Quine's Fluted Fragment is Non-elementary

Ian Pratt-Hartmann Wiesław Szwast Lidia Tendera

University of Manchester/University of Opole

25th EACSL Annual Conference on Computer Science Logic
1st September, 2016

Outline

- Fluted fragment: first identified by W.V.Quine in 1968.
- Order of quantification of variables matches order of appearance in predicates.
- Examples:

No student admires every professor

$$\forall x_1(\text{student}(x_1) \rightarrow \neg\forall x_2(\text{prof}(x_2) \rightarrow \text{admires}(x_1, x_2)))$$

No lecturer introduces any professor to every student

$$\forall x_1(\text{lecturer}(x_1) \rightarrow \neg\exists x_2(\text{prof}(x_2) \wedge \forall x_3(\text{student}(x_3) \rightarrow \text{intro}(x_1, x_2, x_3)))).$$

- The syntax of the fluted fragment is defined as follows. Let $x_1, x_2, \ldots$ be a fixed sequence of variables.
- The fluted fragment with k free variables, $\mathcal{FL}^{[k]}$, is defined by simultaneous induction for all k:
  - any atom $p(x_\ell, x_{\ell+1}, \ldots, x_k)$ is in $\mathcal{FL}^{[k]}$;
  - $\mathcal{FL}^{[k]}$ is closed under Boolean operations;
  - $\mathcal{FL}^{[k]}$ contains $\exists x_{k+1}\varphi$ and $\forall x_{k+1}\varphi$ for any $\varphi \in \mathcal{FL}^{[k+1]}$.
- The fluted fragment, $\mathcal{FL}$ is the union of all the $\mathcal{FL}^{[k]}$.
- For all $m > 0$, we define $\mathcal{FL}^m$, to be the set of fluted formulas containing at most the variables $x_1, \ldots, x_m$, free or bound.
- The allusion seems to be architectural:

- History:
  - Quine (1968): Skolem's proof for the monadic fragment of FOL generalizes to homogeneous m-adic formulas.
  - Quine (1972): We can further generalize to the fluted fragment.

- History:
  - Quine (1968): Skolem's proof for the monadic fragment of FOL generalizes to homogeneous m-adic formulas.
  - Quine (1972): We can further generalize to the fluted fragment.
  - Noah (1980): Quine's further generalization invalidates Skolem's proof.

- History:
  - Quine (1968): Skolem's proof for the monadic fragment of FOL generalizes to homogeneous m-adic formulas.
  - Quine (1972): We can further generalize to the fluted fragment.
  - Noah (1980): Quine's further generalization invalidates Skolem's proof.
  - Purdy (1996): $\mathcal{FL}$ has the finite model property; hence its satisfiability problem is decidable.

- History:
  - Quine (1968): Skolem's proof for the monadic fragment of FOL generalizes to homogeneous m-adic formulas.
  - Quine (1972): We can further generalize to the fluted fragment.
  - Noah (1980): Quine's further generalization invalidates Skolem's proof.
  - Purdy (1996): $\mathcal{FL}$ has the finite model property; hence its satisfiability problem is decidable.
  - Purdy (2002): $\mathcal{FL}$ has the exponential-sized model property; hence its satisfiability problem is in NExpTime.

- History:
  - Quine (1968): Skolem's proof for the monadic fragment of FOL generalizes to homogeneous m-adic formulas.
  - Quine (1972): We can further generalize to the fluted fragment.
  - Noah (1980): Quine's further generalization invalidates Skolem's proof.
  - Purdy (1996): $\mathcal{FL}$ has the finite model property; hence its satisfiability problem is decidable.
  - ~~Purdy (2002): $\mathcal{FL}$ has the exponential-sized model property; hence its satisfiability problem is in NExpTime.~~
- The claims in Purdy 2002 are false.

