Combinatorial Expressions and Lower Bounds

Thomas Colcombet and Amaldev Manuel

STACS 2015
6/3/2015, München

Motivation

Show that BMA is strictly included in BR.

Motivation

Two walking logics over
data words.

Show that BMA is strictly included in BR.

Motivation

Two walking logics over
data words.

Show that BMA is strictly included in BR.

Strictness:
Construct of formula of BR (easy), and assume it is equivalent to a BMA formula.

Motivation

Two walking logics over
data words.

Show that BMA is strictly included in BR.

Strictness:
Construct of formula of BR (easy), and assume it is equivalent to a BMA formula.

Devise a family of specially shaped inputs encoding, e.g., sequence of numbers.

Motivation

Two walking logics over
data words.

Show that BMA is strictly included in BR.

Strictness:
Construct of formula of BR (easy), and assume it is equivalent to a BMA formula.

Devise a family of specially shaped inputs encoding, e.g., sequence of numbers.

Compile the BMA formula over these inputs into a circuit-like model that involves two value types, combinatorial expressions:

- Boolean values, or values ranging over a finite, bounded set F
- large values, ranging over an infinite or unbounded set D (numbers)

Motivation

Two walking logics over data words.

Show that BMA is strictly included in BR.

Strictness:
Construct of formula of BR (easy), and assume it is equivalent to a BMA formula.

Devise a family of specially shaped inputs encoding, e.g., sequence of numbers.

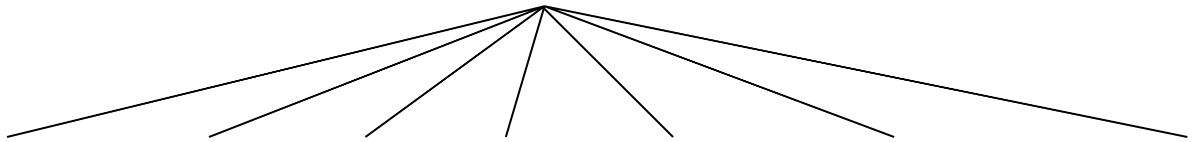
Compile the BMA formula over these inputs into a circuit-like model that involves two value types, combinatorial expressions:

- Boolean values, or values ranging over a finite, bounded set F
- large values, ranging over an infinite or unbounded set D (numbers)

Show a lower bound result on these combinatorial expressions.

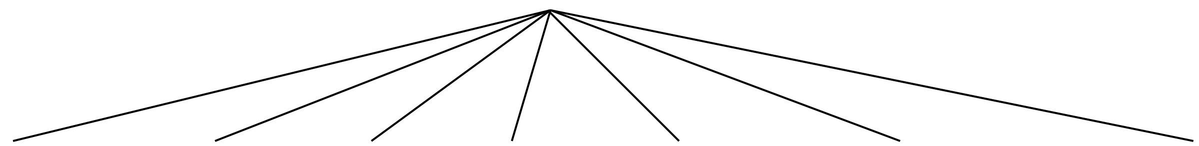
Combinatorial expressions

Combinatorial expressions



gates/functions over a **finite domain F** of unrestricted fan-in.

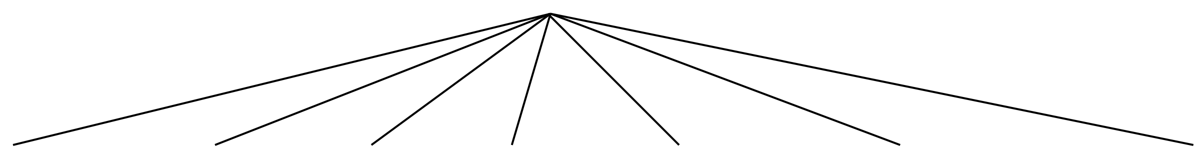
Combinatorial expressions



gates/functions over a **finite domain F** of unrestricted fan-in.

E.g., disjunction, conjunction, majority, modulo, languages…

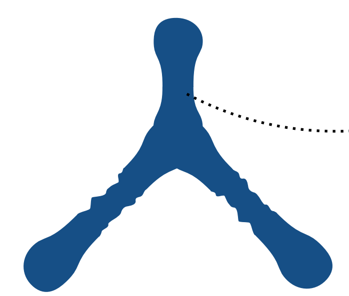
Combinatorial expressions



gates/functions over a **finite domain F** of unrestricted fan-in.

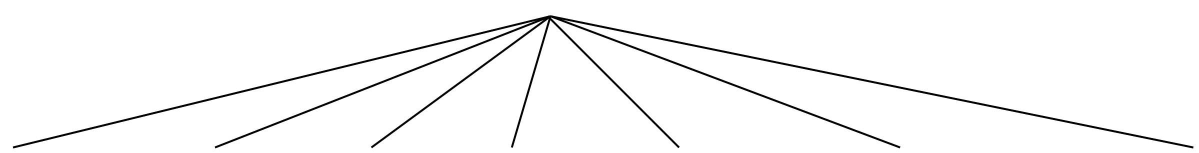
E.g., disjunction, conjunction, majority, modulo, languages…

Binary gates/functions over an un-bounded/infinite domain D (e.g, integers, reals,…) of fan-in 2.



Thick blue lines are wires propagating values from D

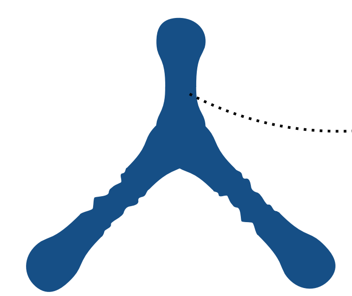
Combinatorial expressions



gates/functions over a **finite domain F** of unrestricted fan-in.

E.g., disjunction, conjunction, majority, modulo, languages…

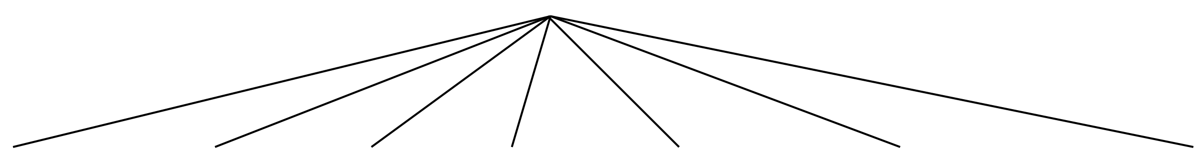
Binary gates/functions over an un-bounded/infinite domain D (e.g, integers, reals,…) of fan-in 2.



Thick blue lines are wires propagating values from D

E.g., +, x, =, <, prime, halt, (any function even non recursive)

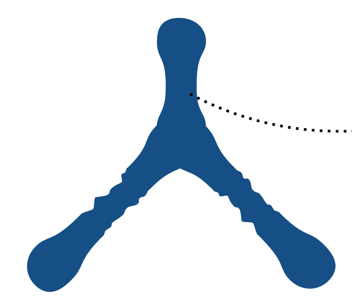
Combinatorial expressions



gates/functions over a **finite domain F** of unrestricted fan-in.

E.g., disjunction, conjunction, majority, modulo, languages…

Binary gates/functions over an un-bounded/infinite domain D (e.g, integers, reals,…) of fan-in 2.



Thick blue lines are wires propagating values from D

E.g., +, x, =, <, prime, halt, (any function even non recursive)

Combinatorial expressions use such gates/functions and have bounded height (say, by h).

