Week 12: Miscellaneous Problems

* 1. For each of the following statements about languages over the alphabet $\{0, 1\}^*$, determine if it is true or false.

(a) Every word of a regular language is of finite length.

(b) The language $\emptyset$ is regular.

(c) The language of all even length strings is regular.

(d) There is a language that is recognisable by an NFA but not recognisable by any DFA.

(e) Every finite subset of $\{0, 1\}^*$ is a regular language.

Solution

(a) True

(b) True

(c) True

(d) False

(e) True

* 2. For each of the following statements, determine if it is true or false.

(a) In the diagram of a DFA, there is exactly one transition for each pair of state and letter.

(b) For each regular language S, there exists an NFA with a single accepting state that recognises S.

(c) The language Σ^* is regular.

(d) No DFA can accept the empty word.

(e) If an NFA with n states accepts any word, then it accepts a word of length $n-1$ or less.

Solution

(a) True

(b) True

(c) True

(d) False

(e) True

* 3. For each of the following statements, determine if it is true or false.

(a) The following While program terminates when the value of `n` is initially the last digit of your student number.

```
while (5 <= n) {
  n := n + 1
}
```

(b) The following While program terminates when the value of `n` is initially the last digit of your student number and the value of `i` is initially 5.

```
while (i <= n) {
  i := i - 1
}
```

(c) The following While program terminates when the value of `n` is initially your student number and the value of `i` is initially negative.

```
while (i <= n) {
  i := i - 1
}
```

(d) `x := true` is a valid While program.

(e) Integer arithmetic in While can overflow.

Solution

(a) True for `n` initially less than 5, False otherwise.

(b) True for `n` initially less than 5, False otherwise.

(c) False (trick! it's false regardless of the student number)

(d) False

(e) False

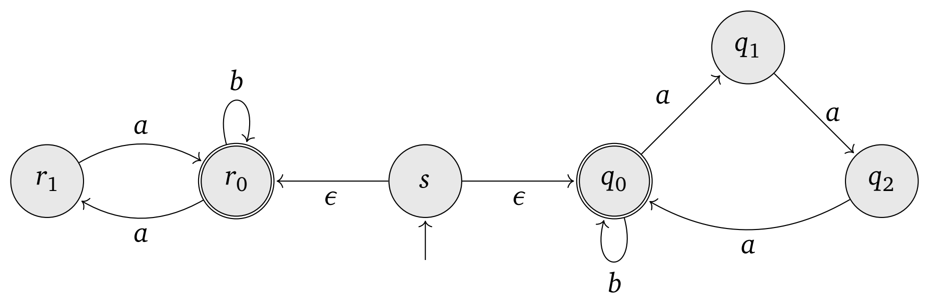


Figure 1: An Automaton.

* 4. For each of the following regular expressions over $\{a, b, c\}$, give (I) one word that is in the language denoted and (II) one word that is not.

Both words should be over the alphabet $\{a, b, c\}$. Label the two words with (I) and (II) so that it is clear which is claimed to be in and which is claimed to be not in.

(a) $(a+b)(a+b)(a+b)$

(b) $(a^*ba^*bc)^*$

(c) $(aa^*+bb^*)cc^*$

(d) $(\epsilon+b^*)^*$

Solution

Lots of answers are possible. One mark for each word correctly given.

* 5. Consider the automaton in Figure 1.

(a) Is this the diagram of a DFA? Justify your answer.

(b) Give two words that are accepted by the automaton.

(c) Give two words that are not accepted by the automaton.

(d) Describe in English the set of words accepted by the automaton (you may use any mathematical notation you find convenient).

Solution

(a) No, because there are epsilon transitions.

(b) E.g. ϵ, b

(c) E.g. a, $aaaaa$

(d) All words in which either:

- every substring a^k of maximal size has k a multiple of 2
- or every substring a^k of maximal size has k a multiple of 3.

Here we say that a substring w of the form a^k is "maximal" just if one cannot find a substring of the same form that contains w as a substring.

* 6. Show that the function $f : \mathbb{N} \rightharpoonup \mathbb{N}$ defined by

$$f(x) \begin{cases} \simeq x + 1 & \text{if } x^2 - 1 \text{ is at least 2022} \\ \uparrow & \text{otherwise} \end{cases}$$

is computable.

