Midterm 2 solutions

(1)

Let T be a binary tree that has a root, and assume that every node has exactly two children (note that that makes T infinite).

a. Show that the set of nodes in T is countable.

b. Using a diagonalization argument show that the set of all infinite paths from the root is uncountable.

Solution:

a.

To prove that the set of nodes is countable we define a correspondence $\mathbb{N} \to T$ by simply counting the nodes level by level, from left to right. We denote this mapping f, where $f(1)$ is the root, and recursively:

$f(2k) =$ Left child of $f(k)$

$f(2k+1) =$ Right child of $f(k)$

We can look at the binary length of the input k which is simply the level in which the node resides: k is mapped to the node at level $\lfloor \lg_2 k \rfloor$, node number $k - 2^{\lfloor \lg_2 k \rfloor}$ at that level (starting count at 0). The $\langle level, node \rangle$ result pair equals $\langle \lfloor \lg_2 k \rfloor, k - 2^{\lfloor \lg_2 k \rfloor} \rangle$.

Next we need to show it is a bijective mapping, however it is immediate: for different sources we always end up at different nodes, so it is one-to-one; for any node we can determine its source k by the level and location on that level, therefore it is onto.

b.

Following is a proof that the set of all infinite paths from the root is **uncountable**:

We can identify each path by an infinite sequence of $\{0,1\}$'s, where 0 indicates choosing the left child and 1 – the right.

Denote the set of paths P. Assume P is countable, then there is a correspondence $f: \mathbb{N} \to P$:

n	$f(n)$
1	0111010010101…
2	1100101000101…
3	0000101000011…

We define a sequence $b = \left(\overline{b_{0,0}}, \overline{b_{1,1}}, \dots\right)$ - each coordinate i is the negation of $f(i)$. But b must have a j such that $f(j) = b$, but then $b_{j,j} \neq b_{j,j}$ – a contradiction.

(2)

Show that the language $L2 = \{\langle A \rangle \mid A \text{ is a DFA that has no useless state}\}$ is decidable (a useless state is a state that is not entered for any input string).

Solution:

The idea: given a string s:

1. Decide whether s is an encoding of a DFA. If so, denote it the DFA A.
2. For all states σ in A do the following:

a. Make σ the only accepting state and call the resulting DFA B.

b. Use decider for emptiness of DFAs to decide whether $L(B) = \emptyset$.

3. If $L(B) \neq \emptyset$ for all σ, then none of the states is useless, so $s \in L_2$. Otherwise, $s \notin L_2$.

(3)

Show that the language $L_3 = \{\langle M, N\rangle \mid M, N \text{ } are \text{ } TMs \text{ } and \text{ } L(M) \subset L(N)\}$ is undecidable.

Solution:

By reduction from A_{TM}:

Given $\langle M, w\rangle$, build 2 TMs, $\langle M, N\rangle$ such that $L(M) \subset L(N) \Leftrightarrow M$ accepts w.

Let N be a TM such that $L(N) = \emptyset$. Next, we construct a machine R:

- Simulate M on w.
- If it accepts, $L(R) = \Sigma^*$, otherwise $L(R) = \emptyset$.

If M doesn't accept w, then $L(M) \subset L(N)$. If M accepts w, $L(M) = \Sigma^*$ is NOT a subset of $L(N)$.

We're actually showing a reduction from $\overline{A_{TM}}$, which is also undecidable, therefore L_4 is undecidable.

Alternative:

Reduce $E_{TM} \leq_m L_4$: define R as above, $L(N) = \emptyset$ and then M accepts $w \Leftrightarrow L(M) \subset L(N)$.

(4)

Show that the language $L_4 = \{\langle M\rangle \mid M \text{ } is \text{ } a \text{ } TM \text{ } and \text{ } |L(M)| = 1\}$ is not Turing-recognizable.

Solution:

Note: $\overline{A_{TM}}$ is not Turing-recognizable, so we can try to show $\overline{A_{TM}} \leq_m L_4$, or equivalently: $A_{TM} \leq_m \overline{L_4}$ where:

$\overline{L_4} = \{\langle M\rangle \mid M \text{ } is \text{ } a \text{ } TM \text{ } and \text{ } |L(M)| \neq 1\}$. We create a machine N:

- If the input is 1, accept.
- Simulate M on w.
- If M accepts w, then accept any input. Otherwise, reject (all but the input 1).

