf{ind}(Y \cup A \cup D)$$
$$\wedge\, \mathsf{ind}(Y \cup B \cup C) \wedge \mathsf{ind}(Y \cup B \cup D) \wedge \mathsf{ind}(Y \cup C \cup D)$$
$$\wedge\, \neg\, \mathsf{ind}(Y \cup A \cup B \cup C) \wedge \neg\, \mathsf{ind}(Y \cup A \cup B \cup D)$$
$$\wedge\, \neg\, \mathsf{ind}(Y \cup A \cup C \cup D) \wedge \neg\, \mathsf{ind}(Y \cup B \cup C \cup D)$$

goal

Question (Whitney?)

Is there an axiom in MSO that precisely captures real representability?

accepting strings

An (finite state) automaton is a machine with a fixed number of internal states. Some of these states are designated as “accepting” states.

accepting strings

An (finite state) automaton is a machine with a fixed number of internal states. Some of these states are designated as "accepting" states.

The automaton starts in some initial state, and takes as input a string of characters. It reads the characters one at a time, and at each character the current internal state is determined by the previous internal state and the character read.

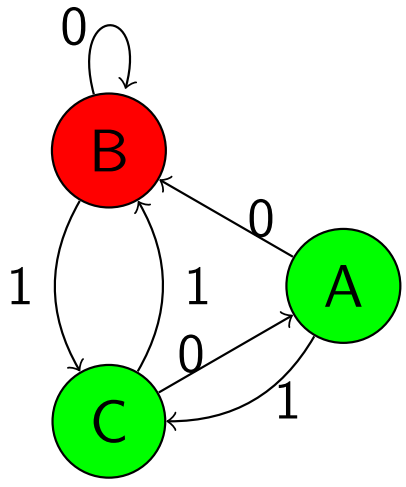
accepting strings

An (finite state) automaton is a machine with a fixed number of internal states. Some of these states are designated as "accepting" states.

The automaton starts in some initial state, and takes as input a string of characters. It reads the characters one at a time, and at each character the current internal state is determined by the previous internal state and the character read.

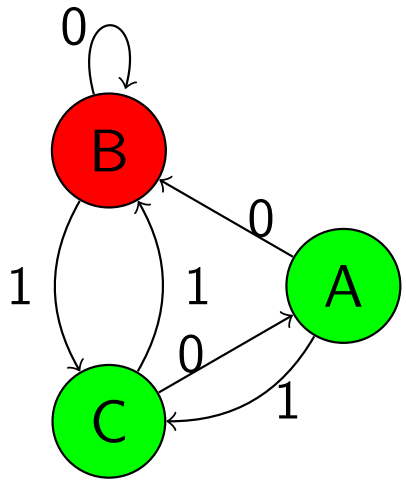
At the end of the string, the machine is in some state. If it is an "accepting" state then the string is deemed to be "accepted", otherwise the string is "rejected".

example



(convention: initial state is “A”)

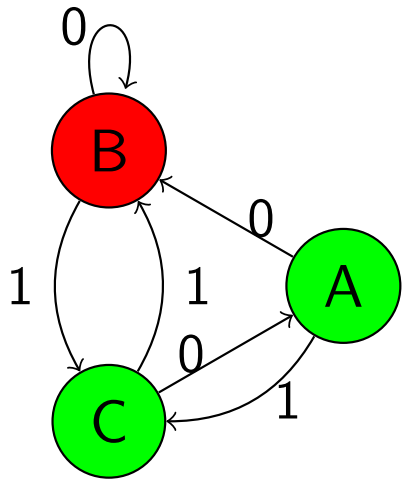
example



(convention: initial state is “A”)

0 1 1 1 0 0 1

example



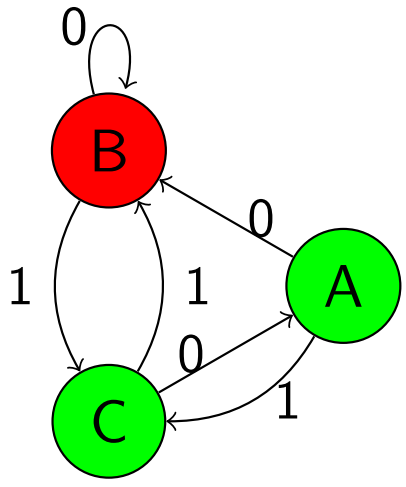
(convention: initial state is “A”)

start:

