Minimal grammars via MaxSAT

Gabriel Ebner

April 13, 2016

(joint work with Sebastian Eberhard and Stefan Hetzl)

Motivation

- Cut-introduction
  - Small grammar $\rightarrow$ small proof
- Inductive theorem proving
  - Grammar $\rightarrow$ proof

Cut-introduction

$$\dfrac{\dfrac{\dfrac{P(0) \vdash P(0) \qquad P(s0), P(s0) \to P(ss0), \ldots, P(s^8 0) \to P(ss^8 0) \vdash P(s^9 0)}{P(0), P(0) \to P(s0), \ldots, P(s^8 0) \to P(ss^8 0) \vdash P(s^9 0)}}{P(0), \forall x\,(Px \to Psx) \vdash P(s^9 0)}}{P(s^8 0), P(s^8 0) \to P(s^9 0) \vdash P(s^9 0)} \qquad \dfrac{P(s^8 0) \vdash P(s^8 0) \qquad P(s^9 0) \vdash P(s^9 0)}{P(s^8 0), P(s^8 0) \to P(s^9 0) \vdash P(s^9 0)}$$

Cut-introduction

$$\frac{\begin{array}{c}\text{(forget all the propositional stuff)}\\ P(0), P(0) \to P(s0), \ldots, P(s^8 0) \to P(ss^8 0) \vdash P(s^9 0)\end{array}}{P(0), \forall x\,(Px \to Psx) \vdash P(s^9 0)}$$

Cut-introduction

$$L = \{r_0, r_1(0), r_1(s0), r_1(s^2 0), r_1(s^3 0), \ldots, r_1(s^7 0), r_1(s^8 0), r_2\}$$

Cut-introduction

$$L = \{r_0, r_1(0), r_1(s0), r_1(s^2 0), r_1(s^3 0), \ldots, r_1(s^7 0), r_1(s^8 0), r_2\}$$

$$\tau \to r_0 \mid r_1(\alpha) \mid r_1(s\alpha) \mid r_1(s^2\alpha) \mid r_2$$
$$\alpha \to 0 \mid s^3 0 \mid s^6 0$$

Cut-introduction

$$L = \{r_0, r_1(0), r_1(s0), r_1(s^2 0), r_1(s^3 0), \ldots, r_1(s^7 0), r_1(s^8 0), r_2\}$$

$$\tau \to r_0 \mid r_1(\alpha) \mid r_1(s\alpha) \mid r_1(s^2\alpha) \mid r_2$$
$$\alpha \to 0 \mid s^3 0 \mid s^6 0$$

$$\dfrac{\dfrac{P(\alpha) \to P(s\alpha), P(s\alpha) \to P(ss\alpha), P(s^2\alpha) \to P(ss^2\alpha) \vdash \varphi(\alpha)}{\forall x\,(Px \to Psx) \vdash \forall y\,\varphi(y)} \qquad \dfrac{P(0), \varphi(0), \varphi(s^3 0), \varphi(s^6 0) \vdash P(s^9 0)}{P(0), \forall y\,\varphi(y) \vdash P(s^9 0)}}{P(0), \forall x\,(Px \to Psx) \vdash P(s^9 0)}\ \text{cut}$$

VTRATGs

- Start non-terminal: α_0
- List of non-terminal vectors:
  $\alpha_0, \quad \overline{\alpha_1} = (\alpha_{1,1}, \ldots, \alpha_{1,k_1}), \quad \ldots, \quad \overline{\alpha_n} = (\alpha_{n,1}, \ldots, \alpha_{n,k_n})$
- Productions: $\overline{\alpha_i} \to \overline{t}$
  - acyclic: $\mathrm{Vars}(t) \subseteq \{\overline{\alpha_{i+1}}, \ldots, \overline{\alpha_n}\}$ when $\overline{\alpha_i} \to t \in P$
- Generated language:

$$L(G) = \{\, \alpha_0[\alpha_0 \backslash s_0][\overline{\alpha_1} \backslash \overline{s_1}] \cdots [\overline{\alpha_n} \backslash \overline{s_n}] \mid \overline{\alpha_i} \to \overline{s_i} \in P \text{ for all } i \,\}$$

