Redesigning Logical Syntax with a Bit of Topology

Alessio Guglielmi

University of Bath

Joint work with
Paola Bruscoli, Tom Gundersen, Michel Parigot and Lutz Straßburger

31 May 2011

This talk is available at `http://cs.bath.ac.uk/ag/t/RDLS.pdf`
It requires Acrobat 9 or later

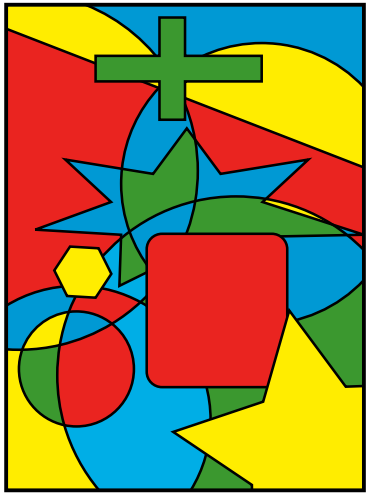
The Dream

The Dream

- No syntax, no symbols, no words.
- An alien could understand this proof.
- Is something like this possible for every proof?

The Reality

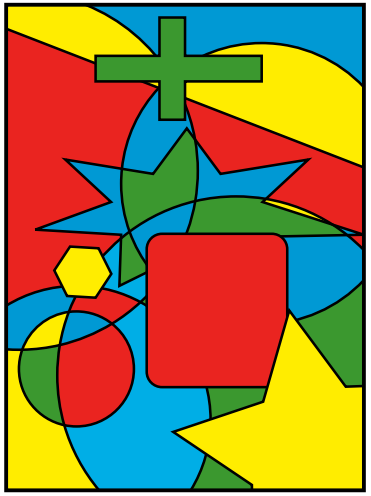
```
Lemma sumt_ctree_pick_rev : forall t t', sumt (ctree_pick_rev t t') = Color0.
Proof.
move=> t' t; rewrite /ctree_pick_rev; set cs0 : colseq := seq0.
have: Color0 +c sumt cs0 = Color0 by done.
elim: t cs0 {1 3}Color0 => [t1 Ht1 t2 Ht2 t3 Ht3|lf _|] et e //.
  move=> Het /=; set cprr := ctree_pick_rev_rec.
  case Det1: (cprr _ _ _ t1) => [|e1 et1].
    case Det2: (cprr _ _ _ t2) => [|e2 et2].
      by apply: Ht3; rewrite [Color3]lock /= -addcA addc_inv.
    by rewrite -Det2; apply: Ht2; rewrite [Color2]lock /= -addcA addc_inv.
  by rewrite -Det1; apply: Ht1; rewrite [Color1]lock /= -addcA addc_inv.
by move=> Het /=; case (ctree_mem t' (etrace (belast e et))).
Qed.
```



- 100s of similar pieces in the four colour theorem proof in Coq.
- Syntactic object with a lot of arbitrary choice: bureaucracy.

The Reality

```
Lemma sumt_ctree_pick_rev : forall t t', sumt (ctree_pick_rev t t') = Color0.
Proof.
move=> t' t; rewrite /ctree_pick_rev; set cs0 : colseq := seq0.
have: Color0 +c sumt cs0 = Color0 by done.
elim: t cs0 {1 3}Color0 => [t1 Ht1 t2 Ht2 t3 Ht3|lf _|] et e //.
  move=> Het /=; set cprr := ctree_pick_rev_rec.
  case Det1: (cprr _ _ _ t1) => [|e1 et1].
    case Det2: (cprr _ _ _ t2) => [|e2 et2].
      by apply: Ht3; rewrite [Color3]lock /= -addcA addc_inv.
    by rewrite -Det2; apply: Ht2; rewrite [Color2]lock /= -addcA addc_inv.
  by rewrite -Det1; apply: Ht1; rewrite [Color1]lock /= -addcA addc_inv.
by move=> Het /=; case (ctree_mem t' (etrace (belast e et))).
Qed.
```



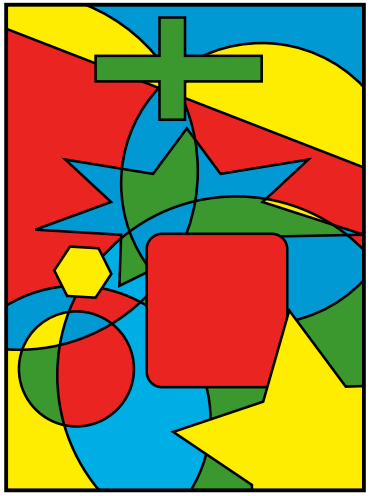
- 100s of similar pieces in the four colour theorem proof in Coq.
- Syntactic object with a lot of arbitrary choice: bureaucracy.

Problems:

- How do we determine whether two proofs are 'the same'?
- Can we free proofs from the idiosyncrasies of language?

The Reality

```
Lemma sumt_ctree_pick_rev : forall t t', sumt (ctree_pick_rev t t') = Color0.
Proof.
move=> t' t; rewrite /ctree_pick_rev; set cs0 : colseq := seq0.
have: Color0 +c sumt cs0 = Color0 by done.
elim: t cs0 {1 3}Color0 => [t1 Ht1 t2 Ht2 t3 Ht3|lf _|] et e //.
  move=> Het /=; set cprr := ctree_pick_rev_rec.
  case Det1: (cprr _ _ _ t1) => [|e1 et1].
    case Det2: (cprr _ _ _ t2) => [|e2 et2].
      by apply: Ht3; rewrite [Color3]lock /= -addcA addc_inv.
    by rewrite -Det2; apply: Ht2; rewrite [Color2]lock /= -addcA addc_inv.
  by rewrite -Det1; apply: Ht1; rewrite [Color1]lock /= -addcA addc_inv.
by move=> Het /=; case (ctree_mem t' (etrace (belast e et))).
Qed.
```



- 100s of similar pieces in the four colour theorem proof in Coq.
- Syntactic object with a lot of arbitrary choice: bureaucracy.

Problems:

- How do we determine whether two proofs are 'the same'?
- Can we free proofs from the idiosyncrasies of language?

Solving these is necessary for the universal mathematics database.

Outline of the Talk

- Strategy
- Proof Complexity and the Oddness of the Cut
- Open Deduction (Deep Inference)
  - Propositional Logic and System SKS
  - Examples
- Atomic Flows
  - Examples
  - Flow Reductions
- Normalisation
  - Cut Elimination: Experiments
  - Streamlining: Generalised Cut Elimination
  - The Path Breaker
  - Quasipolynomial Cut Elimination
  - Overview
- Conjecture
- Conclusion

Strategy:
We conserve the existing proof theory properties ...

Gentzen's major breakthrough (1930s):

- proofs can be analytic, *i.e.*, built in finitary ways,
- by time expensive algorithms,
- that nonetheless allow us to control and analyse them.

$$
\dfrac{\dfrac{\dfrac{\dfrac{a \vdash a}{a \vdash a \vee (a \supset \bot)}\,\vee_{RL} \qquad a, \bot \vdash \bot}{a, (a \vee (a \supset \bot)) \supset \bot \vdash \bot}\,\supset_L \qquad \dfrac{\dfrac{\dfrac{\dfrac{a \vdash a \qquad \bot, a \vdash \bot}{a \supset \bot, a \vdash \bot}\,\supset_L}{a \supset \bot \vdash a \supset \bot}\,\supset_R}{a \supset \bot \vdash a \vee (a \supset \bot)}\,\vee_{RR} \qquad a \supset \bot, \bot \vdash \bot}{a \supset \bot, (a \vee (a \supset \bot)) \supset \bot \vdash \bot}\,\supset_L}{a \vee (a \supset \bot), (a \vee (a \supset \bot)) \supset \bot \vdash \bot}\,\vee_L}{a \vee (a \supset \bot) \vdash ((a \vee (a \supset \bot)) \supset \bot) \supset \bot}\,\supset_R
$$

Strategy:
We conserve the existing proof theory properties ...

Gentzen's major breakthrough (1930s):

- proofs can be analytic, *i.e.*, built in finitary ways,
- by time expensive algorithms,
- that nonetheless allow us to control and analyse them.

$$
\supset_R \dfrac{\vee_L \dfrac{\vee_{RL} \dfrac{a \vdash a}{a \vdash a \vee (a \supset \bot)} \quad a, \bot \vdash \bot}{a, (a \vee (a \supset \bot)) \supset \bot \vdash \bot} \quad \supset_L \dfrac{\vee_{RR} \dfrac{\supset_R \dfrac{\supset_L \dfrac{a \vdash a \quad \bot, a \vdash \bot}{a \supset \bot, a \vdash \bot}}{a \supset \bot \vdash a \supset \bot}}{a \supset \bot \vdash a \vee (a \supset \bot)} \quad a \supset \bot, \bot \vdash \bot}{a \supset \bot, (a \vee (a \supset \bot)) \supset \bot \vdash \bot}}{a \vee (a \supset \bot), (a \vee (a \supset \bot)) \supset \bot \vdash \bot}}{a \vee (a \supset \bot) \vdash ((a \vee (a \supset \bot)) \supset \bot) \supset \bot}
$$

But Gentzen

- only knew classical logic, which is poor for algorithms;
- only wanted finiteness, while we want more: efficiency;
- had no idea of proof complexity.

Strategy:

... while we keep proof complexity under control, ...

Proof complexity = proof size (for propositional logic).

Strategy:

... while we keep proof complexity under control, ...

Proof complexity = proof size (for propositional logic).

Proof system = algorithm that checks proofs in polynomial time.

Strategy:

... while we keep proof complexity under control, ...

Proof complexity = proof size (for propositional logic).

Proof system = algorithm that checks proofs in polynomial time.

Theorem [Cook and Reckhow, 1974]:

there exists a proof system yielding 'short' proofs for every tautology

$$\leftrightarrow$$

$$\mathrm{coNP} = \mathrm{NP}$$

where

'short' = verifiable in polynomial time on the size of the tautology

Strategy:

... while we keep proof complexity under control, ...

Proof complexity = proof size (for propositional logic).

Proof system = algorithm that checks proofs in polynomial time.

Theorem [Cook and Reckhow, 1974]:

there exists a proof system yielding 'short' proofs for every tautology

$$\leftrightarrow$$

$$\mathrm{coNP} = \mathrm{NP}$$

where

'short' = verifiable in polynomial time on the size of the tautology

So:

- we want to keep proof size low (and possibly making it lower),
- but not too low (otherwise we probably don't have proof systems).

Strategy:
... and we remove bureaucracy.

Idea: Let's use the smallest conceivable bricks to build proofs.
(Gentzen's material is too rigid!)

Strategy:
... and we remove bureaucracy.

Idea: Let's use the smallest conceivable bricks to build proofs. (Gentzen's material is too rigid!)

We want proof systems whose inference steps are verifiable in constant time.

Example ('atomic cocontraction'):

$$\mathsf{m}\,\dfrac{\dfrac{\dfrac{a}{a \wedge a}\vee \quad \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}}{}$$

Strategy:
... and we remove bureaucracy.

Idea: Let's use the smallest conceivable bricks to build proofs. (Gentzen's material is too rigid!)

We want proof systems whose inference steps are verifiable in constant time.

Example ('atomic cocontraction'):

$$\mathsf{m}\,\dfrac{\dfrac{\dfrac{a}{a \wedge a}\vee \quad \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}}{}$$

We call this property locality.

Proof Systems for Proof Complexity

- Proof system = algorithm checking proofs in polytime.

