Unit 3: Algorithms

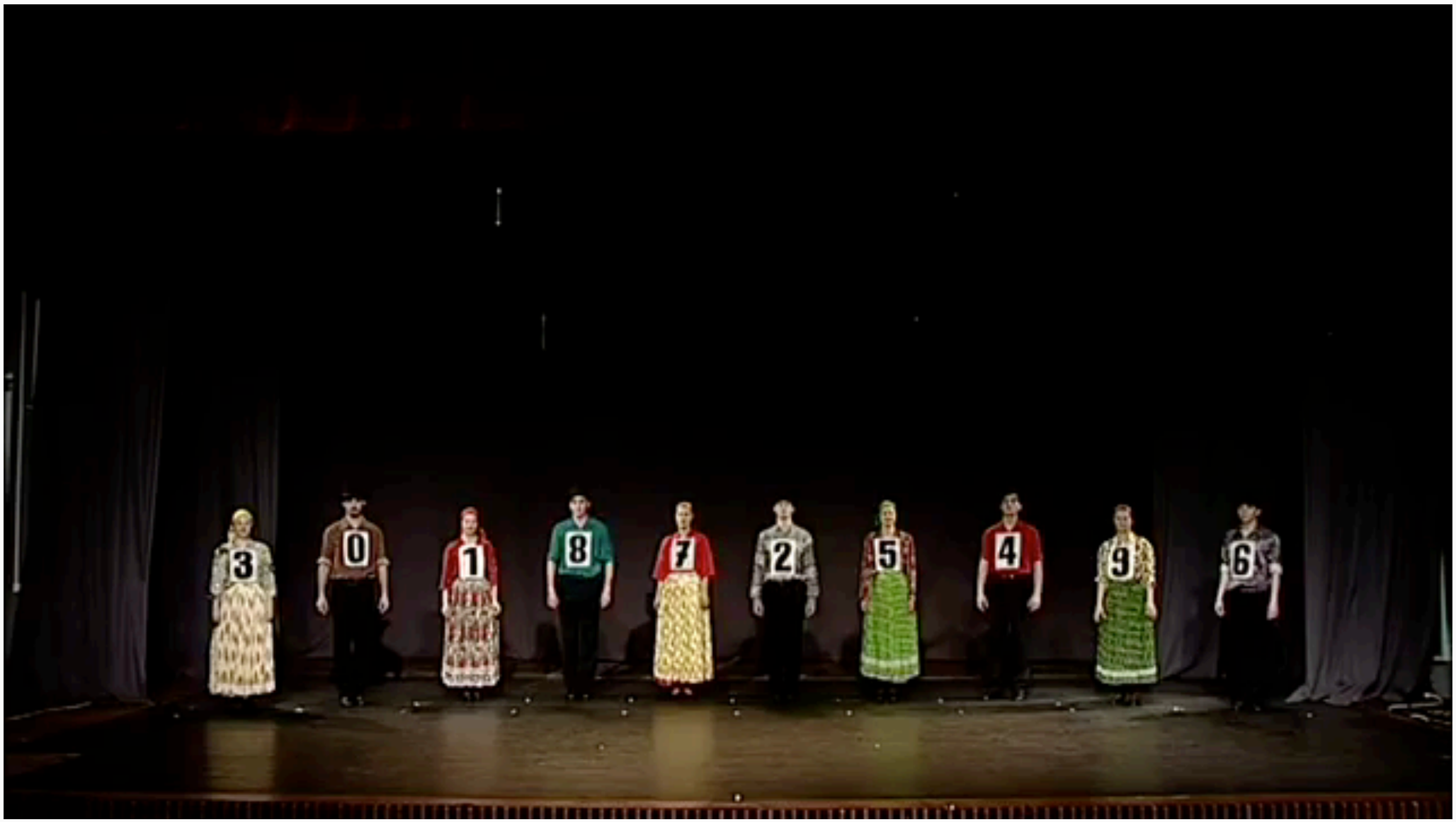
Dave Abel

February 19th, 2016

Outline

- Dancing!
- Growth Rates Part 3
- A third problem: *satisfiability*
  - Two algorithms for solving it..
- A fourth problem: *will it halt?*

Dancing Selection Sort



Selection Sort: Growth Rate

- Do this for the first smallest, second smallest, all the way up to Nth smallest:
  - **“Select” the current smallest item in the list.**
  - Put it at the beginning.

takes N steps

doing this N times

So, $N \times N = N^2$

Selection Sort: Growth Rate

- Do this for the first smallest, second smallest, all the way up to Nth smallest:
  - "Select" the current smallest item in the list.
  - Put it at the beginning.

takes N steps

doing this N times

So, $N \times N = N^2$

Q: Does this *really* take *N* Steps?

Selection Sort: Growth Rate

- Do this for the **first smallest**, second smallest, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.

takes N steps

Q: Does this *really* take *N* Steps?

Selection Sort: Growth Rate

- Do this for the **first smallest**, second smallest, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.

takes N steps

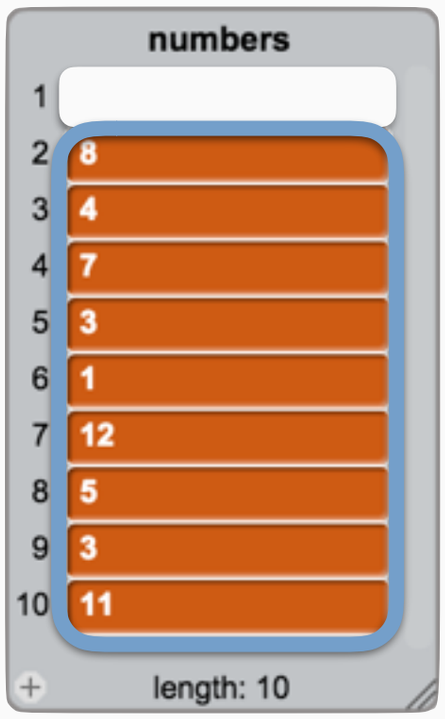
Q: Does this *really* take *N* Steps?

Selection Sort: Growth Rate

- Do this for the first smallest, **second smallest**, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.



takes N steps

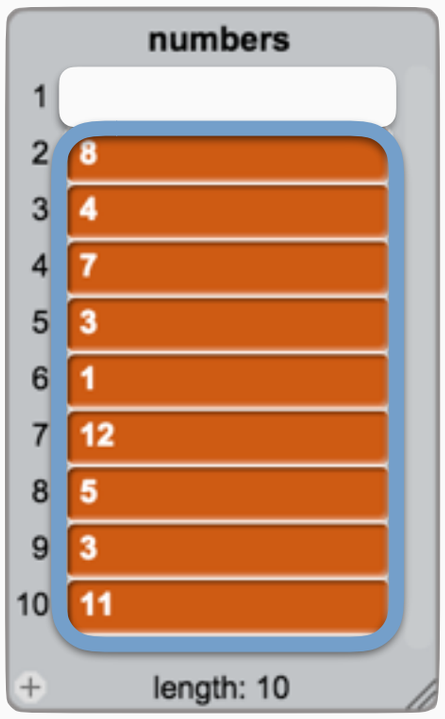
Q: Does this *really* take *N* Steps?

Selection Sort: Growth Rate

- Do this for the first smallest, **second smallest**, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.



takes N steps

But this is only *N-1*

Q: Does this *really* take *N* Steps?

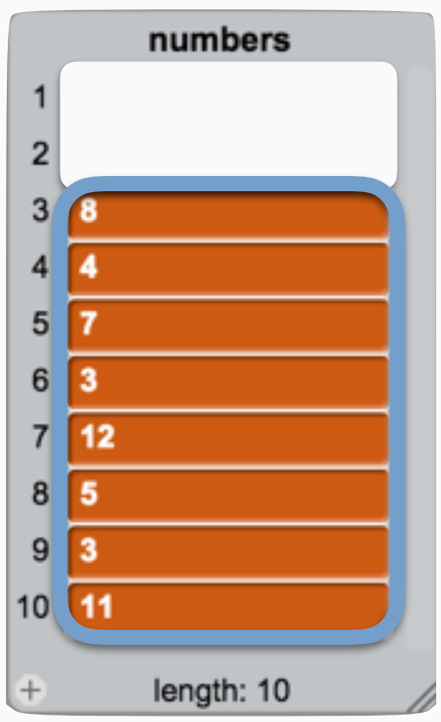
Selection Sort: Growth Rate

- Do this for the first smallest, **second smallest**, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.

takes *N* steps



But this is only *N-2*

Q: Does this *really* take *N* Steps?

