Discussion Section 7

- Viterbi algorithm review
  - Homework 6 questions

- GENSCAN model

- Amortized analysis (for asymptotics)

Viterbi training

Viterbi training

- First step:
  - Find the most probable sequence of states

Viterbi training

- First step:
  - Find the most probable sequence of states

Sequence	A	T	T	C	A	G	C	A

Viterbi training

- First step:
  - Find the most probable sequence of states

Sequence	A	T	T	C	A	G	C	A
AT-rich								

Viterbi training

- First step:
  - Find the most probable sequence of states

Sequence	A	T	T	C	A	G	C	A
AT-rich	●	●	●	●	●	●	●	●

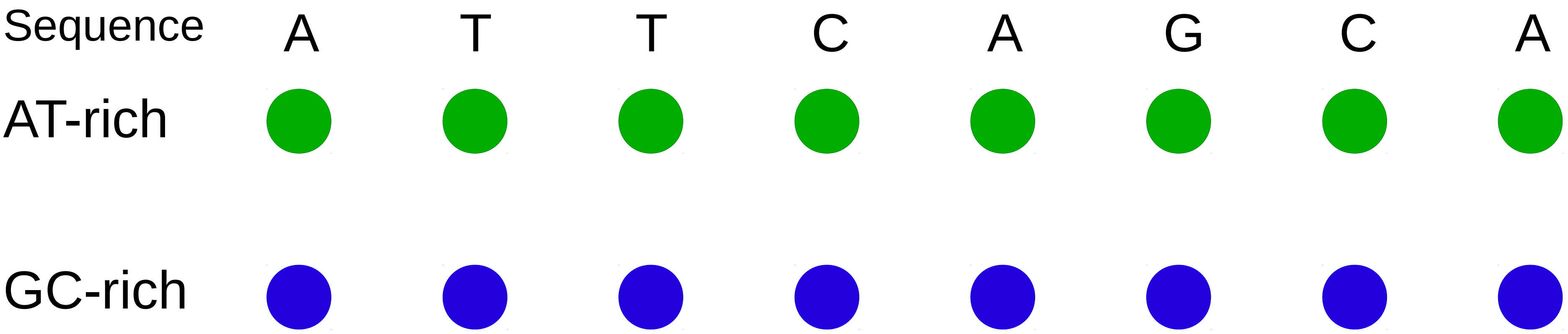
Viterbi training

- First step:
  - Find the most probable sequence of states

Sequence	A	T	T	C	A	G	C	A
AT-rich	●	●	●	●	●	●	●	●
GC-rich								

Viterbi training

- First step:
  - Find the most probable sequence of states



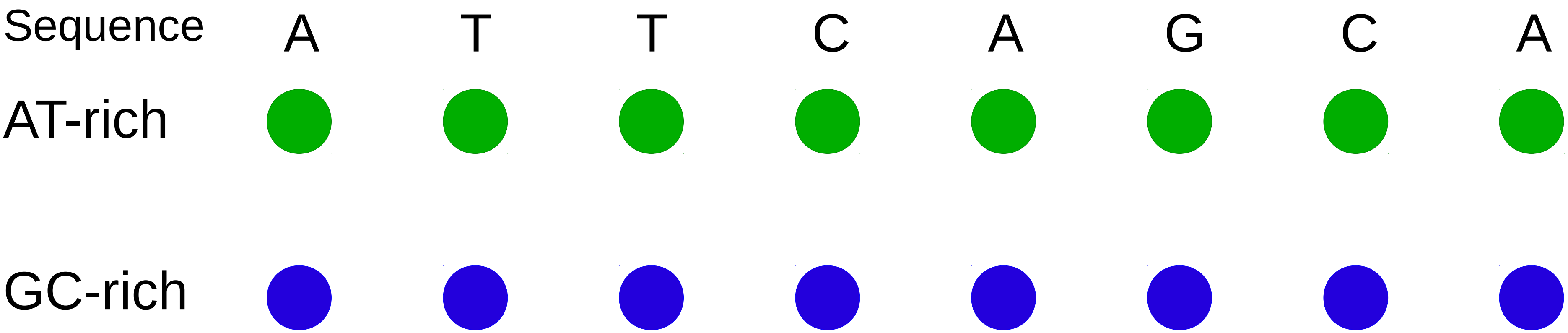
Sequence A T T C A G C A

AT-rich

GC-rich

Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities



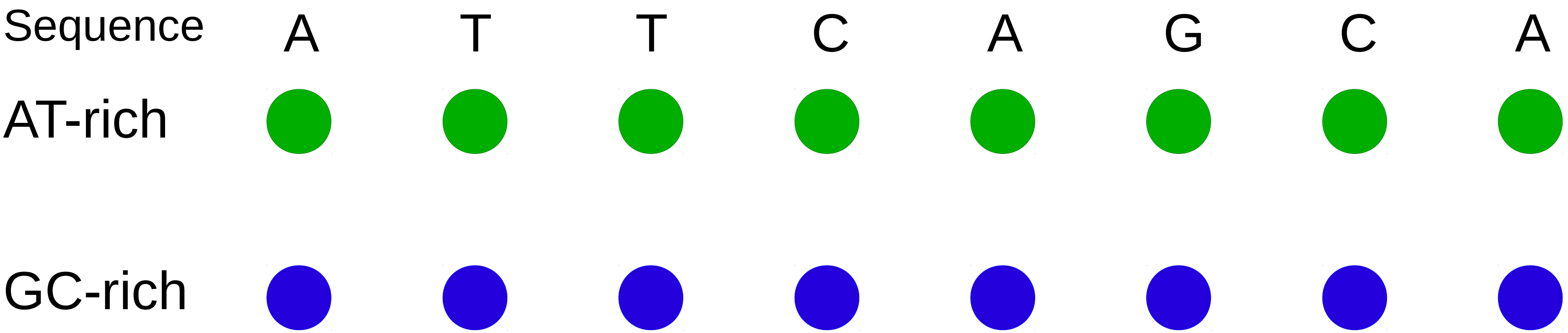
Sequence A T T C A G C A

AT-rich

GC-rich

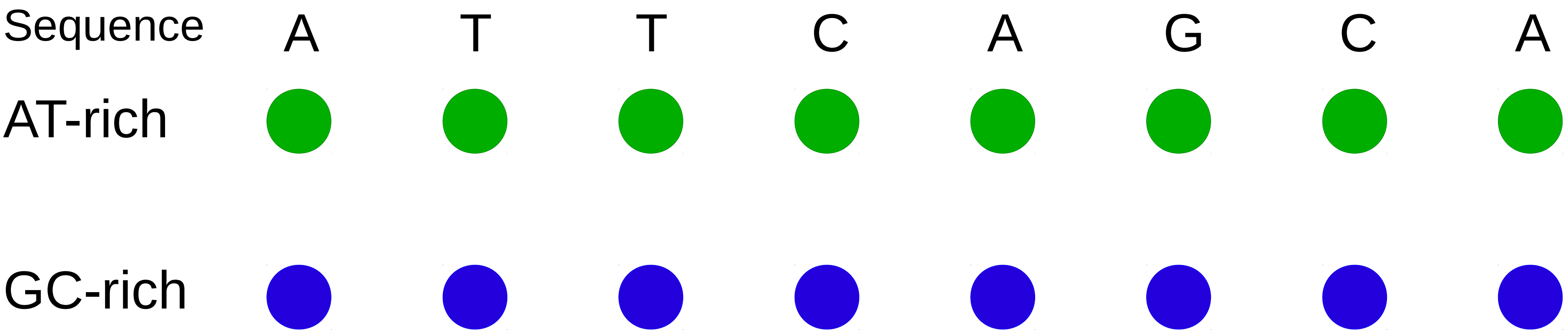
Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation



Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition



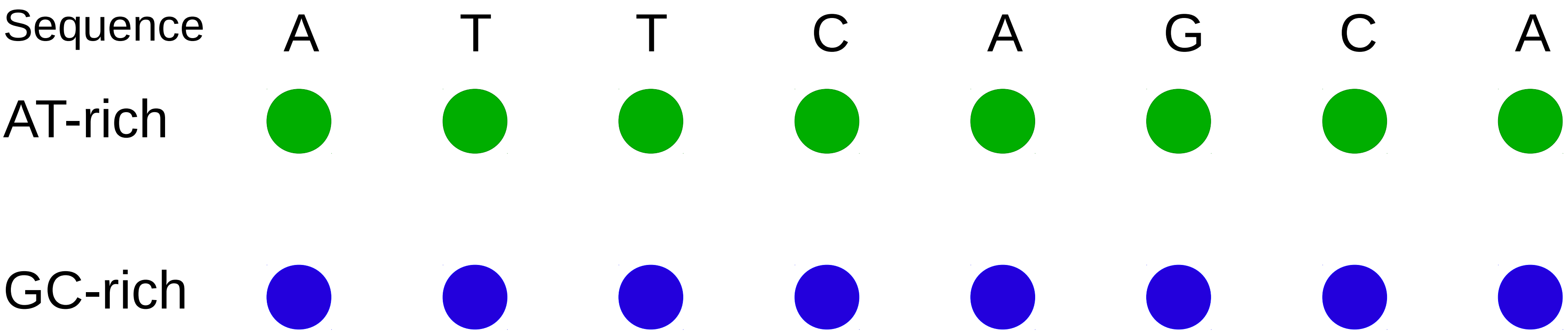
Sequence A T T C A G C A

AT-rich

GC-rich

Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission



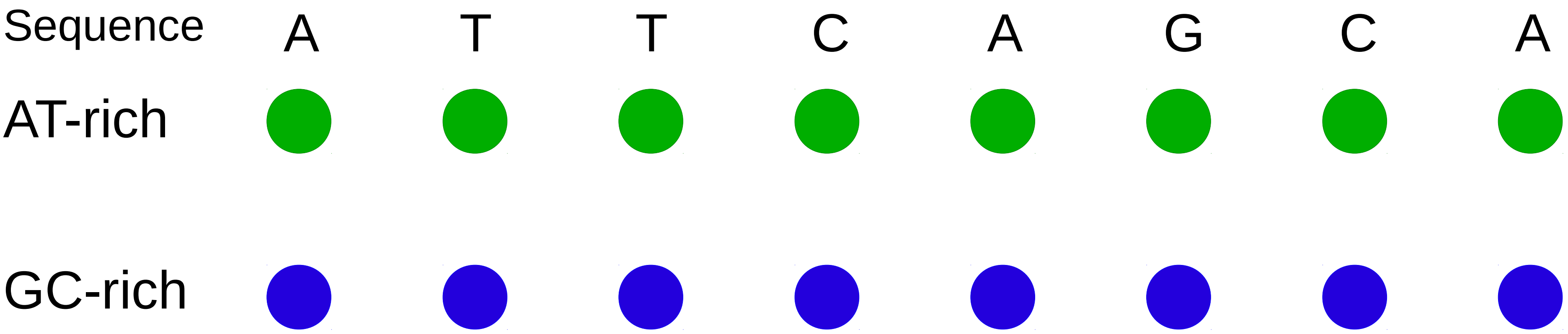
Sequence A T T C A G C A

AT-rich

GC-rich

Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?



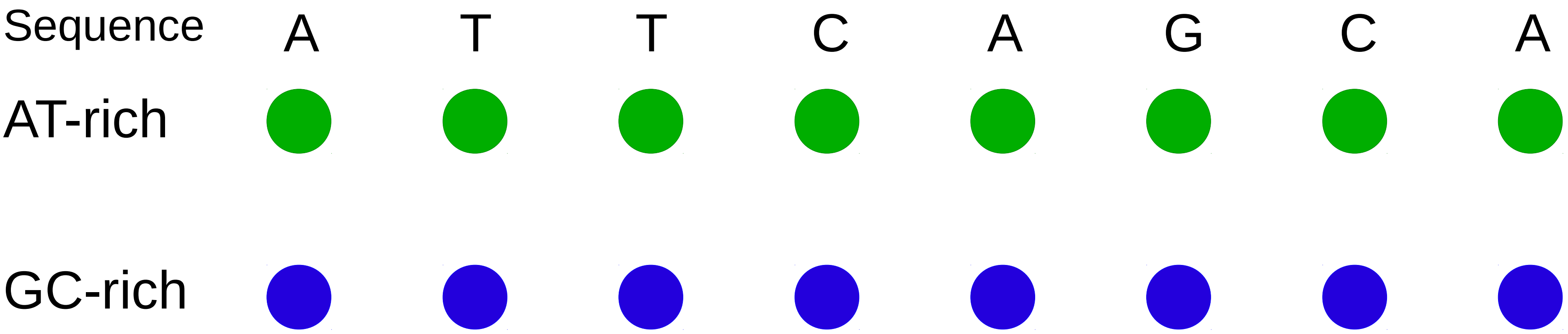
Sequence A T T C A G C A

AT-rich

GC-rich

Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?
      - Correct emissions



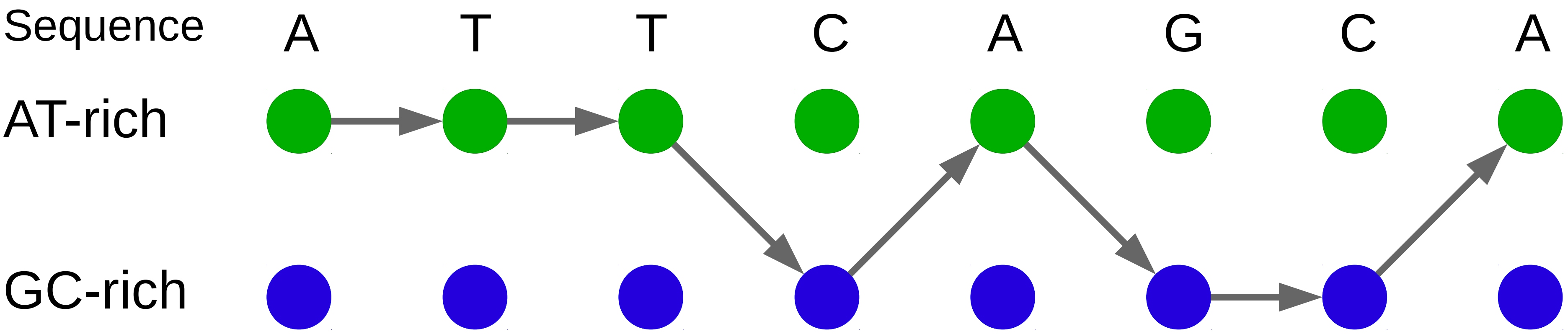
Sequence A T T C A G C A

AT-rich

GC-rich

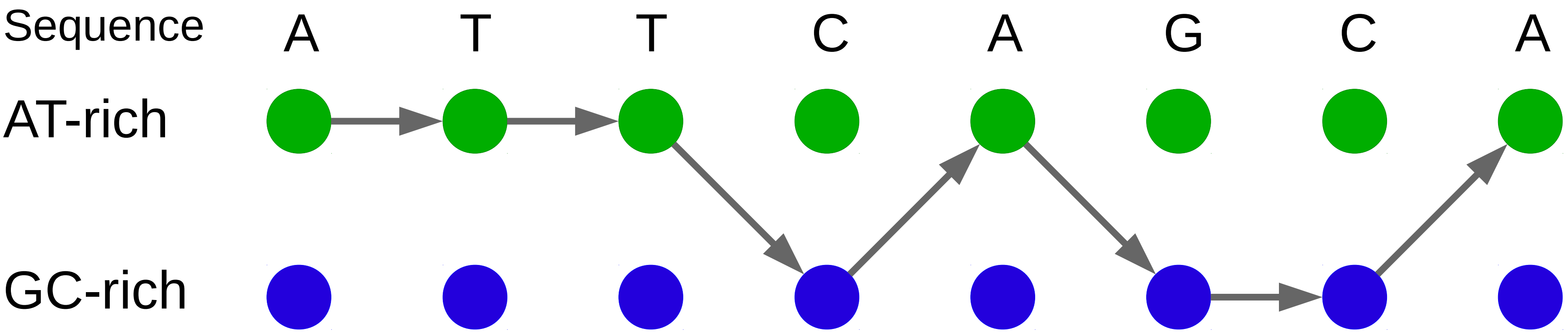
Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?
      - Correct emissions



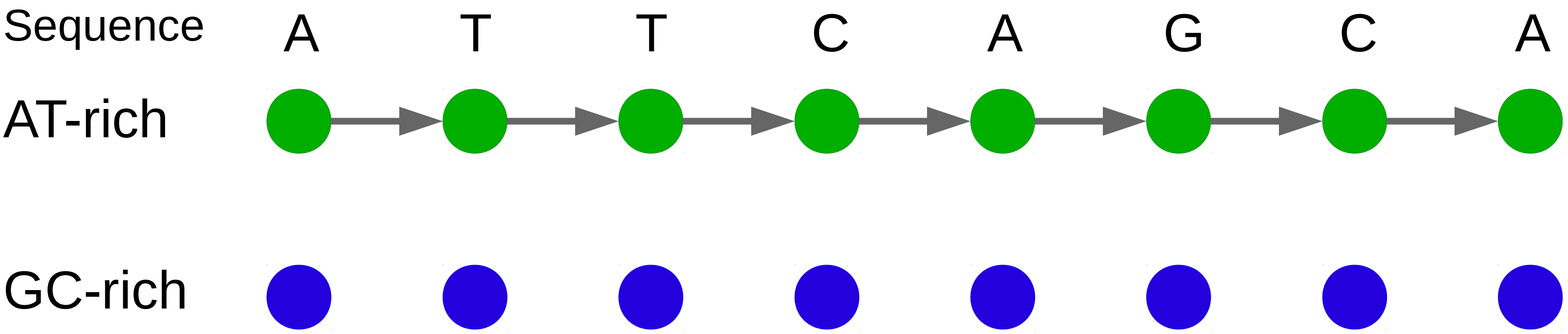
Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?
      - Correct emissions
      - Remaining in the same state



Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?
      - Correct emissions
      - Remaining in the same state



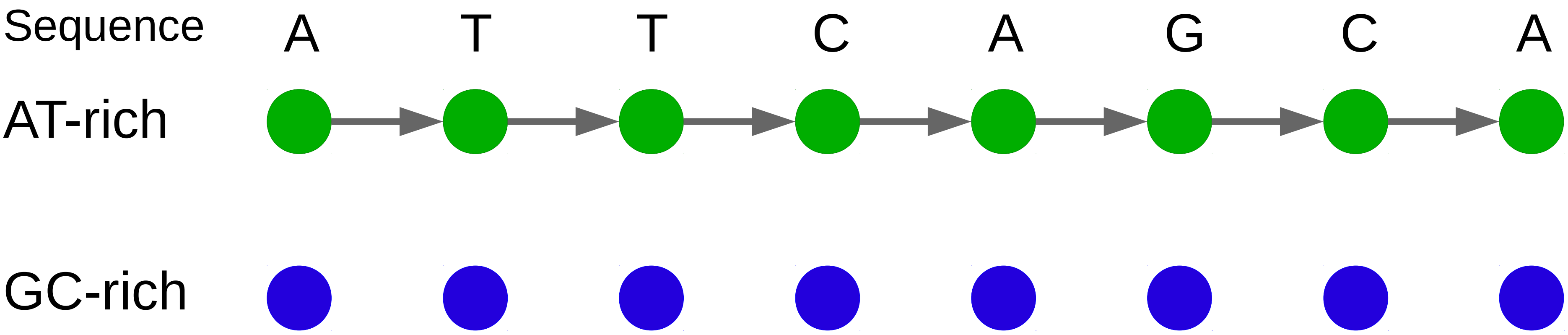
Sequence A T T C A G C A

AT-rich

GC-rich

Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?
      - Correct emissions
      - Remaining in the same state
      - Balance between the two



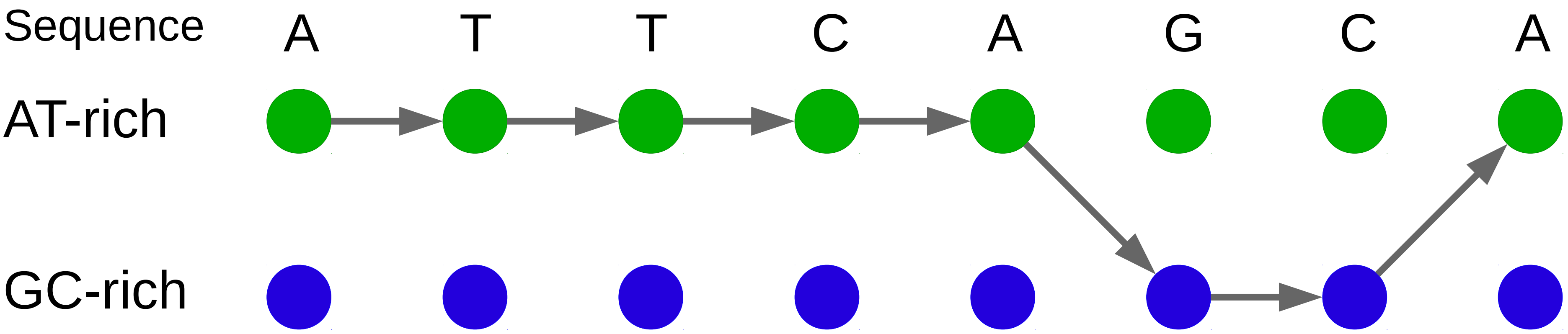
Sequence A T T C A G C A

AT-rich

GC-rich

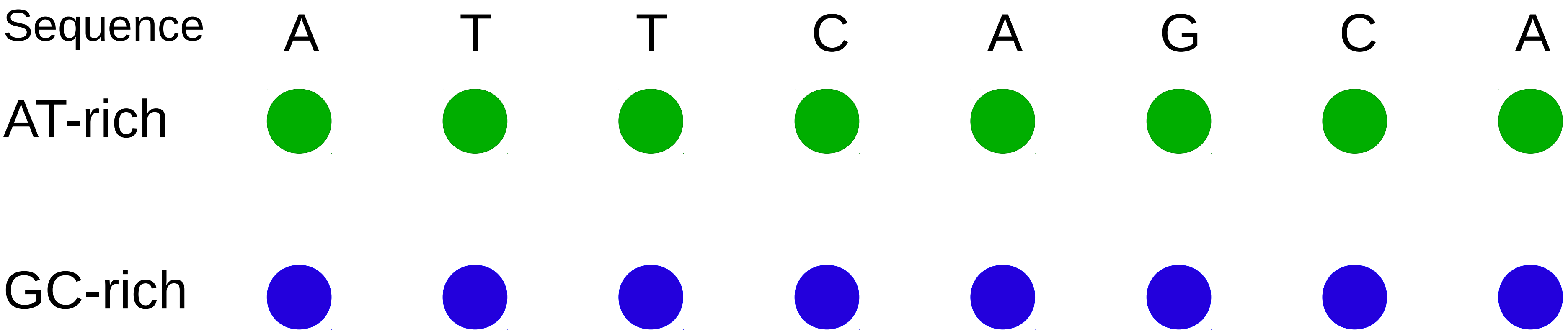
Viterbi training

- First step:
  - Find the most probable sequence of states
    - This depends on the specified probabilities
      - Initiation
      - Transition
      - Emission
    - What is prioritized?
      - Correct emissions
      - Remaining in the same state
      - Balance between the two



Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?



