Complexity of inclusion and universality problems for unambiguous automata and unambiguous context-free grammars

LC

[Added from sharelatex]

1 Complexity of inclusion problems

1.1 Determinisation trick

First of all, we notice that in an inclusion problem $L(M) \subseteq L(N)$, we can assume (under very mild conditions) that $L(M)$ is recognised by a deterministic machine M. The transformation below even preserves whether $L(N)$ is recognised by a deterministic or unambiguous machine N.

Assume that $L(M)$ is recognised by a nondeterministic machine M with transitions of the form $\delta = p \xrightarrow{a,\mathsf{op}} q \in \Delta_M$, where op is an operation that manipulates a local data structure (a stack, queue, etc...). We modify M into a new machine M' by replacing each transition δ above with $p \xrightarrow{\delta,\mathsf{op}} q \in \Delta_{M'}$. It is evident that M' is by construction deterministic (in fact, every transition has a unique label). Intuitively, M' declares which transition will be used to read a. We need to adapt the machine N in order to preserve inclusion. For every transition $r \xrightarrow{a,\mathsf{op}} s \in \Delta_N$ and for every $\delta = p \xrightarrow{b,\mathsf{op}'} q \in \Delta_M$ with $b = a$, we have in N' a transition $r \xrightarrow{\delta,\mathsf{op}} s \in \Delta_{N'}$.

If N is deterministic, so is N'. And if N is unambiguous, so is N'.

Clearly,

$$L(M) \subseteq L(N) \quad \text{if, and only if,} \quad L(M') \subseteq L(N').$$

$\subseteq$	DFA	UFA	NFA	DCFG	UCFG	CFG
DFA	PTIME	PTIME	PSPACE-c.	PTIME	UUCFL	undecid.
UFA	PTIME	PTIME [10]	PSPACE-c.	PTIME	UUCFL	undecid.
NFA	PTIME	PTIME 1.3	PSPACE-c.	PTIME	UUCFL	undecid.
DCFG	PTIME	$\leq$UUCFL	EXPTIME-c.	undecid.	undecid.	undecid.
UCFG	PTIME	$\leq$UUCFL	EXPTIME-c.	undecid.	undecid.	undecid.
CFG	PTIME	$\leq$UUCFL	EXPTIME-c.	undecid.	undecid.	undecid.

Figure 1: Complexity of inclusion problems for various classes of regular and context-free languages

Indeed, let Σ be the original alphabet and let $\Sigma' = \Delta$ be the new alphabet (set of transitions). Consider the homomorphism $h : \Sigma' \to \Sigma$ that maps a transition $\delta = p \xrightarrow{a,\mathsf{op}} q \in \Delta$ to its label $h(\delta) = a \in \Sigma$. Let $A = L(M)$, $B = L(N)$, $A' = L(M')$, and $B' = L(N')$. We have $A = h(A')$ and $B' = h^{-1}(B)$. In general,

$$A' \subseteq h^{-1}(h(A')) \quad \text{and} \quad h(h^{-1}(B)) \subseteq B.$$

If $A \subseteq B$ holds, then $h^{-1}(A) \subseteq h^{-1}(B)$. By the definition of A and B', and the first property above, we have $A' \subseteq h^{-1}(h(A')) \subseteq B'$, as required. If $A' \subseteq B'$ holds, then also $h(A') \subseteq h(B')$ holds. Similarly as above, we have $A = h(A') \subseteq h(B') = h(h^{-1}(B)) \subseteq B$, as required.

By the proof above, $L(N')$ is obtained by applying an inverse homomorphism to $L(N)$. If $L(M) = h(L(M'))$, which operation applied to $L(M)$ yields $L(M')$?

Clarify

1.2 From inclusion to universality

Let $\mathcal{L}$ and $\mathcal{M}$ be two classes of languages. Sometimes we can reduce an inclusion problem $L \subseteq M$ with $L \in \mathcal{L}$ and $M \in \mathcal{M}$ to the universality problem according to the following equivalence:

$$L \subseteq M \quad \text{if, and only if,} \quad M \cup (\Sigma^* \setminus L) = \Sigma^*. \tag{1}$$

In order to apply the equivalence above, we assume that $\mathcal{L}$ is effectively closed under complements. For this reason, we will usually consider deterministic languages $L \in \mathcal{L}$. Thanks to Sec. 1.1, we can in fact always reduce to the case when L is deterministic.

We will see two applications of (1) below, namely CFG $\subseteq$ UFA and NFA $\subseteq$ UCFG, which both reduce to UUCFL.

1.3 NFA $\subseteq$ UFA

By the trick of Sec. 1.1, the problem reduces to DFA $\subseteq$ UFA. By (1), $L \subseteq M$ is equivalent to $N := M \cap L \cup (\Sigma^* \setminus L) = \Sigma^*$. Notice that N is effectively UFA. Since the universality problem for unambiguous automata can be solved in PTIME, our DFA $\subseteq$ UFA is in PTIME as well.

1.4 NFA $\subseteq$ DCFG

The "NFA $\subseteq$ DCFG" problem is solved in PTIME by effectively complementing the DCFG, intersecting it with the NFA, and testing non-emptiness of the resulting CFG.

1.5 NFA $\subseteq$ UCFG

Using the trick of Sec. 1.1, the problem "NFA $\subseteq$ UCFG" reduces to "DFA $\subseteq$ UCFG". Thanks to (1), the latter problem reduces to the universality problem "UCFG $= \Sigma^*$" as follows: Let $L = L(A)$ and $M = L(G)$ for a DFA A and a UCFG G. Then $L \subseteq M$ is equivalent to $N := M \cap L \cup (\Sigma^* \setminus L) = \Sigma^*$. The

language $\Sigma^* \setminus L$ is recognised by a DFA obtained by complementing A, and $M \cap L$ is effectively UCFL (being the intersection of a DFA language and a UCFG language). Finally, notice that the $M \cap L$ and $\Sigma^* \setminus L$ are disjoint, and thus N is effectively UCFL.

1.6 CFG $\subseteq$ NFA

The "CFG $\subseteq$ NFA" problem is easily shown to be EXPTIME-c.. In fact, the following closely related problem is EXPTIME-c.: Given a CFG G and a family of DFA's $A_1, \ldots, A_k$, decide whether $L(G) \cap \bigcap_{i=1}^{k} L(A_k) = \emptyset$ [8, 12]. We can reduce the problem above to the "CFG $\subseteq$ NFA" problem, by simply noticing that $L(G) \cap \bigcap_{i=1}^{k} L(A_k) = \emptyset$ holds iff $L(G) \subseteq \bigcup_{i=1}^{k} \Sigma^* \setminus L(A_k)$, and that, since the A_k's are deterministic, a polynomial size NFA can be built to recognise the language on the right.

1.7 CFG $\subseteq$ UFA

Similarly as earlier, "CFG $\subseteq$ UFA" reduces to "DCFG $\subseteq$ UFA", which in turn reduces to UUCFL: By (1), $L \subseteq M$ is equivalent to $N := M \cap L \cup (\Sigma^* \setminus L) = \Sigma^*$. Notice that $M \cap L$ can be recognised by a UCFG of polynomial size (since L is deterministic and M unambiguous), that $\Sigma^* \setminus L$ can be recognised by a DCFG of polynomial size (since L is deterministic), and since the last two languages are disjoint, their union is UCFL. This gives a PSPACE algorithm.