- In fact, satisfiable formulas of $\mathcal{FL}^{2m}$ force models of m-tuply exponential size, that is, of size bounded below by a function

$$\underbrace{2^{\cdot^{\cdot^{\cdot^{2^{p(\|\varphi\|)}}}}}}_{m \text{ 2's}} = \mathfrak{t}(m, p(\|\varphi\|))$$

  where p is a polynomial.
- Essentially the same construction shows that the satisfiability problem for $\mathcal{FL}^{2m}$ is m-NExpTime-hard.
- On the other hand, we also show that any satisfiable formula of $\mathcal{FL}^{m}$ has a model of m-tuply exponential size; hence, the satisfiability problem for $\mathcal{FL}^{m}$ is in m-NExpTime.
- Therefore, for $m \geq 1$, the complexity of the satisfiability problem for $\mathcal{FL}^{m}$ lies between

$$\lfloor m/2 \rfloor\text{-NExpTime-hard} \quad \text{and} \quad m\text{-NExpTime}.$$

Outline

- For any $z \geq 1$, the canonical representation of any integer n $(0 \leq n < 2^z - 1)$ is its conventional binary encoding

$$s_{z-1}, \ldots, s_0.$$

- For n in this range, to compute $n - 1 \mod 2^z$:

  for all i $(0 < i < z)$, flip the ith digit of n just in case all digits of n less significant than the ith are 0 .

- Let int_1 and $p_0, \ldots, p_{n-1}$ be unary predicates. We refer to any object b satisfying int_1 (in some structure) as a 1-integer.
- Define $\text{val}_1(b)$ to be the integer $s_{n-1}, \ldots, s_0$, where, for all i $(0 \leq i < n)$,

$$s_i = \begin{cases} 1 & \text{if } \mathfrak{A} \models p_i[b]; \\ 0 & \text{otherwise.} \end{cases}$$

- We can easily force a binary predicate pred_1 to denote the predecessor relation for 1-integers:

$$\forall x_1 \forall x_2 \left(\mathsf{pred}_1(x_1, x_2) \leftrightarrow \bigwedge_{i=0}^{n-1} \left(\left[\bigwedge_{0 \leq j < i} \neg p_i(x_1) \right] \leftrightarrow [p_i(x_1) \leftrightarrow \neg p_i(x_2)] \right) \right)$$

- Suppose a structure $\mathfrak{A}$ also makes the following true:

$$\exists x_1 \mathsf{int}_1(x_1) \wedge \forall x_1 (\mathsf{int}_1(x_1) \rightarrow \exists x_2 (\mathsf{int}_1(x_2) \wedge \mathsf{pred}_1(x_1, x_2))).$$

Then $\mathfrak{A}$ contains 1-integers with all values in the range $[0, 2^n - 1]$, whence $|A| \geq 2^n$.

- The above formula defining pred_1 is not fluted; but we can make it so using a trick.
- We shadow the unary predicates $p_0, \ldots, p_{n-1}$ with binary predicates $p_0^1, \ldots, p_{n-1}^1$, and write, for all i $(0 \leq i < n)$

$$\forall x_1 (p_i(x_1) \rightarrow \forall x_2 . p_i^1(x_1, x_2))$$
$$\forall x_1 (\neg p_i(x_1) \rightarrow \forall x_2 \neg p_i^1(x_1, x_2))$$

- Now we can perform a simple replacement

$$\forall x_1 \forall x_2 \left(\text{pred}_1(x_1, x_2) \leftrightarrow \right.$$
$$\left. \bigwedge_{i=0}^{n-1} \left(\left[\bigwedge_{0 \leq j < i} \neg p_i(x_1) \right] \leftrightarrow [p_i(x_1) \leftrightarrow \neg p_i(x_2)] \right) \right)$$

- The above formula defining pred_1 is not fluted; but we can make it so using a trick.
- We shadow the unary predicates $p_0, \ldots, p_{n-1}$ with binary predicates $p_0^1, \ldots, p_{n-1}^1$, and write, for all i $(0 \leq i < n)$

$$\forall x_1 (p_i(x_1) \rightarrow \forall x_2 . p_i^1(x_1, x_2))$$
$$\forall x_1 (\neg p_i(x_1) \rightarrow \forall x_2 \neg p_i^1(x_1, x_2))$$

- Now we can perform a simple replacement

$$\forall x_1 \forall x_2 \left(\text{pred}_1(x_1, x_2) \leftrightarrow \bigwedge_{i=0}^{n-1} \left(\left[\bigwedge_{0 \leq j < i} \neg p_i(x_1) \right] \leftrightarrow [p_i(x_1) \leftrightarrow \neg p_i(x_2)] \right) \right)$$

