Computing Theory
COSC 1107/1105
Assignment 2: Computability

Assessment Type	**Individual assignment.** Submit online via Canvas → Assignments → Assignment 1. Marks awarded for meeting requirements as closely as possible. Clarifications/updates may be made via announcements/relevant discussion forums.
Due Date	Week 12, Sunday 16th October 2022, 11:59pm
Marks	110 worth 25% of your final assessment

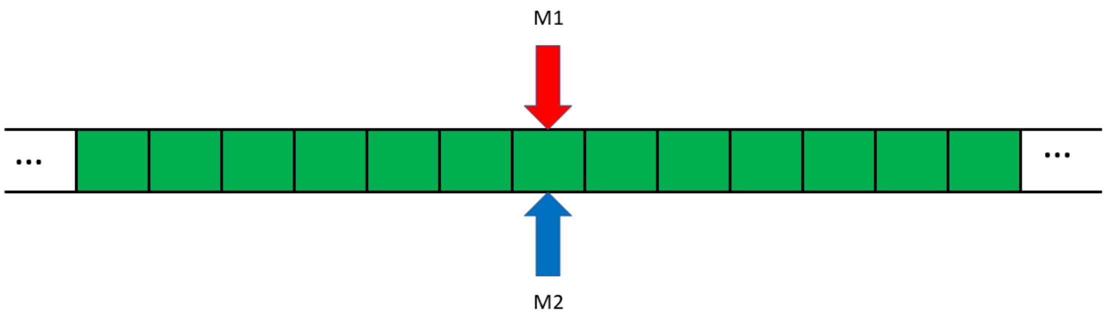
1 Overview

This assignment requires you to demonstrate your knowledge of the key concepts of Turing machines, universality, computability and the Chomsky Hierarchy. You are also required to report on some further experiments with the Platypus game.

2 Assessment details

1. **Computability** (15 marks)

   The generalised Platypus green game (GPGG) is defined as follows. Let M_1 and M_2 be Turing machines, which share the same tape. The tape is initially all green. The initial configuration of the two machines is as shown below.



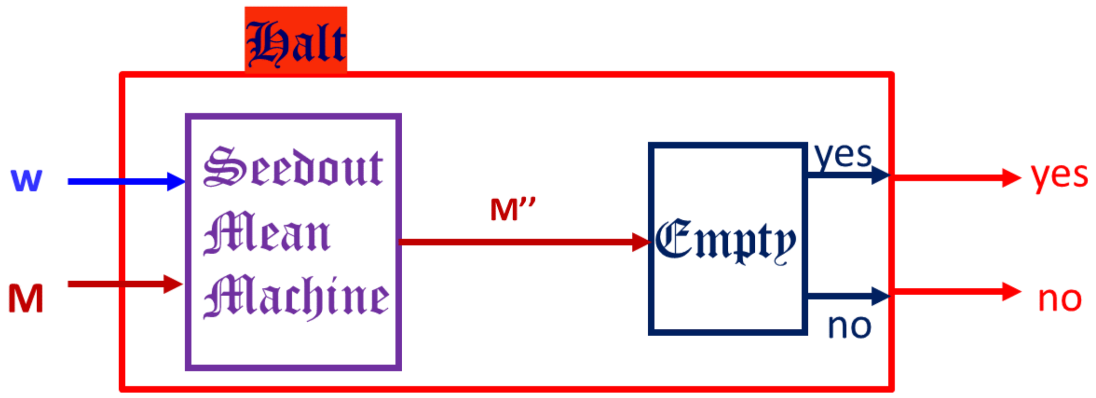
   As in the Platypus game, each machine takes turns to move (but there is no scoring involved).

(a) Show that the halting problem for GPGG is undecidable. You may use any reduction that you like. (6 marks)

(b) Suppose the GPGG is played on a Turing machine with a finite tape (making the halting problem decidable), and that this problem has been shown to be NP-complete. Given your above reduction from some problem A to the GPGG, can this reduction be used to conclude that A is NP-complete? Why or why not? Explain your answer. (2 marks)

(c) Seedout, a rather adventurous and reckless hobbit, has proposed the reduction below as a way of showing the Empty Language problem is undecidable by reducing the Halting Problem to it. Unfortunately there are two errors in this reduction. Identify these errors, explain why these are incorrect and how they may be corrected. (4 marks)