Definition (Parameterized Language Cover Problem)

Given:

- Finite set of terms L
- Non-terminal vectors N

Find: minimal grammar G with N such that $L \subseteq L(G)$

(size = number of productions)

Example

$$L = \{f(c, c, e), f(d, d, e)\}$$

Example

$$L = \{f(c, c, e), f(d, d, e)\}$$

$$\tau \rightarrow f(\alpha_1, \alpha_2, \alpha_3)$$

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} \rightarrow \begin{pmatrix} c \\ c \\ e \end{pmatrix} \left| \begin{pmatrix} d \\ d \\ e \end{pmatrix} \right.$$

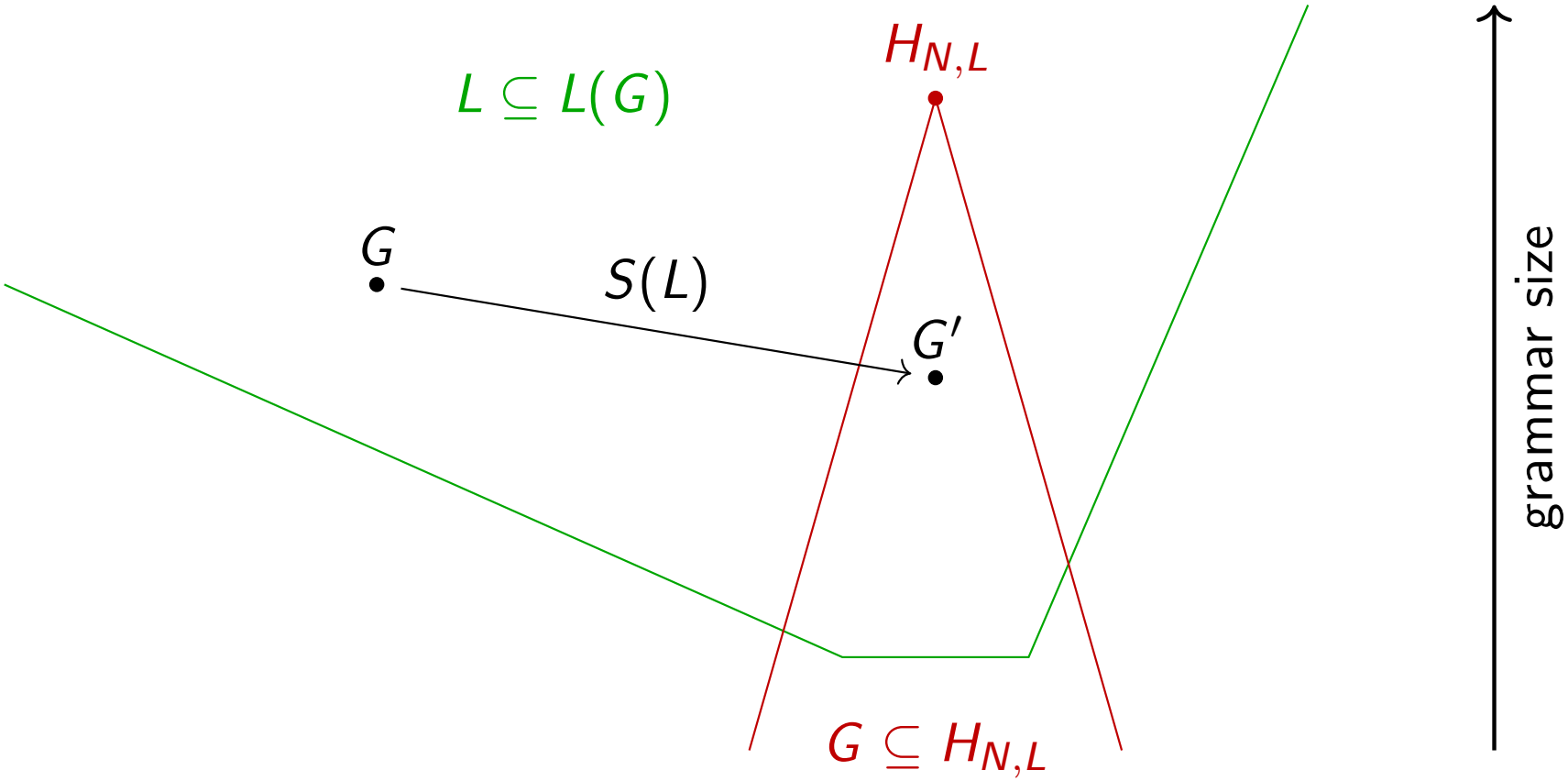
Example

$$L = \{f(c, c, e), f(d, d, e)\}$$

$$\tau \rightarrow \cancel{f(\alpha_1, \alpha_2, \alpha_3)} \quad f(\alpha_1, \alpha_1, e)$$

