Final

March 17, 2017

(804501476)

Problem 1.0:
Prove

$$L_1 \diamond L_2 = \{xy \mid x \in L_1, y \in L_2, \text{ and } |x| = 2|y|\} \tag{1.1}$$

is not context free.
Let $L_1 = \{0^{2n}1^{2n}\}$ and $L_2 = \{0^n1^n\}$. Then,

$$L_1 \diamond L_2 = \{0^{2n}1^{2n}0^n1^n \mid x \in L_1, y \in L_2, \text{ and } |x| = 2|y|\} \tag{1.2}$$

Proof.
Towards contradiction assume $L_1 \diamond L_2$ is context-free.

- By the pumping lemma $\exists$ pumping length p.
- Let $w = 0^{2p}1^{2p}0^p1^p \in L_1 \diamond L_2$ and $|w| \geq p$.
- By pumping lemma $0^{2p}1^{2p}0^p1^p = abcde$ s.t:
  1. $|bd| \geq 1$
  2. $|bcd| \leq p$

Case 1: $bcd = 0^\alpha 1^\beta$ (on the left side)

- We pump down then we have either:
  1. $ace = 0^{2p-\alpha}1^{2p}0^p1^p \notin L_1 \diamond L_2$, since $2p - \alpha + 2p = 4p \implies \alpha = 0$ and $1 \leq \alpha \leq p$ $\Longrightarrow\Longleftarrow$
  2. $ace = 0^{2p}1^{2p-\beta}0^p1^p \notin L_1 \diamond L_2$, since $2p - \beta + 2p = 4p \implies \beta = 0$ and $1 \leq \beta \leq p$ $\Longrightarrow\Longleftarrow$
  3. $ace = 0^{2p-\alpha}1^{2p-\beta}0^p1^p \notin L_1 \diamond L_2$, since $2p - \alpha + 2p - \beta = 4p \implies \alpha + \beta = 0$ and $1 \leq \alpha + \beta \leq p \Longrightarrow\Longleftarrow$

Case 2: $bcd = 0^\alpha 1^\beta$ (on the right side)

- We pump up then we have either:
  1. $ace = 0^{2p}1^{2p}0^{p+\alpha}1^p \notin L_1 \diamond L_2$, since $2(p + \alpha + p) = 4p \implies \alpha = 0$ and $1 \leq \alpha \leq p$ $\Longrightarrow\Longleftarrow$
  2. $ace = 0^{2p}1^{2p}0^p1^{p+\beta} \notin L_1 \diamond L_2$, since $2(p + \beta + p) = 4p \implies \beta = 0$ and $1 \leq \beta \leq p$ $\Longrightarrow\Longleftarrow$

Problem 1.0 continued on next page...

3. $ace = 0^{2p}1^{2p}0^{p+\alpha}1^{p+\beta} \notin L_1 \diamond L_2$, since $2(p+\alpha+p+\beta) = 4p \implies \alpha+\beta = 0$ and $1 \leq \alpha+\beta \leq p \Longrightarrow\Longleftarrow$

Case 3: $bcd = 1^{\alpha}0^{\beta}$ (middle)

- We pump down then we have either:
  1. $ace = 0^{2p}1^{2p-\alpha}0^{p}1^{p} \notin L_1 \diamond L_2$, since $2p-\alpha+2p = 4p \implies \alpha = 0$ and $1 \leq \alpha \leq p$ $\Longrightarrow\Longleftarrow$
  2. $ace = 0^{2p}1^{2p}0^{p-\beta}1^{p} \notin L_1 \diamond L_2$, since $2(p-\beta+p) = 4p \implies \beta = 0$ and $1 \leq \beta \leq p$ $\Longrightarrow\Longleftarrow$
  3. $ace = 0^{2p}1^{2p-\alpha}0^{p-\beta}1^{p} \notin L_1 \diamond L_2$, since $2p-\alpha+2p = 2(p-\beta+p) \implies \beta = \alpha$. This is true if $\alpha = \beta = 0$ but $1 \leq \beta+\alpha \leq p \Longrightarrow\Longleftarrow$. We can also have that $\alpha = \beta$ is true if p is even and each is half of p. However this destroys symmetry in L_1 or L_2, $0^{2p}1^{2p-\alpha} \notin L_1$ or $0^{p-\beta}1^{p} \notin L_2 \Longrightarrow\Longleftarrow$.