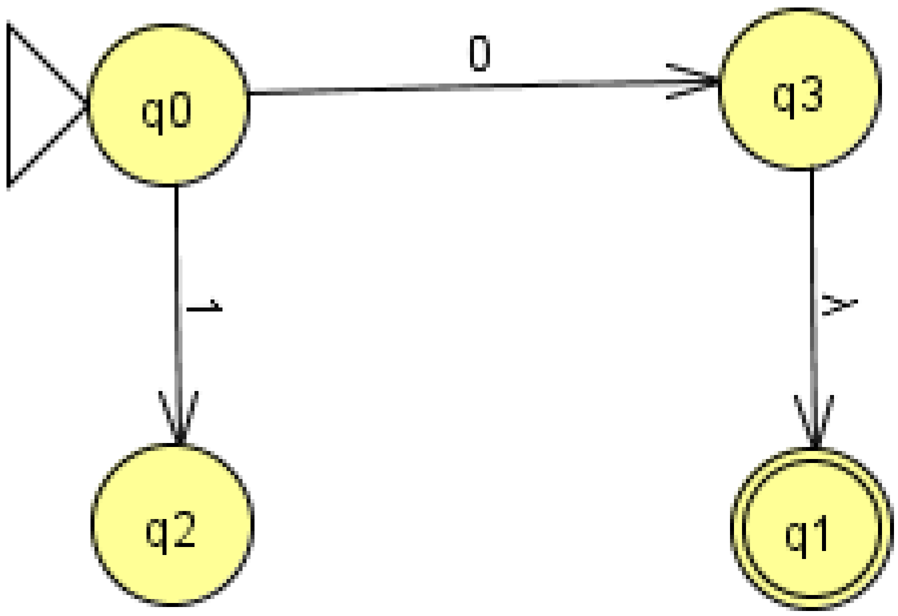
Seedout's explanation is: "Assuming the Empty Language problem is decidable (and hence the existence of a Turing machine *Empty* which decides the problem), we can construct a solution to the Halting Problem as follows. First, given the Turing machine M and input w, we use these to construct another Turning machine M'' which deletes any input on the tape, writes w on the tape and then acts as M would. So the language accepted by M'' is empty iff M halts on w. See the diagram below for more details."



(d) Undecidability results are often given in terms of Turing machines. Explain how these results are relevant to modern programming languages, using three examples of undecidable problems that occur in such languages. (3 marks)

2. **Nondeterminism** (8 marks)

Consider the incomplete NFA M_0 below, whose alphabet is $\{0, 1\}$.



Use M_0 to create three more NFAs M_1, M_2 and M_3 according to the constraints below. Explain in one or two English sentences how you constructed each NFA.

- Each of M_1, M_2 and M_3 must contain at least 10 transitions (potentially but not necessarily including λ-transitions) and must be an NFA but not a DFA. Specifically there must be at least one combination of state and input (either 0 or 1) for which there are at least two possible states. Put another way, removing all λ-transitions must not result in a DFA.
- Use JFLAP to transform M_1, M_2 and M_3 into equivalent DFAs.
- The sizes of the DFA resulting from the determinising algorithm must be as below. Note that the JFLAP implementation of this often omits an "error" state, i.e. it may be necessary to add an extra state to the result from JFLAP in order to account for this. The size constraints below assume a fully deterministic DFA; one way to check for this is that if the DFA has k states, there must be exactly $2k$ transitions (one for each of 0 and 1 in each state).

(a) The size of the DFA corresponding to M_1 is 2 or 3. (3 marks)

(b) The size of the DFA corresponding to M_2 is 5. (2 marks)

(c) The size of the DFA corresponding to M_3 is at least 9 (this may be harder than you think!) (3 marks)

3. **Pumping Lemma** (20 marks)

(a) There are three errors in the statement of the Pumping Lemma for regular languages below. Find all three, state how to correct them, and explain your answers. (4 marks)

Let L be a regular language. Then $\exists n \geq 1$ such that for any $w \in L$ such that $|w| > n$, $\exists x, y, z$ such that $w = xyz$ where

i. $|xy| \leq n$
ii. $x \neq \lambda$
iii. $xy^iz \in L$ for all $i \geq 1$

(b) Galadriel, on her Elftiki tour of Middle-Earth in the Second Age, comes across some scraps of parchment in the Hall of Records of Numenor. These scraps, numbering 10 in all, are partly burned and not always legible, but contain some ancient runes that not even Galadriel can decipher. But she can make some educated guesses, and first must decide on the relevant order of the fragments, before choosing from potential translations for each of them, so that the overall message makes sense.

