Compressed Combinatory Proof Structures and Blending Goal- with Axiom-Driven Reasoning

Perspectives for First-Order ATP with Condensed Detachment and Clausal Tableaux

Christoph Wernhard

University of Potsdam

AITP 2022

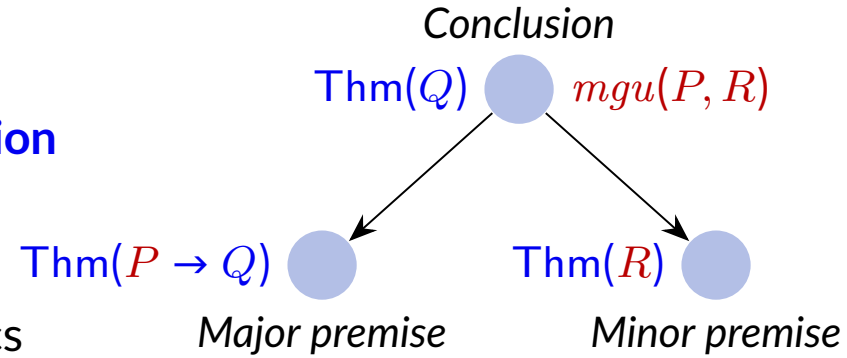
Aussois, France, September 8, 2022

Background: Goal-Driven First-Order Provers of the "TABLEAUX" Family

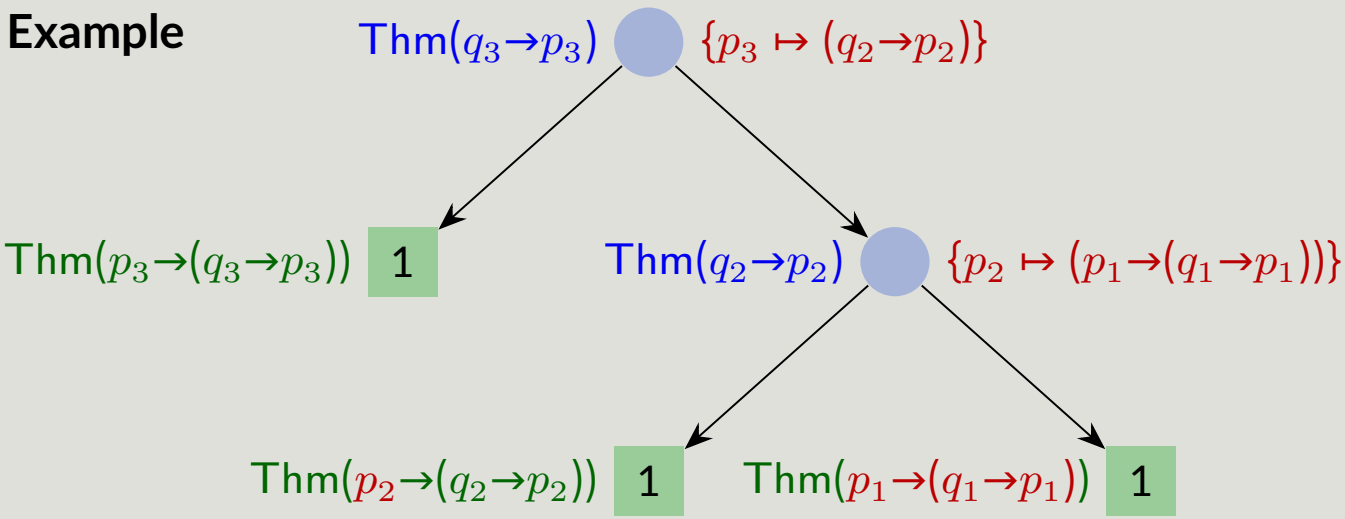
- Can be described as based on
  - Clausal tableaux
  - Connection method
  - Model elimination
- With systems such as
  - PTTP (Prolog Technology Theorem Prover) [Stickel 1988]
  - SETHEO [Letz, Bibel et al. 1992]
  - leanCoP [Otten, Bibel 2000]
  - nanoCoP, ileanCoP, MleanCoP, FEMaLeCoP, rlCoP, plCoP, lazyCoP, SATCoP, ...
- Essential operation
  - **Enumerating proof trees**
  - Interwoven with **unification of formulas** associated with nodes
  - **Iterative deepening** upon size or height of the proof tree

A Simplified Setting: Condensed Detachment

- **Condensed detachment (CD)**
  - **Detachment (modus ponens) with unification**
  - **Proof structure: full binary tree**
  - By Carew A. Meredith in the 1950s
  - Applied to reason about propositional logics



Example

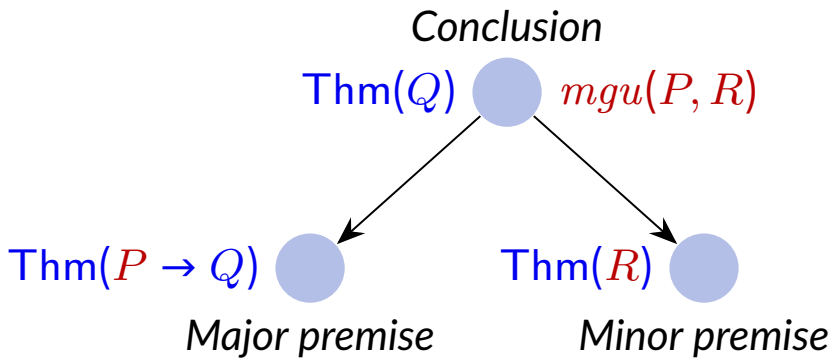


- The "top" formula is the **most general theorem (MGT)** proven by the tree wrt. axioms
- **The tree can be written as term**:

$$1(11)$$

Condensed Detachment in ATP, CD Tools

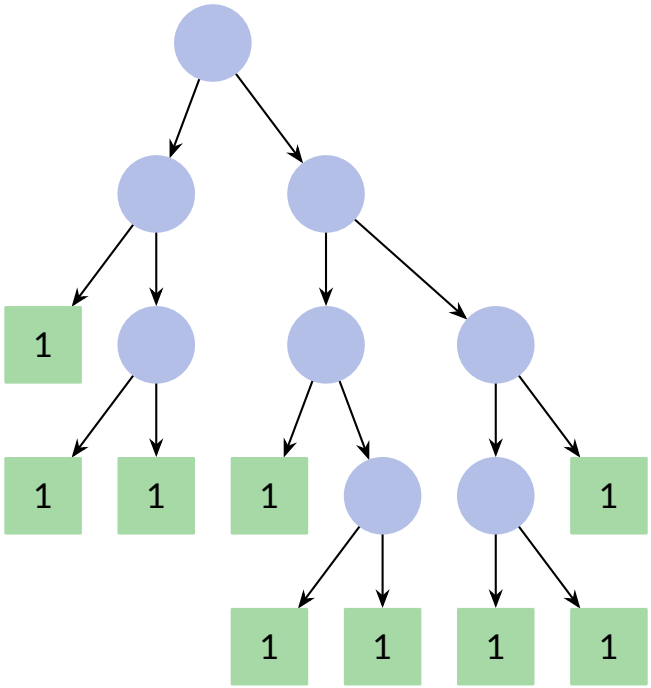
```
 1. CCCpqrCCrpCsp
 2. CCCpqpCrp = DDD1D111n
 3. CCCpqrCqr = DDD1D1D121n
 4. CpCCpqCrq = D31
 5. CCCpqCrsCCCqtsCrs = DDD1D1D1D141n
 6. CCCpqCrsCCpsCrs = D51
 7. CCpCqrCCpsrCqr = D64
 8. CCCCCpqrtCspCCrpCsp = D71
 9. CCpqCpq = D83
10. CCCCrpCtpCCCpqrsCuCCCpqrs = D18
11. CCCCpqrCsqCCCqtsCpq = DD10.10.n
12. CCCCpqrCsqCCCqtpCsq = D 5.11
13. CCCCpqrsCCsqCpq = D12.6
14. CCCpqrCCrpp = D12.9
15. CpCCpqq = D3.14
16. CCpqCCCprqq = D6.15
*17. CCpqCCqrCpr = DD.13D.16.16.13
*18. CCCpqpp = D14.9
*19. CpCqp = D33
```