Solution

The function is computed by the following code with respect to x.

```
y := x * x -  1;
if (! (y < 2022)) { x := x+1; y := 0 }
else { while (true) do skip }
```

Award 1 mark for correctly stating the input/output variable; 2 marks for a mostly correct program; 1 mark for the infinite loop when the output is undefined; and 1 mark for setting all auxiliary variables to zero at the end.

* 7. State whether each of the following statements is true or false.

(a) Every injection has an inverse.

(b) The set $\mathbb{N}$ is decidable.

(c) Some WHILE programs compute total functions.

(d) If a function has an inverse, it must be a surjection.

(e) The set of all WHILE programs is countable.

Solution

(a) False.

(b) True.

(c) True.

(d) True.

(e) True.

* 8. Write a program that demonstrates that the function

$$f : \mathbb{N} \rightharpoonup \mathbb{N}$$

$$f(n) \begin{cases} \simeq 2^n & \text{if } n \text{ is divisible by 3} \\ \uparrow & \text{otherwise} \end{cases}$$

is computable.

Solution

The following program computes f with respect to n.

```
q := 0; r := n;
while (3 <= r) do { q := q + 1; r := r - 3 };
if (r = 0) then {
  j := 0; y := 1;  // Loop invariant: y = 2^j
  while (! (j = n)) { y := 2 * y; j := j + 1 }
  n := y
}
else {
  while (true) do skip
}
q := 0; r := 0; j := 0; y := 0
```

Award 1 mark for correctly stating the input/output variable; 2 marks for a mostly correct program; 1 mark for the infinite loop when the output is undefined; and 1 mark for setting all auxiliary variables to zero at the end.

** 9. Construct a bijection $\mathbb{Z} \times \mathbb{Z} \xrightarrow{\cong} \mathbb{N}$. Prove that it is a bijection by constructing its inverse, and show that it is indeed an inverse.

Solution

We construct a bijection $g : \mathbb{Z} \times \mathbb{Z} \xrightarrow{\cong} \mathbb{N}$ as

$$\mathbb{Z} \times \mathbb{Z} \xrightarrow{\beta \times \beta} \mathbb{N} \times \mathbb{N} \xrightarrow{\phi} \mathbb{N}$$

The inverse $g^{-1} : \mathbb{N} \xrightarrow{\cong} \mathbb{Z} \times \mathbb{Z}$ is given by

$$\mathbb{N} \xrightarrow{\phi^{-1}} \mathbb{N} \times \mathbb{N} \xrightarrow{\beta^{-1} \times \beta^{-1}} \mathbb{Z} \times \mathbb{Z}$$

For any $(a, b) \in \mathbb{Z} \times \mathbb{Z}$

$$\begin{aligned}
(\beta^{-1} \times \beta^{-1})(\phi^{-1}(\phi((\beta \times \beta)(a, b)))) &= (\beta^{-1} \times \beta^{-1})(\phi^{-1}(\phi(\beta(a), \beta(b)))) \\
&= (\beta^{-1} \times \beta^{-1})(\beta(a), \beta(b)) \\
&= (\beta^{-1}(\beta(a)), \beta^{-1}(\beta(b))) \\
&= (a, b)
\end{aligned}$$

For any $n \in \mathbb{N}$, write $(n_1, n_2) = \phi^{-1}(n)$ and calculate

$$\begin{aligned}
\phi((\beta \times \beta)((\beta^{-1} \times \beta^{-1})(\phi^{-1}(n)))) &= \phi((\beta \times \beta)((\beta^{-1} \times \beta^{-1})(\phi^{-1}(n)))) \\
&= \phi((\beta \times \beta)((\beta^{-1} \times \beta^{-1})(n_1, n_2))) \\
&= \phi((\beta \times \beta)(\beta^{-1}(n_1), \beta^{-1}(n_2))) \\
&= \phi((\beta \times \beta)(\beta^{-1}(n_1), \beta^{-1}(n_2))) \\
&= \phi(\beta(\beta^{-1}(n_1)), \beta(\beta^{-1}(n_2))) \\
&= \phi(n_1, n_2) \\
&= n
\end{aligned}$$

** 10. For each of the following regular expressions, draw the diagram of an NFA that recognises the language denoted by the expression.