Proof Systems for Proof Complexity

- Proof system = algorithm checking proofs in polytime.
- Example, a Frege system:
  - Axioms:
    $$A \supset (B \supset A),$$
    $$(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C)),$$
    $$(\neg B \supset \neg A) \supset ((\neg B \supset A) \supset B),$$

    and rules, often just modus ponens, or cut:
    $$\frac{A \quad A \supset B}{B}$$
  - $$\dfrac{\dfrac{}{a \supset (a \supset a)} \quad \dfrac{\dfrac{}{a \supset ((a \supset a) \supset a)} \quad \dfrac{}{(a \supset ((a \supset a) \supset a)) \supset ((a \supset (a \supset a)) \supset (a \supset a))}}{(a \supset (a \supset a)) \supset (a \supset a)}}{a \supset a}$$

Proof Systems for Proof Complexity

- Proof system = algorithm checking proofs in polytime.
- Example, a Frege system:
  - Axioms:
    $$A \supset (B \supset A),$$
    $$(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C)),$$
    $$(\neg B \supset \neg A) \supset ((\neg B \supset A) \supset B),$$

    and rules, often just modus ponens, or cut:
    $$\frac{A \quad A \supset B}{B}$$
  - $$\dfrac{\dfrac{}{a \supset (a \supset a)} \quad \dfrac{\dfrac{}{a \supset ((a \supset a) \supset a)} \quad \dfrac{}{(a \supset ((a \supset a) \supset a)) \supset ((a \supset (a \supset a)) \supset (a \supset a))}}{(a \supset (a \supset a)) \supset (a \supset a)}}{a \supset a}$$
- Robustness Theorem [Cook and Reckhow, 1974]:
  All Frege systems are polynomially equivalent.

Proof Systems for Proof Complexity

- Proof system = algorithm checking proofs in polytime.
- Example, a Frege system:
  - Axioms:
    $$A \supset (B \supset A),$$
    $$(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C)),$$
    $$(\neg B \supset \neg A) \supset ((\neg B \supset A) \supset B),$$

    and rules, often just modus ponens, or cut:
    $$\frac{A \quad A \supset B}{B}$$
  - $$\dfrac{\dfrac{}{a \supset (a \supset a)} \quad \dfrac{\dfrac{}{a \supset ((a \supset a) \supset a)} \quad \dfrac{}{(a \supset ((a \supset a) \supset a)) \supset ((a \supset (a \supset a)) \supset (a \supset a))}}{(a \supset (a \supset a)) \supset (a \supset a)}}{a \supset a}$$
- Robustness Theorem [Cook and Reckhow, 1974]:
  All Frege systems are polynomially equivalent.
- Due to implicational completeness: if $A \supset B$ then A proves B.

Proof Systems for Proof Complexity

- Proof system = algorithm checking proofs in polytime.
- Example, a Frege system:
  - Axioms:
    $$A \supset (B \supset A),$$
    $$(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C)),$$
    $$(\neg B \supset \neg A) \supset ((\neg B \supset A) \supset B),$$

    and rules, often just modus ponens, or cut: $\dfrac{A \quad A \supset B}{B}$
  - $$\dfrac{\dfrac{}{a \supset (a \supset a)} \quad \dfrac{\dfrac{}{a \supset ((a \supset a) \supset a)} \quad \dfrac{}{(a \supset ((a \supset a) \supset a)) \supset ((a \supset (a \supset a)) \supset (a \supset a))}}{(a \supset (a \supset a)) \supset (a \supset a)}}{a \supset a}$$
- Robustness Theorem [Cook and Reckhow, 1974]:
  All Frege systems are polynomially equivalent.
- Due to implicational completeness: if $A \supset B$ then A proves B.

We envy the syntax-independence of proof complexity!

Compressing Proofs

How can we make proofs smaller?

Compressing Proofs

How can we make proofs smaller?

Known mechanisms:

1. Higher orders (*e.g*, second order propositional for propositional formulae).
2. Tseitin extension: $p \leftrightarrow A$ (where p is a fresh atom).
3. Substitution: $\mathsf{sub}\,\dfrac{A}{A\sigma}$.
4. Use the same sub-proof many times: dag-ness, or cocontraction.
5. Use the same sub-proof many times: cut rule.

Compressing Proofs

How can we make proofs smaller?

Known mechanisms:

1. Higher orders (*e.g*, second order propositional for propositional formulae).
2. Tseitin extension: $p \leftrightarrow A$ (where p is a fresh atom).
3. Substitution: $\mathsf{sub}\,\dfrac{A}{A\sigma}$. Equivalent to (2).
4. Use the same sub-proof many times: dag-ness, or cocontraction.
5. Use the same sub-proof many times: cut rule.

Compressing Proofs

How can we make proofs smaller?

Known mechanisms:

1. Higher orders (*e.g*, second order propositional for propositional formulae).
2. Tseitin extension: $p \leftrightarrow A$ (where p is a fresh atom). Optimal?
3. Substitution: $\mathsf{sub}\,\dfrac{A}{A\sigma}$. Equivalent to (2).
4. Use the same sub-proof many times: dag-ness, or cocontraction.
5. Use the same sub-proof many times: cut rule.

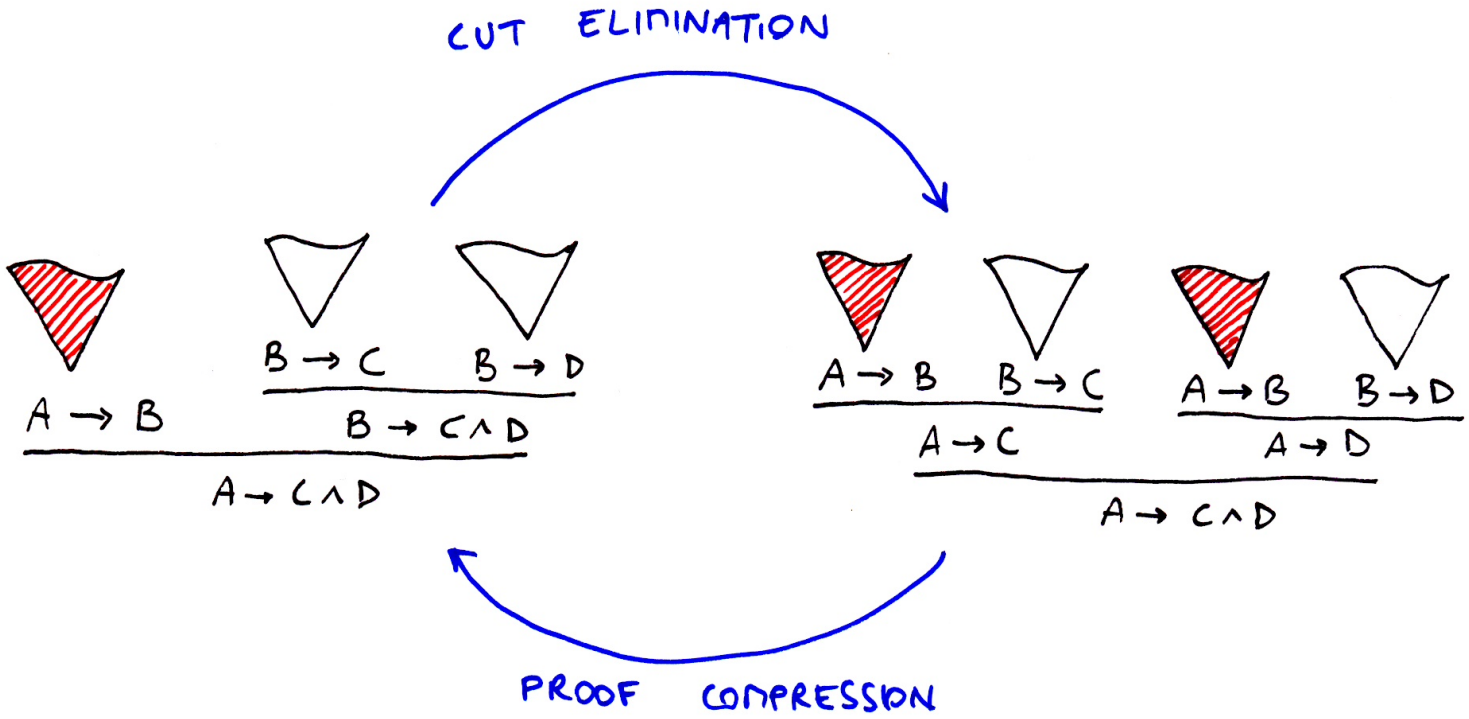
Compressing Proofs

How can we make proofs smaller?

Known mechanisms:

1. Higher orders (*e.g*, second order propositional for propositional formulae).
2. Tseitin extension: $p \leftrightarrow A$ (where p is a fresh atom). Optimal?
3. Substitution: $\mathsf{sub}\,\dfrac{A}{A\sigma}$. Equivalent to (2).
4. Use the same sub-proof many times: dag-ness, or cocontraction.
5. Use the same sub-proof many times: cut rule.
   Most studied, proof theory.

Idea of Cut Elimination



- Cuts are lifted and then eliminated against identity axioms.
- (Hyper-)exponential growth (in Gentzen).

Summary: Where Is Syntax?

Not in the notion of proof system:

- it's any algorithm with certain properties;
- Frege is robust.

Summary: Where Is Syntax?

Not in the notion of proof system:

- it's any algorithm with certain properties;
- Frege is robust.

Not in the compression mechanisms (higher orders, extension/substitution, cocontraction) ...

Summary: Where Is Syntax?

Not in the notion of proof system:

- it's any algorithm with certain properties;
- Frege is robust.

Not in the compression mechanisms (higher orders, extension/substitution, cocontraction) ...

... except for the cut and cut elimination (*i.e.*, Gentzen's proof theory).

Summary: Where Is Syntax?

Not in the notion of proof system:

- it's any algorithm with certain properties;
- Frege is robust.

Not in the compression mechanisms (higher orders, extension/substitution, cocontraction) ...

... except for the cut and cut elimination (*i.e.*, Gentzen's proof theory).

So:

1. Can we capture cut and analyticity independently of syntax?

Summary: Where Is Syntax?

Not in the notion of proof system:

- it's any algorithm with certain properties;
- Frege is robust.

Not in the compression mechanisms (higher orders, extension/substitution, cocontraction) ...

... except for the cut and cut elimination (*i.e.*, Gentzen's proof theory).

So:

1. Can we capture cut and analyticity independently of syntax?
2. Robustness?

Summary: Where Is Syntax?

Not in the notion of proof system:

- it's any algorithm with certain properties;
- Frege is robust.

Not in the compression mechanisms (higher orders, extension/substitution, cocontraction) ...

... except for the cut and cut elimination (*i.e.*, Gentzen's proof theory).

So:

1. Can we capture cut and analyticity independently of syntax?
2. Robustness?

This talk answers YES to Question (1).

(Proof) System SKS [Brünnler and Tiu, 2001]

- Atomic rules:

$\mathsf{ai}{\downarrow}\dfrac{\mathsf{t}}{a \vee \bar{a}}$	$\mathsf{aw}{\downarrow}\dfrac{\mathsf{f}}{a}$	$\mathsf{ac}{\downarrow}\dfrac{a \vee a}{a}$
identity	*weakening*	*contraction*
$\mathsf{ai}{\uparrow}\dfrac{a \wedge \bar{a}}{\mathsf{f}}$	$\mathsf{aw}{\uparrow}\dfrac{a}{\mathsf{t}}$	$\mathsf{ac}{\uparrow}\dfrac{a}{a \wedge a}$
cut	*coweakening*	*cocontraction*

(Proof) System SKS [Brünnler and Tiu, 2001]

- Atomic rules:

$\mathsf{ai}{\downarrow}\dfrac{\mathsf{t}}{a \vee \bar{a}}$	$\mathsf{aw}{\downarrow}\dfrac{\mathsf{f}}{a}$	$\mathsf{ac}{\downarrow}\dfrac{a \vee a}{a}$
identity	*weakening*	*contraction*
$\mathsf{ai}{\uparrow}\dfrac{a \wedge \bar{a}}{\mathsf{f}}$	$\mathsf{aw}{\uparrow}\dfrac{a}{\mathsf{t}}$	$\mathsf{ac}{\uparrow}\dfrac{a}{a \wedge a}$
cut	*coweakening*	*cocontraction*

- Linear rules:

$\mathsf{s}\dfrac{A \wedge [B \vee C]}{(A \wedge B) \vee C}$	$\mathsf{m}\dfrac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]}$
switch	*medial*

(Proof) System SKS [Brünnler and Tiu, 2001]

- Atomic rules:

$\mathsf{ai}{\downarrow}\dfrac{\mathsf{t}}{a \vee \bar{a}}$	$\mathsf{aw}{\downarrow}\dfrac{\mathsf{f}}{a}$	$\mathsf{ac}{\downarrow}\dfrac{a \vee a}{a}$
identity	*weakening*	*contraction*
$\mathsf{ai}{\uparrow}\dfrac{a \wedge \bar{a}}{\mathsf{f}}$	$\mathsf{aw}{\uparrow}\dfrac{a}{\mathsf{t}}$	$\mathsf{ac}{\uparrow}\dfrac{a}{a \wedge a}$
cut	*coweakening*	*cocontraction*

- Linear rules:

$\mathsf{s}\dfrac{A \wedge [B \vee C]}{(A \wedge B) \vee C}$	$\mathsf{m}\dfrac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]}$
switch	*medial*

- Plus an '$=$' linear rule (associativity, commutativity, units).