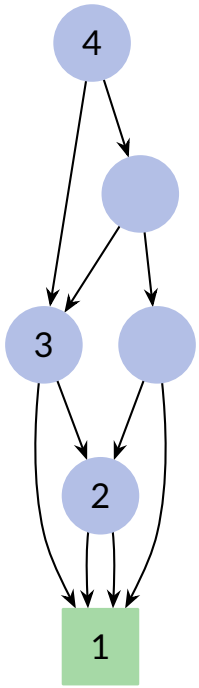


- CD is in ATP often rendered by hyperresolution with a single ternary clause
- CD is also considered in type theory [Hindley 1997]
- CD was studied as special case of the connection method [Wernhard,Bibel 2021]
- The TPTP contains about 200 CD problems (in LCL), from work by Wos and others
- CD works also for first-order Horn problems in general
- **CD Tools** – a **SWI-Prolog** library to experiment with CD
  - Includes two specialized provers: SGCD, CCS

Useful Size Measures for Proof Structures (Full Binary Trees)

- **Tree size**: 8
- **Height**: 4
- **Compacted size**: 5 – size of minimal DAG; number of distinct compound subterms





$$1(11)(1(11)(111))$$

$$\begin{aligned} 2 &= 11 \\ 3 &= 12 \\ 4 &= 3(3(21)) \end{aligned}$$

Blending Goal- with Axiom-Driven Proof Structure Enumeration

- Axiom-driven mode

```
enum_tree_and_mgt_at_treesize(+TreeSize, -Tree, -MGT)
```

  - `MGT` is the MGT of `Tree` wrt. given Axioms
  - Solutions for subproblems at lower levels can be retrieved from a **cache**

- Goal-driven mode

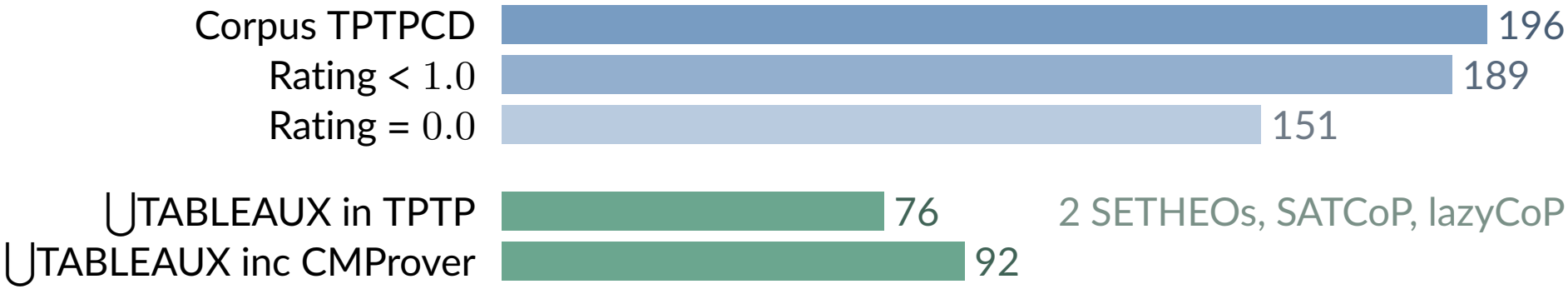
```
enum_tree_and_mgt_at_treesize(+TreeSize, -Tree, +Goal)
```

- Blending both modes

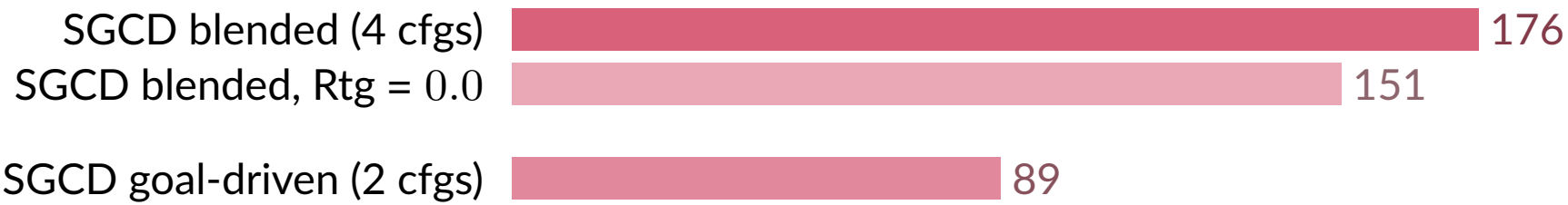
```
for MainLevel := 0 to ∞ do
    tmpstore := Tree/MGT pairs of axiom-driven mode at MainLevel
    cache := cache ∪ tmpstore
    for GoalDrivenLevel := MainLevel+1 to MainLevel+k do
        if goal-driven mode at GoalDrivenLevel with Goal succeeds
        then return Tree
```

- Axiom-driven mode **materializes MGTs**
  - Basis for **heuristic restrictions**: limiting MGT size; limiting cache size

Corpus TPTPCD: Of the 206 **CD problems in the TPTP** exclude those 10 with: status *satisfiable*; detachment with disj. and neg.; goal theorem not an atom