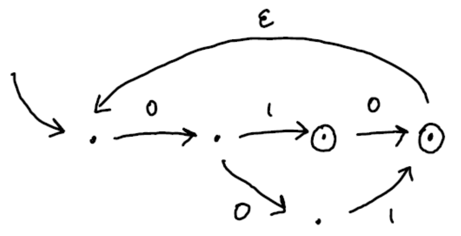
(a) $(01+001+010)^*$

(b) $0(10^*1+10)^*10$

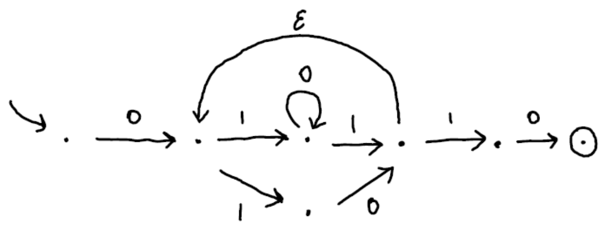
(c) $(((00)^*(11))+01)^*$

Solution

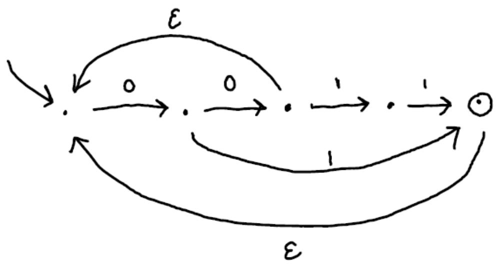
i.



ii.



iii.



*** 11. Suppose A is a regular language and $A \subseteq \{0,1\}^*$. Show that $A' = \{uv \mid u1v \in A\}$ is therefore also a regular language.

Solution

Let M be a DFA recognising A. Since M is a DFA, we can think of its transition relation as a function of the set of pairs of states and letters, call it δ. This is because there is exactly successor state for each pair (q,a) of state and a letter, and $\delta(q,a)$ is that unique successor state. Then we construct an NFA $(Q', \{0,1\}^*, \Delta', q_0', F')$ that recognises A', with the following definitions. Let $J = \{I, II\}$

- $Q' = Q \times J$
- $\Delta' = \{((q,i), b, (\delta(q,b), i)) \mid q \in Q,\ b \in \{0,1\},\ i \in J\} \cup \{((q,I), \epsilon, (\delta(q,1), II))\}$
- $q_0' = (q_0, I)$

- $F' = \{(q, II) \mid q \in F\}$

** 12. Write a While program, that, given 3 inputs given in variables a, b, and c, sorts them and returns the sorted values through the same variables. Your program can freely use other variables without needing to zero them out. (For example, running your program in memory $[\mathtt{a} = 42; \mathtt{b} = 12; \mathtt{c} = 1664]$ would yield a final memory that extends the mapping $[\mathtt{a} = 12; \mathtt{b} = 42; \mathtt{c} = 1664]$.)

Solution

A valid program is shown below (it's an inlined upside-down bubble sort, for some reason). Other programs are possible. Ignore minor syntax issues, in particular due to use of a more complete set of operators than technically available in the core While language.

```
if (c <= b)
then { t := b; b := c; c := t }
if (b <= a)
then { t := a; a := b; b := t }
if (c <= b)
then { t := b; b := c; c := t }
```

*** 13. In this exercise, consider the following While program. This program always terminates (you can take this as given).

```
s := 0
while (0 <= n) {
  s := s + n
  n := n - 1
}
```

(a) Write down the final semantic configuration (in the While semantics) of the execution of the program when the state is initially such that the value of variable n is the final digit of your student number (and all other variables are initially set to 0, as usual.)

(b) Express the final value of variable s as an expression of the initial value of variable n.

(c) Identify a loop invariant that formally establishes that your previous answer is correct. You do not need to give a formal proof, but writing down your thoughts and reasoning may demonstrate understanding better than a partially correct answer.