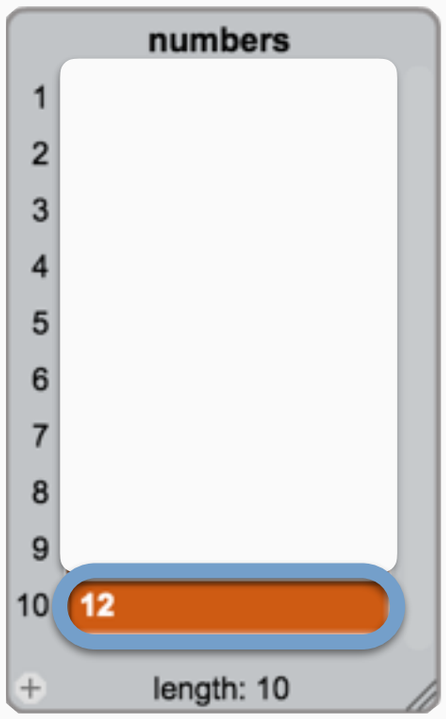
Selection Sort: Growth Rate

- Do this for the first smallest, **second smallest**, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.

takes *N* steps



But this is only *1* step

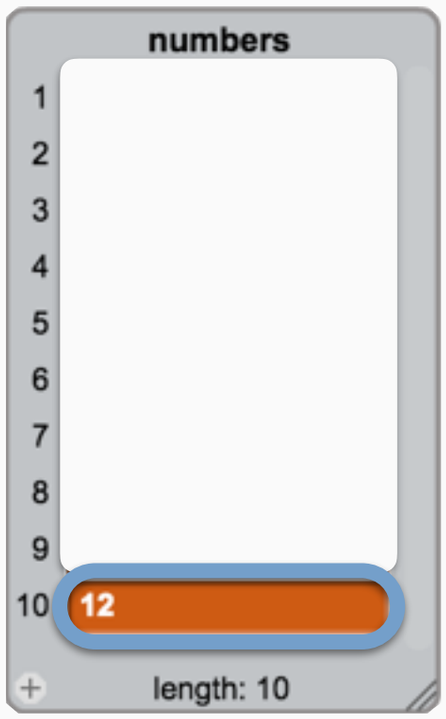
Q: Does this *really* take *N* Steps?

Selection Sort: Growth Rate

- Do this for the first smallest, **second smallest**, all the way up to Nth smallest:

  - "Select" the current smallest item in the list.

  - Put it at the beginning.



But this is only *1* step

takes N steps

Q: Does this *really* take *N* Steps?

A: First its *N*, then *N - 1*, then *N - 2*...