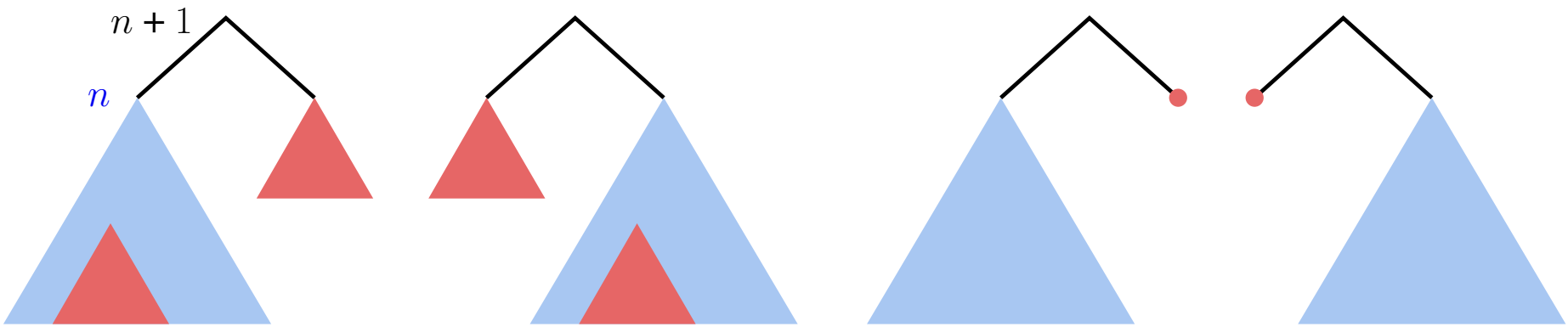


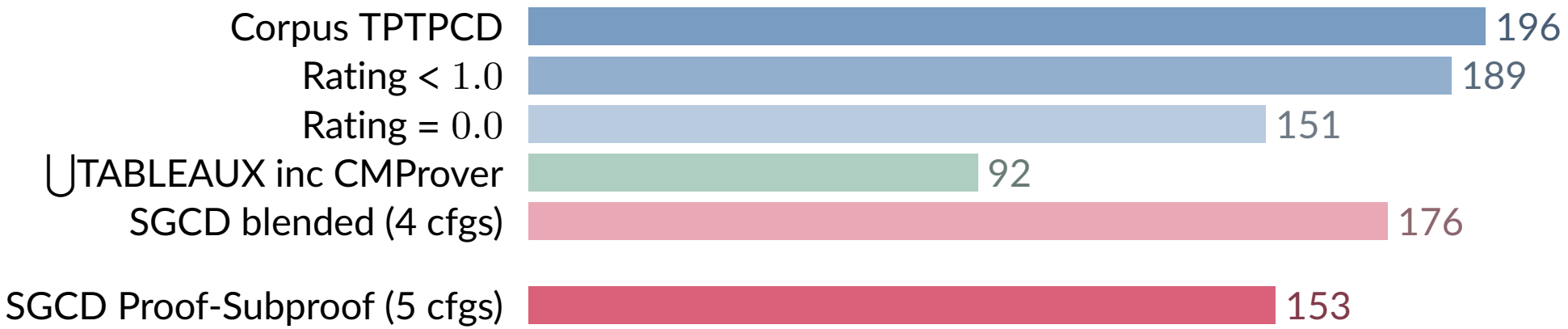
SGCD – Structure Generating theorem proving for Condensed Detachment



The “Proof-Subproof” Level Characterization

- Structural pattern found in proofs by Meredith and Łukasiewicz [W,Bibel 2021]

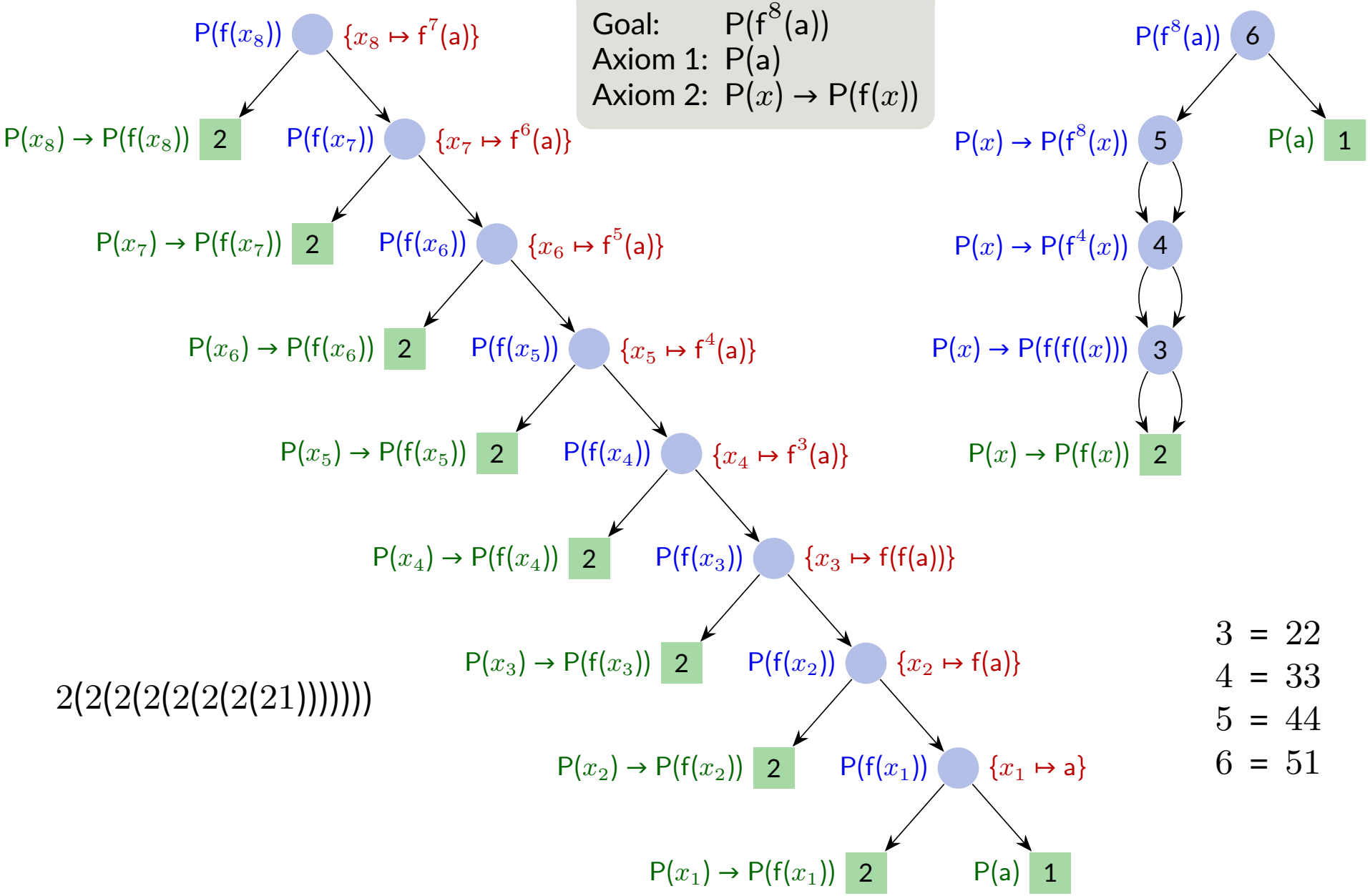




- “Proof-Subproof” yields proofs with rather small compacted size

Combinatory Compression of Proof Structures – Background: Trees vs. DAGs

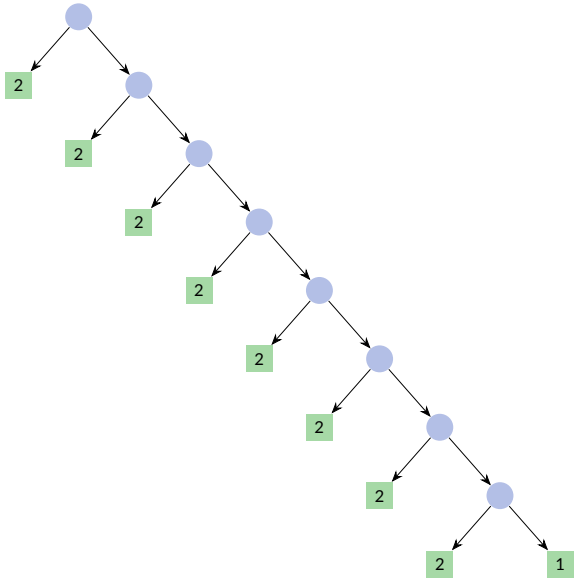
Goal: $P(f^8(a))$
Axiom 1: $P(a)$
Axiom 2: $P(x) \to P(f(x))$