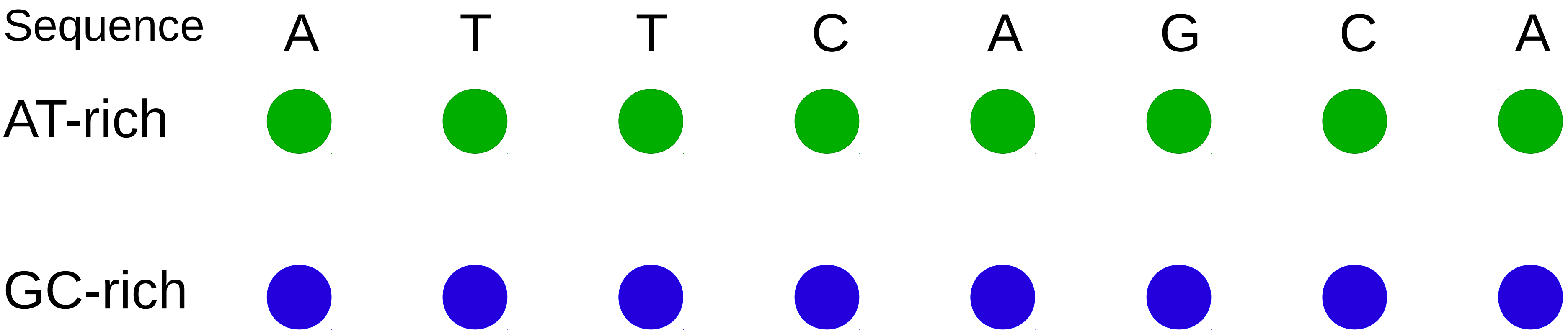
Sequence A T T C A G C A

AT-rich

GC-rich

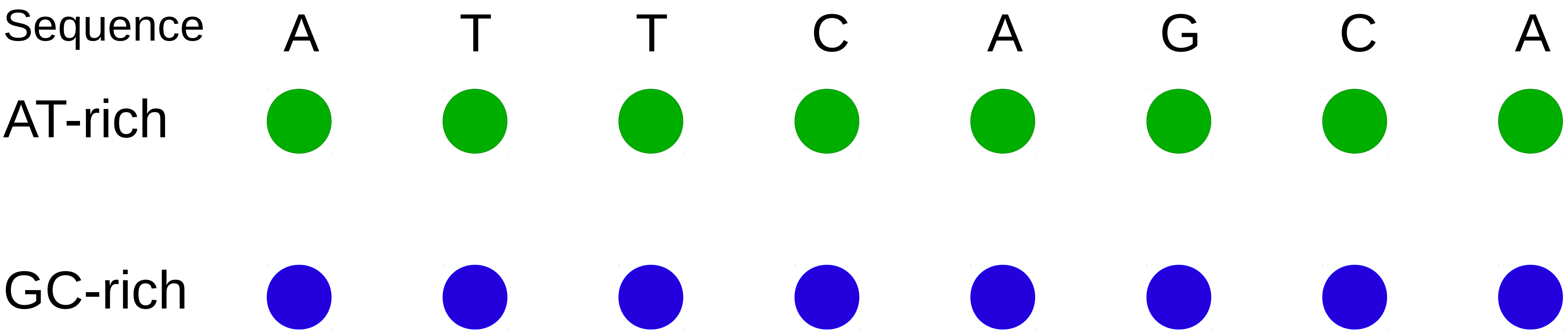
Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?



Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?



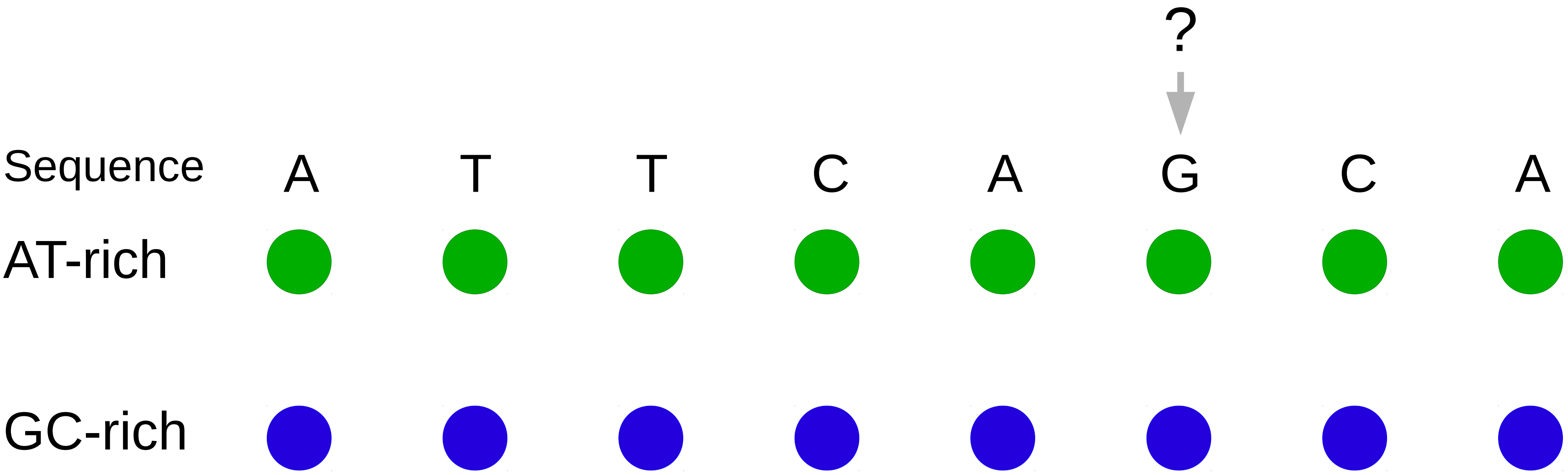
Sequence A T T C A G C A

AT-rich

GC-rich

Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?

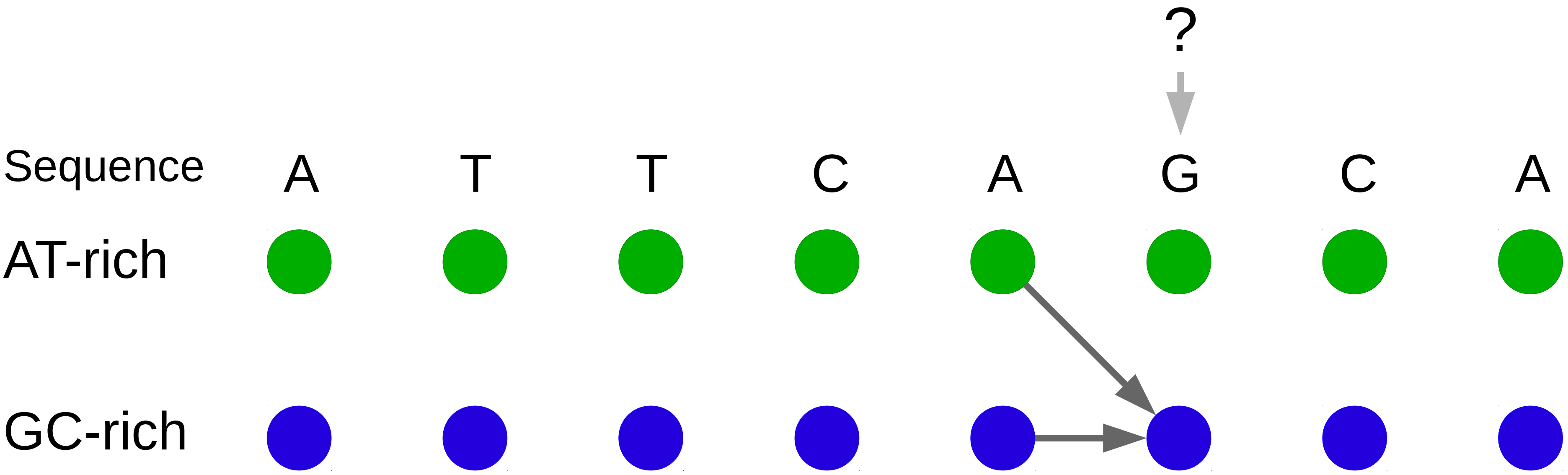


?

Sequence	A	T	T	C	A	G	C	A
AT-rich	●	●	●	●	●	●	●	●
GC-rich	●	●	●	●	●	●	●	●

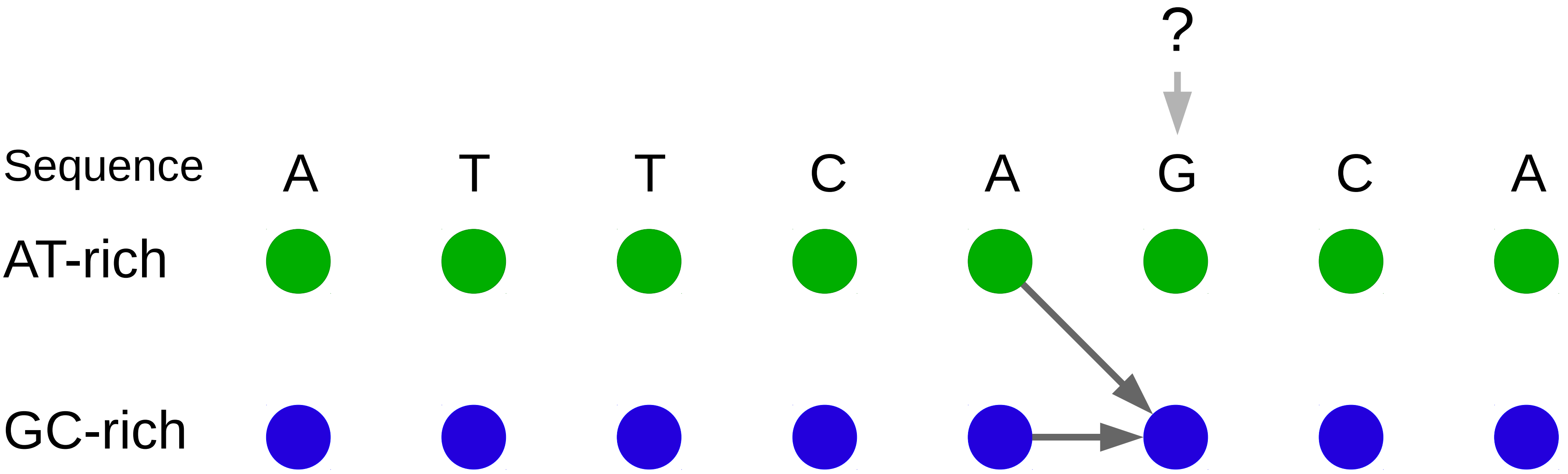
Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?



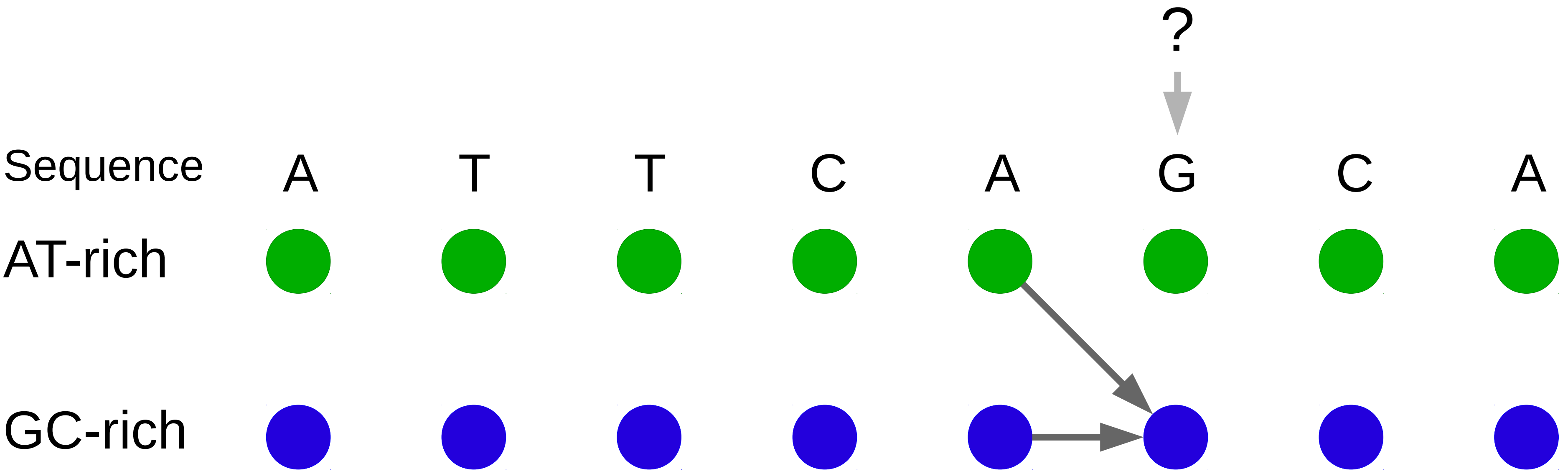
Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?
      - Most probable path before, accounting for preceding transition



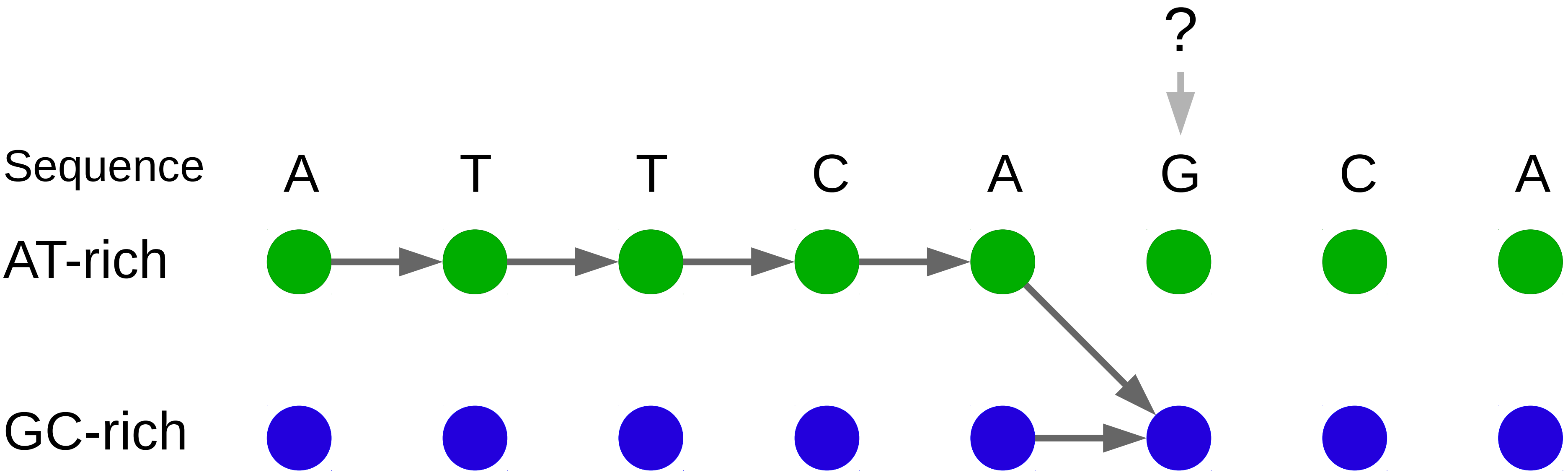
Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?
      - Most probable path before, accounting for preceding transition
      - Trace back the most probable path at the end



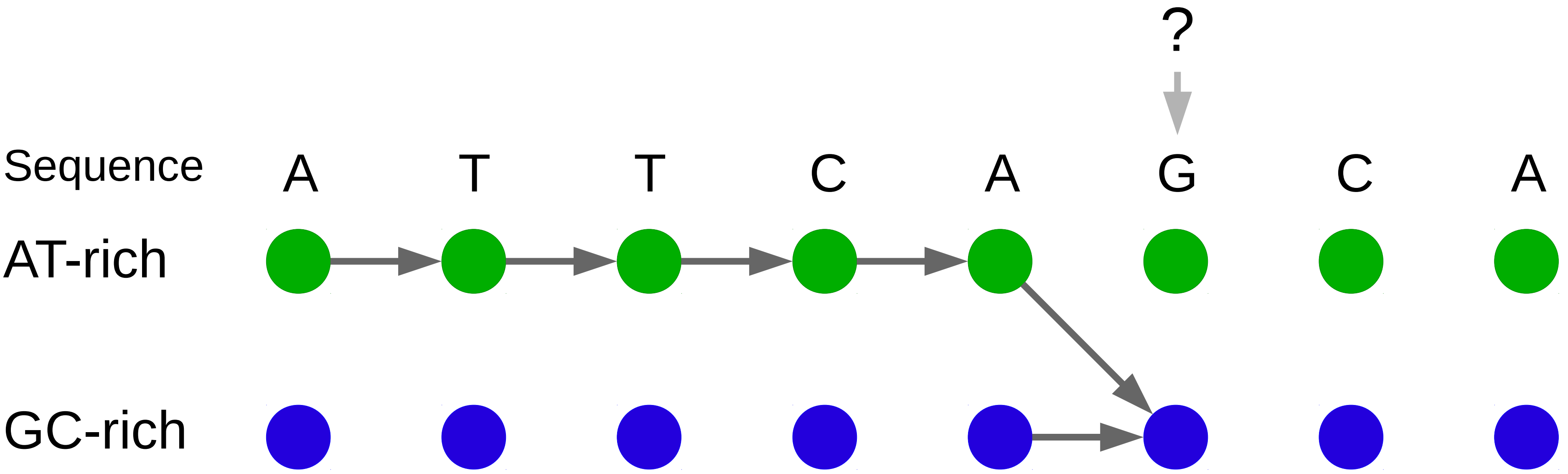
Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?
      - Most probable path before, accounting for preceding transition
      - Trace back the most probable path at the end



Viterbi training

- First step:
  - Find the most probably sequence of states
    - How do we actually determine the path?
    - Can we avoid testing every path?
      - What's the most probable path that uses state *j* at sequence position *i*?
      - Most probable path before, accounting for preceding transition
      - Trace back the most probable path at the end
      - Aren't you glad you've already written this algorithm?