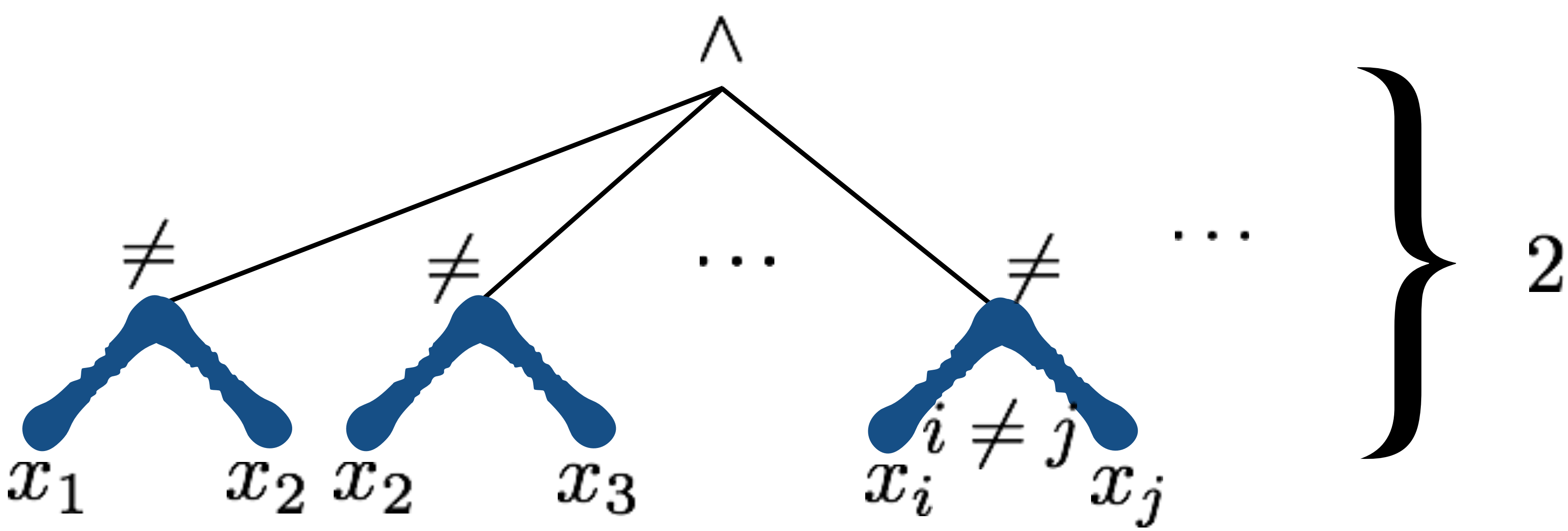
Example

Example

All inputs are distinct

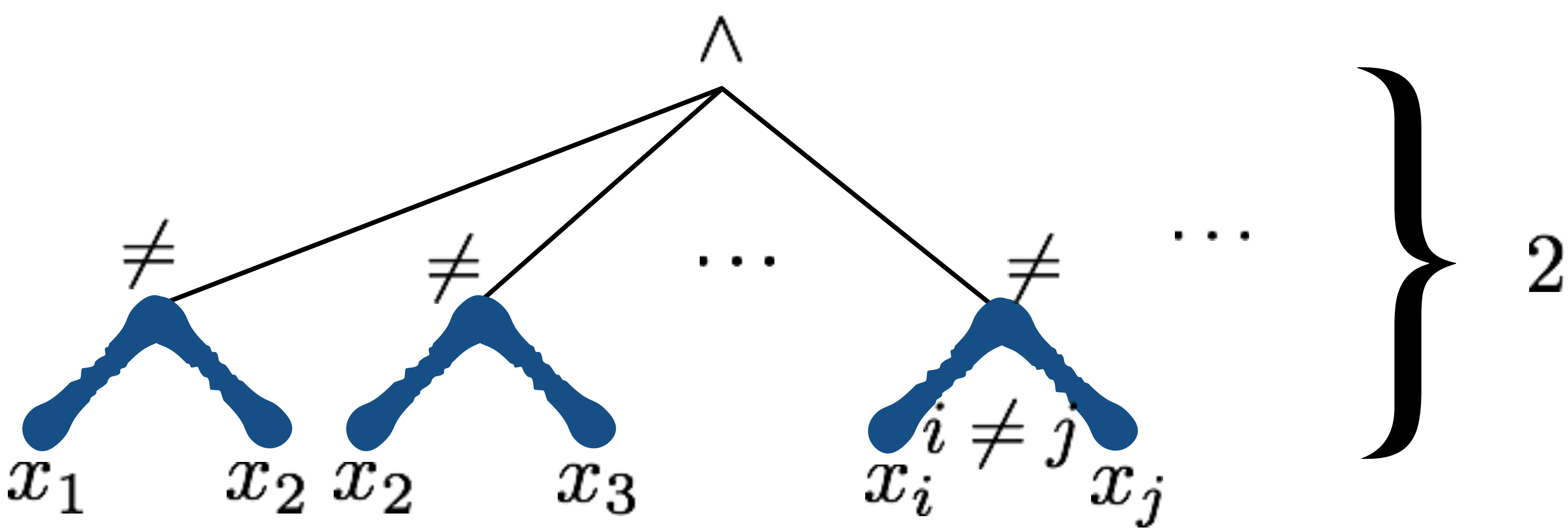
Example

All inputs are distinct

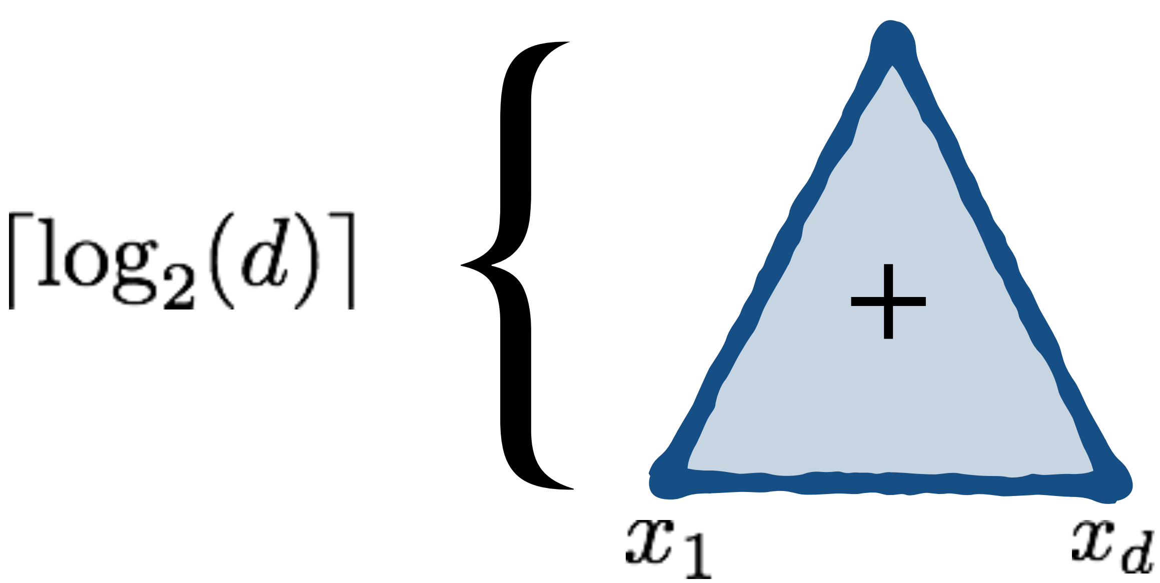


Example

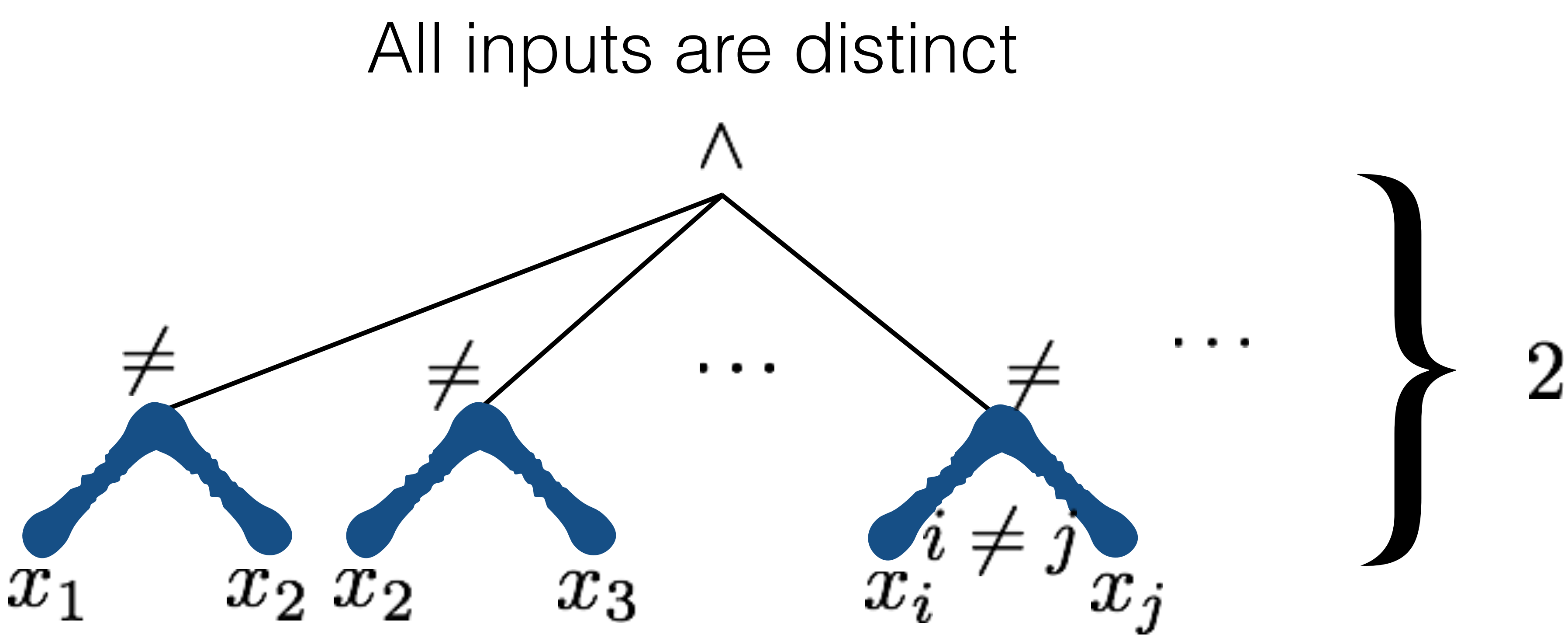
All inputs are distinct

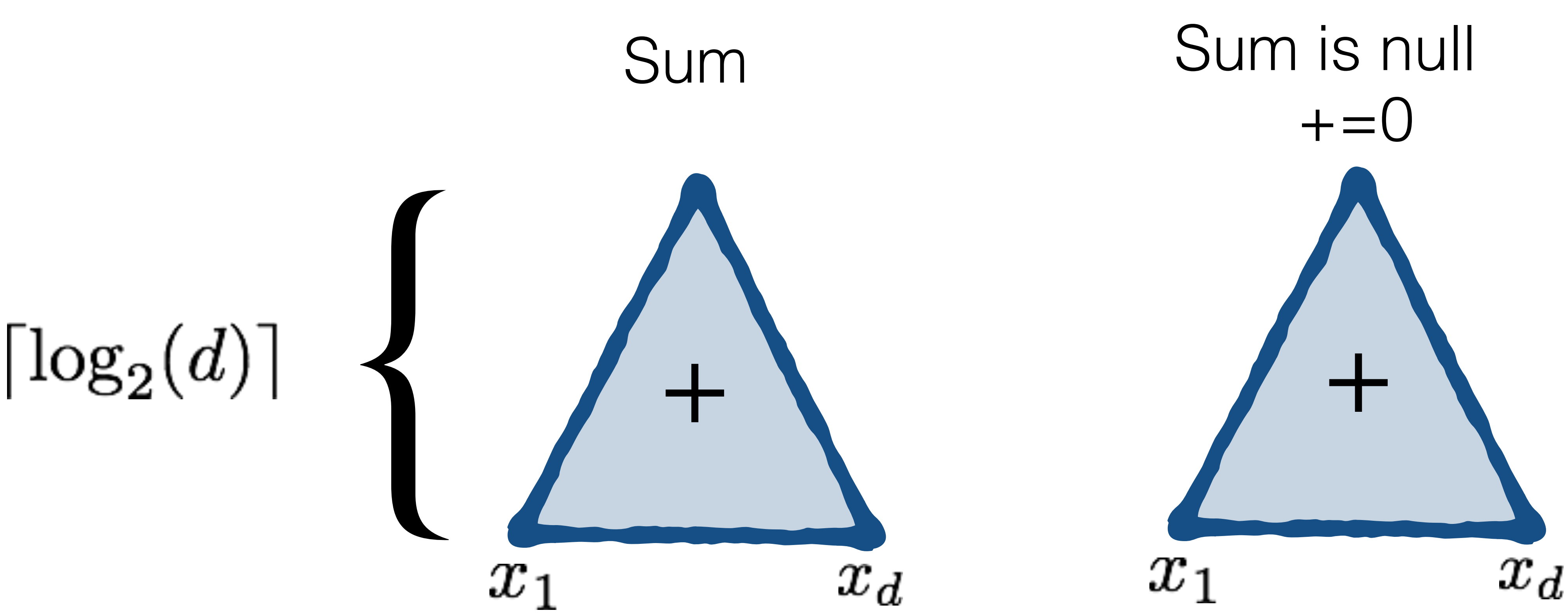


Sum



Example

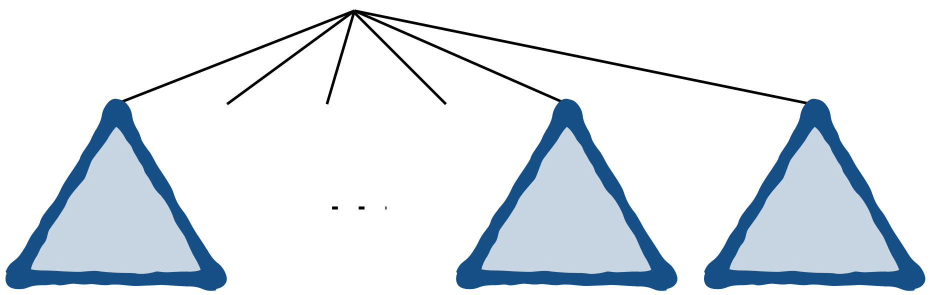

All inputs are distinct
$\wedge$
$\neq$
$\neq$
$\cdots$
$\neq$
$\cdots$
2
$i \neq j$
x_1
x_2
x_2
x_3
x_i
x_j


Sum
Sum is null
+=0
$\lceil \log_2(d) \rceil$
+
+