Selection Sort: Growth Rate

A: First its *N*, then *N - 1*, then *N - 2*…

$$N + (N - 1) + (N - 2) + \ldots 3 + 2 + 1$$

Selection Sort: Growth Rate

A: First its *N*, then *N - 1*, then *N - 2*…

N + (N - 1) + (N - 2) + … 3 + 2 + 1

From Homework: this is just *N(N+1)/2*

Selection Sort: Growth Rate

A: First its *N*, then *N - 1*, then *N - 2…*

N + (N - 1) + (N - 2) + … 3 + 2 + 1

From Homework: this is just *N(N+1)/2* = $(N^2 + N)/2$

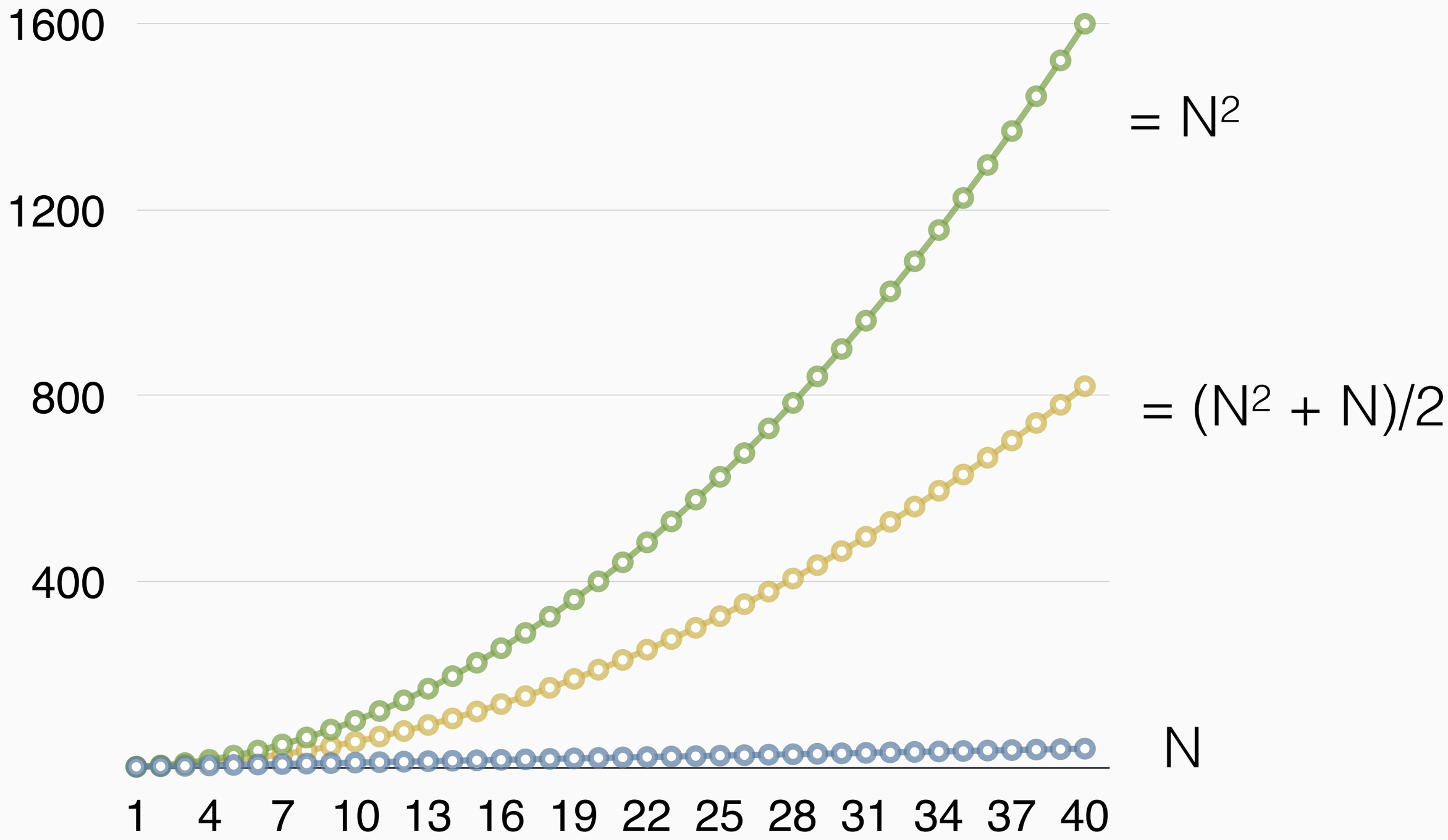
Selection Sort: Growth Rate

A: First its *N*, then *N - 1*, then *N - 2*…

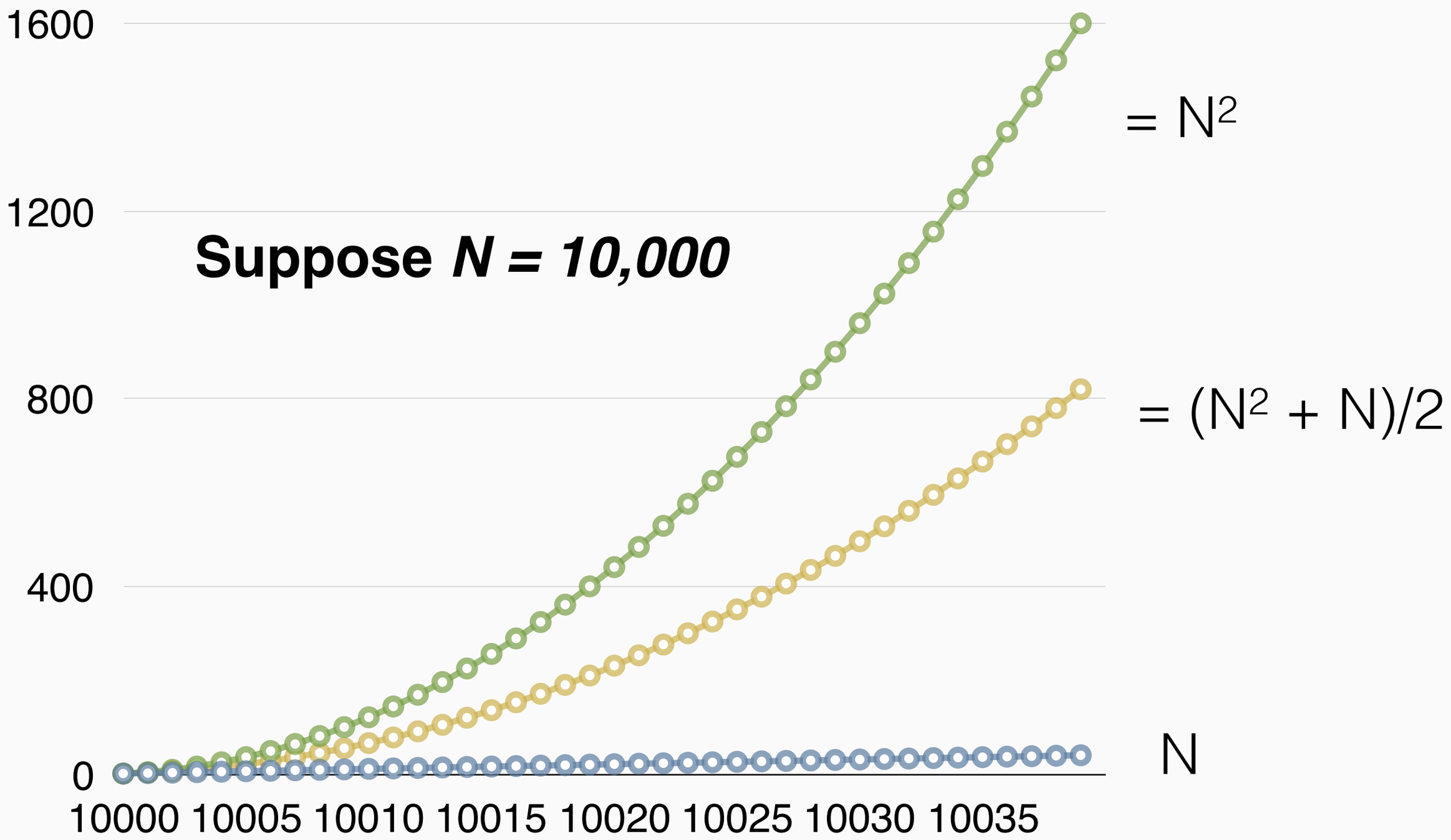
N + (N - 1) + (N - 2) + … 3 + 2 + 1

From Homework: this is just *N(N+1)/2* = (**N²** + N)/2

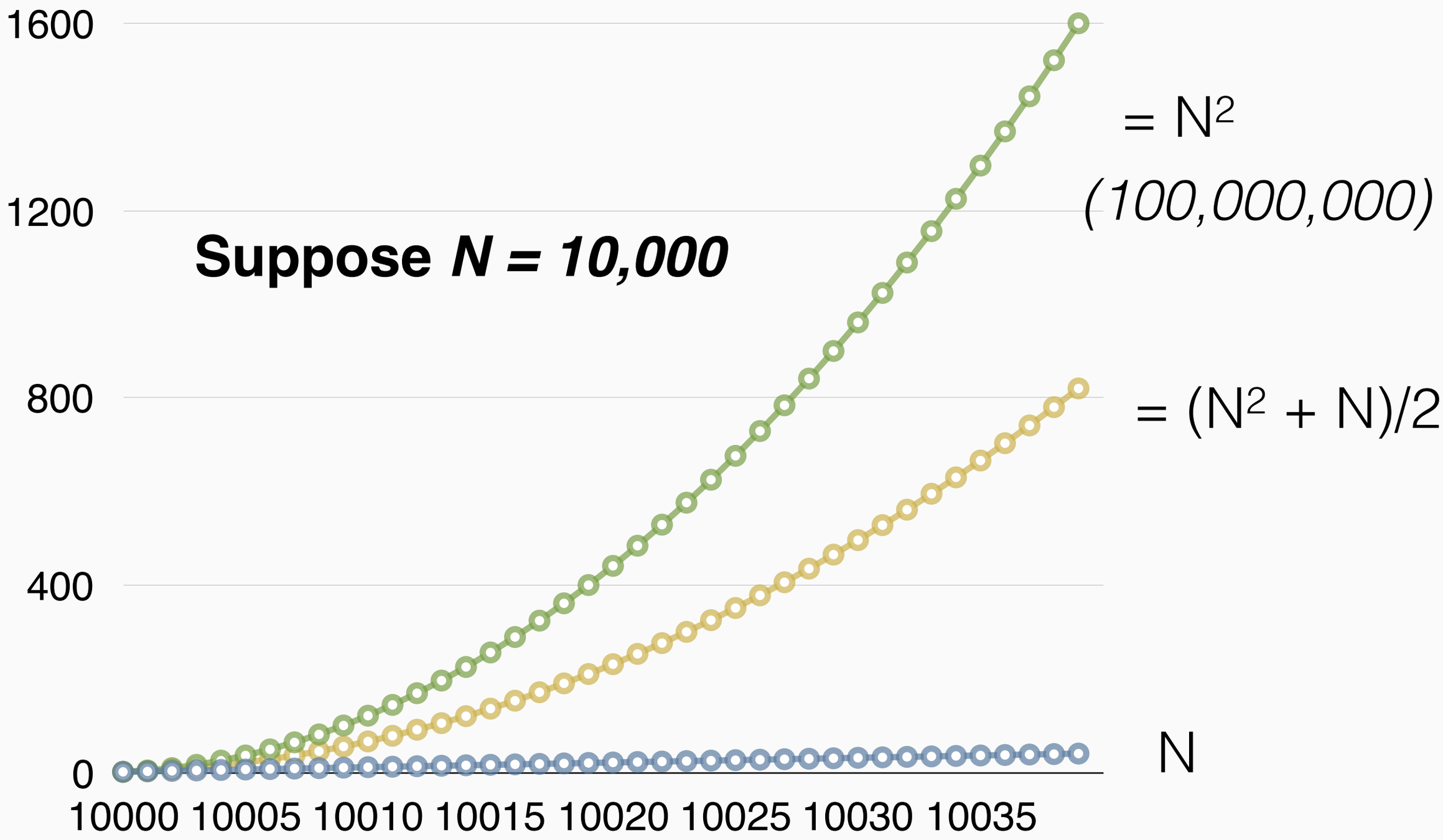
Selection Sort: Growth Rate



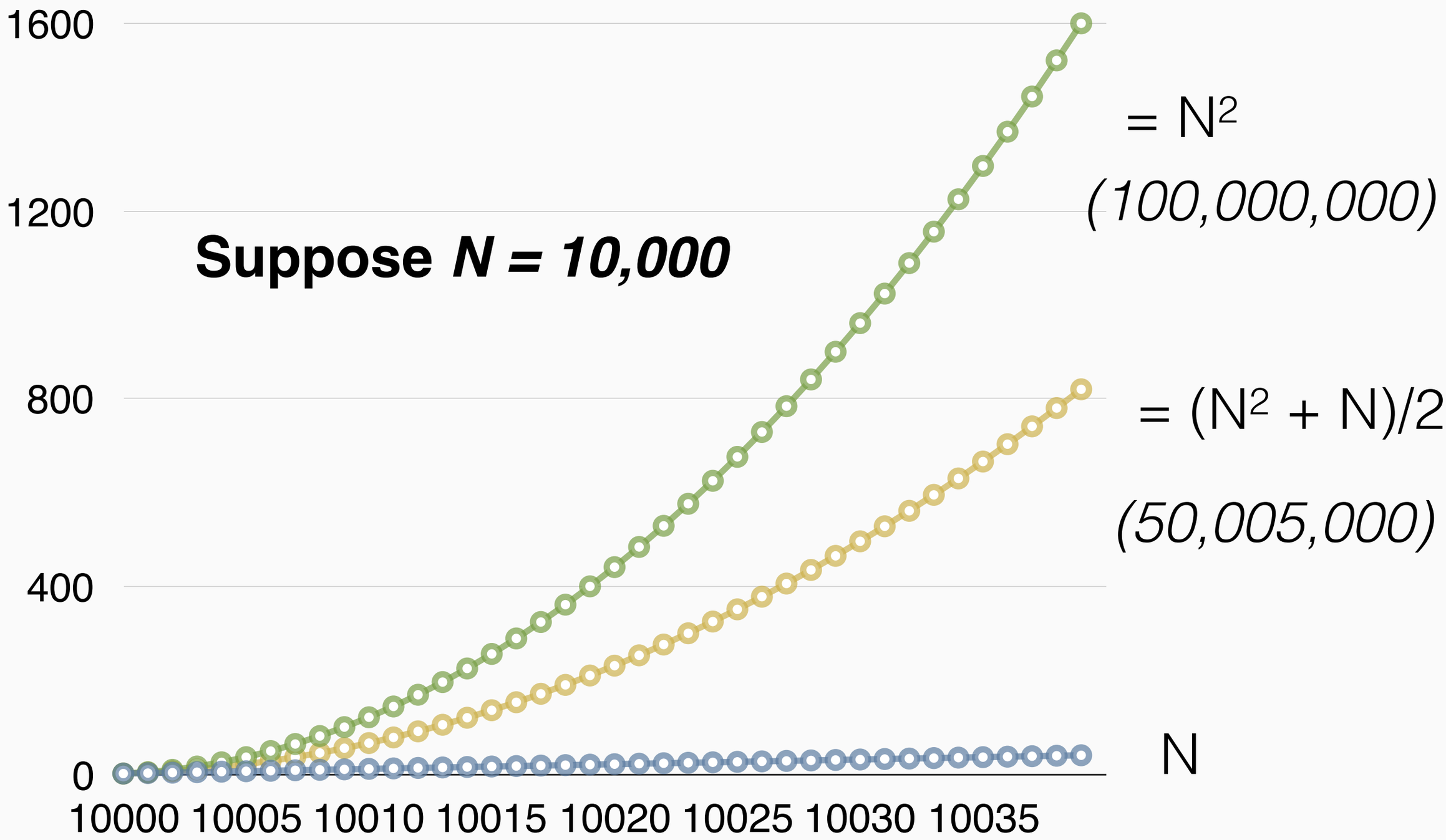
Selection Sort: Growth Rate



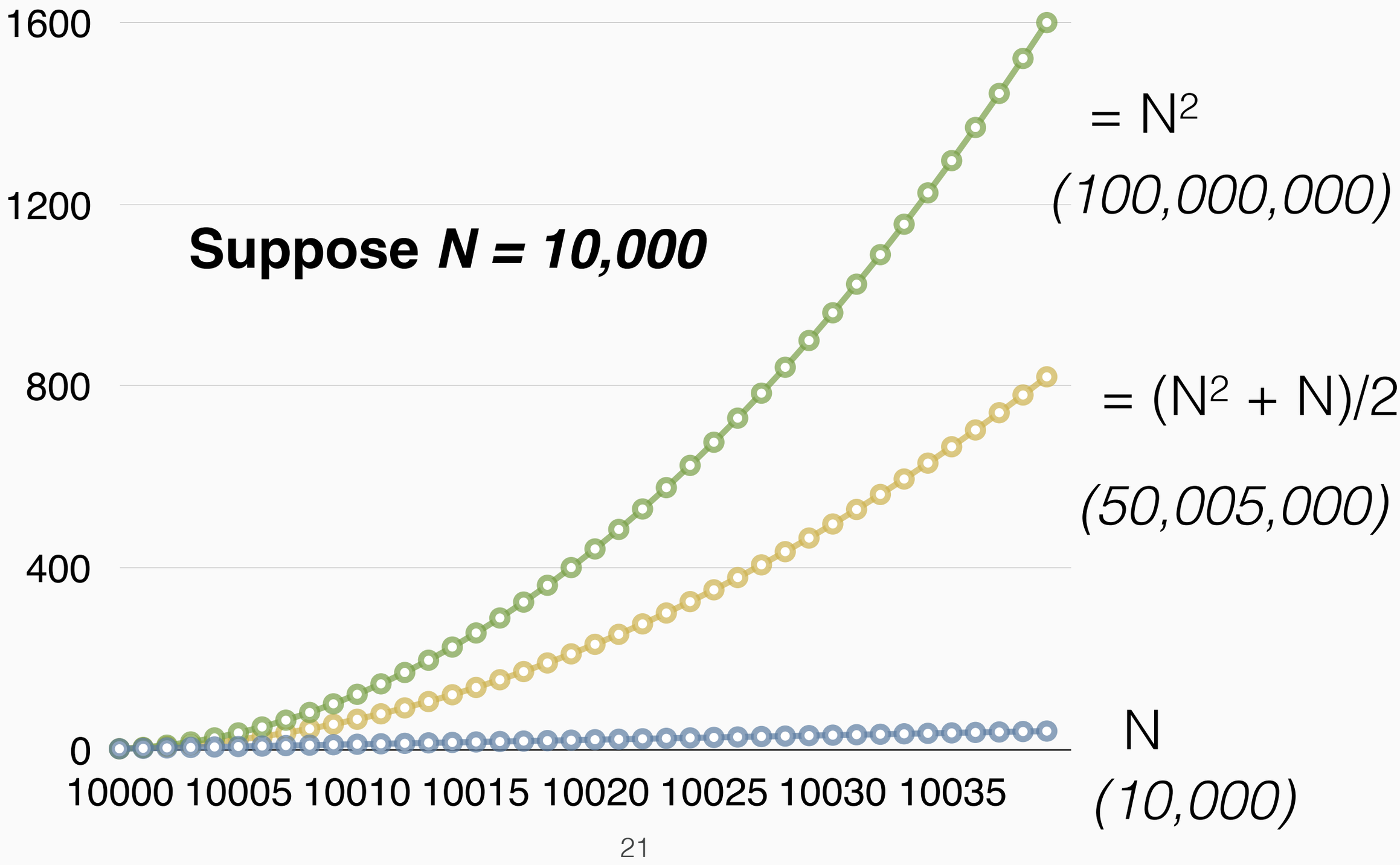
Selection Sort: Growth Rate



Selection Sort: Growth Rate



Selection Sort: Growth Rate



Selection Sort: Growth Rate

Takeaway: we only *really* pay attention to the biggest & most dominant term in the growth rate.

Selection Sort: Growth Rate

Takeaway: we only *really* pay attention to the biggest & most dominant term in the growth rate.

So: *2N + 1*? Actually just *N*

Selection Sort: Growth Rate

Takeaway: we only *really* pay attention to the biggest & most dominant term in the growth rate.

So: $2N + 1$? Actually just N

So: $3^N + N^2$? Actually just 3^N

Clicker Question!

What is the growth rate of $4N^3 + 2N^2 + N + 3\log_2 N$?

Clicker Question!

[A] $4N^3$

[B] $4N^3 + 2N^2$

[C] N^2

[D] N^3

[D] N

What is the growth rate of $4N^3 + 2N^2 + N + 3log_2N$?

Clicker Answer!

[A] $4N^3$ [B] $4N^3 + 2N^2$ [C] N^2

[D] N^3 [D] N

What is the growth rate of $4N^3 + 2N^2 + N + 3log_2N$?

Clicker Answer!

[A] $4N^3$ [B] $4N^3 + 2N^2$ [C] N^2

[D] N^3 [D] N

Since N^3 is the most dominant term… We don't care about constants

What is the growth rate of $4N^3 + 2N^2 + N + 3log_2N$?

A Third Problem:
Can this logical formula be true?

- *INPUT:*
  - *A* logical formula
- *OUTPUT:*
  - True if there is some way we can make the formula true
  - False if there is no way we can make the formula true

A Third Problem: *Satisfiability*

- *INPUT:*
  - A boolean sentence.
- *OUTPUT:*
  - True if there is some way we can make the formula true
  - False if there is no way we can make the formula true

Satisfiability

I give you: *AND(P,Q)*

Q: Can this be true?

(our tools are how we set the truth value of *P, Q*)

Satisfiability

I give you: *AND(P,Q)*

Q: Can this be true?

A: Sure! *P = True, Q = True*

(our tools are how we set the truth value of *P, Q*)

Satisfiability

- Instead of our growth rate capturing the length of the list, now it measures the *number of atomic sentences, e.g. P, Q, R, etc.*

- So, how hard is this problem?

- Let's try a harder one…

Clicker Question!

Q: Can the following logical formula be made True?

NOT(OR(AND(OR(P,Q),NOT(R)),R))

or if you'd prefer the other way of writing things…

not(((P or Q) and not(R)) or R)

Clicker Question!

Q: Can the following logical formula be made True?

NOT(OR(AND(OR(P,Q),NOT(R)),R))

or if you'd prefer the other way of writing things…

not(((P or Q) and not(R)) or R)

[A] Yes, I'm very confident! [B] No, I'm very confident!

[C] I'm guessing randomly!

Clicker Answer!

Q: Can the following logical formula be made True?

NOT(OR(AND(OR(P,Q),NOT(R)),R))

or if you'd prefer the other way of writing things…

not(((P or Q) and not(R)) or R)

A: Yes! *P = False, Q = False, R = False*

Algorithm 1: Satisfiability

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(P and Q) or not(P)