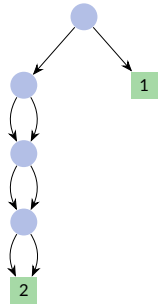


$2(2(2(2(2(2(2(21)))))))$

$3 = 22$
$4 = 33$
$5 = 44$
$6 = 51$

Enumerating DAGs Instead of Trees: Three Aspects to Consider

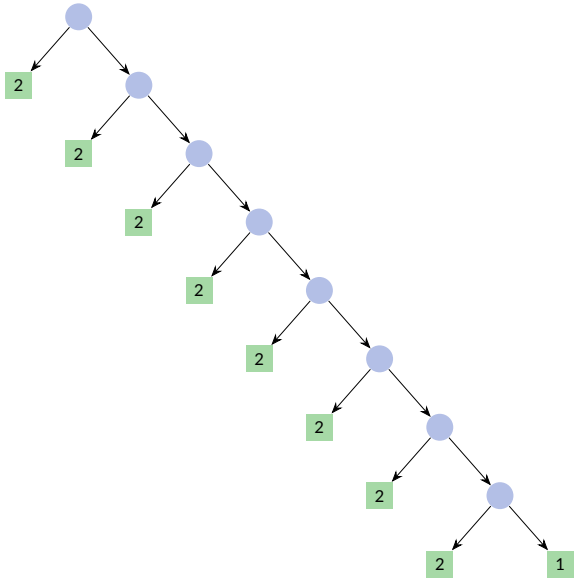


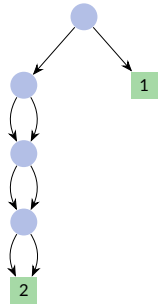


1. Each re-use of a lemma, a MGT, requires a **fresh copy**
2. **Iterative deepening on compacted size** seems more adequate than tree size or height

n	0	1	2	3	4	5	6	OEIS
Tree size	1	1	2	5	14	42	132	A000108
Height	1	1	3	21	651	457,653	210,065,930,571	A001699
Compacted size	1	1	3	15	111	1,119	14,487	A254789

Enumerating DAGs Instead of Trees: Three Aspects to Consider





1. Each re-use of a lemma, a MGT, requires a **fresh copy**
2. **Iterative deepening on compacted size** seems more adequate than tree size or height
3. Forms of duplication in the proof structure should be converted such that they **materialize as duplicated subtrees**, and become shareable in the DAG
   – the combinators come into play
   - Combinator: λ-term without free variables, e.g., $\mathbf{B} \stackrel{\text{def}}{=} \lambda f g x \,.\, f(gx)$
   - Moses Schönfinkel: *Über die Bausteine der mathematischen Logik* (1924, talk 1920)
   - Haskell Curry: *Combinatory Logic* (1958)
   - Simon L. Peyton Jones: *The Implementation of Functional Prog. Languages* (1987)
   - Stephen Wolfram: *Combinators: A Centennial View* (2021)

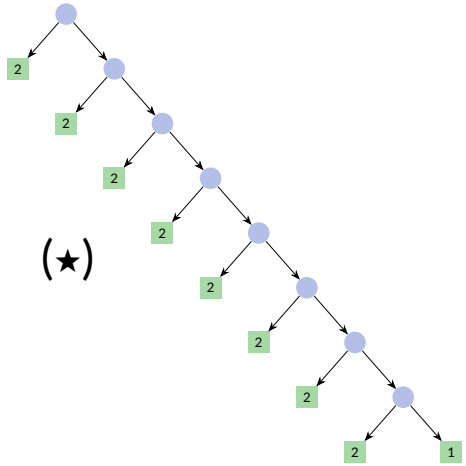
Combinatory Compression of Proof Structures

Goal: P(f^8(a)) Axioms: 1 P(a)
2 P(x) → P(f(x))

A single structure proves Goal from Axioms:

$$2(2(2(2(2(2(2(21)))))))$$

Compound subterms are 21, $2(21)$, $2(2(21))$, . . . each occur once



(★)

Goal: P(f^8(a)) Axioms: 1 P(a)
2 P(x) → P(f(x))

A single structure proves Goal from Axioms:

$$2(2(2(2(2(2(2(21)))))))) \qquad (\star)$$

Compound subterms are 21, $2(21)$, $2(2(21))$, . . . each occur once

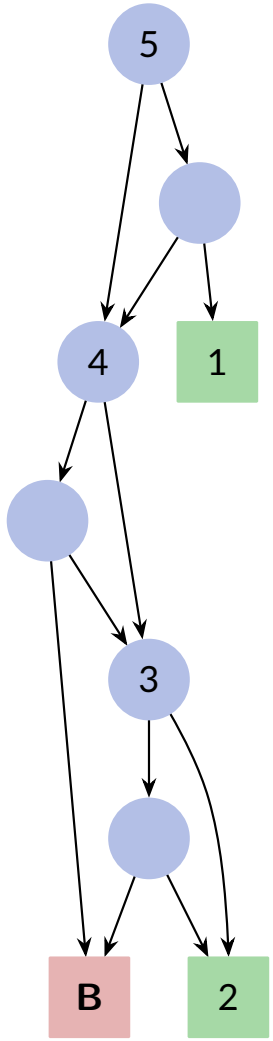
$$\mathbf{B} \stackrel{\text{def}}{=} \lambda xyz\,.\,x(yz)$$
$$\mathbf{B}xyz \to_{\text{rew}} x(yz)$$

CL-term: Proof structure term in which **combinators** are permitted

$$\mathbf{B}(\mathbf{B}22)(\mathbf{B}22)(\mathbf{B}(\mathbf{B}22)(\mathbf{B}22)1) \qquad (\star\star)$$

