The Busy Beaver Problem PRIMES Circle

Turing Machines Docidabili

Busy Beave Problem The Busy Beaver Problem Esther Fu and Sarah Pan (Mentor: Alexandra Hoey)

PRIMES Circle

MIT

May 21, 2022

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三 のへぐ

Computers are great! The Busy Beaver Problem Why do we love computers?

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

Computers are great!



Turing Machines

Busy Beaver Problem

Why do we love computers?

Maybe they can help resolve long-standing mathematical questions!

<□ > < 同 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ < つ < ○</p>

A Few Problems

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidability

Busy Beaver Problem

The Riemann Hypothesis (1859)

The Riemann zeta function has zeros only at negative even integers and complex numbers with real part $\frac{1}{2}$.

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

A Few Problems

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidability

Busy Beaver Problem

The Riemann Hypothesis (1859)

The Riemann zeta function has zeros only at negative even integers and complex numbers with real part $\frac{1}{2}$.

Goldbach's conjecture (1742)

Every even integer greater than two can be expressed as the sum of two primes.

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

The Busy Beaver Problem PRIMES

Turing Machines Decidabilit<u></u>

Busy Beaver Problem

Goldbach's conjecture (1742)

Every even integer greater than two can be expressed as the sum of two primes.

▲ロ▶ ▲周▶ ▲ヨ▶ ▲ヨ▶ ヨ のなべ

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidability

Busy Beaver Problem

Goldbach's conjecture (1742)

Every even integer greater than two can be expressed as the sum of two primes.

Example

$$10 = 3 + 7$$

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidability

Busy Beaver Problem

Goldbach's conjecture (1742)

Every even integer greater than two can be expressed as the sum of two primes.

Example

10 = 3 + 7148 = 101 + 47

◆□▶ ◆□▶ ◆目▶ ◆目▶ ▲□ ◆ ��や

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidability

Busy Beaver Problem

Goldbach's conjecture (1742)

Every even integer greater than two can be expressed as the sum of two primes.

Example

10 = 3 + 7 148 = 101 + 474390 = 1091 + 3299

◆□▶ ◆□▶ ◆目▶ ◆目▶ ▲□ ◆ ��や

A possible solution to Goldbach's conjecture?

