Weighted Automata for Proving Termination of String Rewriting

Johannes Waldmann
Hochschule für Technik, Wirtschaft und Kultur (FH) Leipzig
waldmann@imn.htwk-leipzig.de
String rewriting

“... is what the rules of a type-0 grammar do”

- rewriting system \( R = \{ l_1 \rightarrow r_1, \ldots \} \) over \( \Sigma \)
  is set of pairs of words over \( \Sigma \)
- defines relation \( \rightarrow_R \) on \( \Sigma^* \) by \( u \rightarrow_R v \) \( \iff \exists x, y \in \Sigma^* \), \((l \rightarrow r) \in R : u = x \cdot l \cdot y, x \cdot r \cdot y = v \)

example: \( R = \{ a^2 \rightarrow bc, b^2 \rightarrow ac, c^2 \rightarrow ab \} \)
- allows derivation \( bb|aa|b \rightarrow_R b|bb|c \rightarrow_R ba|cc| \rightarrow_R b|aa|b \rightarrow_R bb|cb \rightarrow_R a|cc|b \rightarrow_R aabb \rightarrow_R \ldots \)
- is there an infinite \( \rightarrow_R \)-chain?
Problems in String Rewriting

given a finite rewrite system $R$,

- is $R$ terminating?
  there are no infinite $\rightarrow_R$ chains
- does $R$ preserve REG? ... preserve CF?

$$R^*(L) := \{v \mid u \in L, u \rightarrow^*_R v\}.$$  

$R$ preserves $\mathcal{L}$ iff $\forall L \in \mathcal{L} : R^*(L) \in \mathcal{L}$

Focus of this talk: automatic termination  
(two meanings: automatically find weighted automata that are certificates of termination)
Plan of this talk

weighted finite automata allow unified view of:

- D. H., J. W.: Termination of \( \{aa \rightarrow bc, \ bb \rightarrow ac, \ cc \rightarrow ab\} \), to appear in IPL, 2006
(Global) Compatibility

general idea: use monotone interpretation into well-founded domain

- $A$ is a $V$-weighted automaton over $\Sigma$, defines a weight function $A : \Sigma^* \rightarrow V$
- $A$ is called compatible with relation $\rightarrow$ on $\Sigma^*$ if $u \rightarrow v \Rightarrow A(u) > A(v)$.
- $(V, >)$ well-founded and $A$ compatible with $\rightarrow$ implies $\rightarrow$ is well-founded.

special plan: ensure compatibility of automaton $A$ with rewrite relation $\rightarrow_R$ by local conditions on $A$. 
Local compatibility

If \((V, >)\) is ordered semi-ring with

- \((a > b) \Rightarrow (a + c) > (b + c)\)
- \((a > b) \land (c \neq 0) \Rightarrow (a \cdot c) > (b \cdot c)\)

and \(A\) over \(\Sigma\) (states \(Q\) with \(i\) initial, \(f\) final) is locally compatible with \(R\):

- \(\forall x \in \Sigma : A(i, x, i) > 0 \land A(f, x, f) > 0\)
- \(\forall p, q \in Q, (l \rightarrow r) \in R : A(p, l, q) \geq A(p, r, q)\)
- \(\forall(l \rightarrow r) \in R : A(i, l, f) > A(i, r, f)\)

then \(A\) is (globally) compatible with \(\rightarrow_R\).
Example (1)

\[ R = \{ aa \rightarrow bc, bb \rightarrow ac, cc \rightarrow ab \}, \Sigma = \{ a, b, c \} \]

\[ V = (\mathbb{N}, +, \cdot, 0, 1) \text{ and standard ordering } > \]

\[ A(i, aa, f) = 2 > 1 = A(i, bc, f) \]
\[ A(r, bb, r) = 4 \geq 4 = A(r, ac, r). \]
How to find such automata

- fix number $d$ of states, say 5. Automaton is mapping $t : \Sigma \rightarrow \mathbb{N}^{d \times d}$
- local compatibility $\Rightarrow$ constraint system with $|\Sigma| \cdot d^2$ unknowns and $|R| \cdot d^2$ constraints
- fix maximal value for entries, say 7. $\Rightarrow$ finite domain constraint system
- represent unknowns in binary $\Rightarrow$ boolean satisfiability problem, (15,000 variables, 90,000 clauses, 300,000 literals) $\Rightarrow$ solve by SAT solver (SateliteGTI) (takes 7 seconds)
Example (2)

standard test case for automated termination

\[ R = \{ a^2b^2 \rightarrow b^3a^3 \}, \Sigma = \{ a, b \} \]

\[ V = (\mathbb{N}, +, \cdot, 0, 1) \] and standard ordering $>$

\[ A(i, a^2b^2, f) = 1 > 0 = A(i, b^3a^3, f) \]

\[ A(q, a^2b^2, p) = 4 \geq 1 = A(q, b^3a^3, p). \]
Summary (so far)

- new automated termination method for string rewriting with powerful implementation
  (can solve problems that no other method can)
- developed in joint work with Dieter Hofbauer
- generalized to term rewriting in joint work with Jörg Endrullis and Hans Zantema
- can’t handle more than 5 states well via SAT solver, more synthetic construction of automata
  (matrices) would be much desirable

Part 2: we show a variant of this method where we already have a synthetic construction
A Multi-Set Semi-Ring

Idea: given $V$-weighted automaton $A$ over $\Sigma$.