UFA over a unary alphabet $\Sigma = \{a\}$ can be complemented with quasi-polynomial complexity $n^{O(\log n)}$ [1]. In this case, we can decide CFG $\subseteq$ UFA in deterministic quasi-polynomial time.

While "DCFG $\subseteq$ UFA" reduces to UUCFL, which is in PSPACE, this needs not be optimal. Another more direct way could be the following. Let A be a DPDA and B an UFA. Then:

1. Compute in PTIME the measure $\mu(q)$ of every control state q of $L(B)$.
2. Construct a UPDA A' for $L(A) \cap L(B)$. Control locations of A' are of the form $\langle p, q \rangle$ with $p \in A$ and $q \in B$. Notice that by construction $L(\langle p, q \rangle) \subseteq L(q)$ and thus $\mu(\langle p, q \rangle) \leq \mu(q)$. Then, $\mu(\langle p, q \rangle) \geq \mu(q)$ iff $L(\langle p, q \rangle) = L(q)$ iff $L(p) \subseteq L(q)$.
3. Thus it suffices to approximate $\mu(\langle p, q \rangle)$ within $\log(\mu(q))$ bits of precision. The latter problem is on a UPDA obtained by taking the product of a DPDA and a UFA. This in general can be simpler than computing the measure of an arbitrary UPDA/UCFL. Idea: extend the method of [3] from a DFA to a UFA.

1.8 Core problems—summary

We identify the following core computational problems.

1. Compute $\mu(L)$ for L UCFL. We show this problem is SQRTSUM-hard, however it is not directly used to decide inclusion/universality problems. We don't know whether $\mu(L) = 1$ is SQRTSUM-hard.
2. Compute $\mu(L)$ for L DCFL. It may be easier than for UCFL.

3. Compute $\mu(L \cap M)$ for L DCFL and M UFA. This has applications to the inclusion problem DCFL $\subseteq$ UFA, since the latter reduces to deciding $\mu(L \cap M) + \mu(\Sigma^* \setminus L) = 1$, where $\Sigma^* \setminus L$ is effectively DCFL.

2 Preliminaries

Let $\Sigma_n = \{a_1, \ldots, a_n\}$ be an alphabet of size n and let $m \leq n$ a bound on the number of allowed letters. Then,

$$\mu_{\Sigma_n}(\Sigma_m^*) = \frac{1}{n-m+1}. \tag{2}$$

Representation lemma. The following lemma shows that we can represent arbitrary rational numbers as measures of suitably constructed regular languages.

Lemma 1 (Representation lemma)**.** *Let $n+1 \in \mathbb{N}$ with $n \geq 2$ be a base, let $m \in \mathbb{N}$ s.t. $1 \leq m \leq n$, and let $\alpha \in \mathbb{R}$ with $0 \leq \alpha \leq \frac{1}{n-m+1}$ be written in reduced form as*

$$\alpha = \frac{p}{q}, \qquad \text{with } p, q \in \mathbb{N},\ p \leq q.$$

There exists a DFA A over alphabet $\Sigma_n = \{a_1, \ldots, a_n\}$ using only letters from $\Sigma_m \subseteq \Sigma_n$, and thus $L(A) \subseteq \Sigma_m^*$, s.t.

$$\mu_{\Sigma_n}(L(A)) = \alpha.$$

Moreover, if there exists $\ell \in \mathbb{N}$ s.t.

$$q \,|\, (n+1)^\ell, \tag{3}$$

then A can be taken of size polynomial in $\log q$, n, and ℓ.

Proof. If $\alpha = \frac{1}{n-m+1}$, then just take A to be the automaton recognising Σ_m^*, and we are done.

Otherwise, assume $\alpha < \frac{1}{n-m+1}$. Our aim is to write α as the following infinite geometric sum:

$$\alpha = \sum_{i=0}^{\infty} \frac{\alpha_i}{(n+1)^{i+1}}, \qquad \text{with } 0 \leq \alpha_i \leq m^i, \tag{4}$$

where α_i intuitively counts the number of words of length i in $L(A)$. Moreover, we would like the sequence $\alpha_0, \alpha_1, \ldots$ to be eventually periodic (with small prefix and period), in order to construct a (small) finite automaton A recognising a language $L(A)$ of measure α. Let $k \in \mathbb{N}$ be a length threshold. The measure of all words of length at most k is

$$\mu_{\Sigma_n}(\Sigma_m^{\leq k}) = \sum_{i=0}^{k} \frac{m^i}{(n+1)^{i+1}} = \frac{1}{n-m+1}\left(1 - \left(\frac{m}{n+1}\right)^{k+1}\right), \tag{5}$$

Since the quantity in (5) goes to $\frac{1}{n-m+1} > \alpha$ for large k, fix in the following the unique $k \in \mathbb{N}$ s.t. $\alpha_0 = m^0, \alpha_1 = m^1, \ldots, \alpha_{k-1} = m^{k-1}$, and $0 \leq \alpha_k < m^k$ s.t.

$$\mu_{\Sigma_n}(\Sigma_m^{\leq k-1}) + \frac{\alpha_k}{(n+1)^{k+1}} \leq \alpha < \mu_{\Sigma_n}(\Sigma_m^{\leq k-1}) + \frac{\alpha_k + 1}{(n+1)^{k+1}}. \tag{6}$$

For complexity considerations to be made later, we note here that k satisfies $\alpha \leq \mu_{\Sigma_n}(\Sigma_m^k)$, and thus $k \geq \frac{\log(1-(n-m+1)\alpha)}{\log m - \log(n+1)} - 1 = \frac{-\log(1-(n-m+1)\alpha)}{\log(n+1) - \log m}$. The minimal denominator is achieved by $m = n$, and the maximal numerator by $m = 1$. Replacing, it suffices to take $k \geq \frac{-\log(1-n\alpha)}{\log(n+1)-\log n}$. Since $-\log(1-n\alpha) = O(\log q)$ and $\log(n+1) - \log n = \log \frac{n+1}{n} = \log(1 + \frac{1}{n}) = O(\frac{1}{n})$, we obtain

$$k = O(n \log q). \tag{7}$$

Let

$$\beta = \alpha - \left(\mu_{\Sigma_n}(\Sigma_m^{\leq k-1}) + \frac{\alpha_k}{(n+1)^{k+1}} \right), \tag{8}$$

and thus $0 \leq \beta < \frac{1}{(n+1)^{k+1}}$. We can write β in base-$(n+1)$ as

$$\beta = \frac{1}{(n+1)^k} \sum_{j=1}^{\infty} \frac{\beta_j}{(n+1)^{j+1}}, \quad \text{with } 0 \leq \beta_j \leq n. \tag{9}$$

Intuitively, we can interpret β_j as the number of words of length $k+j$ that our language contains. Since β is a rational number, the sequence $\beta_1, \beta_2, \ldots$ is ultimately periodic [7], i.e., there exists a threshold $j_1 \in \mathbb{N}$ and a period $l \in \mathbb{N}$ s.t. for every $j \geq j_1$

$$\beta_{j+l} = \beta_j. \tag{10}$$

Let $\gamma_1 = \beta_{j_1}, \gamma_2 = \beta_{j_1+1}, \ldots, \gamma_l = \beta_{j_1+l-1}$. Consequently,

$$\begin{aligned} \beta &= \frac{1}{(n+1)^k} \left(\sum_{j=1}^{j_1-1} \frac{\beta_j}{(n+1)^{j+1}} + \sum_{s=0}^{\infty} \left(\frac{\gamma_1}{(n+1)^{j_1+sl+1}} + \cdots + \frac{\gamma_l}{(n+1)^{j_1+sl+l}} \right) \right) \\ &= \frac{1}{(n+1)^k} \left(\sum_{j=1}^{j_1-1} \frac{\beta_j}{(n+1)^{j+1}} + \frac{1}{(n+1)^{j_1-1}} \sum_{s=0}^{\infty} \frac{\gamma}{(n+1)^{l(s+1)+1}} \right), \quad \text{where} \end{aligned} \tag{11}$$

$\gamma = \gamma_1(n+1)^{l-1} + \cdots + \gamma_l(n+1)^0$.