x_1
x_d
x_1
x_d

Normalization of expressions

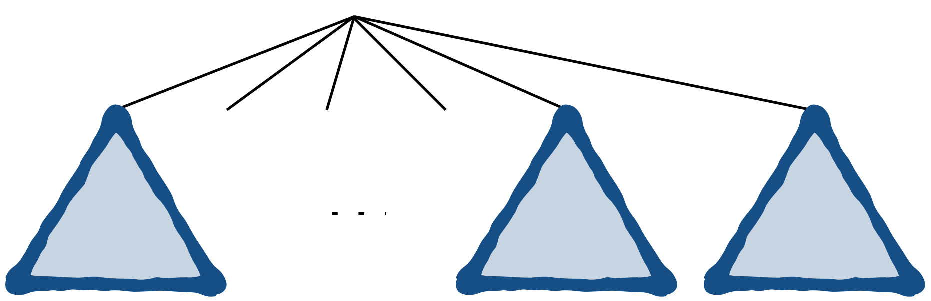
Normalization of expressions

All expressions of height h and output in B can be transformed into a expressions of height h+1 and shape:

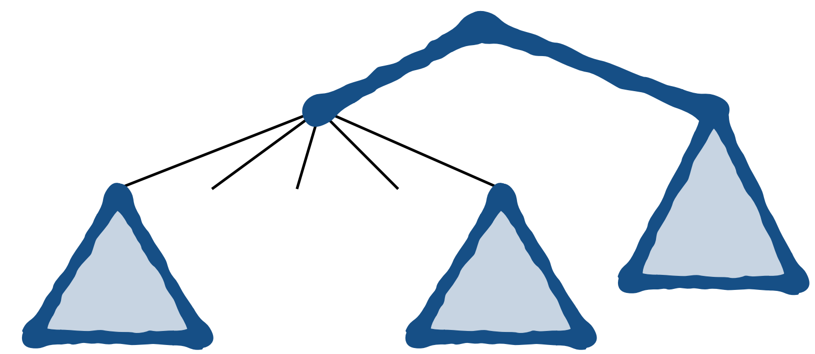


Normalization of expressions

All expressions of height h and output in B can be transformed into a expressions of height h+1 and shape:

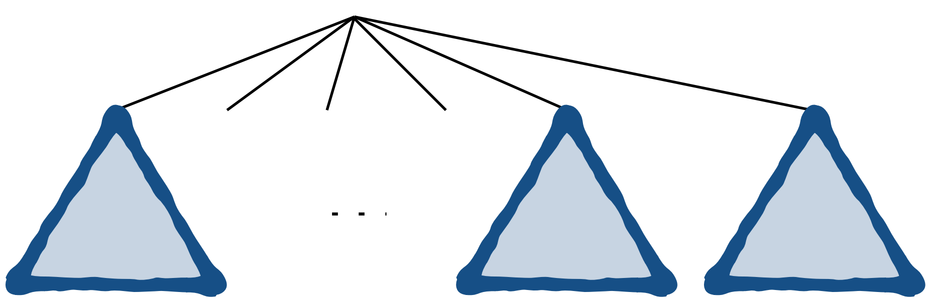


All expressions of height h can be transformed into an expression of height h+2 and shape:

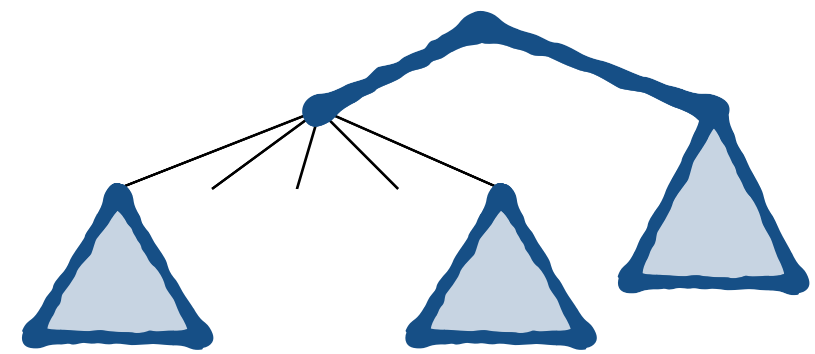


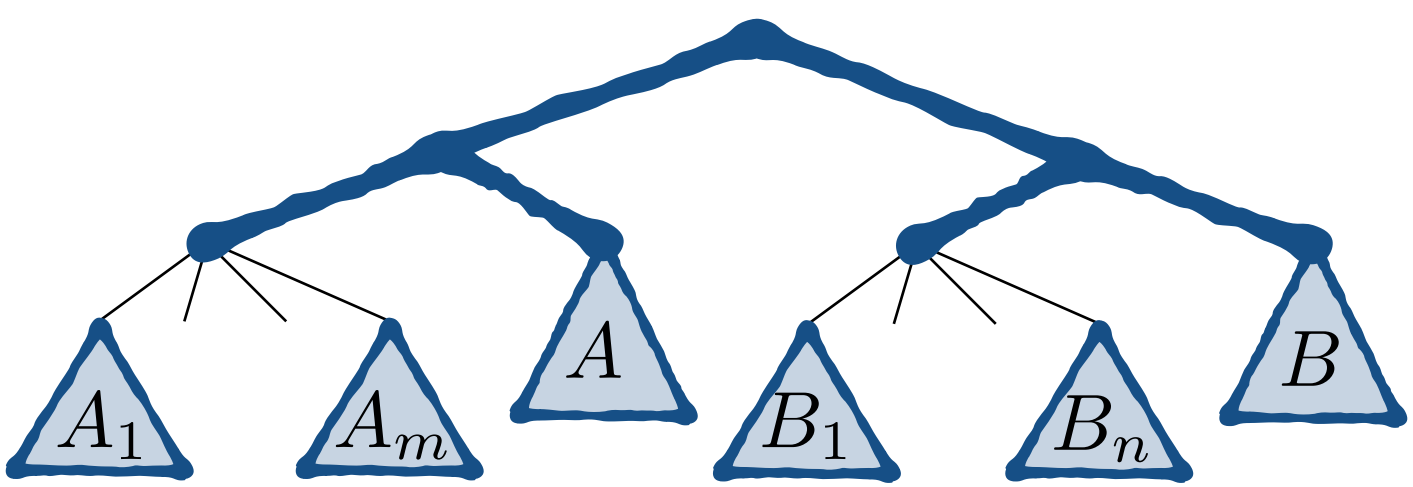
Normalization of expressions

All expressions of height h and output in B can be transformed into a expressions of height h+1 and shape:



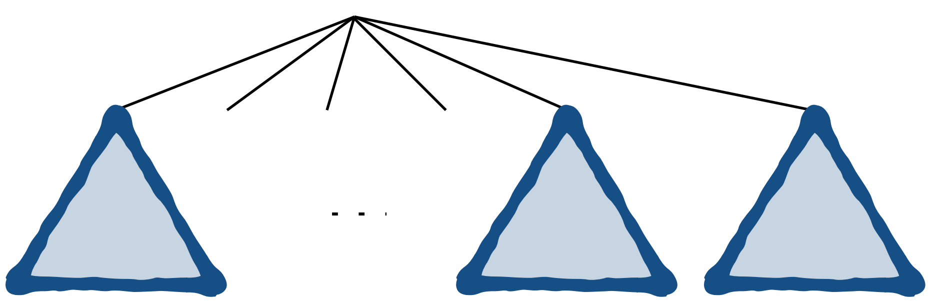
All expressions of height h can be transformed into an expression of height h+2 and shape:



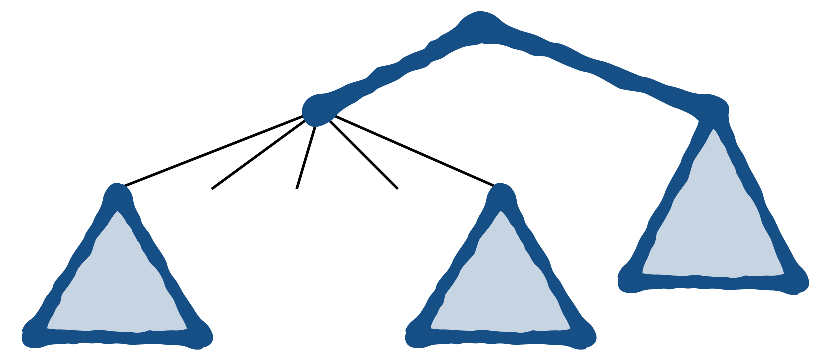


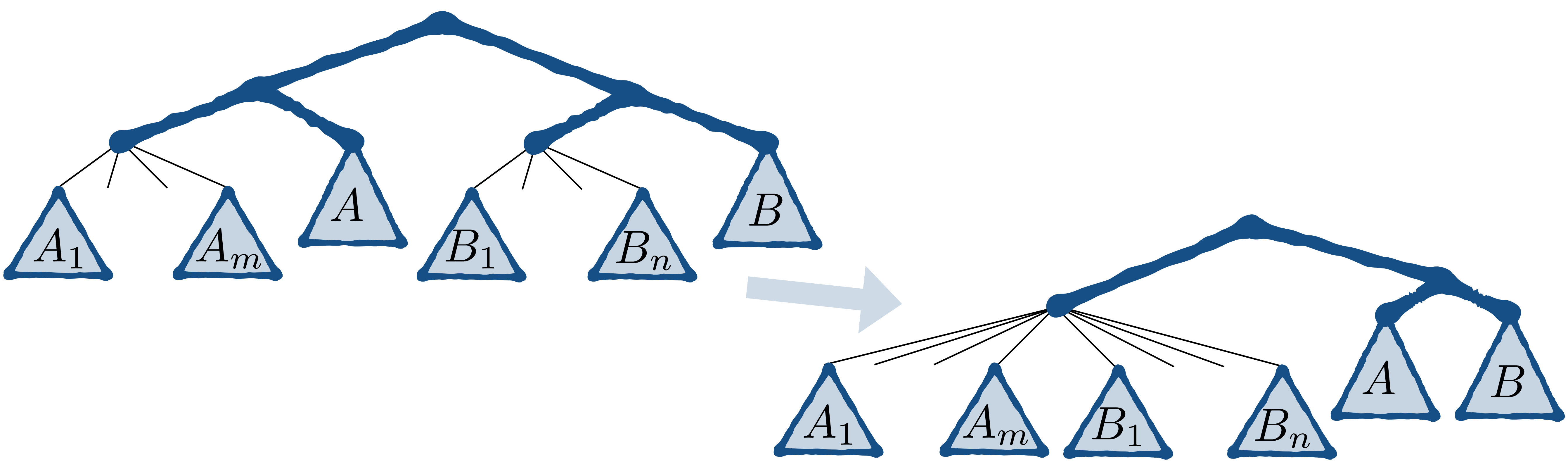
Normalization of expressions

All expressions of height h and output in B can be transformed into a expressions of height h+1 and shape:



All expressions of height h can be transformed into an expression of height h+2 and shape:





Expressiveness questions

Expressiveness questions

For representing functions (output in D)

Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

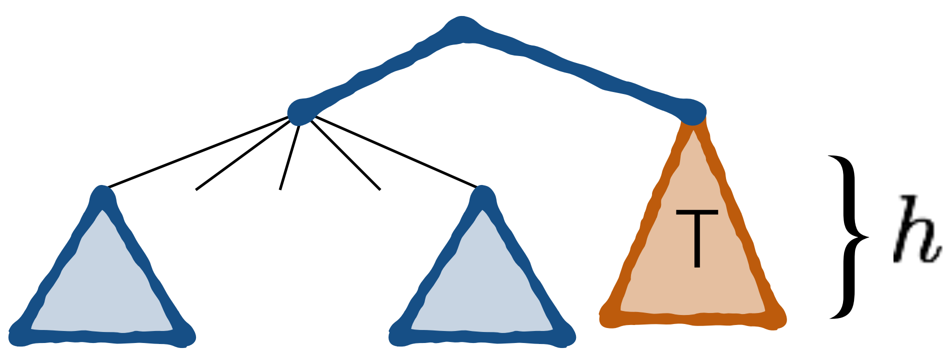
Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;



normalized expression

Expressiveness questions

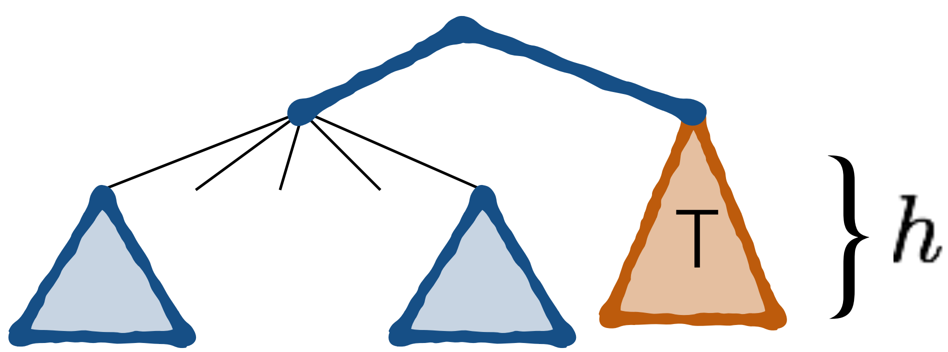
For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;

There is an input x not used in T.



normalized expression

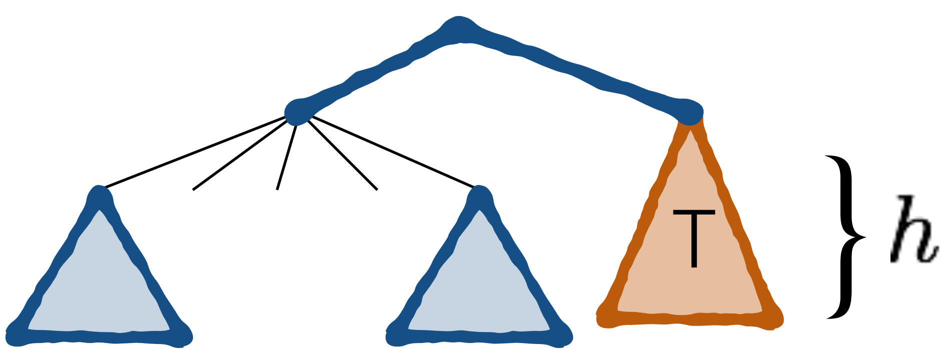
Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;



normalized expression

There is an input x not used in T.

When only this input ranges, the output can only take finitely many values.

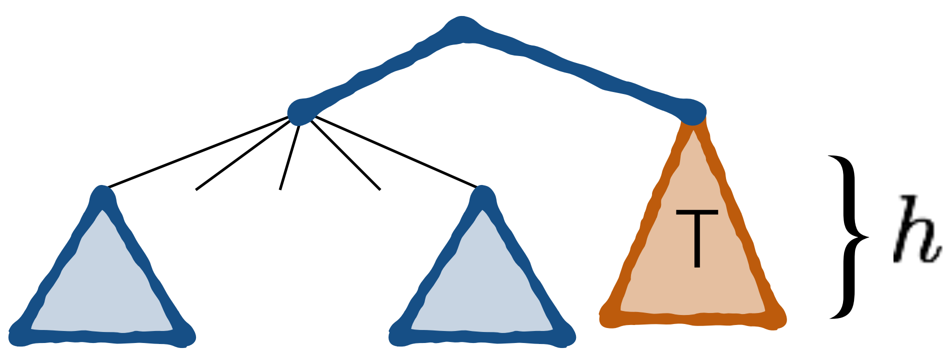
Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;



normalized expression

There is an input x not used in T.

When only this input ranges, the output can only take finitely many values.

This is not the case for **sum**.

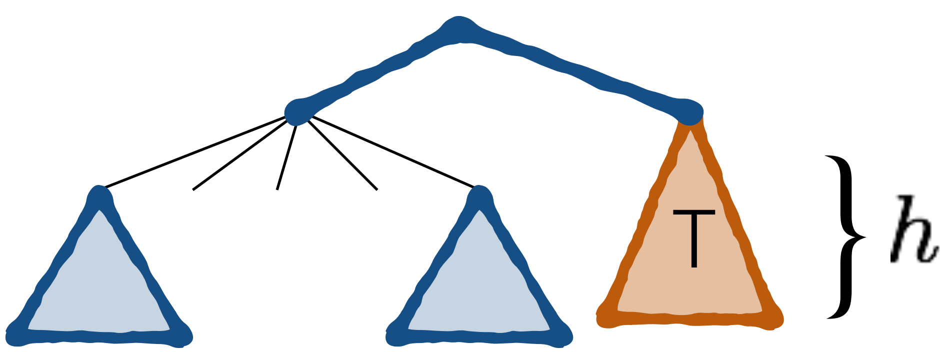
Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;



normalized expression

There is an input x not used in T.

When only this input ranges, the output can only take finitely many values.

This is not the case for **sum**.

For computing problems (Boolean output)

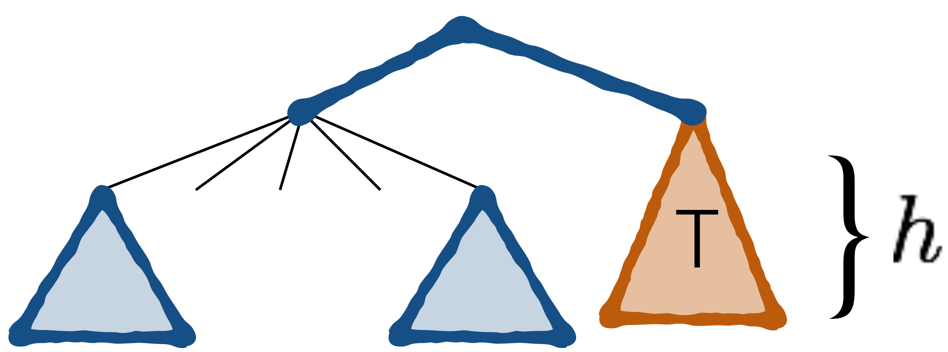
Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;



normalized expression

There is an input x not used in T.

When only this input ranges, the output can only take finitely many values.

This is not the case for **sum**.

For computing problems (Boolean output)

Is it possible to express that a sum is 0 ?

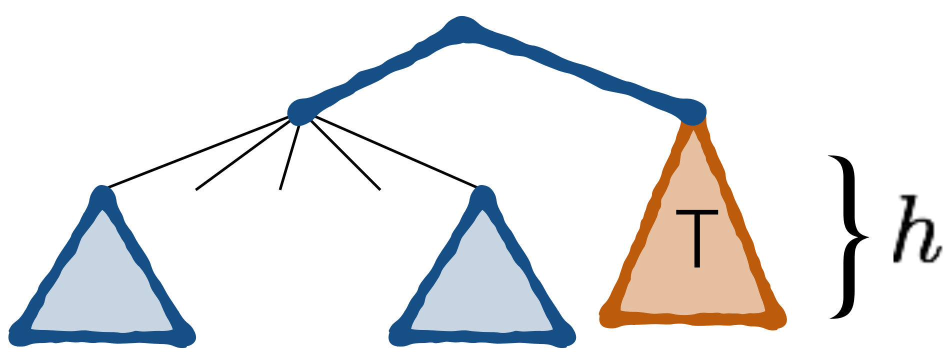
Expressiveness questions

For representing functions (output in D)

Can the **sum** of d integers as input be computed by a combinatorial expression?

No if $d > 2^h$.

Proof: by contradiction;



normalized expression

There is an input x not used in T.

When only this input ranges, the output can only take finitely many values.

This is not the case for **sum**.

For computing problems (Boolean output)

Is it possible to express that a sum is 0 ?