(Proof) System SKS [Brünnler and Tiu, 2001]

- Atomic rules:

$\mathsf{ai}{\downarrow}\dfrac{\mathsf{t}}{a \vee \bar{a}}$	$\mathsf{aw}{\downarrow}\dfrac{\mathsf{f}}{a}$	$\mathsf{ac}{\downarrow}\dfrac{a \vee a}{a}$
identity	*weakening*	*contraction*
$\mathsf{ai}{\uparrow}\dfrac{a \wedge \bar{a}}{\mathsf{f}}$	$\mathsf{aw}{\uparrow}\dfrac{a}{\mathsf{t}}$	$\mathsf{ac}{\uparrow}\dfrac{a}{a \wedge a}$
cut	*coweakening*	*cocontraction*

- Linear rules:

$\mathsf{s}\dfrac{A \wedge [B \vee C]}{(A \wedge B) \vee C}$	$\mathsf{m}\dfrac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]}$
switch	*medial*

- Plus an '$=$' linear rule (associativity, commutativity, units).
- Negation on atoms only.

(Proof) System SKS [Brünnler and Tiu, 2001]

- Atomic rules:

$\mathsf{ai}{\downarrow}\dfrac{\mathsf{t}}{a \vee \bar{a}}$	$\mathsf{aw}{\downarrow}\dfrac{\mathsf{f}}{a}$	$\mathsf{ac}{\downarrow}\dfrac{a \vee a}{a}$
identity	*weakening*	*contraction*
$\mathsf{ai}{\uparrow}\dfrac{a \wedge \bar{a}}{\mathsf{f}}$	$\mathsf{aw}{\uparrow}\dfrac{a}{\mathsf{t}}$	$\mathsf{ac}{\uparrow}\dfrac{a}{a \wedge a}$
cut	*coweakening*	*cocontraction*

- Linear rules:

$\mathsf{s}\dfrac{A \wedge [B \vee C]}{(A \wedge B) \vee C}$	$\mathsf{m}\dfrac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]}$
switch	*medial*

- Plus an '$=$' linear rule (associativity, commutativity, units).
- Negation on atoms only.
- Cut is atomic.

(Proof) System SKS [Brünnler and Tiu, 2001]

- Atomic rules:

$\mathsf{ai}{\downarrow}\dfrac{\mathsf{t}}{a \vee \bar{a}}$	$\mathsf{aw}{\downarrow}\dfrac{\mathsf{f}}{a}$	$\mathsf{ac}{\downarrow}\dfrac{a \vee a}{a}$
identity	*weakening*	*contraction*
$\mathsf{ai}{\uparrow}\dfrac{a \wedge \bar{a}}{\mathsf{f}}$	$\mathsf{aw}{\uparrow}\dfrac{a}{\mathsf{t}}$	$\mathsf{ac}{\uparrow}\dfrac{a}{a \wedge a}$
cut	*coweakening*	*cocontraction*

- Linear rules:

$\mathsf{s}\dfrac{A \wedge [B \vee C]}{(A \wedge B) \vee C}$	$\mathsf{m}\dfrac{(A \wedge B) \vee (C \wedge D)}{[A \vee C] \wedge [B \vee D]}$
switch	*medial*

- Plus an '$=$' linear rule (associativity, commutativity, units).
- Negation on atoms only.
- Cut is atomic.
- SKS is complete and implicationally complete for propositional logic.

Open Deduction

- $$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$

Open Deduction

- $$\mathsf{m}\,\dfrac{\dfrac{\dfrac{a}{a \wedge a}\vee \quad \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}}{}$$

- $$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{\dfrac{a \wedge \bar{a}}{\mathsf{f}}\vee \mathsf{t}} \quad \vee \quad \mathsf{t}\right]}$$

Open Deduction

- $$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a}\vee \quad \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$

- $$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\dfrac{\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{a \wedge \bar{a}}}{\mathsf{f}}\vee \mathsf{t} \quad \vee \quad \mathsf{t}\right]}$$

Proofs are composed by the same operators as formulae.

Open Deduction

- $$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$

- $$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\dfrac{\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{a \wedge \bar{a}}}{\mathsf{f}} \vee \mathsf{t} \quad \vee \quad \mathsf{t}\right]}$$

Proofs are composed by the same operators as formulae.

Top-down symmetry: so inference steps can be made atomic (the medial rule, m, is impossible in Gentzen).

Open Deduction

- $$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a}\vee \quad \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$

- $$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\dfrac{\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{a \wedge \bar{a}}}{\mathsf{f}}\vee \mathsf{t} \quad \vee \quad \mathsf{t}\right]}$$

Proofs are composed by the same operators as formulae.

Top-down symmetry: so inference steps can be made atomic (the medial rule, m, is impossible in Gentzen).

(In [Guglielmi et al., 2010a].)

Locality

Deep inference allows locality,

i.e.,

inference steps can be checked in constant time
(so, they are small).

Locality

Deep inference allows locality,

i.e.,

inference steps can be checked in constant time
(so, they are small).

E.g., atomic cocontraction:

$$\mathsf{m}\,\frac{\dfrac{\dfrac{a}{a \wedge a}\vee \quad \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}}{}$$

Locality

Deep inference allows locality,

i.e.,

inference steps can be checked in constant time
(so, they are small).

E.g., atomic cocontraction:

$$\mathsf{m}\,\frac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \frac{a}{a \wedge a}$$

In Gentzen:

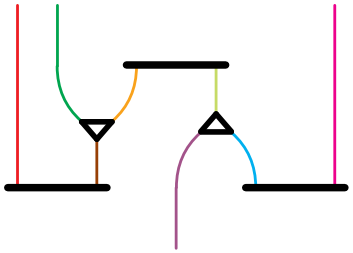
- no locality for (co)contraction (counterexample in [Brünnler, 2004]),
- no local reduction of cut into atomic form.

(Atomic) Flows

$$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{\dfrac{a \wedge \bar{a}}{\mathsf{f}} \vee \mathsf{t}} \quad \vee \quad \mathsf{t}\right]}$$

$$= \dfrac{\left(\mathsf{s}\,\dfrac{a \wedge \left[\bar{a} \vee \dfrac{\mathsf{t}}{\bar{a} \vee a}\right]}{a \wedge \dfrac{\dfrac{\bar{a} \vee \bar{a}}{\bar{a}}}{\mathsf{f}} \quad \vee \quad \dfrac{a}{a \wedge a}} \quad \wedge \quad \bar{a}\right)}{a \wedge \dfrac{a \wedge \bar{a}}{\mathsf{f}}}$$

$$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$



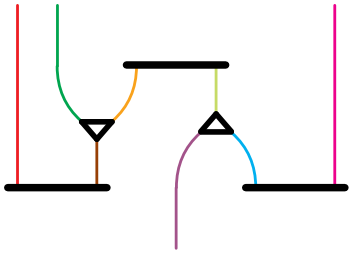
Below proofs, their (atomic) flows are shown:

(Atomic) Flows

$$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{\dfrac{a \wedge \bar{a}}{\mathsf{f}} \vee \mathsf{t}} \quad \vee \quad \mathsf{t}\right]}$$

$$= \dfrac{\left(\mathsf{s}\,\dfrac{a \wedge \left[\bar{a} \vee \dfrac{\mathsf{t}}{\bar{a} \vee a}\right]}{a \wedge \dfrac{\dfrac{\bar{a} \vee \bar{a}}{\bar{a}}}{\mathsf{f}} \quad \vee \quad \dfrac{a}{a \wedge a}} \quad \wedge \quad \bar{a}\right)}{a \wedge \dfrac{a \wedge \bar{a}}{\mathsf{f}}}$$

$$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$



Below proofs, their (atomic) flows are shown:

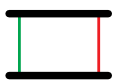
- only structural information is retained in flows;

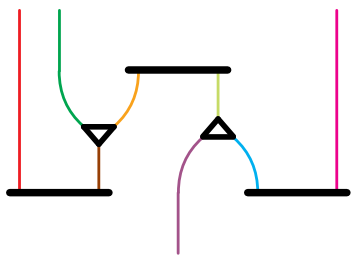
(Atomic) Flows

$$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{\dfrac{a \wedge \bar{a}}{\mathsf{f}} \vee \mathsf{t}} \quad \vee \quad \mathsf{t}\right]}$$

$$= \dfrac{\left(\mathsf{s}\,\dfrac{a \wedge \left[\bar{a} \vee \dfrac{\mathsf{t}}{\bar{a} \vee a}\right]}{a \wedge \dfrac{\dfrac{\bar{a} \vee \bar{a}}{\bar{a}}}{\mathsf{f}} \quad \vee \quad \dfrac{a}{a \wedge a}} \quad \wedge \quad \bar{a}\right)}{a \wedge \dfrac{a \wedge \bar{a}}{\mathsf{f}}}$$

$$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$





Below proofs, their (atomic) flows are shown:

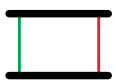
- only structural information is retained in flows;
- logical information is lost;

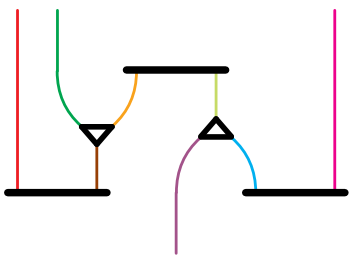
(Atomic) Flows

$$\mathsf{s}\,\dfrac{\mathsf{m}\,\dfrac{\dfrac{\mathsf{t}}{a \vee \bar{a}}}{[a \vee \mathsf{t}] \wedge [\mathsf{t} \vee \bar{a}]}}{\left[\mathsf{s}\,\dfrac{[a \vee \mathsf{t}] \wedge \bar{a}}{\dfrac{a \wedge \bar{a}}{\mathsf{f}} \vee \mathsf{t}} \quad \vee \quad \mathsf{t}\right]}$$

$$= \dfrac{\left(\mathsf{s}\,\dfrac{a \wedge \left[\bar{a} \vee \dfrac{\mathsf{t}}{\bar{a} \vee a}\right]}{a \wedge \dfrac{\dfrac{\bar{a} \vee \bar{a}}{\bar{a}}}{\mathsf{f}} \quad \vee \quad \dfrac{a}{a \wedge a}} \quad \wedge \quad \bar{a}\right)}{a \wedge \dfrac{a \wedge \bar{a}}{\mathsf{f}}}$$

$$\mathsf{m}\,\dfrac{\dfrac{a}{a \wedge a} \vee \dfrac{b}{b \wedge b}}{[a \vee b] \wedge [a \vee b]} \quad \wedge \quad \dfrac{a}{a \wedge a}$$