Intuitively, γ is the number of words of length $k + j_1 - 1 + (s+1)l$, for every $s \in \mathbb{N}$.

We now build a regular expression e recognising a language of measure α. For a given length k and cardinality $0 \leq h \leq m^k$, we build an expression $e_{h,k}$ recognising a language of measure $\frac{h}{(n+1)^{k+1}}$. Write h as a base-m number

$$m = \sum_{i=0}^{k} h_i \cdot m^i, \quad \text{with } 0 \leq h_i < m. \tag{12}$$

Then, the following regular expression $e_{h,k}$ recognises precisely h words of length k and no other word: If $h = m^k$, take $e_{h,k} = \Sigma_m^k$, and otherwise

$$\begin{aligned} e_{h,k} =& (a_1 + \cdots + a_{h_0}) \cdot \Sigma_m^0 \cdot a_1^{k-1} + \\ &+ (a_1 + \cdots + a_{h_1}) \cdot \Sigma_m^1 \cdot a_1^{k-2} + \cdots + \\ &+ (a_1 + \cdots + a_{h_{k-1}}) \cdot \Sigma_m^{k-1} \cdot a_1^0. \end{aligned}$$

The size of $e_{h,k}$ is $O(k(m+k)) = O(k(n+k))$.

We represent $\alpha - \beta = \mu(\Sigma_m^{\leq k-1}) + \frac{\alpha_k}{(n+1)^{k+1}}$ as the measure of the language recognised by the regular expression, also of size $O(k(n+k))$,

$$f = \Sigma_m^{\leq k-1} + e_{\alpha_k,k}. \tag{13}$$

Similarly, by (11) we can represent β as

$$g = e_{\beta_1,k+1} + \cdots + e_{\beta_{j_1-1},k+j_1-1} + e_{\gamma,k+j_1-1+l} \cdot e^*_{1,l}. \tag{14}$$

Since $n \leq m^k$ for k sufficiently large and $\beta_j \leq n$ ($k \geq \frac{\log n}{\log m}$ suffices), there are enough words β_j of length $k+j$ over alphabet Σ_m, and thus $e_{\beta_j,k+j}$ is well-defined. Similarly, in order for $e_{\gamma,k+j_1-1+l}$ to be well-defined, we need $\gamma \leq m^{k+j_1-1+l}$, which is satisfied for k large enough since $\gamma = O(n^l)$. The expression g above is of size $O(k(n+k) + j_1^2(n+j_1) + l(n+l))$.

Finally, the sought expression of measure α is $e := f + g$, which is of the same asymptotic size as g above. Note that e is unambiguous, since the length of the string uniquely specifies how it is parsed in $L(e)$. (Perhaps one could even derive a small deterministic automaton...?) By (7), and assuming $j_1, l \geq n$, e is of size

$$O(n^2 \log q + j_1^3 + l^2). \tag{15}$$

For the second part of the claim, assume (3) for some ℓ. By (7) and the assumption above,

$$k = O(n\ell \log(n+1)). \tag{16}$$

Since β is defined as $\alpha - \frac{\cdots}{(n+1)^{k+1}}$, we can write $\beta = \frac{r}{s}$ with $r, s \in \mathbb{N}$ and $r \leq s$, where $s \mid (n+1)^{k+1}$. If we decompose the base $n+1$ in prime factors as

$$n + 1 = p_1^{z_1} p_2^{z_2} \cdots p_m^{z_m}, \quad \text{with } z_1, \ldots, z_m \in \mathbb{N},$$

then s is of the form $s = p_1^{t_1} p_2^{t_2} \cdots p_m^{t_m}$ with $t_i \leq (k+1)z_i$. By [7, Theorem 136], β can be written in base-$(n+1)$ with a *finite* expansion of length $j_1 = \max\left\{\frac{t_1}{z_1}, \ldots, \frac{t_m}{z_m}\right\} = O(k+1)$ (therefore the period is $l = 0$), and thus by (16)

$$j_1 = O(n\ell \log(n+1)). \tag{17}$$

By applying (15) to this case, we obtain a regular expression e of size

$$O(n^3 \ell^3 \log^3(n+1)), \tag{18}$$

which is polynomial in $\log q = O(\ell \log n)$, ℓ, and n, as required. □

3 Closure properties of UCFL

Complement. The complement of UCFL need not be CFL [9]. Consider the language $L = L_1 L_2 \cup L_3$, where

$$L_1 = \{a^m b^n \mid 10m < n < 12m \vee 10n < m < 12n\},$$
$$L_2 = \{c^p d^q \mid 10p < q < 12p \vee 10q < p < 12q\}, \text{ and}$$
$$L_3 = \{a^m b^n c^p d^q \mid 10n < p < 12n \wedge 6m < q < 8m\}.$$

We verify the following facts.

- L_1 is the union of two disjoint UCFL, and thus itself UCFL. To see why

  $$\{a^m b^n \mid 10m < n < 12m\}$$

  is UCFL, observe that if $10m < n < 12m$ holds, then n can be written in an unique way as the sum of m numbers

  $$n = 10 + \cdots + 10 + 11 + \cdots + 11 + 12 + \cdots + 12,$$

  with 11 appearing at least once. Based on the observation above, we construct an UPDA which upon reading a's pushes a on the stack. Afterwards, for every b read, it pops from the stack 10, 11, or 12 a's according to the decomposition above (popping first in multiples of 10, then 11, and then 12).

  The second component $\{a^m b^n \mid 10n < m < 12n\}$ is UCFL with a similar argument and clearly disjoint from the first one, and thus L_1 is UCFL.

- A similar argument as above shows that L_2 and L_3 are UCFL. Moreover, also $L_1 L_2$ is UCFL since L_1, L_2 are over disjoint alphabets.

- $L_1 L_2$ and L_3 are disjoint. Assume the defining condition of L_3 holds, namely

  $$10n < p < 12n \wedge 6m < q < 8m.$$

  There are four cases to consider, depending on which component of L_1 we are in, and similarly for L_2.

  - Assume $10m < n < 12m$ and $10p < q < 12p$ hold. Then, $100m < p < 144m$ and thus $1000m < q < 1728m$, which is a contradiction.
  - Assume $10m < n < 12m$ and $10q < p < 12q$ hold. Then, as above $100m < p < 144m$ holds, but also $60m < p < 96m$, which is a contradiction.

    The other two cases are similar to the one above.