Image: Second
PRIMES Circle 1 000/023b; (0018b) 2 001(017a) (002b) 3 002(021a) (003b) 4 003(021a) (003b) 5 004(007b) (004a) 6 003(021a) (003b) Machines 7 006(008b) (0051a) 0 004(007b) (0051a) 004(007b) (0051a) Decidability 0 006(008b) (005a) 11 014(015b) (015b) 014(015b) (015b) Problem 12 014(015b) (015b) 13 012(041a) (015b) 13 14 013(015b) (012b) 13 15 014(015b) (012b) 13 16 015(015b) (012b) 13 17 015(015b) (012b) 14 18 017(015b) (012b) 14 19 015(015b) (012b) 14
Circle 2 001(011_a)(028b) Turing 3 002(011_a)(08bb) Turing 6 005 (0041_a)(085b) Machines 8 007 (0048b)(087b) Decidability 10 005 (0041_a)(087b) Busy Beaver 12 001 (018bb) (013b) Problem 13 012 (018b) (013b) 15 014 (018bb) (013b) 15 16 015 (018bb) (013b) 16 17 016 (018b) (013b) 16 18 017 (018b) (011b) 17 19 018 (012b) (012b) 19
Urrele 3 002(211a)(038b) 4 003(211a)(038b) 5 004(098b)(095La) 6 005(098b)(095La) Machines 7 006(098b)(095La) 0 007(098b)(095La) 0 007(098b)(097La) 0 008(1098b)(097La) 0 009(1018b)(0766a) 11 019(1018b)(018b) 9 001(1018c)(018b) 19 013(106b)(011b) 10 013(106b)(013b) 11 014(1018b)(013b) 12 014(1018b)(013b) 13 012(1018b)(013b) 14 013(106b)(013b) 15 014(013b)(013b) 16 015(015b)(013b) 17 016(015b)(014b) 18 017(014b)(015La) 19 013(004b)(05La)
4 003 (2011a) (0041a) 5 004 (0098h) (0051b) 6 005 (0048h) (0051b) Machines 6 9 006 (0098h) (0071a) Decidability 10 10 009 (1008h) (0071a) 11 009 (1008h) (0071a) 12 009 (1008h) (0013h) 13 012 (108h) (0118h) 14 013 (0061b) (0131b) 15 014 (0151a) (014b) 16 015 (0158h) (014b) 16 015 (0158h) (014b) 17 016 (0171b) (198b) 18 017 (018b) (012b) 19 013 (0052b) (012b) 19 013 (0052b) (012b)
Turing 6 004(0098b; 008Lb) Machines 6 005(008La) 008La) Machines 7 005(008La) 007La) Decidability 0 007(0098b; 008La) Busy Beaver 12 011(018b; 0118b) Problem 13 012(018Lb; 013Lb) 15 014(018b; 013Lb) 15 16 015(018b; 013Lb) 16 17 015(018b; 013Lb) 16 18 017(018b; 013Lb) 17 19 015(015b; 013Lb) 17 10 015(015b; 013Lb) 17 10 015(015b; 013Lb) 18 10 015(015b; 013Lb) 19
Turing 6 005 (004Ra) (005La) Machines 7 006 (004Ra) (007La) Machines 8 007 (009Rb) (007La) Decidability 10 006 (009Rb) (007La) Busy Beaver 12 011 (012Rb) (012Rb) Problem 13 012 (014La) (013Lb) 15 014 (013Lb) (013Lb) 15 16 015 (015Rb) (013Lb) 16 17 015 (013Lb) (013Lb) 17 18 017 (018Lb) (012La) 19
Machines 7 005 (088La) (07La) Bestan (07La) 8 007 (0097La) (07La) Decidability 10 009 (0198La) (07La) Busy Beaver 12 011 (0127Lb) (018b) (0118b) Problem 13 012 (018La) (013Lb) 15 014 (015La) (013Lb) 15 16 015 (015Lb) (015Lb) 16 17 016 (015La) (015Lb) 17 18 017 (015La) (015La) 19
8 407(4994b) (407(L3) 9 4061(4994b) (407(L3) 9 4061(4994b) (408(L3)) 10 4061(494b) (108La) 11 4061(494b) (108La) Busy Beaver 12 12 111(1124b) (118La) 13 412(1614La) (103La) 14 113(494b) (104Lb) 15 414(151La) (104Lb) 16 115(164b) (104Lb) 17 16(164b) (107Lb) 18 417(164Lb) (17Lb) 19 13(1642b, 107Lb) 19 13(1642b, 107Lb)
Decidability 10 4081 (49943) (4981.3) Busy Beaver 11 416 (1984) (318.4) Problem 12 611 (412.8) (31.8) 13 612 (412.8) (31.8) 14 613 (468.1) (31.8) 15 614 (612.5.4) (31.6) 16 615 (6127.16) (91.16) 17 616 (6127.16) (91.16) 18 617 (618.16) (97.16) 19 613 (692.16.3)
Dectuability 10 definition[10:04] Busy Beaver 12 01101786[01186] Problem 13 0120141[01316] 14 012044[01416] 15 15 0140156[0146] 16 16 01501686[01416] 16 16 01501686[01416] 17 17 01601716[01416] 17 18 017016816[01416] 19 19 01600916[0146] 19
Busy Beaver 12 010(100)[01109] Problem 13 012(1010)[0110b] 14 013(0000)[0130b] 15 014(0151a)[0140b] 16 015(0150)[0140b] 17 016(0171b)[080-) 18 017(0180b)[0151a) 19 012(009b)[0151a)
Busy Beaver 13 012(014La)013La) Problem 14 012(044La)013Lb) 15 014(015La)014Lb) 16 015(016A)015(Lb) 17 016(017Lb)ERA-] 18 017(018Lb)017Lb) 19 013(009A0,015La)
Problem 14 013(006Lb)(013Lb) 15 014(015La)(014Lb) 16 015(0168)(019Lb) 17 016(017Lb)(RR) 18 017(015Lb)(017Lb) 19 018(009Ra)(025La)
13 814(13154)914(b) 16 015(01506)019(b) 17 016(017b)[678-) 18 017(015b)(071b) 19 013(00904)051a)
16 015(016Rb)019Lb) 17 016(017Lb)ERR
17 016(017Lb)ERE) 18 017(018Lb)(017Lb) 19 013(009A1(025La)
18 017(018Lb)017Lb) 19 018(009Ra]025La)
19 018(009Ra 025La)
20 019(020Rb/019Lb)
21 020(002Rb 020Rb)
22 021(022La 021Lb)
23 022(000Ra 024Lb)
24 023(024Lb 023Rb)
25 024(000Ra 024Lb)
26 025(HALT- 024Rb)
27 026(018Lb 026Rb)

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

A possible solution to Goldbach's conjecture?