Solution

(a) The final configuration is $\left\langle \epsilon, \left[\begin{array}{c} \mathtt{n} \mapsto 0 \\ \mathtt{s} \mapsto \sum_{i=0}^{n} i \end{array} \right] \right\rangle$.

(b) The final value of s is $\frac{n \cdot (n+1)}{2}$, if n is the initial value of n.

(c) Observation: At the start of each loop iteration, we have $s + \sum_{i=0}^{n} = \frac{n \cdot (n+1)}{2}$.

** 14. Show that the predicate

$$U = \{\ulcorner S_1 \urcorner \mid \text{for all } k \leq 100 \text{ it is true that } [\![S_1]\!]_x(k) + 1 = [\![S_1]\!]_x(k+1)\}$$

is semi-decidable.

Solution

For each $k = 0, \ldots, 100$ simulate S_1 on input k, and then S_2 on input $k+1$. If these simulations terminate, check whether the output of the first plus one is equal to the output of the second. If not, return false and halt. Otherwise, after all the simulations are over return 1. Because this is a semi-decision procedure, the simulations need not terminate.

*** 15. Use the pumping lemma to show that the language $\{a^n b^m \mid n = 2 * m\}$ is not regular.

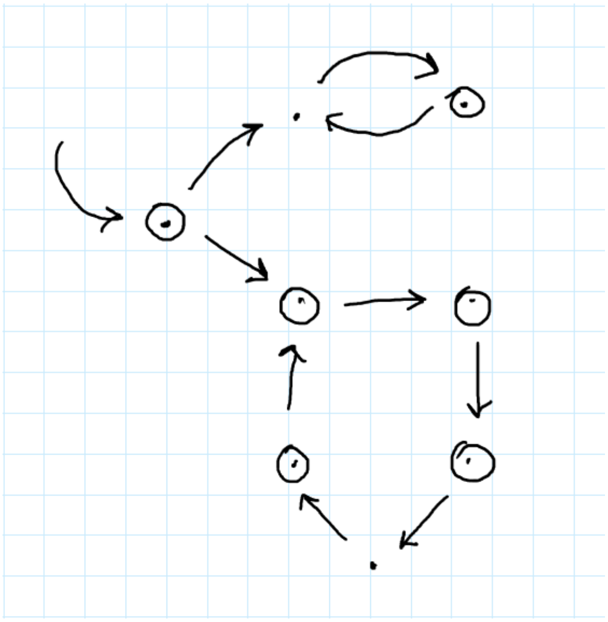
Solution

Let this language be A. Suppose A is regular. Then by the pumping lemma, there is some $p > 0$ such that, for all $s \in A$, s can be divided as uvw with (1) $|uv| \leq p$, (2) $|v| > 0$ and (3) $\forall i. uv^i w \in A$. So take $s = a^{2p} b^p$ and consider any division satisfying 1–3. It must be, by (1), that $uv = a^k$ with $k \leq p$ and hence $v = a^\ell$ for (2) $0 < \ell \leq k \leq p$. Then by (3) with $i = 2$ we have $a^{2p+\ell} b^p \in A$, but $2p + \ell \neq 2p$ so $a^{2p+\ell} b^p \notin A$: contradiction.

**** 16. Construct an NFA N with alphabet $\{1\}$ (consisting of a single letter: 1) and that satisfies the following two properties:

- N rejects at least one string.
- All strings of length less than or equal to the number of states of N are accepted by N.

Solution



** 17. Write a While program S_0 over a single variable r such that:

- The trace starting in configuration $\langle S, \sigma_0 \rangle$ with $\sigma_0 = \emptyset$ is infinite:

$$\langle S_0, \emptyset \rangle \to \langle S_1, \sigma_1 \rangle \to \langle S_2, \sigma_2 \rangle \to \langle S_3, \sigma_3 \rangle$$

- and every finite prefix of valuations of r:

$$\sigma_0(r), \sigma_1(r), \sigma_2(r), \ldots, \sigma_n(r)$$

(for any choice of n) is a word in the regular language $(0^+1^+2^+3^+)^*(0^+1^+2^+ + 0^+1^+ + 0^+ + \epsilon)$.