The fragments are below, with the parts where there are alternative translations labelled with capital letters, like this:

$\boxed{\text{(Translated text) A (Translated text)}}$

A1: Alternative 1
A2: Alternative 2
A3: Alternative 3

1: $\boxed{\text{As the Pumping Lemma requires } xy^iz \in L\text{, this is a A}}$

A1: paradox A2: problem A3: tautology A4: contradiction

2: $\boxed{\text{so we have } w \in L,\ |w| \geq n \text{ and B, and so we have C for some } 1 \leq j \leq n.}$

B1: $|xy| < n$ B2: $|xy| \leq n$ B3: $|xy| > n$ B4: $|xy| \geq n$

C1: $y = 0^j$ C2: $y = 1^j$ C3: $y = 0^j1^j$ C4: $y = (012)^j$

3: $\boxed{\text{D and } xy^iz \in L \text{ for all } i \geq 0}$

D1: $|y| = 0$ D2: $|y| \neq 0$ D3: $|y| > 1$ D4: $|y| = 1$

4: $\boxed{\text{Then } \exists n \geq 1 \text{ such that E and F,}}$

E1: for all $w \in L$ and $|w| \geq n$ E2: for all $w \in L$ and $|w| \leq n$ E3: for some $w \in L$ and $|w| \geq n$ E4: for some $w \in L$ and $|w| \leq n$

F1: $\exists x, y, z$ such that $w = xyz, |xy| \leq n$ F2: $\exists x, y, z$ such that $w = xyz, |xy| \geq n$ F3: $\forall x, y, z$ such that $w = xyz, |xy| \leq n$ F4: $\forall x, y, z$ such that $w = xyz, |xy| \geq n$

5: $\boxed{\text{and so we have shown that our assumption is false, i.e. that G.}}$

G1: context-free G2: not regular G3: empty G4: regular

6: $\boxed{\text{Beren's tale: A proof that the language } L = \text{H is not I}}$.

H1: $\{ww | w \in \{0,1,2\}^*\}$ H2: $\{w2w | w \in \{0,1,2\}^*\}$ H3: $\{w1w | w \in \{0,1,2\}^*\}$ H4: $\{w1w2w | w \in \{0,1,2\}^*\}$

I1: context-free I2: regular I3: empty I4: recursively enumerable

7: $\boxed{\text{and so } xy^iz \text{ is J}}$

J1: $1^{n+j}21^n$ J2: $0^{n+j}0^n$ J3: $0^{n-j}10^n20^n$ J4: $1^{n+j}11^n20^n$

8: $\boxed{\text{Let } w \text{ be K.}}$

K1: $0^n10^n20^n$. K2: 1^n21^n. K3: $1^n11^n20^n$. K4: 0^n10^n.

9: $\boxed{\text{Assume } L \text{ is L.}}$

L1: not regular L2: regular L3: not context-free L4: context-free

10: $\boxed{\text{Now consider } M}$

M1: $i = 0$ M2: $i = 1$ M3: $i = 2$ M4: $i = 3$

Re-assemble the proof for Galadriel. (6 marks)

You do not need to write the proof out in full. You can submit your final version on Canvas.

(c) The Pumping Lemma for regular languages states a necessary property for such languages. Is it possible that the same property is true for context-free and context-sensitive languages? Explain your answer. (3 marks)

(d) Let L the language of a DFA with n states. Explain why either $L = \emptyset$ or there is a string $w \in L$ such that $|w| < n$. (3 marks).

(e) The Pumping Lemma for context-free languages is below.

Let L be a context-free language. Then there is an $n \geq 1$ such that for any string $w \in L$ with $|w| \geq n$ there exists strings x, y, z, u, v such that $w = xyzuv$ and

i. $|yzu| \leq n$

ii. $|y| + |u| > 0$

iii. $xy^izu^iv \in L$ for all $i \geq 0$

Use this to show that the language $L = \{a^{i^3} \mid i \geq 2\}$ is not context-free by filling in the gaps below. (4 marks)

Proof: Assume ____________. So the Pumping Lemma applies, and so for any string $w \in L$ with $|w| \geq n$ there exist strings x, y, z, u, v such that $w = xyzuv$ and

i. $|yzu| \leq n$

ii. $|y| + |u| > 0$

iii. $xy^izu^iv \in L$ for all $i \geq 0$

Let ______. So $w \in L$ and ______, and $w = xyzuv$, $|yzu| \leq n$, $|y| + |u| > 0$ and $xy^izu^iv \in L$ for all $i \geq 0$. Now as ______, this means $|yzu| = |y| + |z| + |u| \leq n$ and so $|y| + |u| \leq n$.