- The above formula defining pred_1 is not fluted; but we can make it so using a trick.
- We shadow the unary predicates $p_0, \ldots, p_{n-1}$ with binary predicates $p_0^1, \ldots, p_{n-1}^1$, and write, for all i $(0 \leq i < n)$

$$\forall x_1 (p_i(x_1) \rightarrow \forall x_2 . p_i^1(x_1, x_2))$$
$$\forall x_1 (\neg p_i(x_1) \rightarrow \forall x_2 \neg p_i^1(x_1, x_2))$$

- Now we can perform a simple replacement

$$\forall x_1 \forall x_2 \left(\text{pred}_1(x_1, x_2) \leftrightarrow \right.$$
$$\left. \bigwedge_{i=0}^{n-1} \left(\left[\bigwedge_{0 \leq j < i} \neg p_i^1(x_1, x_2) \right] \leftrightarrow \left[p_i^1(x_1, x_2) \leftrightarrow \neg p_i(x_2) \right] \right) \right)$$

- By writing further fluted formulas, we can fix the predicates zero_1 and eq_1 such that, for all 1-integers b and b':1

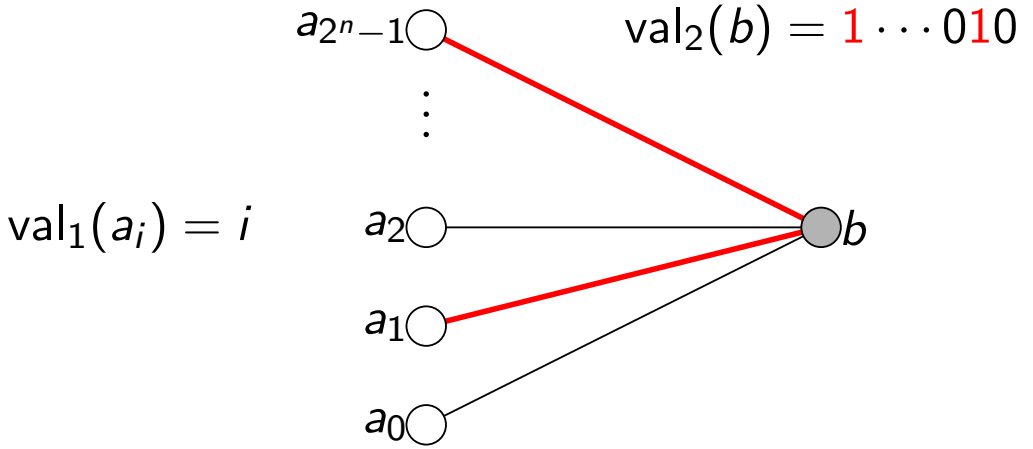
$$\mathfrak{A} \models \mathsf{zero}_1[b] \Leftrightarrow \mathsf{val}_1(b') = 0$$
$$\mathfrak{A} \models \mathsf{eq}_1[b, b'] \Leftrightarrow \mathsf{val}_1(b') = \mathsf{val}_1(b).$$

- Actually, if $\mathsf{eq}_{1,\ell}$ is a $\ell + 2$-ary predicate, we can write fluted formulas ensuring

$$\mathfrak{A} \models \mathsf{eq}_{1,\ell}[b, c_1, \ldots, c_\ell, b'] \Leftrightarrow \mathsf{val}_1(b') = \mathsf{val}_1(b).$$

- That is, we can insert semantically inert arguments into the equality predicate.

- We introduce a unary predicate int_2; any object satisfying int_2 in a model of interest will be call a 2-integer. We also introduce a binary predicate in_1 relating 1- and 2-integers.