0	1	1	1	0	0	1

A

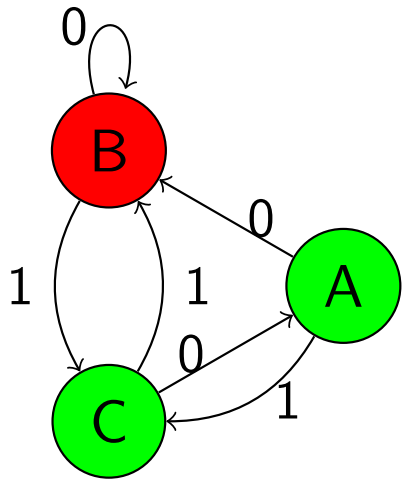
example



(convention: initial state is “A”)

0	1	1	1	0	0	1
B						

example

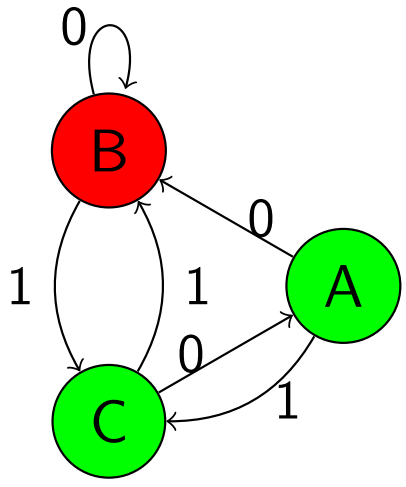


(convention: initial state is “A”)

0 1 1 1 0 0 1

C

example

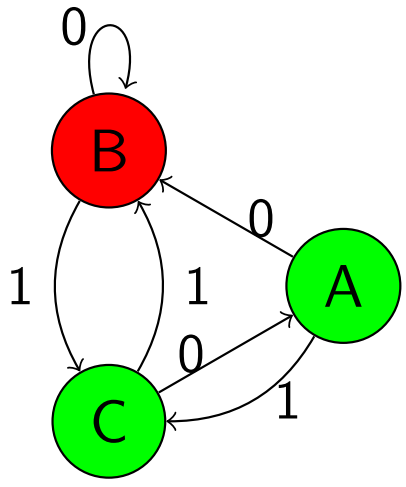


(convention: initial state is “A”)

0 1 1 1 0 0 1

B

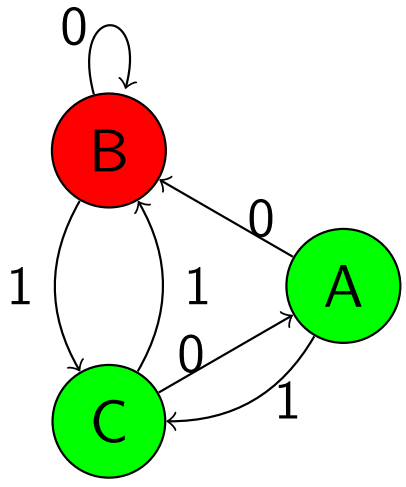
example



(convention: initial state is “A”)

0	1	1	1	0	0	1
			C			

example

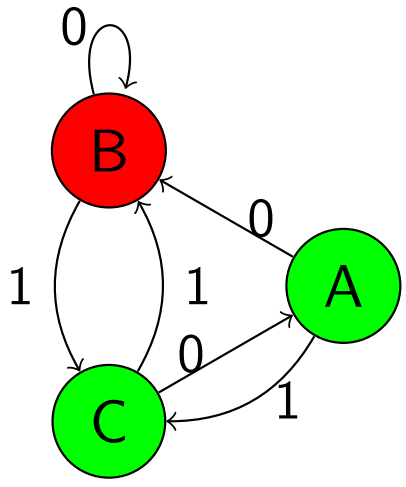


(convention: initial state is “A”)

0 1 1 1 0 0 1

A

example

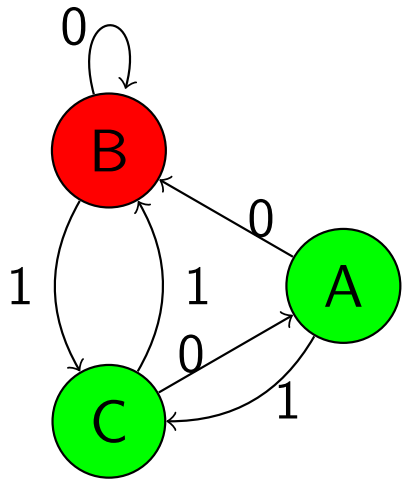


(convention: initial state is “A”)