- For a path in $A$ labelled $(w_1/v_1)(w_2/v_2)\ldots$, consider multi-set of weights $\{v_1, v_2, \ldots\}$.
- For a word $w$ over $\Sigma$, consider lowest weight-multi-set of paths with $w = w_1w_2\ldots$

$M(V) = \top \cup \mathbb{N}^V$ (with finite support) is semi-ring:

- $0 := \top$, $1 := \emptyset$
- $A + B := \min>(A, B)$ (multiset extension of $>$)
- $A \cdot B := A \cup B$ (adding weights), $A \cdot 0 := 0$
Multi-set ordering

given \((V, >)\), define \(\gg\) on \(V\)-multi-sets as \(\gg_1^+\) for
\[(x > y_1 \land \ldots \land x > y_n) \Rightarrow (A \setminus x \gg_1 B \cup \{y_1, \ldots, y_n\})\]

- if \(>\) total, then \(\gg\) total
- if \(>\) well-founded, then \(\gg\) well-founded

\((\mathcal{M}(V), \gg)\) is ordered semi-ring (make \(\top\) maximal)
An alternative picture

... of this ordered semi-ring of multi-sets:

- domain is $\mathbb{N}^*$ (but no leading 0): multiplicities, starting with largest element
  for $V = \{a > b > c > d\}$,
  $\{a, c, c\} \mapsto 1020$ and $\{b, c, d\} \mapsto 111$.

- ordering is length-lexicographic: $1020 > 111$

- multiplication is point-wise addition, right-aligned: $1020 \cdot 111 = 1131$

- addition is minimum w.r.t. ordering
  $1020 + 111 = 111$
A $\mathcal{M}(V)$-weighted Automaton

\[ A(\text{aba}) = \begin{pmatrix} 
\{1, 0, 1\} & \{1, 0, 0\} & \top \\
\top & \top & \top \\
\{0, 0, 1\} & \{0, 0, 0\} & \top 
\end{pmatrix} \]

\[ A(aa) = \begin{pmatrix} 
\{2, 2\} & \{2, 1\} & \top \\
\top & \top & \top \\
\{1, 2\} & \{1, 1\} & \top 
\end{pmatrix} \]

For $A(1, \text{aba}, 1)$ note $\{2, 2, 2\} \gg \{1, 0, 1\}$ etc.
Compatibility

for $\mathbb{M}(V)$, we have

- $(a \gg b) \land (c \neq 0) \Rightarrow (a \cdot c) \gg (b \cdot c)$

we do not have

- $(a \gg b) \Rightarrow (a + c) \gg (b + c)$

instead, will use

- $(a \gg b) \land (c \gg d) \Rightarrow (a + c) \gg (b + d)$

to infer global compatibility (of a $\mathbb{M}(V)$-automaton with $\rightarrow_R$), need something sharper than local compatibility.
Strict local compatibility

If \((V, >)\) is ordered semi-ring with

- \((a > b) \land (c > d) \Rightarrow (a + c) > (b + d)\)
- \((a > b) \land (c \neq 0) \Rightarrow (a \cdot c) > (b \cdot c)\)

and \(A\) over \(\Sigma\) (states \(\mathcal{Q}\) with \(i\) initial and final) is strictly locally compatible with \(R\):

- \(\forall x \in \Sigma : A(i, x, i) \neq 0\)
- \(\forall p, q \in \mathcal{Q}, (l \rightarrow r) \in R :\)
  \(A(p, l, q) = 0 \lor A(p, l, q) > A(p, r, q)\)

then \(A\) is (globally) compatible with \(\rightarrow_R\).
Given \((V, >)\), consider \(V' = V \cup \{-\infty, +\infty\}\) and semi-ring \((V', -\infty, +\infty, \min_>, \max_>)\).

\(\text{flat} : \mathcal{M}(V) \to V' : B \mapsto \max B, \top \mapsto +\infty\)
is a morphism of ordered semi-rings.

- \((\text{flat } B > \text{flat } C) \Rightarrow (B \gg C)\) (but not “\(\Leftarrow\)"
- \((\text{flat } B \geq \text{flat } C) \iff (B \gg C)\)

...will use the stronger ordering via \(\text{flat}\)
Strict “flat” compatibility

If the \((V', >)\)-weighted automaton \(A\) is strictly locally compatible with \(R\), then its “lifted” \((\mathcal{M}(V), \gg)\)-weighted automaton is compatible with \(\longrightarrow_R\) (... but \(A\) itself is not)

\[
\text{flat } A(aa) = \begin{pmatrix}
2 & 2 & + \\
+ & + & + \\
2 & 1 & +
\end{pmatrix} \quad \text{flat } A(aba) = \begin{pmatrix}
1 & 1 & + \\
+ & + & + \\
1 & 0 & +
\end{pmatrix}
\]

this is the concept of match-boundedness.
Match-Bounded Rewriting

Annotate letters by numbers ("match heights").
In each rewrite step $x \cdot l \cdot y \rightarrow_R x \cdot r \cdot y$,
  
  • annotate each letter in $r$
  by $(1 + \text{minimal annotation in } l)$.

Example $R = \{aa \rightarrow aba\}, a_2a_3a_0 \rightarrow a_2a_1b_1a_1$

If heights (starting from 0) are bounded, then
  
  • $R$ is terminating
  • $R$ effectively preserves REG
  • $R$ has certificate automaton (see prev. slide!)
  • $R^-$ effectively preserves CF
two termination methods using weighted automata:

- weights in \((\mathbb{N}, +, \cdot)\): “matrix method”, automata are “guessed” (finite domain constraint system)
- weights in \((\mathbb{N}, \text{min}, \text{max})\): match bounds, (huge) automata can be efficiently constructed

Questions:

- efficient construction of \((\mathbb{N}, +, \cdot)\) automata?
- existence of \((\mathbb{N}, \text{min}, \text{max})\) automaton \(\Rightarrow\) existence of \((\mathbb{N}, +, \cdot)\) automaton?
- other semi-rings for termination?