HW11 Solution

CFG Comp is Undecidable

1) $\overline{ACC_e}$ is a CFL:

CFG Comp is Undecidable

1) $\overline{ACC_e}$ is a CFL:

For $\overline{\mathrm{ACC}_{e,x}}$ we had the set of strings
w's prefix is NOT $\#x(s,\#)\#^*\$$.

CFG Comp is Undecidable

1) $\overline{ACC_e}$ is a CFL:

For $\overline{\mathrm{ACC}_{e,x}}$ we had the set of strings
w's prefix is NOT $\#x(s,\#)\#^*\$$.

For ACC_e we replace x with ANY elements of Σ^*. Hence
w's prefix is NOT $\#\Sigma^*(s,\#)\#^*\$$.

CFG Comp is Undecidable (cont)

INF is $\{e : M_e$ accepts an infinite number of inputs $\}$
2) Show: If $e \in \mathrm{INF}$ then ACC_e is NOT a CFL.

CFG Comp is Undecidable (cont)

INF is $\{e : M_e$ accepts an infinite number of inputs $\}$
2) Show: If $e \in \mathrm{INF}$ then ACC_e is NOT a CFL.
Omitted

CFG Comp is Undecidable (cont)

3) Show that if $e \notin \mathrm{INF}$ then ACC_e IS a CFL.

If $e \notin \mathrm{INF}$ then ACC_e is FINITE, hence a CFL.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

- Input e. Create a CFG G for $\overline{\mathrm{ACC}_e}$.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

- Input e. Create a CFG G for $\overline{\mathrm{ACC}_e}$.
- Use the algo for CFG-COMP to determine if $\overline{L(G)} = \mathrm{ACC}_e$ is a CFL.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

- Input e. Create a CFG G for $\overline{\mathrm{ACC}_e}$.
- Use the algo for CFG-COMP to determine if $\overline{L(G)} = \mathrm{ACC}_e$ is a CFL.
- If $\overline{L(G)}$ IS a CFL then $e \notin \mathrm{INF}$, so output NOT and halt.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

- Input e. Create a CFG G for $\overline{\mathrm{ACC}_e}$.
- Use the algo for CFG-COMP to determine if $\overline{L(G)} = \mathrm{ACC}_e$ is a CFL.
- If $\overline{L(G)}$ IS a CFL then $e \notin \mathrm{INF}$, so output NOT and halt.
- If $\overline{L(G)}$ IS NOT a CFL then $e \in \mathrm{INF}$, so output YES and halt.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

- Input e. Create a CFG G for $\overline{\mathrm{ACC}_e}$.
- Use the algo for CFG-COMP to determine if $\overline{L(G)} = \mathrm{ACC}_e$ is a CFL.
- If $\overline{L(G)}$ IS a CFL then $e \notin \mathrm{INF}$, so output NOT and halt.
- If $\overline{L(G)}$ IS NOT a CFL then $e \in \mathrm{INF}$, so output YES and halt.

$e \in \mathrm{INF} \implies \mathrm{ACC}_e$ not CFL $\implies \overline{L(G)} = \mathrm{ACC}_e$ NOT CFG.

CFG Comp is Undecidable (cont)

Show that if CFG-COMP is decidable then INF is decidable.

- Input e. Create a CFG G for $\overline{\mathrm{ACC}_e}$.
- Use the algo for CFG-COMP to determine if $\overline{L(G)} = \mathrm{ACC}_e$ is a CFL.
- If $\overline{L(G)}$ IS a CFL then $e \notin \mathrm{INF}$, so output NOT and halt.
- If $\overline{L(G)}$ IS NOT a CFL then $e \in \mathrm{INF}$, so output YES and halt.

$e \in \mathrm{INF} \implies \mathrm{ACC}_e$ not CFL $\implies \overline{L(G)} = \mathrm{ACC}_e$ NOT CFG.
$e \notin \mathrm{INF} \implies \mathrm{ACC}_e$ is CFL $\implies \overline{L(G)} = \mathrm{ACC}_e$ is a CFG.

Diophantine Sets

$$A = \left\{ x : \bigwedge_{i=1}^{k} x \equiv a_i \pmod{m_i} \right\}.$$

Diophantine Sets

$$A = \left\{ x : \bigwedge_{i=1}^{k} x \equiv a_i \pmod{m_i} \right\}.$$

$x \in A$ iff

$$(\exists y_1, \ldots, y_k)[(\sum_{i=1}^{k} (x - a_i - y_i m_i)^2 = 0)]$$

More Dio Sets

$p_1, \ldots, p_k$ are primes.