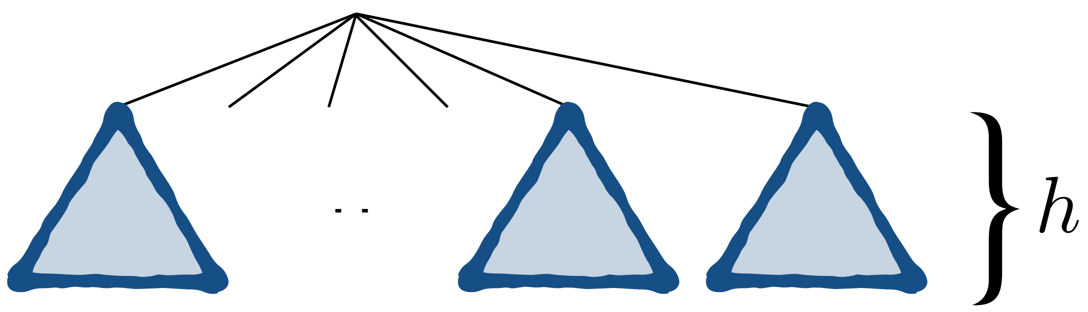
Is it possible to express that the gcd is 1 ?

Window definability

After normalization:

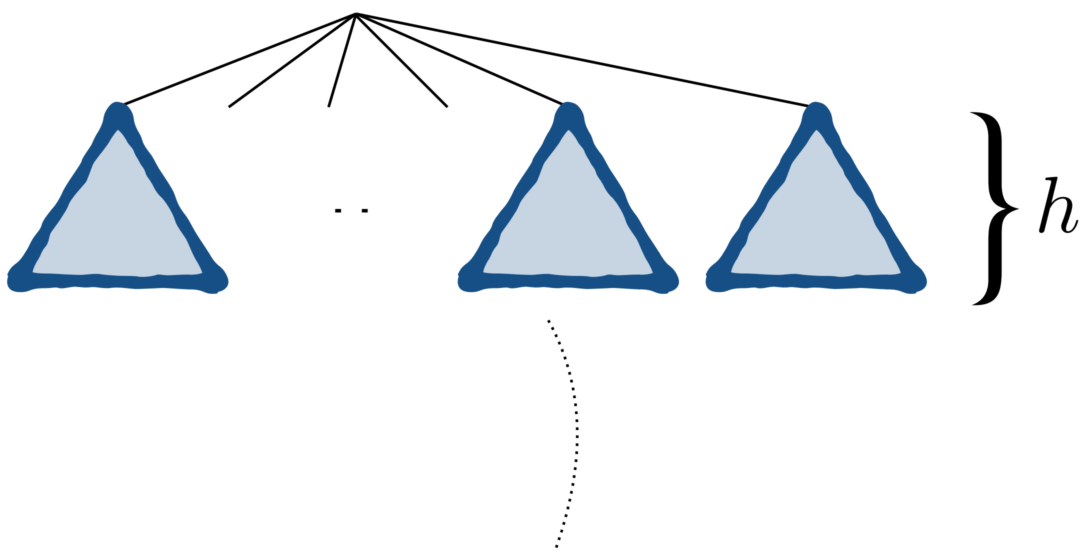
Window definability

After normalization:



Window definability

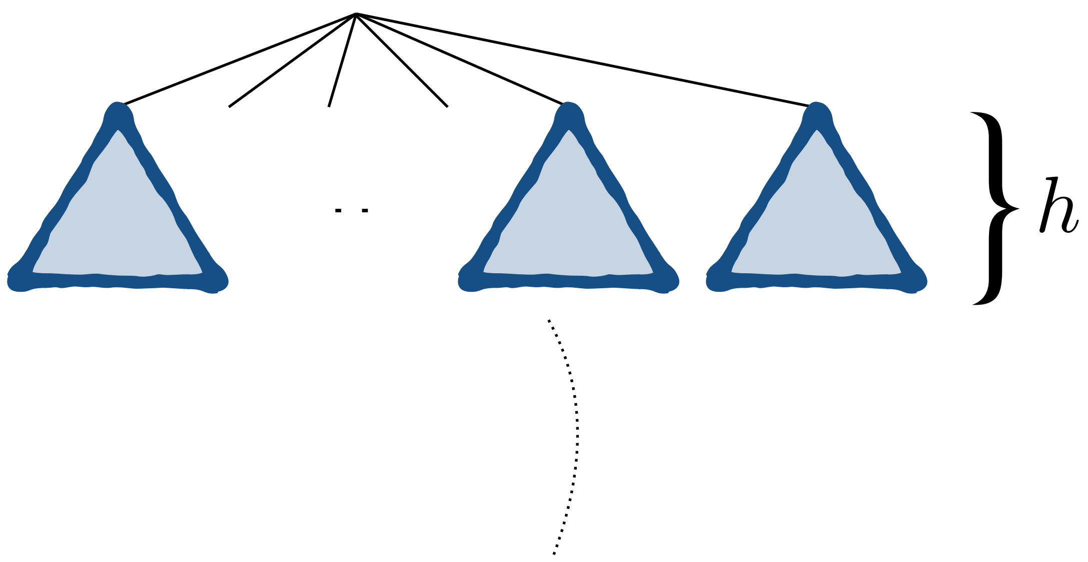
After normalization:



Each sub-tree uses at most
2^h distinct inputs.

Window definability

After normalization:

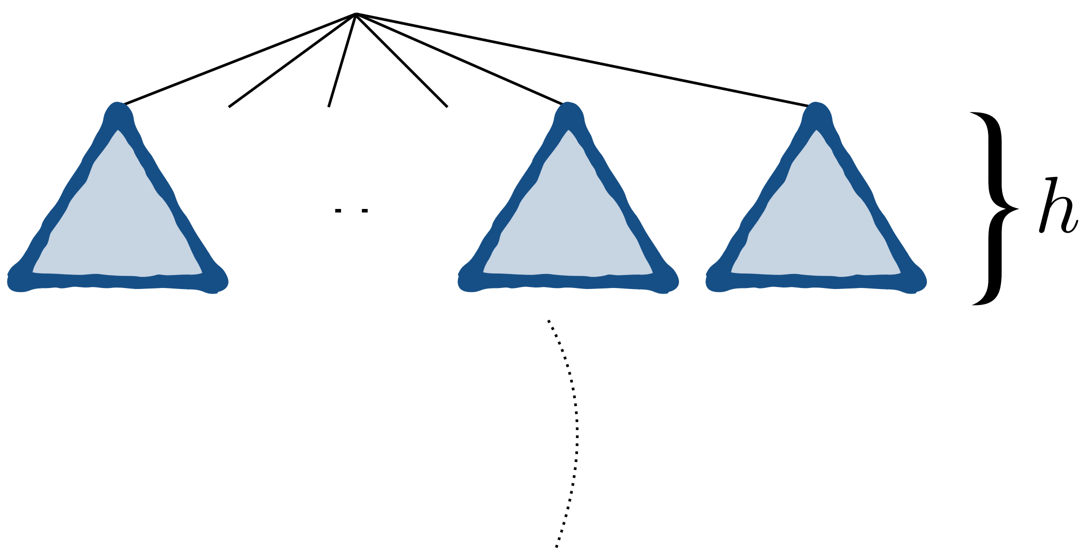


Each sub-tree uses at most
2^h distinct inputs.

Hence: If a problem is expressible with a combinatorial expression of height at most h, it is a Boolean combination of problems involving at most 2^h inputs.

Window definability

After normalization:



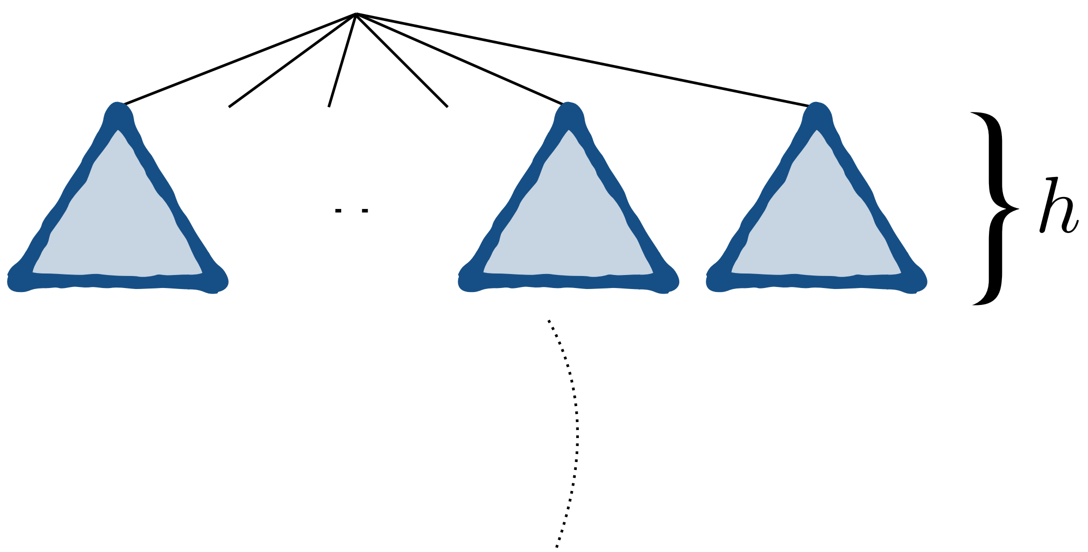
Each sub-tree uses at most 2^h distinct inputs.