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(P and Q) or not(P)

P = True, Q = False

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(P and Q) or not(P)

P = True, Q = False

.....(True and False) or not(True)

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(P and Q) or not(P)

P = True, Q = False

(True and False) or not(True)

report True!

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(P and Q) or not(P)

P = True, Q = False

(True and False) or not(True)

report True! ✓

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

The usual questions…

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

The usual questions… Q: Does it halt?

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

The usual questions… Q: Does it halt? A: No!

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(what if there is no true assignment?)

The usual questions… Q: Does it halt? A: No!

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

The usual questions… Q: It is *correct?* A: No!

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

(what if there is no true assignment?)

The usual questions… Q: It is *correct?* A: No!

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

The usual questions… Q: What's the growth rate?

Satisfiability: Random Checker

Random Checker

1. Pick a random assignment for each atomic sentence's truth value.

2. Ask, is the sentence True?

3. If it is, report True.

4. If not, repeat from step 1.

The usual questions…

Q: What's the growth rate?

A: Actually hard to say…

Algorithm 2: Satisfiability

Build The Truth Table! (Brute Force)

1. *Build the truth table for the logical formula.*
2. *Check to see if it has any row that is True.*
3. *If it does, report True.*
4. *If not, report False.*

Satisfiability: Build the Table!

(P and Q) or not(P)

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Clicker Question!

(((P and Q) or not(R)) or S)

Q: How many rows will there be in the truth table?

[A] 8 [B] 12 [C] 14 [D] 16 [E] 18

Clicker Question!

(((P and Q) or not(R)) or S)

Q: How many rows will there be in the truth table?

[A] 8 [B] 12 [C] 14 [D] 16 [E] 18

Clicker Question!

(((P and Q) or not(R)) or S)

Q: How many rows will there be in the truth table?

[A] 8 [B] 12 [C] 14 [D] 16 [E] 18

Hint 1: We need each combination of P, Q, and R being True/False…

Clicker Question!

(((P and Q) or not(R)) or S)

Q: How many rows will there be in the truth table?

[A] 8 [B] 12 [C] 14 [D] 16 [E] 18

Hint 1: We need each combination of P, Q, and R being True/False…

Hint 2: True can be 1, False can be 0…

Clicker Question!

(((P and Q) or not(R)) or S)

Q: How many rows will there be in the truth table?

[A] 8 [B] 12 [C] 14 [D] 16 [E] 18

Hint 1: We need each combination of P, Q, and R being True/False…

Hint 2: True can be 1, False can be 0…

Hint 3: We have three atomic sentences, i.e. three “bits”

Clicker Answer!

(((P and Q) or not(R)) or S)

Q: How many rows will there be in the truth table?

[A] 8 [B] 12 [C] 14 **[D] 16** [E] 18

Hint 1: We need each combination of P, Q, and R being True/False…

Hint 2: True can be 1, False can be 0…

Hint 3: We have three atomic sentences, i.e. three “bits”

Clicker Answer!

P	*Q*	*R*	*S*
T	T	T	T
T	T	T	F
T	T	F	T
T	T	F	F
T	F	T	T
T	F	T	F
T	F	F	T
T	F	F	F
F	T	T	T
F	T	T	F
F	T	F	T
F	T	F	F
F	F	T	T
F	F	T	F
F	F	F	T
F	F	F	F

Q: How many rows will there be in the truth table?

Clicker Answer!

Q: How many rows will there be in the truth table?

Can think about it as an N bit binary number!

P	*Q*	*R*	*S*
T	T	T	T
T	T	T	F
T	T	F	T
T	T	F	F
T	F	T	T
T	F	T	F
T	F	F	T
T	F	F	F
F	T	T	T
F	T	T	F
F	T	F	T
F	T	F	F
F	F	T	T
F	F	T	F
F	F	F	T
F	F	F	F

Clicker Answer!