Viterbi training

- Second step:
  - Update probabilities according to current most probable path

Viterbi training

- Second step:
  - Update probabilities according to current most probable path
    - For the homework, you'll only be updating the transition probabilities

Viterbi training

- Second step:
  - Update probabilities according to current most probable path
    - For the homework, you'll only be updating the transition probabilities
      - Calculate how often transitions happen between all pairs of states

Viterbi training

- Second step:
  - Update probabilities according to current most probable path
    - For the homework, you'll only be updating the transition probabilities
      - Calculate how often transitions happen between all pairs of states
      - Set the transition probabilities to those frequencies for the next step

Viterbi training

- Second step:
  - Update probabilities according to current most probable path
    - For the homework, you'll only be updating the transition probabilities
      - Calculate how often transitions happen between all pairs of states
      - Set the transition probabilities to those frequencies for the next step
      - Do this 10 times

Homework 5 tips

Homework 5 tips

- Calculate the first few steps by hand and check that your program matches

Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions

Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions
- Try a few different small test cases

Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions
- Try a few different small test cases
- Avoid underflow

Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions
- Try a few different small test cases
- Avoid underflow
  - Take the logarithm

Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions
- Try a few different small test cases
- Avoid underflow
  - Take the logarithm
  - Add log-transformed probabilities instead of multiplying

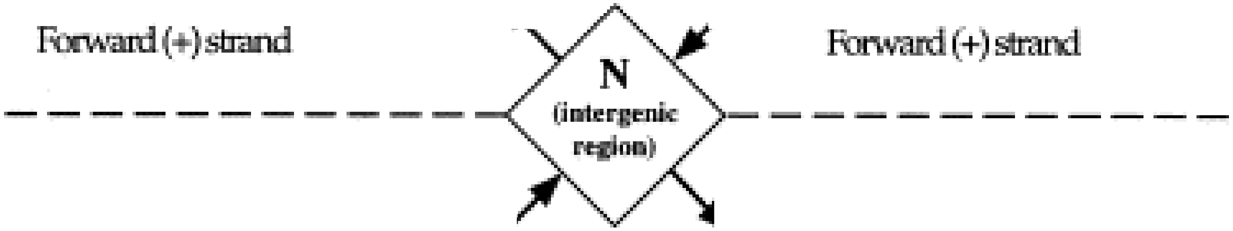
Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions
- Try a few different small test cases
- Avoid underflow
  - Take the logarithm
  - Add log-transformed probabilities instead of multiplying
    - What about log(0)?

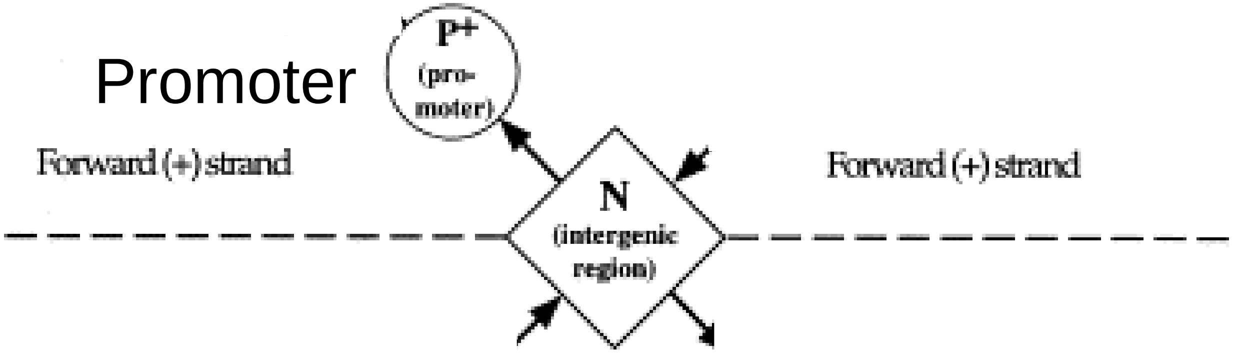
Homework 5 tips

- Calculate the first few steps by hand and check that your program matches
  - We won't be providing example solutions
- Try a few different small test cases
- Avoid underflow
  - Take the logarithm
  - Add log-transformed probabilities instead of multiplying
    - What about log(0)?
  - See Mann and Rabiner (pg 273) for more details (link on website)

GENSCAN model

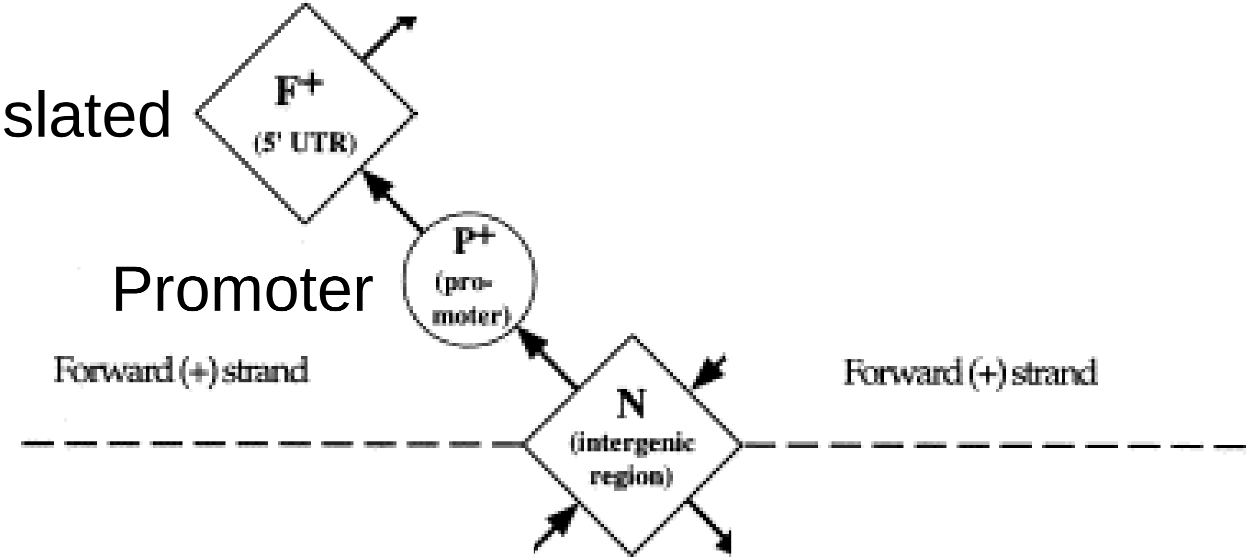


GENSCAN model



Promoter

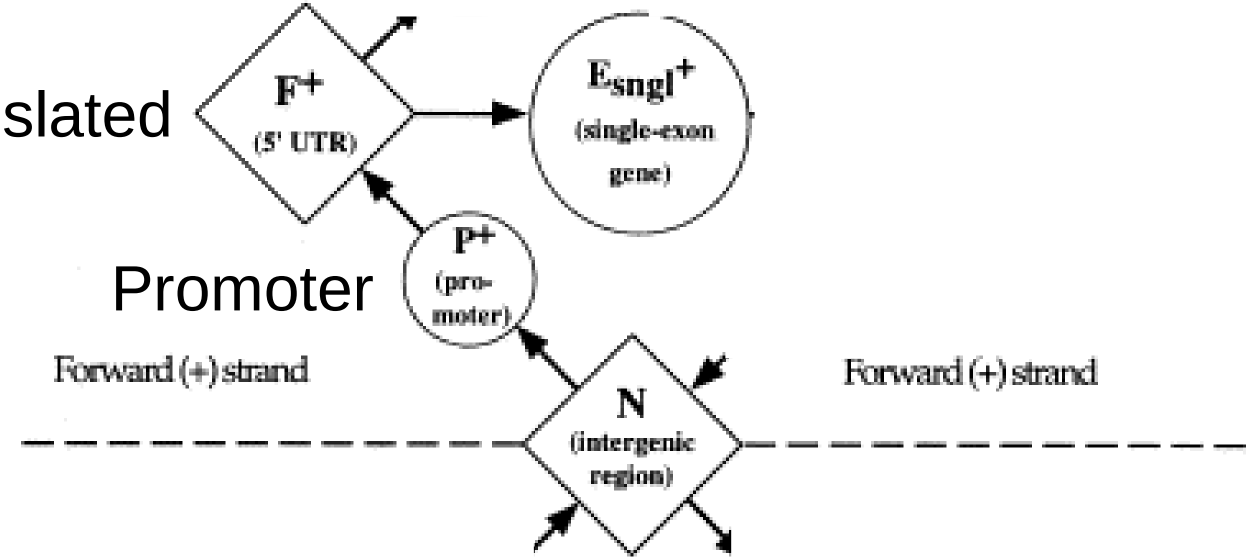
GENSCAN model



5' untranslated

Promoter

GENSCAN model

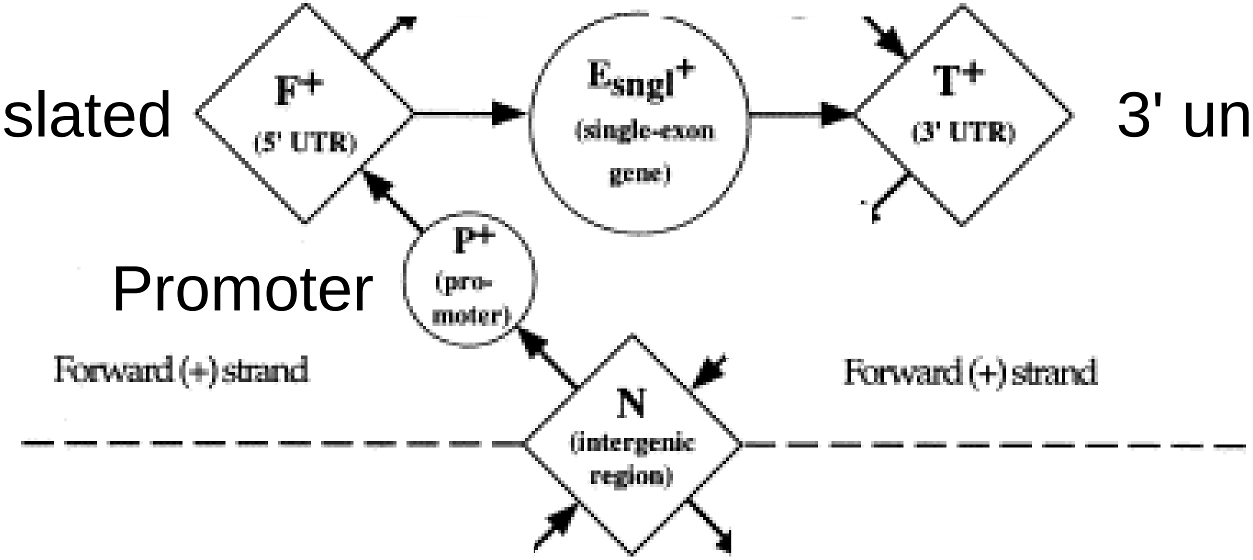


5' untranslated

Promoter

GENSCAN model

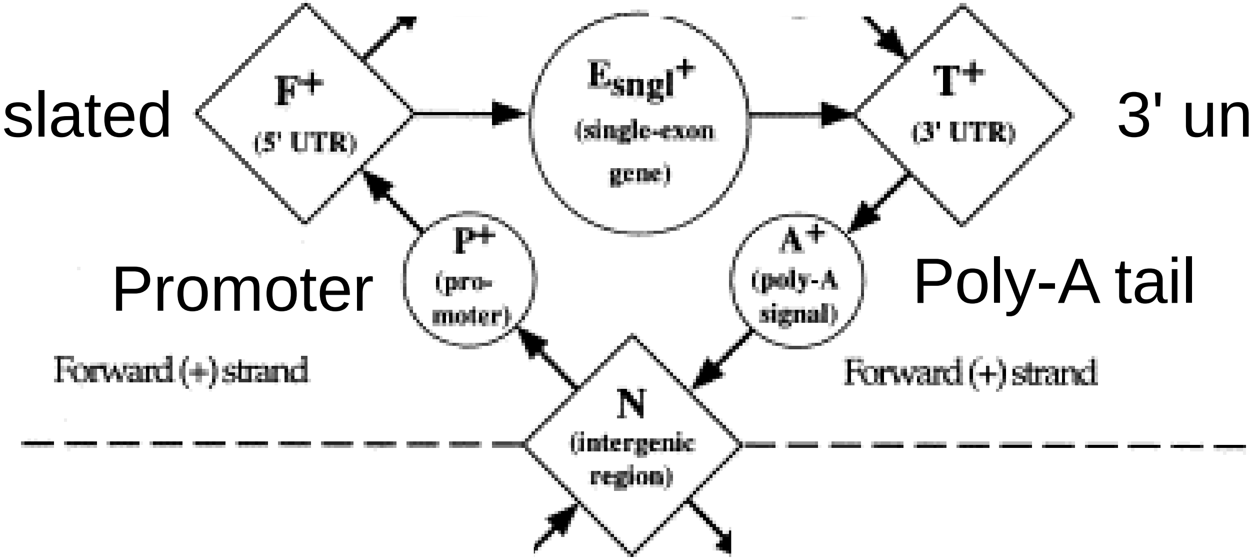
5' untranslated



3' untranslated

Promoter

GENSCAN model



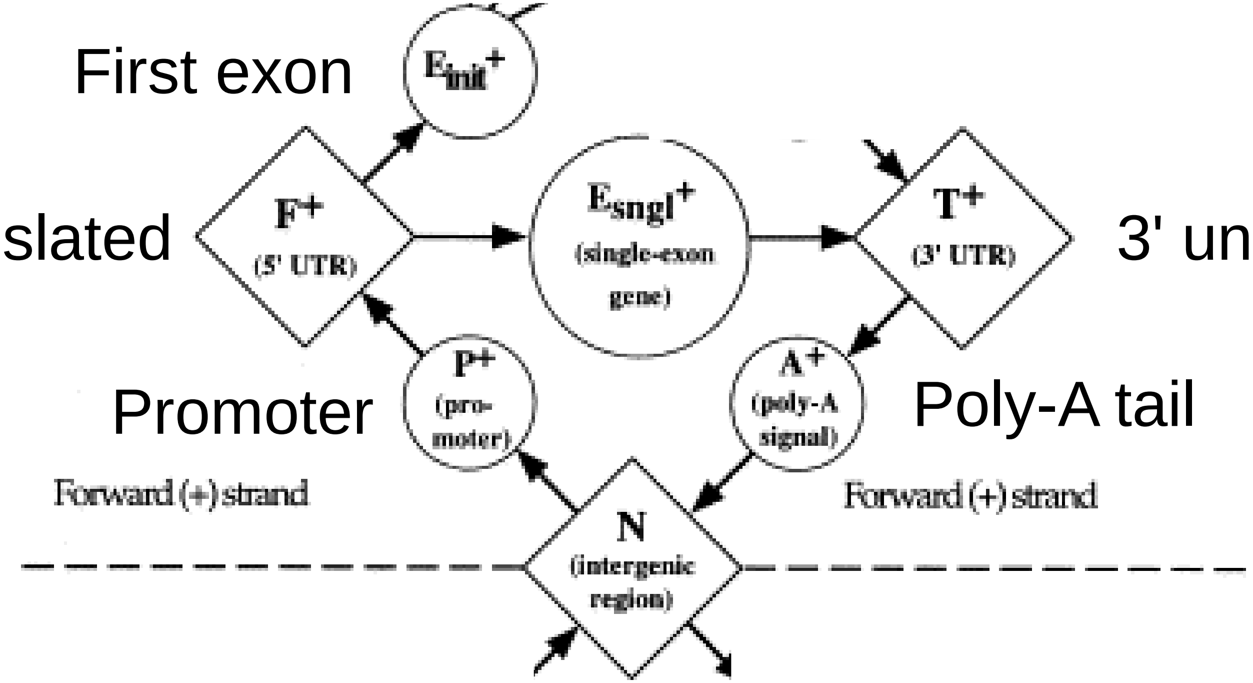
5' untranslated

3' untranslated

Promoter

Poly-A tail

GENSCAN model



First exon

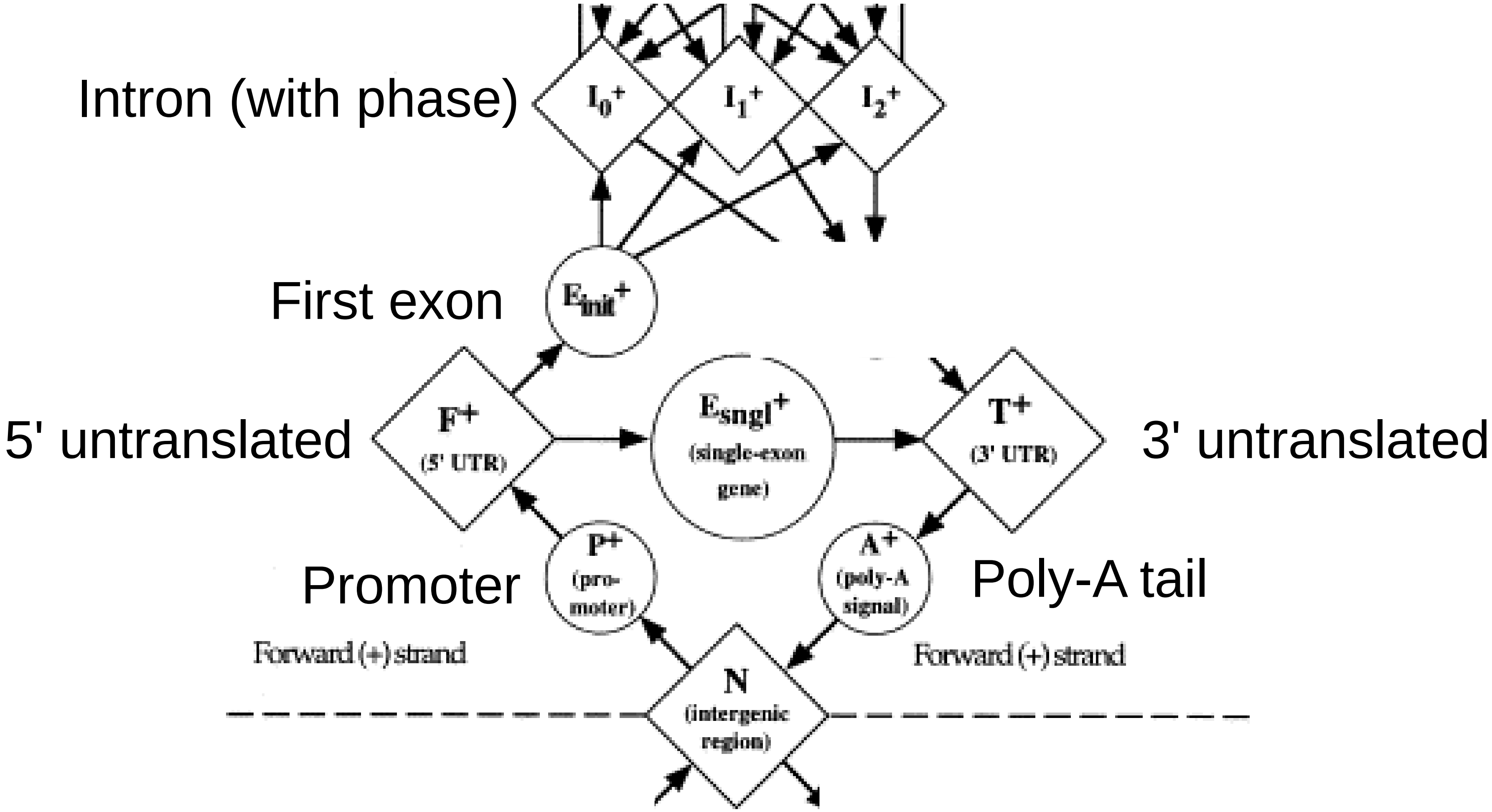
5' untranslated

3' untranslated

Promoter

Poly-A tail

GENSCAN model



Intron (with phase)

First exon

5' untranslated

3' untranslated

Promoter

Poly-A tail

GENSCAN model

Intron (with phase)

Intron (with phase)