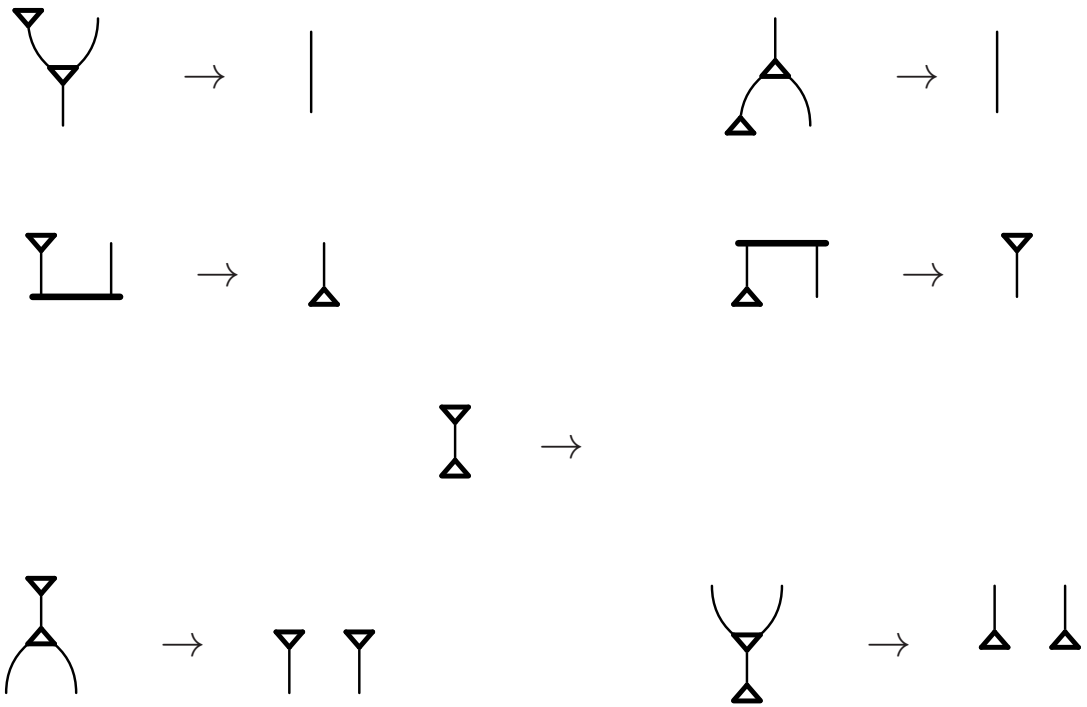




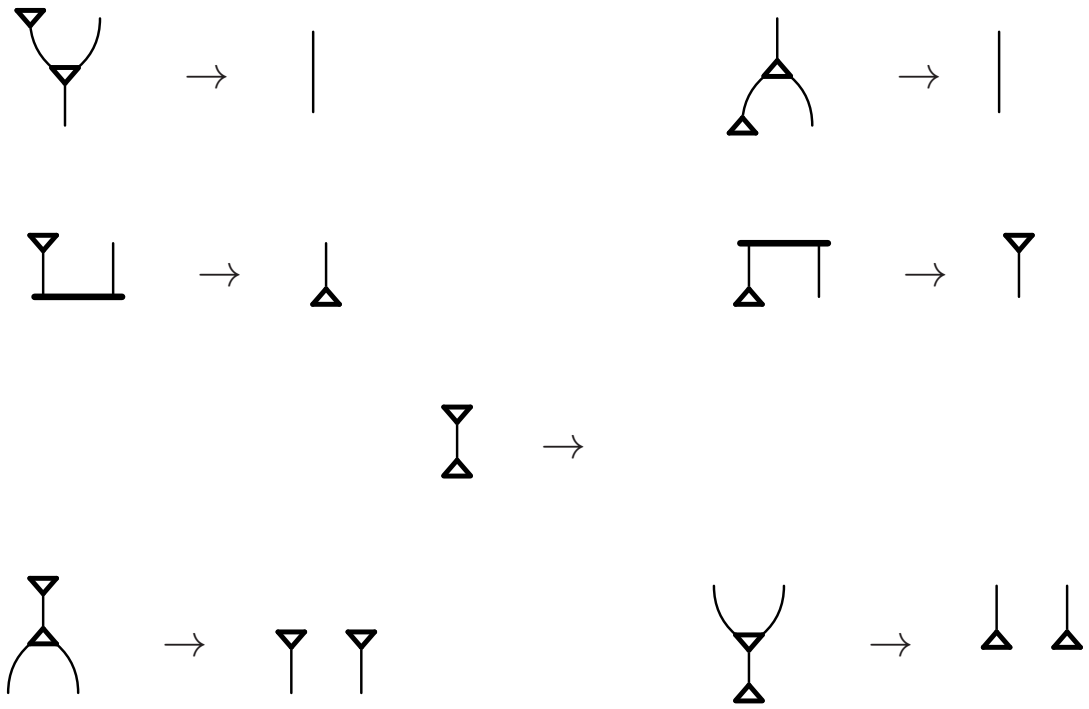
Below proofs, their (atomic) flows are shown:

- only structural information is retained in flows;
- logical information is lost;
- flow size is polynomially related to derivation size.

Flow Reductions: (Co)Weakening (1)

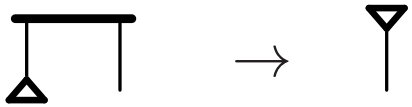


Flow Reductions: (Co)Weakening (1)



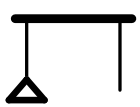
Each flow reduction corresponds to a correct proof reduction.

Flow Reductions: (Co)Weakening (2)

E.g.,  specifies that

$$\begin{array}{c} \Pi'' \Big\| \\ \xi\left\{\dfrac{\mathsf{t}}{a^{\epsilon} \vee \bar{a}}\right\} \\ \Phi \Big\| \\ \zeta\left\{\dfrac{a^{\epsilon}}{\mathsf{t}}\right\} \\ \Psi \Big\| \\ \alpha \end{array} \quad \text{becomes} \quad \begin{array}{c} \Pi'' \Big\| \\ \xi\left[\mathsf{t} \vee \dfrac{\mathsf{f}}{\bar{a}}\right] \\ \Phi\{a^{\epsilon}/\mathsf{t}\} \Big\| \\ \zeta\{\mathsf{t}\} \\ \Psi \Big\| \\ \alpha \end{array}$$

Flow Reductions: (Co)Weakening (2)

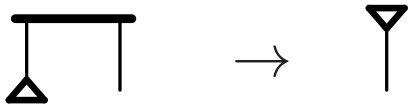
E.g.,  $\rightarrow$  specifies that

$$\begin{array}{c} \Pi'' \Vert \\ \xi\left\{\dfrac{\mathsf{t}}{a^{\epsilon} \vee \bar{a}}\right\} \\ \Phi \Vert \\ \zeta\left\{\dfrac{a^{\epsilon}}{\mathsf{t}}\right\} \\ \Psi \Vert \\ \alpha \end{array} \quad \text{becomes} \quad \begin{array}{c} \Pi'' \Vert \\ \xi\left[\mathsf{t} \vee \dfrac{\mathsf{f}}{\bar{a}}\right] \\ \Phi\{a^{\epsilon}/\mathsf{t}\} \Vert \\ \zeta\{\mathsf{t}\} \\ \Psi \Vert \\ \alpha \end{array}$$

We can operate on flow reductions instead than on derivations:

- much easier,

Flow Reductions: (Co)Weakening (2)

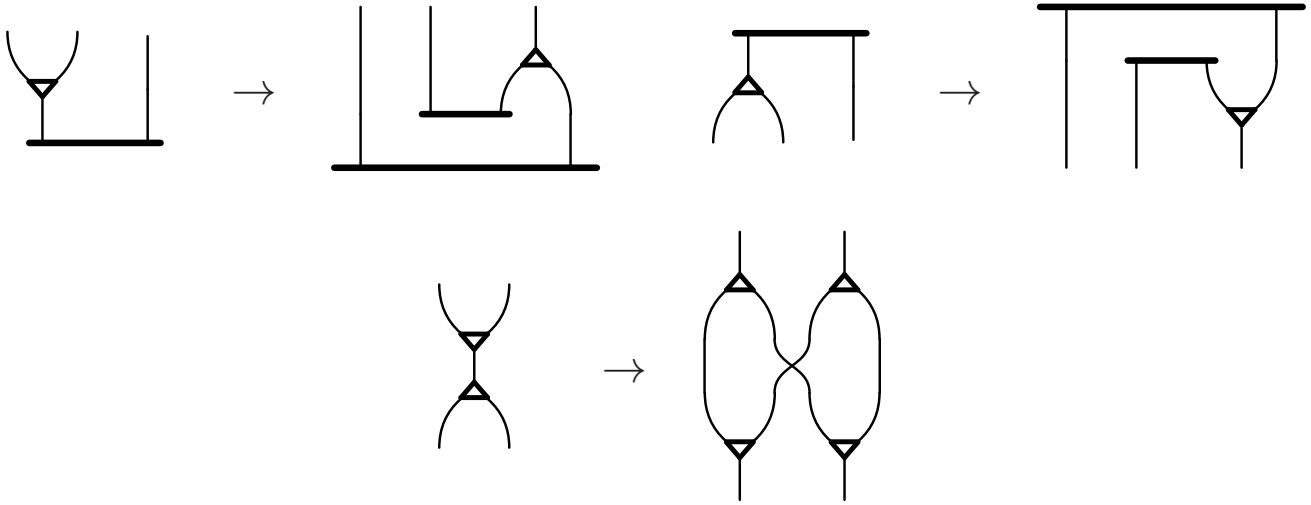
E.g.,  specifies that

$$\begin{array}{c} \Pi'' \Big\| \\ \xi\left\{\dfrac{\mathsf{t}}{a^{\epsilon} \vee \bar{a}}\right\} \\ \Phi \Big\| \\ \zeta\left\{\dfrac{a^{\epsilon}}{\mathsf{t}}\right\} \\ \Psi \Big\| \\ \alpha \end{array} \quad \text{becomes} \quad \begin{array}{c} \Pi'' \Big\| \\ \xi\left[\mathsf{t} \vee \dfrac{\mathsf{f}}{\bar{a}}\right] \\ \Phi\{a^{\epsilon}/\mathsf{t}\} \Big\| \\ \zeta\{\mathsf{t}\} \\ \Psi \Big\| \\ \alpha \end{array}$$

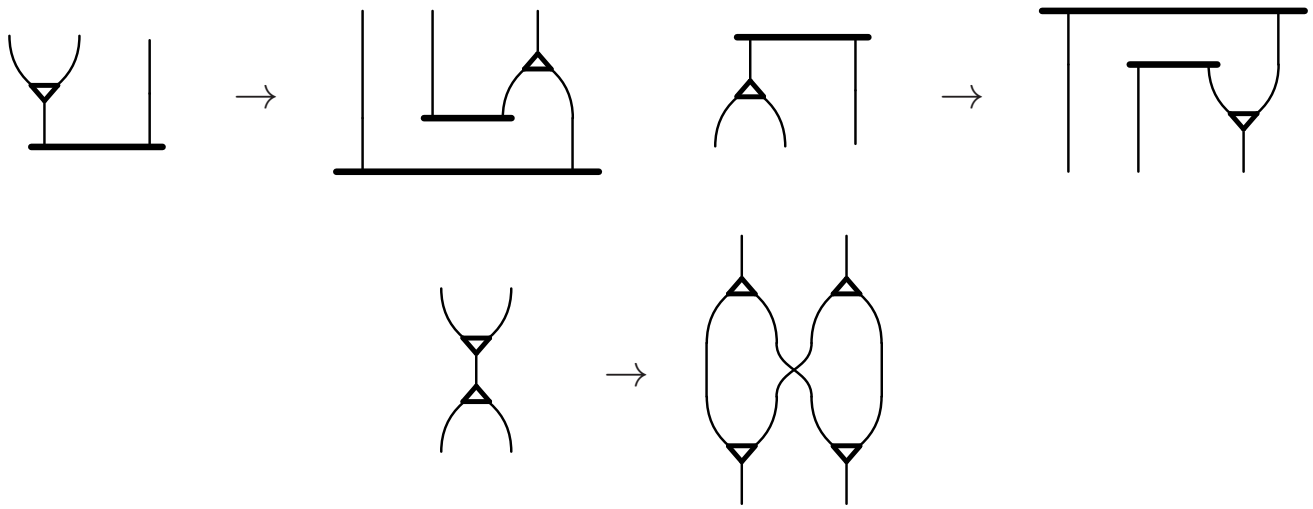
We can operate on flow reductions instead than on derivations:

- much easier,
- we get natural, syntax-independent induction measures.