Q: How many rows will there be in the truth table?

Can think about it as an N bit binary number!

P	*Q*	*R*	*S*
1	1	1	1
1	1	1	0
1	1	0	0
1	1	0	0
1	0	1	0
1	0	1	0
1	0	0	1
1	0	0	0
0	1	1	1
0	1	1	0
0	1	0	1
0	1	0	0
0	0	1	1
0	0	1	0
0	0	0	1
0	0	0	0

Clicker Answer!

Q: How many rows will there be in the truth table?

Can think about it as an N bit binary number!

More generally: for a sentence of N variables, there will be 2^N combinations.

P	*Q*	*R*	*S*
1	1	1	1
1	1	1	0
1	1	0	0
1	1	0	0
1	0	1	0
1	0	1	0
1	0	0	1
1	0	0	0
0	1	1	1
0	1	1	0
0	1	0	1
0	1	0	0
0	0	1	1
0	0	1	0
0	0	0	1
0	0	0	0

Satisfiability: Build the Table!

(P and Q) or not(P)

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Satisfiability: Build the Table!

Build the Table!

1. **Build the Truth Table**
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

(P and Q) or not(P)

P	*Q*	*R*	*(P and Q)*	*or*	*not(P)*
T	T	T			
T	T	F			
T	F	T			
T	F	F			
F	T	T			
F	T	F			
F	F	T			
F	F	F			

Satisfiability: Build the Table!

Build the Table!

1. **Build the Truth Table**
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

(P and Q) or not(P)

P	*Q*	*R*	*(P and Q)*	*or*	*not(P)*
T	T	T	T	T	F
T	T	F	T	T	F
T	F	T	F	F	F
T	F	F	F	F	F
F	T	T	F	T	T
F	T	F	F	T	T
F	F	T	F	T	T
F	F	F	F	T	T

Satisfiability: Build the Table!

Build the Table!

1. Build the Truth Table
2. **Check to see if any row is True.**
3. If so, report True.
4. Otherwise, report False.

(P and Q) or not(P)

P	*Q*	*R*	*(P and Q)*	*or*	*not(P)*
T	T	T	T	T	F
T	T	F	T	T	F
T	F	T	F	F	F
T	F	F	F	F	F
F	T	T	F	T	T
F	T	F	F	T	T
F	F	T	F	T	T
F	F	F	F	T	T

Satisfiability: Build the Table!

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. **If so, report True.**
4. Otherwise, report False.

True!

(P and Q) or not(P)

P	*Q*	*R*	*(P and Q)*	*or*	*not(P)*
T	T	T	T	T	F
T	T	F	T	T	F
T	F	T	F	F	F
T	F	F	F	F	F
F	T	T	F	T	T
F	T	F	F	T	T
F	F	T	F	T	T
F	F	F	F	T	T

Satisfiability: Build the Table!

- Q1: Will it halt?
- Q2: Is it correct?
- Q3: What's the growth rate?

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Satisfiability: Build the Table!

- Q1: Will it halt?

  A: Yes! Build a truth table…

- Q2: Is it correct?

- Q3: What's the growth rate?

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Satisfiability: Build the Table!

- Q1: Will it halt?

  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- Q3: What's the growth rate?

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Satisfiability: Build the Table!

- Q1: Will it halt?

  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- Q3: What's the growth rate?

  A: Clicker question!

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Clicker Question!

[A] 2^N [B] N [C] N^2 [D] N*log(N)

N = number of atomic sentences in the formula (e.g. *P, Q*)

Q3: What's the growth rate of the "build the table" algorithm?

Clicker Answer!

[A] 2^N [B] N [C] N^2 [D] N*log(N)

N = number of atomic sentences in the formula (e.g. *P, Q*)

Q3: What's the growth rate of the "build the table" algorithm?

Clicker Answer!

"Primitive operation" here is building a truth table row

[A] 2^N [B] N [C] N^2 [D] N*log(N)

N = number of atomic sentences in the formula (e.g. *P, Q*)

Q3: What's the growth rate of the "build the table" algorithm?

Clicker Answer!

"Primitive operation" here is building a truth table row

[A] 2^N [B] N [C] N^2 [D] N*log(N)

We just said there were 2^N rows for an *N* variable formula

N = number of atomic sentences in the formula (e.g. *P, Q*)

Q3: What's the growth rate of the "build the table" algorithm?

Satisfiability: Build the Table!

- Q1: Will it halt?

  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- Q3: What's the growth rate?

  A: 2^N

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Satisfiability: Build the Table!

- Q1: Will it halt?

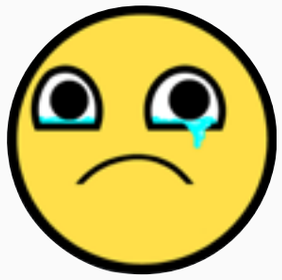
  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- Q3: What's the growth rate?

  A: 2^N



If $N = 100$, this is already basically unsolvable…

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Satisfiability: Build the Table!

- Q1: Will it halt?

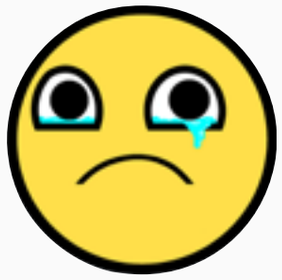
  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- Q3: What's the growth rate?

  A: 2^N



If $N = 100$, this is already basically unsolvable…

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Can We Do Better?

- Q1: Will it halt?

  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- Q3: What's the growth rate?

  A: 2^N

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Can We Do Better?

- Q1: Will it halt?

  A: Yes! Build a truth table…

- Q2: Is it correct?

  A: Yes! Truth table tells all…

- **Q3: What's the growth rate?**

  A: 2^N

Build the Table!

1. Build the Truth Table
2. Check to see if any row is True.
3. If so, report True.
4. Otherwise, report False.

Clicker Question!

[A] Yes!

[B] No!

Q: Is there a correct algorithm to solve the Satisfiability problem with a growth rate faster than 2^N?

Clicker Answer!

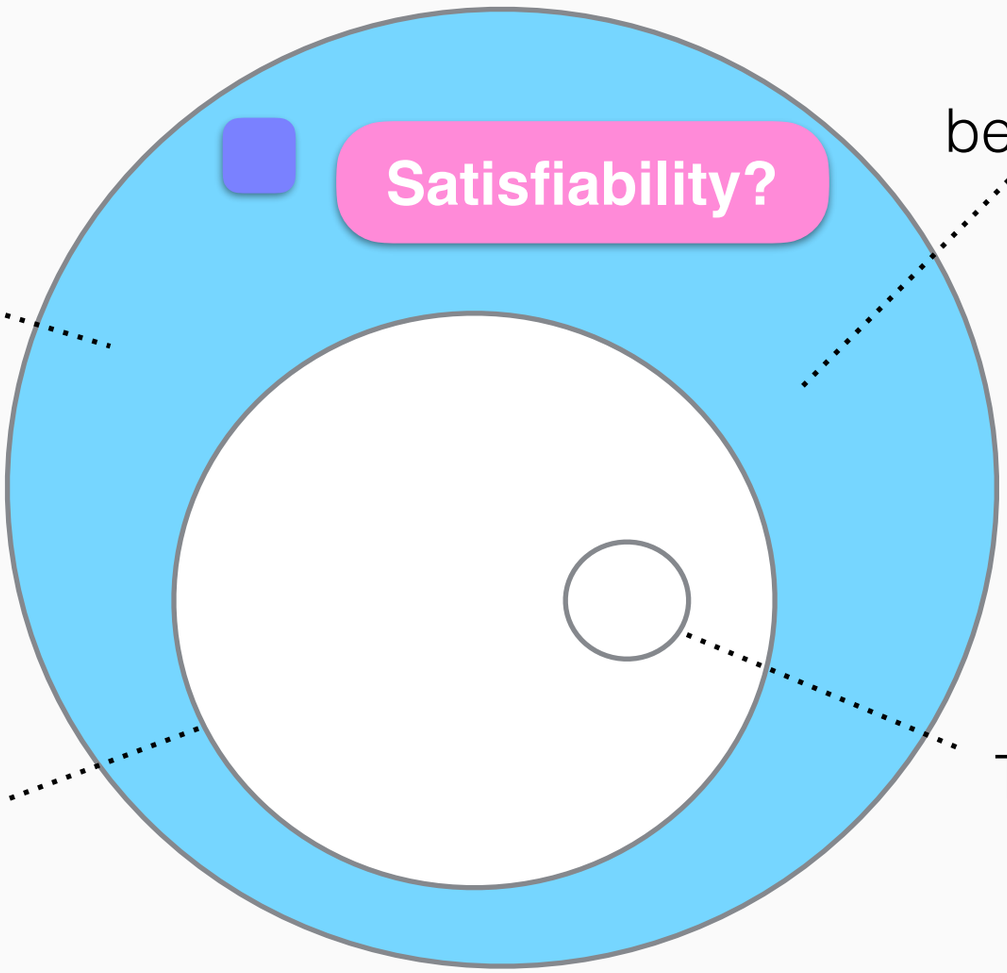
[A] Yes! [B] No!

A: We don't know!

Q: Is there a correct algorithm to solve the Satisfiability problem with a growth rate faster than 2^N?

Computability

Q: What, if anything, is out here?