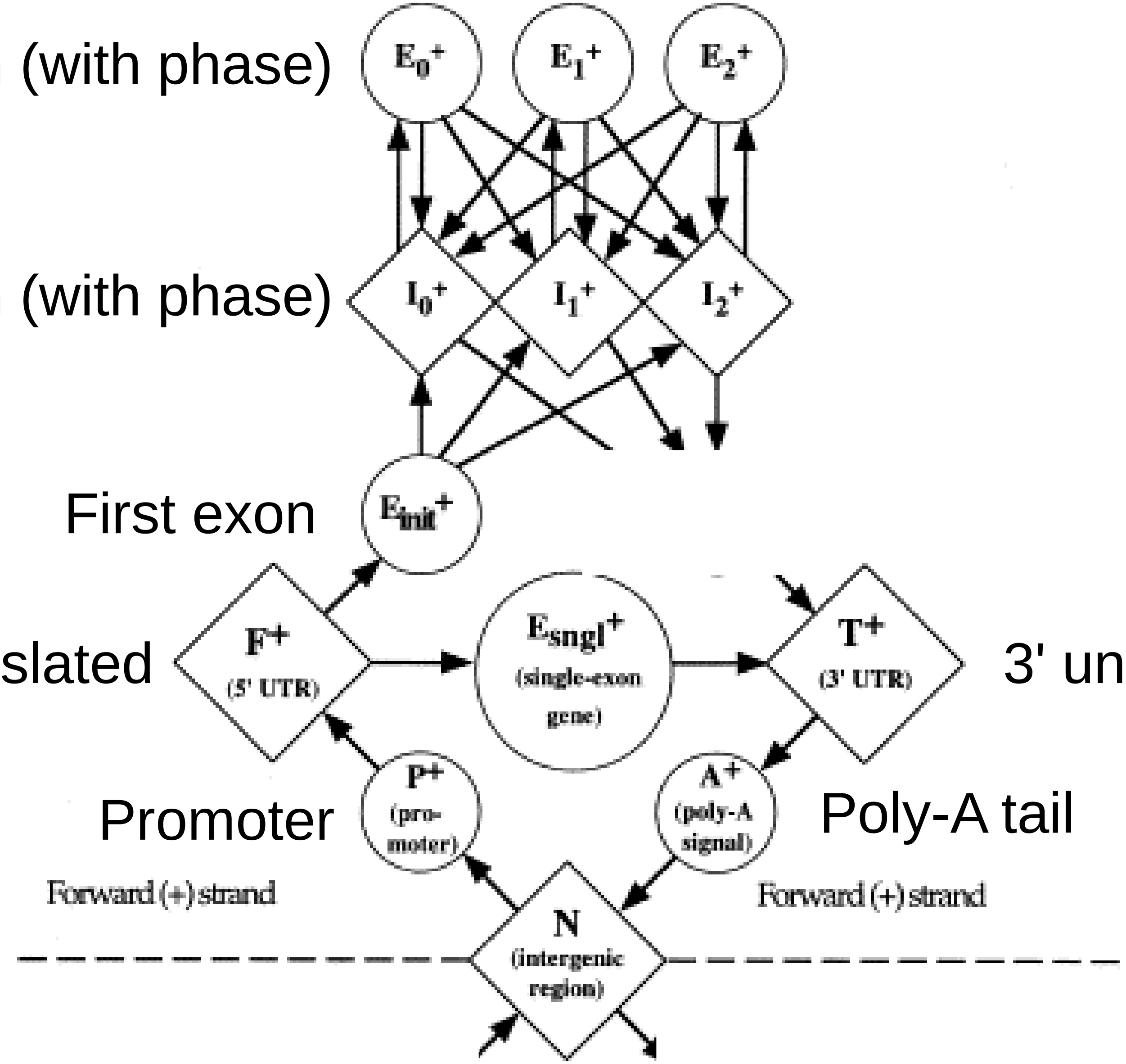
First exon

5' untranslated

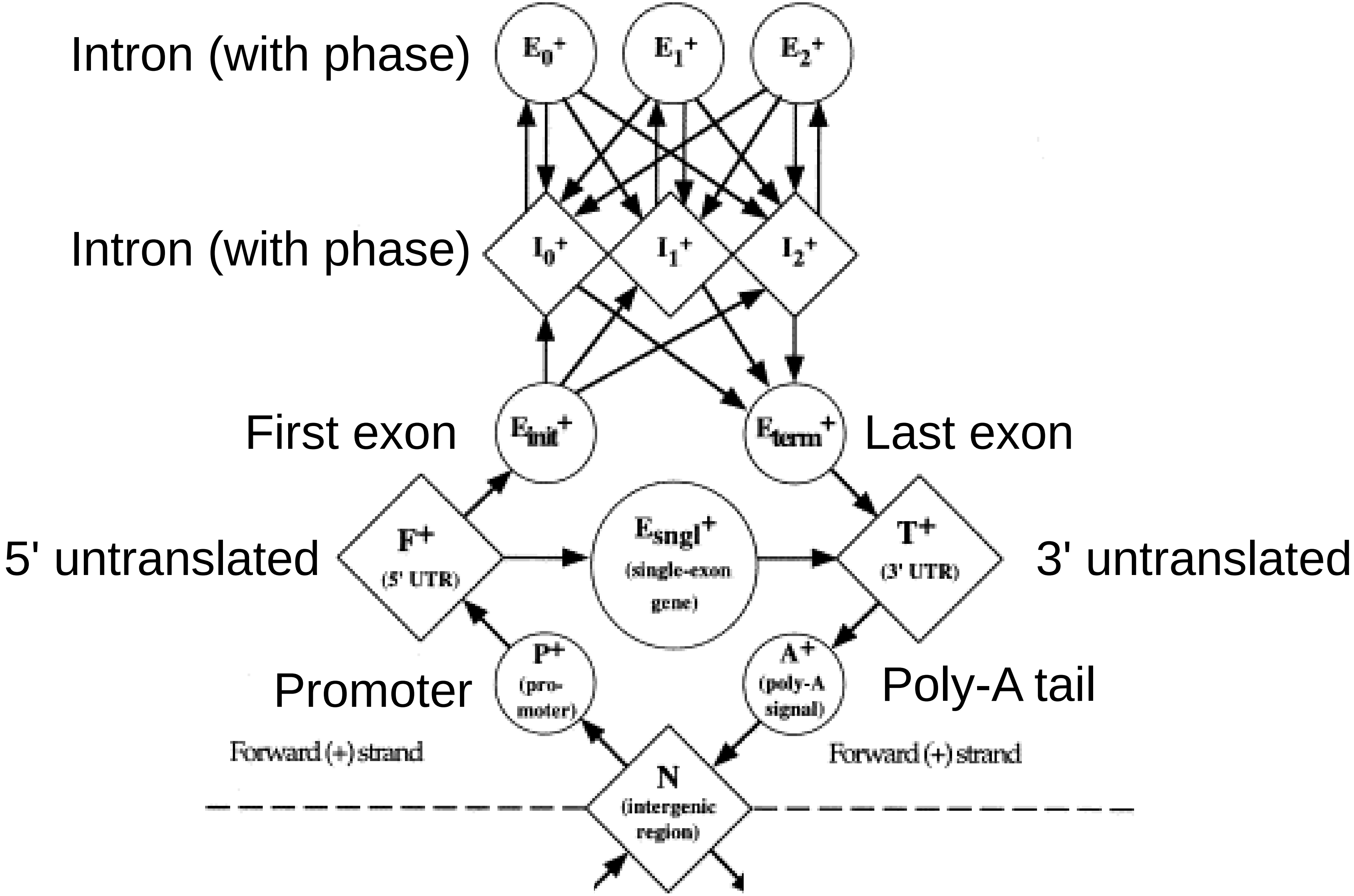
3' untranslated

Promoter

Poly-A tail



GENSCAN model



Intron (with phase)

Intron (with phase)

First exon

Last exon

5' untranslated

3' untranslated

Promoter

Poly-A tail

GENSCAN details

GENSCAN details

- 4 different sets of initiation/transition probabilities for overall GC content

GENSCAN details

- 4 different sets of initiation/transition probabilities for overall GC content
  - Gene density higher in GC-rich regions

GENSCAN details

- 4 different sets of initiation/transition probabilities for overall GC content
  - Gene density higher in GC-rich regions
- Length (duration) probabilities of states based on observed sequence data

GENSCAN details

- 4 different sets of initiation/transition probabilities for overall GC content
  - Gene density higher in GC-rich regions
- Length (duration) probabilities of states based on observed sequence data
- Each higher level state (promoter, splice site, etc.) has embedded specialized HMM

GENSCAN details

- 4 different sets of initiation/transition probabilities for overall GC content
  - Gene density higher in GC-rich regions
- Length (duration) probabilities of states based on observed sequence data
- Each higher level state (promoter, splice site, etc.) has embedded specialized HMM
- Exons modeled with 5th-order HMM

GENSCAN details

- 4 different sets of initiation/transition probabilities for overall GC content
  - Gene density higher in GC-rich regions
- Length (duration) probabilities of states based on observed sequence data
- Each higher level state (promoter, splice site, etc.) has embedded specialized HMM
- Exons modeled with 5th-order HMM
  - Corresponds to hexamer probabilities

Amortized analysis

Amortized analysis

- Useful for evaluating the cost of performing multiple steps with variable cost

Amortized analysis

- Useful for evaluating the cost of performing multiple steps with variable cost
  - Example: arrays with dynamic resizing

Amortized analysis

- Useful for evaluating the cost of performing multiple steps with variable cost
  - Example: arrays with dynamic resizing
- 3 different classes (all closely related)

Amortized analysis

- Useful for evaluating the cost of performing multiple steps with variable cost
  - Example: arrays with dynamic resizing

- 3 different classes (all closely related)
  - Aggregate analysis

Amortized analysis

- Useful for evaluating the cost of performing multiple steps with variable cost
  - Example: arrays with dynamic resizing

- 3 different classes (all closely related)
  - Aggregate analysis
  - Accounting method

Amortized analysis

- Useful for evaluating the cost of performing multiple steps with variable cost
  - Example: arrays with dynamic resizing

- 3 different classes (all closely related)
  - Aggregate analysis
  - Accounting method
  - Potential method

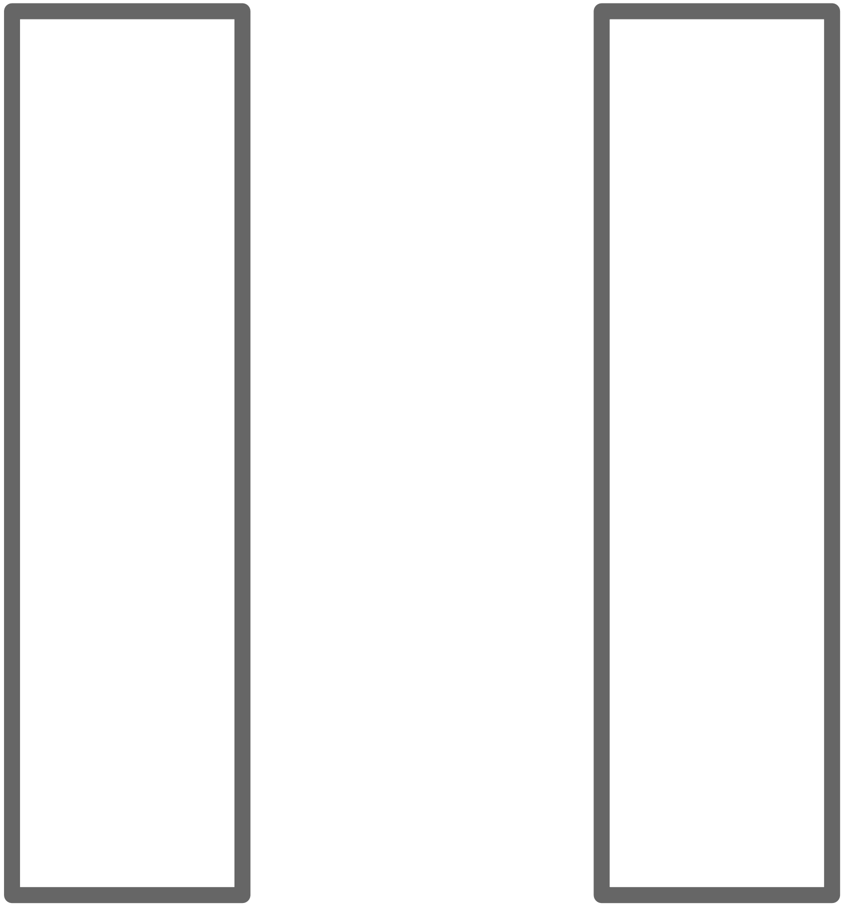
Amortized analysis example: aggregate analysis

Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)

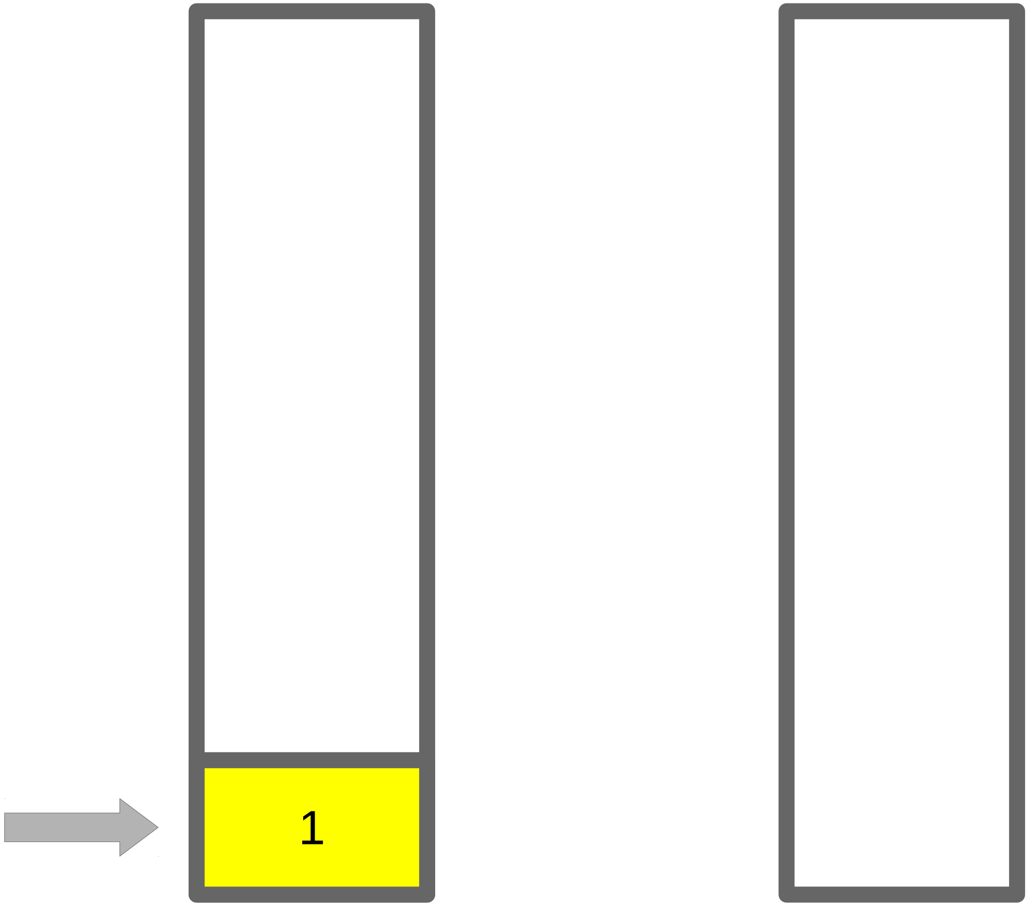
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



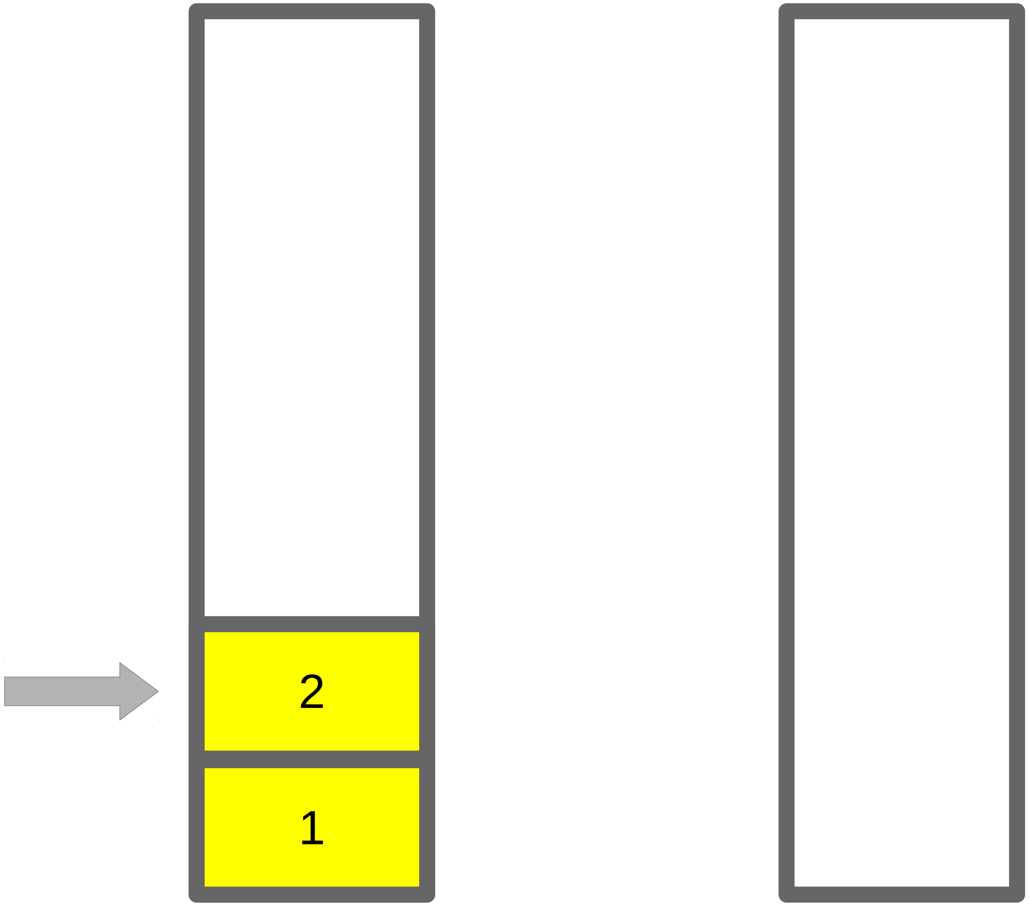
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



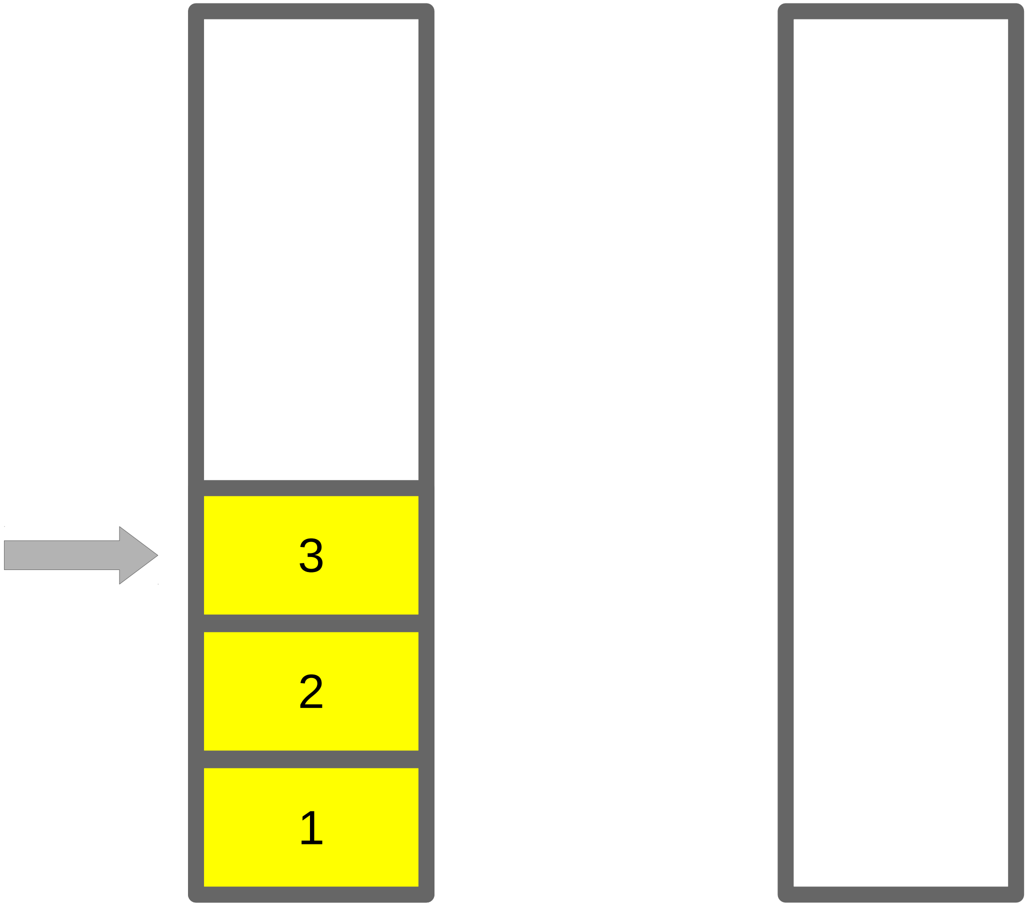
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