- ($\star\star$) **normalizes** with $\to_{\text{rew}}$ to ($\star$)

$$\begin{aligned}
& \mathbf{B}(\mathbf{B}22)(\mathbf{B}22)(\mathbf{B}(\mathbf{B}22)(\mathbf{B}22)1) && (\star\star)\\
\to_{\text{rew}}\ & \mathbf{B}(\mathbf{B}22)(\mathbf{B}22)(\mathbf{B}22(\mathbf{B}221))\\
\to_{\text{rew}}\ & \mathbf{B}(\mathbf{B}22)(\mathbf{B}22)(\mathbf{B}22(2(21)))\\
\to_{\text{rew}}\ & \mathbf{B}(\mathbf{B}22)(\mathbf{B}22)(2(2(2(21))))\\
\to_{\text{rew}}\ & \mathbf{B}22(\mathbf{B}22(2(2(2(21)))))\\
\to_{\text{rew}}\ & \mathbf{B}22(2(2(2(2(2(21))))))\\
\to_{\text{rew}}\ & 2(2(2(2(2(2(2(21))))))) && (\star)
\end{aligned}$$



$3 = \mathbf{B}22$
$4 = \mathbf{B}33$
$5 = 4(41)$

Goal: $\mathsf{P}(\mathsf{f}^8(\mathsf{a}))$ Axioms: 1 $\mathsf{P}(\mathsf{a})$
2 $\mathsf{P}(x) \to \mathsf{P}(\mathsf{f}(x))$

A single structure proves Goal from Axioms:

$$2(2(2(2(2(2(2(21)))))))) \qquad (\star)$$

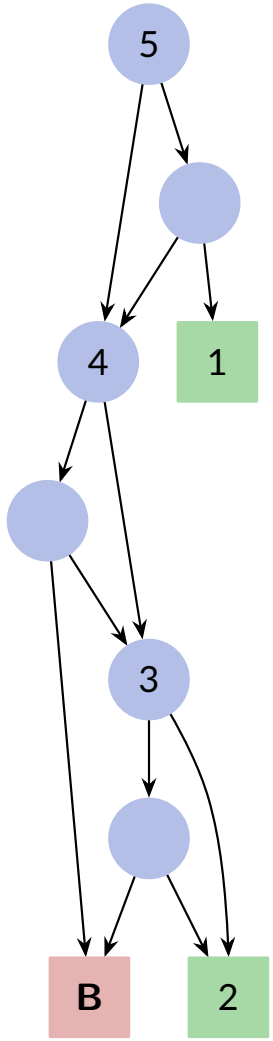
Compound subterms are 21, $2(21)$, $2(2(21))$, . . . each occur once

$$\mathbf{B} \stackrel{\text{def}}{=} \lambda xyz\,.\,x(yz)$$
$$\mathbf{B}xyz \to_{\text{rew}} x(yz)$$

CL-term: Proof structure term in which **combinators** are permitted

$$\mathbf{B}(\mathbf{B}22)(\mathbf{B}22)(\mathbf{B}(\mathbf{B}22)(\mathbf{B}22)1) \qquad (\star\star)$$

- $(\star\star)$ **normalizes** with $\to_{\text{rew}}$ to $(\star)$
- $(\star\star)$ has **multiple occurrences** of $\mathbf{B}22$ and $\mathbf{B}(\mathbf{B}22)(\mathbf{B}22)$, reflected in multiple incoming edges in its minimal DAG
- In factors **B** has 2 arguments: $\to_{\text{rew}}$ **is not applicable *within* a factor**
- $(\star\star)$ has compacted size 6, where $(\star)$ has 8; generalizes to goals $\mathsf{P}(\mathsf{f}^{2^n}(\mathsf{a}))$ with $n \geq 3$: CL-term with **B** has compacted size $2n$
- For **determining the MGT** of a CL-term, combinators are taken like axiom identifiers, denoting their principal type

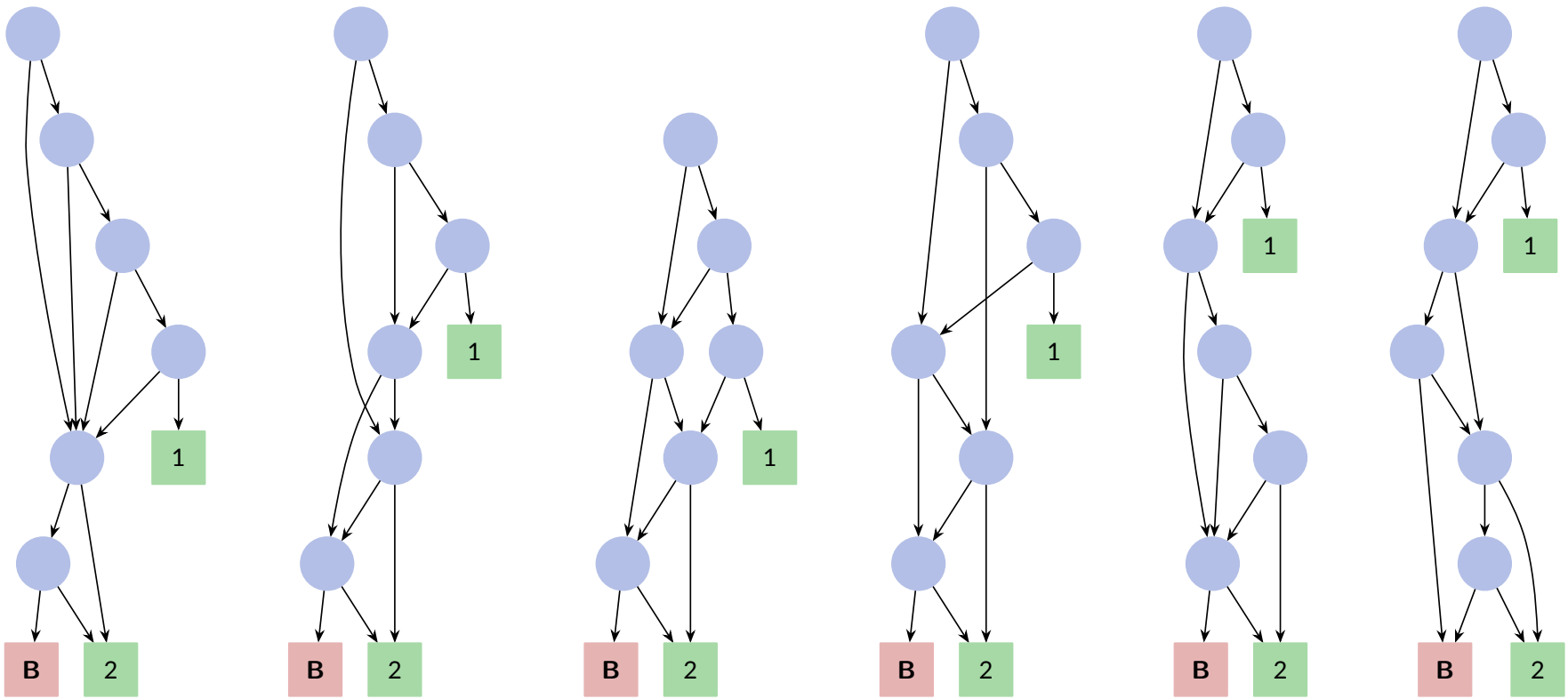


$3 = \mathbf{B}22$
$4 = \mathbf{B}33$
$5 = 4(41)$

Enumerating CL-Terms

- **CL-terms (structure terms with combinators permitted) can be enumerated analogously to pure structure terms**, interwoven with unification
- For our goal P(f^8(a)) we obtain 6 proofs with minimal compacted size 6, which all normalize to $2(2(2(2(2(2(21)))))))$

$[3 = \mathbf{B}22,\ 4 = 3(3(31))]$
$[3 = \mathbf{B}2,\ 4 = 32,\ 5 = 34,\ 6 = 4(5(51))]$
$[3 = \mathbf{B}2,\ 4 = 32,\ 5 = 34,\ 6 = 5(5(41))]$
$[3 = \mathbf{B}2,\ 4 = 32,\ 5 = 34,\ 6 = 5(4(51))]$
$[3 = \mathbf{B}2,\ 4 = 3(3(32)),\ 5 = 4(41)]$
$[3 = \mathbf{B}22,\ 4 = \mathbf{B}33,\ 5 = 4(41)]$



- Can we get rid of those proofs where **B** appears in factors with just one argument?