- Since L is the union of disjoint UCFL, it is itself UCFL.

- $\Sigma^* \setminus L$ is not CFL. TODO.

Union and intersection with regular languages. UCFL are closed under union and intersection with regular languages, with quadratic complexity when the regular language is presented as a DFA.

Closure under intersection with a regular language is clear by taking the product of a UPDA and a DFA.

For closure under union, the input is a UCFL L (presented by a UPDA A) and a regular language M (presented by a DFA B). We build a UPDA C recognising $L \setminus M$ and then take the product with B in order to build a PDA $D = C \times B$ recognising $(L \setminus M) \cup M$: Since the union is disjoint, D is unambiguous, as required.

4 Non-trivial lower bounds

The universality problem for UFA can be reduced to solving linear equations and checking that the solution has 1 in a given component. Consequently, this gives both a PTIME and a NC^2 upper bound.

Any lower-bound? Can we reduce solving linear equations to the UFA universality problem?

5 Lower bounds

5.1 SQRTSUM problem

The $\mathsf{SQRTSUM}$ asks, given $d_0, d_1, \ldots, d_n \in \mathbb{N}$, whether the following holds:

$$\sum_{i=1}^{n} \sqrt{d_i} \geq d_0. \tag{19}$$

In the following, let $d = \max_{i=1}^{n} d_i$.

We remark that we could equivalently have taken "$\leq$" in the definition of $\mathsf{SQRTSUM}$ above, since the two problems reduce to each other. By doing binary search in the interval $\{0, 1, \ldots, nd\}$, with only $O(\log(nd))$ (polynomially many) queries to 19 we can find the unique $\hat{d}_0$ s.t. $\hat{d}_0 \leq \sum_{i=1}^{n} \sqrt{d_i} \leq \hat{d}_0 + 1$, which obviously allows one to answer queries of the form $\sum_{i=1}^{n} \sqrt{d_i} \leq d_0$.

Let

$$x_i = 1 - \frac{\sqrt{d_i}}{d} = 1 - \sqrt{\frac{d_i}{d^2}}. \tag{20}$$

Consequently, $\sqrt{d_i} = d(1 - x_i)$, which implies

$$\sum_{i=1}^{n} \sqrt{d_i} = d \sum_{i=1}^{n} (1 - x_i) = d \left(n - \sum_{i=1}^{n} x_i \right) \geq d_0.$$

This in turn is equivalent to

$$\sum_{i=1}^{n} x_i \leq n - \frac{d_0}{d}. \tag{21}$$

From (20), x_i is the least non-negative solution of

$$x_i = \frac{1}{2} \left(1 - \frac{d_i}{d^2} \right) + \frac{1}{2} x_i^2. \tag{22}$$

5.2 SQRTSUM-hardness for Markov chain + DCFL

We begin with an easy reduction. We consider a finite Markov chain M generating the measure (instead of the fixed flip-coin measure) (equivalently, right/left linear SCFG). We show that $\mu_M(G) \sim x$ is SQRTSUM-hard already for G a DCFG.

By normalising (21) we have

$$\frac{1}{n}\sum_{i=1}^{n} x_i \leq 1 - \frac{d_0}{dn}, \text{ where} \tag{23}$$

$$x_i = \underbrace{\frac{1}{2}\left(1 - \frac{d_i}{d^2}\right)}_{\alpha_i} + \frac{1}{2}x_i^2. \tag{24}$$

We design a Markov chain M and a DCFG G s.t. $\mu_M(G) = \frac{1}{n}\sum_{i=1}^{n} x_i$. Let M be the one-state $\{p\}$ chain, with p both initial and final, and transitions

$$p \xrightarrow{a_1,\frac{1}{n}} p, \ldots, p \xrightarrow{a_n,\frac{1}{n}} p,$$
$$p \xrightarrow{b_1,\alpha_1} p, \ldots, p \xrightarrow{b_n,\alpha_n} p,$$
$$p \xrightarrow{c_1,\frac{1}{2}} p, \ldots, p \xrightarrow{b_n,\frac{1}{2}} p.$$

Let G be the DCFG

$$X \leftarrow a_1 \cdot X_1 \mid \cdots \mid a_n \cdot X_n,$$
$$X_i \leftarrow b_i \mid c_i \cdot X \cdot X, \qquad \text{for every } i = 1, \ldots, n.$$

Thus, we take $x = 1 - \frac{d_0}{dn}$.

Can we take $x = \frac{1}{2}$?

5.3 SQRTSUM-hardness of μ(UCFL) $\sim x$

The UUCFL problem asks whether a given unambiguous context-free grammar recognises the universal language Σ^*. We show that deciding whether $\mu(L) \sim x$ for a UCFL L and $x \in \mathbb{Q}$ is hard for SQRTSUM, where $\sim \in \{\leq, <, \geq, >\}$. We assume that the rational number x is presented as a pair of integers encoded in binary.

It is not clear whether L is DCFL.

We construct a UCFG G over a n-ary alphabet $\Sigma = \{a_1, \ldots, a_n\}$ and a constant $\alpha \in \mathbb{Q}$ s.t.

$$\mu(L(G)) \geq \alpha \quad \text{iff} \quad (21) \text{ holds (and thus (19)).}$$

We build a UCFG G_i s.t.

$$\mu(L(G_i)) = x_i. \tag{25}$$

Since x_i is an irrational number, $|\Sigma| \geq 2$ is necessary, otherwise G would recognise a regular language, and thus its measure would be rational. (However, in general it might still be the case that computing such a rational measure is expensive...) We assume w.l.o.g. that

(NOT USED) All the d_i's are distinct. If not, say $d_1 = d_2$, then we can replace them by the single $4d_1$, since $\sqrt{d_1} + \sqrt{d_2} = 2\sqrt{d_1} = \sqrt{4d_1}$.

1. We can assume that the maximal d is a square of the form:

$$d = (n+1)^{2h}. \tag{26}$$

If not, add a new integer $d_{n'} = (n'+1)^{2h}$ for h large enough, where $n' = n+1$, and replace d_0 with $d_0 + \sqrt{d_{n'}} = d_0 + (n'+1)^h$.

We look for a grammar G_i with initial nonterminal X_i and a rule of the form:

$$X_i \leftarrow C_i \mid A_i \cdot X_i \cdot a_n \cdot X_i,$$

where $A_i, C_i \subseteq \Sigma_{n-1}^*$ are regular languages represented by DFA of polynomial size; moreover, all words in A_i will have the same length $k-1$. If we call x the measure $\mu(L(X_i))$, a the measure of $A_i \cdot a_n$, and c the measure of C_i, under the assumption that G is unambiguous, we obtain (recall that $\mu(LM) = (n+1)\mu(L)\mu(M)$ over an alphabet of size n for an unambiguous product LM)

$$x = c + (n+1)^2 a x^2.$$

By comparing the equation above with (22), we derive

$$a = \frac{1}{2(n+1)^2} = \frac{\frac{(n+1)^{k-1}}{2}}{(n+1)^{k+1}}, \text{ for every } k \geq 1, \text{ and} \tag{27}$$

$$c = \frac{1}{2}\left(1 - \frac{d_i}{d^2}\right). \tag{28}$$

We assume w.l.o.g. that n is odd and, consequently the numerator $\frac{(n+1)^k}{2}$ above is an integer. Assume $n \geq 3$ in order to have $\frac{(n+1)^{k-1}}{2} \leq (n-1)^{k-1}$, and let $A_i \subseteq \Sigma_{n-1}^{k-1}$ be a finite language containing precisely $\frac{(n+1)^{k-1}}{2}$ strings of length $k-1$.