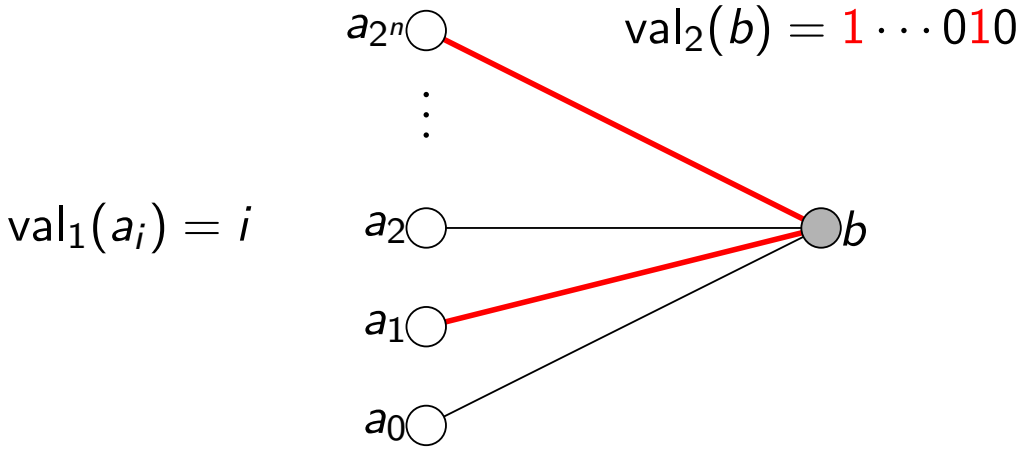


- For any 2-integer b, define $\mathsf{val}_2(b)$ to be the integer in the range $[0, 2^{2^n} - 1]$ canonically represented by the 2^n-element bit-string $s_{2^n-1}, \ldots, s_0$, where, for all i $(0 \leq i < 2^n)$,

$$s_i = \begin{cases} 1 & \text{if } \mathfrak{A} \models \mathsf{in}_1[a, b] \text{ for some 1-integer } a \text{ s.t. } \mathsf{val}_1(a) = i; \\ 0 & \text{otherwise.} \end{cases}$$

- We have to be a bit careful here: we want any 2-integer b in some structure $\mathfrak{A}$ to satisfy a 'harmony' requirement:

  If a and a' are 1-integers with $\mathrm{val}_1(a) = \mathrm{val}_1(a')$ then $\mathfrak{A} \models \mathrm{in}_1[a, b]$ iff $\mathfrak{A} \models \mathrm{in}_1[a', b]$.

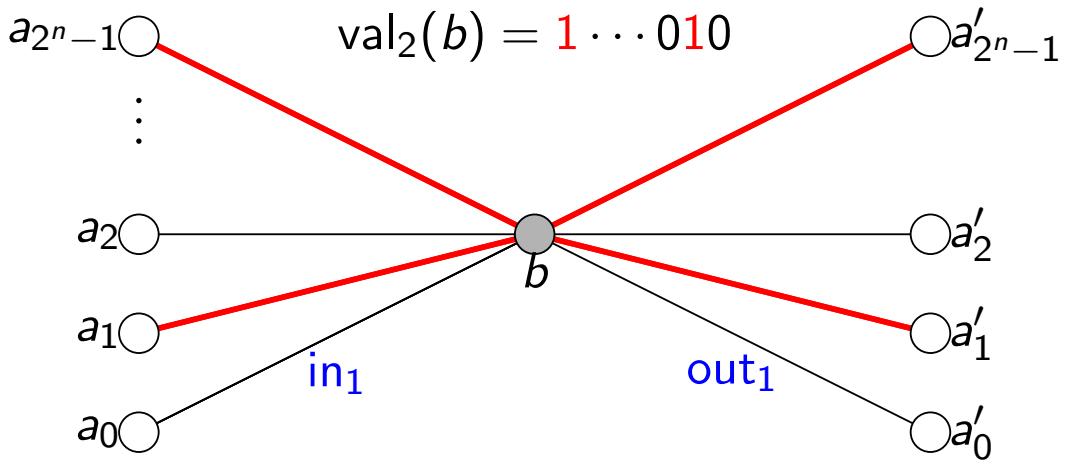


- We can do this by writing the (non-fluted) formula

$$\forall x_1 x_1' x_2 (\mathrm{eq}_1(x_1, x_1') \to (\mathrm{in}_1(x_1, x_2) \leftrightarrow \mathrm{in}_1(x_1', x_2))).$$

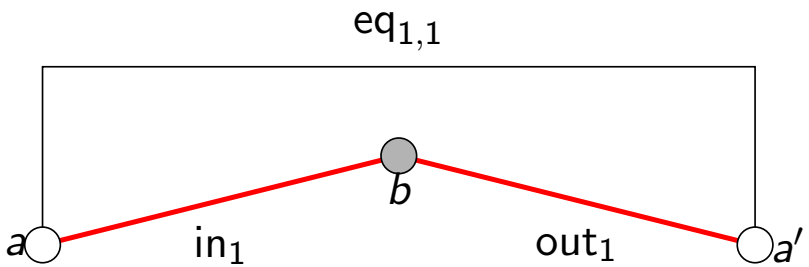
- Can we harmonize 2-integers using fluted formulas?
- Yes, provided that we represent them as 'mirrored' structures related to 1-integers via the binary predicates in_1 and out_1, satisfying the strengthened harmony requirement:

  For any 2-integer b and 1-integers a, a'
  $\mathsf{val}_1(a) = \mathsf{val}_1(a')$ implies $\mathfrak{A} \models \mathsf{in}_1[a, b] \Leftrightarrow \mathfrak{A} \models \mathsf{out}_1[b, a']$.