Let $i = 0$ and consider $|xy^0zu^0v| = |xzv| =$ _____ $-|y| - |u| = n^3 - (|y| + |u|) \geq n^3 - n >$. Now as $n \geq 2$, $4n < 3n^2$ and so $n < 3n^2 - 3n$. This means $n^3 - n > n^3 - 3n^2 + 3n >$ _______ $= (n-1)^3$.

So $n^3 =$ _____ $> |xy^0zu^0v| > (n-1)^3$, and so _______ This is a contradiction, and so L is not context-free.

4. **Complexity** (9 marks)

(a) "And long there he lay, an image of the splendour of the Kings of Men in glory undimmed before the breaking of the world." – *The Lord of the Rings*, by J.R.R. Tolkien.

It may be a very long time before "the breaking of the world". How would you interpret this time? This may be thought of as estimating how much time is left before "the end of the world", however you may interpret that. Give your estimation of this length of time in years. Justify your answer. (3 marks)

(b) The Towers of Hanoi is a classic problem in computing. To move a tower of height n from one pole to another (observing the rules of course) takes a minimum of $2^n - 1$ steps. It is said the world will end before this problem can be solved when $n = 64$.

Calculate how long it will take to solve this problem for the cases where $32 \leq n \leq 64$, assuming 0.1 seconds per move. Does this support the claim about the end of the world being sooner?

Use a spreadsheet for this calculation, and report the cases for $n = 32, 40, 48, 56$ and 64. (3 marks)

(c) It is estimated that the Big Bang was 13.7 billion years ago. This is

$$13,700,000,000 \times 365.25 \times 24 \times 60 \times 60 = 432,339,120,000,000,000$$

seconds ago.

Calculate the value of n for which the Towers of Hanoi problem can be solved in this time at the rate of 10 seconds per move.

Compare this value with the number of machines that could play a complete Platypus tournament in this time at the rate of 0.1 seconds per match.

You will also find a spreadsheet very useful for this. (3 marks)

5. **Intractable problems** (12 marks)

Intractable problems are decidable problems, but for which the best known solution is exponential (or worse). Describe two intractable problems and their practical application. You should write one introductory paragraph on intractable problems, and two further paragraphs, one on each problem, and a reason that you selected each one. Some suggestions will be given in class and on Canvas.

Please note that I am not at all interested in what you can find on Google or Wikipedia or anything like that. What I really want to see is some evidence of you thinking about intractable problems and what effect these have, such as the relationship between the vertex cover problem and sensor networks.

Perhaps an interesting way to address this is to consider what differences it would make if your intractable problem could be solved efficiently. For example, if we could solve say the Travelling Salesperson problem in $O(n^2)$ time, what would be possible that is impossible now?

Another possibility is to consider how intractability can be a useful thing, such as keeping information secure in cryptosystems or in related applications such as blockchain.

Whatever problems you choose, please avoid the temptation to 'cut, copy and edit'; as soon as you do that, you have done something wrong. I would far prefer to read your own words and your own perspective on these problems.

6. **Universality** (30 marks)

In a nutshell, you are expected to revise and extend your work on this topic in Assignment 1.

In Assignment 1, you investigated one of the following three topics, or came up with your own related topic or creative story.

- Two-dimensional Turing machines
- Small universal Turing machines
- Notable universal Turing machines

For this assignment, **you are expected to either continue your investigation from Assignment 1 on the same topic in more depth, or to make a different choice.** In other words, you can either continue with your choice from Assignment 1, or make a different one now. Whatever your decision, **you are expected to write about 1800-2000 words (9 or 10 paragraphs) overall.** This should include a revised version of your Assignment 1 submission, so that if you continue with the same choice as in Assignment 1, this is will be an extended form of that work. If you make a different choice, that is fine, but you should include your (potentially revised) Assignment 1 submission as part of this submission. So you have two different choices for the two assignments, you are expected to write about the same length on each; if you have the same choice for each, you should write about 1800-2000 words overall. Either way, **the submission for Assignment 2 will involve around 1000 words over and above what you submitted for Assignment 1.**

As in Assignment 1, you may also propose an alternative topic, or write a **creative story** involving a Turing machine of some kind. You can do this even if you did not choose either of these in Assignment 1. **However, for any alternative topic or creative story, please seek approval from the lecturer before commencing work on it.** This is to make sure that what you are doing is appropriate for this assignment — I would hate to see an outcome where you do a lot of work, only for it not to count because it does not address the intended content.