Since $\mu_{\Sigma_n}(\Sigma_{n-1}^*) = \frac{1}{2}$ and $c < \frac{1}{2}$, by Lemma 1, there exists a DFA C_i recognising a language $L(C_i) \subseteq \Sigma_{n-1}^*$ of measure c. Moreover, since c can be put in the form $c = \frac{\frac{d}{2}(d^2 - d_i)}{(n+1)^{6h}} = \frac{p}{q}$ with $p, q \in \mathbb{N}$ relatively prime and $q \mid (n+1)^{6h}$, C_i is of polynomial size.

We argue that G_i is unambiguous. Any word produced by X_i has the same number of blocks in A_i's as a_n's. By way of contradiction, suppose $w \in L(X_i)$ is a word with two different derivations. Necessarily w contains a_n, otherwise the first production is applied and $w \in C_i$ can be derived in only one way. Thus, the second production is applied and w can be put in the two forms $w = uva_nt = u'v'a_nt'$ with $u, u' \in A_i$, and $v, t, v', t' \in L(X_i)$. First, $u = u'$ since all strings in A_i have length $k-1$. Assume w.l.o.g. that $|v| < |v'|$, which implies that v' is of the form $v' = va_nz$ with $v = xy$, $x \in A_i$ and $y, z \in L(X_i)$. Since y is in $L(X_i)$, it has the same number of A_i's blocks as well as a_n's. Thus, $v = xy$ cannot be in $L(X_i)$ because it has one more $x \in A_i$, which is a contradiction.

We obtain $L(X_i) \subseteq (\Sigma_0 \cup \Sigma_1)^{\geq 2}$ and $\mu(L(X_i)) = x_i$. By constructing the unambiguous grammar G with initial nonterminal X and productions

$$X \leftarrow a_1 \cdot X_1 \mid \cdots \mid a_n \cdot X_n, \tag{29}$$

we have that $L(X) \subseteq \Sigma \cdot (\Sigma_0 \cup \Sigma_1)^{\geq 2}$ and

$$\begin{aligned}\mu(L(X)) &= \mu(L(a_1 \cdot X_1)) + \cdots + \mu(L(a_n \cdot X_n)) = \\ &= \frac{1}{n+1}(x_1 + \cdots + x_n) = \\ &= \frac{1}{n+1}\left(n - \frac{\sqrt{d_1} + \cdots + \sqrt{d_n}}{d}\right)\end{aligned}$$

and thus

$$\sum_{i=1}^{n} \sqrt{d_i} \geq d_0 \quad \text{iff} \quad \mu(L(X)) \leq \alpha := \frac{1}{n+1}\left(n - \frac{d_0}{d}\right). \tag{30}$$

Consequently, approximating $\mu(L(G))$ in time polynomial in the number of bits of precision would imply a major breakthrough for SQRTSUM.

5.4 SQRTSUM hardness for $\mu(\mathbf{DCFL}) \geq x$

Direction: Prove hardness for $\mu(L) \geq x$ with L DCFL. This is the same question as $\mu(L) \leq x$ because $1 - \mu(L) = \mu(\Sigma^* \setminus L)$ and $\Sigma^* \setminus L$ is effectively DCFL. Or maybe in PTIME following [3]?

??? WORK IN PROGRESS ???

If we consider the coin-flip measure, then μ(DCFL) satisfies a PPS, i.e. the sum of the coefficients is ≥ 1.

5.5 SQRTSUM hardness for UUCFL

If $\mu(\text{UCFL}) = 1 - \alpha$ with $\alpha > 0$, how small can α be? Can we give a polynomial bound on $\log \frac{1}{\alpha}$?

What is the nonlinear depth of the resulting system of equations? Is it logarithmically bounded? (Number of nonlinear SCC on any path to a bottom SCC.)

??? WORK IN PROGRESS ???

We construct a regular language $L' \subseteq \Sigma_2^*$ of measure $\mu(L') = 1 - \alpha$. Since $0 < \alpha < 1$, the same holds for $1 - \alpha$. We have

$$1 - \alpha = \frac{d + d_0}{(n+1)d} = \frac{(n+1)^{2h} + d_0}{(n+1)^{2h+1}} = \frac{1}{n+1} + \frac{d_0}{(n+1)^{2h+1}},$$

where $d_0 \leq n\sqrt{d} = n(n+1)^h \leq (n+1)^{h+1}$. We can write d_0 in base $n+1$ as $d_0 = \sum_{i=0}^{k} f_i(n+1)^i$, where $k := \lfloor \log_{n+1} d_0 \rfloor \leq h+1$ and, for every $0 \leq j \leq k$, $0 \leq f_i \leq n$. Consequently, we can write $1 - \alpha$ as

$$1 - \alpha = \frac{1}{n+1} + \frac{f_0}{(n+1)^{2h+1}} + \frac{f_1}{(n+1)^{2h}} + \cdots + \frac{f_k}{(n+1)^{2h-k+1}}, \tag{31}$$

allowing us to interpret the quantity above as the measure of a regular language $L' \subseteq \Sigma_2^*$ containing the empty word ε, and containing f_i words of length $2h-i+1$ for every $0 \leq i \leq k$. The shortest such word is of length $\geq 2h-(h+1)+1 = h \geq 2$,

and there are enough such words $|\Sigma_2^2| = \left(\frac{n-1}{4}\right)^2 \geq n$ for n sufficiently large. Consider the language over Σ recognised by the nonterminal S and a rule

$$S \leftarrow L' \mid X. \tag{32}$$

First L' and $L(X)$ are disjoint since $L' \subseteq \{\varepsilon\} \cup \Sigma_2^{\geq 2}$. Consequently,

$$\mu(S) = \mu(L') + \mu(L(X)) = \frac{d + d_0}{(n+1)d} + \frac{1}{n+1}\left(n - \frac{\sqrt{d_1} + \cdots + \sqrt{d_n}}{d}\right) =$$
$$= 1 + \frac{d_0 - (\sqrt{d_1} + \cdots + \sqrt{d_n})}{(n+1)d}.$$

5.6 Bounded ambiguity

Notice that the universality problem for the union of two DCFL is undecidable: the first DCFL recognises runs of a deterministic Minsky machine that has a mistake on the first counter, and the second DCFL does the same on the second counter, in such a way that their (possibly overlapping union) encodes runs with at least some mistake. Then their union is universal iff the Minsky machine is empty.

This implies that already universality of 2-ambiguous grammars is undecidable.

5.7 μ(UCFL) can be very close to 1

We show a family of examples where the measure is $1 - \frac{1}{2^{2^n}}$. We start from the unary UCFL containing exactly the word a^{2^n}. This is generated by X_n in the following family of productions:

$$X_{n+1} \leftarrow X_n \cdot X_n,$$
$$X_0 \leftarrow a.$$

Clearly, if we write $x_n = \mu(L(X_n))$, we have

$$x_0 = \frac{1}{2} \quad \text{and} \quad x_{n+1} = x_n^2,$$

from which we obtain $x_n = \frac{1}{2^{2^n}}$.