□

Problem 2.0:

(a)

Show that

$$HALT = \{(\langle M \rangle, x) \mid M \text{ halts on input } x\} \tag{2.3}$$

is oracle decidable.

Proof.
We construct OBTM $O(\langle M \rangle, x)$:

- O writes $\langle M \rangle$ to machine tape and w to input tape.
- O enters query state:

  1: $x \in L(M)$ then accept.
  2: $x \notin L(M)$ the reject.

The query is immediate therefore if $x \notin L(M)$, we can reject without looping. Therefore O always terminates thus it is a decider for HALT. □

(b)

Show that

$$NEQ = \{(\langle M_1 \rangle, \langle M_2 \rangle) \mid L(M_1) \neq L(M_2)\} \tag{2.4}$$

is oracle recognizable.

Proof.
We construct OBTM $O(\langle M_1 \rangle, \langle M_2 \rangle)$:
Tapes:
In class we showed that a multiple tapes can be simulated with a single tape so we split the regular tape into 4 tapes w_1, w_2, w_3, and w_4.

1 Write $\langle M_1 \rangle$ onto w_1
2 Write $\langle M_2 \rangle$ onto w_2
3 Will keep a binary count starting at 0 in w_3.

- We are assuming that all strings can be converted to binary.

4 Will maintain a tuple starting at $(\$, \$)$ in w_4

Problem 2.0 continued on next page...

States:
We will have states S_1, S_{oracle}, S_3, S_4, q_{accept}

S_1: Write contents of tape w_1 onto the machine tape and contents of w_3 onto the input tape.

S_{oracle}: Enter query state. After the result has been written onto the input tape write the result onto w_4 and move head of w_4 right.

S_3: Clear the machine tape and write the contents of w_2 onto machine tape. Clear input tape and write w_3 onto input tape.

S_4: Reset tape w_4 to (\$,\$) and increment value of tape w_3 by one.

Transitions:

$$\delta(S_1, w_4 = (\$, \$)) \to (S_{oracle}) \tag{2.5}$$
$$\delta(S_{oracle}, w_4 = (x, \$)) \to (S_3),\ x \in \{0, 1\} \tag{2.6}$$
$$\delta(S_3, w_4 = (x, \$)) \to (S_{oracle}) \tag{2.7}$$
$$\delta(S_3, w_4 = (x, y)) \to (q_{accept}) \text{ if } x \neq y \tag{2.8}$$
$$\delta(S_3, w_4 = (x, y)) \to (S_4) \text{ if } x = y \tag{2.9}$$
$$\delta(S_4, w_4) \to (S_1) \tag{2.10}$$

This is still an OBTM as we have not changed the function of query and do not misuse the input and machine tapes. We make use of the regular tape like a tape of any TM and supplied states which allow us to recognize NEQ by determining whether a binary representations of a string is ever accepted by one an rejected by the other, if so we will accept. If not the machine will continue to increment counter and process the counter as strings and potentially loop if these two machines indeed accept the same language. □

(c)

The language:

$$Infinite = \{\langle M \rangle \mid |L(M)| = \infty\} \tag{2.11}$$

is not oracle recognizable. An OBTM that would try to recognize this language would have to check and infinite amount of strings to determine whether they all belong to M and so it would never halt.

Problem 3.0:

(a)

Show that

$$CLOSEBY = \{(\langle M_1 \rangle, \langle M_2 \rangle) \mid \forall x \in L(M_1)\ \exists y \in L(M_2) : ||x - y|| \leq 1\} \tag{3.12}$$

is undecidable.