More Dio Sets

$p_1, \ldots, p_k$ are primes.

$$\bigwedge_{i=1}^{k} x \not\equiv 0 \pmod{p_i} \equiv \bigwedge_{i=1}^{k} \bigvee_{j=1}^{p_i-1} x \equiv j \pmod{p_i}$$

More Dio Sets

$p_1, \ldots, p_k$ are primes.

$$\bigwedge_{i=1}^{k} x \not\equiv 0 \pmod{p_i} \equiv \bigwedge_{i=1}^{k} \bigvee_{j=1}^{p_i - 1} x \equiv j \pmod{p_i}$$

Let $p_{i,j}(x, y_{i,j}) = (x - j + y_i p_i)$.

More Dio Sets

$p_1, \ldots, p_k$ are primes.

$$\bigwedge_{i=1}^{k} x \not\equiv 0 \pmod{p_i} \equiv \bigwedge_{i=1}^{k} \bigvee_{j=1}^{p_i-1} x \equiv j \pmod{p_i}$$

Let $p_{i,j}(x, y_{i,j}) = (x - j + y_i p_i)$.

Let $p_i(x, y_{i,1}, y_{i,2}, \ldots, y_{i,p_i-1}) = \prod_{j=1}^{p_i-1}(x - p_i y_{i,j} + j)$

More Dio Sets

$p_1, \ldots, p_k$ are primes.

$$\bigwedge_{i=1}^{k} x \not\equiv 0 \pmod{p_i} \equiv \bigwedge_{i=1}^{k} \bigvee_{j=1}^{p_i-1} x \equiv j \pmod{p_i}$$

Let $p_{i,j}(x, y_{i,j}) = (x - j + y_i p_i)$.

Let $p_i(x, y_{i,1}, y_{i,2}, \ldots, y_{i,p_i-1}) = \prod_{j=1}^{p_i-1}(x - p_i y_{i,j} + j)$

The final polynomial is

$$\sum_{i=1}^{k} p_i(x, y_{i,1}, \ldots, y_{i,p_i-1})^2$$

Horse Number Variant

For $n \geq 2$. $B(n)$: numb of ways that n horses, $x_1, \ldots, x_n$, can finish a race (equalities allowed) such that $x_1 < x_2$.

Horse Number Variant Case 1

Case 1 x_1 is one of the mins. x_2 CANNOT be a min. For $0 \leq i \leq n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to also be mins.

Horse Number Variant Case 1

Case 1 x_1 is one of the mins. x_2 CANNOT be a min. For $0 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to also be mins.

This can be done in $\binom{n-2}{i}$ ways.

Horse Number Variant Case 1

Case 1 x_1 is one of the mins. x_2 CANNOT be a min. For $0 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to also be mins.

This can be done in $\binom{n-2}{i}$ ways.
Then there are $n-i-1$ left which can be ordered in $H(n-i-1)$ ways.

Horse Number Variant Case 1

Case 1 x_1 is one of the mins. x_2 CANNOT be a min. For $0 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to also be mins.

This can be done in $\binom{n-2}{i}$ ways.
Then there are $n-i-1$ left which can be ordered in $H(n-i-1)$ ways.

$$\sum_{i=0}^{n-2} \binom{n-2}{i} H(n-i-1)$$

Horse Number Variant Case 2

Case 2 x_1 is NOT one of the mins.

Horse Number Variant Case 2

Case 2 x_1 is NOT one of the mins.
For $1 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to be mins.

Horse Number Variant Case 2

Case 2 x_1 is NOT one of the mins.
For $1 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to be mins.

This can be done in $\binom{n-2}{i}$ ways.

Horse Number Variant Case 2

Case 2 x_1 is NOT one of the mins.
For $1 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to be mins.

This can be done in $\binom{n-2}{i}$ ways.

Then there are $n-i$ left which can be ordered in $B(n-i)$ ways. So

$$\sum_{i=1}^{n-2} \binom{n-2}{i} B(n-i)$$

Horse Number Variant Case 2

Case 2 x_1 is NOT one of the mins.
For $1 \le i \le n-2$ choose i of $\{x_3, x_4, \ldots, x_n\}$ to be mins.

This can be done in $\binom{n-2}{i}$ ways.

Then there are $n-i$ left which can be ordered in $B(n-i)$ ways. So

$$\sum_{i=1}^{n-2} \binom{n-2}{i} B(n-i)$$

So the total is

$$B(n) = \sum_{i=0}^{n-2} \binom{n-2}{i} H(n-i-1) + \sum_{i=1}^{n-2} \binom{n-2}{i} B(n-i)$$

CFG for Singleton Sets

G is a CFL then $L(G)$ is the set of strings that G generates.
$\Sigma = \{a, b\}$.