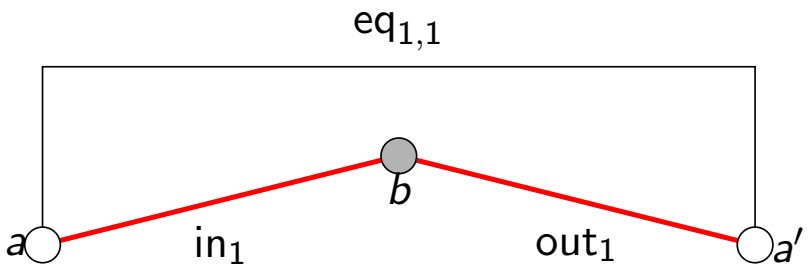
- This harmony requirement is easily enforced by fluted formulas.
- Remember $\mathsf{eq}_{1,1}(x_1, x_2, x_3)$ is to be read as $\mathsf{val}_1(x_1) = \mathsf{val}_1(x_3)$.
- We then write:

$$\forall x_1(\mathsf{int}_1(x_1) \rightarrow \forall x_2(\mathsf{int}_2(x_2) \wedge \mathsf{in}_1(x_1, x_2) \rightarrow \forall x_3(\mathsf{int}_1(x_3) \wedge \mathsf{eq}_{1,1}(x_1, x_2, x_3) \rightarrow \mathsf{out}_1(x_2, x_3)))).$$

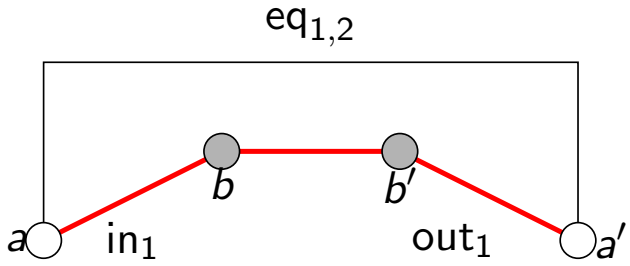


- This harmony requirement is easily enforced by fluted formulas.
- Remember $\mathsf{eq}_{1,1}(x_1, x_2, x_3)$ is to be read as $\mathsf{val}_1(x_1) = \mathsf{val}_1(x_3)$.
- We then write:

$$\forall x_1(\mathsf{int}_1(x_1) \rightarrow \forall x_2(\mathsf{int}_2(x_2) \wedge \neg \mathsf{in}_1(x_1, x_2) \rightarrow \forall x_3(\mathsf{int}_1(x_3) \wedge \mathsf{eq}_{1,1}(x_1, x_2, x_3) \rightarrow \neg \mathsf{out}_1(x_2, x_3)))).$$

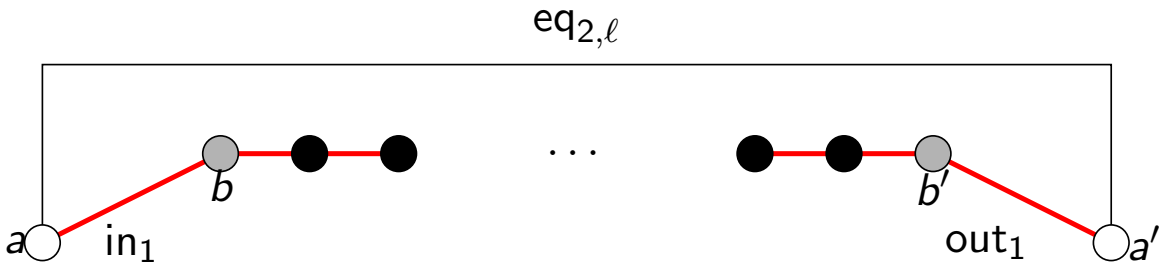


- These mirrored structures allow us to sat that the 2-integers b and b' are equal, in a particular digit (represented by a 1-integer a):



and hence we can define an equality predicate eq_2 for 2-integers. (We need four variables.)