0 1 1 1 0 0 1

B

example



(convention: initial state is “A”)

0 1 1 1 0 0 1

C

accept!

machines and axioms

Theorem (model theory)

There exists an automaton that accepts exactly the strings in a particular class if and only if there exists an MSO axiom that captures that class exactly.

machines and axioms

Theorem (model theory)

There exists an automaton that accepts exactly the strings in a particular class if and only if there exists an MSO axiom that captures that class exactly.

How do we encode a matroid as a string?

machines and axioms

Theorem (model theory)

There exists an automaton that accepts exactly the strings in a particular class if and only if there exists an MSO axiom that captures that class exactly.

How do we encode a matroid as a string?

Good question!!

matroid automata

We encode a matroid as an array of characteristic vectors of subsets of E in lexicographical order, followed by a flag for independence. We encode a set of sets $X_1, X_2, \cdots, X_k$ as an array of characteristic vectors. The machine must decide whether some "property" of these sets is true for the given matroid.

matroid automata

We encode a matroid as an array of characteristic vectors of subsets of E in lexicographical order, followed by a flag for independence. We encode a set of sets $X_1, X_2, \cdots, X_k$ as an array of characteristic vectors. The machine must decide whether some "property" of these sets is true for the given matroid.

The machine reads the vectors of the given sets one column at a time, and has a number of "readers" that can move down the current column. At each step, depending on its current state and what the readers see, it may direct the readers to move down one step or wait. When all readers are waiting, the machine moves to the next column, and continues.

matroid automata

We encode a matroid as an array of characteristic vectors of subsets of E in lexicographical order, followed by a flag for independence. We encode a set of sets $X_1, X_2, \cdots, X_k$ as an array of characteristic vectors. The machine must decide whether some "property" of these sets is true for the given matroid.

The machine reads the vectors of the given sets one column at a time, and has a number of "readers" that can move down the current column. At each step, depending on its current state and what the readers see, it may direct the readers to move down one step or wait. When all readers are waiting, the machine moves to the next column, and continues.

urghk?

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\, \mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

example 1

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	A			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	0	**1**	1	
X_2	1	**1**	0	
X_3	1	**0**	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	0	**1**	1	
X_2	1	**1**	0	
X_3	1	**0**	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathsf{ind}(X_1 \cap X_2) \wedge \neg\,\mathsf{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	0	**1**	1	
X_2	1	**1**	0	
X_3	1	**0**	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state			B	
X_1	0	1	**1**	
X_2	1	1	**0**	
X_3	1	0	**0**	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state				B
X_1	0	1	1	
X_2	1	1	0	
X_3	1	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state				C
X_1	0	1	1	
X_2	1	1	0	
X_3	1	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 1

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge X_3 \subseteq X_2$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state				:-(
X_1	0	1	1	
X_2	1	1	0	
X_3	1	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state				
X_1	0	1	1	
X_2	1	1	0	
X_3	1	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	A			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	**0**	1	1	
X_2	**1**	1	0	
X_3	**1**	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state	B			
X_1	0	**1**	1	
X_2	1	**1**	0	
X_3	1	**0**	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state			D	
X_1	0	1	**1**	
X_2	1	1	**0**	
X_3	1	0	**0**	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state				D
X_1	0	1	1	
X_2	1	1	0	
X_3	1	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

example 2

Machine implementing $\mathrm{ind}(X_1 \cap X_2) \wedge \neg\,\mathrm{ind}(X_3) \wedge \mathrm{sing}(X_2)$.

Input matroid $M = U_{2,3}$ on ground set $\{a_1, a_2, a_3\}$.
Checking $X_1 = \{a_2, a_3\}$, $X_2 = \{a_1, a_2\}$, $X_3 = \{a_1\}$.

state				:-(
X_1	0	1	1	
X_2	1	1	0	
X_3	1	0	0	
	0	0	0	ind
	0	0	1	ind
	0	1	0	ind
$U_{2,3}$	0	1	1	ind
	1	0	0	ind
	1	0	1	ind
	1	1	0	ind
	1	1	1	dep

quantifiers?