We now build a UCFG recognising all words of length $< 2^n$, i.e., $L_n = \{a^0, a^1, \ldots, a^{2^n-1}\}$. We introduce a nonterminal Y_n recognising L_n. When writing productions for Y_{n+1}, we guess whether the input word has length $< 2^n$ (first case), or not (second case):

$$Y_{n+1} \leftarrow Y_n \mid X_n \cdot Y_n,$$
$$Y_0 \leftarrow \varepsilon$$

The case $X_n \cdot Y_n$ above stems from the fact that, if there are $2^n \leq k < 2^{n+1}$ a's, then after reading 2^n of them (by X_n) there are $< 2^{n+1} - 2^n = 2^n$ left (read by Y_n). This yields corresponding polynomial equations

$$y_0 = \frac{1}{2} \quad \text{and} \quad y_{n+1} = y_n + x_n y_n = y_n(1 + x_n).$$

Note that it is an MPS, but not a PPS.

Can we improve the example to a PPS?

We can verify by induction that $y_n = 1 - \frac{1}{2^{2^n}}$ holds: For $n = 0$, it's immediate, and for $n = m + 1$ we have

$$1 - \frac{1}{2^{2^{n+1}}} = \left(1 - \frac{1}{2^{2^n}}\right)\left(1 + \frac{1}{2^{2^n}}\right).$$

This shows that the shortest witneess of non-universality can have exponential length, already for unary alphabet. Viceversa, since a word of length n has measure 2^{-n}, if the grammar is not universal $\mu(X) < 1$, then there is a non-universality witness of length at most $\log(1 - \mu(X))$. Thus upper bounds on $1 - \mu(X)$ yield upper bounds on the shortest non-universality witness.

5.8 Hardness of approximation

Can we show that even any nontrivial approximation of μ(UCFL) is hard?

6 PSPACE upper bound

In this section we present an PSPACE algorithm for UUCFL, the universality problem of unambiguous context-free languages. More generally, we consider the inequality problem for generating functions.

Does the argument leading to the PSPACE upper bound yield any upper bound on the length of the shortest rejected word in case the UCFG is not universal? We do have exponential lower-bounds from non-emptiness.

Let G be a CFG with m nonterminals $X_1, \ldots, X_m$, and for every nonterminal X_i and length $n \in \mathbb{N}$, let $T_n(X_i) = |L(X_i) \cap \Sigma^n|$ be the number of words of length n generated by X_i. The *generating function* $g_i : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \cap \{\infty\}$ of X_i is defined as

$$g_i(x) = \sum_{n=0}^{\infty} T_n(X_i) \cdot x^n.$$

Since $|T_n(X_i)| \leq \Sigma^n$, the series above converges to a finite value in $\mathbb{R}_{\geq 0}$ for every $x < |\Sigma|$. For two generating functions f, g, we write $f \leq g$ if, for every $x \in \mathbb{R}_{\geq 0}$, $f(x) \leq g(x)$. Given two nonterminals X_i, X_j of G, the *generating function inequality problem* asks whether $g_i \leq g_j$ holds.

Let $g : \mathbb{R}_{\geq 0} \to \mathbb{R}^m_{\geq 0}$ be defined as $g(x) = (g_1(x), \ldots, g_m(x))$. We assume that the grammar is in Chomsky normal form. Let A be the set of continuous functions in $\mathbb{R}_{\geq 0} \to \mathbb{R}^m_{\geq 0}$, and consider the mapping $F : A \to A$ defined as $F(f)(x) = (F_1(f)(x), \ldots, F_m(f)(x))$, where

$$F_i(f)(x) = 1_{X_i \to \varepsilon?} + \sum_{X_i \to a} x + \sum_{X_i \to X_j X_k} f_j(x) \cdot f_k(x) \tag{33}$$

TODO: check that F maps continuous functions to continuous functions.

Lemma 2. *If G is unambiguous, then g is the least fixpoint of F.*

Proof. The fact that g is a fixpoint $F(g) = g$ follows immediately from unambiguity. Notice that F is itself a monotonic and continuous function on A. Monotonicity is clear. If $g_1 \leq g_2 \leq \cdots$ is a non-decreasing sequence of continuous functions $g_n \in A$ with limit $g = \lim_n g_n$, then

$$\begin{aligned}
F_i(\lim_n g_n) &= 1_{X_i \to \varepsilon?} + \sum_{X_i \to a} x + \sum_{X_i \to X_j X_k} (\lim_n g_n)_j(x) \cdot (\lim_n g_n)_k(x) = \\
&= 1_{X_i \to \varepsilon?} + \sum_{X_i \to a} x + \sum_{X_i \to X_j X_k} (\lim_n g_{nj})(x) \cdot (\lim_n g_{nk})(x) = \\
&= 1_{X_i \to \varepsilon?} + \sum_{X_i \to a} x + \sum_{X_i \to X_j X_k} \lim_n g_{nj}(x) \cdot \lim_n g_{nk}(x) = \\
&= 1_{X_i \to \varepsilon?} + \sum_{X_i \to a} x + \sum_{X_i \to X_j X_k} \lim_n (g_{nj}(x) \cdot g_{nk}(x)) = \\
&= \lim_n \left(1_{X_i \to \varepsilon?} + \sum_{X_i \to a} x + \sum_{X_i \to X_j X_k} g_{nj}(x) \cdot g_{nk}(x) \right) = \\
&= \lim_n (F_i(g_n)).
\end{aligned}$$

Consider the non-decreasing sequence of continuous functions $g_1 \leq g_2 \leq \cdots$ defined as $g_1(x) = (0, \ldots, 0)$, and, for every $i \geq 1$, $g_{i+1} = F(g_i)$. Then 1) $g^* = \lim_n g_n$ is the least fixpoint of F, and 2) $g = g^*$. Regarding 1), g^* is a fixpoint of F since $F(g^*) = \lim_n F(g_n) = \lim_n g_{n+1} = g^*$, and it is clearly the least one since g_1 is the minimal element of A. Regarding 2), it follows directly from the following characterisation of the m-th approximant g_m:

Claim. *Let $T_{mn}(X_i)$ be the number of words of length n that can be derived from X_i by at most m rewriting steps. Then,*

$$g_{ni}(x) = \sum_{n=0}^{\infty} T_{mn}(X_i) \cdot x^n. \tag{34}$$

Clearly, $T_n = \lim_m T_{mn}$, and thus ... TODO □

Lemma 3. *The generating function inequality problem can be solved in* EXPTIME *for unambiguous grammars.*

Proof. Thanks to Lemma 2, the generating function g of an ambiguous grammar is the least fixpoint of F from (33). Let

$$p_i(x, \overline{y}) \equiv y_i - 1_{X_i \to \varepsilon?} - \sum_{X_i \to a} x - \sum_{X_i \to X_j X_k} y_j \cdot y_k, \tag{35}$$

$$\widehat{\varphi}(x, \overline{y}) \equiv \bigwedge_i p_i(x, \overline{y}) = 0, \text{ and} \tag{36}$$

$$\varphi(x, \overline{y}) \equiv \widehat{\varphi}(x, \overline{y}) \wedge \forall \overline{z} \cdot \widehat{\varphi}(x, \overline{z}) \implies \overline{y} \leq \overline{z}. \tag{37}$$

Consequently,

$$g(x) = \overline{y} \quad \text{iff} \quad \varphi(x, \overline{y}).$$

Thus, the inequality problem $g_i \leq g_j$ is equivalent to

$$\forall x, \overline{y} \cdot \varphi(x, \overline{y}) \implies y_i \leq y_j.$$

which is a formula of Tarski algebra of fixed alternation depth, and this fragment is solvable in EXPTIME. □

Let $\|x\| = \max_i x_i$ be the max-norm. A function $F : \mathbb{R}^m \to \mathbb{R}^m$ is *contractive* if there exists a constant $0 \leq \alpha < 1$ s.t. for every vectors $x, y \in \mathbb{R}^m$, $\|F(x) - F(y)\| \leq \alpha \cdot \|x - y\|$. By Banach's Fixpoint Theorem, a contractive function F has a unique fixpoint $x^* = F(x^*)$, which can moreover be found by iterating $x^* = \lim_n F^n(x_0)$ for any initial vector $x_0 \in \mathbb{R}^m$.