Solution

A valid program is shown below. Other programs are possible. Ignore minor syntax issues, in particular due to use of a more complete set of operators than technically available in the core While language.

```
while true {
  if   (r = 3)
  then r := 0
  else r := r + 1
}
```

** 18. For each of the following languages over $\{a, b, c\}$, construct an NFA that recognises it. Your automata should have at most 4 states.

(a) $\{w \mid w$ contains substring "baa"$\}$

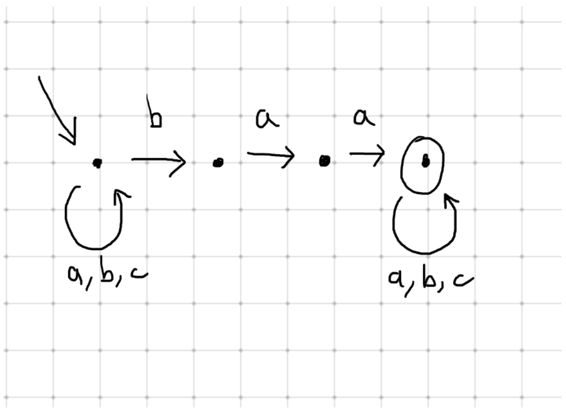
(b) $\{w \mid w$ ends with two occurrences of 'a'$\}$

(c) $\{w \mid w$ does *not* contain substring "abc"$\}$

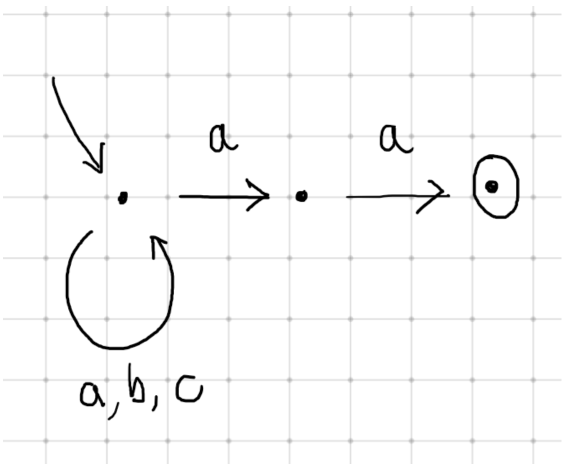
Solution

3 marks for i. and ii. and 4 marks for iii.

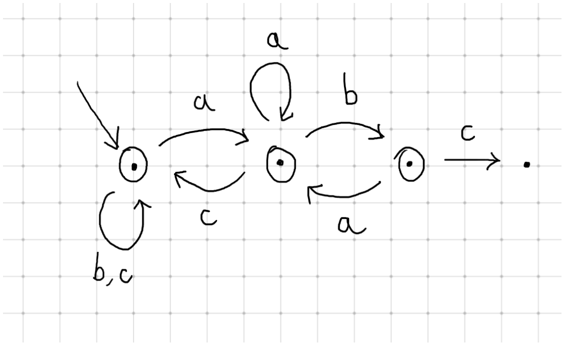
(a)



(b)



(c)



** 19. Show that if the predicates $A \subseteq \mathbb{N}$ and $B \subseteq \mathbb{N}$ are decidable then their *symmetric difference*, i.e. the set

$$A \oplus B = \{x \in \mathbb{N} \mid (x \in A \wedge x \notin B) \vee (x \notin A \wedge x \in B)\} = (A \cup B) - (A \cap B)$$

is also decidable.

Solution

Let S decide A wrt x, $andletT decide B wrt y$. Assume there are no variables in common in S and T. The program

```
x := z; S;
y := z; T;
if (x = 0 /\ y = 1) { z := 1 }
else if (x = 1 /\ y = 0) { z := 1 }
else { z := 0 }
x := 0; y := 0
```

decides $A \oplus B$ wrt z.

Award 1 mark for recognising there are programs that decide A and B; 1 mark for realising that you need to run both of them; and 1 mark for a correct solution that sets all auxiliary variables to zero.

*** 20. Show that the following language over $\{0,1\}$ is regular and justify your answer:

$$\{0^k u 0^k \mid k \geq 1 \wedge u \in \{0,1\}^*\}$$

Solution

3 marks for the witness. 2 marks for the justification.