The Busy Beaver Problem	Tadaa!!			
PRIMES		1	000(023Rb 001Rb)	
		2	001(017La 002Rb)	
Circie		3	002(021La 003Rb)	
		4	003(021La 004La)	
		5	004(009Rb 005Lb)	
		ь -	005(004Ra 005La)	
		/	000(008La 007La)	
		°	007(009RD[007L3)	
		10	009(010Ra1026Ra)	
		11	010(010Rb 011Ra)	
		12	011(012Rb 011Rb)	
		13	012(014La 013La)	
		14	013(006Lb 013Lb)	
		15	014(015La 014Lb)	
		16	015(016Rb 019Lb)	
		17	016(017Lb ERR)	
		18	017(018Lb 017Lb)	
		19	018(009Ra 025La)	
		20	019(020Rb 019Lb)	
		21	020(002Rb 020Rb)	
		22	021(022La 021Lb)	
		23	022(000Ra[024Lb)	
		24	023(024L0 023K0)	
		25	024(000Ka 024LD) 025(HALT_ (024Pb)	
		20	026(018Lb10268b)	
		27	820(810LD]820(D)	

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

Except it runs for a very long time...

A possible solution to Goldbach's conjecture?

The Busy Beaver Problem	Tadaa!!
Problem PRIMES Circle Turing Machines Decidability Busy Beaver Problem	1 000(023Rb)001Rb) 2 001(017La]002Rb) 3 002(02Lla]004La) 4 003(021La]004La) 5 004(009Rb]005Lb) 6 005(004Ra]007La) 7 006(008Ra]007La) 8 007(009Rb]007La) 9 008(009Rc]001La) 10 009(018Ra]02Ra) 11 012(018Ra]02Ra) 12 011(012Rb]01Rb) 12 011(012Rb]01Rb) 13 012(014La]03La) 14 013(005Lb]01Ab) 15 014(015La]014Lb) 16 015(017Lb][08R-)
	10 017(011L) [17(1b)] 10 018(0000) [05(1b)] 20 018(0000) [05(1b)] 21 028(00200) [05(1b)] 22 021(00000) [02(1b)] 23 022(00000) [02(1b)] 24 023(00000) [02(1b)] 25 022(00000) [02(1b)] 26 023(100000) [02(1b)] 27 026(018Lb) [0260b)] 27 026(018Lb) [0260b)] 28 026(018Lb) [0260b)]

How do we know if the program will stop running?

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidabili

Busy Beaver Problem



イロト 不得 トイヨト イヨト

3

The busy beaver problem was introduced by mathematician Tibor Radó in 1962.

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidabilit

Busy Beaver Problem



- The busy beaver problem was introduced by mathematician Tibor Radó in 1962.
- Objective: find the "busiest" algorithm of a given size.

イロト 不得 トイヨト イヨト

3

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidabilit

Busy Beaver Problem



- The busy beaver problem was introduced by mathematician Tibor Radó in 1962.
- Objective: find the "busiest" algorithm of a given size.
- Using this bound, we can figure out whether programs loop indefinitely!