Hence: If a problem is expressible with a combinatorial expression of height at most h, it is a Boolean combination of problems involving at most 2^h inputs.

Let $\mathcal{W} \subseteq \mathcal{P}(\{1, \ldots, d\})$ be a set of windows.

Window definability

After normalization:



Each sub-tree uses at most 2^h distinct inputs.

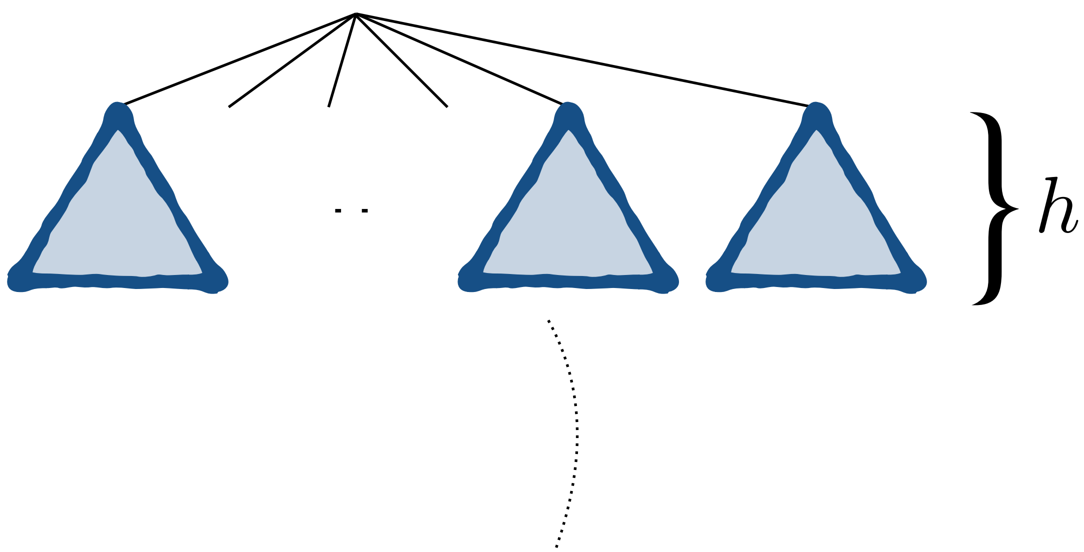
Hence: If a problem is expressible with a combinatorial expression of height at most h, it is a Boolean combination of problems involving at most 2^h inputs.

Let $\mathcal{W} \subseteq \mathcal{P}(\{1, \ldots, d\})$ be a set of windows.

A problem $P \subseteq D^d$ is $\mathcal{W}$-definable if it is a Boolean combination of W-definable languages for $W \in \mathcal{W}$.

Window definability

After normalization:



Each sub-tree uses at most 2^h distinct inputs.

Let $\mathcal{W} \subseteq \mathcal{P}(\{1, \ldots, d\})$ be a set of windows.

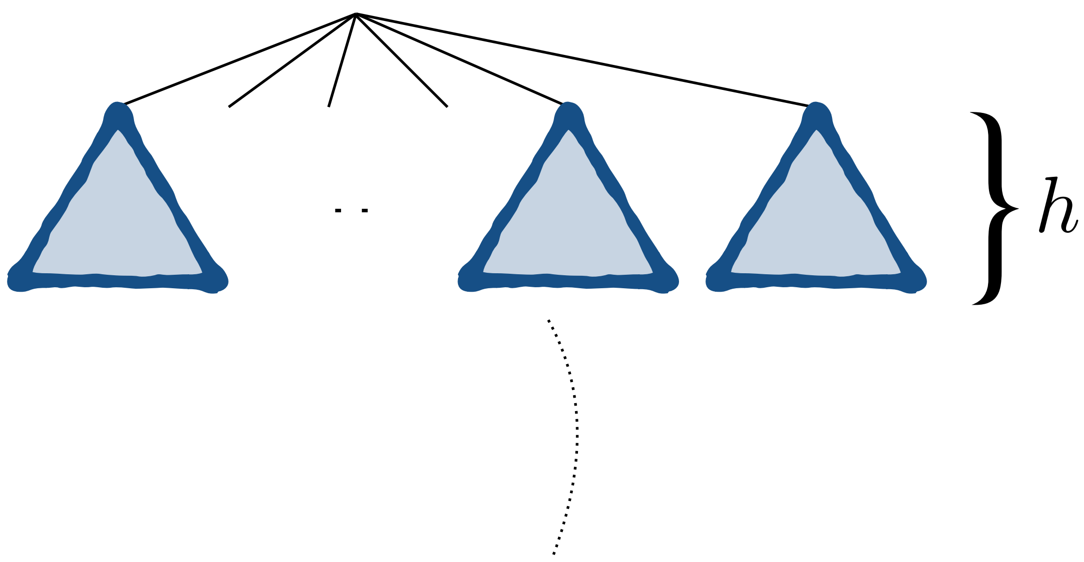
A problem $P \subseteq D^d$ is $\mathcal{W}$-definable if it is a Boolean combination of W-definable languages for $W \in \mathcal{W}$.

That depend only of the inputs from W.

Hence: If a problem is expressible with a combinatorial expression of height at most h, it is a Boolean combination of problems involving at most 2^h inputs.

Window definability

After normalization:



Each sub-tree uses at most 2^h distinct inputs.

Hence: If a problem is expressible with a combinatorial expression of height at most h, it is a Boolean combination of problems involving at most 2^h inputs.

Let $\mathcal{W} \subseteq \mathcal{P}(\{1, \ldots, d\})$ be a set of windows.

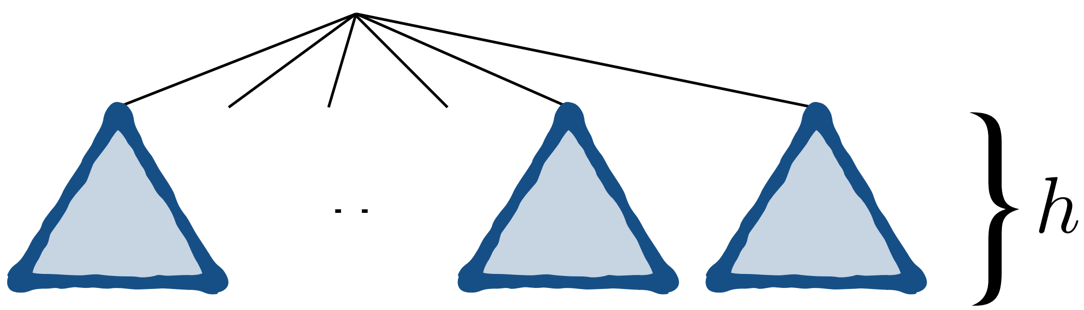
A problem $P \subseteq D^d$ is $\mathcal{W}$-definable if it is a Boolean combination of W-definable languages for $W \in \mathcal{W}$.

That depend only of the inputs from W.

It is an extension of the `input on the forehead' model.

Window definability

After normalization:



Each sub-tree uses at most 2^h distinct inputs.

Hence: If a problem is expressible with a combinatorial expression of height at most h, it is a Boolean combination of problems involving at most 2^h inputs.

Let $\mathcal{W} \subseteq \mathcal{P}(\{1,\ldots,d\})$ be a set of windows.

A problem $P \subseteq D^d$ is $\mathcal{W}$-definable if it is a Boolean combination of W-definable languages for $W \in \mathcal{W}$.

That depend only of the inputs from W.

It is an extension of the `input on the forehead' model.

Are the problems **sum=0** and **gcd=1** $\mathcal{W}$-definable for $\mathcal{W}$ non-trivial (*i.e.*, not containing the full window)?