How do we do $\forall X_1 X_2 X_3 \{\mathsf{ind}(X_1 \cap X_2) \wedge \neg\, \mathsf{ind}(X_3) \wedge X_2 \subseteq X_3\}$?

We consider a larger machine that runs all possible subsets simultaneously. . .

upgrade

Hypothesis

There exists a matroid automaton that accepts exactly the matroids in a particular class if and only if there exists an MSO axiom that captures that class exactly.

idea

There are a finite number of states in the machine, so there must be two different matroids that leave the machine in the same state.

We want to attach some other matroid to both of them such that one of the two resulting compound matroids is representable and the other is not.

The machine will get half-way through and be in the same state for both matroids, after which all remaining input is identical, so the final state will be the same.

Thus, the machine will not be able to tell them apart.

sums

Consider matroids of the form $M = M_1 \oplus M_2$, where we represent them as an array for M_1 followed by an array for M_2.
The independent sets of M are unions of independent sets of M_1 and independent sets of M_2. So in order to decide if a set is independent in M, it suffices to decide if both restrictions are independent.

sums

Consider matroids of the form $M = M_1 \oplus M_2$, where we represent them as an array for M_1 followed by an array for M_2.
The independent sets of M are unions of independent sets of M_1 and independent sets of M_2. So in order to decide if a set is independent in M, it suffices to decide if both restrictions are independent.

The machine will finish processing M_1, and then be in some state, and then process M_2.

Key observation: The only information it has about M_1 while it is looking at M_2 is the final state after M_1. This information is *bounded*.

representability

Theorem (Mayhew, N, Whittle)

There is no axiom in MSO that precisely captures the matroids that are representable over some field.

representability

Theorem (Mayhew, N, Whittle)

There is no axiom in MSO that precisely captures the matroids that are representable over some field.

For each prime p_i let P_i be a projective geometry over a field of characteristic p_i.

Since the machine has a bounded number of states then there must be distinct j and k such that reading P_j or reading P_k leaves the machine in the same state (Euclid -300).

Consider the matroids $P_j \oplus P_k$ and $P_k \oplus P_k$.

The automaton must end in the same final state and must therefore accept both or reject both. But the second is representable while the first is not.

amalgams

Consider matroids of the form $M_1 \star M_2$, where "$\star$" means perform a proper amalgam on a specified set T. Precisely, we assume that the intersection of the ground sets is T and the restriction of each M_i to T gives a line.

Lemma

This proper amalgam exists. (Oxley p###)
The circuits of $M_1 \star M_2$ are either circuits of M_1, circuits of M_2, or symmetric differences of a circuit of M_1 with a circuit of M_2.

Dowling gadgets

Consider the Dowling matroids X_i and Y_i described on the ~~blackboard~~ next slide.

Dowling gadgets

Consider the Dowling matroids X_i and Y_i described on the ~~blackboard~~ next slide.

Note that X_i and Y_i are matroids, not representations.

