The Parikh image of languages and linear constraints

Peter.Habermehl@liafa.univ-paris-diderot.fr[1]

[1]LIAFA, Université Paris Diderot, Sorbonne Paris Cité, CNRS

CP meets CAV, Turunç

June 28th, 2012

Overview

- Parikh image
- The Parikh image of the language of a finite-state automaton
- Some applications

The Parikh image of a language

- Let $\Sigma = \{a_1, \ldots, a_n\}$. Let $L \subseteq \Sigma^*$ be a language.
- The *Parikh image* of any $w \in \Sigma^*$ is defined as $\sigma(w) = (x_1, \ldots, x_n)$ such that $x_i = w|_{a_i}$ for all $i \in \{1, \ldots, n\}$.
- The *Parikh image* $\sigma(L)$ of L is defined as $\{\sigma(w) \mid w \in L\}$.
- Examples:
  - $\sigma((ab)^*) = \{(x_1, x_2) \mid x_1 = x_2\}$
  - $\sigma(\{a^n b^n \mid n \geq 0\}) = \{(x_1, x_2) \mid x_1 = x_2\}$
  - $\sigma(\{a^n b^n c^n \mid n \geq 0\}) = \{(x_1, x_2, x_3) \mid x_1 = x_2 = x_3\}$
  - $\sigma((aa)^*) = \{(x_1) \mid x_1 \text{ is divisible by } 2\} = \{\exists k.k \geq 0 \land 2 * k = x_1\}$
  - etc.

Parikh's theorem, Presburger arithmetic and semilinear sets

Theorem 1 (Parikh JACM 66)

Every context-free language L has a Parikh image definable by a formula of Presburger arithmetic.

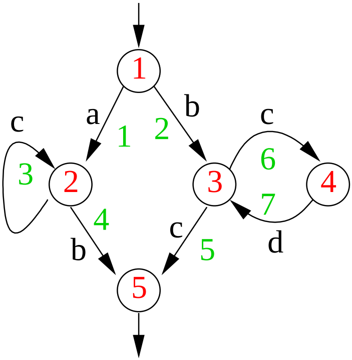
- Presburger arithmetic: first-order logic over integers with addition and equality
- corresponds to quantifier free formulae with linear ($\vec{x}\vec{a} \leq d$) and modulo constraints ($\vec{a}\vec{x} \equiv_c d$)
- corresponds to semilinear sets
  - A subset of $\mathbb{N}^n$ is called *linear* if it can be written as (for some $m \geq 0$)

$$\vec{v_0} + \mathbb{N}\vec{v_1} + \ldots + \mathbb{N}\vec{v_m}$$

    ($\vec{v_0}$ is the *base* vector and $\vec{v_i}$ the period vectors)
  - A subset of $\mathbb{N}^n$ is called *semilinear* if it is a finite union of linear sets.

The Parikh image of an automaton

- Let $A = (Q, \Sigma, \delta, q_0, F)$ be an automaton
- We will give an existential Presburger formula φ_A defining the Parikh image of $L(A)$ whose size is linear in the size of A
  [Seidl et al. ICALP 2004]
- Example:



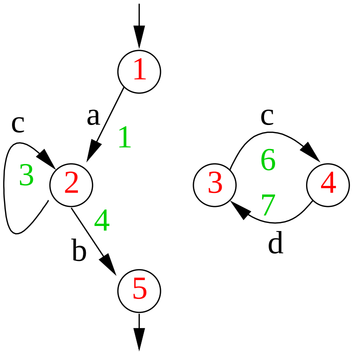
Consistent flow

- A *flow* of $A = (Q, \Sigma, \delta, q_0, \{q_f\})$ is a function F which maps triples (p, a, q) with $q \in \delta(p, a)$ to natural numbers. We write

$$in_F(q) = \sum_{\substack{p \in Q, a \in \Sigma \\ q \in \delta(p,a)}} F(p, a, q) \quad \text{and} \quad out_F(p) = \sum_{\substack{p \in Q, a \in \Sigma \\ q \in \delta(p,a)}} F(p, a, q)$$

- A flow F is *consistent* if, for each $p \in Q$, one of the following holds
  - $in_F(p) = out_F(p)$
  - $p = q_0$ and $1 + in_F(p) = out_F(p)$
  - $p = q_f$ and $in_F(p) = out_F(p) + 1$

Connectedness



$t_1 = 1, t_3 = 5, t_4 = 1, t_6 = 3, t_7 = 3$ is a consistent flow. Therefore consistency is not enough.

