Semantic Kachinuki Order

Alfons Geser, Dieter Hofbauer,
Johannes Waldmann

Workshop on Termination 2018

An Example Termination Proof

- $R = \{ca \to b^2c^3, bcb \to a\}$
- split at occurrences of a and b, obtain words of syllables
- $\text{split}_{a,b}(R) = \{[c, \epsilon] \to [\epsilon, \epsilon, ccc], [\epsilon, c, \epsilon] \to [\epsilon, \epsilon]\}$
  is decreasing w.r.t. lexicographic extension of the length order on syllables
  but number of syllables may increase
- weight function $w : a \mapsto 2, b \mapsto 1, c \mapsto 0$ is non-decreasing, and $w(a) > 0$ and $w(b) > 0$
- bounds number of syllables, this implies termination

Kachinuki Order (as published)

- **Def:** total precedence $(\Sigma, >)$, $u >_K^\Sigma v$ (on Σ^*): let a be the highest letter in Σ, then compare $\text{split}_a(u)$ to $\text{split}_a(v)$ w.r.t. the length-lexicographic extension of $>_K^{\Sigma \setminus \{a\}}$
- for $R = \{cba \to bcac\}$ use $a > b > c$,
  $\text{split}_a(R) = [cb, \epsilon] \to [bc, c]$,
  $\text{split}_b(\text{split}_a(R)) = [[c, \epsilon], [\epsilon]] \to [[\epsilon, c], [c]]$
- Kachinuki Order equals RPO on reversed strings
- Ko Sakai: Knuth-Bendix Algorithm for Thue System Based on Kachinuki Ordering, 1984.
- Joachim Steinbach: Comparing on Strings: Iterated syllable ordering and RPO, 1989.

Kachinuki Order (folklore extension)

- assign status (*left* or *right*) to each symbol;
  status determines direction of lex. extension.
- **Ex:** For $R = \{bca \to ccbab\}$,
  use precedence and status $(a, \text{Left}) > (b, \text{Right}) > (c, \text{Left})$,
  $\text{split}_a(R) = [bc, \epsilon] \to [ccb, b]$,
  $\text{split}_c(\text{split}_a(R)) = [[\epsilon, c], \epsilon] \to [[cc, \epsilon], [\epsilon, \epsilon]]$
- this shows that status increases power

Γ-Quasi-Termination

- Goal: use syllable decomposition, but allow increasing number of syllables during rewriting, if it is bounded
- **Def:** For $\Gamma \subseteq \Sigma$, a relation $\to$ on Σ^* is called Γ-*quasi-terminating*, if for each x, there is a bound on the number of occurrences of letters from Γ in words reachable from x
- Note: Σ-quasi-termination (i.e., on the full alphabet) *is* quasi-termination (Nachum Dershowitz 1987)
- **Theorem:** For R on Σ and $\Gamma \subseteq \Sigma$:
  if $\to_R$ is Γ-quasi-terminating
  and there is a well-founded reduction order $>$ on $(\Sigma \setminus \Gamma)^*$
  such that $\text{split}_\Gamma(R)$ is included in $>^{\text{L}}_{\text{Lex}}$ or in $>^{\text{R}}_{\text{Lex}}$,
  then R is terminating.

Interpretations that Prove Γ-Quasi-Termination

- **Def:** uniformly well-founded domain $(D, >)$: for each x, the length of $>$-chains starting at x is bounded
- **Def:** interpretation $i : \Sigma \to (D \to D)$ is *weakly simple*:
  $\forall a, x : i_a(x) \geq x$
- **Def:** letter a is *strong*: $\forall x : i_a(x) > x$.
- **Lemma:** If i from Σ into uniformly well-founded $(D, \geq)$ is weakly simple and compatible with $\to$, and $\Gamma \subseteq \Sigma$ is strong, then $\to$ is Γ-quasi-terminating.
- **Ex:** $i_a(x) = x + 2, i_b(x) = x + 1, i_c(x) = x$
  $\{ca \to b^2c^3, bcb \to a\}$ is $\{a, b\}$-quasi-terminating.