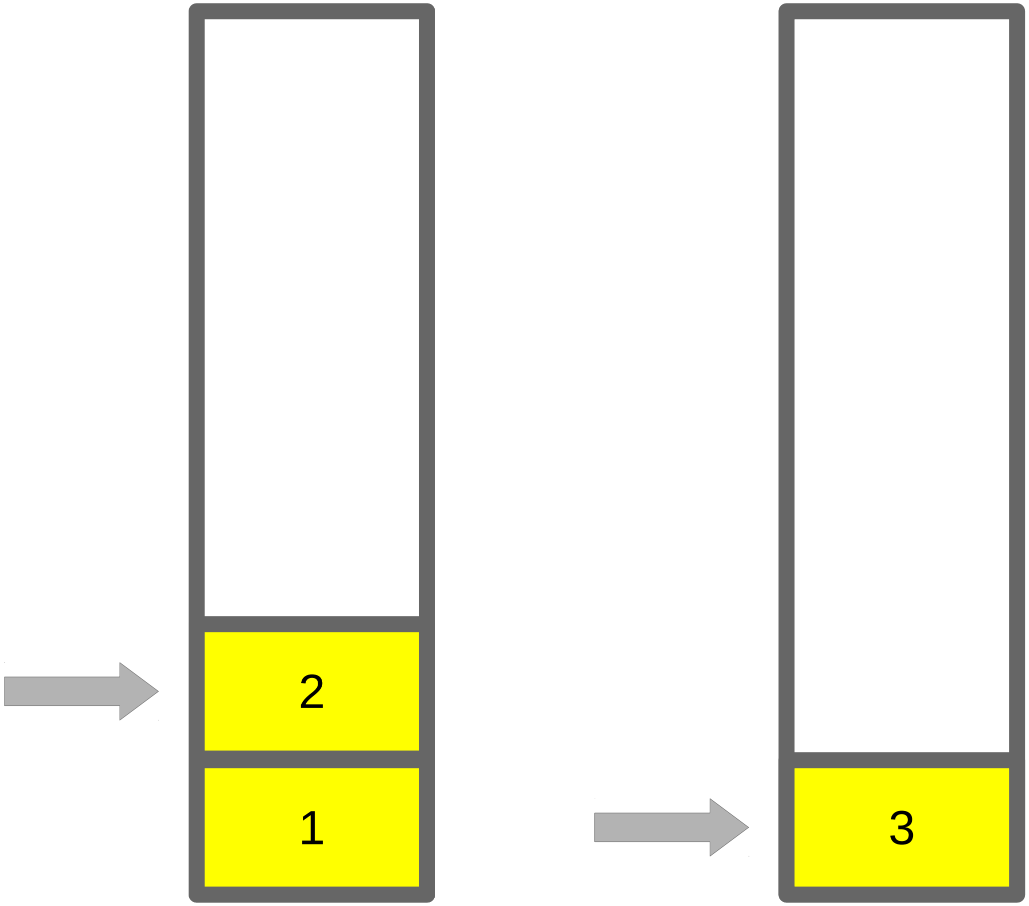
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



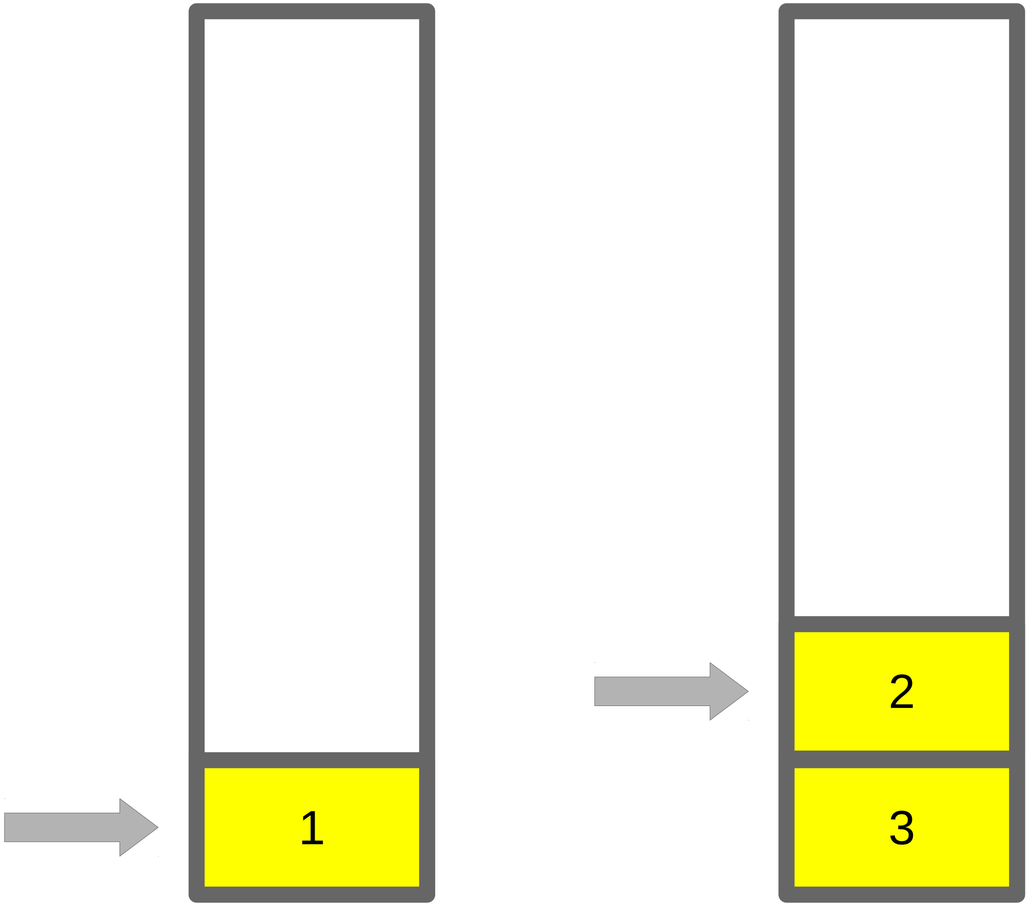
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



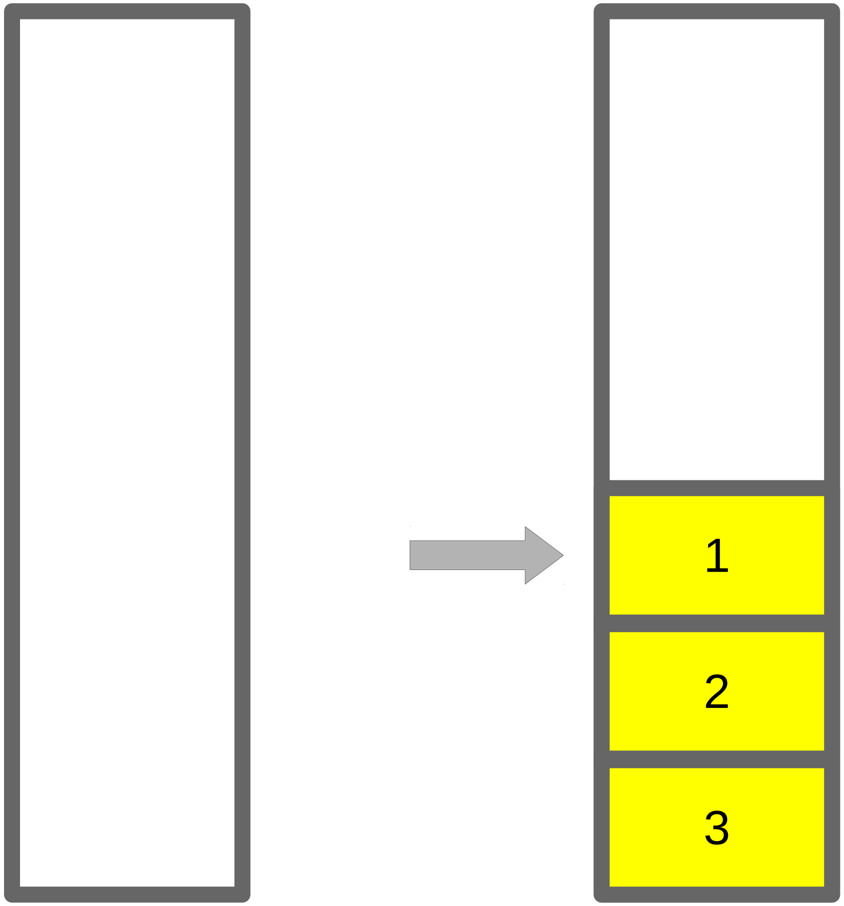
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



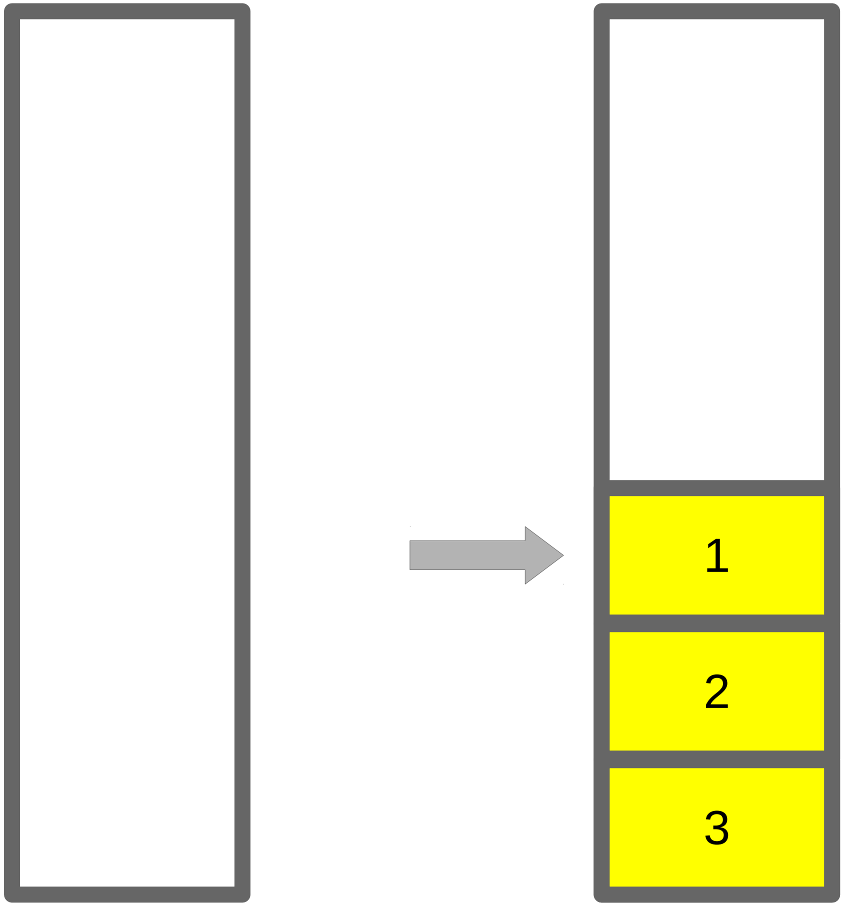
Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



Amortized analysis example: aggregate analysis

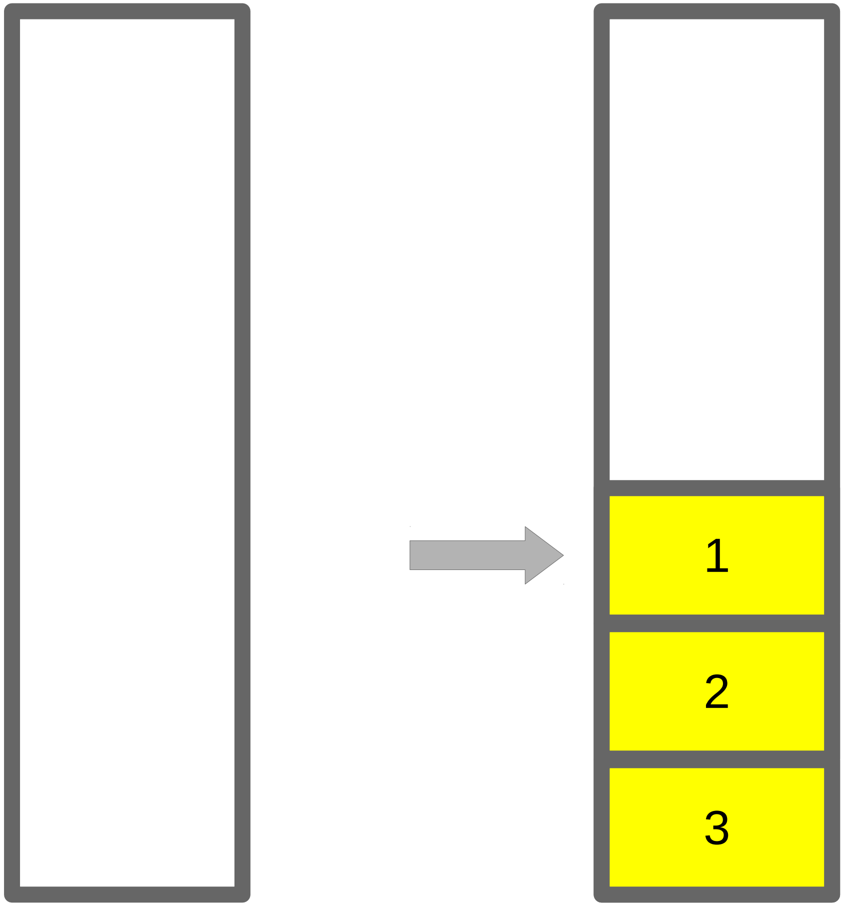
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- How long does each operation take?

Amortized analysis example: aggregate analysis

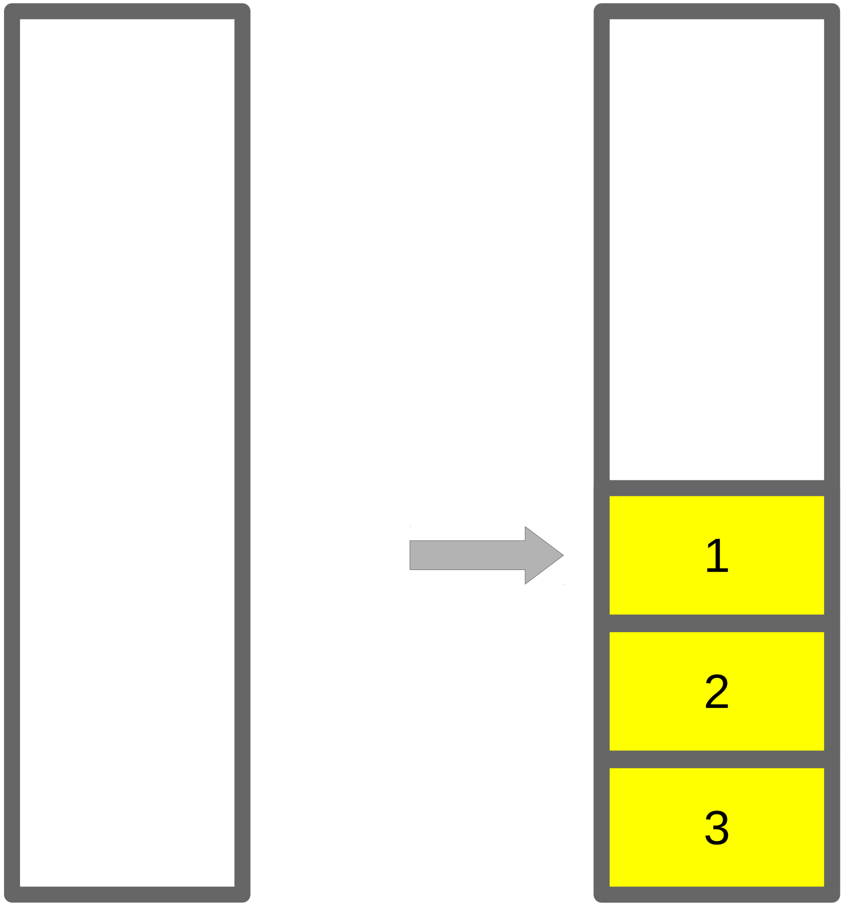
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- How long does each operation take?
- How many times does each element move?

Amortized analysis example: aggregate analysis

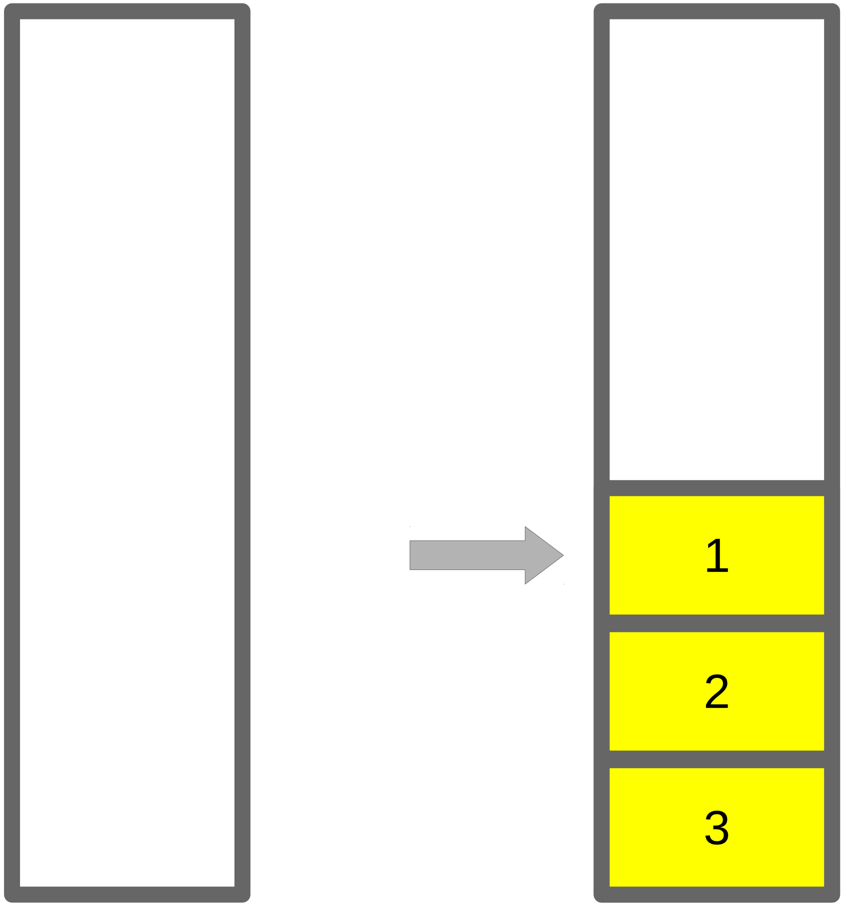
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- How long does each operation take?
- How many times does each element move?
  - Added, moved, removed