Call this language A and let $\Sigma = \{0,1\}$. Then we have $A = 00^*\Sigma^*00^*$. Clearly, $A \subseteq 00^*\Sigma^*00^*$, now consider any word $w \in 00^*\Sigma^*00^*$. It follows that w has shape $0^i u 0^j$ for $i, j \geq 1$ and $u \in \Sigma^*$. Then let $k = \mathsf{min}(i, j)$ and it follows immediately that w has shape $0^k v 0^k$ for some $v \in \Sigma^*$. So, also $00^*\Sigma^*00^* \subseteq A$.

*** 21. Let Σ be some alphabet. We say that $a_1 a_2 \cdots a_m$, with each $a_i \in \Sigma$, is a *subword* of $w \in \Sigma^*$ just if there are words $v_0, v_1, \ldots v_m$ over Σ such that $w = v_0 a_1 v_1 a_2 v_2 \cdots a_m v_m$.

For a given language A over Σ, define

$$\mathsf{SUBWORDS(A)} = \{u \mid \exists w \in A.\ u \text{ is a subword of } w\}$$

Prove that the class of regular languages is closed under the SUBWORDS operation.

Solution

1 mark for attempting to constructing a witness for the regularity of SUBWORDS(A) (i.e. understanding this is what it means to be closed). 1 mark for evidence of comprehension of the defn, e.g. realising that subwords are obtained by erasing letters. 3 marks for a correct witness.

Suppose A is a regular language, so that there is some DFA $M = (Q, \Sigma, \Delta_M, q_0, F)$ that recognises A. Since M is a DFA, we view the transition relation δ_M rather as a function of $Q \times \Sigma$ since there is exactly one successor state $\delta(q, a)$ for each pair $(q, a) \in Q \times \Sigma$. Then construct an NFA $N = (Q, \Sigma, \Delta_N, q_0, F)$ with:

$$\Delta_N = \Delta_M \cup \{(q, \epsilon, q') \mid \exists a \in \Sigma.\ (q, a, q') \in \Delta_M\}$$

Since every subword of w can be obtained by replacing some letters of w by ϵ, it follows that this NFA recognises SUBWORDS(A).

*** 22. Show that the predicate

$$V = \{\phi(\gamma(S_1), \gamma(S_2)) \mid \text{there exists } k \in \mathbb{N} \text{ such that } [\![S_1]\!]_{\mathtt{x}}(k) + 1 = [\![S_2]\!]_{\mathtt{x}}(k+1)\}$$

is undecidable. (The use of $=$ above means that both sides of the equation must be defined, and equal.)

Solution

Construct the code transformation $F : \mathbf{Stmt} \times \mathbb{N} \to \mathbf{Stmt} \times \mathbf{Stmt}$ given by

$$F(\mathtt{D}, n) = (\mathtt{x\ :=\ n;\ D;\ x\ :=\ 0}, \mathtt{x\ :=\ 1})$$

It is easy to argue that (the reflection of) this code transformation is computable. Let the first component of that pair be $S_{\mathtt{D,\ n}}$. We have

$$\langle \ulcorner S_{\mathtt{D},n} \urcorner, \ulcorner \mathtt{x\ :=\ 1} \urcorner \rangle \in V \iff \langle \ulcorner \mathtt{D} \urcorner, n \rangle \in \mathsf{HALT}$$

As the latter is not decidable, neither is the former.

Award 1 mark for recognising that a reduction is the most appropriate proof method; 3 marks for constructing the reduction, and arguing that it is computable; and 1 mark for correctly stating the reduction property in this particular instance.

*** 23. Show that the predicate

$$U = \{\langle \ulcorner S \urcorner, \ulcorner T \urcorner \rangle \mid [\![S]\!]_{\mathtt{x}}(0) \simeq [\![T]\!]_{\mathtt{x}}(0)\}$$

is undecidable.

Solution

By reduction from the Halting Problem.

Construct the code transformation $F : \mathbf{Stmt} \times \mathbb{N} \to \mathbf{Stmt} \times \mathbf{Stmt}$ given by

$$F(\mathtt{D}, n) = (\mathtt{x\ :=\ n;\ D;\ x\ :=\ 0}, \mathtt{x\ :=\ 0})$$

This code transformation is computable. Its reflection is computed as follows.