Proof. Assume for contradiction $\exists$ a decider D for CLOSEBY, create a TM N:

- $N(w)$:
  - Let $u = \langle N \rangle$, by Recursion Theorem.
  - Let $\langle M \rangle$ be a TM that only accepts ε.
  - Run $D(u, \langle M \rangle)$:
    1: $D(u, \langle M \rangle)$: Accepts
       · Accept all w
    2: $D(u, \langle M \rangle)$: Rejects
       · Accept w iff $w = \varepsilon$

Analysis:

- Case 1: $D(u, \langle M \rangle)$: **Accepts** $\implies$ $L(N) = L(M) = \{\varepsilon\}$. This is true since the length of ε is zero $\implies$ the only string in $L(N)$ is ε. However we accept all w $\implies$ $L(N) = \{0, 1\}^*$ and this is contradiction.
- Case 2: $D(u, \langle M \rangle)$: **Rejects** $\implies$ $L(N) \neq L(M)$, since $L(M) = \{\varepsilon\}$. However we only accept ε $\implies$ $L(N) = \{\varepsilon\}$ and this is a contradiction.

□

Problem 3.0 continued on next page...

(b)

Show that

$$CLOSEBY = \{(\langle M_1 \rangle, \langle M_2 \rangle) \mid \forall x \in L(M_1)\ \exists y \in L(M_2) : ||x - y|| \leq 1\} \tag{3.13}$$

is unrecognizable.

Proof. Assume for contradiction $\exists$ a recognizer R for CLOSEBY, create a TM N:

- $N(w)$:
  - Let $u = \langle N \rangle$, by Recursion Theorem.
  - Let $\langle M \rangle$ be a TM that only accepts ε.
  - if $w = \varepsilon$ accept.
  - Run $R(u, \langle M \rangle)$:
    1: $R(u, \langle M \rangle)$: Accepts
       · Accept all w
    2: $R(u, \langle M \rangle)$: Rejects
       · Accept w iff $w = \varepsilon$

Analysis:

- Case 1: $R(u, \langle M \rangle)$: **Accepts** $\implies$ $L(N) = L(M) = \{\varepsilon\}$. This is true since the length of ε is zero $\implies$ the only string in $L(N)$ is ε. However we accept all w $\implies$ $L(N) = \{0,1\}^*$. $\Longrightarrow\Longleftarrow$
- Case 2: $R(u, \langle M \rangle)$: **Rejects** $\implies$ $L(N) \neq L(M)$, since $L(M) = \{\varepsilon\}$. However we only accept ε $\implies$ $L(N) = \{\varepsilon\}$. $\Longrightarrow\Longleftarrow$
- Case 3: $R(u, \langle M \rangle)$: **Loops** $\implies$ $L(N) \neq L(M)$, since $L(M) = \{\varepsilon\}$. But by construction $L(N) = \{\varepsilon\}$. $\Longrightarrow\Longleftarrow$

□

Problem 3.0 continued on next page...

(c)

Show that

$$LEQ - HALT = \{(\langle M \rangle, \langle N \rangle) \mid \forall x \in \Sigma^* : M(x) \text{ halts in fewer steps than } N(x)\} \tag{3.14}$$

is unrecognizable.

Proof. Assume for contradiction $LEQ - HALT$ is regular $\implies$ *exists* an enumerator E for $LEQ - HALT$. We construct M:

- M(w):
  - Let $z = \langle M \rangle$, by Recursion Theorem.
  - $task_A =$ run $E(\varepsilon)$.
  - $task_B =$ look at the sequence of produced by E. Wait until we find a tuple of form $(z, \langle N \rangle)$ is found.
  - run $task_A$ and $task_B$ in parallel.
    - When $task_B$ finishes, run N(w).