Enumerating Proof Terms Built from Proof Schemas Defined by CL-Terms

Proof schema: schema pattern and defining CL-term or λ-term

Schema	*CL-term*	*λ-term*
$\mathsf{r}_0(p,q)$	pq	pq
$\mathsf{r}_1(p,q)$	$\mathbf{B}pq$	$\lambda x.p(qx)$

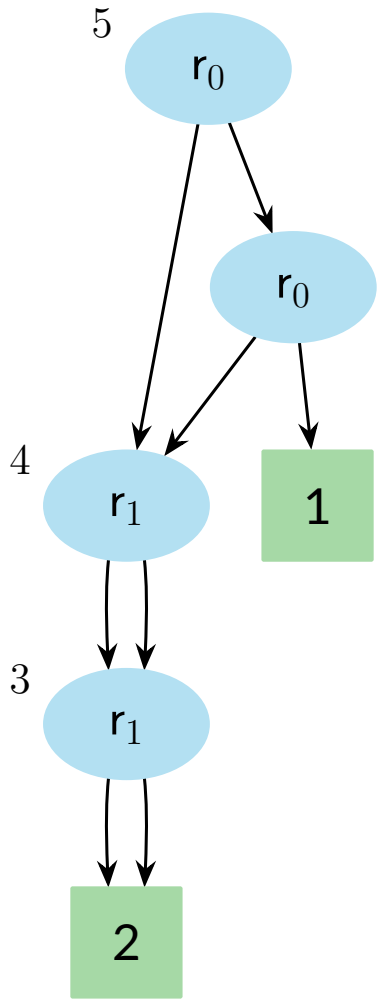
Proof schema term: built from specified schema patterns and axiom identifiers

Arity types – a refinement

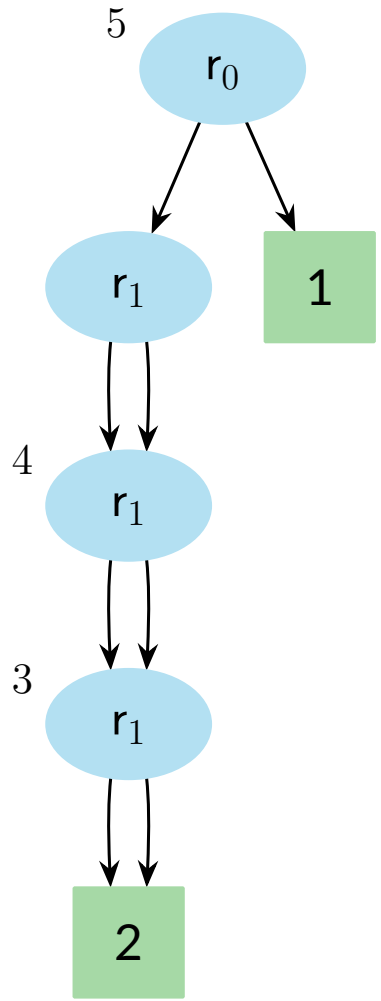
Axiom ID	*Axiom formula*
$1{:}0$	$\mathsf{P}(\mathsf{a})$
$2{:}1$	$\mathsf{P}(x) \rightarrow \mathsf{P}(\mathsf{f}(x))$

Goal
$\mathsf{P}(\mathsf{f}^n(\mathsf{a})){:}0$

Schema	*CL-term*
$\mathsf{r}_0(p{:}1, q{:}0){:}0$	pq
$\mathsf{r}_1(p{:}1, q{:}1){:}1$	$\mathbf{B}pq$



$3 = \mathsf{r}_1(2,2)$
$4 = \mathsf{r}_1(3,3)$
$5 = \mathsf{r}_0(4, \mathsf{r}_0(4,1))$

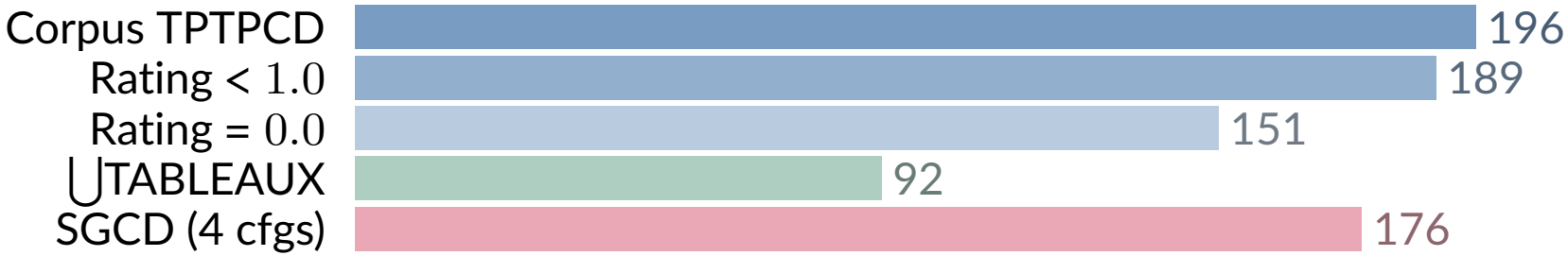


$3 = \mathsf{r}_1(2,2)$
$4 = \mathsf{r}_1(3,3)$
$5 = \mathsf{r}_0(\mathsf{r}_1(4,4), 1)$

Implementation: The CCS Prover

- **CCS** – Compressed Combinatory proof Structures
- Basically performs **iterative deepening upon compacted size**
- User input:
  1. **Problem specification**
     - Horn clauses as proper axioms
     - Optionally a goal atom
  2. **Specification of building blocks for proof structures**
     - Detachment
     - Combinators
     - Proof schemas (optionally with arity types)
- The system **compiles** this into Prolog code for search at a given compacted size
- Experiments so far: **goal-driven**, exhaustive search **without heuristic restrictions**

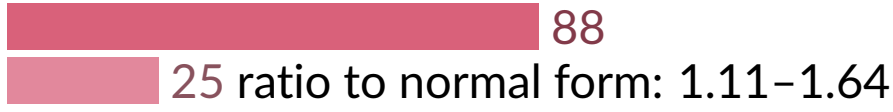
CCS: Experiments with Exhaustive Goal-Driven Search over Compacted Size



