

Exercise walk-through

11.3

Computability

University Computer Science

Computability: diagonalisation

- 1 **D** Assume halting decider H ; $D(M)$ runs $H(M,M)$ and loops exactly when H says halts - D on D contradicts.
- 2 **Rice** Every nontrivial semantic property is undecidable - does P print banana reduces halting to printing.
- 3 **Cousin** DFA emptiness IS decidable: finite reachability search - the infinite tape is exactly what breaks the argument.

Machine model

D flips H answers about itself — no H exists



table view: enumerate machines x inputs; D disagrees with the diagonal of every row

M3	1	1	0
M2	0	1	0
M1	1	0	1

diagonal

decidable cousins:

DFA emptiness: reachability search — finite, so decidable

Rice: every nontrivial semantic property undecidable

D flips H answers about itself - no H can exist.

Exercises 2.1-2.3

2.1 State the halting problem in one sentence.

[3]

2.2 What does $D(M)$ do in the proof?

[3]

2.3 What does Rice's theorem cover?

[3]

Exercises 2.1-2.3

2.1 State the halting problem in one sentence. [3]

2.1

- Does machine M halt when run on input w ? ✓.

2.2 What does $D(M)$ do in the proof? [3]

2.2

- Runs $H(M, M)$: if it answers halts, loop forever; otherwise halt ✓.

2.3 What does Rice's theorem cover? [3]

2.3

- Every nontrivial semantic property of programs (what they compute, not how they are written) ✓.

Exercise 3.1

3.1 Suppose $H(M, w)$ decides halting.

Write the construction of D from H precisely (both branches). Trace what happens when D runs on its own description. State the contradiction and what it proves about H . State the two mistakes in the claim “so no program ever stops” ✓. [8]

Exercise 3.1

3.1 Suppose $H(M, w)$ decides halting.

Write the construction of D from H precisely (both branches). Trace what happens when D runs on its own description. State the contradiction and what it proves about H . State the two mistakes in the claim “so no program ever stops” ✓. [8]

3.1

- $D(M)$: run $H(M, M)$; if the answer is “halts”, enter an infinite loop; if “loops”, halt immediately ✓.
- $D(D)$: $H(D, D)$ must answer something; either answer makes D do the opposite of what the answer says about itself ✓.
- Contradiction: no consistent H exists – halting is undecidable ✓.
- Mistakes: undecidable is about a general decider for ALL pairs, and specific instances often are answerable – the theorem is not about any program stopping ✓.

Exercise 3.2

3.2 Three questions, three verdicts:

(1) Does DFA D accept any string at all? (2) Does TM M accept any string at all? (3) Does program P print banana on input x ?

Give each verdict (decidable / undecidable) with a one-line reason. For the decidable one, give the algorithm. For (3), name the theorem and give the reduction's wrapper logic. [7]

Exercise 3.2

3.2 Three questions, three verdicts:

(1) Does DFA D accept any string at all? (2) Does TM M accept any string at all? (3) Does program P print banana on input x ?

Give each verdict (decidable / undecidable) with a one-line reason. For the decidable one, give the algorithm. For (3), name the theorem and give the reduction's wrapper logic. [7]

3.2

- (1) Decidable: finite graph – check whether any accepting state is reachable from the start ✓.
- (2) Undecidable: $L(M) \neq \emptyset$ is a nontrivial property of what M computes – Rice, or reduce halting by making M accept exactly when the halting computation finishes ✓.
- (3) Undecidable by Rice: wrap P so banana prints only after the wrapped computation halts – a banana-detector would decide halting ✓.