The enumerator is producing a sequence $(\langle M \rangle, \langle N \rangle) \in LEQ - HALT$. However by construction $\langle M \rangle$ now takes longer to halt than $\langle N \rangle$. $\implies\Longleftarrow$ □

Problem 4.0:
Show that

$$ALICE = \{(M, R) \mid (M, R) \text{ is recognizable}\} \tag{4.15}$$

is undecidable.

We can convert the rules for Alice in Turing Land into tiling rules and we also convert a Turing machine to follow the rules of the tiling problem.

Tiles: have the following elements within them

1 $w_i \in \{ON, OFF\}$

2 $q_i \in Q$ if head is in cell or ☻ if head is not in cell.

3 r, the result of i % M, where % is the modulo operator and i is the position of the head on the tape.

4 Boundary tiles which are marked by an X.

Rules:
Initial State: For simplicity assume $\llcorner = \llcorner \backslash ☻$

X	q_0 / $\llcorner$	$\llcorner$
X	X	X

q_0 / $\llcorner$	$\llcorner$	$\llcorner$
X	X	X

$\llcorner$	$\llcorner$	$\llcorner$
X	X	X

The rules $(b_1, \ldots, b_n) \to (c_1, \ldots, c_n)$ in Alice in Turing Land correspond to tiling rules where different n will create a different tiling size. These tiling rules correspond to a set of transitions for a TM. For simplicity assume that $w_i = w_i \backslash ☻, r$ we always start with the head in a position where $r = 0$.

Transition Right:

w_{i-1}	w_i	w'_{i+1}	w'_{i+2}	$\cdots$	w'_{i+n}
w_{i-1}	w_i	w'_{i+1}	w'_{i+2}	$\cdots$	q_{i+n} / w_{i+n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
w_{i-1}	w_i	w'_{i+1}	w'_{i+2}	$\cdots$	w_{i+n}
w_{i-1}	w_i	w'_{i+1}	q_{i+2} / w_{i+2}	$\cdots$	w_{i+n}
w_{i-1}	w_i	q_{i+1} / w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}
w_{i-1}	$0, q_i$ / w_i	w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i+1}, R) \\ \delta(w_{i+1}, q_{i+1}) \to (w'_{i+1}, q_{i+2}, R) \\ \delta(w_{i+2}, q_{i+2}) \to (w'_{i+2}, q_{i+3}, R) \\ \qquad\qquad \cdots \\ \delta(w_{i+(n-1)}, q_{i+(n-1)}) \to (w'_{i+(n-1)}, q_{i+n}, R) \\ \delta(w_{i+n}, q_{i+n}) \to (w'_{i+n}, q_{i+(n+1)}, R) \end{cases} \tag{4.16}$$

Problem 4.0 continued on next page...

Transition Left:

w'_{i-n}	$\cdots$	w'_{i-2}	w'_{i-1}	w_i	w_{i+1}
q_{i-n} / w_{i-n}	$\cdots$	w'_{i-2}	w'_{i-1}	w_i	w_{i+1}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
w_{i-n}	$\cdots$	w'_{i-2}	w'_{i-1}	w_i	w_{i+1}
w_{i-n}	$\cdots$	q_{i-2} / w'_{i-2}	w'_{i-1}	w_i	w_{i+1}
w_{i-n}	$\cdots$	w_{i-2}	q_{i-1} / w_{i-1}	w_i	w_{i+1}
w_{i-n}	$\cdots$	w_{i-2}	w_{i-1}	$0,\ q_i$ / w_i	w_{i+1}

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i-1}, L) \\ \delta(w_{i-1}, q_{i-1}) \to (w'_{i-1}, q_{i-2}, L) \\ \delta(w_{i-2}, q_{i-2}) \to (w'_{i-2}, q_{i-3}, L) \\ \qquad\qquad \cdots \\ \delta(w_{i-(n-1)}, q_{i-(n-1)}) \to (w'_{i-(n-1)}, q_{i-n}, L) \\ \delta(w_{i-n}, q_{i-n}) \to (w'_{i-n}, q_{i-(n+1)}, L) \end{cases} \tag{4.17}$$