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidabilit

Busy Beavei Problem



- The busy beaver problem was introduced by mathematician Tibor Radó in 1962.
- Objective: find the "busiest" algorithm of a given size.
- Using this bound, we can figure out whether programs loop indefinitely!
- Except we can't compute these upper bounds :(

Table of Contents The Busy Beaver Problem 1 Turing Machines 2 Decidability 3 **Busy Beaver Problem**

▲□▶▲圖▶▲≣▶▲≣▶ ≣ めんの

Ti M

The Busy	.			
Beaver	Recall:			
Problem				
DDIMEC		1	000(023Rb 001Rb)	
		2	001(017La 002Rb)	
Circle		3	002(021La 003Rb)	
		4	003(021La 004La)	
t.a.a.		5	004(009Rb 005Lb)	
ining		6	005(004Ra 005La)	
achines		7	006(008La 007La)	
		8	007(009Rb 007La)	
		9	008(009Ra 008La)	
		10	009(010Ra 026Ra)	
		11	010(010Rb 011Ra)	
		12	011(012Rb 011Rb)	
		13	012(014La 013La)	
		14	013(006Lb 013Lb)	
		15	014(015La 014Lb)	
		16	015(016Rb 019Lb)	
		17	016(017Lb ERR)	
		18	017(018Lb 017Lb)	
		19	018(009Ra 025La)	
		20	019(020Rb 019Lb)	
		21	020(002Rb 020Rb)	
		22	021(022La 021Lb)	
		23	022(000Ra 024Lb)	
		24	023(024Lb 023Rb)	
		25	024(000Ra 024Lb)	
		26	025(HALT- 024Rb)	
		27	026(018Lb 026Rb)	

◆□▶ ◆□▶ ◆ 臣▶ ◆ 臣▶ ○ 臣 ○ の Q @

Τι Μ

The Busy Beaver	Recall:		
Problem			
		1	000(023Rb 001Rb)
PRIMES		2	001(017La 002Rb)
Circle		3	002(021La 003Rb)
		4	003(021La 004La)
		5	004(009Rb 005Lb)
iring		6	005(004Ra 005La)
achines		7	006(008La 007La)
		8	007(009Rb 007La)
		9	008(009Ra 008La)
		10	009(010Ra 026Ra)
		11	010(010Rb 011Ra)
		12	011(012Rb 011Rb)
		13	012(014La 013La)
		14	013(006Lb 013Lb)
		15	014(015La 014Lb)
		16	015(016Rb 019Lb)
		17	016(017Lb ERR)
		18	017(018Lb 017Lb)
		19	018(009Ra 025La)
		20	019(020Rb 019Lb)
		21	020(002Rb 020Rb)
		22	021(022La 021Lb)
		23	022(000Ra 024Lb)
		24	023(024Lb 023Rb)
		25	024(000Ra 024Lb)
		26	025(HALT- 024Rb)
		27	026(018Lb 026Rb)

What is this?

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Answer

Busy Beaver Problem

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Answer

It's the description of a Turing machine!

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

Turing Machines

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beave Problem **Turing machines** are a theoretical model of computation that behave similarly to modern computers.



Turing machine schematic

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

	Languages
The Busy Beaver Problem	
PRIMES Circle	
Turing Machines	 Turing machines recognize whether its input belongs in a language.
Decidability	
Busy Beaver Problem	

◆□▶ ◆□▶ ◆目▶ ◆目▶ ◆□▶

Languages The Busy Beaver Problem Turing machines recognize whether its input belongs in a Turing language. Machines Definition A language is a set of strings that usually follow a certain rule.

- イロト イロト イヨト イヨト ヨー のへぐ

Languages The Busy Beaver Problem Turing machines recognize whether its input belongs in a Turing language. Machines Definition

A language is a set of strings that usually follow a certain rule.

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Example

 $L = \{w | w \in \{0, 1\}\}$



 Turing machines are finite state machines with infinite memory (the tape).

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 のへぐ

Turing Machines Cont.



Turing Machines

Busy Beaver Problem



- Turing machines are finite state machines with infinite memory (the tape).
- Most importantly, the Church-Turing Thesis states that all algorithms map to a corresponding Turing machine

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Turing Machine Example



State diagram of Turing machine

Turing Machine Example

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beave Problem



State diagram for a TM that recognizes $A = \{0^{2^n} | n \ge 0\}$

1 ↓ 0 0 0 0 0 0 0 0 0



Decidability

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaveı Problem

Definition

Language A is decidable if there exists some Turing machine M such that

M accepts all $s \in A$ and rejects all $s \notin A$.

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Decidability

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem

Definition

Language A is decidable if there exists some Turing machine M such that

M accepts all
$$s \in A$$
 and rejects all $s \notin A$.

Corollary

If a language is undecidable, there is no algorithm that decides whether $s \in A$.

