An Application of Kolmogorov Complexity to Context Free Grammars
Exposition by William Gasarch

1 Introduction

Recall the definition of a Context Free Grammar (CFG) and of a Context Free Language (CFL).

Def 1.1 A *CFG* is a tuple $G = (N, \Sigma, R, S)$ such that the following holds:

- N is a finite set of *nonterminals*. These will be denoted by capital letters.
- Σ is a finite *alphabet*. We require $\Sigma \cap N = \emptyset$. These will be denoted by small letters.
- $R \subseteq N \times (N \cup \Sigma)^*$ and are called *Rules*. Here is an example of how we write the rules

$$A \rightarrow aBBaA$$

- $S \in N$, the *start symbol*.

Convention 1.2 We often just write the rules. The start symbol is S, the nonterminals are the capital letters mentioned, the alphabet is the small letters mentioned.

Notation 1.3 As usual e denotes the empty string.

Example 1.4

1. Let G be the CFG

$$S \rightarrow aSb \quad | \quad bSa \quad | \quad SS \quad | \quad e$$

Our interest is in what strings of terminals can be generated. Here that set is

$$\{w : \#_a(w) = \#_b(w)\}$$

where $\#_\sigma(w)$ is the number of σ's in w.

2. $S \to S_1 S_1$
   $S_1 \to S_2 S_2$
   $S_2 \to S_3 S_3$
   $S_3 \to a$
   The only string this can generate is a^8.

Notation 1.5 Let G be a CFG with start symbol S.

1. Let A be a nonterminal. Then

$$A \Rightarrow \alpha$$

   means that if you start from A and apply the rules you can get to α. Note that α may contain both terminals and nonterminals.

2. Recall that S is the start nonterminal.

$$L(G) = \{w : S \Rightarrow w \wedge w \in \Sigma^*\}$$

We can now finally define a context free language

Def 1.6 L is a CFL if there exists a CFG G such that $L = L(G)$.

We will be looking at CFG's of a particular form.

Def 1.7 A CFG G is in *Chomsky Normal Form* if the rules are all of the following form:

1. $A \to BC$ where $A, B, C \in N$ (nonterminals).
2. $A \to \sigma$ (where $A \in N$ and $\sigma \in \Sigma$).
3. $S \to e$ (where S is the start symbol and e is the empty string).

Notation 1.8 We use the notation CNF CFG to mean a CFG in Chomsky Normal Form. Do not confuse this use of CNF with Conjunctive Normal Form.

The following is true though we are not going to prove it.

Def 1.9 If L is a CFL then there exists a CNF CFG G such that $L = L(G)$.

2 Sizes of CFGs

Def 2.1 Let G be a CNF CFG. The *size of* G is the number of rules in G.

Fact 2.2 *If G is a CNF CFG of size s then it has at most $3s$ nonterminals. Note that each one can be represented with $\lg(s) + O(1)$ bits.*

Theorem 2.3 *Let $L = \{0^n\}$. There is a CNF CFG G of size $\lg(n) + O(1)$.*

Proof:

We will assume n is a power of 2 and that $\ell = \lg(n)$.
Let S_0 be the start symbol.
Here is the CNF CFG:
$S_0 \rightarrow S_1 S_1$
$S_1 \rightarrow S_2 S_2$
$\vdots$
$S_{\ell-1} \rightarrow S_\ell S_\ell$
$S_L \rightarrow 0$.
Clearly, for $1 \le i \le L - 1$, $S_0 \Rightarrow S_i^{2^i}$
Hence

$$S_0 \Rightarrow S_\ell^{2^\ell} \Rightarrow 0^{2^\ell} = 0^n.$$

The number of rules is $\ell + 1 = \lg(n) + O(1)$.
▮

Exercise 1 Show that *any* CNF CFG for $\{0^n\}$ requires $\Omega(\lg(n))$ rules.