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} \rightarrow \begin{pmatrix} c \\ c \\ e \end{pmatrix} \Bigg| \begin{pmatrix} d \\ d \\ e \end{pmatrix}$$

Overview



$S(L)$

Definition (Stability)

Term t is stable under TRS R iff $\forall s\,(t \to_R s \;\Rightarrow\; t = s)$

$$T(R) = \{t \mid t \text{ is stable under } R\}$$

Definition (Stabilizer)

$S(L)$ is the largest TRS such that L is stable under $S(L)$:

$$S(L) = \{l \to r \mid L \text{ is stable under } \{l \to r\}\}$$

Definition ($H_{N,L}$)

$H_{N,L}$ is the VTRATG with non-terminals N that contains all productions $\overline{\alpha_i} \to (t_1, \ldots, t_k)$ such that $t_1, \ldots, t_k \in T(S(L))$.

$S(L)$

- Not confluent
- Not strongly normalizing
- Includes almost all rules
- Weakly normalizing*
  - Normalization ends in $T(S(L))$

Computing $H_{N,L}$

Definition ($H_{N,L}$)

$H_{N,L}$ is the VTRATG with non-terminals N that contains all productions $\overline{\alpha_i} \to (t_1, \ldots, t_k)$ such that $t_1, \ldots, t_k \in T(S(L))$.

- $H_{N,L}$ is finite
- $H_{N,L}$ is polynomial-time computable (for fixed N)

Anti-unification

Subsumption: $t \preceq s \quad :\Leftrightarrow \quad \exists \sigma \; t\sigma = s$

The anti-unifier is the infimum in the semi-lattice of terms ordered by $\preceq$.

$$f(\dots) \wedge g(\dots) = x_{f(\dots),g(\dots)}$$
$$f(t_1, \dots, t_n) \wedge f(s_1, \dots, s_n) = f(t_1 \wedge s_1, \dots, t_n \wedge s_n)$$

Computing $T(S(L))$

Let $t \in T(S(L))$ with n variables.

- $t \in T(S(L')), \quad |L'| \leq n+1$
- $t \preceq L', \quad L' \subseteq \mathrm{subterms(L)}$