Edge Cases:

Head in left corner

w_i	w'_{i+1}	w'_{i+2}	$\cdots$	w'_{i+n}
w_i	w'_{i+1}	w'_{i+2}	$\cdots$	q_{i+n} / w_{i+n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
w_i	w'_{i+1}	w'_{i+2}	$\cdots$	w_{i+n}
w_i	w'_{i+1}	q_{i+2} / w_{i+2}	$\cdots$	w_{i+n}
w_i	q_{i+1} / w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}
$0,\ q_i$ / w_i	w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i+1}, R) \\ \delta(w_{i+1}, q_{i+1}) \to (w'_{i+1}, q_{i+2}, R) \\ \delta(w_{i+2}, q_{i+2}) \to (w'_{i+2}, q_{i+3}, R) \\ \qquad\qquad \cdots \\ \delta(w_{i+(n-1)}, q_{i+(n-1)}) \to (w'_{i+(n-1)}, q_{i+n}, R) \\ \delta(w_{i+n}, q_{i+n}) \to (w'_{i+n}, q_{i+(n+1)}, R) \end{cases} \tag{4.18}$$

w_i	w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}
w_i	w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}
w_i	w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}
$0,\ q_i$ / w_i	w_{i+1}	w_{i+2}	$\cdots$	w_{i+n}

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i-1}, L) \\ \delta(w_{i-1}, q_{i-1}) \to (w'_{i-1}, q_{i-2}, L) \\ \delta(w_{i-2}, q_{i-2}) \to (w'_{i-2}, q_{i-3}, L) \\ \qquad\qquad \cdots \\ \delta(w_{i-(n-1)}, q_{i-(n-1)}) \to (w'_{i-(n-1)}, q_{i-n}, L) \\ \delta(w_{i-n}, q_{i-n}) \to (w'_{i-n}, q_{i-(n+1)}, L) \end{cases} \tag{4.19}$$

Head in right corner:

w'_{i-n}	$\cdots$	w'_{i-2}	w'_{i-1}	w_i
q_{i-n} / w_{i-n}	$\cdots$	w'_{i-2}	w'_{i-1}	w_i
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
w_{i-n}	$\cdots$	w'_{i-2}	w'_{i-1}	w_i
w_{i-n}	$\cdots$	q_{i-2} / w'_{i-2}	w'_{i-1}	w_i
w_{i-n}	$\cdots$	w_{i-2}	q_{i-1} / w_{i-1}	w_i
w_{i-n}	$\cdots$	w_{i-2}	w_{i-1}	$0,\ q_i$ / w_i

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i-1}, L) \\ \delta(w_{i-1}, q_{i-1}) \to (w'_{i-1}, q_{i-2}, L) \\ \delta(w_{i-2}, q_{i-2}) \to (w'_{i-2}, q_{i-3}, L) \\ \qquad\qquad \cdots \\ \delta(w_{i-(n-1)}, q_{i-(n-1)}) \to (w'_{i-(n-1)}, q_{i-n}, L) \\ \delta(w_{i-n}, q_{i-n}) \to (w'_{i-n}, q_{i-(n+1)}, L) \end{cases} \tag{4.20}$$

w_{i-n}	$\cdots$	w_{i-2}	w_{i-1}	w_i
w_{i-n}	$\cdots$	w_{i-2}	w_{i-1}	w_i
w_{i-n}	$\cdots$	w_{i-2}	w_{i-1}	w_i
w_{i-n}	$\cdots$	w_{i-2}	w_{i-1}	0, q_i / w_i

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i+1}, R) \\ \delta(w_{i+1}, q_{i+1}) \to (w'_{i+1}, q_{i+2}, R) \\ \delta(w_{i+2}, q_{i+2}) \to (w'_{i+2}, q_{i+3}, R) \\ \qquad\qquad \cdots \\ \delta(w_{i+(n-1)}, q_{i+(n-1)}) \to (w'_{i+(n-1)}, q_{i+n}, R) \\ \delta(w_{i+n}, q_{i+n}) \to (w'_{i+n}, q_{i+(n+1)}, R) \end{cases} \tag{4.21}$$