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Two Undecidable Languages

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

The Busy Beaver Problem
PRIMES Circle
Decidability

Two Undecidable Languages

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaveı Problem

The first one:

$$A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$$

◆□▶ ◆□▶ ◆ 臣▶ ◆ 臣▶ ○ 臣 ○ の Q @

Two Undecidable Languages

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem The first one:

$$A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$$

The second:

 $HALT_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ halts on input } w \}$

▲□▶▲□▶▲≡▶▲≡▶ ≡ めぬぐ

Proving Undecidability Through Diagonalization

The Busy Beaver Problem PRIMES

Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall: $A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$

Theorem

The language A_{TM} is undecidable.

Proof.

Assume to the contary that TM H decides A_{TM} .

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Proving Undecidability Through Diagonalization

The Busy Beaver Problem PRIMES

Circle

Turing Machines

Decidability

Busy Beaver Problem Recall: $A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$

Theorem

The language A_{TM} is undecidable.

Proof.

Assume to the contary that TM H decides A_{TM} .

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Create TM D as follows:
The Busy Beaver Problem PRIMES

Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall: $A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$

Theorem

The language A_{TM} is undecidable.

Proof.

Assume to the contary that TM H decides A_{TM} .

- Create TM D as follows:
 - It runs H on $\langle M, \langle M \rangle \rangle$.

The Busy Beaver Problem PRIMES

Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall: $A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$

Theorem

The language A_{TM} is undecidable.

Proof.

Assume to the contary that TM H decides A_{TM} .

- Create TM D as follows:
 - It runs H on $\langle M, \langle M \rangle \rangle$.
 - If H accepts, reject.

The Busy Beaver Problem PRIMES

Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall: $A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}$

Theorem

The language A_{TM} is undecidable.

Proof.

Assume to the contary that TM H decides A_{TM} .

▲□▶ ▲□▶ ▲□▶ ▲□▶ □ のQで

- Create TM *D* as follows:
 - It runs H on $\langle M, \langle M \rangle \rangle$.
 - If H accepts, *reject*.
 - If H rejects, accept.

The Busy Beaver Problem PRIMES Circle

-

Turing Machines

Decidability

Busy Beaveı Problem

Fo visualize H	(a decider for A_{TM}).	
----------------	----------------------------	--

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	 $\langle D \rangle$
M1	accept	reject	accept	 accept
M2	reject	reject	reject	 reject
M3	accept	accept	accept	 accept
D	reject	accept	reject	 accept

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

The Busy Beaver Problem PRIMES Circle

-

Turing Machines

Decidability

Busy Beaver Problem

To visualize H	(a decider for A_{TM})	
----------------	---------------------------	--

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	 $\langle D \rangle$
M1	accept	reject	accept	 accept
M2	reject	reject	reject	 reject
M3	accept	accept	accept	 accept
D	reject	accept	reject	 accept

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem To visualize H (a decider for A_{TM})...

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	 $\langle D angle$
M1	accept	reject	accept	 accept
M2	reject	reject	reject	 reject
М3	accept	accept	accept	 accept
D	reject	accept	reject	 accept ?

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beave Problem To visualize H (a decider for A_{TM})...

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	 $\langle D angle$
M1	accept	reject	accept	 accept
M2	reject	reject	reject	 reject
M3	accept	accept	accept	 accept
D	reject	accept	reject	 accept ?

Turing machine D accepts $\langle D \rangle$ if and only if D rejects $\langle D \rangle$.

▲□▶ ▲□▶ ▲□▶ ▲□▶ □ のQで

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beave Problem To visualize H (a decider for A_{TM})...

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	 $\langle D \rangle$
M1	accept	reject	accept	 accept
M2	reject	reject	reject	 reject
М3	accept	accept	accept	 accept
D	reject	accept	reject	 accept ?

Turing machine D accepts $\langle D \rangle$ if and only if D rejects $\langle D \rangle$. There is no such TM that decides A_{TM} .

HALTTN

The Busy Beaver Problem Recall: Decidability $HALT_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ halts on input } w \}.$

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

$HALT_{TM}$

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

 $HALT_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ halts on input } w \}.$

Let's take advantage of its similarity with A_{TM} .

	Reducibility
The Busy Beaver Problem PRIMES Circle	
Turing Machines Decidability Busy Beaver Problem	Example Is it possible for me to go to the beach tomorrow?