One alternative topic of particular interest is **quantum computing;** if anyone is interested in pursuing this, you are strongly encouraged to do so (but as above, please do consult me first). You may also be interested in Amazon Braket. Another possibility is **zero-knowledge proofs**, but again please consult me before doing this.

A further point is that ***you can present your information in other ways that a formal report if you wish***. **You are also encouraged to consider the use of the Adobe Creative Cloud suite,** to which all RMIT students have access. You can find the details about the Adobe Creative Cloud at this link.

Some suggestions are below. Please keep in mind that you still need to discuss the technical content; the point is to find a way that assists you with this, rather than being a blockage for you.

- Pick a side in the debate about the 2007 universal TM competition
- Langton's Ant vs Paterson's worm
- 'Cellular automata are better than Turing machines'
- Write a children's story, movie scene, poem, . . .
- 2D TMs as a game, map, drawing a picture, annotating photos, . . .
- Implementation of some aspects (be careful of rabbit holes!)
- Experiment with Java implementation of 2D Universal TM

- Langton's ant with 'boundaries' (see an example here)

You will be marked according to the rubric below.

Points	**Description**	**Details**
18-24	Exemplary	You have explored your chosen topic well. Your report is clear and well-written, and has the appropriate length. This is interesting and informative.
12-17	Accomplished	You have explored your chosen topic reasonably well. Your report is generally good, but can be improved with some more attention to detail, particularly concerning the amount of technical detail.
6-11	Developing	Your report needs some further work, either to increase the level of content or to improve the choice of material and its presentation.
0-5	Beginning	Your report does not show evidence of sufficient investigation, and is lacking in detail.

7. **The Platypus game.** (16 marks)

We have previously talked about running as large a tournament as possible with the Platypus game. The way we will do this in this assignment is for each of your to run a tournament of 2,500 machines. From these, you will report your top 10 machines (see below for details). These 10 will then be part of a knock-out tournament to determine the overall winner.

Before answering the questions below, do the following.

- Get your allocation of 2,500 machines. These will be in this folder. Look for a file in the folder with your student number, i.e. if your student number is 7654321, look for the file 7654321.pl. Store this somewhere that suits you with a name like `machines.pl` for convenience.
- Open a new SWI-Prolog shell and consult both `platypus.pl` and `machines.pl` (or whatever you called it). Then run the command below. This will classify your list of machines into three categories, and will generate one file each category named `none.pl`, `reachable.pl` and `unreachable.pl`, as per the description below. This will be in the directory specified by the `results_directory` predicate in `platypus.pl`; you can change this to something more convenient for you if you like.

  ?- classify_machines.

None	The machine has no platypus state in the fourth row of columns 1-6
Reachable	It is possible reach the platypus state from the kangaroo state
Unreachable	It is not possible reach the platypus state from the kangaroo state

  The point of this analysis is to work out whether it is possible for the machine to terminate the game or not. An example of each class of machine is below. In the first case, the machine cannot terminate the game, as it is not possible for it to go from the start state (`kangaroo`) to the `platypus` state as there is no transition that will move the machine into the `platypus` state.

Y	G	Y	G	Y	G	Y
K	K	E	E	W	W	P
g	y	y	g	y	y	g
Emu	**Emu**	**Wombat**	**Kangaroo**	**Wombat**	**Emu**	**Kangaroo**
w	gg	gg	w	w	w	gg

The second and third cases occur when there is indeed such a `platypus` state. When it is possible to move from the `kangaroo` state to the `platypus` state (depending on the cells on the tape of course), then the machine is classified as reachable, as in the machine below. This is because it is possible to move from the `kangaroo` state to the `emu` state (column 1 or 2), from the `emu` state to the `wombat` state (column 3) and from the `wombat` state to the `platypus` state (column 6).

Y	G	Y	G	Y	G	Y
K	K	E	E	W	W	P
g	y	y	g	y	y	g
Emu	**Emu**	**Wombat**	**Kangaroo**	**Wombat**	**Platypus**	**Kangaroo**
w	gg	gg	w	w	w	gg

It is also possible that even if such a `platypus` state exists, it is not possible to move into that state from the `kangaroo` state. In this case, the machine is classified as unreachable, as in the machine below. In this machine we can get from the `kangaroo` state to the `wombat` state and vice-versa, but we can never get to the `emu` state or `platypus` state from either of these.