An Example

- $R = \{cab \to bcba, bcba \to abc\}$,
- $i_a = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}, i_b = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, i_c = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$,
- a not strong on $\mathbb{N}^2$ since $i_a((0, 0)^T) = (0, 0)^T$
- use $D = \mathbb{N}^2_+$ with $x > y$ if $x_1 > y_1$,
- then $\Gamma = \{a, b\}$ strong, then R is Γ-quasi-terminating
- $\text{split}_\Gamma(R) =$
  $\{[c, \epsilon, \epsilon] \to [\epsilon, c, \epsilon, \epsilon], [\epsilon, c, \epsilon, \epsilon] \to [\epsilon, \epsilon, c]\}$
  is included in $>^{\text{L}}_{\text{Lex}}$ where $>$ compares lengths
- By Lemma and Theorem, R is terminating

Implementation

- prototypical implementation as part of `https://gitlab.imn.htwk-leipzig.de/waldmann/pure-matchbox`
- 1. find strong letters $\emptyset \neq \Gamma \subseteq \Sigma$ (Lemma) for i into certain sets of matrices
  2. find $>$ on $\Sigma \setminus \Gamma$ s.t. $\text{split}_\Gamma(R)$ is in $>^{\text{L}}_{\text{Lex}}$ or in $>^{\text{R}}_{\text{Lex}}$ (Theorem)
     2.1 using standard matrix interpretations
     2.2 or using this method recursively
- solve constraints 1 and 2 separately $\Rightarrow$ tree search (over candidates of Γ)
- obvious modifications for relative termination (“removing rules”)

The “Killer” Example (well, sort of)

- `SRS/Zantema06/rel09` not solved in TC 2014–2018

  ```
  a d -> d b, a -> b b b, d -> , a -> ,
  b c -> c d d, a c -> b b c d,
  b d b ->= a d, a d ->= b d b
  ```
- use Kachinuki order with strong `c`, lex. from left,
  interp. in syllables: $a(x) = x + 4, b(x) = x + 1, d(x) = 3x$
  but — before you get all excited about “progress” here
- `YES`
  ```
  TORPA 1.6 is applied to ...
  [L] Split on c, resulting in:
     .. , b c  -> c , a c  -> b b c , ..
  ```
  Hans Zantema (2006?) `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.97.8035`

It Terminates Since It Always Grows

- Lemma: If $\to_R$ is Γ-quasi-terminating,
  and for each $(l, r) \in \text{split}_\Gamma(R)$ we have $|l| < |r|$,
  then R is terminating.
- i.e., length of syllables is bounded (by quasi-termination)
  and strictly increasing
- **Ex:** (SRS/Zantema/z050) *abbaab* $\to$ *aabbaba*,

  $$i_a = \begin{pmatrix} 1 & -\infty & 1 \\ -\infty & -\infty & 1 \\ 0 & -\infty & -\infty \end{pmatrix}, i_b = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 2 & 4 \\ 1 & 0 & -\infty \end{pmatrix},$$

  on $D = \mathbb{N} \times \mathbb{A} \times \mathbb{A}$ with $x > y$ iff $x_1 > y_1$
  and a strong, b weak

Summary, Discussion

- our method:
  - find weakly compatible interpretation
    to prove Γ-quasi-termination
  - find $>$ such that $\text{split}_\Gamma(R)$ in $>^{\text{L}}_{\text{Lex}}$ or in $>^{\text{R}}_{\text{Lex}}$
- includes Kachinuki order on strings (with status)
- might be related to semantic path order, or semantic labelling with a quasi-model (on terms)
  but these could not handle status
- performance (predictions):
  more powerful after labelling (larger alphabet),
  but then have larger constraints