On input m,

1. Write $m = \langle \ulcorner \mathtt{D} \urcorner, n \rangle$.
2. Construct the program $S_{\mathtt{D},n}$, which is given by

```
x := n; D; x := 0
```

3. Return $\langle \ulcorner S_{\mathtt{D},n} \urcorner, \ulcorner \mathtt{x\ :=\ 0} \urcorner \rangle$.

Now we have

$$\langle \ulcorner S_{\mathtt{D},n} \urcorner, \ulcorner \mathtt{x\ :=\ 0} \urcorner \rangle \in U \iff \langle \ulcorner \mathtt{D} \urcorner, n \rangle \in \mathsf{HALT}$$

As the latter is not decidable, neither is the former.

Award 1 mark for recognising that a reduction is the most appropriate proof method; 3 marks for constructing the reduction, and arguing that it is computable; and 1 mark for correctly stating the reduction property in this particular instance.

**** 24. One of the following two languages over the alphabet $\{0, 1, 2\}$ is regular and the other is not. Which is which? Give a full justification for your answer.

$$\{uv \mid \#_0(u) = \#_1(v)\}$$
$$\{u2v \mid \#_0(u) = \#_1(v)\}$$

Here, for any word w, $\#_0(w)$ is the number of occurrences of 0 in w and $\#_1(w)$ is the number of occurrences of 1 in w. For example, for $w = 0010101$, $\#_0(w) = 4$ and $\#_1(w) = 3$.

Solution

1 mark for sorting out which is which. 1 mark for stating that the former language is all words over the alphabet. 5 marks for a proof of that fact - this goes beyond what they have been taught on this unit, but not beyond what they know. 5 marks for the proof that the latter language is not regular.

The former language, call it A, is regular. In fact it is exactly $\{0, 1, 2\}^*$. We prove $\{0, 1, 2\}^* \subseteq A$ by induction on the length n of words w.

- When $n = 0$ the word w is empty and we can set $u = v = \epsilon$. It follows that $\#_0(u) = \#_1(v) = 0$.
- When $n = k + 1$, the word has shape $w'a$ for $a \in \{0, 1, 2\}$. It follows from the induction hypothesis that w', being of length k, may be split as $u'v'$ with $\#_0(u') = \#_1(v')$. We consider cases on a:

- When $a = 0$ or $a = 2$, set $u := u'$ and $v := v'a$. Then:

$$\#_0(u) = \#_0(u') = \#_1(v') = \#_1(v'a) = \#_1(v)$$

- When $a = 1$, consider cases on v'. We analyse according to the first non-2 character.
  - When $v' = 2^j$ for some $j \geq 0$, set $u := u'2^j a$ and $v := \epsilon$. Then $\#_0(u) = \#_0(u') = \#_1(v') = \#_1(v)$.
  - When $v' = 2^j 0 v''$ for some $j \geq 0$, set $u := u'2^j 0$ and $v := v''a$. Then $\#_0(u) = \#_0(u') + 1 = \#_1(v') + 1 = \#_1(v)$.
  - When $v' = 2^j 1 v''$ for some $j \geq 0$, set $u := u'2^j 1$ and $v := v''a$. Then $\#_0(u) = \#_0(u') = \#_1(v') = \#_1(v'') + 1 = \#_1(v)$.

The latter language, call it B, is not regular. We argue as follows. Suppose B is regular. Then by the Pumping Lemma, there is a number $p > 0$ such that, for all $s \in B$, if $|s| \geq p$ then s may be split as uvw satisfying (i) $|v| > 0$, (ii) $|uv| \leq p$ and (iii) $\forall i \in \mathbb{N}. uv^i w \in B$. By definition, $0^p 2 1^p \in B$ and has length greater than p. Hence, it may be split as uvw satisfying (i)–(iii). By (i) and (ii) we have $v = 0^k$ for some $k > 0$. By (iii) we have $uv^2w \in B$. However, $uv^2w = 0^{p+k} 2 1^p$. Hence, by definition, $uv^2w \notin B$.