Y	G	Y	G	Y	G	Y
K	K	E	E	W	W	P
g	y	y	g	y	y	g
Wombat	**Wombat**	**Platypus**	**Kangaroo**	**Wombat**	**Kangaroo**	**Kangaroo**
w	gg	gg	w	w	w	gg

Intuitively, machines classified as reachable are in some sense "genuine" with the others being "imposters". Accordingly, having separate tournaments for each seems only fair.

- Run a tournament for each of the following classes of machines
  - All 2,500 machines
  - Only machines classified as reachable
  - Only machines classified none or unreachable (ie all those not classified as reachable).

  To do this, you will first have to prepare a file containing only the machines as above (the above `classify_machines` command will more or less do this for you). You will also need to use the command below, which will run the tournament with all of the options below.

?- my_tournament([competition]).

Variation	**Description**
Tree	5 points for whenever either tree is reached
Green	2 points rather than 1 for changing green to yellow
Animal	1 point every time a change of animal occurs
Tiebreaker	A random starting configuration is chosen with game length is 200

(a) For each of your tournaments, report the overall time taken, the top 10 machines (by 'football' ranking), the overall number of wins and draws, and the number of winless machines. How many machines were classified as none, reachable and unreachable respectively? Report your results in a table as below. (2 marks)

Class	**Number**	**Percentage**
Reachable		
Unreachable		
None		

(b) Re-run your tournament of all machines, but this time you are to include 10 extra machines of your own choice. These should be different from any machines in your list already, but otherwise you are free to choose them however you like.

You can use `platypus.pl` from Canvas (link here) to assist with this if you wish. This will check whether your 10 added machines are 'legal', and whether or not they were already part of your allocation. To do this, consult `platypus.pl` and `machines.pl` as above, and a third file `extra.pl`. Then run the command

?- check_new.

This will then output whether your added machines are legal, and whether they are already part of your allocation or not (and if they are, which ones are already in their allocation).

You should report where each of these 10 machines finish in the 'football' ranking, and your reasons for choosing each of them. (2 marks)

(c) Were you surprised how your chosen 10 machines performed? What can you conclude from this about high performing Platypus machines? If you had to choose one machine to represent you in a tournament, what machine would it be? Briefly explain your decision. (4 marks)

(d) In the previous assignment, your calculated the largest Platypus tournament you can play on your machine in 4 hours, ie $4 \times 60 \times 60 = 14,400$ seconds. This is of course for the 'standard' 2-player game. 3-player and 4-player tournaments will of course take longer. Calculate the largest 3- and 4-player tournaments you can play on your machine in 4 hours. You may assume that a 3- or 4-player match takes the same time as a 2-player one. You may also find the following table useful (see the notes on 3- and 4-player tournaments for how these are derived). (2 marks)

Players	Matches required
2	$n(n+1)/2$
3	$n(n+1)(n+2)/6$
4	$n(n+1)(n+2)(n+3)/24$

(e) If the Platypus tournament were to be run again, what alterations would you recommend? Some ideas are below; you can add others as you see fit. You should have at least three suggestions. (6 marks)

- Once students are allocated their machines, they play a tournament amongst these to find the best 10. These 10 then take on the best 10 from other students. This could include the possibility of students choosing their 10 machines from their allocation or from the set of machines which are not allocated to any student.
- The initial tape and scoring processes are changed from tournament to tournament. These are announced in advance, and allow choices to be made for which machines will be used.
- Non-player machines can be added. These are not competitors, but may change the tape in ways that influence the game. Such machines would not be allowed to terminate the game (presumably by allowing them transitions for a green cell with a platypus).
- When a player has a platypus on a green cell, the game does not halt, but is "rebooted", i.e. the tape reverts to its initial state, the player which 'halted' the game gets a bonus or a penalty, and the machines involved are changed in some way. This change could be swapping rows 3, 4 and 5 of column 1 and 2 (kangaroo), and the same for columns 3 and 4 (emu) and 5 and 6 (wombat). There could be a maximum of say 5 reboots per game.
- (insert your idea here!)

3 Submission

You should submit the following.

- A PDF file containing your answers to the questions.
- All `.csv` files generated by running your tournaments.
- Submit your answer to Question 4b via Canvas.

No other file formats will be accepted.

4 Marking guidelines

Your assessment will be marked according to the criteria below.

- Completeness and accuracy of your answers to the first five questions.
- Quality of your report on your chosen topic for Question 6.

- Completion of your section of the Platypus game tournament.
- Quality of your comments on the Platypus game tournament.