Lemma 4. *If the grammar G is linear, then F_x is contractive for x sufficiently small and thus it has a unique fixpoint.*

Proof. If G is a linear grammar, then by letting $y_i = f_i(x)$, we can write

$$F_i(y) = 1_{X_i \to \varepsilon?} + \sum_{X_i \to \alpha X_j \beta} x^{|\alpha\beta|} y_j. \tag{38}$$

Therefore,

$$F_i(y) - F_i(z) = \sum_{X_i \to \alpha X_j \beta} x^{|\alpha\beta|} y_j - \sum_{X_i \to \alpha X_j \beta} x^{|\alpha\beta|} z_j = \sum_{X_i \to \alpha X_j \beta} x^{|\alpha\beta|} (y_j - z_j),$$

and thus $\|F(y) - F(z)\| = \max_i \left(\sum_{X_i \to \alpha X_j \beta} x^{|\alpha\beta|} (y_j - z_j) \right) \leq \gamma \|y - z\|$, where $\gamma = \sum_{X_i \to \alpha X_j \beta} x^{|\alpha\beta|}$. Therefore, F is contractive $\gamma < 1$ provided that

$$x < k^{-l},$$

where k is the number of productions of G and $l = \max_{X_i \to \alpha X_j \beta} |\alpha\beta|$ is the maximum length of a r.h.s. of a production. □

Lemma 5. *The inequaliy problem between an UCFG gf f and an ULCFG gf g can be solved in PSPACE.*

Proof. By Lemma 4, generating functions of ULCFG are unique solutions of polynomial equations, and thus the problem is equivalent to

$$\exists x, \overline{y}, \overline{z} \cdot \varphi_f(x, \overline{y}) \wedge \varphi_g(x, \overline{z}) \to y_i \leq z_j.$$

which is a formula of the existential fragment of Tarski algebra, and thus solvable in PSPACE. □

Corollary 6. *The universality problem for unambiguous grammars is in PSPACE.*

Proof. Let $k = |\Sigma|$. The generating function of Σ^* is $g_{\Sigma^*}(x) = \sum_{n=0}^{\infty} k^n x^n = \frac{1}{1-kx}$, and thus L is universal iff $\forall x \cdot g_L(x) \geq \frac{1}{1-kx}$. This is the same as

$$\forall x, \overline{y} \cdot \bigwedge_i p_i(x, \overline{y}) = 0 \implies y_j \geq \frac{1}{1 - kx},$$

which is a formula of the existential fragment of Tarski algebra, and thus solvable in PSPACE. □

7 Connection with Etessami et al. [3]

A *stochastic context-free grammar* (SCFG) is a grammar G with weighted productions of the form

$$A \xrightarrow{p} \alpha$$

with $p \in \mathbb{R}_{\geq 0}$ and $\alpha \in (\Sigma \cup N)^*$, where for every nonterminal A,

$$\sum_{A \xrightarrow{p} \alpha} p = 1.$$

A SCFG G defines in a natural way a probability measure on finite words $\mu_G : \Sigma^* \to \mathbb{R}_{\geq 0}$. Intuitively, a SCFG is a branching process that induces a probability distribution on all the parse trees, and $\mu(w)$ is the sum of the probabilities of all parse trees yielding w. Given a SCFG G and a language $L \subseteq \Sigma^*$, one is interested in computing $\mu_G(L) \in \mathbb{R}_{\geq 0}$.

Example:

$$X \xrightarrow{1/2} XX,$$
$$X \xrightarrow{2/10} aXb,$$
$$X \xrightarrow{3/10} \varepsilon.$$

The termination probability $\mu_X(\Sigma^*)$ is the least non-negative solution of the equation:

$$x = \frac{5}{10}x^2 + \frac{2}{10}x + \frac{3}{10}.$$

SCFG can express the discounted flip-coin measure on words μ as μ_X, where:

$$X \xrightarrow{1/3} \varepsilon,$$
$$X \xrightarrow{1/3} aX,$$
$$X \xrightarrow{1/3} bX.$$

Consequently, we can reduce computing the flip-coin measure of a DPDA language L to the measure of a SCFG (in the form of a PPDA A). Every rule in the DPDA $X, \alpha \xrightarrow{a} Y, \beta\gamma$ yields a rule in the PPDA not sure

$$X, \alpha \xrightarrow{a, \frac{1}{3}} Y, \beta\gamma$$

Moreover, for every control location X and stack symbol α we have a rule $X, \alpha \xrightarrow{\varepsilon, \frac{1}{3}} X_\perp, \alpha$, where $X_\perp$ is a sink which is accepting iff X is accepting. Then, $\mu(L) = \mu_A(\Sigma^*)$.

If we do the same for a UPDA, then we get a weighted PDA where the sum of the weights for every X, α can be larger than one. However, still $\mu(X) \leq 1$.

8 Review of previous results on solving polynomial equations

For probabilistic polynomial systems (PPS, the sum of the coefficients of every equation is ≤ 1), one can decide in PTIME whether the least non-negative solution is 0. The same holds for 1 [5], and the latter is even in strongly polynomial time [2]. The decision problem of whether the solution $x_0^* \geq \frac{1}{2}$ is SQRTSUM-hard and also PosSLP-hard already for PPS generated by SCFG [5]. For arbitrary monotone systems of polynomial equations, hardness holds already for computing any non-trivial approximation.

What about checking whether $x_0^* = 1$ for arbitrary monotone systems?

The approximation problem clearly reduces to the decision problem $x_0^* \geq \alpha$ by doing binary search: If the first n bits of x_0^* are the rational α, then we can determine the next $(n+1)$-th bit by asking $x_0^* \geq \alpha + 2^{-(n+1)}$.

The decision problem is strictly harder than the approximation one, since in order to decide whether $x_0^* \geq \frac{1}{2}$ holds, it is in general not clear how many bits of x_0^* we need to know, i.e., how close an algebraic number represented by a PPS/MPS can be to a rational number.

Fo PPS, deciding whether $x_0^* \geq p$ is decidable in polynomial time in the unit-cost RAM model (unit-cost-PTIME) [4], and this problem is in fact unit-cost-PTIME-complete since it is PosSLP-hard [5]. Moreover, we can approximate x^* to within additive error 2^{-j} in time polynomial in j *in the Turing model of computation* [4]. Approach: after removing variables which are either 0 or 1 in the least solution, we can get within the desired accuracy with a linear number of iterations of Newton's method, and moreover the same holds even if we truncate the approximants to within polynomially many bits.