- A state p *occurs* in F if $p \in \{q_0, q_f\}$ or $in_F(p) > 0$
- A flow is *connected* if the directed graph G which has the occurring states as vertices and has edges $\{(p, q) \mid F(p, a, q) > 0$, for some $a \in \Sigma\}$ is connected.

The Parikh image of an automaton

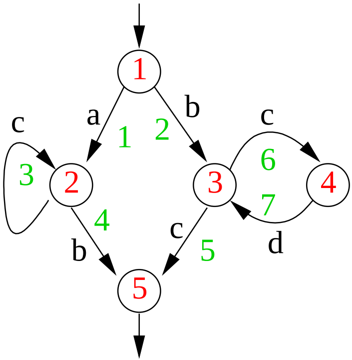
Lemma 2

A vector $(x_1, \ldots, x_n)$ is in the Parikh image of A iff there is a consistent and connected flow F such that

- *for each $a_i \in \Sigma$, $x_i = \displaystyle\sum_{p,q \in \delta(p,a)} F(p,a,q)$*

- We can construct a formula φ'_A with free variables $t_{(p,a,q)}$ where $p, q \in Q$,$a \in \Sigma$ and $q \in \delta(p, a)$ which characterizes all consistent and connected flows.
- φ'_A is a conjunction of ψ_A and ϕ_A where ψ_A corresponds to all consistent flows and ϕ_A checks that they are connected.
- ψ_A is easy to give

Example



state 1: $1 = t_1 + t_2$
state 2: $t_1 + t_3 = t_3 + t_4$
state 3: $t_2 + t_7 = t_6 + t_5$
state 4: $t_6 = t_7$
state 5: $t_4 + t_5 = 1$

What about connectedness ?

- One could give constraints saying that for each transition taken, there is a path to it composed of transitions taken.
  $\Rightarrow$ exponential
- The graph G is connected iff we can label each node of G by a natural number such that
  - The initial state q_0 gets 0
  - Each other node gets a number > 0
  - Each node of G different from q_0 has a neighbour in G with a smaller number
- We can give a linear size formula ϕ_A for that
- Finally, φ_A is given as

$$\exists (t_{p,a,q})_{q \in \delta(p,a)} \phi_A \wedge \psi_A \wedge \bigwedge_{a_i \in \Sigma} x_i = \sum_{p,q} t_{p,a,q}$$

Computing the Parikh image using semilinear sets I

- Fix an automaton A with alphabet $\Sigma = \{a_1, \ldots, a_n\}$
- Each transition with letter a_i of an automaton corresponds to a vector $\vec{v} = (v_1, \ldots, v_n)$ where $n = |\Sigma|$ and $v_j = 0$ for $j \neq i$ and $v_i = 1$
- One can define generalized transitions obtained by concatenation, union and the star operator (regular expressions)
- Instead of computing a regular expression equivalent to A (this is a standard algorithm) one can compute a representation of the Parikh image of A by replacing concatenation, union and star by the corresponding operations on sets of Parikh images.
  - for example concatenation corresponds to addition
    $(aab(b^* + (aabbb)^*).ab(bbb)^*)$
    $((2,1) + \mathbb{N}(0,1) + \mathbb{N}(2,3)) \oplus ((1,1) + \mathbb{N}(0,3)) =$
    $(3,2) + \mathbb{N}(0,1) + \mathbb{N}(2,3) + \mathbb{N}(0,3)$

Computing the Parikh image using semilinear sets II

- Fix an automaton A with k states and alphabet $\Sigma = \{a_1, \ldots, a_n\}$.
- Let $||\vec{v}||_\infty$ be the sum of all components of $\vec{v}$.