- And of course we can insert ℓ semantically inert arguments



(for which we need $4 + \ell$ variables).

- Can we now define a decrement relation on 2-integers?
- We first introduce a binary predicate $\mathsf{in}_1^{\triangleleft}(x_1, x_2)$, and say:

  if $\mathsf{zero}_1(x_1)$ then $\mathsf{in}_1^{\triangleleft}(x_1, x_2)$;
  otherwise, if $\mathsf{pred}_1(x_1, x_1')$ then $\mathsf{in}_1^{\triangleleft}(x_1, x_2)$ iff $\mathsf{in}_1^{\triangleleft}(x_1', x_2)$ and $\neg\mathsf{in}_1(x_1', x_2)$.

- Now we can define $\mathsf{pred}_2(x_1, x_2)$, by writing a fluted formula saying:

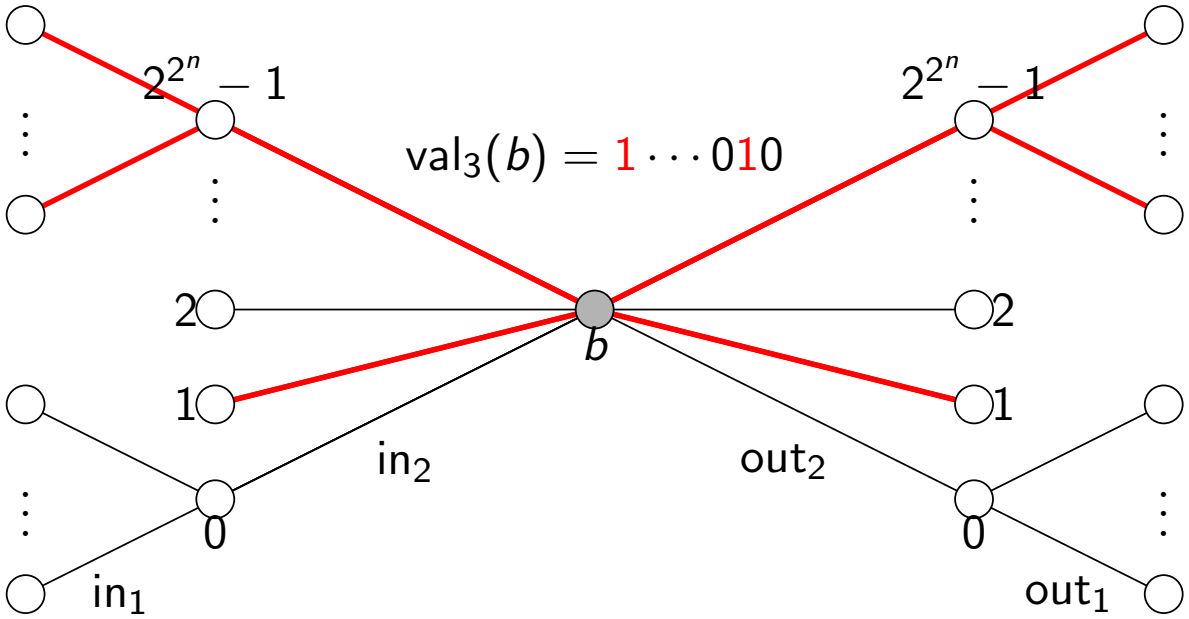
  for all 1-integers y, $\mathsf{in}_1(y, x_1)$ and $\mathsf{in}_1(y, x_2)$ differ in truth-value just in case $\mathsf{in}_1^{\triangleleft}(y, x_1)$.

- By writing

$$\exists x_1.\mathsf{int}_2(x_1) \wedge \forall x_1(\mathsf{int}_2(x_1) \rightarrow \exists x_2(\mathsf{int}_2(x_2) \wedge \mathsf{pred}_2(x_1, x_2))),$$

  we can enforce models of size $2^{2^n} = \mathfrak{t}(2, n)$.

- This process can be continued, to build 3-integers, with values up to $2^{2^{2^n}} - 1$:



- Thus, we can force models of size $2^{2^{2^n}} = \mathfrak{t}(3, n)$. But we appear to need six variables.