Theorem 2.4 *Let $n \in \mathsf{N}$ be of the form $\frac{m^2+3m}{2}$. Let*

$$w = 10^1 10^2 10^3 \cdots 10^m$$

Note that

$$|w| = m + 1 + 2 + \cdots + m = m + \frac{m(m+1)}{2} = \frac{m^2+3m}{2} = n.$$

Let $L = \{w\}$. There is a CNF CFG G of size $O(\sqrt{n}\log n)$.

Proof:

We first give a grammar that is not in Chomsky Normal Form.

The first rule is:

$$S \rightarrow 1A_1 1A_2 \cdots 1A_m$$

For $1 \le i \le m$ have the CNF CFG with start symbol A_i that generates 0^i and is of size $\lg(i) + O(1)$.

Since A_i has $\lg(i)$ rules, all of the A_i-grammars add up to have $\lg(1) + \cdots + \lg(m) = O(m \log m)$ rules.

We then take the rules

$$S \rightarrow 1A_1 1A_2 \cdots 1A_m$$

and break it into $O(m)$ rules of the right form.

Hence the final grammar is of size $O(m \log m) = O(\sqrt{n}\log n)$.

∎

Open Problem 2.5 Let L be as in Theorem 2.4. Prove or disprove that there is a smaller grammar for L than $O(\sqrt{n}\log n)$.

3 Short Introduction to Kolmogorov Complexity

Intuitively the string 000000000000000000000 does not seem random. How to make this rigorous? Note that there is a program of length $\lg n + O(1)$ that prints out 0^n:

$$\texttt{for } i = 1 \texttt{ to } n \texttt{ print(0)}$$

Conversely, the string 01101000110000001110101010001100 does seem random. The shortest program to print it out might be

```
print(01101000110000001110101010001100)
```

which is roughly the length of the string itself.

Taking a cue from the above two examples, we will define the *randomness of a string* x to be the size of the shortest program that prints x.

Def 3.1 Fix a programming language (we will later see that the definition is largely independent of the choice of programming language).

1. If $w \in \{0,1\}^n$ then $C(x)$ is the length of the shortest program that, on input e, prints out x. Note that $C(x) \le n + O(1)$.
2. If $w \in \{0,1\}^n$ then $C(x|y)$ is the length of the shortest program that, on input y, prints out x. Note that $C(x|y) \le n + O(1)$.
3. A string is *Kolmogorov random* if $C(x) \ge n$. A string is *Kolmogorov random relative to* y if $C(x|y) \ge n$.

We note some facts about C.

Note 3.2 Let y be a string.

1. If C_1 is defined using one programming language, and C_2 is defined using another programming language, then, for all w, $C_1(w)$ and $C_2(w)$ differ by a constant.
2. There exists a string that is Kolmogorov random relative to y. This is a counting argument and is nonconstructive.
3. Most strings of length n are Kolmogorov random relative to y. This is the same counting argument used to show that such strings exist.

4 Is There a w such that $\{w\}$ requires a large CNF CFG?

Theorem 4.1 *Let w be of length n. Then there exists a CNF CFG of size $2n-1$ for $\{w\}$.*

Proof:

Let $w = w_1 \cdots w_n$.
Here is the CNF CFG for $\{w\}$
$S \rightarrow W_1 U_1$
$U_1 \rightarrow W_2 U_2$
$U_2 \rightarrow W_3 U_3$.
$\vdots$
$U_{n-2} \rightarrow W_{n-1} W_n$.
$W_1 \rightarrow w_1$
$W_2 \rightarrow w_2$
$\vdots$
$W_n \rightarrow w_n$
This CNF CFG has $2n-1$ rules.

∎

Is There a w such that $\{w\}$ Requires a large CNF CFG?
Yes.

Theorem 4.2 *Let w be a Kolmogorov random string of length n. Any CNF CFG for $\{w\}$ has size at least $\Omega(\frac{n}{\lg(n)})$.*

Proof:

Let G be a CNF CFG for $\{w\}$ with r rules. We will assume r is a power of 2. From this we will obtain a description of w.

Since there are r rules there are at most $3r$ nonterminals. Hence each nonterminal can be expressed with $\lg(r) + O(1)$ bits. Hence to describe the entire grammar takes at most $r\lg(r) + O(r)$ bits.

