Proof Nets and Boolean Circuits

Kazushige Terui

terui@nii.ac.jp

National Institute of Informatics, Tokyo
**Motivation (1)**

 proofs-as-programs (curry-howard) correspondence:

\[
\begin{align*}
\text{Proofs} & \quad = \quad \text{Programs} \\
\text{Cut-Elimination} & \quad = \quad \text{Computation} \\
& \quad \quad \text{(Normalization)}
\end{align*}
\]

usually interested in **infinite, uniform, sequential** computation such as **functional programs**.

Can be extended to **finite, nonuniform, parallel** computation such as **boolean circuits**?
Motivation (2)

Our goal:

Proofs = Circuits
Cut-Elimination = Evaluation

What system of Proofs?

Cut-elimination in Classical/Intuitionistic Logics:
Non-elementary time. Too much!
Motivation (3)

Linear Logic (Girard 87): a decomposition of Classical/Intuitionistic Logics:

- **Multiplicatives**
- **Additives**
- **Exponentials**

MLL: Classical Logic without weakening nor contraction.

Has a nice parallel syntax: Proof Nets (Girard 87).

Quadratic time cut-elimination procedure.
Motivation (4)

Our specific goal: correspondence between MLL proof nets and circuits.

(Mairson & Terui 2003) Encoding of circuits by linear size MLL proof nets \(\Rightarrow\) P-completeness of cut-elimination in MLL.

“Proof nets can represent all finite functions as SIZE-efficient as circuits.”

What about the DEPTH-efficiency? What about the converse?
Parallel computation

- **Effective parallelization:** Achieve a dramatic speed-up by use of a reasonable number of processors.

- **Addition:** School method \((O(n)\) time)  
  \(\implies\) Parallel algorithm: \((\text{constant time})\)

- In boolean circuits,
  
  \[
  \text{time} = \text{depth} \\
  \text{num of processors} = \text{size (num of gates)}
  \]

- **Fundamental question:** Are all feasible algorithms effectively parallelizable?

- **\(NC = P\) problem:** Are all problems in \(P\) solvable by polynomial size poly-log depth circuits?
Outline

- Unbounded fan-in boolean circuits.
- Proof nets for MLLu and parallel cut-elimination procedure.
- Simulation of circuits by proof nets
- Simulation of proof nets by circuits
- Proof net complexity classes
Boolean circuits (1)

An unbounded fan-in circuit with $n$ inputs (and 1 output): a directed acyclic graph made of

- input nodes $x_1, \ldots, x_n$

- boolean gates $\neg$, $\land$, $\lor$ (and possibly more).

(there is a distinguished node for output.)

$\land$ and $\lor$ may have an arbitrary number of inputs.

size = the number of gates

depth = the length of the longest path
Boolean circuits (2)

A circuit $C$ accepts $w = i_1 \cdots i_n \in \{0, 1\}^n$ if

$C[x_1 := i_1, \ldots, x_n := i_n]$ evaluates to 1.

$C$ accepts $X \subseteq \{0, 1\}^n$ if $C$ accepts $w \iff w \in X$. 
Formulas of MLLu

Formulas:

$\alpha, \alpha^\perp$  
Literals

$\otimes^n(A_1, \ldots, A_n), \ n \geq 2$  
n-ary Conjunction

$\oslash^n(A_1, \ldots, A_n), \ n \geq 2$  
n-ary Disjunction

Negation: defined by

$$(\otimes^n(A_1, \ldots, A_n))^\perp \equiv \oslash^n(A_n^\perp, \ldots, A_1^\perp)$$

$$(\oslash^n(A_1, \ldots, A_n))^\perp \equiv \otimes^n(A_n^\perp, \ldots, A_1^\perp)$$

Notation:

$$A \otimes B \equiv \otimes^2(A, B) \quad A \oslash B \equiv \oslash^2(A, B) \quad A \oslash B \equiv A^\perp \oslash B$$
Sequent calculus for MLLu

- **Sequents**: \( \vdash \Gamma \), where \( \Gamma \) is a multiset of formulas.