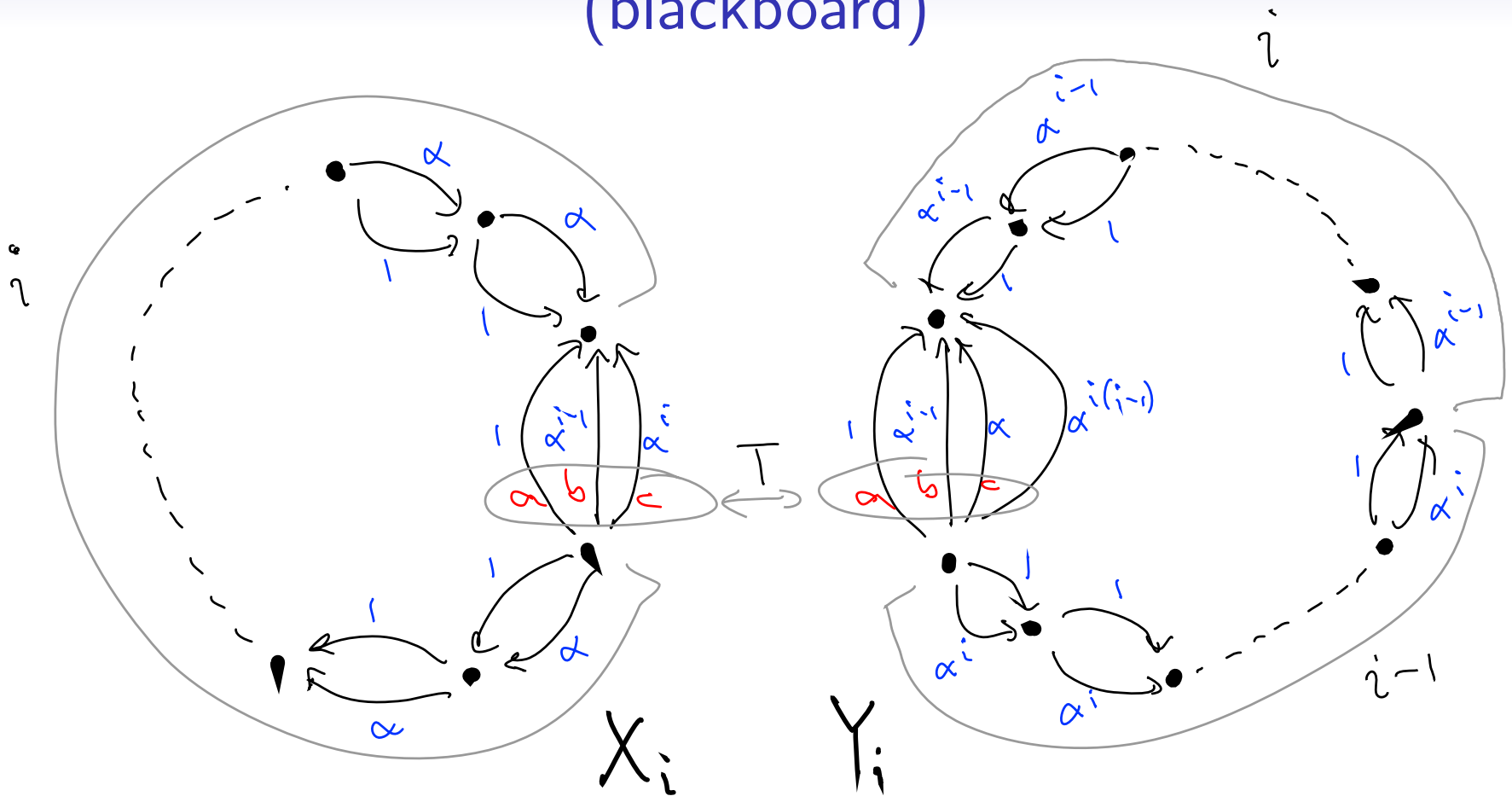
Lemma

X_i and Y_i are representable over any field with an element of sufficiently large order (e.g. infinite).

Lemma

$X_j \star Y_k$ is representable if and only if $j = k$.

(blackboard)


i
a
1
a
1
a
1
a^{i-1}
a^i
a
b
c
1
1
a
1
a
T
i
a^{i-1}
a^{i-1}
1
1
1
a^{i-1}
a^{i(i-1)}
a
a^{i-1}
b
c
1
a^i
1
1
a^i
a^i
i-1
X_i
Y_i

real representability

Theorem (Mayhew, N, Whittle)

There is no axiom in MSO that precisely captures the matroids that are representable over the reals.

(Actually, this applies to any infinite field.)

real representability

Theorem (Mayhew, N, Whittle)

There is no axiom in MSO that precisely captures the matroids that are representable over the reals.

(Actually, this applies to any infinite field.)

Since the machine has a bounded number of states then there must be distinct j and k such that reading X_j or reading X_k leaves the machine in the same state.

Consider the matroids $X_j \star Y_k$ and $X_k \star Y_k$.

The automaton must end in the same final state and must therefore accept both or reject both. But the second is real representable while the first is not.

axioms and classes

Consider any axiom; it defines a class of matroids (those matroids for which it is true). Call a class axiomatizable if there is an axiom that precisely captures it.

So: binary matroids are axiomatizable, real-representable matroids are not.

axioms and classes

Consider any axiom; it defines a class of matroids (those matroids for which it is true). Call a class axiomatizable if there is an axiom that precisely captures it.

So: binary matroids are axiomatizable, real-representable matroids are not.

Question: What are the axiomatizable classes? What can we say about them?

Question: Is there an axiomatizable class $\mathcal{M}$ such that approximates the class of real representale matroids?