From the grammar you can obtain the string w by generating strings in all possible ways until you get one that is all terminals.

Since w is Kolmogorov random

$$r\lg(r) + O(r) \geq n$$

We leave it as an exercise to show this implies $r = \Omega(\frac{n}{\lg(n)})$. ∎

Open Problem 4.3 Is there a constructive proof that there is a string w such that $\{w\}$ requires a large CNF CFG?

5 The Most General Theorem on This Topic

The main result so far is that there is a string w such that any CNF CFG G for $\{w\}$ requires $= \Omega(\frac{n}{\log n})$ rules. What if the CFG is not in CNF? What if its not even a CFG? What if we seek w such that $\{w\}$ requires (say) $\sqrt{n}$ rules? In this section we answer such questions.

Def 5.1 A *Context Sensitive Grammar (CSG)* is a tuple $G = (N, \Sigma, R, S)$ such that the following holds:

1. N is a finite set of *nonterminals*. These will be denoted by capital letters.
2. Σ is a finite *alphabet*. We require $\Sigma \cap N = \emptyset$. These will be denoted by small letters.
3. $R \subseteq (\Sigma \cup N)^*N(\Sigma \cup N)^* \times (N \cup \Sigma)^*$ and are called *Rules*. Here is an example of how we write the rules

$$aAbB \to aBBaA$$

4. $S \in N$, the *start symbol*.

Def 5.2

1. Let G be a CSG. If $A \in N$ then $L(A)$ is defined similarly to how it was for a CFG.
2. L is a *Context Sensitive Language (CSL)* if there exists a CSG G such that $L = L(G)$.

Def 5.3 Let $f(n)$ be a monotone non-decreasing function (so it could be constant) such that $3 \leq f(n) \leq n$. Let w be a string of length n.

1. An *f-CFG for w* is a CFG where (1) every rule has $\le f(n)$ symbols, and (2) $L(G) = \{w\}$.

2. An *f-CSG for w* is a CSG where (1) every rule has $\le f(n)$ symbols, and (2) $L(G) = \{w\}$.

Fact 5.4 *Let w, f, n be as in Definition 5.3. Let G be an f-CSG for $\{w\}$ with r rules. Then G has at most $r \times f(n)$ nonterminals.*

Theorem 5.5 *Let f be a monotone non-decreasing function (so it may be constant) such that $3 \le f(n) \le n$. Let w be a string of length n. Then there is an f-CFG for $\{w\}$ of size $O(\frac{n}{f(n)})$.*

Proof:

The first rule is
$S \rightarrow A_1 \cdots A_{f(n)}$.
We call the A_i's *level-1 nonterminals*.

For each A_i we have a rule that takes it to $f(n)$ new nonterminals. We call these new nonterminals *level-2 nonterminals*.

We keep going. The level i nonterminals each go to $f(n)$ level $i+1$ nonterminals. The first time that there are $\ge \frac{n}{f(n)}$ level i nonterminals, instead of mapping to another level of nonterminals, we would have the first $\frac{n}{f(n)}$ of those nonterminals go to blocks of at most $f(n)$ letters of w in order, and have the remaining level-i nonterminals go to e.

We leave it to the reader to show that there are $O(\frac{n}{f(n)})$ rules. ∎

Theorem 5.6 *Let $f(n)$ be a monotone non-decreasing function (so it could be constant) such that $3 \le f(n) \le n$. Let $g(n)$ be a computable monotone increasing function such that $3 \le g(n) \le n$. There exists a string w such that the following hold.*

1. *There is an f-CFG for $\{w\}$ of size $O(\frac{g(n)}{f(n)} + \lg(n))$.*

2. *If G is an f-CSG for $\{w\}$ of size r then*

$$r = \Omega\Big(\frac{g(n)^{1-o(1)}}{f(n)}\Big).$$

3. If $f = O(1)$ then one can obtain

$$r = \Omega\Big(\frac{g(n)}{\log n}\Big).$$

Proof:

Let w' be a Kolmogorov random string of length $g(n)$ relative to n. Let $w = w'0^{n-g(n)}$.