- **Inference Rules**:
  
  \[
  \begin{align*}
  \vdash A, \bot & \quad (Axiom) \\
  \vdash \Gamma, C & \quad \vdash \Delta, C^\bot \quad (Cut) \\
  \vdash \Gamma, A_1, \ldots, A_n & \quad \vdash \Gamma_1, \ldots, \Gamma_n, \otimes^n (A_1, \ldots, A_n) \quad \otimes^n \\
  \vdash \Gamma, \otimes^n (A_1, \ldots, A_n) & \quad \vdash \Gamma, \otimes^n (A_1, \ldots, A_n) \quad \otimes^n
  \end{align*}
  \]

- Exchange is implicit. No weakening, no contraction.
Proof nets for MLLu

Links:

Each link has several ports. Principal port(s) numbered 0.

Cut: an edge connecting two principal ports.

Proof nets are obtained from sequent proofs by extracting their structures (forgetting about formulas).
Example: Negation

\[ \vdash \alpha \perp \alpha, \alpha \otimes \alpha \quad \vdash \alpha \perp, \alpha \quad \vdash \alpha \perp, \alpha \]

\[ \vdash \otimes^3 (\alpha \perp \perp, \alpha, \alpha), \alpha \perp, \alpha \perp, \alpha \otimes \alpha \]

\[ \vdash \otimes^3 (\alpha \perp \perp, \alpha, \alpha), \otimes^3 (\alpha \perp, \alpha \perp, \alpha \otimes \alpha) \]
Example: Booleans

\[ B \equiv \alpha \rightarrow \alpha \rightarrow \alpha \otimes \alpha \equiv \mathcal{G}^3(\alpha^\perp, \alpha^\perp, \alpha \otimes \alpha) \]

\[ \begin{array}{c}
\vdash \alpha^\perp, \alpha \quad \vdash \alpha^\perp, \alpha \\
\vdash \alpha^\perp, \alpha \otimes \alpha \\
\vdash \mathcal{G}^3(\alpha^\perp, \alpha^\perp, \alpha \otimes \alpha)
\end{array} \implies \begin{array}{c}
\mathcal{G}^3(\alpha^\perp, \alpha^\perp, \alpha ^\otimes \alpha) \\
\equiv b_1
\end{array} \]

\[ \begin{array}{c}
\vdash \alpha^\perp, \alpha \quad \vdash \alpha^\perp, \alpha \\
\vdash \alpha^\perp, \alpha \otimes \alpha \\
\vdash \mathcal{G}^3(\alpha^\perp, \alpha^\perp, \alpha \otimes \alpha)
\end{array} \implies \begin{array}{c}
\mathcal{G}^3(\alpha^\perp, \alpha^\perp, \alpha \otimes \alpha) \\
\equiv b_0
\end{array} \]
Reduction rules

Axiom reduction:

Multiplicative reduction:
Example: Computing Negation of True
Example: Computing Negation of True
Example: Computing Negation of True
Sequential cut elimination

Size $|P|$: number of links.
Sequential cut elimination

- **Size** $|P|$: number of links.
- **Theorem (Girard 87):** Every proof net $P$ reduces to a cut-free proof net in $|P|$ steps.
Sequential cut elimination

- **Size** $|P|$: number of links.

- **Theorem (Girard 87):** Every proof net $P$ reduces to a cut-free proof net in $|P|$ steps.

- **Too slow!**
Parallel cut elimination (1)

Applying two axiom reductions in parallel may conflict:

![Diagram 1]

Global axiom reduction:

![Diagram 2]
Parallel cut elimination (2)

- Parallel multiplicative reduction:

- Parallel axiom reduction:

\[ P_1 \Rightarrow P_2 \text{ if } P_2 \text{ is obtained from } P_1 \text{ either by parallel m-reduction or by parallel a-reduction.} \]
Parallel cut elimination (3)

What controls the runtime of parallel cut-elimination?