- In this way, we can construct, in time bounded by a polynomial function of m and n, a (finitely) satisfiable formula $\Phi_{m,n}$ of $\mathcal{FL}^{2m}$ featuring int_m, such that any model of φ has cardinality at least $\mathfrak{t}(m, n)$.
- Hence, there is no elementary bound on the sizes of models forced by satisfiable formulas of $\mathcal{FL}$.
- A simple modification of this argument shows that $\mathcal{FL}^{2m}$ is m-NExpTime-hard.
- Hence, the satisfiability problem for $\mathcal{FL}$ is not elementary.
- Purdy's (2002) bound of NExpTime is wrong.

Outline

- We employ a fluted analogue of Hintikka constituents .
- An atomic fluted m-type is a maximal consistent conjunction of fluted literals in $FL^{[m]}$, e.g.:

$$p(x_3) \wedge \neg q(x_2, x_3) \wedge \neg r(x_1, x_2, x_3)$$

  is an atomic fluted 3-type over the signature $p/1$, $q/2$, $r/3$.
- A fluted (m, m)-constituent is an atomic fluted m-type.
- Let k satisfy $m > k \geq 0$. A fluted (k, m)-constituent is a formula

$$\lambda = t(x_1, \ldots, x_k) \wedge \bigwedge_{\lambda' \in \Lambda} \exists x_{k+1}.\lambda' \wedge \forall x_{k+1} \bigvee_{\lambda' \in \Lambda} \lambda',$$

  where t is an atomic fluted k-type and Λ is some set of fluted $(k+1, m)$-constituents, and t is a atomic fluted k-type.

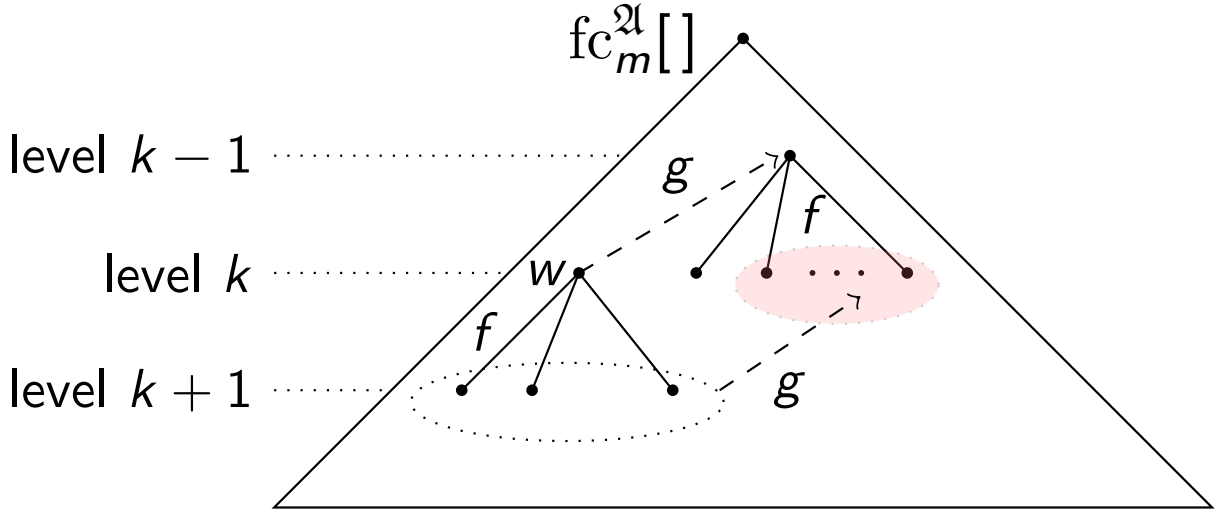
- Fluted constituents have analogous properties to Hintikka constituents.
- In particular, in any structure $\mathfrak{A}$, for any $m > 0$, there exists a unique $(0, m)$-fluted constituent true in $\mathfrak{A}$, denoted $\mathrm{fc}^{\mathfrak{A}}_m[\,]$.
- It is useful to think of a fluted constituent

$$\lambda = t(x_1, \ldots, x_k) \wedge \bigwedge_{\lambda' \in \Lambda} \exists x_{k+1}.\lambda' \wedge \forall x_{k+1} \bigvee_{\lambda' \in \Lambda} .\lambda',$$

  as a (labelled) tree in which t is the label of the root and the $\lambda' \in \Lambda$ are the top-level daughters.