- $t \preceq \bigwedge L'$

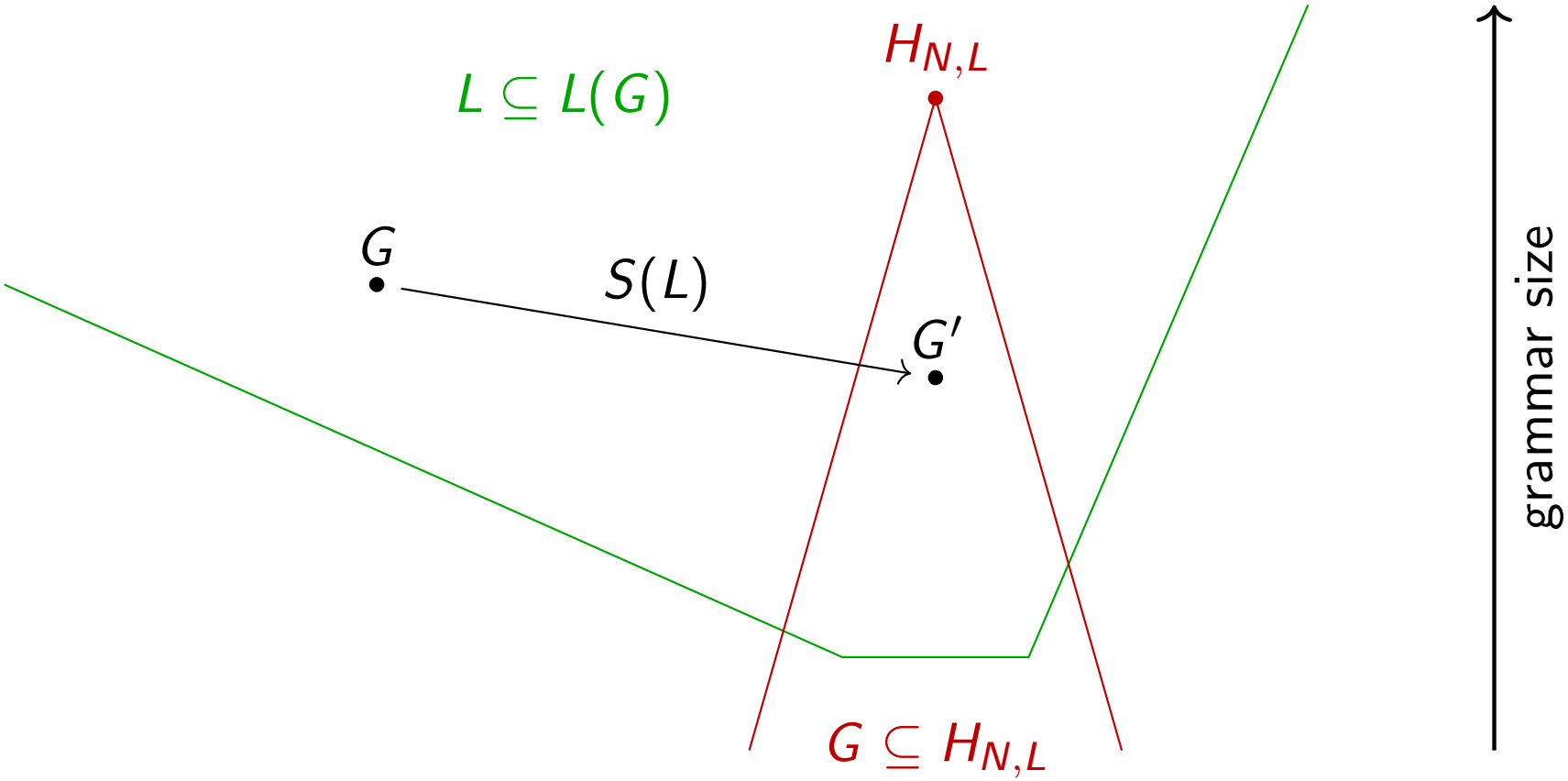
- $t\sigma = \bigwedge L', \quad \sigma$ injective

Computing $T(S(L))$

Let $t \in T(S(L))$ with n variables.

- $t \in T(S(L')), \quad |L'| \leq n+1$
- $t \preceq L', \quad L' \subseteq \mathrm{subterms(L)}$
  - pick a subset
- $t \preceq \bigwedge L'$
  - compute the anti-unifier
- $t\sigma = \bigwedge L', \quad \sigma$ injective
  - replace subterms by variables

Overview



Language & grammar rewriting

Definition ($L_1 \rightarrow_R^* L_2$)

- $\forall x_1 \in L_1 \, \exists x_2 \in L_2 \, (x_1 \rightarrow_R^* x_2)$, and
- $\forall x_2 \in L_2 \, \exists x_1 \in L_1 \, (x_1 \rightarrow_R^* x_2)$

Language & grammar rewriting

Definition ($L_1 \rightarrow_R^* L_2$)

- $\forall x_1 \in L_1 \, \exists x_2 \in L_2 \, (x_1 \rightarrow_R^* x_2)$, and
- $\forall x_2 \in L_2 \, \exists x_1 \in L_1 \, (x_1 \rightarrow_R^* x_2)$

Definition ($G_1 \rightarrow_R^* G_2$)

- $\forall (\overline{\alpha} \rightarrow \overline{t} \in P_1) \, \exists (\overline{\beta} \rightarrow \overline{s} \in P_2) \, \forall i \, (t_i \rightarrow_R^* s_i)$, and
- $\forall (\overline{\beta} \rightarrow \overline{s} \in P_2) \, \exists (\overline{\alpha} \rightarrow \overline{t} \in P_1) \, \forall i \, (t_i \rightarrow_R^* s_i)$