Depth $d(P)$: Maximal depth of cut formulas in it.  
($P$ is assumed to be typed by principal types (most general types))

Depth of formulas:

$$
\begin{align*}
    d(\alpha) &= d(\alpha^\perp) &= 1 \\
    d(\otimes^n(A_1,\ldots,A_n)) &= d(\otimes^n(A_1,\ldots,A_n)) \\
    &= \max(d(A_1),\ldots,d(A_n)) + 1
\end{align*}
$$
**Parallel cut elimination (4)**

**Theorem:** Every proof net \( P \) reduces to a normal form in \( 2 \cdot d(P) \) parallel reduction steps.

**Proof:** By applying parallel a-reduction, every cut becomes multiplicative. By applying parallel m-reduction, the depth decreases by 1.

\( \square \)
Limitation on parallelization

$P_n$:

$|P_n|$ and $d(P_n)$ are linear in $n$.

Parallel cut-elimination takes almost as long time as sequential cut-elimination.
Representing circuits by proof nets

**Idea:** Represent

- Circuits by Proof nets
- Boolean values by Proof nets \((b_1, b_0)\)
- Assignment by Cuts
- Evaluation by Cut elimination
**Representation of Parity**

- **Parity**: \( \text{PARITY}^n(x_1, \ldots, x_n) = 1 \iff \text{the sum of } x_1, \ldots, x_n \text{ is odd.} \)
- **CANNOT** be represented by (poly-size) circuits of constant depth.

The conclusion is \( \vdash B^\perp, \ldots, B^\perp, B \)

\[ n \text{ times} \]
Correctness of parity

There are exactly 2 crossings in the drawing.

Multiplicative reduction does not change the num of crossings.

Axiom reduction preserves the parity of the num of crossings:

Therefore, the parity is 0 after cut-elimination.
Expressivity of flat proof nets

Would break down if there were a self-crossing:

But self-crossing can be avoided.

As far as proof nets of conclusion \( \vdash B^\perp, \ldots, B^\perp, B \) are concerned, all what matters is the parity of the num of crossings.

Theorem: Every proof net with the above conclusion represents either PARITY\(^n\) or \(\neg\text{PARITY}\(^n\).\)
**Boolean proof nets**

- **Boolean proof net**: a proof net with conclusion

\[ \vdash B[A_1/\alpha] \perp, \ldots, B[A_n/\alpha] \perp, \otimes^m (B, \vec{G}) \]

for some \( A_1, \ldots, A_n \) and \( \vec{G} \) (garbage).

- Given \( w = i_1 \cdots i_n \in \{0, 1\}^n \), define \( P(w) \) to be:

\[ P \text{ accepts } w \in \{0, 1\}^n \text{ if } P(w) \text{ reduces to } \otimes (b_1, \vec{P}_G) \]

for some proof nets \( \vec{P}_G \) with conclusions \( \vec{G} \).
Conditional and Disjunction

\[
\text{if } q \text{ then } P_1 \text{ else } P_2
\]

Disjunction: \( \text{or}(p, q) \equiv \text{if } p \text{ then } b_1 \text{ else } q \)

Disjunctions of arbitrary arity can be represented by proof nets of constant depth.
Composition

Composition:

The depth may increase:

\[ \vdash \Gamma, B \quad \vdash P_2 \]

\[ \vdash \Gamma[A/\alpha], B[A/\alpha] \quad \vdash B \perp[A/\alpha], \Delta, B \]

\[ \vdash \Gamma[A/\alpha], \Delta, B \]
Proof Net for Majority (1)

\( \text{MAJ}^n(x_1 \cdots x_n) = 1 \) if at least half of \( x_i \)'s are 1.