The removing $= 1$ components is necessary, as shown by the following example [11]:

$$x_i = \frac{1}{2}x_i^2 + \frac{1}{2}x_{i-1}, \text{ for } 1 \leq i \leq n,$$
$$x_0 = \frac{1}{2}x_0^2 + \frac{1}{2}.$$

The solution is $x^* = 1$, but we need $\geq 2^n \log \frac{1}{\varepsilon}$ iterations of Newton's method to get an approximation $\geq 1 - \varepsilon$, where n is the nonlinear depth of the system.

For monotone systems with $x^* > 0$ of nonlinear depth d (number of nonlinear SCCs to a bottom SCC) we can get to within additive error 2^{-j} of x^* (i.e., j bits of precision) in time polynomial in j, $\log 1/x_{\min}^*$, $\log x_{\max}^*$ and 2^d [11]. Can this be applied to UCFG?

Notice that with a simple repeated squaring PPS $x_{i+1} = x_i^2$ and $x_0 = \frac{1}{2}$ we can get solutions as small as $x_n^* = \frac{1}{2^{2^n}}$. We can implement this with a simple DCFG.

There are also MPS obtained from UCFG achieving values $x_n^* = 1 - \frac{1}{2^{2^n}}$ very close to 1; cf. Sec. 5.7. This suggests that Newton's method needs exponentially many iterations on those.

9 Nonnegative matrices

Let $B(x)$ be the Jacobian matrix obtained from the system of polynomial equations $x = F(x)$ corresponding to a given UCFG.

a) Is it the case that $\lim_{d\to\infty} B(x)^d = 0$? Conjecture: Yes. TO PROVE.

b)

10 Observations

If $f(x)$ is a polynomial equation with nonnegative coefficients, then it is not only monotone, but also *expansive*, in the sense that there is a nonnegative constant α s.t.

$$x \leq y \quad \text{implies} \quad y - x \leq \alpha \cdot (f(y) - f(x)). \tag{39}$$

Indeed, if $f(x) = \sum_{i=0}^{n} a_i x^i$, then

$$\begin{aligned} f(y) - f(x) &= \sum_{i=0}^{n} a_i y^i - \sum_{i=0}^{n} a_i x^i = \\ &= \sum_{i=0}^{n} a_i (y^i - x^i) = \\ &= \sum_{i=0}^{n} a_i (y - x)(\cdots) \geq \\ &\geq \sum_{i=0}^{n} a_i (y - x), \end{aligned}$$

where the expression in parenthesis is nonnegative.

11 Open problems

1. Are there examples of UCFG for which the induced equations do not have a unique solution?
2. Can we massage the UCFG in order to induce a system of polynomial equations with a unique solution?
3. Consequently, can we place the universality problem for UCFG in FIXP [6]?
4. For PPS, the decision problem $x_0^* \geq p$ is in PTIME in the unit cost RAM model. Can we do the same for UCFG?
5. Can one build a gadget of the form

   $$X \leftarrow L \mid ? \cdot X \cdot X$$

   such that $\mu(L) \geq \alpha$ iff $\mu(X) = 1$? In other words, if L contains sufficiently many words, then X is universal.

This rests on the analysis that the least nonnegative solution to $x = c + ax^2$ is 1 iff $a + c = 1$ and $0 \leq a \leq \frac{1}{2}$ (i.e., $\frac{1}{2} \leq c \leq 1$), where $c = \mu(L)$.

This would imply that UUCFL is SQRTSUM-hard. But it is also seem unlikely to be possible to do, due to the restrictions imposed by the grammar being unambiguous.

6. Universality of linear UCFG in PTIME? The probabilities are rational. Can they be computed exactly in PTIME?

7. Can we decide whether $\mu_G(L(A)) = 1$ for a SCFG G and a UFA A? Can we approximate that probability in PTIME? Cf. the analogous results [3] for a DFA A.

   Motivation: This would solve DCFG $\subseteq$ UFA in PTIME, since $L(G) \subseteq L(A)$ is equivalent to $\mu_G(L(A)) = 1$ if we interpret G as a SCFG (for any assignment of positive probabilities to its rules s.t. $\mu_G(\Sigma^*) = 1$, i.e., it terminates almost surely; how to find such probabilities?).

Deciding UUCFL requires at least to solve the following two non-trivial subcases:

1. Given two disjoint DCFL $A, B \subseteq \Sigma^*$, is $A \cup B = \Sigma^*$?

2. Universality of a strongly connected UCFG, i.e., every variable depends on every other variable.

References

[1] M. Debski. State complexity of complementing unambiguous automata. Master's thesis, University of Warsaw, Dec 2017.

[2] J. Esparza, A. Gaiser, and S. Kiefer. Computing Least Fixed Points of Probabilistic Systems of Polynomials. In J.-Y. Marion and T. Schwentick, editors, *STACS'10*, volume 5 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 359–370, Dagstuhl, Germany, 2010. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[3] K. Etessami, A. Stewart, and M. Yannakakis. Stochastic context-free grammars, regular languages, and newton's method. In *In Proc. of ICALP'13*, ICALP'13, pages 199–211, Berlin, Heidelberg, 2013. Springer-Verlag.

[4] K. Etessami, A. Stewart, and M. Yannakakis. A polynomial time algorithm for computing extinction probabilities of multitype branching processes. *SIAM Journal on Computing*, 46(5):1515–1553, 2017.

[5] K. Etessami and M. Yannakakis. Recursive markov chains, stochastic grammars, and monotone systems of nonlinear equations. *J. ACM*, 56(1):1:1–1:66, Feb. 2009.

[6] K. Etessami and M. Yannakakis. On the complexity of nash equilibria and other fixed points. *SIAM J. Comput.*, 39(6):2531–2597, Apr. 2010.

[7] W. E. Hardy G.H. *An introduction to the theory of numbers*. OUP, 6ed. edition, 2008.

[8] A. Heußner, J. Leroux, A. Muscholl, and G. Sutre. Reachability analysis of communicating pushdown systems. *LMCS*, 8(3):1–20, September 2012.

[9] T. N. Hibbard and J. Ullian. The independence of inherent ambiguity from complementedness among context-free languages. *J. ACM*, 13(4):588–593, Oct. 1966.

[10] R. E. Stearns and H. B. Hunt. On the equivalence and containment problems for unambiguous regular expressions, grammars, and automata. In *Proceedings of the 22nd Annual Symposium on Foundations of Computer Science*, SFCS '81, pages 74–81, Washington, DC, USA, 1981. IEEE Computer Society.

[11] A. Stewart, K. Etessami, and M. Yannakakis. Upper bounds for newton's method on monotone polynomial systems, and p-time model checking of probabilistic one-counter automata. *J. ACM*, 62(4):30:1–30:33, Sept. 2015.

[12] J. Swernofsky and M. Wehar. On the complexity of intersecting regular, context-free, and tree languages. In M. M. Halldórsson, K. Iwama, N. Kobayashi, and B. Speckmann, editors, *Automata, Languages, and Programming*, volume 9135 of *Lecture Notes in Computer Science*, pages 414–426. Springer Berlin Heidelberg, 2015.