Flow Reductions: (Co)Contraction

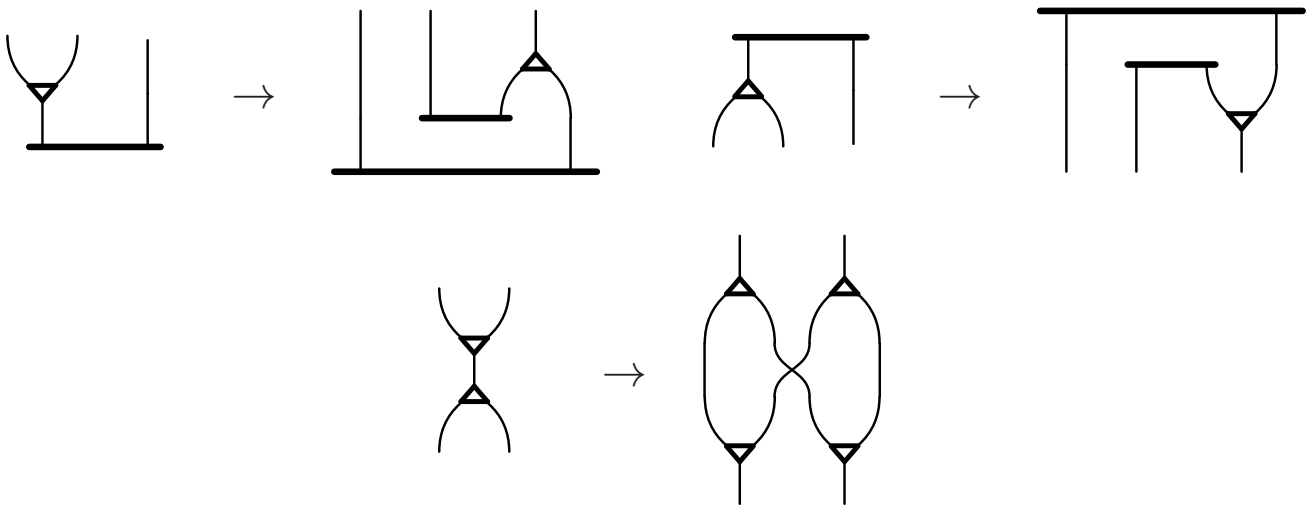


Flow Reductions: (Co)Contraction



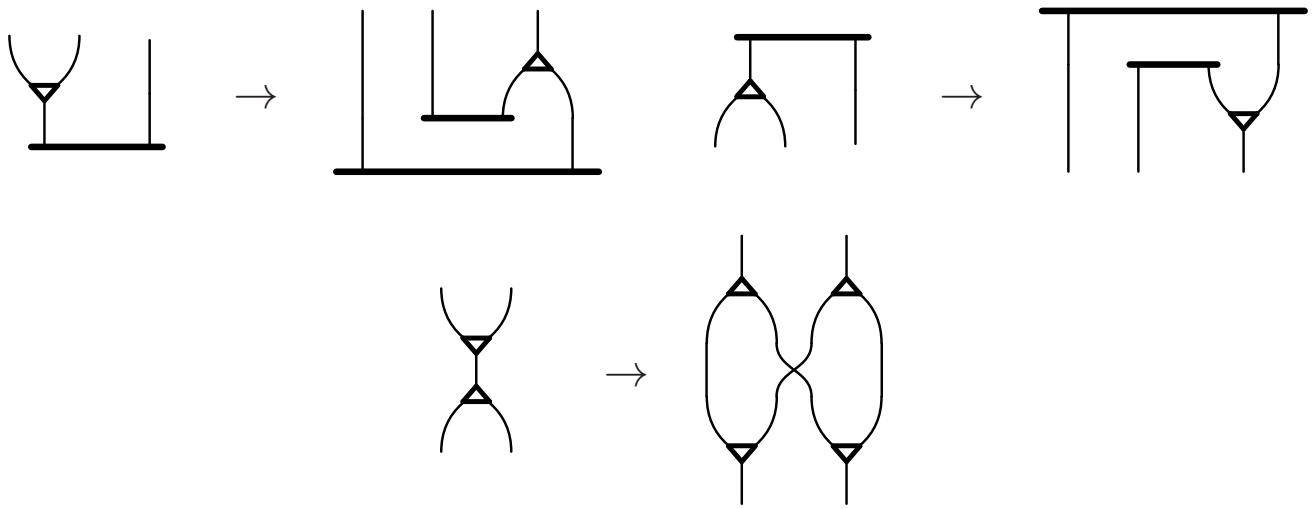
- These reductions conserve the number and length of paths.

Flow Reductions: (Co)Contraction



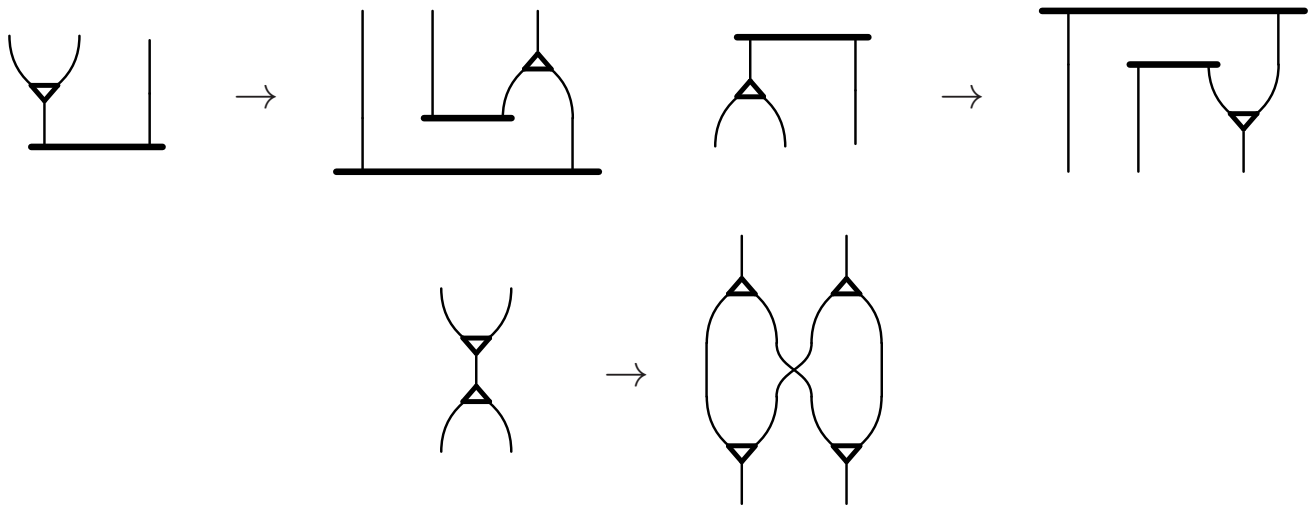
- These reductions conserve the number and length of paths.
- They can blow up a derivation exponentially.

Flow Reductions: (Co)Contraction



- These reductions conserve the number and length of paths.
- They can blow up a derivation exponentially.
- It's a good thing: cocontraction is a new compression mechanism (dag-ness?).

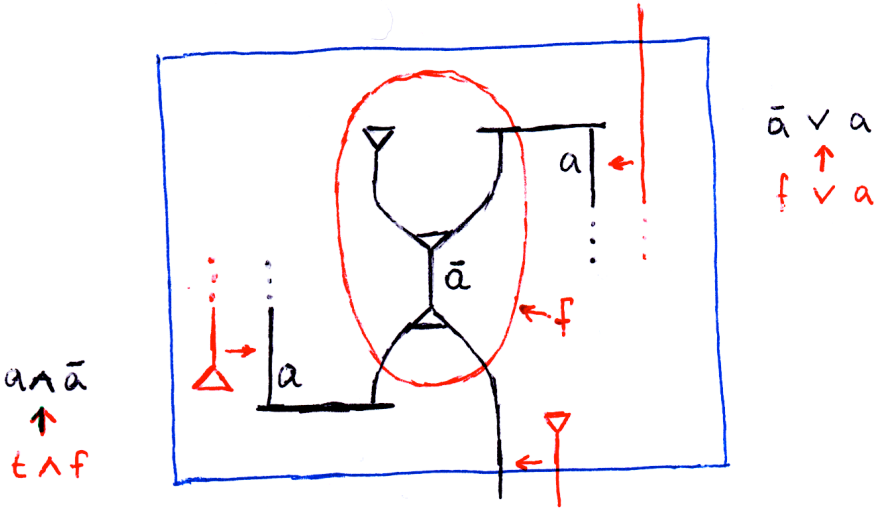
Flow Reductions: (Co)Contraction



- These reductions conserve the number and length of paths.
- They can blow up a derivation exponentially.
- It's a good thing: cocontraction is a new compression mechanism (dag-ness?).
- Open problem: does cocontraction yield exponential compression? Conjecture: yes.

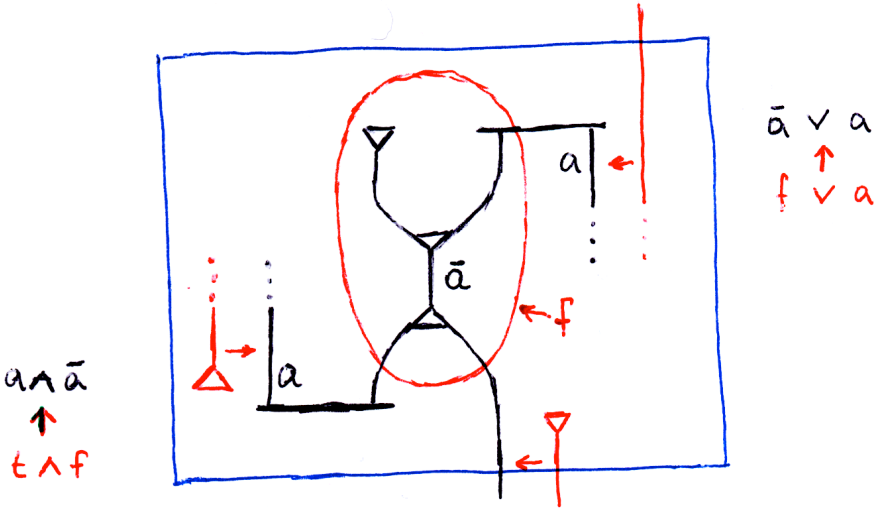
Cut Elimination by ‘Experiments’

Experiment:

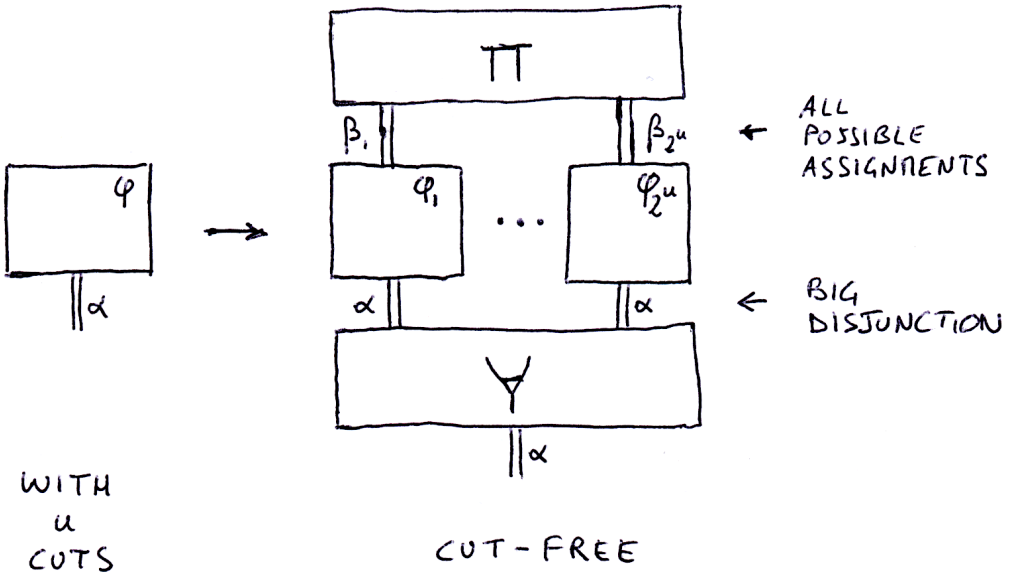


Cut Elimination by 'Experiments'

Experiment:

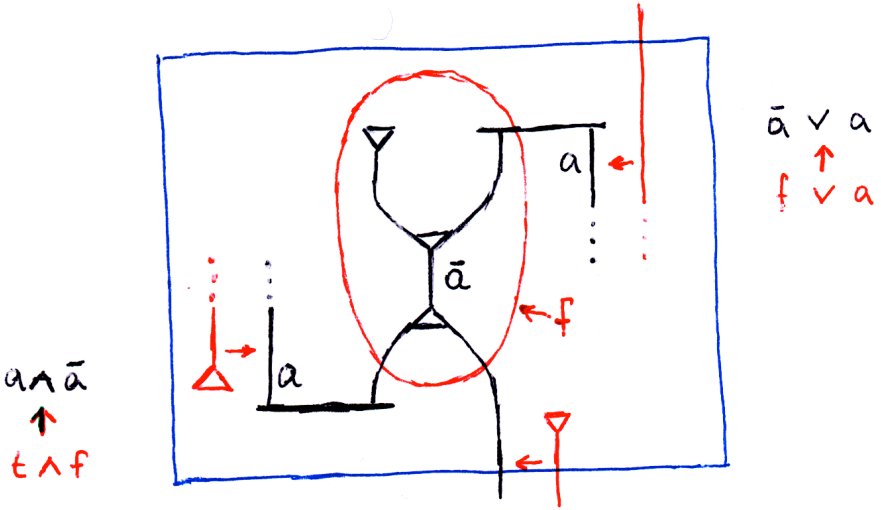


We do:

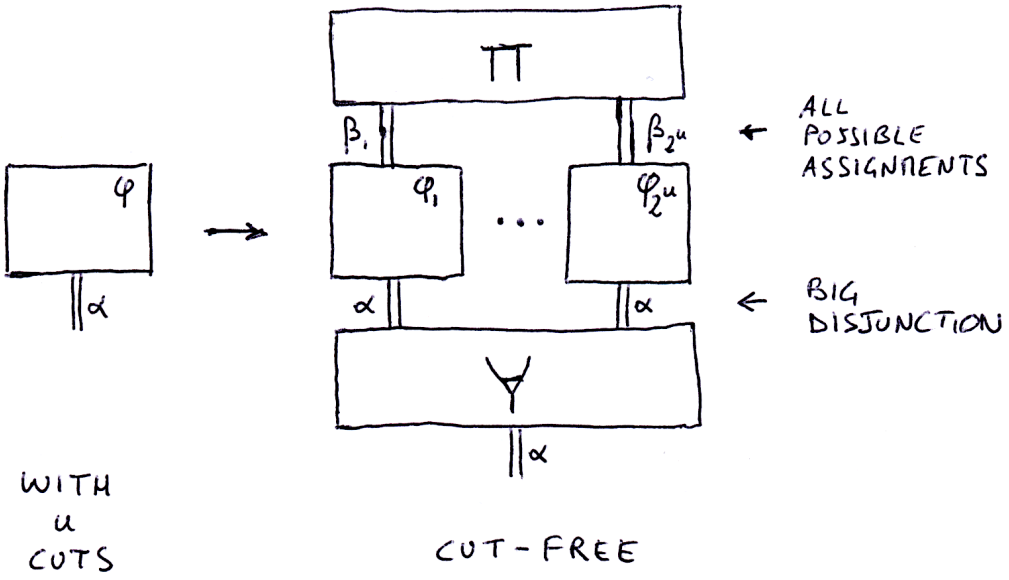


Cut Elimination by 'Experiments'

Experiment:

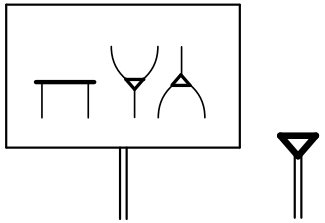


We do:



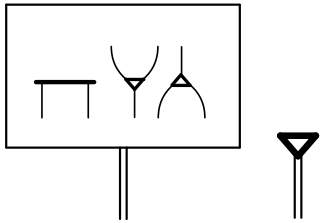
Simple, exponential cut elimination; proof generates 2^n experiments.

Generalising the Cut-Free Form

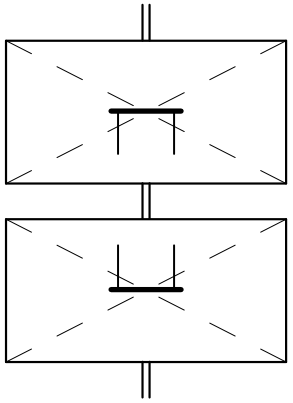
- Normalised proof: 

Generalising the Cut-Free Form

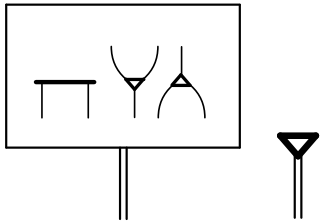
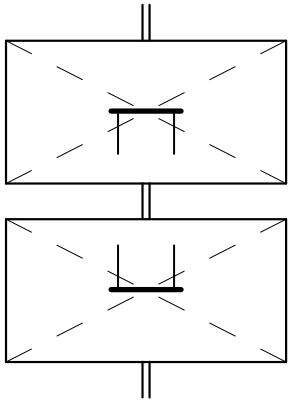
- Normalised proof:



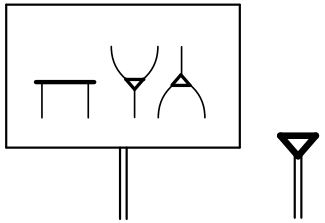
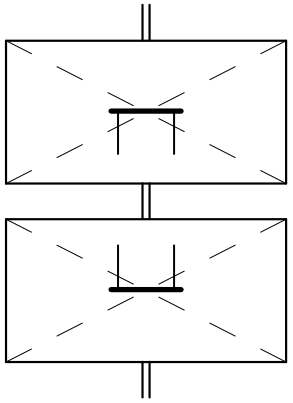
- Normalised derivation:



Generalising the Cut-Free Form

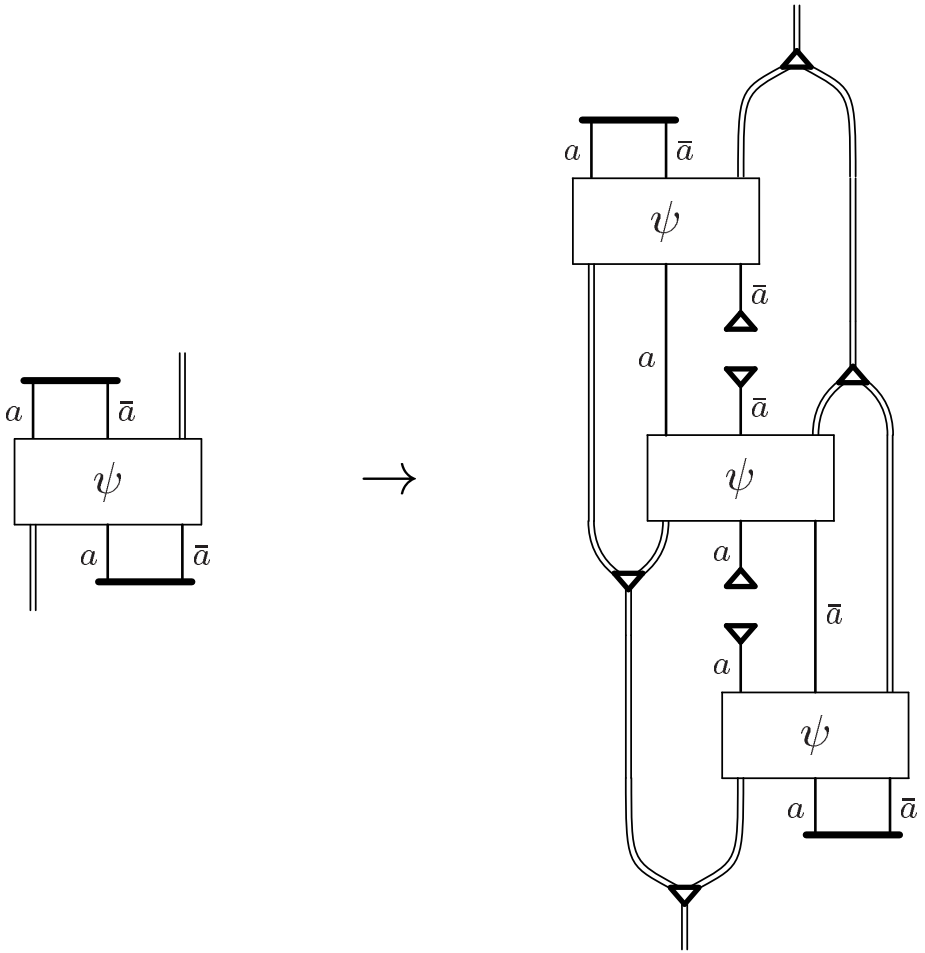
- Normalised proof: 
- Normalised derivation: 
- The symmetric form is called streamlined.
- Cut elimination is a corollary of streamlining.

Generalising the Cut-Free Form

- Normalised proof: 
- Normalised derivation: 
- The symmetric form is called streamlined.
- Cut elimination is a corollary of streamlining.
- We just need to break the paths between identities and cuts, and (co)weakenings do the rest.

How Do We Break Paths?

With the path breaker [Guglielmi et al., 2010b]:



Even if there is a path between identity and cut on the left, there is none on the right.

We Can Do This on Derivations, of Course

$$\begin{array}{c} A \\ \hline [a \vee \bar{a}] \wedge A \\ \Psi \Big\| \\ B \vee (a \wedge \bar{a}) \\ \hline B \end{array} \quad \longrightarrow \quad \begin{array}{c} A \\ \Big\| \{\mathsf{c}\uparrow, \mathsf{ai}\downarrow, =\} \\ (([a \vee \bar{a}] \wedge A) \wedge A) \wedge A \\ (\Psi \wedge A) \wedge A \Big\| \\ ([B \vee (a \wedge \bar{a})] \wedge A) \wedge A \\ \Phi_a \wedge A \Big\| \\ [B \vee ([a \vee \bar{a}] \wedge A)] \wedge A \\ [B \vee \Psi] \wedge A \Big\| \\ B \vee ([B \vee (a \wedge \bar{a})] \wedge A) \\ B \vee \Phi_{\bar{a}} \Big\| \\ B \vee [B \vee ([a \vee \bar{a}] \wedge A)] \\ B \vee [B \vee \Psi] \Big\| \\ B \vee [B \vee [B \vee (a \wedge \bar{a})]] \\ \Big\| \{\mathsf{c}\downarrow, \mathsf{ai}\uparrow, =\} \\ B \end{array}$$

We Can Do This on Derivations, of Course

$$
\begin{array}{c}
A \\
\hline
[a \vee \bar{a}] \wedge A \\
{\scriptstyle \Psi}\Big\| \\
B \vee (a \wedge \bar{a}) \\
\hline
B
\end{array}
\quad \longrightarrow \quad
\begin{array}{c}
A \\
\Big\|{\scriptstyle \{\mathsf{c}\uparrow,\mathsf{ai}\downarrow,=\}} \\
(([a \vee \bar{a}] \wedge A) \wedge A) \wedge A \\
{\scriptstyle (\Psi \wedge A) \wedge A}\Big\| \\
([B \vee (a \wedge \bar{a})] \wedge A) \wedge A \\
{\scriptstyle \Phi_a \wedge A}\Big\| \\
[B \vee ([a \vee \bar{a}] \wedge A)] \wedge A \\
{\scriptstyle [B \vee \Psi] \wedge A}\Big\| \\
B \vee ([B \vee (a \wedge \bar{a})] \wedge A) \\
{\scriptstyle B \vee \Phi_{\bar{a}}}\Big\| \\
B \vee [B \vee ([a \vee \bar{a}] \wedge A)] \\
{\scriptstyle B \vee [B \vee \Psi]}\Big\| \\
B \vee [B \vee [B \vee (a \wedge \bar{a})]] \\
\Big\|{\scriptstyle \{\mathsf{c}\downarrow,\mathsf{ai}\uparrow,=\}} \\
B
\end{array}
$$

- We can compose this as many times as there are paths between identities and cut.