Amortized analysis example: aggregate analysis

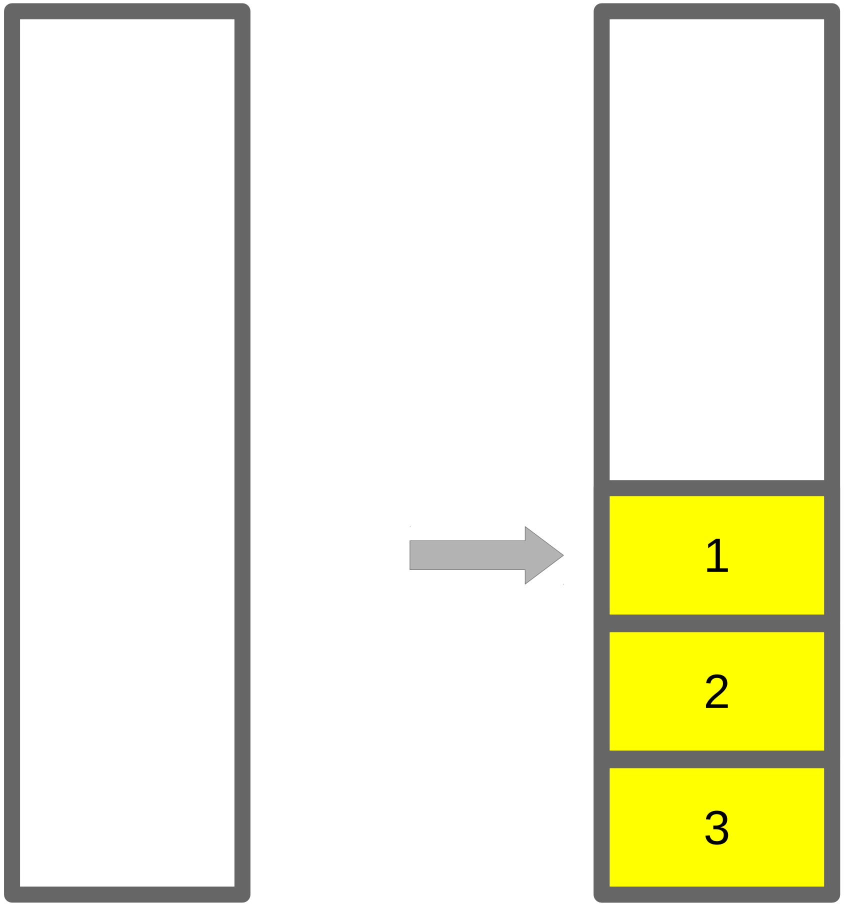
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- How long does each operation take?
- How many times does each element move?
  - Added, moved, removed
  - At most 3 moves

Amortized analysis example: aggregate analysis

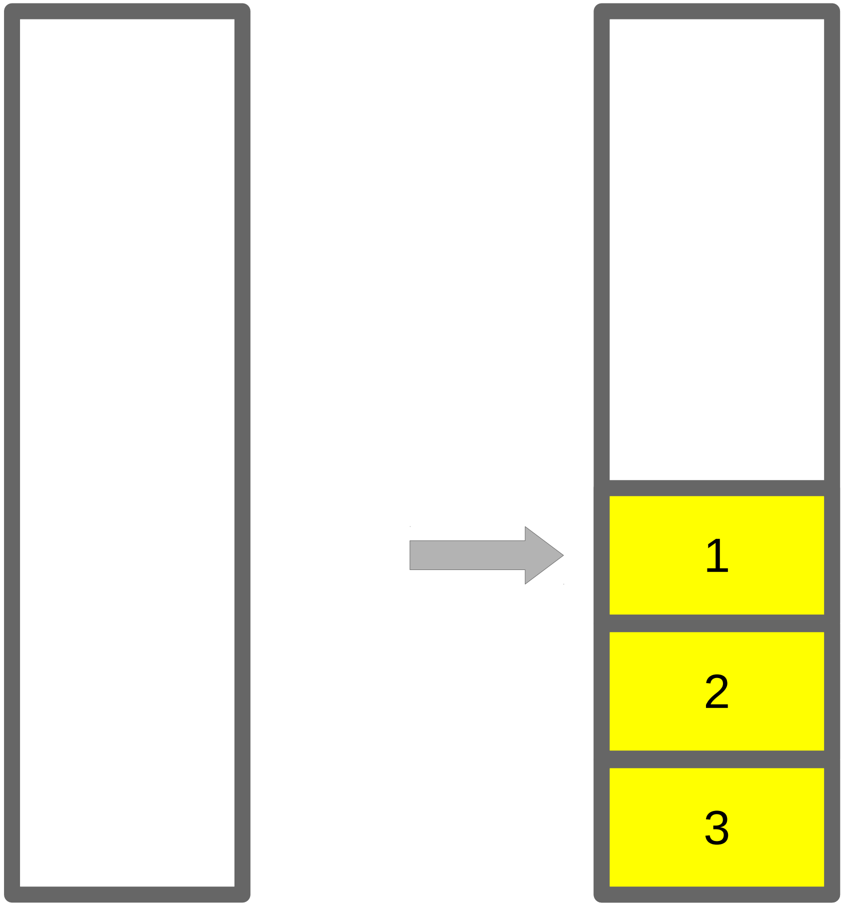
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- How long does each operation take?
- How many times does each element move?
  - Added, moved, removed
  - At most 3 moves
    - O(3n) for n operations

Amortized analysis example: aggregate analysis

- Implementing a queue (first in, first out) with two stacks (first in, last out)



- How long does each operation take?
- How many times does each element move?
  - Added, moved, removed
  - At most 3 moves
    - O(3n) for n operations
    - Constant time for a single operation

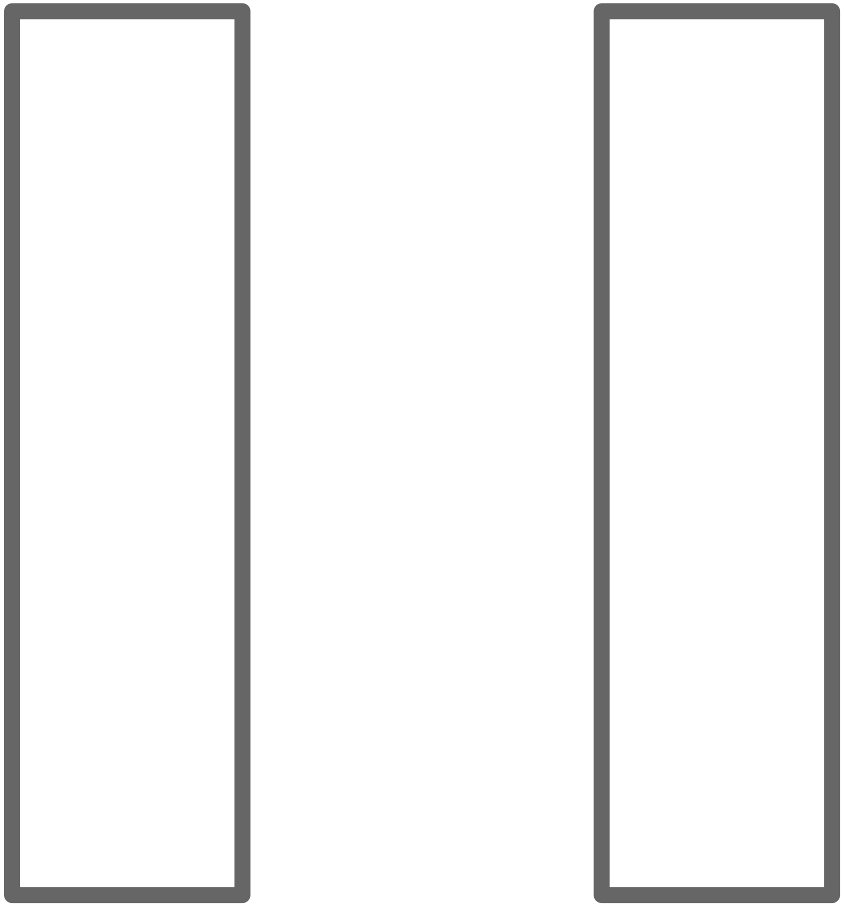
Amortized analysis example: accounting method

Amortized analysis example: accounting method

- Implementing a queue (first in, first out) with two stacks (first in, last out)

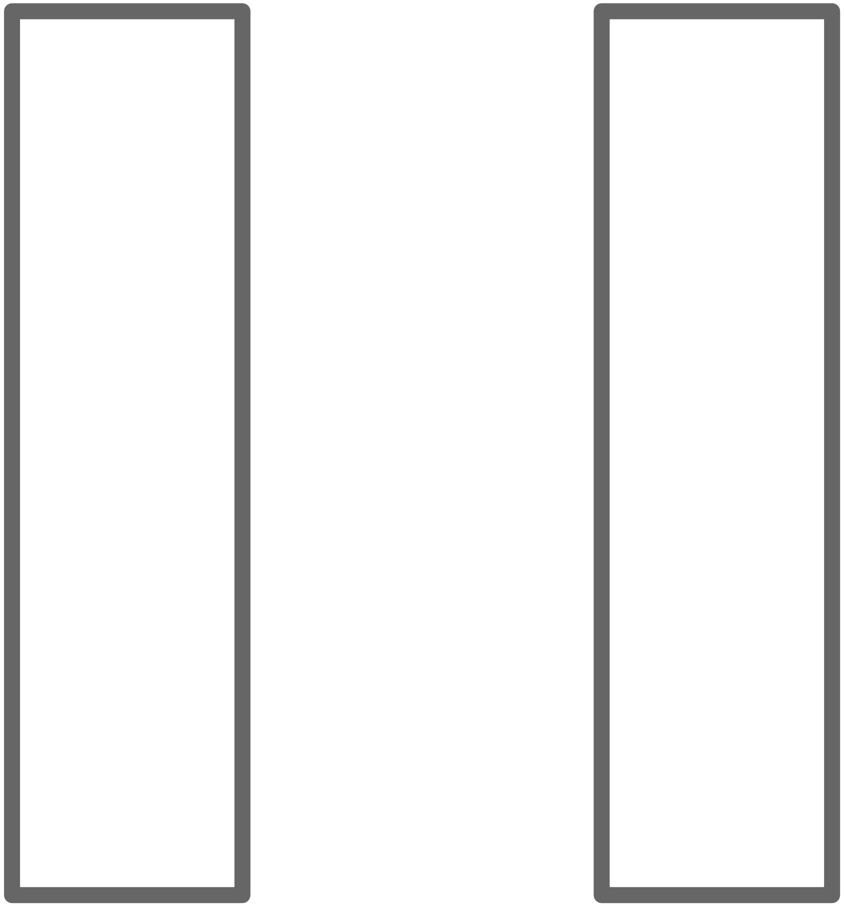
Amortized analysis example: accounting method

- Implementing a queue (first in, first out) with two stacks (first in, last out)



Amortized analysis example: accounting method

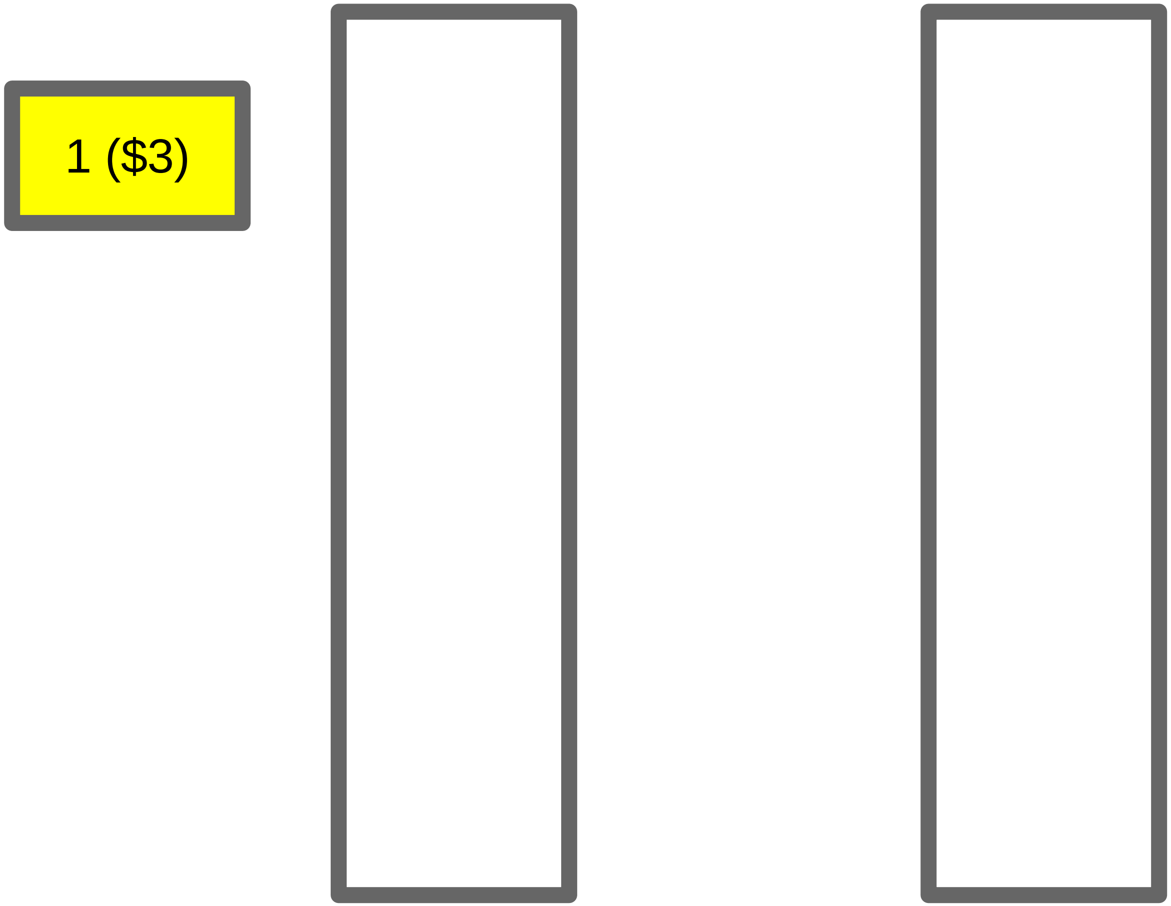
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)

Amortized analysis example: accounting method

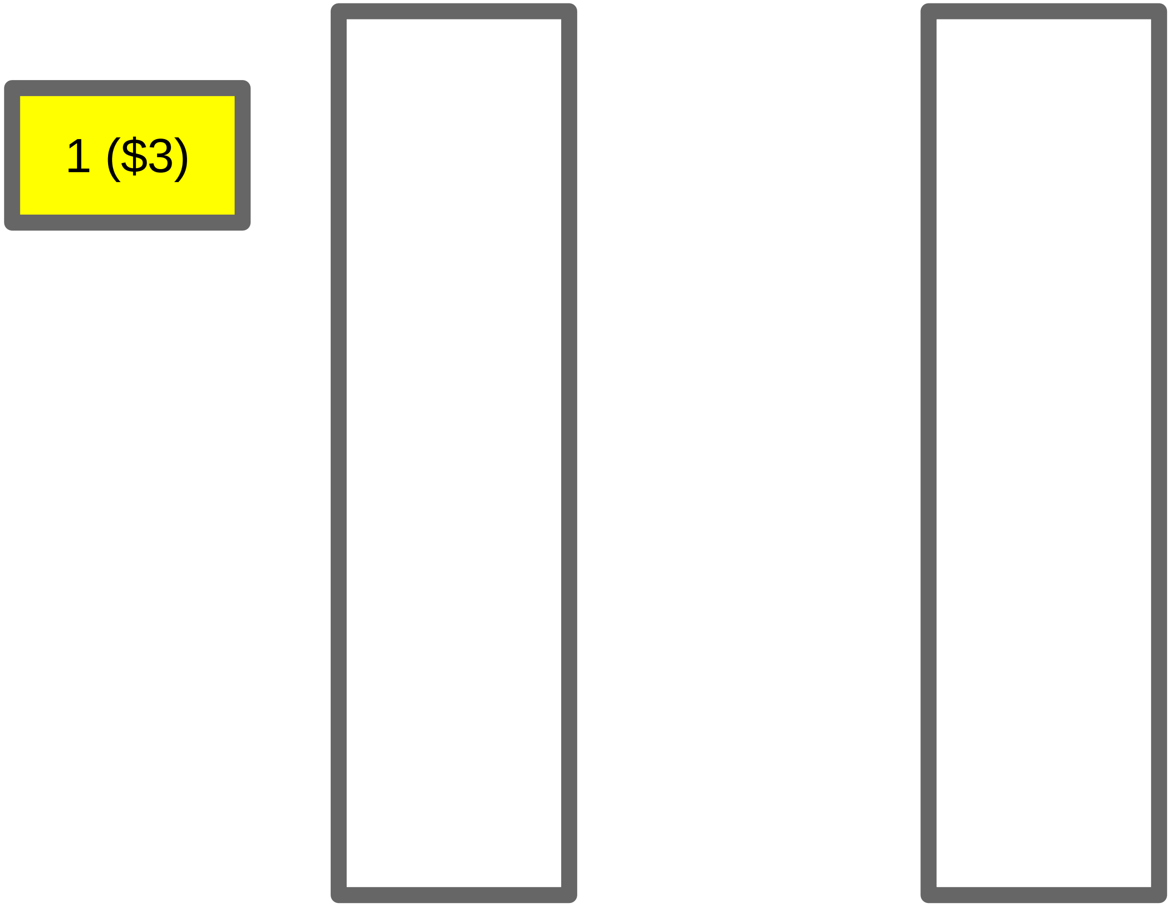
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)

Amortized analysis example: accounting method

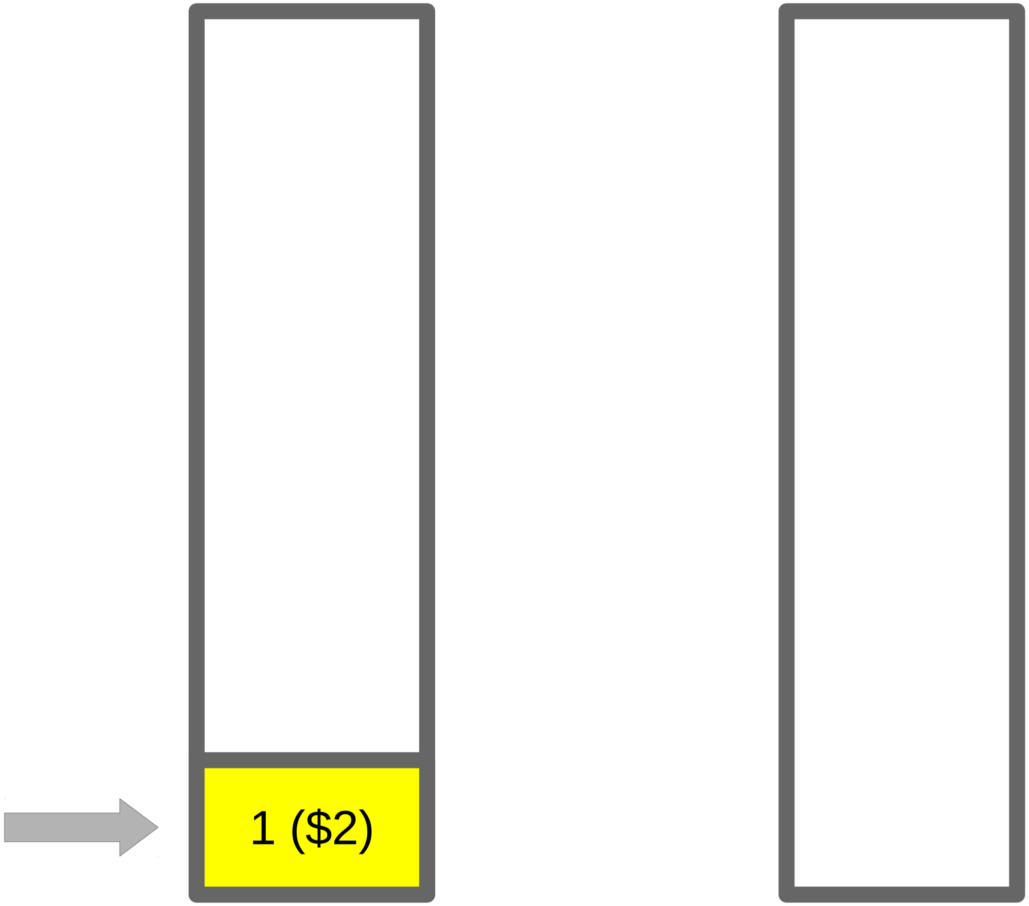
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