So $1 \in L(N)$ in any case, but if M accepts w there are even more words in the language. Then we get: M accepts $w \Leftrightarrow |L(N)| = |\Sigma^*| \neq 1$, as required.

(5)

Show that any infinite subset of MIN_{TM} is not Turing-recognizable.

Solution:

Let $MIN_{TM}^* \subset MIN_{TM}$ be an infinite subset. Assume (by contradiction) that MIN_{TM}^* **IS** Turing-recognizable, then MIN_{TM}^* can be enumerated by some enumerator E. From this point the proof is exactly like for MIN_{TM} in the book. we build C:

- Obtain $\langle C\rangle$.
- Use E to find a TM $D \in MIN_{TM}^*$ such that $|\langle C\rangle| < |\langle D\rangle|$.

- Simulate D.

Therefore the assumption that an enumerator E exists is false, and so MIN_{TM}^{*} is not Turing-recognizable.

(6)

a. Is the statement $\forall x \exists y[x \cdot y = 1]$ a member of $Th(\mathbb{N},\cdot)$?

b. Is the statement $\forall x \exists y[x \cdot y = 1]$ a member of $Th(\mathbb{Q},\cdot)$?

c. Give a formula that defines the usual relation $\leq$, "less than or equal to", in $(\mathbb{R},+,\cdot)$. The only relations you may use in your definition are "$+$" and "$\cdot$".

Solution:

a.

The statement is NOT a member of $Th(\mathbb{N},\cdot)$, because for $x = 2$ we have that $\forall y: 2y \neq 1$ (the idea: $\frac{1}{2} \notin \mathbb{N}$).

b.

The statement is NOT a member of $Th(\mathbb{N},\cdot)$, because for $x = 0$, $\forall y: oy \neq 1$.

c.

We express "$x \leq y$" in $(\mathbb{R},+,\cdot)$ as follows:

$$\exists z[x + z \cdot z = y]$$

$z \cdot z$ is the square of z, and all squares are non-negative.

Note:

We can express "$x < y$" by imposing $z = 0$ as follows: 0 is the only $z \in \mathbb{R}$ such that $z + z = z$, so:

$$\exists z[(z + z = z) \wedge (x + z \cdot z = y)]$$

Or alternatively just the first plus "$x \neq y$":

$$\exists z[x + z \cdot z = y] \wedge (x \neq y)$$

□

Group-work: understanding Theorem 6.12

$Th(\mathbb{N},+)$ is decidable.

The proof in the book (page 227) gives a decision procedure, and the construction is similar to exercise 1.32, a DFA that accepts all encodings of correct binary additions, for instance $\begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}1\\1\\0\end{bmatrix}$. It did it by remembering at each state whether the carry is 0 or 1, checking the correctness from the LSB toward the MSB.

Since the set of regular languages is closed under union, complementation and intersection, a combination of several such operations is still decidable with a DFA.

But, we have quantifiers in our formula: $\varphi = Q_1 x_1 \ldots Q_l x_l[\psi]$, where ψ is a quantifiers-free formula (that is discussed above and can be decided by a DFA).

We follow the construction of the book for $\forall x \exists y[y+y=x]$. This is a case where $l=2$ (we have 2 quantifiers).

Let $\varphi_i = Q_{i+1}x_{i+1} \dots Q_l x_l[\psi]$, then $\varphi_l = \psi$, and it has l free variables (no quantifiers to bound them).

Let A_i be a decider for φ_i and the input alphabet consists of i-tuples: $\Sigma_i = \left\{ \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_i \end{bmatrix}, \dots \right\}, b_j \in \{0,1\}$. For instance, $\Sigma_1 = \{0,1\}$,

$\Sigma_2 = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$.

The $\forall$ are taken care of by taking $\neg\exists\neg$ instead. The $\exists$ is taken care of by using a non-deterministic automata that guesses the additional coordinate $i+1$.