CFG for Singleton Sets

G is a CFL then $L(G)$ is the set of strings that G generates.
$\Sigma = \{a, b\}$.

Show that there is a CFL G in Chomsky normal form with $L(G) = \{a^n\}$ with $O(\log n)$ rules.

CFG for Singleton Sets

G is a CFL then $L(G)$ is the set of strings that G generates.
$\Sigma = \{a, b\}$.

Show that there is a CFL G in Chomsky normal form with $L(G) = \{a^n\}$ with $O(\log n)$ rules.

Omitted- did it earlier in the semester.

CFG for Singleton Sets

w is Kolm-rand string of length n.

CFG for Singleton Sets

w is Kolm-rand string of length n.

Let G be a CFL in Chomsky Normal Form such that $L(G) = \{w\}$.

CFG for Singleton Sets

w is Kolm-rand string of length n.

Let G be a CFL in Chomsky Normal Form such that $L(G) = \{w\}$.

Show that Then G has at least $\Omega(n^{0.9})$ rules.

CFG for Singleton Sets

w is Kolm-rand string of length n.

Let G be a CFL in Chomsky Normal Form such that $L(G) = \{w\}$.

Show that Then G has at least $\Omega(n^{0.9})$ rules.

Hint If a CFL has R rules then it has at most $3R$ nonterminals. In this case each nonterminal can be represented with $O(\log R)$ bits. Hence the size of the CFL is $O(R \log R)$ bits.

CFG for $\{w\}$

The following program outputs w.

CFG for $\{w\}$

The following program outputs w.
For $x \in \{a, b\}^*$ (in lex order)

CFG for $\{w\}$

The following program outputs w.
For $x \in \{a, b\}^*$ (in lex order)

1. Run the Algorithm to test if $x \in L(G)$.

CFG for $\{w\}$

The following program outputs w.
For $x \in \{a, b\}^*$ (in lex order)

1. Run the Algorithm to test if $x \in L(G)$.
2. If it says YES then output x.

CFG for $\{w\}$

The following program outputs w.
For $x \in \{a, b\}^*$ (in lex order)

1. Run the Algorithm to test if $x \in L(G)$.
2. If it says YES then output x.
3. If not then go to the next x.

CFG for $\{w\}$

The following program outputs w.
For $x \in \{a,b\}^*$ (in lex order)

1. Run the Algorithm to test if $x \in L(G)$.
2. If it says YES then output x.
3. If not then go to the next x.

Since $L(G) = \{w\}$ this algorithm will eventually output w.

CFG for $\{w\}$

The following program outputs w.
For $x \in \{a, b\}^*$ (in lex order)

1. Run the Algorithm to test if $x \in L(G)$.
2. If it says YES then output x.
3. If not then go to the next x.

Since $L(G) = \{w\}$ this algorithm will eventually output w.
How big is the program?

CFG for $\{w\}$

The program needs to have the rules but not much else. Hence the length of the program is $O(R \log R)$.

CFG for $\{w\}$

The program needs to have the rules but not much else. Hence the length of the program is $O(R \log R)$.

Since w is Kolmogorov Random of length n,

CFG for $\{w\}$

The program needs to have the rules but not much else. Hence the length of the program is $O(R \log R)$.

Since w is Kolmogorov Random of length n,

$$n \leq O(R \log R)$$

CFG for $\{w\}$

The program needs to have the rules but not much else. Hence the length of the program is $O(R \log R)$.

Since w is Kolmogorov Random of length n,

$$n \leq O(R \log R)$$

Assume, BWOC that $R < O(n^{0.9})$. Then

CFG for $\{w\}$

The program needs to have the rules but not much else. Hence the length of the program is $O(R \log R)$.

Since w is Kolmogorov Random of length n,

$$n \leq O(R \log R)$$

Assume, BWOC that $R < O(n^{0.9})$. Then

$$n \leq O(R \log R) < O(n^{0.9} \log n^{0.9}) = O(n^{0.9} \log n).$$

CFG for $\{w\}$

The program needs to have the rules but not much else. Hence the length of the program is $O(R \log R)$.

Since w is Kolmogorov Random of length n,

$$n \leq O(R \log R)$$

Assume, BWOC that $R < O(n^{0.9})$. Then

$$n \leq O(R \log R) < O(n^{0.9} \log n^{0.9}) = O(n^{0.9} \log n).$$

This is a contradiction. Hence $R \geq \Omega(n^{0.9})$.