Head occupying corner after transition:

r, q_{i+n} / w_{i+n}	$w_{i+(n+1)}$	$w_{i+(n+2)}$	$\cdots$	$w_{i+(n+m)}$
w_{i+n}	$w_{i+(n+1)}$	$w_{i+(n+2)}$	$\cdots$	$w_{i+(n+m)}$

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i+1}, R) \\ \delta(w_{i+1}, q_{i+1}) \to (w'_{i+1}, q_{i+2}, R) \\ \delta(w_{i+2}, q_{i+2}) \to (w'_{i+2}, q_{i+3}, R) \\ \qquad\qquad \cdots \\ \delta(w_{i+(n-1)}, q_{i+(n-1)}) \to (w'_{i+(n-1)}, q_{i+n}, R) \end{cases} \tag{4.22}$$

$w_{i-(n+m)}$	$\cdots$	$w_{i-(n+2)}$	$w_{i-(n+1)}$	r, q_{i-n} / w_{i-n}
$w_{i-(n+m)}$	$\cdots$	$w_{i-(n+2)}$	$w_{i-(n+1)}$	w_{i-n}

$$\begin{cases} \delta(w_i, q_i) \to (w_i, q_{i-1}, L) \\ \delta(w_{i-1}, q_{i-1}) \to (w'_{i-1}, q_{i-2}, L) \\ \delta(w_{i-2}, q_{i-2}) \to (w'_{i-2}, q_{i-3}, L) \\ \qquad\qquad \dots \\ \delta(w_{i-(n-1)}, q_{i-(n-1)}) \to (w'_{i-(n-1)}, q_{i-n}, L) \end{cases} \tag{4.23}$$

In class we showed that if D decides the tiling then we can decide $M(\varepsilon)$ halts. Therefore since we can the rules of Alice in Turing Land into tiling rules and convert a Turing machine into a tiling instance then we cannot decide Alice in Turing land.

Problem 5.0:
First I will create a function

$$Pattern(i, j, b) \tag{5.24}$$

This function takes 3 parameters

1 i - the width of a block.

2 j - the length of a block.

3 b - the base we will use to produce the patterns.

It will produce patterns and map them to a block with dimensions $i \times j$. For example:

$$Pattern(2,2,2) = \left\{ \begin{array}{|c|c|}\hline 0&0\\\hline 0&0\\\hline\end{array} \begin{array}{|c|c|}\hline 1&0\\\hline 0&0\\\hline\end{array} \begin{array}{|c|c|}\hline 0&1\\\hline 0&0\\\hline\end{array} \begin{array}{|c|c|}\hline 1&1\\\hline 0&0\\\hline\end{array} \cdots \begin{array}{|c|c|}\hline 0&1\\\hline 1&1\\\hline\end{array} \begin{array}{|c|c|}\hline 1&1\\\hline 1&1\\\hline\end{array} \right\} \tag{5.25}$$

(a)

We perform triangle scheduling on the length of the rug vs the pattern of the carpet. The graph is numbered with the order in which we build a rug with a specific pattern. This allows us to not traverse one dimension up to ∞. In addition since i is countably infinite then $2^{1\times i}$ is also countably infinite, therefore we also will not weave rugs for length i for an infinite amount of time. Therefore we weave in one infinite day.

```
Weaver 1:
for i from 1 → ∞: // Tile size 1 × i
    for p in pattern(1, i, 2): /* Here we traverse the pattern encoded with binary digits*/
        weave rug with pattern p_i
```

Length																
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
n=4	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	$\cdots$
n=3	7	8	9	10	11	12	13	14								$\cdots$
n=2	3	4	5	6												$\cdots$
n=1	1	2														$\cdots$
P_i	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	$\cdots$

Problem 5.0 continued on next page...