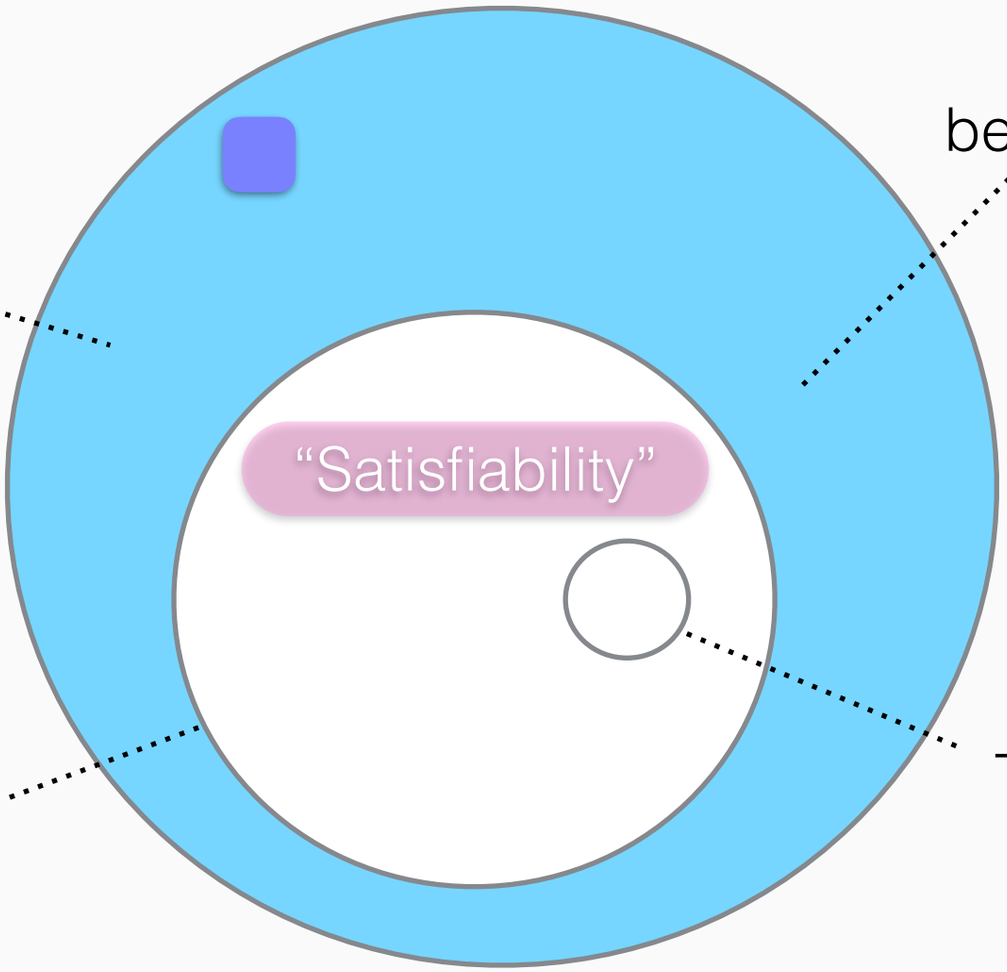
Things that can be computed, period.

Things a regular computer can compute before the sun goes supernova

Things a domino computer could compute before the sun goes supernova

Computability

Q: What, if anything, is out here?



Things that can be computed, period.

Things a regular computer can compute before the sun goes supernova

Things a domino computer could compute before the sun goes supernova

A Final Problem: *Does it Halt?*

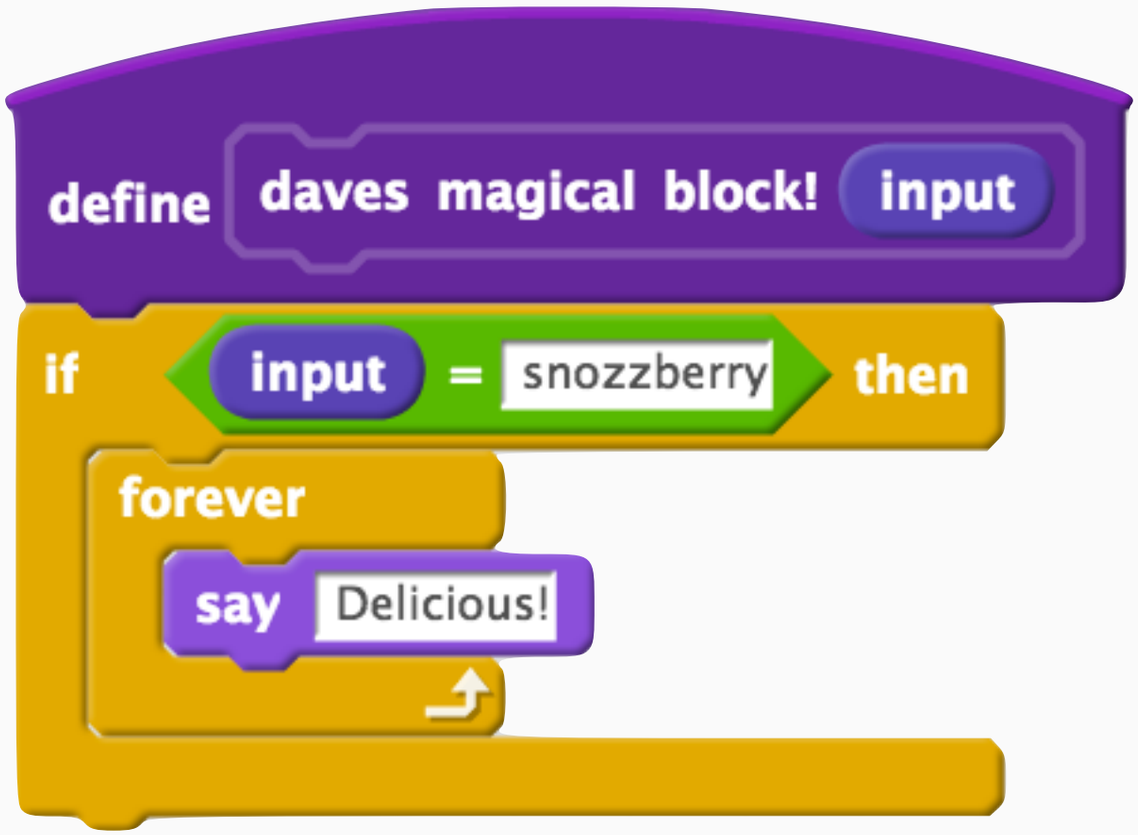
- *INPUT:*
  - A description of an algorithm, input to the algorithm.
- *OUTPUT:*
  - True if the algorithm will halt when run on that input.
  - False if the algorithm does not halt with that input.

Example: *Does it Halt?*

Example: *Does it Halt?*

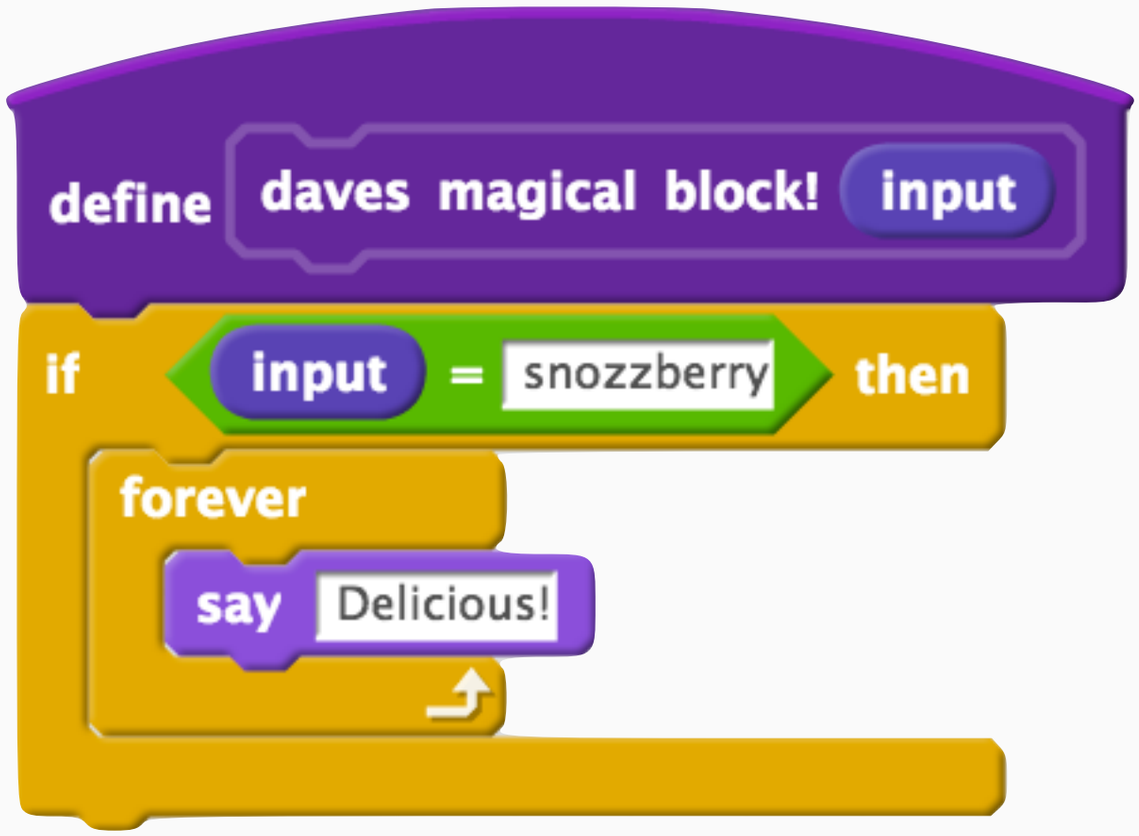
A: No!