Language & grammar rewriting

Definition ($L_1 \rightarrow_R^* L_2$)

- $\forall x_1 \in L_1 \, \exists x_2 \in L_2 \, (x_1 \rightarrow_R^* x_2)$, and
- $\forall x_2 \in L_2 \, \exists x_1 \in L_1 \, (x_1 \rightarrow_R^* x_2)$

Definition ($G_1 \rightarrow_R^* G_2$)

- $\forall (\overline{\alpha} \rightarrow \overline{t} \in P_1) \, \exists (\overline{\beta} \rightarrow \overline{s} \in P_2) \, \forall i \, (t_i \rightarrow_R^* s_i)$, and
- $\forall (\overline{\beta} \rightarrow \overline{s} \in P_2) \, \exists (\overline{\alpha} \rightarrow \overline{t} \in P_1) \, \forall i \, (t_i \rightarrow_R^* s_i)$

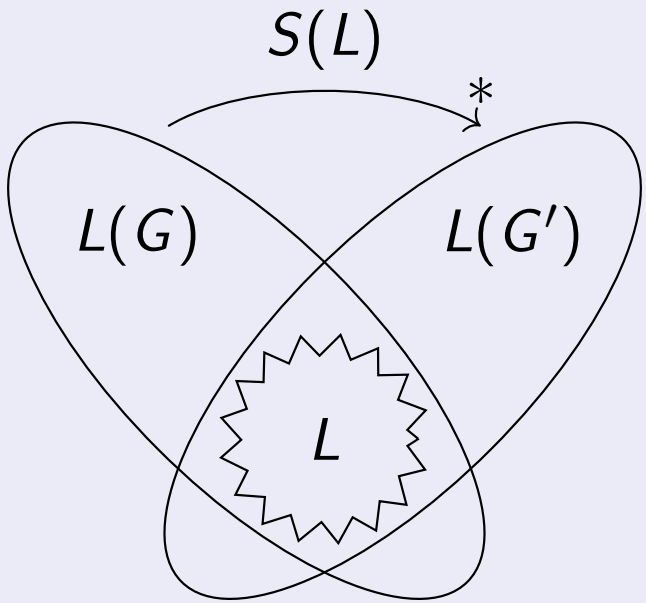
Theorem

$H \rightarrow_R^* G \quad \Rightarrow \quad L(H) \rightarrow_R^* L(G)$

Preservation of language cover

Theorem

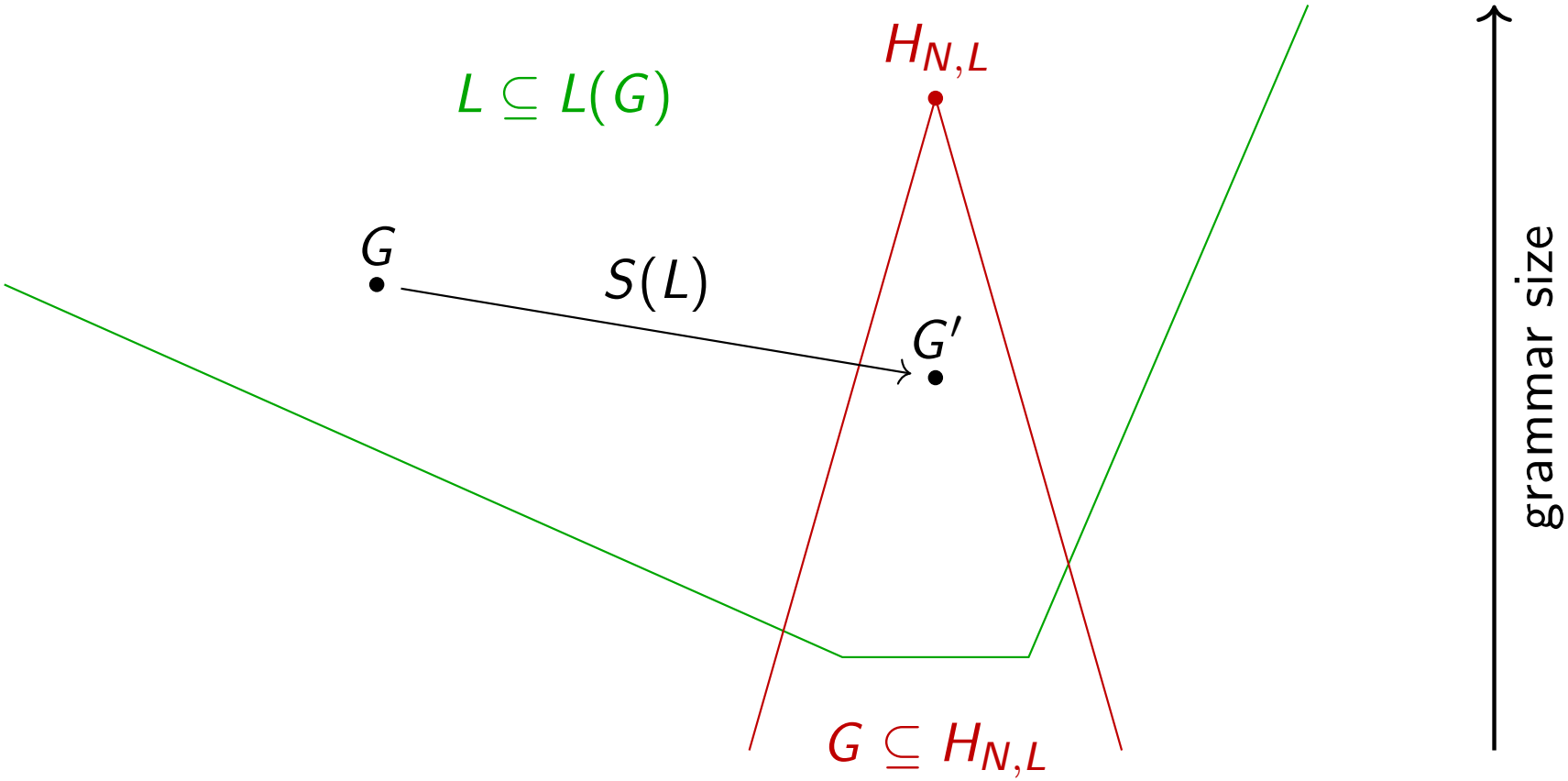
If $L \subseteq L(G)$ *and* $G \to^*_{S(L)} G'$, *then* $L \subseteq L(G')$.



Proof.

$L(G) \to^*_{S(L)} L(G')$, and $\to^*_{S(L)}$ does not change L. □

Overview



Definition (Grammar Minimization Problem)

Given:

- Finite set of terms L
- Grammar H such that $L \subseteq L(H)$

Find: minimal grammar $G \subseteq H$ such that $L \subseteq L(G)$

Definition (Partial, unweighted MaxSAT)

Given:

- "hard" clauses H
- "soft" clauses S

Find: interpretation I such that

- $I \models H$
- Number of soft clauses satisfied by I is maximal.

Definition (Partial, unweighted MaxSAT)

Given:

- "hard" clauses H
- "soft" clauses S

Find: interpretation I such that

- $I \models H$
- Number of soft clauses satisfied by I is maximal.

In our case:

- $H =$ "$L \subseteq L(G)$"
- $S = \bigwedge_{p \in P}$ "do not include production p"