We Can Do This on Derivations, of Course

$$
\begin{array}{c}
A \\
\hline
[a \vee \bar{a}] \wedge A \\
\Psi \Big\| \\
B \vee (a \wedge \bar{a}) \\
\hline
B
\end{array}
\quad \longrightarrow \quad
\begin{array}{c}
A \\
\Big\| \{\mathsf{c}\uparrow, \mathsf{ai}\downarrow, =\} \\
(([a \vee \bar{a}] \wedge A) \wedge A) \wedge A \\
(\Psi \wedge A) \wedge A \Big\| \\
([B \vee (a \wedge \bar{a})] \wedge A) \wedge A \\
\Phi_a \wedge A \Big\| \\
[B \vee ([a \vee \bar{a}] \wedge A)] \wedge A \\
[B \vee \Psi] \wedge A \Big\| \\
B \vee ([B \vee (a \wedge \bar{a})] \wedge A) \\
B \vee \Phi_{\bar{a}} \Big\| \\
B \vee [B \vee ([a \vee \bar{a}] \wedge A)] \\
B \vee [B \vee \Psi] \Big\| \\
B \vee [B \vee [B \vee (a \wedge \bar{a})]] \\
\Big\| \{\mathsf{c}\downarrow, \mathsf{ai}\uparrow, =\} \\
B
\end{array}
$$

- We can compose this as many times as there are paths between identities and cut.
- We obtain a family of normalisers that only depends on n.

We Can Do This on Derivations, of Course

$$
\begin{array}{c}
A \\
\hline
[a \vee \bar{a}] \wedge A \\
\Psi \Big\| \\
B \vee (a \wedge \bar{a}) \\
\hline
B
\end{array}
\quad \longrightarrow \quad
\begin{array}{c}
A \\
\Big\| \{\mathsf{c}\uparrow, \mathsf{ai}\downarrow, =\} \\
(([a \vee \bar{a}] \wedge A) \wedge A) \wedge A \\
(\Psi \wedge A) \wedge A \Big\| \\
([B \vee (a \wedge \bar{a})] \wedge A) \wedge A \\
\Phi_a \wedge A \Big\| \\
[B \vee ([a \vee \bar{a}] \wedge A)] \wedge A \\
[B \vee \Psi] \wedge A \Big\| \\
B \vee ([B \vee (a \wedge \bar{a})] \wedge A) \\
B \vee \Phi_{\bar{a}} \Big\| \\
B \vee [B \vee ([a \vee \bar{a}] \wedge A)] \\
B \vee [B \vee \Psi] \Big\| \\
B \vee [B \vee [B \vee (a \wedge \bar{a})]] \\
\Big\| \{\mathsf{c}\downarrow, \mathsf{ai}\uparrow, =\} \\
B
\end{array}
$$

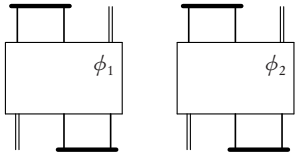
- We can compose this as many times as there are paths between identities and cut.
- We obtain a family of normalisers that only depends on n.
- The construction is exponential.

We Can Do This on Derivations, of Course

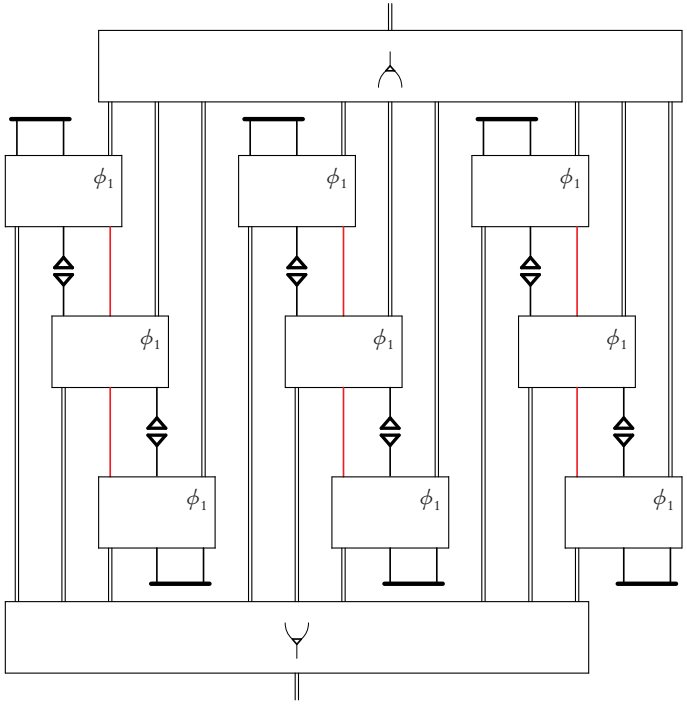
$$
\begin{array}{c}
A \\ \hline
[a \vee \bar{a}] \wedge A \\
{\scriptstyle \Psi}\Big\| \\
B \vee (a \wedge \bar{a}) \\ \hline
B
\end{array}
\quad \longrightarrow \quad
\begin{array}{c}
A \\
\Big\| {\scriptstyle \{\mathsf{c}\uparrow, \mathsf{ai}\downarrow, =\}} \\
(([a \vee \bar{a}] \wedge A) \wedge A) \wedge A \\
{\scriptstyle (\Psi \wedge A) \wedge A}\Big\| \\
([B \vee (a \wedge \bar{a})] \wedge A) \wedge A \\
{\scriptstyle \Phi_a \wedge A}\Big\| \\
[B \vee ([a \vee \bar{a}] \wedge A)] \wedge A \\
{\scriptstyle [B \vee \Psi] \wedge A}\Big\| \\
B \vee ([B \vee (a \wedge \bar{a})] \wedge A) \\
{\scriptstyle B \vee \Phi_{\bar{a}}}\Big\| \\
B \vee [B \vee ([a \vee \bar{a}] \wedge A)] \\
{\scriptstyle B \vee [B \vee \Psi]}\Big\| \\
B \vee [B \vee [B \vee (a \wedge \bar{a})]] \\
\Big\| {\scriptstyle \{\mathsf{c}\downarrow, \mathsf{ai}\uparrow, =\}} \\
B
\end{array}
$$

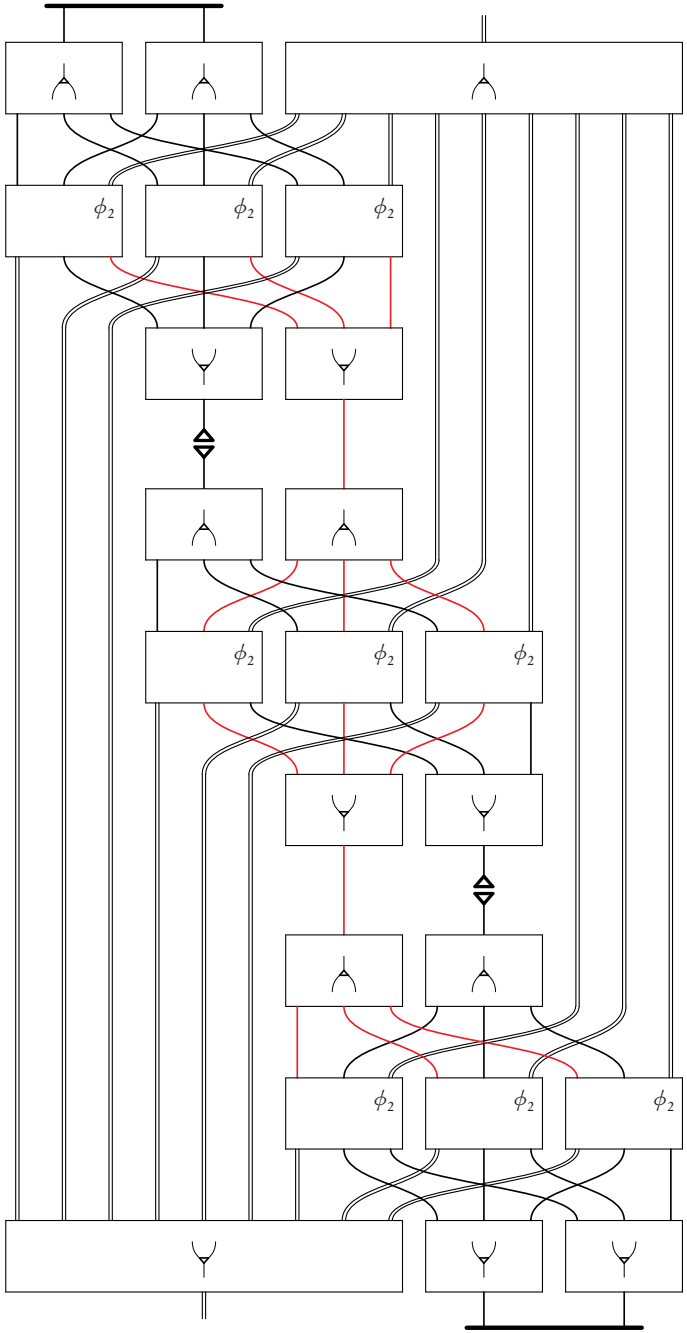
- We can compose this as many times as there are paths between identities and cut.
- We obtain a family of normalisers that only depends on n.
- The construction is exponential.
- Finding something like this is unthinkable without flows.