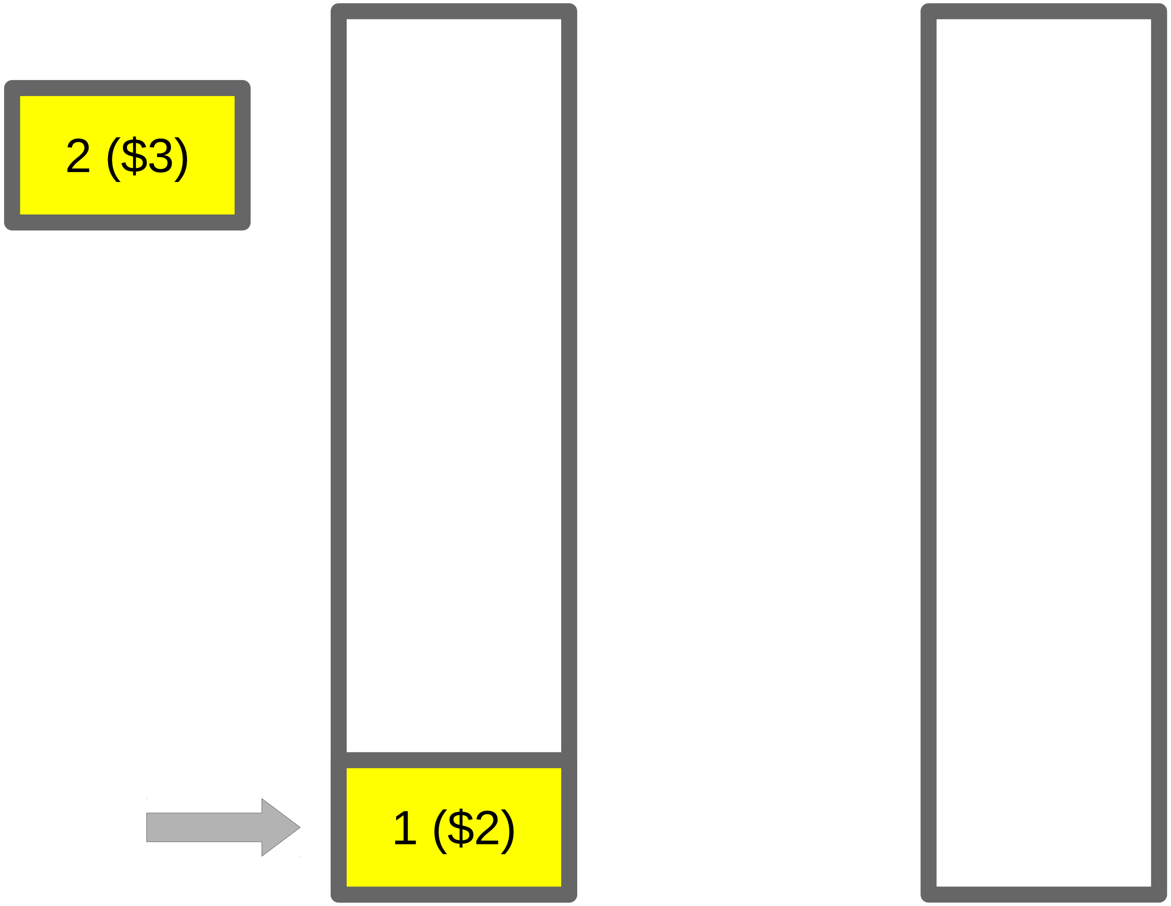
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

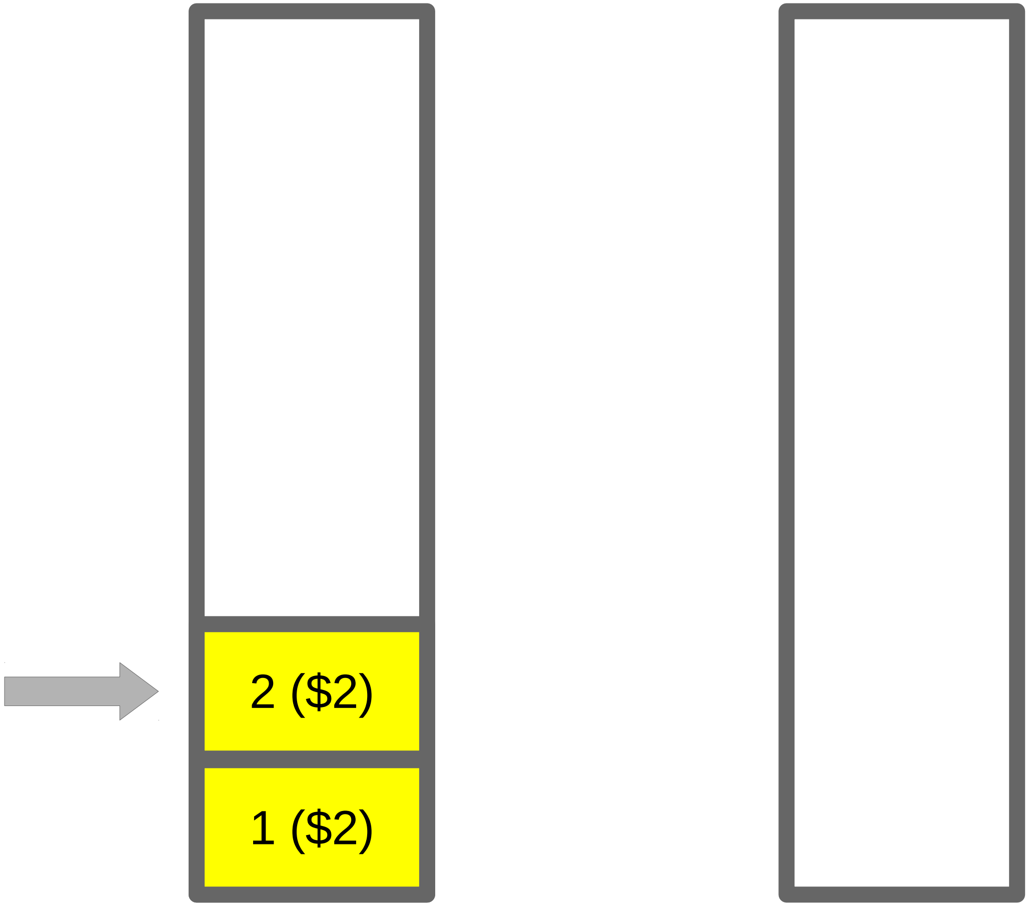
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

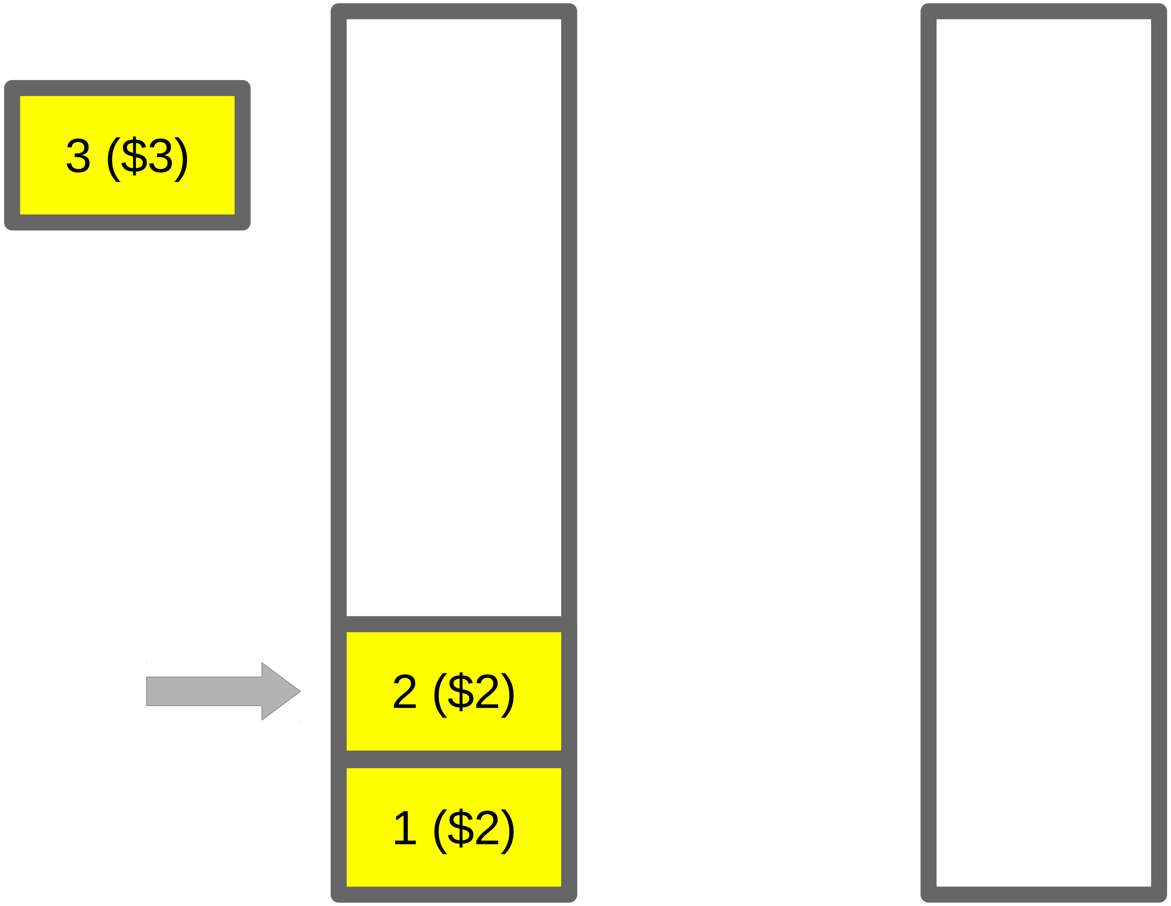
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

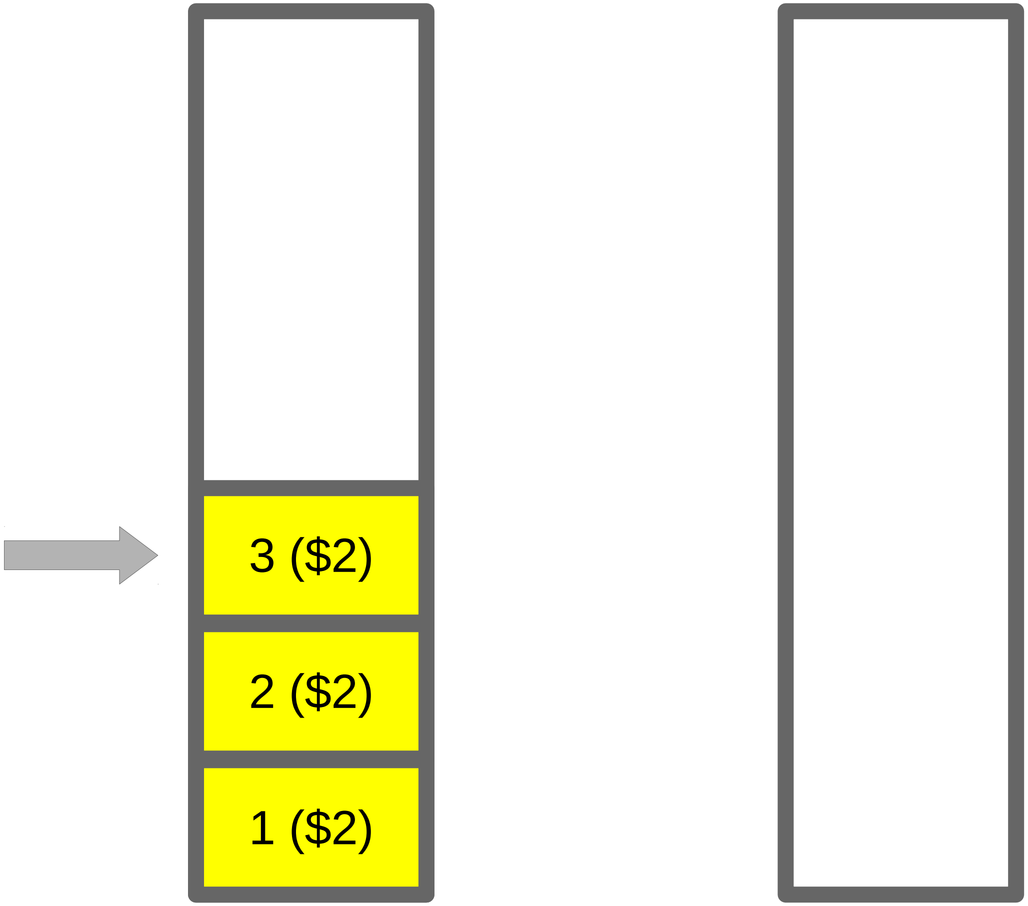
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

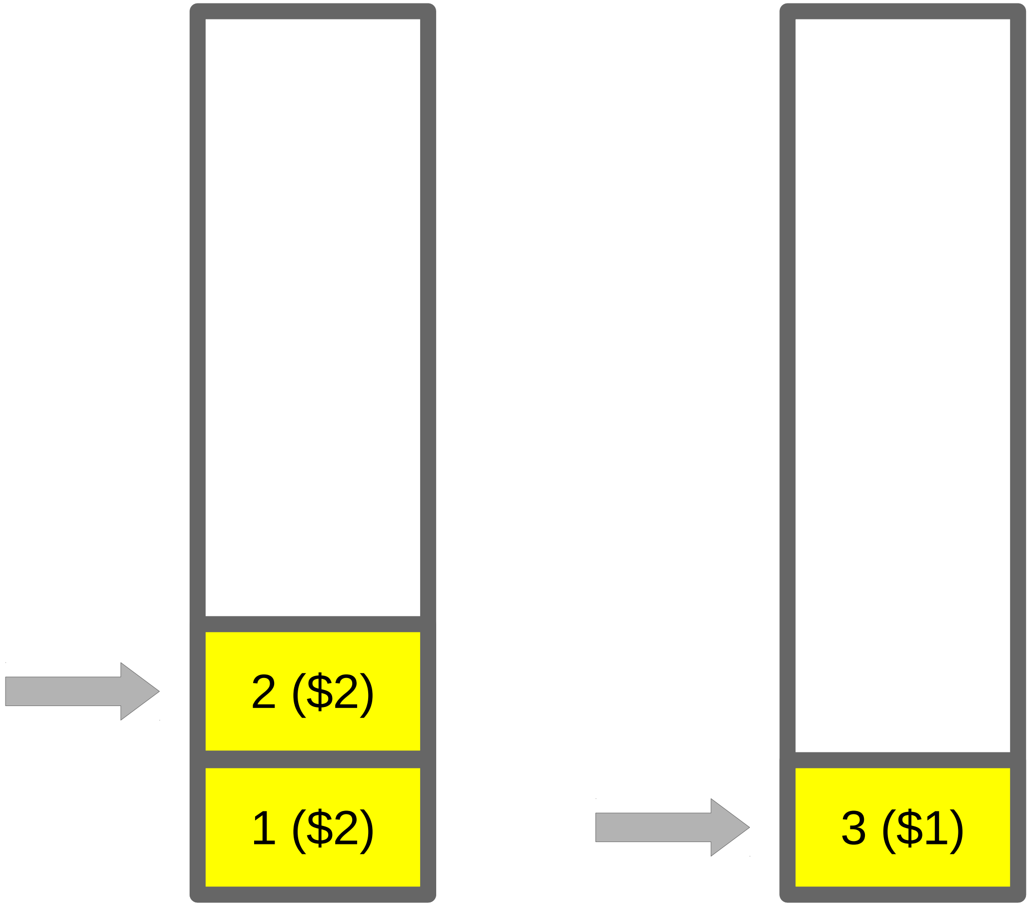
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

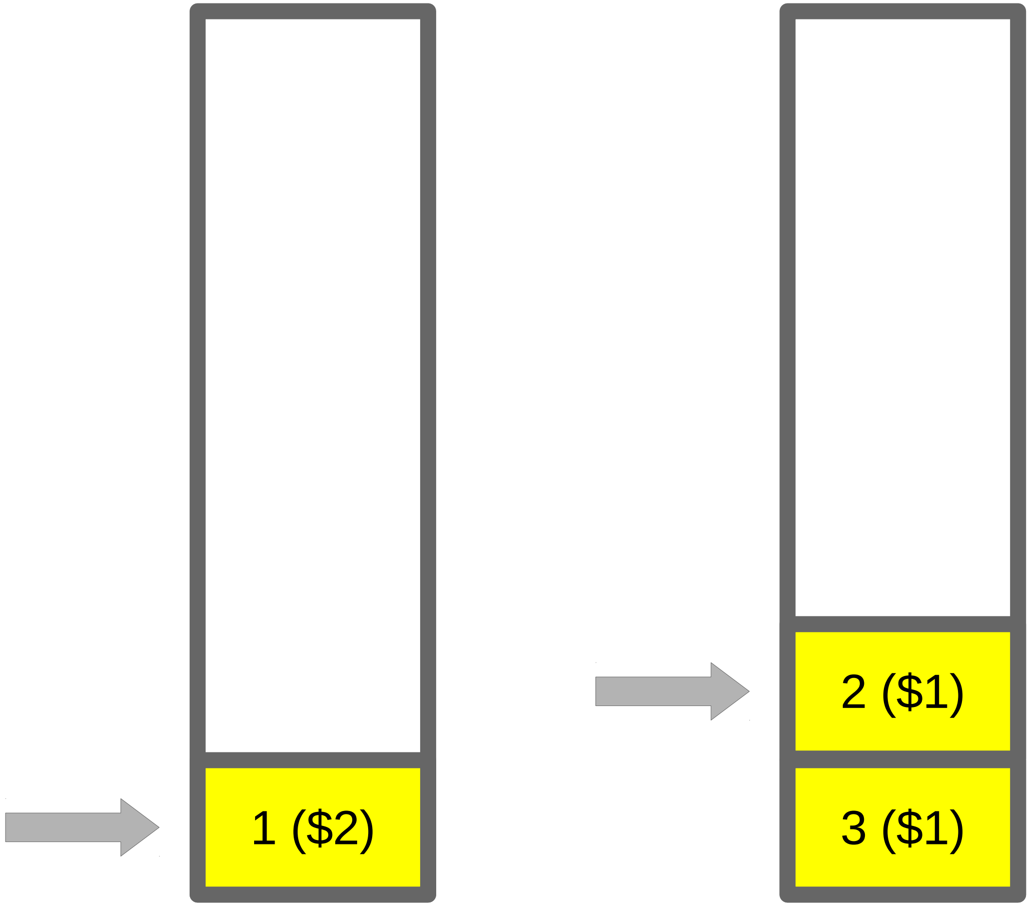
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

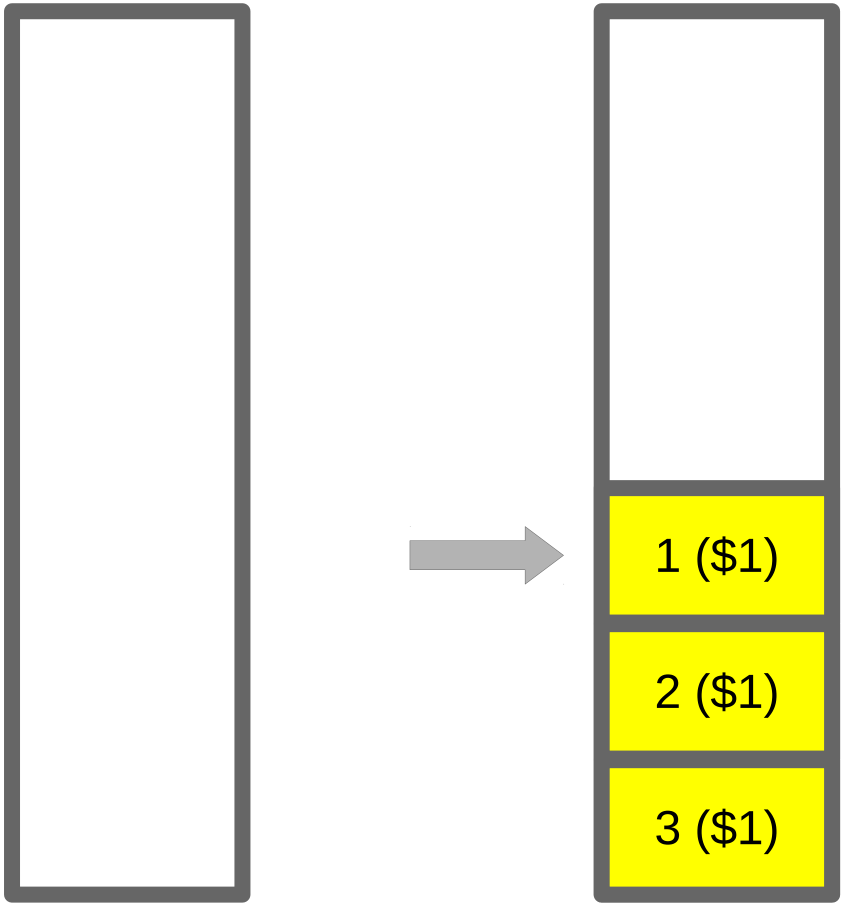
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