Example: *Does it Halt?*



Example: *Does it Halt?*

A: No!



Algorithm: Does *it Halt?*

Loop Checker!

1. Does the given algorithm use any infinite loops?

2. If so, report False.

3. If not, report True.

Algorithm: Does *it Halt?*

- Q1: Will it halt?

  A: Yes!

- Q2: Is it correct?

Loop Checker!

1. Does the given algorithm use any infinite loops?

2. If so, report False.

3. If not, report True.

- Q3: What's the growth rate?

Algorithm: Does *it Halt?*

- Q1: Will it halt?

  A: Yes!

- Q2: Is it correct?

  A: No!

- Q3: What's the growth rate?

Loop Checker!

1. Does the given algorithm use any infinite loops?
2. If so, report False.
3. If not, report True.

Algorithm: Does *it Halt?*

- Q1: Will it halt?

  A: Yes!

- Q2: Is it correct?

  A: No!

- Q3: What's the growth rate?

  A: *N*, have to check each step of the given algorithm.

Loop Checker!

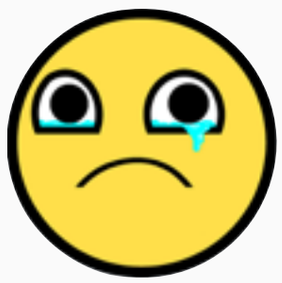
1. Does the given algorithm use any infinite loops?
2. If so, report False.
3. If not, report True.

Algorithm: Does *it Halt?*

- Q1: Will it halt?

  A: Yes!

- Q2: Is it correct?

  A: No! 

- Q3: What's the growth rate?

  A: *N*, have to check each step of the given algorithm.

Loop Checker!

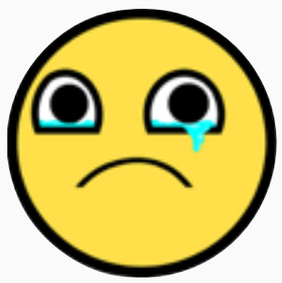
1. Does the given algorithm use any infinite loops?
2. If so, report False.
3. If not, report True.

Q: Can We Do Better?

- Q1: Will it halt?

  A: Yes!

- Q2: Is it correct?

  A: No! 

- Q3: What's the growth rate?

  A: *N*, have to check each step of the given algorithm.

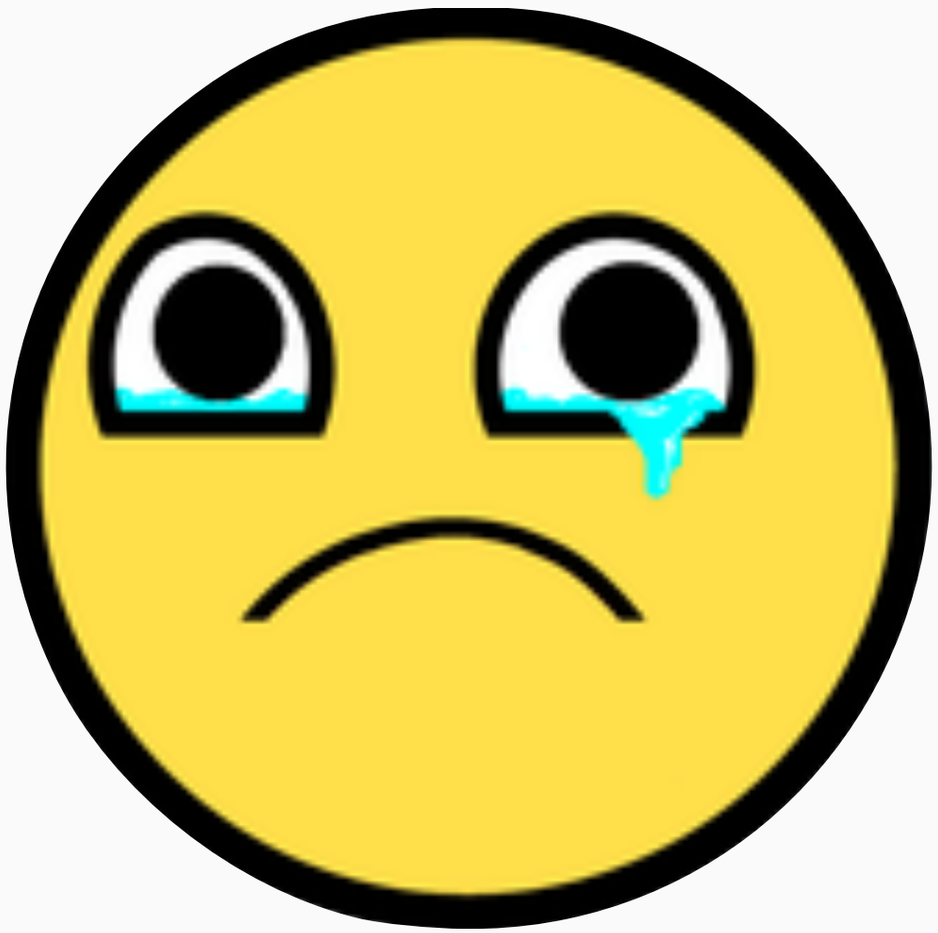
Loop Checker!

1. Does the given algorithm use any infinite loops?

2. If so, report False.

3. If not, report True.

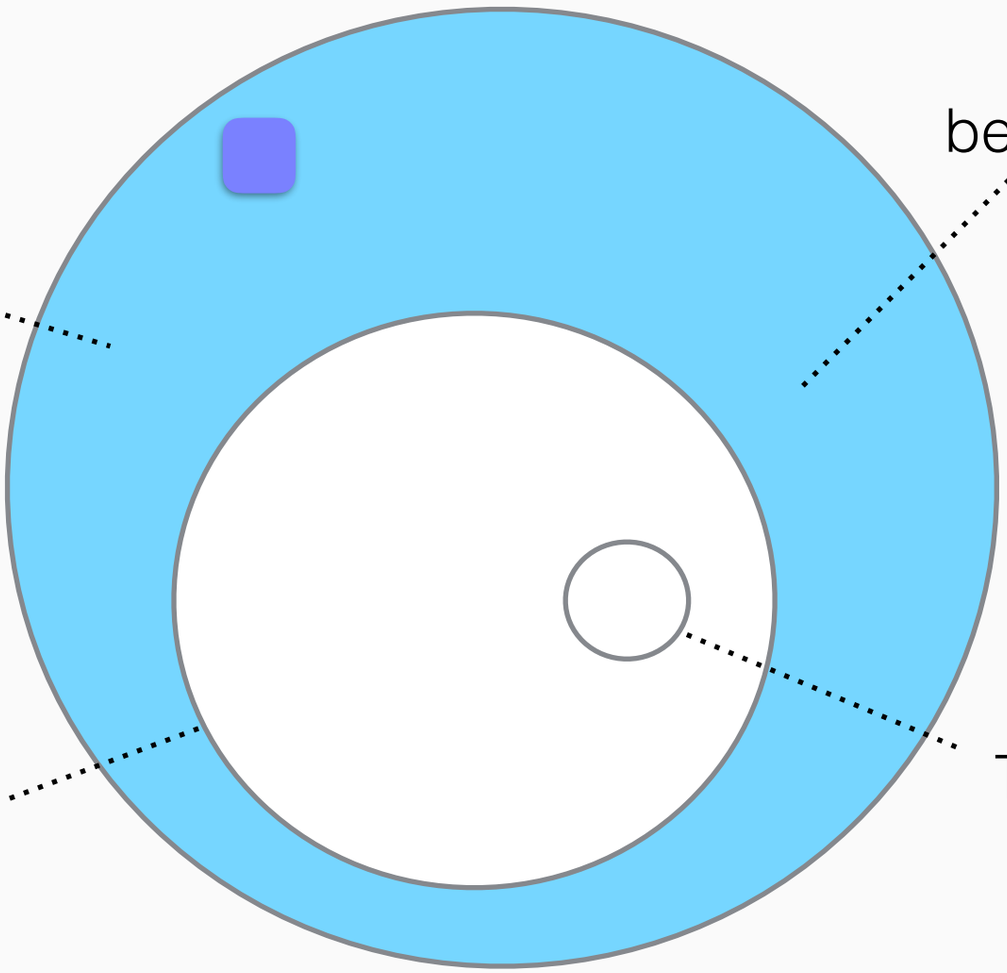
Q: Can We Do Better?



Q: We Can’t!

Computability

Q: What, if anything, is out here?