◆□▶ ◆□▶ ◆目▶ ◆目▶ ◆□ ◆

Reducibility The Busy Beaver Problem Example Decidability Is it possible for me to go to the beach tomorrow? Reduces to: Do I have a ride?

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem

Recall:

 $HALT_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ halts on input } w \}, \\ A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}.$

◆□▶ ◆□▶ ◆□▶ ◆□▶ □ のQ@

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem

Recall:

 $\begin{aligned} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ halts on input } w \}, \\ & A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}. \end{aligned}$

▲ロ ▶ ▲周 ▶ ▲ 国 ▶ ▲ 国 ▶ ● の Q @

Theorem

The language $HALT_{TM}$ is undecidable.

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

$$\begin{split} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ halts on input } w \}, \\ & \textit{A}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ accepts } w \}. \end{split}$$

Theorem

The language $HALT_{TM}$ is undecidable.

Proof. Assume to the contrary that TM S decides $HALT_{TM}$.

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

$$\begin{split} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ halts on input } w \}, \\ & \textit{A}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ accepts } w \}. \end{split}$$

Theorem

The language $HALT_{TM}$ is undecidable.

Proof. Assume to the contrary that TM *S* decides $HALT_{TM}$. • Create TM *H* that takes in $\langle M, w \rangle$ as follows:

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

$$\begin{split} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ halts on input } w \}, \\ & \textit{A}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ accepts } w \}. \end{split}$$

Theorem

The language $HALT_{TM}$ is undecidable.

Proof. Assume to the contrary that TM S decides $HALT_{TM}$.

▲□▶ ▲□▶ ▲□▶ ▲□▶ □ のQで

- Create TM H that takes in $\langle M, w \rangle$ as follows:
 - It runs S on $\langle M, w \rangle$.

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

 $\begin{aligned} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ halts on input } w \}, \\ & A_{TM} = \{ \langle M, w \rangle | M \text{ is a TM and } M \text{ accepts } w \}. \end{aligned}$

Theorem

The language $HALT_{TM}$ is undecidable.

Proof. Assume to the contrary that TM S decides $HALT_{TM}$.

▲□▶ ▲□▶ ▲□▶ ▲□▶ □ のQで

- Create TM H that takes in $\langle M, w \rangle$ as follows:
 - It runs S on $\langle M, w \rangle$.
 - If S rejects, reject.

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

$$\begin{split} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ halts on input } w \}, \\ & \textit{A}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ accepts } w \}. \end{split}$$

Theorem

The language $HALT_{TM}$ is undecidable.

Proof. Assume to the contrary that TM S decides $HALT_{TM}$.

- Create TM H that takes in $\langle M, w \rangle$ as follows:
 - It runs S on $\langle M, w \rangle$.
 - If S rejects, reject.
 - If S accepts, run M on w. Do what M does.

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Recall:

$$\begin{split} & \textit{HALT}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ halts on input } w \}, \\ & \textit{A}_{TM} = \{ \langle M, w \rangle | \textit{M} \text{ is a TM and } \textit{M} \text{ accepts } w \}. \end{split}$$

Theorem

The language $HALT_{TM}$ is undecidable.

Proof. Assume to the contrary that TM S decides $HALT_{TM}$.

- Create TM H that takes in $\langle M, w \rangle$ as follows:
 - It runs S on $\langle M, w \rangle$.
 - If S rejects, reject.
 - If S accepts, run M on w. Do what M does.

Because $HALT_{TM}$ reduces to A_{TM} , it is also undecidable.

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

The Busy Beaver Game

The Busy Beaver Problem PRIMES

Circle

Turing Machines

Decidability

Busy Beaver Problem

To reiterate: the busy beaver problem

Find the maximum number of computations a halting algorithm with a given size can perform.

The Busy Beaver Game

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

To reiterate: the busy beaver problem

Find the maximum number of computations a halting algorithm with a given size can perform.

But more specifically: the busy beaver problem

Find the maximum the number of computations a halting algorithm **Turing machine** with a given size **number of states** can perform.

▲□▶ ▲□▶ ▲□▶ ▲□▶ ▲□ ● ● ●

The Busy Beaver Game cont.