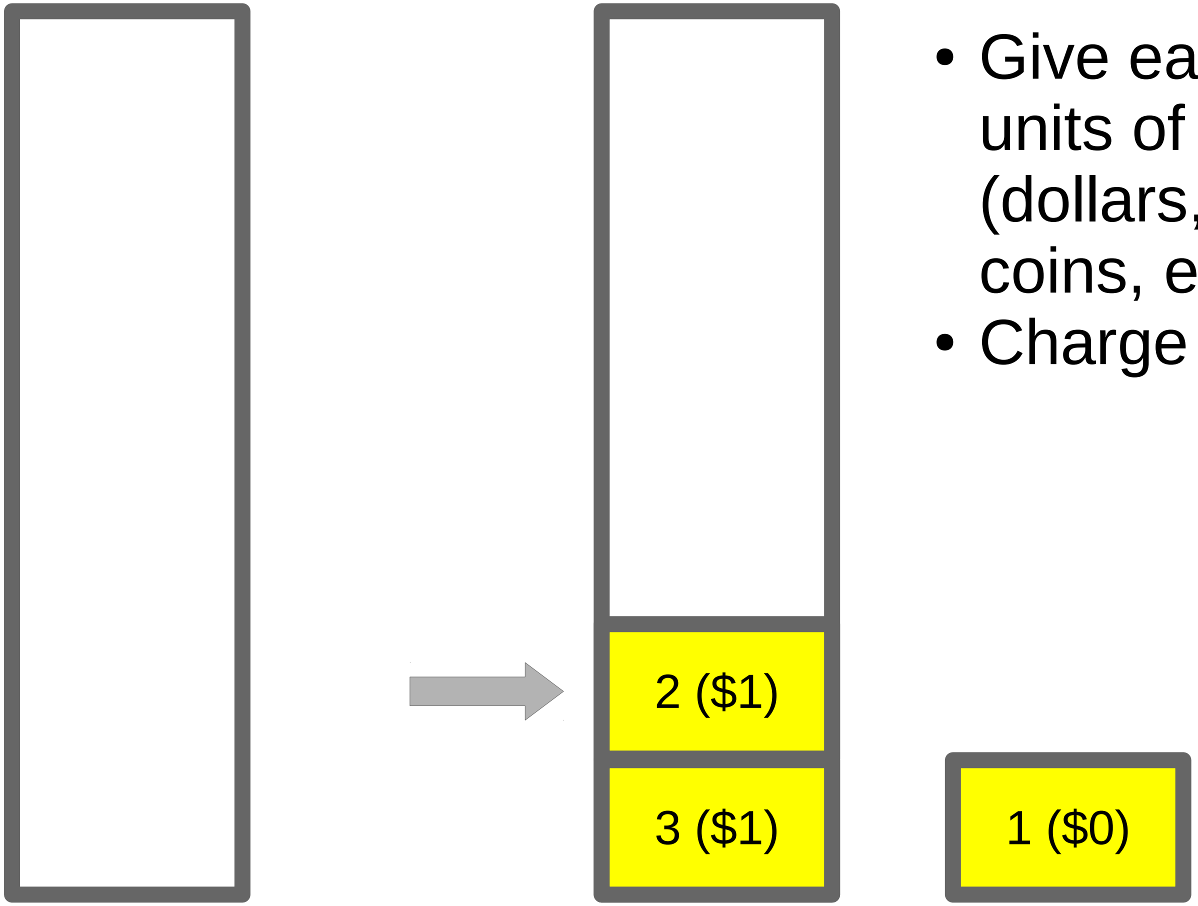
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

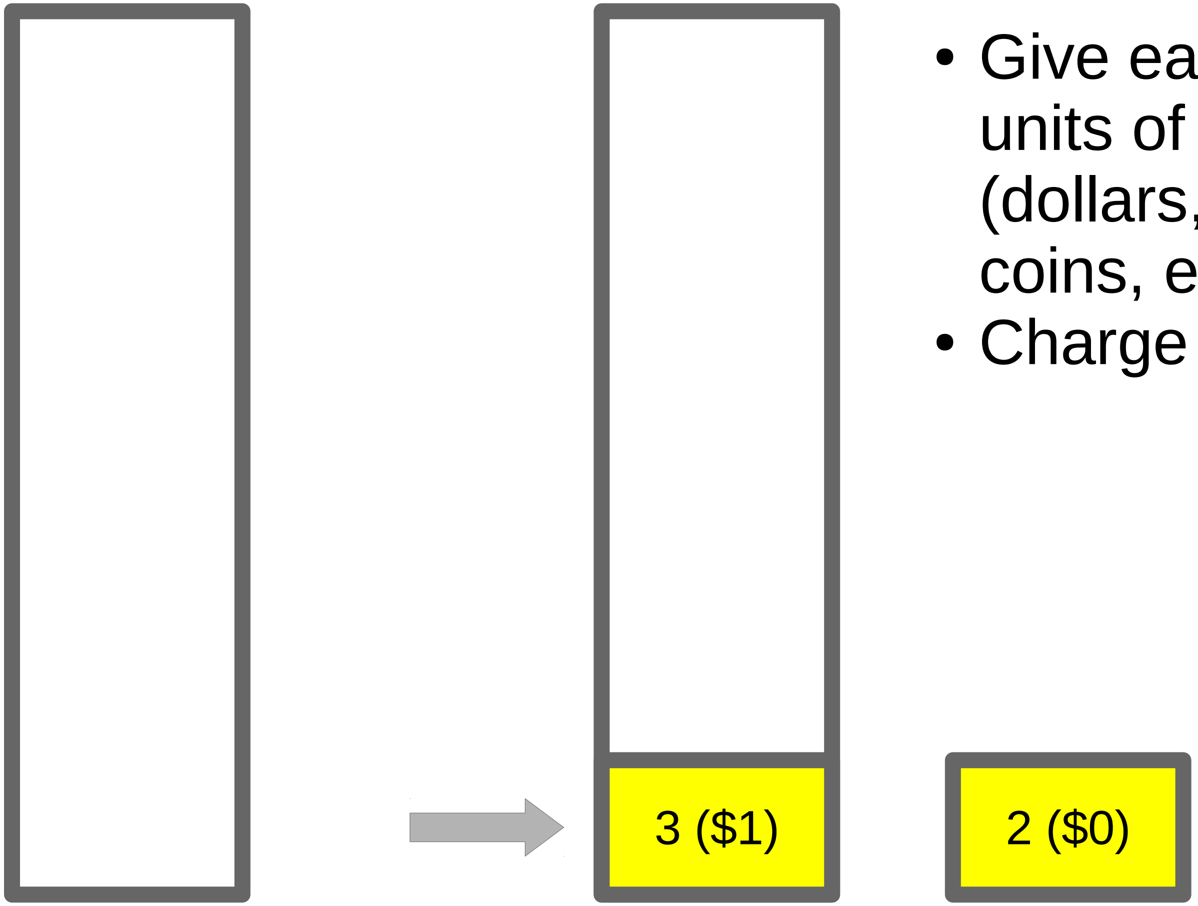
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

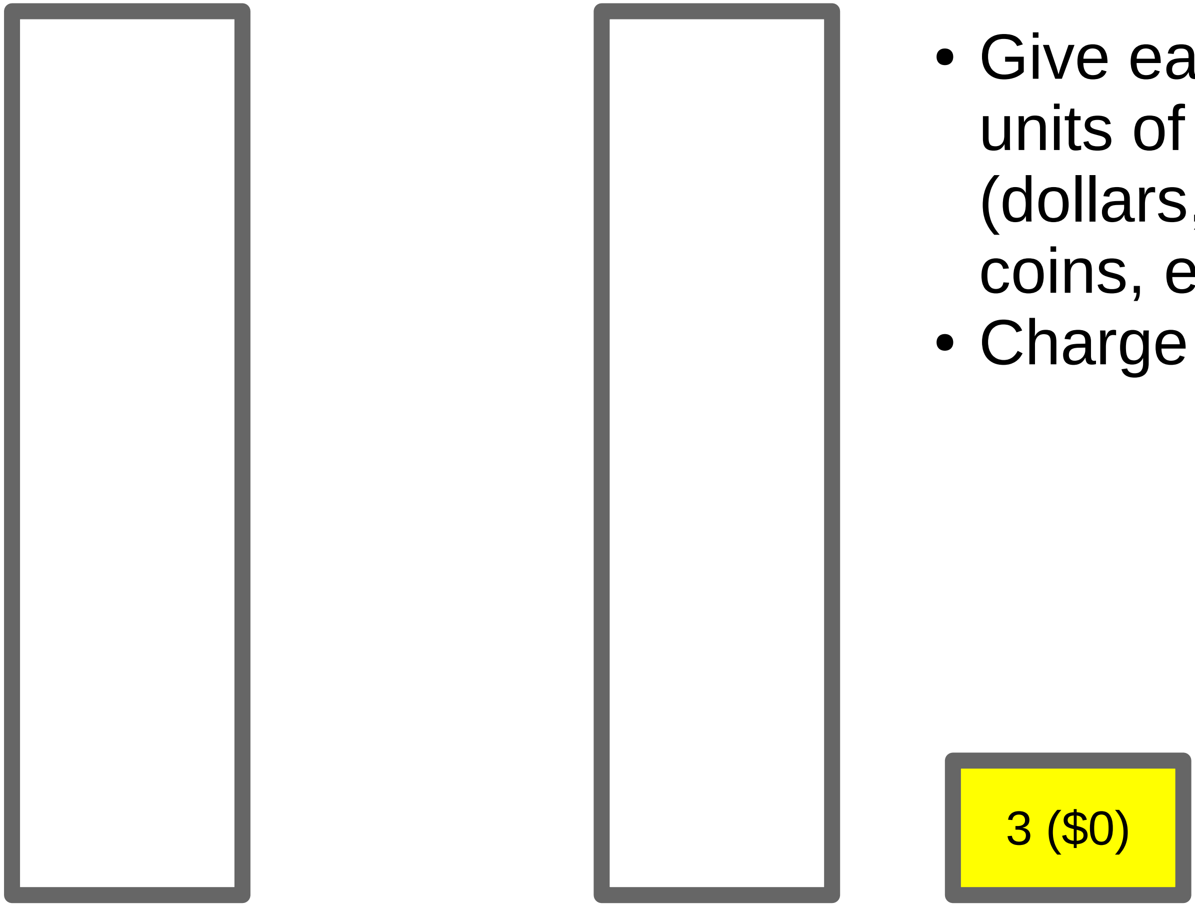
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

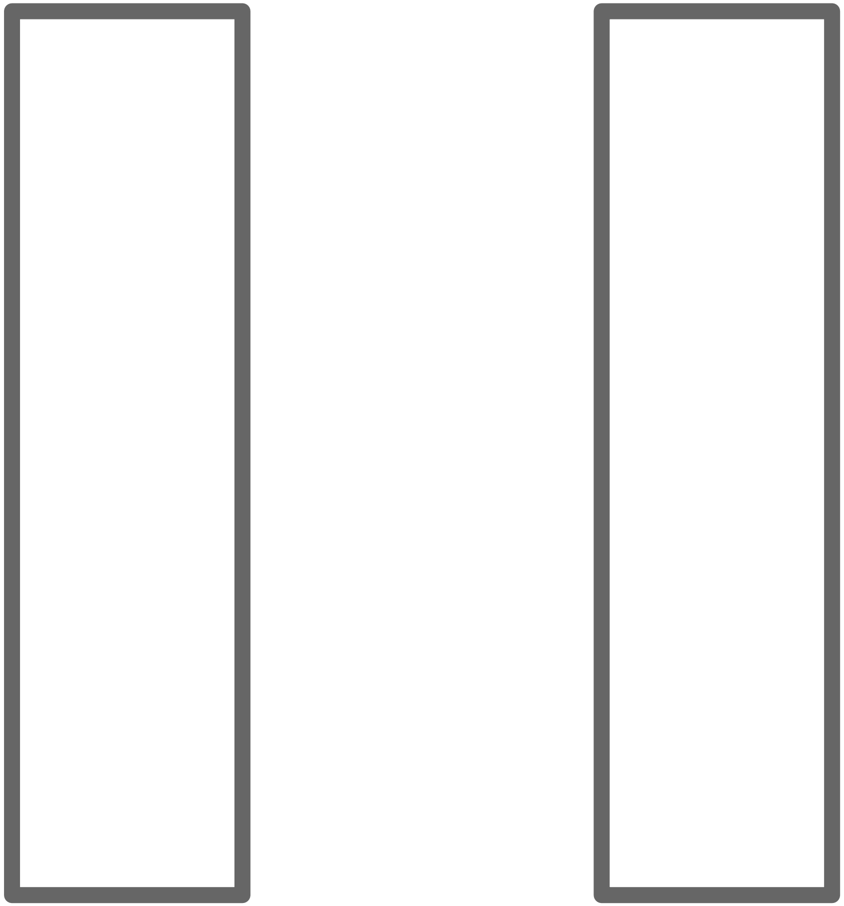
- Implementing a queue (first in, first out) with two stacks (first in, last out)



- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move

Amortized analysis example: accounting method

- Implementing a queue (first in, first out) with two stacks (first in, last out)

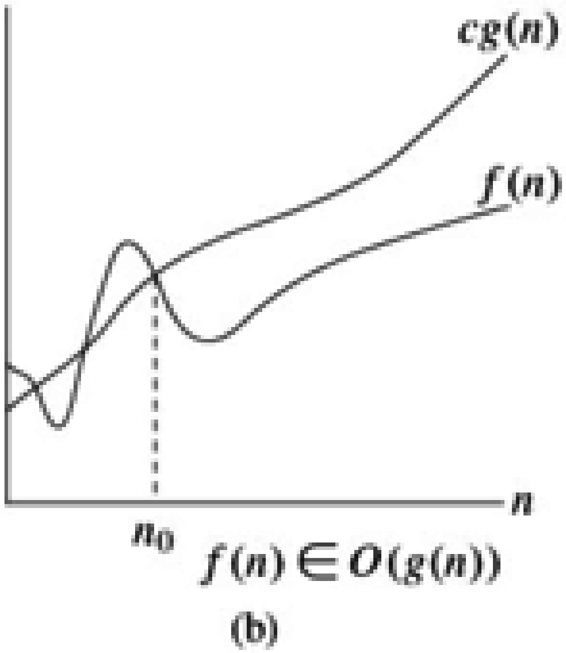


- Give each element 3 units of currency (dollars, bit coins, doge coins, etc.)
- Charge for each move
- We used 3 units of currency (constant time operations) for each element

3 ($0)

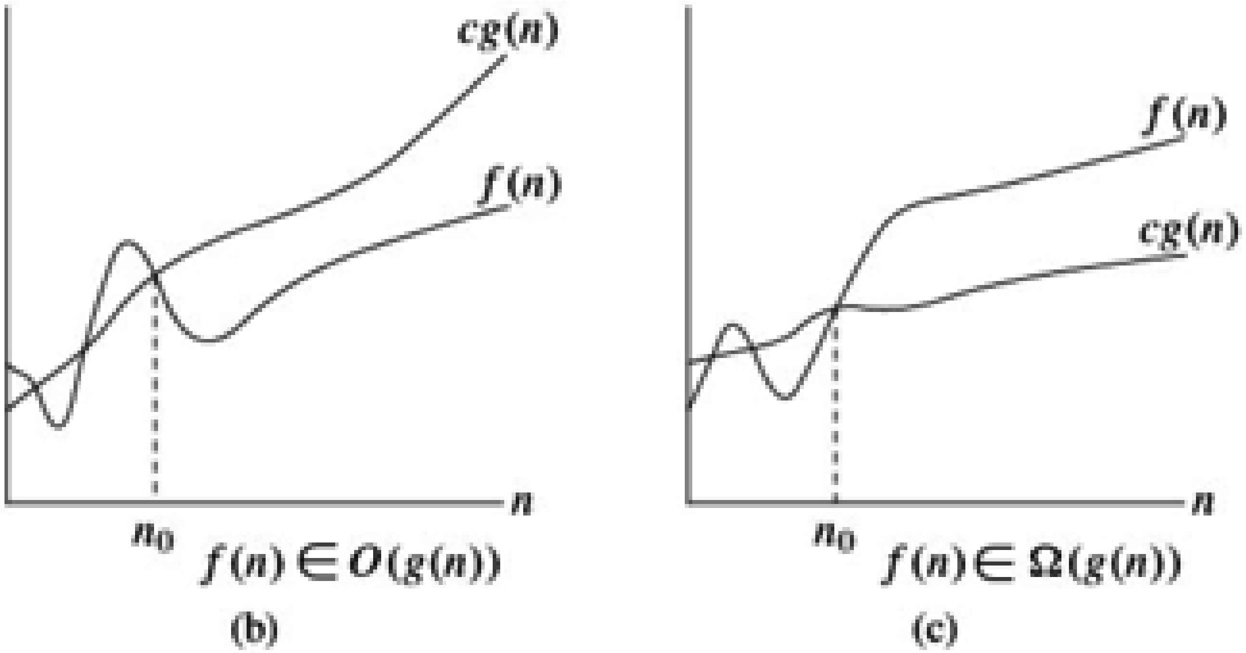
Asymptotic analysis: Big-O, Θ, and Ω notation

Asymptotic analysis: Big-O, Θ, and Ω notation



O: bounded above
given some constant

Asymptotic analysis: Big-O, Θ, and Ω notation



$f(n) \in O(g(n))$

(b)

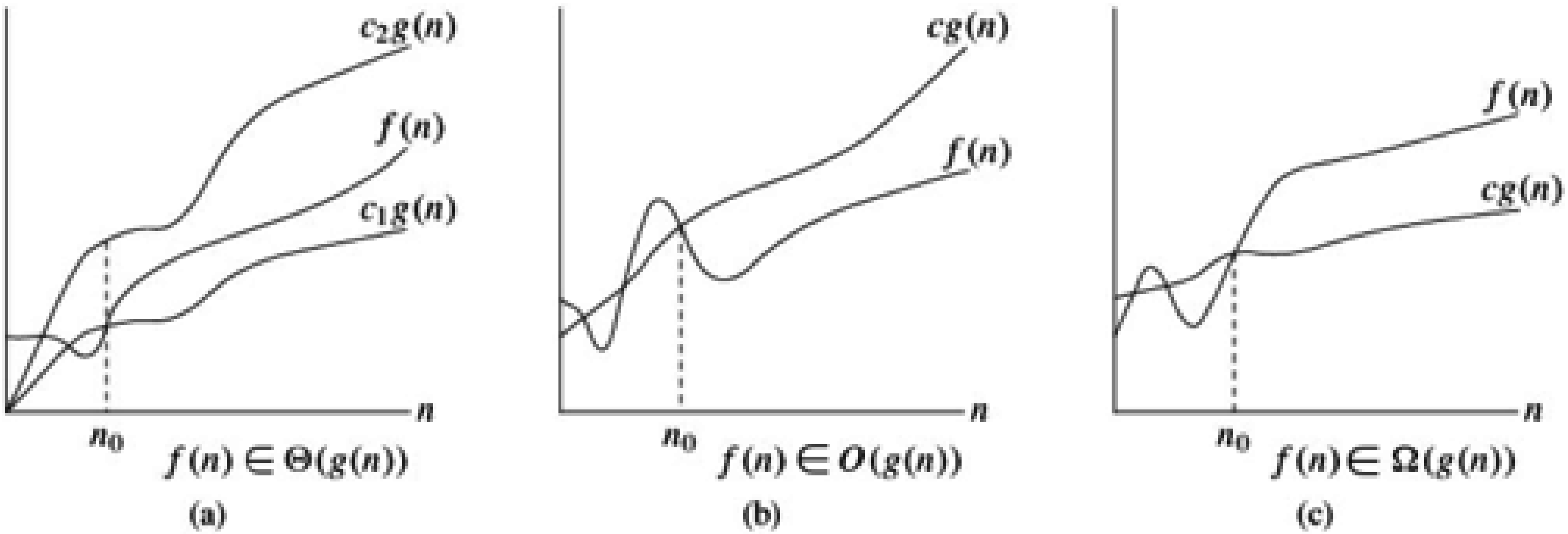
$f(n) \in \Omega(g(n))$

(c)

O: bounded above
given some constant

Ω: bounded above
given some constant

Asymptotic analysis: Big-O, Θ, and Ω notation



$f(n) \in \Theta(g(n))$

(a)

$f(n) \in O(g(n))$

(b)

$f(n) \in \Omega(g(n))$

(c)

Θ: bounded above given some constant, bounded below given some constant

O: bounded above given some constant

Ω: bounded above given some constant