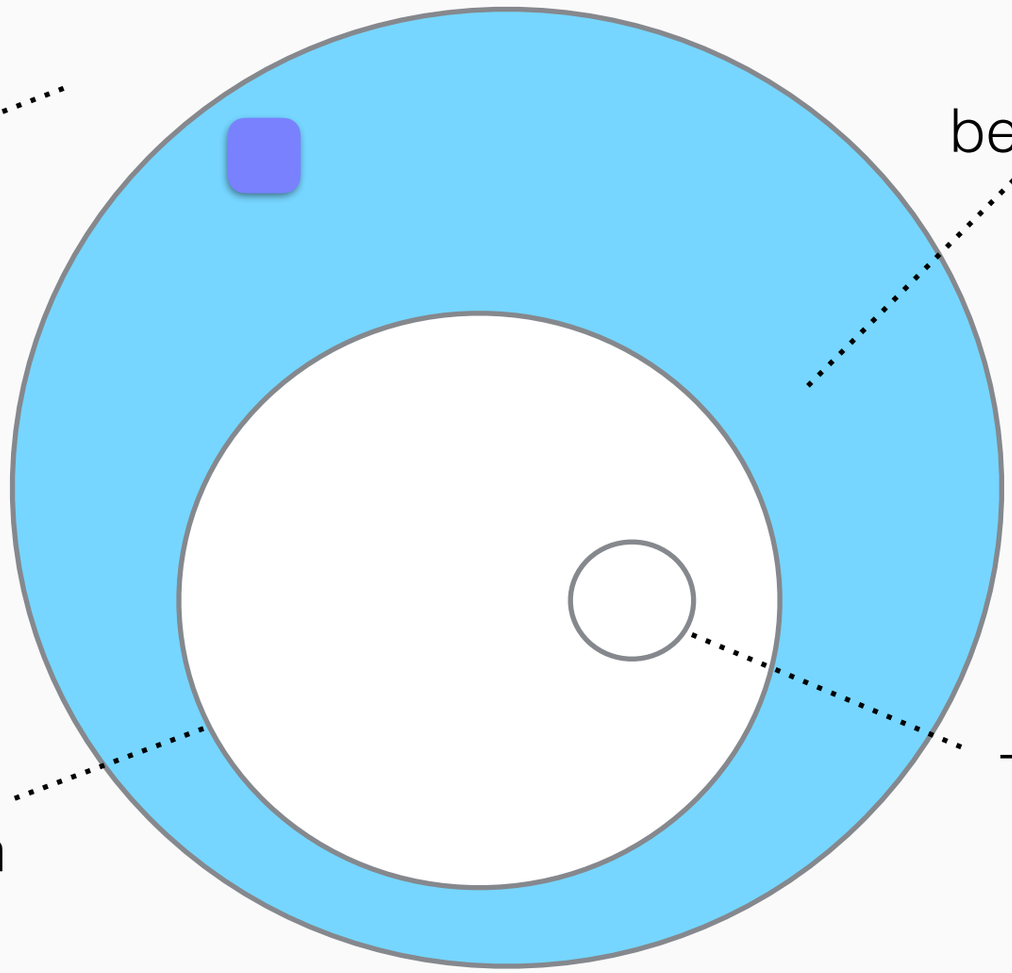
Things that can be computed, period.

Things a regular computer can compute before the sun goes supernova

Things a domino computer could compute before the sun goes supernova

Computability

Q: What, if anything, is out here?



Things that can be computed, period.

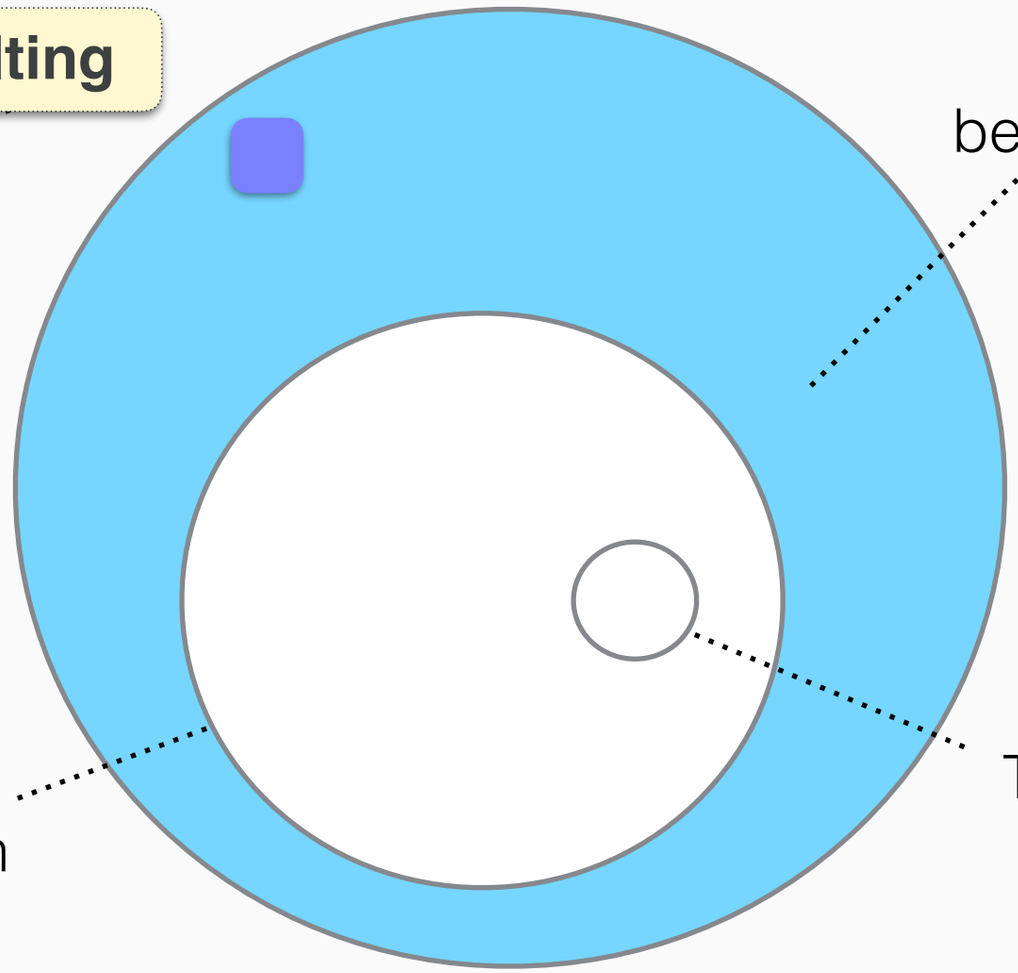
Things a regular computer can compute before the sun goes supernova

Things a domino computer could compute before the sun goes supernova

Computability

Q: What, if anything, is out here?

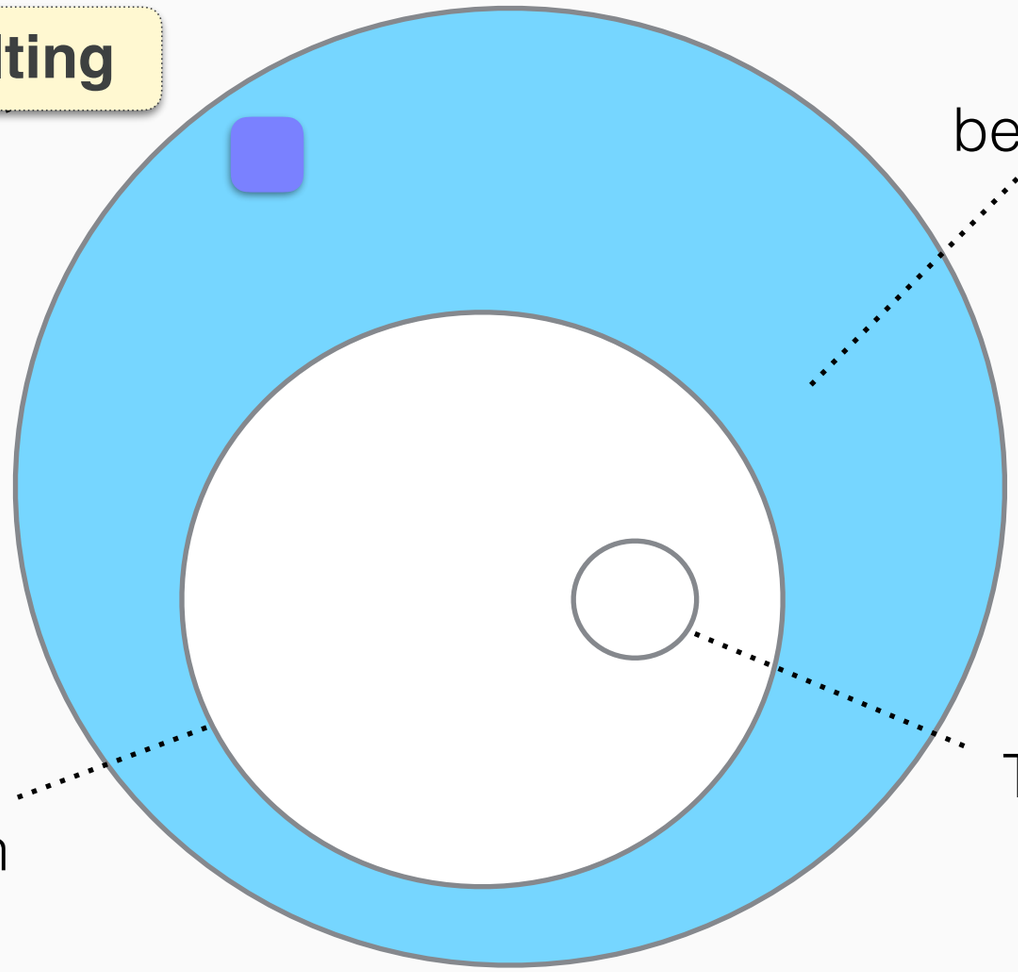
A: The Halting Problem



Computability

Q: What, if anything, is out here?

A: The Halting Problem



For more, see Unit 7!

Unit 3 Reflection

- An algorithm is a recipe for solving a problem.
- An algorithm has three properties of interest:
  - Does it halt?
  - Does it correctly solve the problem?
  - What is its growth rate? (How efficient is it?)
- Four Problems:
  - Search (algorithms = Random Search, Linear Search, Binary Search)
  - Sort (algorithms = Random Sort, Selection Sort)
  - Satisfiability (algorithms = Random Checker, Build The Truth Table)
  - Will it Halt? (algorithms = Loop Checker)