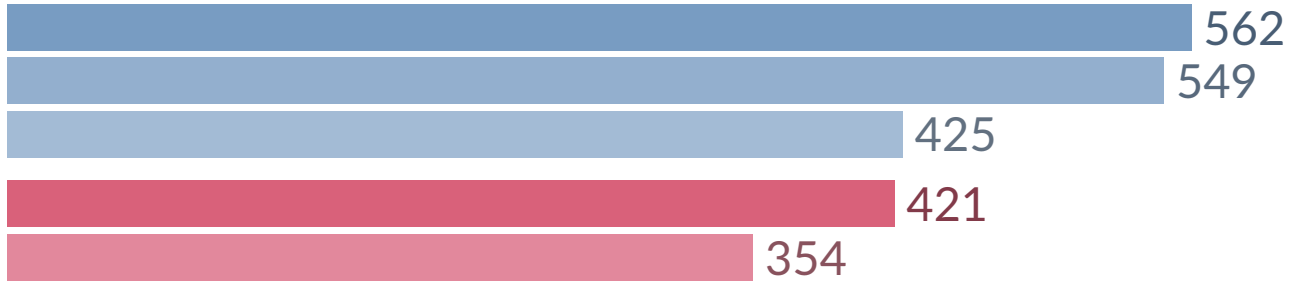
Just Detachment, no Combinators: Proofs with Minimal Compacted Size

With Selections of Proof Schemas

Schema	CL-term	λ-term	Resolution-like view
$\mathsf{r}_0(p{:}1, q{:}0){:}0$	pq	pq	$A{\to}B,\ D \vdash B\mathsf{mgu}(A, D)$
$\mathsf{r}_1(p{:}2, q{:}0, r{:}0){:}0$	pqr	pqr	$A_1{\to}(A_2{\to}B),\ D_1,\ D_2 \vdash B\mathsf{mgu}(\{(A_1, D_1), (A_2, D_2)\})$
$\mathsf{r}_4(p{:}1, q{:}1){:}1$	$\mathbf{B}pq$	$\lambda x.p(qx)$	$A{\to}B,\ C{\to}D \vdash (C{\to}B)\mathsf{mgu}(A, D)$
$\mathsf{r}_6(p{:}2, q{:}1){:}2$	$\mathbf{B}(\mathbf{C}p)q$	$\lambda xy.py(qx)$	$A_1{\to}(A_2{\to}B),\ C{\to}D \vdash (C{\to}(A_1{\to}B))\mathsf{mgu}(A_2, D)$



For General Horn Clauses: TPTP Specialist Class CNF_UNS_RFO_NEQ_HRN



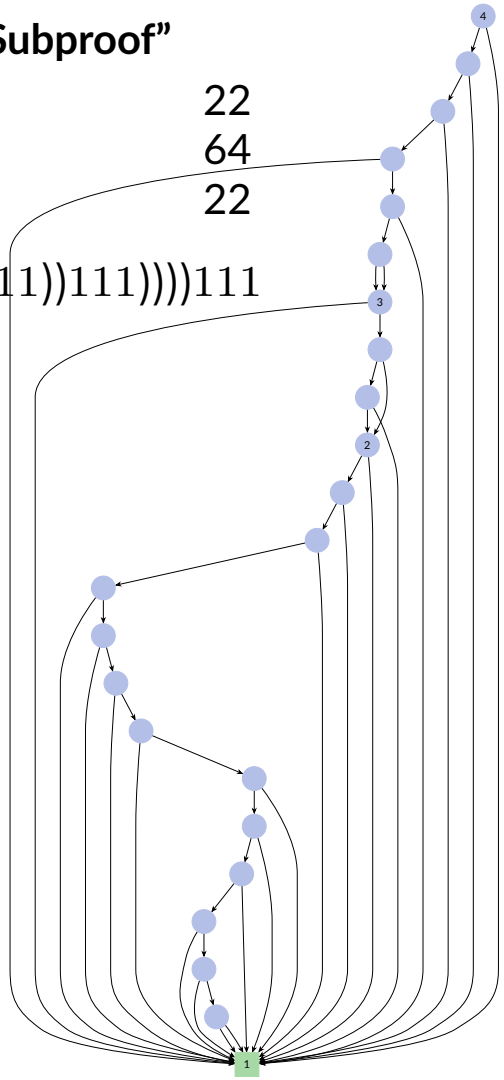
Experiment: Compressing Given Proofs: *Łukasiewicz* → *Syll* (LCL038-1)

Compressing to a **tree grammar** by TreeRePair [Lohrey, Maneth, Mennicke 2013]
⇒ Expressing **rules for non-terminals with parameters as λ-terms**
⇒ Applying the ***optimized method* from [Peyton Jones 1987]** to obtain a CL-term

SGCD with "Proof-Subproof"

Compacted size:	22
Tree size:	64
Height:	22

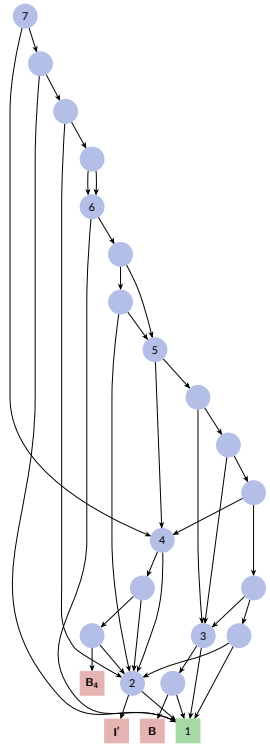
$2 = 1(1(1(1(1(1(11))111))))111$
$3 = 1(212)$
$4 = 1(331)111$



Combinatory Compression

Compacted size:	19
Tree size:	119
Height:	15

$2 = \mathbf{I'}1$
$3 = \mathbf{B}11$
$4 = \mathbf{B_4}222$
$5 = 4(3(3(4(3(21)))))$
$6 = 1(255)$
$7 = 4(1(2(66)))$



Meredith 1963

Compacted size:	31
Tree size:	491
Height:	29

Prover9		**n-simp.** [W,Bibel 2021]
Compacted size:	94	83
Tree size:	304,890	8,217
Height:	40	38

Some Related Work

- **Connection structure calculus** [Eder 1989]
  - Never implemented
- Finding **shortest proofs** [Veroff 2001]
  - Linked inference rules, expressed in Otter
- **Tree grammars** for compression, e.g., [Lohrey 2015]
  - Relationship to combinators?
  - Used in proof theory for formula instantiations [Hetzl 2012]
- Combinator terms for higher-order unification [Dougherty 1993] [Bhayat,Reger 2019] and basis for superposition for higher-order logic [Bhayat,Reger 2021]

CD Tools, SGCD, CCS: Short-Term Application Possibilities

- **Small proofs** – as basis for further processing
  - Explanations; Craig interpolation
  - **Examples for ML – e.g., dataset *FragSeq* on the CD Tools page**
- Overview on **"all" proofs** of a problem
  - Insights in structure of search space
- **Structures with combinators** as **calculus-neutral compact** proof representations
  - Proof schemas for inference rules can be **rewritten linearly** into CL-terms
- Using **just CD** as inference rule [Prover9] [Veroff 2011] can be stretched
  - Combinators can always be **normalized (but not linearly)** to pure CD
- **CD problems** themselves and their proofs [Ulrich 2001]

References I

[Bhayat and Reger, 2019] Bhayat, A. and Reger, G. (2019).
Restricted combinatory unification.
In Fontaine, P., editor, *CADE 27*, number 11716 in LNAI, pages 74–93. Springer.