Picture problems and reductions

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

Picture problems and reductions

A picture problem is when:
- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:
$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

1	1	1	1	0	1	1
0	1	1	1	1	1	0
1	1	0	1	1	1	0
1	1	1	1	0	1	1
1	1	1	0	1	1	1
1	1	1	1	0	1	1
0	1	1	1	1	1	1
1	1	1	1	0	1	1
1	1	1	0	1	1	1
x_1	x_2	x_3	x_4	x_5	x_6	x_7

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

1	1	1		0	1	1
0	1	1		1	1	0
1	1	0		1	1	0
1	1	1		0	1	1
1	1	1		1	1	1
1	1	1		0	1	1
0	1	1		1	1	1
1	1	1		0	1	1
1	1	1		1	1	1
x_1	x_2	x_3	x_4	x_5	x_6	x_7

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

1	1	1	1		1	1
0	1	1	1		1	0
1	1	0	1		1	0
1	1	1	1		1	1
1	1	1	0		1	1
1	1	1	1		1	1
0	1	1	1		1	1
1	1	1	1		1	1
1	1	1	0		1	1
x_1	x_2	x_3	x_4	x_5	x_6	x_7

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

For instance:

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

1	1	1	1		1	1
0	1	1	1		1	0
1	1	0	1		1	0
1	1	1	1		1	1
1	1	1	0		1	1
1	1	1	1		1	1
0	1	1	1		1	1
1	1	1	1		1	1
1	1	1	0		1	1
x_1	x_2	x_3	x_4	x_5	x_6	x_7

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

1	1	1	1		1	1
0	1	1	1		1	0
1	1	0	1		1	0
1	1	1	1		1	1
1	1	1	0		1	1
1	1	1	1		1	1
0	1	1	1		1	1
1	1	1	1		1	1
1	1	1	0		1	1
x_1	x_2	x_3	x_4	x_5	x_6	x_7

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

For instance:

Not closed!

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

x_1	x_2	x_3	x_4	x_5	x_6	x_7
1	1	1	1		1	1
0	1	1	1		1	0
1	1	0	1		1	0
1	1	1	1		1	1
1	1	1	0		1	1
1	1	1	1		1	1
0	1	1	1		1	1
1	1	1	1		1	1
1	1	1	0		1	1

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

Not closed!

$$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$$

1	1	1	1		1	1
0	1	1	1		1	0
1	1	0	1		1	0
1	1	1	1		1	1
1	1	1	0		1	1
1	1	1	1		1	1
0	1	1	1		1	1
1	1	1	1		1	1
1	1	1	0		1	1
x_1	x_2	x_3	x_4	x_5	x_6	x_7

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

Reduction to **gcd=1**

Picture problems and reductions

A picture problem is when:
- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

Not closed!

$$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
23	1	1	1	1		1	1
19	0	1	1	1		1	0
17	1	1	0	1		1	0
13	1	1	1	1		1	1
11	1	1	1	0		1	1
7	1	1	1	1		1	1
5	0	1	1	1		1	1
3	1	1	1	1		1	1
2	1	1	1	0		1	1

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

Reduction to **gcd=1**

Picture problems and reductions

A picture problem is when:
- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

For instance:

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

Not closed!

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
23	1	1	1	1		1	1
19	0	1	1	1		1	0
17	1	1	0	1		1	0
13	1	1	1	1		1	1
11	1	1	1	0		1	1
7	1	1	1	1		1	1
5	0	1	1	1		1	1
3	1	1	1	1		1	1
2	1	1	1	0		1	1

2348346

Reduction to **gcd=1**

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as ‘columns’
- an input is accepted if all ‘lines’ belong to a given $L \subseteq A^d$.

For instance: Not closed!

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
23	1	1	1	1		1	1
19	0	1	1	1		1	0
17	1	1	0	1		1	0
13	1	1	1	1		1	1
11	1	1	1	0		1	1
7	1	1	1	1		1	1
5	0	1	1	1		1	1
3	1	1	1	1		1	1
2	1	1	1	0		1	1

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

Reduction to **gcd=1**

2348346 223092870 13123110 10140585. .

Picture problems and reductions

A picture problem is when:

- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

For instance:

Not closed!

$$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
23	1	1	1	1		1	1
19	0	1	1	1		1	0
17	1	1	0	1		1	0
13	1	1	1	1		1	1
11	1	1	1	0		1	1
7	1	1	1	1		1	1
5	0	1	1	1		1	1
3	1	1	1	1		1	1
2	1	1	1	0		1	1

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

Reduction to **gcd=1**

2348346 223092870 13123110 10140585. . .

gcd=1 if and only all lines have a 0!

Picture problems and reductions

A picture problem is when:
- $D = A^\omega$ understood as 'columns'
- an input is accepted if all 'lines' belong to a given $L \subseteq A^d$.

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belong to L.

For instance: Not closed!

$L = \{u \in \{0,1\}^d \text{ that contains a '0'}\}$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
23	1	1	1	1		1	1
19	0	1	1	1		1	0
17	1	1	0	1		1	0
13	1	1	1	1		1	1
11	1	1	1	0		1	1
7	1	1	1	1		1	1
5	0	1	1	1		1	1
3	1	1	1	1		1	1
2	1	1	1	0		1	1

Reduction to **gcd=1**

2348346 223092870 13123110 10140585. . .

gcd=1 if and only all lines have a 0!

This shows that the **gcd=1** problem is at least as hard as the picture problem 'all lines contain a 0'.

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

Easy direction: upward.
Assume L $\mathcal{W}$-closed.

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

$$u \in L \quad \text{iff} \quad \left(\begin{array}{c} \text{for all windows } W \\ u|_W = v|_W \text{ for some } v \in L \end{array} \right)$$

Easy direction: upward.
Assume L $\mathcal{W}$-closed.

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

$$u \in L \quad \text{iff} \quad \left(\begin{array}{c} \text{for all windows } W \\ u|_W = v|_W \text{ for some } v \in L \end{array} \right)$$

Easy direction: upward.
Assume L $\mathcal{W}$-closed.

The input is accepted
iff all lines u belong to L
iff for all lines u, and all windows W, $u|_W = v|_W$ for some $v \in L$
iff for all windows, and all lines, $u|_W \in \{v|_W \mid v \in L\}$

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

$u \in L$ iff (for all windows W $u|_W = v|_W$ for some $v \in L$)

Easy direction: upward.
Assume L $\mathcal{W}$-closed.

The input is accepted
iff all lines u belong to L
iff for all lines u, and all windows W, $u|_W = v|_W$ for some $v \in L$
iff for all windows, and all lines, $u|_W \in \{v|_W \mid v \in L\}$

$\mathcal{W}$ -definition

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

$$u \in L \quad \text{iff} \quad \left(\begin{array}{c} \text{for all windows } W \\ u|_W = v|_W \text{ for some } v \in L \end{array} \right)$$

Easy direction: upward.
Assume L $\mathcal{W}$-closed.

The input is accepted
iff all lines u belong to L
iff for all lines u, and all windows W, $u|_W = v|_W$ for some $v \in L$
iff for all windows, and all lines, $u|_W \in \{v|_W \mid v \in L\}$

$\mathcal{W}$ -definition

Difficult direction: Appeals to Hales-Jewett theorem.

Theorem

Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language L is $\mathcal{W}$-closed.

The lines that resemble a line from L through any window, belongs to L.

$$u \in L \quad \text{iff} \quad \left(\begin{array}{c} \text{for all windows } W \\ u|_W = v|_W \text{ for some } v \in L \end{array} \right)$$

Easy direction: upward.
Assume L $\mathcal{W}$-closed.

The input is accepted
iff all lines u belong to L
iff for all lines u, and all windows W, $u|_W = v|_W$ for some $v \in L$
iff for all windows, and all lines, $u|_W \in \{v|_W \mid v \in L\}$

$\mathcal{W}$ -definition

Difficult direction: Appeals to Hales-Jewett theorem.

Close to the proof in:
[Pascal Tesson. An application of the Hales-Jewett theorem to multiparty communication complexity. Extract from the PhD Thesis, 2004]

Variants

Variants

Selection gates:

A selection gate computes

$$(i, x_1, x_2, \ldots, x_k) \; \mapsto \; x_i$$

Variants

Selection gates:
A selection gate computes

$$(i, x_1, x_2, \ldots, x_k) \;\mapsto\; x_i$$

Selection gates strictly increase the expressive power for computing values in D, but not in B.

Variants

Selection gates:
A selection gate computes

$$(i, x_1, x_2, \ldots, x_k) \mapsto x_i$$

Selection gates strictly increase the expressive power for computing values in D, but not in B.

Finite variants:
As usual if the domain D is finite, but sufficiently large, similar results holds (compactness):

- Fix B to be {0,1}. For all h and all s, there exists n such that, **sum=0 mod n** over h inputs ranging over [0,n-1] is not doable by a formula of height at most h and size at most s.

Conclusion

Applications:

- these expressions are motivated for logic separation results
  - a toy example is present in the paper (metafinite structures)
  - a more difficult example is the BMA - BR separation,
  - others ?

Thank you!