- The following property of fluted constituents is a bit less obvious.
- If λ is a fluted (k, m)-constituent and $k > 0$, define $\lambda^{\leftarrow}$ to be the formula obtained by deleting all literals of λ containing x_1 and then shifting remaining variables left, i.e. replacing each variable x_{i+1} by x_i, for $i > 1$.
- Then $\lambda^{\leftarrow}$ is a fluted $(k-1, m-1)$-constituent.
- Call $\lambda^{\leftarrow}$ the pull-back of λ.

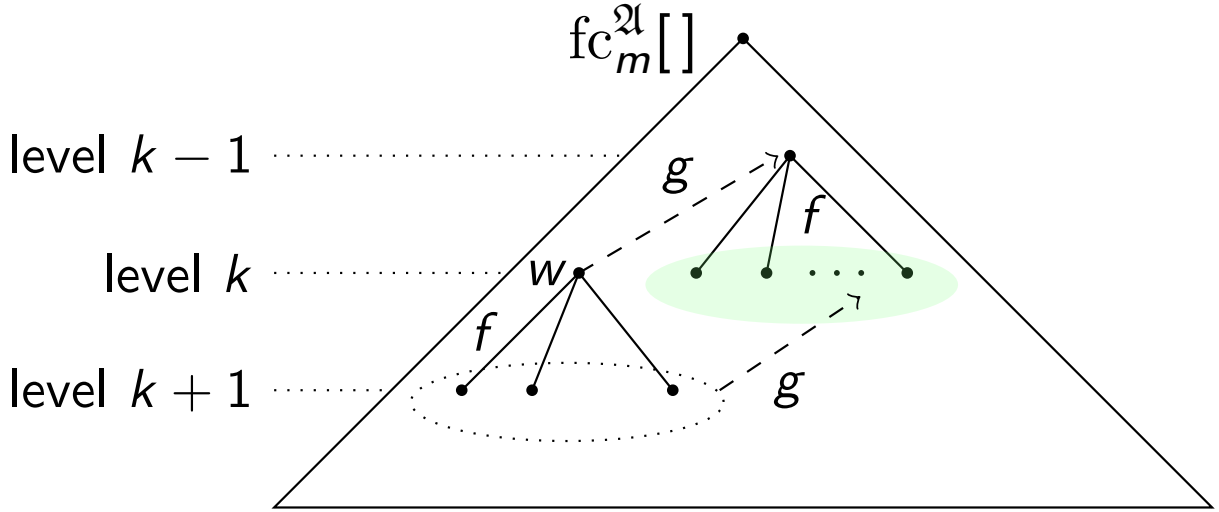
- Thus, the fluted constituent $\mathrm{fc}^{\mathfrak{A}}_m[\,]$ forms a kind of double-tree whose vertices, at level k, are labelled by realized fluted (k, m)-constituents.



- These two trees obey a confluence property:

$$g(f^{-1}(w)) \subseteq f^{-1}(g(w)).$$

- Thus, the fluted constituent $\mathrm{fc}^{\mathfrak{A}}_m[\,]$ forms a kind of double-tree whose vertices, at level k, are labelled by realized fluted (k, m)-constituents.



- These two trees obey a confluence property:

$$g(f^{-1}(w)) = f^{-1}(g(w)).$$

- By duplicating nodes, we can ensure that every non-leaf vertex is at the end of some g-arrow.

- Given any such double tree (with this property), we can generate a structure realizing exactly the given fluted (k, m)-constituents.
- The domain of this structure coinsists of the set of leaves of the tree and so is bounded by $\mathfrak{t}(m, p(\|\varphi\|))$.
- We can check from the double tree alone whether the corresponding structure satisfies φ.
- Any satisfiable formula of $\mathcal{FL}^m$ has a model of m-tuply exponential size; hence, the satisfiability problem for $\mathcal{FL}^m$ is in m-NExpTime.

Outline

- The complexity of the satisfiability problem for $\mathcal{FL}^m$ lies between

$$\lfloor m/2 \rfloor\text{-NExpTime-hard} \quad \text{and} \quad m\text{-NExpTime}.$$

- For $m \geq 3$, we can improve the upper bound to $(m-2)$-NExpTime with a little effort.
- Also, $\mathcal{FL}^1$ is in NPTime, and $\mathcal{FL}^2$ is in NExpTime.
- Thus, for $m \leq 4$, we have tight bounds of $\lfloor m/2 \rfloor$-NExpTime-complete; but still have a gap when $m \geq 5$.
- Despite considerable effort, we have not been able to close this gap.