[Bhayat and Reger, 2020] Bhayat, A. and Reger, G. (2020).
A combinator-based superposition calculus for higher-order logic.
In Peltier, N. and Sofronie-Stokkermans, V., editors, *IJCAR 2020*, volume 12166 of *LNCS*, pages 278–296. Springer.

[Bibel and Eder, 1993] Bibel, W. and Eder, E. (1993).
Methods and calculi for deduction.
In Gabbay, D. M., Hogger, C. J., and Robinson, J. A., editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 1, chapter 3, pages 67–182. Oxford Univ. Press.

[Curry and Feys, 1958] Curry, H. and Feys, R. (1958).
Combinatory Logic, volume I.
North-Holland.

[Dougherty, 1993] Dougherty, D. J. (1993).
Higher-order unification via combinators.
Theor. Comput. Sci., 114(2):273–298.

References II

[Eder, 1989] Eder, E. (1989).

A comparison of the resolution calculus and the connection method, and a new calculus generalizing both methods.

In Börger, E., Kleine Büning, H., and Richter, M. M., editors, *CSL '88*, volume 385 of *LNCS*, pages 80–98. Springer.

[Eder, 1992] Eder, E. (1992).

Relative Complexities of First Order Calculi.

Vieweg, Braunschweig.

[Hetzl, 2012] Hetzl, S. (2012).

Applying tree languages in proof theory.

In *LATA 2012*, volume 7183 of *LNCS*, pages 301–312.

[Hindley, 1997] Hindley, J. R. (1997).

Basic Simple Type Theory.

Cambridge University Press.

[Hindley and Meredith, 1990] Hindley, J. R. and Meredith, D. (1990).

Principal type-schemes and condensed detachment.

Journal of Symbolic Logic, 55(1):90–105.

References III

[Letz et al., 1992] Letz, R., Schumann, J., Bayerl, S., and Bibel, W. (1992).
SETHEO: A high-performance theorem prover.
J. Autom. Reasoning, 8(2):183–212.

[Lohrey, 2015] Lohrey, M. (2015).
Grammar-based tree compression.
In Potapov, I., editor, *DLT 2015*, volume 9168 of *LNCS*, pages 46–57. Springer.

[Lohrey et al., 2013] Lohrey, M., Maneth, S., and Mennicke, R. (2013).
XML tree structure compression using RePair.
Inf. Syst., 38(8):1150–1167.
System available from `https://github.com/dc0d32/TreeRePair`, accessed Jun 30, 2022.

[McCune, 2010] McCune, W. (2005–2010).
Prover9 and Mace4.
`http://www.cs.unm.edu/~mccune/prover9`.

[Meredith and Prior, 1963] Meredith, C. A. and Prior, A. N. (1963).
Notes on the axiomatics of the propositional calculus.
Notre Dame J. of Formal Logic, 4(3):171–187.

References IV

[Otten and Bibel, 2003] Otten, J. and Bibel, W. (2003).
leanCoP: lean connection-based theorem proving.
J. Symb. Comput., 36(1-2):139–161.

[Peyton Jones, 1987] Peyton Jones, S. L. (1987).
The Implementation of Functional Programming Languages.
Prentice Hall.

[Schönfinkel, 1924] Schönfinkel, M. (1924).
Über die Bausteine der mathematischen Logik.
Math. Ann., 92(3–4):305–316.

[Stickel, 1988] Stickel, M. E. (1988).
A Prolog technology theorem prover: implementation by an extended Prolog compiler.
J. Autom. Reasoning, 4(4):353–380.

[Ulrich, 2001] Ulrich, D. (2001).
A legacy recalled and a tradition continued.
J. Autom. Reasoning, 27(2):97–122.

[Ulrich, 2007] Ulrich, D. (2007).
Sentential calculi pages.
Online: `https://web.ics.purdue.edu/~dulrich/Home-page.htm`, accessed Jun 30, 2022.

References V

[Veroff, 2001] Veroff, R. (2001).

Finding shortest proofs: An application of linked inference rules.

J. Autom. Reasoning, 27(2):123–139.

[Veroff, 2011] Veroff, R. (2011).

Challenge problems with condensed detachment.

Online: `https://www.cs.unm.edu/~veroff/CD/`, accessed Jun 30, 2022.

[Wernhard, 2016] Wernhard, C. (2016).

The PIE system for proving, interpolating and eliminating.

In Fontaine, P., Schulz, S., and Urban, J., editors, *PAAR 2016*, volume 1635 of *CEUR Workshop Proc.*, pages 125–138. CEUR-WS.org.

[Wernhard, 2020] Wernhard, C. (2020).

Facets of the PIE environment for proving, interpolating and eliminating on the basis of first-order logic.

In Hofstedt, P. et al., editors, *DECLARE 2019*, volume 12057 of *LNCS (LNAI)*, pages 160–177.

[Wernhard, 2022a] Wernhard, C. (2022a).

CD Tools – Condensed detachment and structure generating theorem proving (system description).

`https://arxiv.org/abs/2207.08453`.

References VI

[Wernhard, 2022b] Wernhard, C. (2022b).

Generating compressed combinatory proof structures – an approach to automated first-order theorem proving.

In Konev, B., Schon, C., and Steen, A., editors, *PAAR 2022*, volume 3201 of *CEUR Workshop Proc.* CEUR-WS.org.

Preprint: `http://cs.christophwernhard.com/papers/css.pdf`.

[Wernhard and Bibel, 2021] Wernhard, C. and Bibel, W. (2021).

Learning from Łukasiewicz and Meredith: Investigations into proof structures.

In Platzer, A. and Sutcliffe, G., editors, *CADE 28*, volume 12699 of *LNCS (LNAI)*, pages 58–75. Springer.

[Wielemaker et al., 2012] Wielemaker, J., Schrijvers, T., Triska, M., and Lager, T. (2012).

SWI-Prolog.

Theory and Practice of Logic Programming, 12(1-2):67–96.

[Wolfram, 2021] Wolfram, S. (2021).

Combinators – A Centennial View.

Wolfram Media Inc.

Accompanying webpage: `https://writings.stephenwolfram.com/2020/12/combinators-a-centennial-view/`, accessed Jun 30, 2022.

Acknowledgments

- The author thanks Wolfgang Bibel for inspirations and for sharing an idea – more directly based on the connection method – to prove a problem series, which gave impetus to the presented combinator approach.
- The author also thanks anonymous reviewers for helpful feedback.
- Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 457292495.
- The work was supported by the North-German Supercomputing Alliance (HLRN).