Encoding

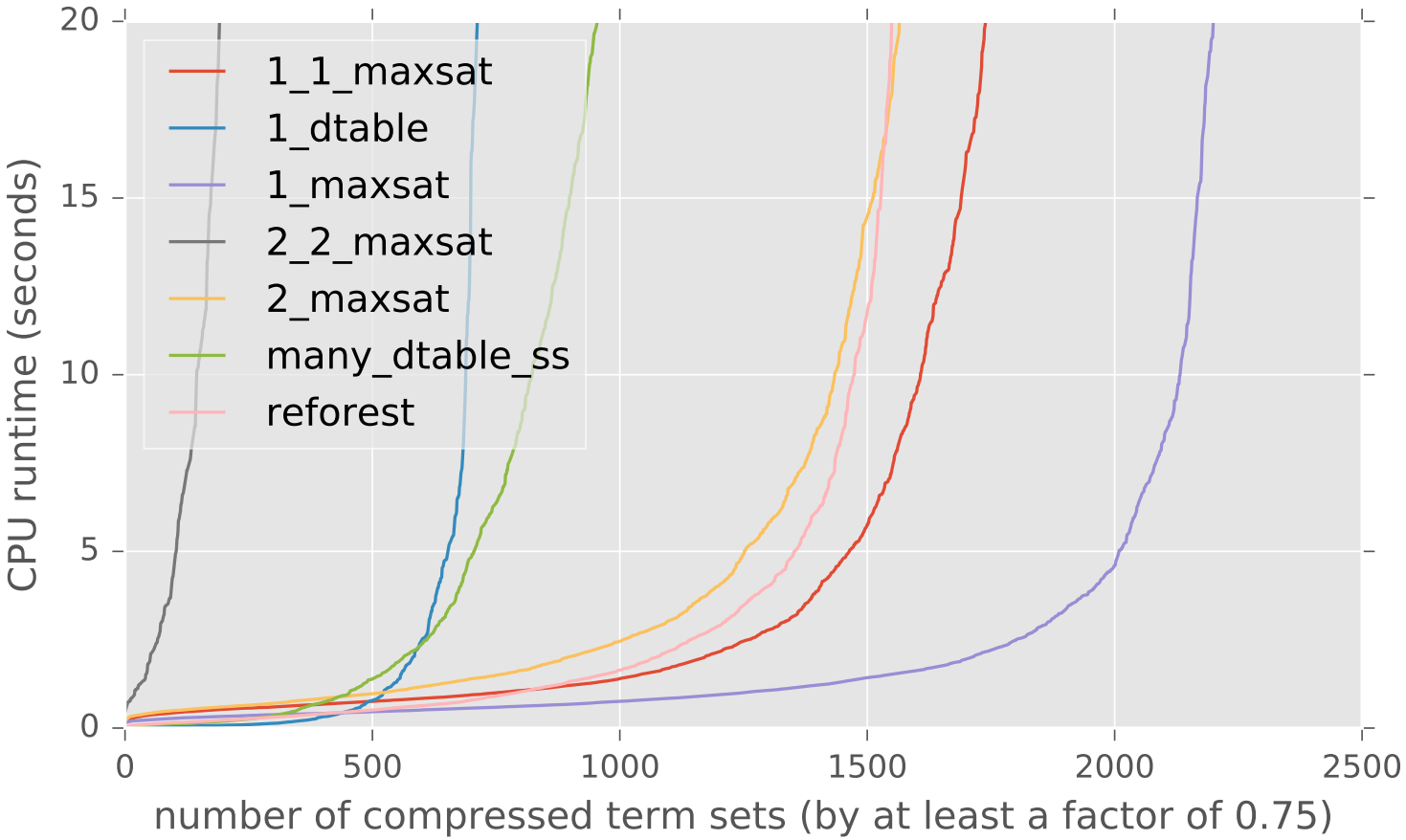
$$\text{“}L \subseteq L(G)\text{”} = \bigwedge_{t \in L} (\exists \delta_t \text{“}\delta_t \text{ is a derivation of } t \text{ in } G\text{”})$$

Two sorts of atoms:

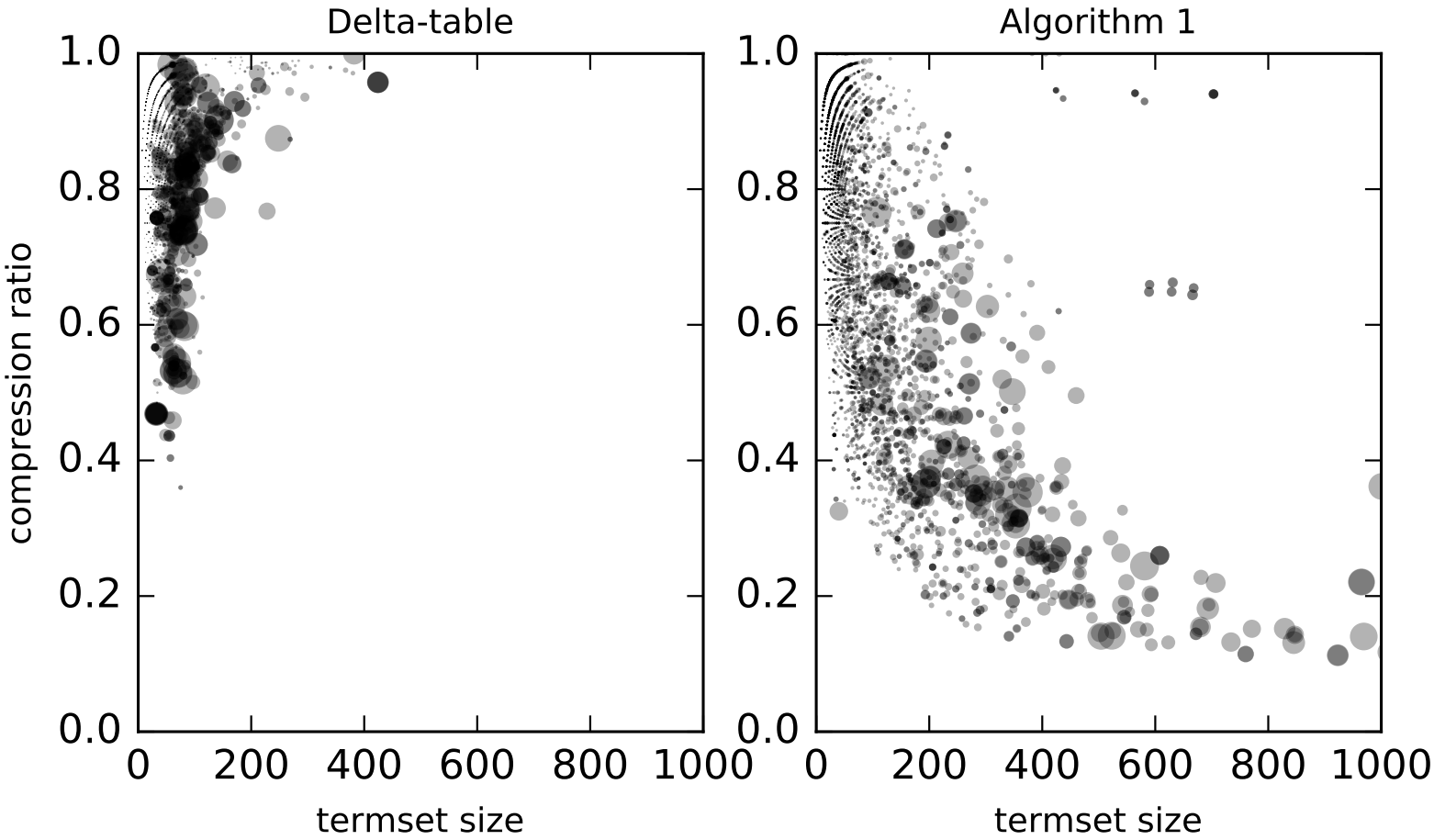
- $p \in P$ “production p is included in G”
- $\alpha_{i,j}\delta_t = r$ where $t \in L$ and r is a subterm of t.

$$\alpha_{i,j}\delta_t = \alpha_{i,j}[\tau \backslash s_0][\overline{\alpha_1} \backslash \overline{s_1}] \cdots [\overline{\alpha_n} \backslash \overline{s_n}] = r$$

Experimental results



Experimental results



New results

- Theoretical justification by rewriting with $S(L)$
- Easy extension to VTRATGs (and more)
- Reduced size of $H_{N,L}$
- MaxSAT encoding of GMP for VTRATGs

Conclusion

- Polytime reduction from PLCP to MaxSAT
  - $H_{N,L}$ contains minimal grammar as sub-grammar
  - $H_{N,L}$ is polynomial-time computable (for fixed N)
  - GMP is polynomial-time reducible to MaxSAT
- Only requires that rewriting commutes with language generation
- Future work:
  - Minimize vector productions weighted with their length
  - Minimize symbolic complexity
  - Grammars for Π_2-cuts
  - Optimal XML compression
- Open source implementation:
  `https://github.com/gapt/gapt`