(b)

We are extending the triangle scheduling to one more dimension j, the width of a rug. This triangle scheduling is on 3 dimensions. Creating a triangular plane and then we triangulate that plane. Once again this allows us to not spend all the time along one dimension and j is also countably infinite so $2^{i \times j}$ is also countably infinite. Therefore we can weave in one infinite day.

```
Weaver 2:
for i from 1 → ∞:
    for j from 1 → i
        for p in pattern(i, j, 2):
            weave rug for pattern p_i
```

(c)

Here we take the example in part a) and now extend the base chosen in the *pattern* function to be of the set c_i of colors. Here once again the colors are countably infinite so the possible patterns for a specific length i is also countably infinite. Since we are performing triangle scheduling then we can weave in one infinite day.

```
Weaver 3:
for i from 1 → ∞:
    for c in 1 → i: // Colors
        for p in pattern(1, i, c):
            weave rug for pattern p_i
```

(d)

We continue to extend the example to one more dimension. Here we add the weavers dimension and perform triangle scheduling. The number of weavers is countably infinite therefore we will not traverse the weaver dimension for an infinite amount of time.

```
Weaver 4:
for i from 1 → ∞: // i is the weaver w_i
    for j from 1 → i: // j is the width
        for k from 1 → j: // k is the length
            for c from 1 → k: // Colors
                for p in pattern(j, k, c):
                    weave rug for pattern p_i
```

Problem 6.0: Extra Credit

a

We can apply cantors diagonalization argument on the width of the rugs and show that the weaver will always miss a rug of a specific width.

	pos_1	pos_2	pos_3	pos_4	pos_5	pos_6	pos_7	$\cdots$
w_1	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	$\cdots$
w_2	x_{21}	x_{22}	x_{23}	x_{24}	x_{25}	x_{26}	x_{27}	$\cdots$
w_3	x_{31}	x_{32}	x_{33}	x_{34}	x_{35}	x_{36}	x_{37}	$\cdots$
w_4	x_{41}	x_{42}	x_{43}	x_{44}	x_{45}	x_{46}	x_{47}	$\cdots$
w_5	x_{51}	x_{52}	x_{53}	x_{54}	x_{55}	x_{56}	x_{57}	$\cdots$
w_6	x_{61}	x_{62}	x_{63}	x_{64}	x_{65}	x_{66}	x_{67}	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

Let

$$w_{Diag} = (x_{11} - 1)(x_{22} - 1)(x_{33} - 1)(x_{44} - 1)(x_{55} - 1) \tag{6.26}$$

Claim: w_{Diag} is not in the table.

Proof. Towards contradiction suppose w_{Diag} is in the table. Then $\exists\ i$ s.t $w^{'} = x_{1i}x_{2i}x_{3i}\cdots = w_{Diag}$. Look at position x_{ii}. If x_{ii} is in $w^{'}$ then it is not in w_{Diag}. $\Longrightarrow\Longleftarrow$ □

b

We can map $\mathbb{R} \to \mathbb{R}^2$ as such

$$Day = 0.a_1a_2a_3a_4a_5a_6\ldots \tag{6.27}$$
$$Width = 0.a_1a_3a_5\ldots \tag{6.28}$$
$$Length = 0.a_2a_4a_6\ldots \tag{6.29}$$
$$Day \to (Length, Width) \tag{6.30}$$

On a specific day we weave a rug with a length and width derived from the odd and even integers in the real representation of the day. We calculate the set of patterns for a set of colors as follows

$$c_k^{\lceil i \rceil \times \lceil j \rceil} \tag{6.31}$$

Then we perform the triangle scheduling.

Problem 6.0 continued on next page...

```
Weaver 4:
for i,j in day_k: // j is the width
        for c from 1 → i: // Colors
            for p in pattern(i, j, c):
                weave rug for pattern p_i
```

This will allow us to deliver the order in one infinite year.