Let \( \text{id}, \text{sh} : \{0, 1\}^{n+1} \rightarrow \{0, 1\}^{n+1} \) be:

\[
\begin{align*}
\text{id}(i_1 \cdots i_{n+1}) & = i_1 \cdots i_{n+1} \\
\text{sh}(i_1 \cdots i_{n+1}) & = i_2 \cdots i_{n+1}i_1
\end{align*}
\]

\( F \): higher order functional

\[
\begin{align*}
F(0) & = \text{id} \\
F(1) & = \text{sh}
\end{align*}
\]
**Proof Net for Majority (2)**

\[ \text{MAJ}^n \text{ given by} \]

\[
\text{MAJ}^n(x_1 \cdots x_n) = \text{FstBit}(F(x_1) \circ \cdots \circ F(x_n))(0 \cdots 01 \cdots 1)
\]

\[n/2 \quad n/2+1\]

**Example:**

\[\text{MAJ}^6(101101) = \text{FstBit}(F(1)F(0)F(1)F(1)F(0)F(1)(0001111)) \]
\[= \text{FstBit}(sh \circ id \circ sh \circ id \circ sh(0001111)) \]
\[= \text{FstBit}(1110001) \]
\[= 1\]

Represented by proof nets of **constant depth.** (CANNOT be represented by (poly-size) circuits of constant depth.)
**St-connectivity**

- **Input**: an undirected graph of degree 2 (i.e., nonbranching graph) with vertices \( \{1, \ldots, n\} \). Assume it is coded by a \( n \times n \) boolean matrix.

- **Output**: is 1 if vertices 1 and \( n \) are connected.
St-connectivity

- **Input:** an undirected graph of degree 2 (i.e., nonbranching graph) with vertices \( \{1, \ldots, n\} \). Assume it is coded by a \( n \times n \) boolean matrix.

- **Output:** is 1 if vertices 1 and \( n \) are connected.

Can simulate \( \text{MAJ} \) in constant depth.

Represented by proof nets of **constant depth**.
**Theorem:** For every circuit $C$ of size $s$ and depth $d$ (possibly equipped with $st\text{CONN}_2$), there is a boolean proof net $P_C$ of size $O(s^5)$ and depth $O(d)$ which accepts the same set as $C$. 
Theorem: For every circuit $C$ of size $s$ and depth $d$ (possibly equipped with $st\text{CONN}_2$), there is a boolean proof net $P_C$ of size $O(s^5)$ and depth $O(d)$ which accepts the same set as $C$. 

\[
\begin{array}{c}
\text{Circuit } C \\
\text{size } s, \text{ depth } d
\end{array} \quad \Longrightarrow \quad 
\begin{array}{c}
\text{Proof Net } P_C \\
\text{size } O(s^5), \text{ depth } O(d)
\end{array}
\]
**From Circuits to Proof nets**

**Theorem:** For every circuit $C$ of size $s$ and depth $d$ (possibly equipped with $stCONN_2$), there is a boolean proof net $P_C$ of size $O(s^5)$ and depth $O(d)$ which accepts the same set as $C$.

\[
\begin{array}{c}
\text{Circuit } C \\
\text{size } s, \text{ depth } d
\end{array}
\quad \Rightarrow \quad
\begin{array}{c}
\text{Proof Net } P_C \\
\text{size } O(s^5), \text{ depth } O(d)
\end{array}
\]

**Proof:** $\neg$, $\wedge^n$, $\vee^n$, $stCONN_2^n$ represented by a proof net of size $O(n^4)$ and of constant depth. Composition increases the depth linearly.
Representing proof nets by circuits

Idea: Represent

A proof net $P$ by a set of boolean values

One-step reduction by constant depth circuit

$2 \cdot d(P)$ reduction steps by $O(d)$-depth circuit
Coding proof nets by boolean values

- \( P \): a proof net with links \( \subseteq L \).

