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, combinational 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 $\mathbf{F}$ 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.

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

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.

E.g., $+$, $\times$, $=$, $<$, 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

\[ \bigwedge \left\{ x_1 \neq x_2, x_2 \neq x_3, \ldots, x_i \neq x_j, \ldots \right\} = 2 \]

Sum

\[ \left\lfloor \log_2(d) \right\rfloor \]

\[ \{ x_1 + \ldots + x_d \} \]
Example

All inputs are distinct

\[ \bigwedge \{ x_1, x_2, x_2, x_3, x_i \neq x_j \} \]

Sum

\[ \log_2(d) \]

Sum is null

\[ + = 0 \]
Normalization of expressions
Normalization of expressions

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

...
Normalization of expressions

All expressions of height $h$ and output in $B$ can be transformed into 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 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 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?

Proof: by contradiction;

No if \( d > 2^h \)

\[
\begin{array}{c}
\text{normalized expression} \\
\end{array}
\]
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?

Proof: by contradiction;

No if \( d > 2^h \)

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;

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 \textbf{sum} of \(d\) integers as input be computed by a combinatorial expression?

Proof: by contradiction;

No if \(d > 2^h\)

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 \textbf{sum}.

For computing problems (Boolean output)
Expressiveness questions

For representing functions (output in $D$)

Can the $\text{sum}$ of $d$ integers as input be computed by a combinatorial expression?

Proof: by contradiction;

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 $\text{sum}$.

No if $d > 2^h$

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?

Proof: by contradiction;

No if \( d > 2^h \)

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:

\[
\ldots \} h
\]
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.
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.
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 $\mathcal{W}$-definable languages for $W \in \mathcal{W}$. 
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 $W \subseteq \mathcal{P} \{1, \ldots, d\}$ be a set of windows.

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

That depend only of the inputs from $W$. 
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 $W \subseteq \mathcal{P}([1, \ldots, d])$ be a set of windows.

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

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 $\mathcal{W}$-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 $\text{sum}=0$ and $\gcd=1$ $\mathcal{W}$-definable for $\mathcal{W}$ non-trivial (i.e., not containing the full window)?
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 $L \subseteq A^d$. 
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:
\[
L = \{ u \in \{0, 1\}^d \text{ that contains a '0'} \}
\]
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:

\[
L = \{ u \in \{0, 1\}^d \text{ that contains a } '0' \}
\]
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:
$L = \{ u \in \{0, 1\}^d \text{ that contains a '0'} \}$
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:
\[
L = \{ u \in \{0, 1\}^d \text{ that contains a } '0' \}
\]
### 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:
$L = \{u \in \{0, 1\}^d \text{ that contains a ’0’}\}$

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**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 $L$. 
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: \( L = \{ u \in \{0, 1\}^d \text{ that contains a ’0’} \} \)

Not closed!

The lines that resemble a line from \( L \) through any window, belong to \( L \).
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: $L = \{u \in \{0, 1\}^d$ that contains a ’0’$\}$

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

\[
\begin{array}{ccccccc}
23 & 1 & 1 & 1 & 1 & \text{[Highlighted]} & 1 & 1 \\
19 & 0 & 1 & 1 & 1 & & 1 & 0 \\
17 & 1 & 1 & 0 & 1 & & 1 & 0 \\
13 & 1 & 1 & 1 & 1 & \text{[Highlighted]} & 1 & 1 \\
11 & 1 & 1 & 1 & 0 & & 1 & 1 \\
7  & 1 & 1 & 1 & 1 & \text{[Highlighted]} & 1 & 1 \\
5  & 0 & 1 & 1 & 1 & & 1 & 1 \\
3  & 1 & 1 & 1 & 1 & \text{[Highlighted]} & 1 & 1 \\
2  & 1 & 1 & 1 & 0 & & 1 & 1 \\
\end{array}
\]

$x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7$

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

\begin{align*}
23 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
19 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
17 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
13 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
11 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
7  & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
5  & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
3  & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
2  & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
\end{align*}

Not closed!

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 $\text{gcd}=1$
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:
\[
L = \{ u \in \{0, 1\}^d \text{ that contains a ’0’} \}
\]

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 \( \text{gcd}=1 \)

\[
\begin{array}{cccccccc}
23 & 1 & 1 & 1 & 1 & & & 1 & 1 \\
19 & 0 & 1 & 1 & 1 & & & 1 & 0 \\
17 & 1 & 1 & 0 & 1 & & & 1 & 0 \\
13 & 1 & 1 & 1 & 1 & & & 1 & 1 \\
11 & 1 & 1 & 1 & 0 & & & 1 & 1 \\
11 & 1 & 1 & 1 & 1 & & & 1 & 1 \\
5 & 0 & 1 & 1 & 1 & & & 1 & 1 \\
3 & 1 & 1 & 1 & 1 & & & 1 & 1 \\
2 & 1 & 1 & 1 & 0 & & & 1 & 1 \\
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7
\end{array}
\]
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: $L = \{u \in \{0, 1\}^d \text{ that contains a } '0'\}$

\[\begin{array}{cccccc}
23 & 1 & 1 & 1 & 1 & 1 \\
19 & 0 & 1 & 1 & 1 & 0 \\
17 & 1 & 1 & 0 & 1 & 1 \\
13 & 1 & 1 & 1 & 1 & 1 \\
11 & 1 & 1 & 1 & 0 & 1 \\
\text{7} & 1 & 1 & 1 & 1 & 1 \\
5 & 0 & 1 & 1 & 1 & 1 \\
3 & 1 & 1 & 1 & 1 & 1 \\
2 & 1 & 1 & 1 & 0 & 1 \\
\end{array}\]

Not closed!

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 $\gcd=1$

$\gcd=1$ 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 $L \subseteq A^d$.

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

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 $\gcd=1$

$\gcd=1$ 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: A picture problem is $\mathcal{W}$-definable if and only if the line language $L$ is $\mathcal{W}$-closed.
**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: A picture problem is $\mathcal{W}$-definable if and only if the line language $L$ is $\mathcal{W}$-closed.

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

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

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

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

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

$$ u \in L \iff \left( \text{for all windows } W, u|_W = v|_W \text{ for some } v \in L \right) $$
Theorem: A picture problem is $\mathcal{W}$-definable if and only if the line language $L$ is $\mathcal{W}$-closed.

Easy direction: upward.
Assume $L$ is $\mathcal{W}$-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, $u|_W \in \{ v|_W \mid v \in L \}$

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

\[ u \in L \iff \left( u|_W = v|_W \text{ for some } v \in L \right) \]
Theorem

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

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, $u|_W \in \{v|_W \mid v \in L\}$

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

\[ u \in L \iff \left( \text{for all windows } W \right) \]
\[ u|_W = v|_W \text{ for some } v \in L \]
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 $L$.

$$u \in L \iff \left( \text{for all windows } W, \ u|_W = v|_W \text{ for some } v \in L \right)$$

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.

**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, $u|_W \in \{v|_W \mid v \in L\}$

**Difficult direction**: Appeals to Hales-Jewett theorem.
Close to the proof in:
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 \(s\).
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!