Example for $n = 2$

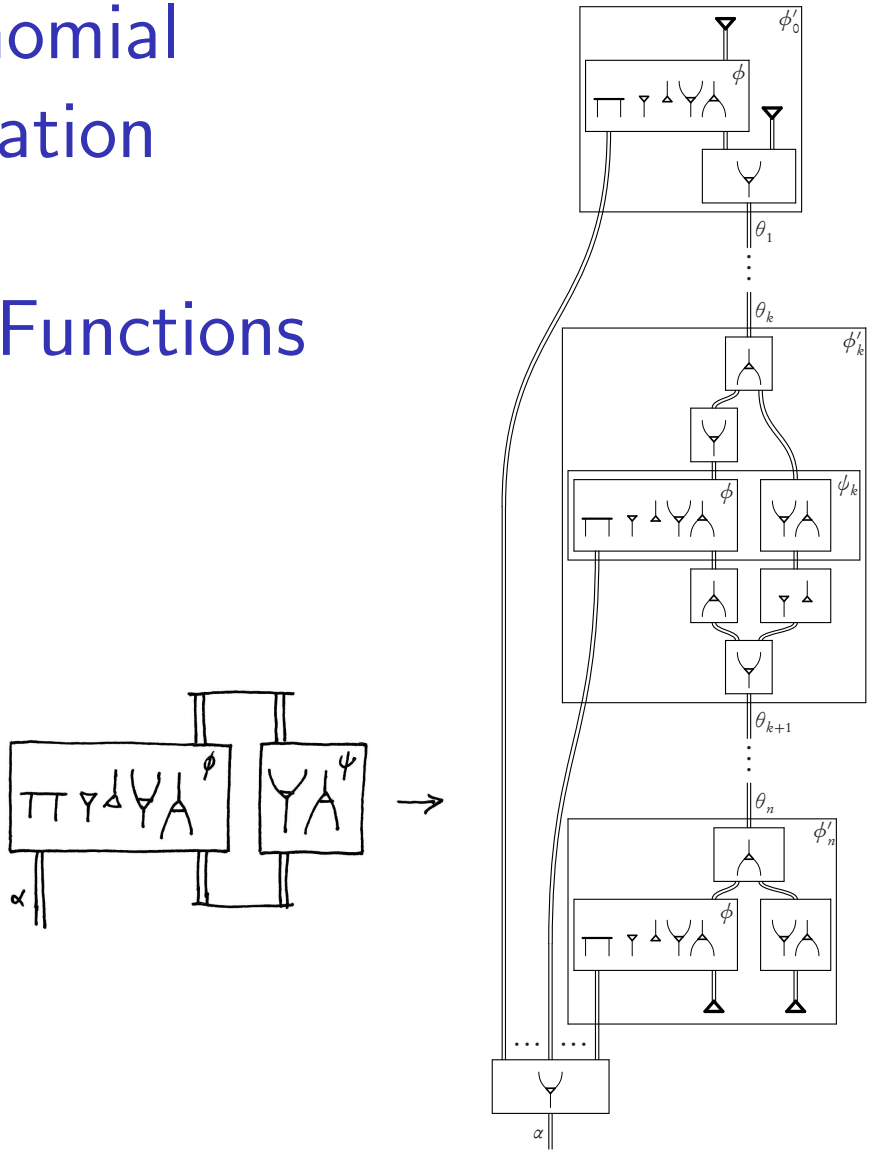


$\rightarrow$



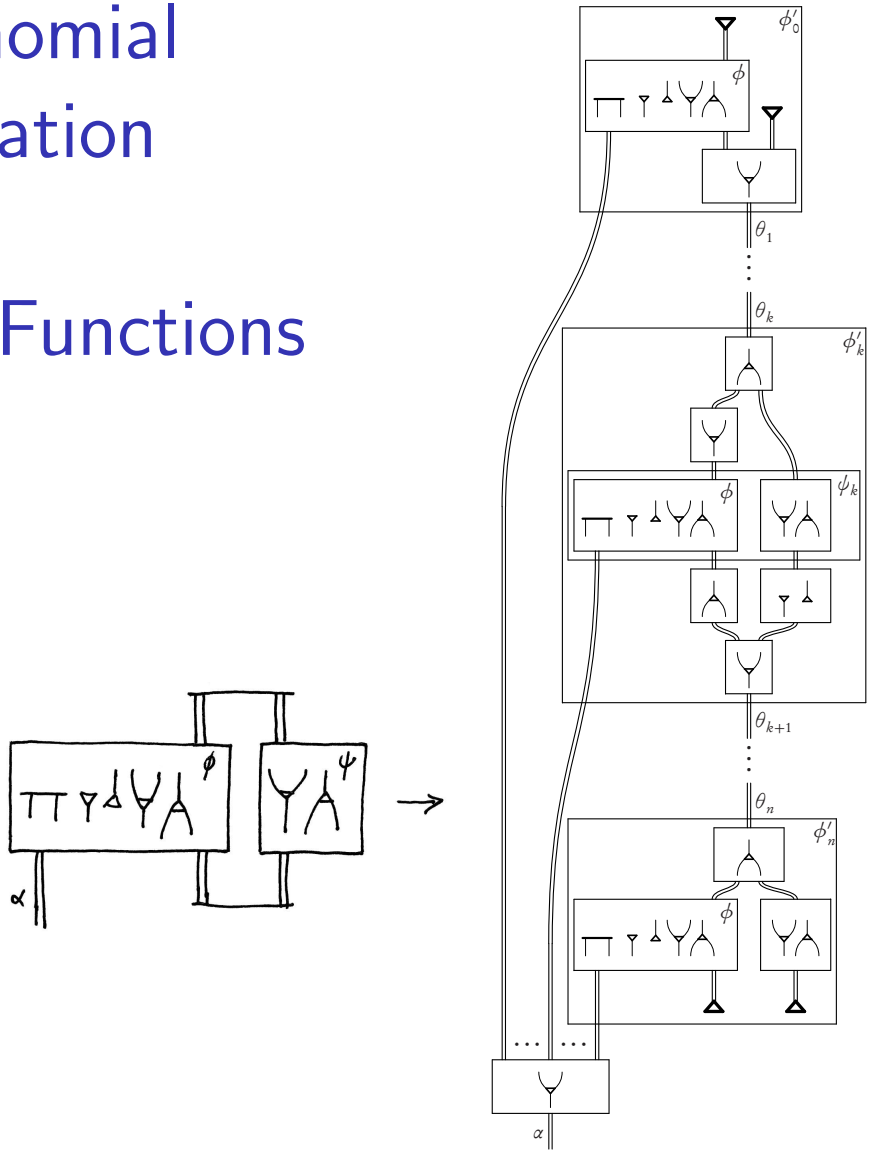


Quasipolynomial Cut Elimination by Threshold Functions



- Only $n+1$ copies of the proof are stitched together.

Quasipolynomial Cut Elimination by Threshold Functions



- Only $n+1$ copies of the proof are stitched together.
- Note local cocontraction (= better sharing, not available in Gentzen).

Normalisation
Overview

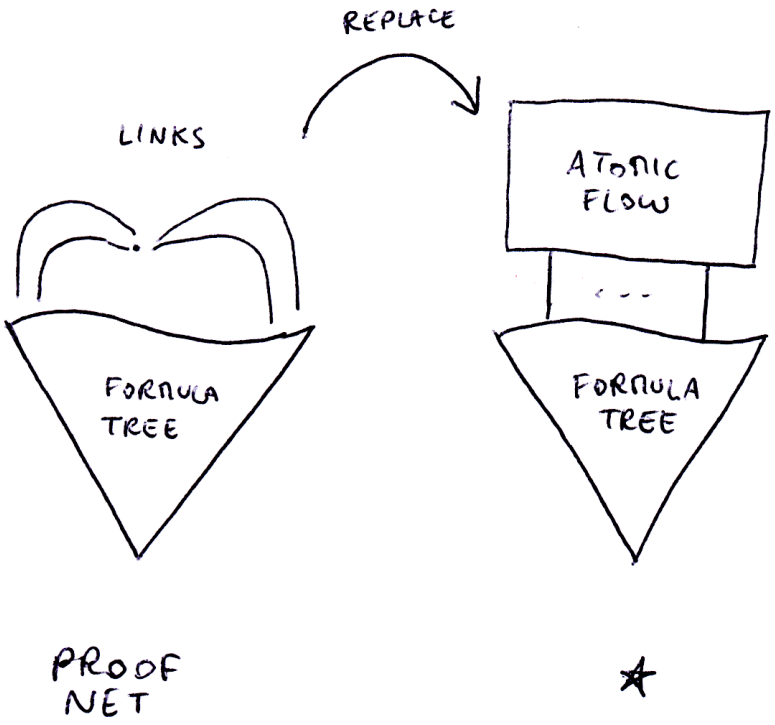
SYMMETRIC GENERALISATION

	CUT ELIMINATION	STREAMLINING
EXPONENTIAL	• SIMPLE EXPERIMENTS	• 'OPTIMISABLE' PROCEDURE (1) • 'PATH BREAKER' (2)
QUASIPOLYNOMIAL (I.E. $u^{O(\log u)}$)	• BY 'THRESHOLD FUNCTIONS' (3)	• THRESHOLD FUNCTIONS + PATH BREAKER (FORTHCOMING)

- None of these methods existed before atomic flows, none of them requires permutations or other syntactic devices.
- Quasipolynomial procedures are surprising.

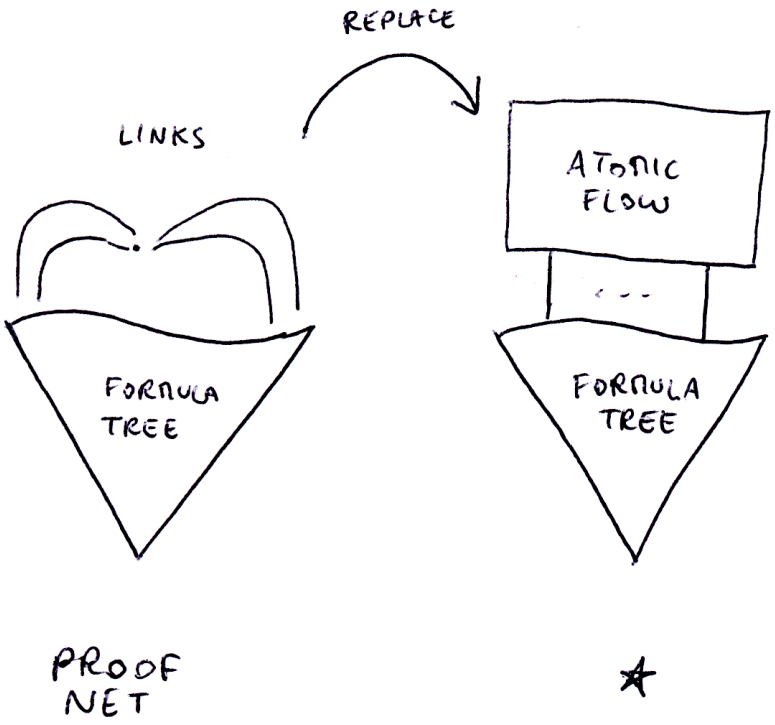
(1, 2) [Guglielmi et al., 2010b]; (3) [Bruscoli et al., 2010].

Conjecture



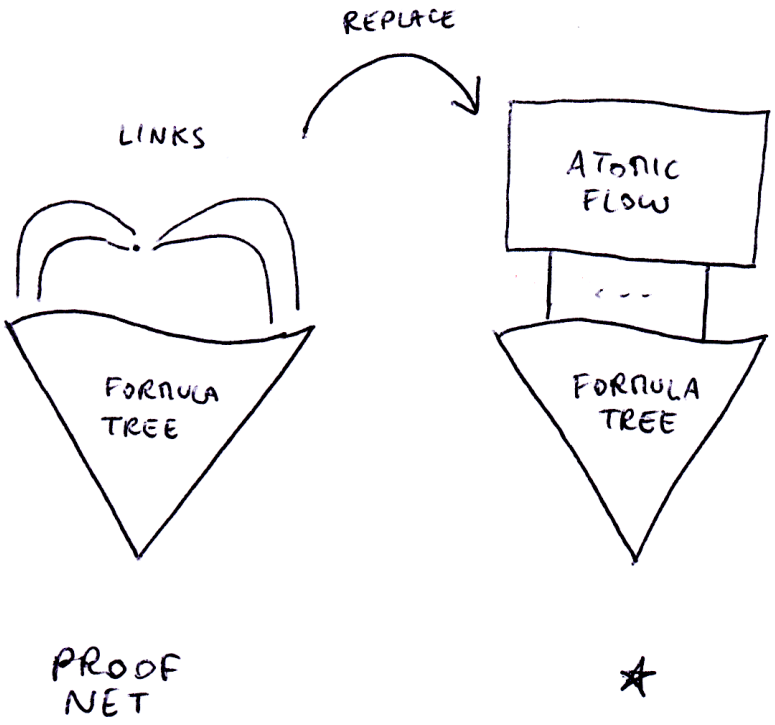
- We think that $*$ might make for a proof system.

Conjecture



- We think that $*$ might make for a proof system.
- If true, excellent bureaucracy-free formalism.

Conjecture



- We think that $*$ might make for a proof system.
- If true, excellent bureaucracy-free formalism.
- Note: if such a thing existed for proof nets, then coNP $=$ NP (because proof nets are [too?] small).

Conclusion

- Normalisation does not depend on logical rules.
- It only depends on structural information, *i.e.*, geometry.
- This is crucial progress for capturing the essence of proofs.

This talk is available at `http://cs.bath.ac.uk/ag/t/RDLS.pdf`

References

Brünnler, K. (2004).

Deep Inference and Symmetry in Classical Proofs.
Logos Verlag, Berlin.
http://www.iam.unibe.ch/~kai/Papers/phd.pdf.

Brünnler, K. and Tiu, A. F. (2001).

A local system for classical logic.
In Nieuwenhuis, R. and Voronkov, A., editors, *LPAR 2001*, volume 2250 of *Lecture Notes in Computer Science*, pages 347–361. Springer-Verlag.
http://www.iam.unibe.ch/~kai/Papers/lcl-lpar.pdf.

Bruscoli, P., Guglielmi, A., Gundersen, T., and Parigot, M. (2010).

A quasipolynomial cut-elimination procedure in deep inference via atomic flows and threshold formulae.
In Clarke, E. M. and Voronkov, A., editors, *LPAR-16*, volume 6355 of *Lecture Notes in Computer Science*, pages 136–153. Springer-Verlag.
http://cs.bath.ac.uk/ag/p/QPNDI.pdf.

Cook, S. and Reckhow, R. (1974).

On the lengths of proofs in the propositional calculus (preliminary version).
In *Proceedings of the 6th annual ACM Symposium on Theory of Computing*, pages 135–148. ACM Press.

Guglielmi, A., Gundersen, T., and Parigot, M. (2010a).

A proof calculus which reduces syntactic bureaucracy.
In Lynch, C., editor, *21st International Conference on Rewriting Techniques and Applications*, volume 6 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 135–150. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
http://drops.dagstuhl.de/opus/volltexte/2010/2649.

Guglielmi, A., Gundersen, T., and Straßburger, L. (2010b).

Breaking paths in atomic flows for classical logic.
In Jouannaud, J.-P., editor, *25th Annual IEEE Symposium on Logic in Computer Science*, pages 284–293. IEEE.
http://www.lix.polytechnique.fr/~lutz/papers/AFII.pdf.