- \( Conf(P) \): consists of the following boolean values:
  
  For \( p, q \in L \),

  \[
  \begin{align*}
  alive(p) & \iff p \text{ is a link of } P \\
  sort(p, s) & \iff \text{link } p \text{ is of sort } s \in \{ \otimes, \otimes, \bullet \} \\
  edge(p, i, q, j) & \iff \text{there is an edge between port } i \text{ of link } p \text{ and port } j \text{ of link } q
  \end{align*}
  \]

- Build a circuit \( C \) s.t.

  if \( P_1 \Rightarrow P_2 \) then \( C \) computes \( Conf(P_2) \) from \( Conf(P_1) \).
Multiplicative Reduction

Easily implemented by a constant depth circuit: E.g,

\[ edge'(p_i, 0, q_i, 0) = edge(p_i, 0, q_i, 0) \lor \bigvee_{p,q \in L} \left( edge(p, 0, q, 0) \land \bigvee_j edge(p_i, 0, p, j) \land edge(q_i, 0, q, j) \right) \]
Naive attempt to build a constant depth circuit leads to exponential size.

Use \textit{stCONN}_2 gates.

Apply to the axiom-cut subgraph of $P$, that is of degree 2.
Theorem: For every boolean proof net $P$ of size $s$ and depth $d$, there is a circuit $C$ (with $st\text{CONN}_2$ gates) of size $O(s^4)$ and depth $O(d)$ which accepts the same set as $P$. 
Theorem: For every boolean proof net $P$ of size $s$ and depth $d$, there is a circuit $C$ (with $st\text{CONN}_2$ gates) of size $O(s^4)$ and depth $O(d)$ which accepts the same set as $P$. 

Proof Net $P$
size $s$, depth $d$ $\implies$ Circuit $C_P$
size $O(s^4)$, depth $O(d)$
Theorem: For every boolean proof net $P$ of size $s$ and depth $d$, there is a circuit $C$ (with $st\text{CONN}_2$ gates) of size $O(s^4)$ and depth $O(d)$ which accepts the same set as $P$.

Proof: Cut-elimination of $P$ requires of $2 \cdot d$ steps. Each step simulated by a constant depth circuit (with $st\text{CONN}_2$ gates). Initialization and acceptance checking also by constant depth circuits.
Proof net complexity classes (1)

For $X \subseteq \{0, 1\}^*$, 

$$X \in AC^i(F) \iff X \text{ is accepted by a polynomial size } \log^i\text{-depth family of unbounded fan-in circuits with additional gates from } F.$$ 

$$X \in APN^i \iff X \text{ is accepted by a polynomial size } \log^i\text{-depth family of proof nets}.$$ 

**Theorem:** For every $i \geq 0$,

$$APN^i = AC^i(st\text{CONN}_2).$$
Proof net complexity classes (2)

\( APN = \bigcup_i APN^i. \)

Theorem \( APN = NC. \)

Proof: \( AC^i \subseteq AC^i(st\text{CONN}_2) = APN^i \subseteq AC^{i+1} \) and \( NC = \bigcup_i AC^i. \)

\( P/poly \): nonuniform version of \( P. \)

Theorem \( P/poly = \) the class of languages \( X \subseteq \{0,1\}^* \)
accepted by polynomial size families of proof nets.

Note \( AC^0(st\text{CONN}_2) = L/poly \) (nonuniform logspace). Hence,

\[
AC^0 \subsetneq NC^1 \subseteq L/poly = APN^0 \subseteq AC^1
\]
Conclusion

Extended the Proofs-as-Programs paradigm to a finite, nonuniform, parallel setting.

Characterized proof net complexity by circuit complexity.

Proof nets represent "higher order gates."

Future Work

\( NC^{i+1} \subseteq APN^i \)?

Comparison with other approaches to proofs-circuits correspondence (Propositional Proof Systems and Bounded Arithmetic).

Study the bounded fan-in case.

Implicit parallel complexity.