1) We form the f-CFG for $\{w\}$ as follows.

From Theorem 4.1 there is an f-CFG of size for $\{w'\}$ of size

$$O\Big(\frac{g(n)}{f(n)}\Big).$$

From Theorem 2.3 there is a 3-CFG for $0^{n-g(n)}$ of size

$$\leq \lg(n - g(n)) + O(1) \leq \lg(n) + O(1).$$

These two CFGs can easily be combined to obtain an f-CFG for $\{w\}$ of size

$$O\Big(\frac{g(n)}{f(n)} + \lg(n)\Big).$$

2) We show that any f-CSG for $\{w\}$ has a large size.

Let G be an f-CSG for $\{w\}$ with r rules. From G one can easily obtain a description of w: generate strings with G until a string of terminals appears, and that's w. From w, the Turing machine for g (which is of size $O(1)$), and n, one easily obtains a description of w': Take w and strip off the last $n - g(n)$ 0's. In short, w' can be described from G.

Since G has r rules, G has at most $rf(n)$ nonterminals. Hence each nonterminal can be expressed with $\lg(rf(n)) + O(1)$ bits. Hence to describe the G takes at most

$$rf(n)(\lg(rf(n)) + O(1)) = O(rf(n)\lg(rf(n))) \text{ bits }.$$

Since w' is a Kolmogorov random string of length $g(n)$,

$$g(n) \leq O(rf(n)\lg(rf(n)))$$

Let $\epsilon > 0$. We show that, for large enough n,

$$r = \Omega\Big(\frac{g(n)^{1-\epsilon}}{f(n)}\Big)$$

Let δ be such that $\frac{1}{1+\delta} = 1 - \epsilon$. For large enough n we have

$$g(n) \le O(rf(n)\lg(rf(n))) \le O((rf(n))^{1+\delta})$$

Hence

$$r = \Omega\Big(\frac{g(n)^{1/(1+\delta)}}{f(n)}\Big).$$

Hence

$$r = \Omega\Big(\frac{g(n)^{1-\epsilon}}{f(n)}\Big)$$

3) We leave it to the reader to show, using

$$g(n) \le O(rf(n)\lg(rf(n)))$$

and $f(n) = O(1)$, that $r = \Omega(\frac{g(n)}{\log n})$.

∎

6 Open Questions

We gather up all of the open problems we have come across in this paper, even those we already stated.

Recall that *size* means *number of rules.*

1. Theorem 2.4 gives a string w such that there is a CNF CFG for $\{w\}$ of size $O(\sqrt{n}\log n)$. Prove or disprove that there is a smaller CNF CFG for $\{w\}$.

2. Theorem 4.2 states that, for all n, there is a string w of length n such that any CNF CFG for $\{w\}$ has size at least $\Omega(\frac{n}{\log n})$. The proof is nonconstructive. Can the proof be made constructive? Formally, is there a poly time program P such that $P(0^n)$ is a string w of length n such that any CNF CFG for $\{w\}$ has size at least $\Omega(\frac{n}{\log n})$? Perhaps we will get a not-as-good bound that is constructive.

3. A corollary to Theorem 5.6 is that, for all n, there is a string w of length n such that any CNF CFG for $\{w\}$ has size at least $\Omega(\frac{\sqrt{n}}{\log n})$. The proof is nonconstructive. Can the proof be made constructive?

4. In the last two open questions we asked for constructive proofs for strings w such that any CNF CFG has size at least $\Omega(\frac{n}{\log n})$ and at least $\Omega(\frac{\sqrt{n}}{\log n})$. One can replace n and $\sqrt{n}$ with any computable increasing function of n. Note that for $\log n$ we have an answer: take $w = 0^n$. How much higher than $\log n$ is it that there are no constructive proofs?

5. Are there strings w such that the smallest CSG for $\{w\}$ is much smaller than the smallest CFG for $\{w\}$? As a concrete example, is there a CSL of size $\ll \log n$ for 0^n?

7 Acknowledgments

We thank Ming Li, Erika Melder, and Paul Vitanyi for helpful discussions.