Lemma 3 (Xie, Ling, Dang, CIAA 03)

The Parikh image of A is a union of linear sets Q_i. Each Q_i is of the form $\vec{v_0} + \mathbb{N}\vec{v_1} + \ldots + \mathbb{N}\vec{v_m}$ where

- $||\vec{v_0}||_\infty \leq k^2$
- $||\vec{v_j}||_\infty \leq k$ *for* $1 \leq j \leq m$
- $m \leq k^n$

see also [Kopczynski, Widjaja To, LICS 2010]

Context-free grammars

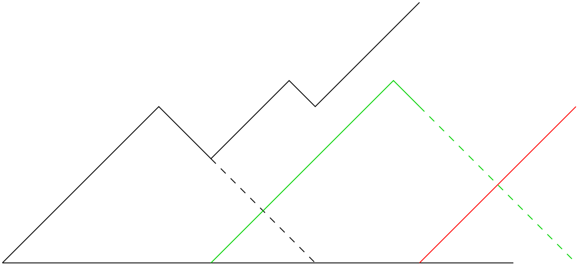
- or tree-automata
- The construction of a PA formula can be easily generalized [Verma et al., CADE 2005]
- Example:
  $1 : S \rightarrow AB, 2 : S \rightarrow BC$
  $3 : A \rightarrow DAAA, 4 : B \rightarrow a$
  $5 : D \rightarrow b, 6 : C \rightarrow CC, 7 : C \rightarrow c$
  - One variable for each production
  - One constraint for each non-terminal
    $S : t_1 + t_2 = 1$, $A : t_3 = 3 * t_3 + t_1$, $B : t_4 = t_1 + t_2$
    $C : t_6 + t_7 = t_2 + 2 * t_6$, $D : t_5 = t_3$
  - Plus connectedness
- One can construct from a CFG an automaton with the same Parikh image [Esparza et al. IPL 11]

Applications

- Reversal bounded counter automata
- Constraint automata
- Combining theories with BAPA
  - WS1S $\rightarrow$ automata $\rightarrow$ Parikh image
- Several works on verification of concurrent systems
- etc.

Reversal bounded counter automata [Ibarra JACM 78]

- An RBCA is an automaton A_R equipped with n counters
  - Counters can be incremented, decremented and tested for 0
- Only runs of the automaton where the number of reversals between increasing and decreasing of the counter is bounded by a fixed constant k are taken into account
- k can be reduced to 1 by adding additional counters



Reachability of an RBCA is decidable

- Reachability of an RBCA A_R is decidable
  - Construct finite-state automaton A'_R from A_R by replacing
    - ★ increments of counter i by inc_i
    - ★ decrements of counter i by dec_i
    - ★ A'_R has alphabet $\{inc_1, dec_1, \ldots, inc_n, dec_n\}$
    - ★ guess when each counter is 0
  - check that $\sigma(A'_R) \cap \{(x_1, x_2, \ldots, x_{2n-1}, x_{2n}) \mid x_1 = x_2 \wedge \cdots \wedge x_{2n-1} = x_{2n}\}$ is not empty

Constraint automata

- There are lots of variations of the basic theme.
- A CA (A, φ) is a finite-state automaton A together with a Presburger formula $\varphi(x_1, \ldots, x_n)$.
- φ constrains the number of times letters of A appear.
- $w \in L((A, \varphi))$ iff $w \in L(A)$ and $\sigma(w) \models \varphi$
  - can accept languages like $\{a^n b^n \mid n \geq 0\}$
- If we allow union of CA, then this class of automata is closed under union, intersection, negation, determinisation
- A transition CA is a finite-state automaton A together with a Presburger formula which constraints the number of times transitions are taken in an accepting run.
  - can accept languages like $\{a^n b^n a^m b^m \mid n, m \geq 0\}$
  - correspond to RBCA
- Transition CA are not closed under determinisation and complementation

Conclusion

- Parikh image: fundamental concept for language theory, verification
- An existentially quantified Presburger formula of linear size can be obtained for automata and CFG
- Is satisfiability of these formulae together with additional constraints efficiently solvable in practice ?
- A systematic study of the practical complexity has yet to be done