The Busy Beaver Problem PRIMES Circle

Turing Machines Decidabilit

Busy Beaver Problem Our metric for *number of computations* is the number of state shifts a Turing machine undergoes.

Definition

BB(n) := the maximum number of shifts for a Turing machine with *n* non-halt states.

Example: BB(2) = 6

Modification: this Turing machine has an all-encompassing halt state.



Example: BB(2) = 6

Modification: this Turing machine has an all-encompassing halt state.



▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三 のへぐ

Example: BB(2) = 6

Modification: this Turing machine has an all-encompassing halt state.



The Busy Beaver Problem PRIMES Circle

A few busy beaver numbers:

$$BB(1) = 1$$

Turing Machines

Decidability

Busy Beaver Problem

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem A few busy beaver numbers:

$$\blacksquare BB(1) = 1$$

$$BB(2)=6$$

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem A few busy beaver numbers:

▲□▶ ▲□▶ ▲ 三▶ ▲ 三▶ 三三 - のへぐ

•
$$BB(1) = 1$$

$$BB(2)=6$$

■ *BB*(3) = 21

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem A few busy beaver numbers:

◆□▶ ◆□▶ ◆ 臣▶ ◆ 臣▶ ○ 臣 ○ の Q @

$$BB(1) = 1$$

$$\blacksquare BB(2) = 6$$

B
$$B(3) = 21$$

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem A few busy beaver numbers:

- BB(1) = 1
- *BB*(2) = 6
- *BB*(3) = 21

B
$$B(5) \stackrel{?}{=} 47\,176\,870$$

▲□▶ ▲□▶ ▲ □▶ ▲ □▶ □ のへぐ

The Busy Beaver Problem PRIMES

Turing Machines

Decidability

Busy Beaver Problem A few busy beaver numbers:

- BB(1) = 1
- *BB*(2) = 6
- *BB*(3) = 21
- *BB*(4) = 107
- **BB**(5) $\stackrel{?}{=}$ 47 176 870
- $BB(6) \ge 7.4 \times 10^{36534}$

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00

The Busy Beaver Problem

Busy Beaver Problem

A few busy beaver numbers:

- BB(1) = 1
- **B**B(2) = 6
- **B**B(3) = 21
- BB(4) = 107
- **BB**(5) $\stackrel{?}{=}$ 47 176 870
- $BB(6) \ge 7.4 \times 10^{36534}$

 $\blacksquare BB(7) \ge 10^{10^{10^{10^{10^{18705353}}}}$

An exaFLOPS computer system can perform $\sim 10^{18}$ floating-point operations per second.

Why do Busy Beaver numbers matter?

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem

Theorem (Matiyasevich, O'Rear 2016)

There is a 744-state Turing machine that halts if and only if the Riemann Hypothesis is false.

Why do Busy Beaver numbers matter?

The Busy Beaver Problem PRIMES Circle

Turing Machines

Busy Beaver Problem

Theorem (Matiyasevich, O'Rear 2016)

There is a 744-state Turing machine that halts if and only if the Riemann Hypothesis is false.

Theorem (Anonymous GitHub user 2015)

There is a 27-state Turing machine that halts if and only if Goldbach's conjecture is false.

Acknowledgements

The Busy Beaver Problem PRIMES Circle

Turing Machines

Decidability

Busy Beaver Problem First and foremost, we would like to thank to our phenomenal, cool, intelligent mentor Alexandra!
Thanks to the amazing Primes Circle coordinators Marisa and Mary!
Thanks to Sarah's computer science teacher, Tom, Michael Sipser for writing such an informative book, and Scott Aaronson for being a g.

And thank you all for coming to our presentation!

▲□▶ ▲□▶ ▲□▶ ▲□▶ ■ ●の00
References

The Busy Beaver Problem

Busy Beaver Problem



Kun, Jeremy. (2012). Busy Beavers, and Quest for Big Numbers https://jeremykun.com/2012/02/08/ busy-beavers-and-the-quest-for-big-numbers/



Mullins, Robert. (2012). What is a Turing machine? [illustration] https://www.cl.cam.ac.uk/projects/raspberrypi/ tutorials/turing-machine/one.html



Sipser, M. (2013). Introduction to the Theory of *Computation* (3rd ed.). Cengage Learning.