Ch 5. Induction and Recursion

Recursive Algorithms and Recurrence Relations

Sungwon Kang

Acknowledgement

- [Rosen 19] Kenneth H. Rosen, for Discrete Mathematics & Its Applications (8th Edition), Lecture slides
- [Hunter 11] David J. Hunter, Essentials of Discrete Mathematics, 2nd Edition, Jones & Bartlett Publishers, 2011, Lecture Slides

Ch 5. Induction and Recursion

Recursive Algorithms and Recurrence Relations

1. Recursive Algorithms
2. Recurrence Relations
3. Closed Form Solutions

1. Recursive Algorithms

Definition: An algorithm is said to be *recursive* if it solves a problem by reducing it to an instance of the same problem with smaller input.

For a recursive algorithm to terminate, the instance of the problem it solves must eventually be reduced to some initial case for which the solution is known.

Factorial Algorithm

Example: Give a recursive algorithm for computing n!, where n is a nonnegative integer.

Solution:

Example: Give a recursive algorithm for computing $n!$, where n is a nonnegative integer.

Solution: Use the recursive definition of the factorial function.

```
procedure factorial(n: nonnegative integer)
if  n = 0 then return 1
          else return n·factorial(n − 1)
{output is n!}
```

Exponentiation Algorithm

Example: Give a recursive algorithm for computing a^n, where a is a nonzero real number and n is a nonnegative integer.

Solution:

Example: Give a recursive algorithm for computing a^n, where a is a nonzero real number and n is a nonnegative integer.

Solution: Use the recursive definition of a^n.

```
procedure power(a: nonzero real number, n:
  nonnegative integer)
if  n = 0 then return 1
          else  return a· power (a, n − 1)
{output is aⁿ}
```

GCD Algorithm

Example: Give a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$.

Solution:

Example: Give a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$.

Solution: Use the reduction

$$\gcd(a,b) = \gcd(b \textbf{ mod } a, a)$$

and the condition $\gcd(0,b) = b$ when $b > 0$.

Example: Give a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$.

Solution: Use the reduction

$$\gcd(a,b) = \gcd(b \textbf{ mod } a, a)$$

and the condition $\gcd(0,b) = b$ when $b > 0$.

$$
\begin{aligned}
\gcd(18,84) &= \gcd(84 \text{ mod } 18, 18) \\
&= \gcd(12, 18) \\
&= \gcd(18 \text{ mod } 12, 12) \\
&= \gcd(6, 12) \\
&= \gcd(12 \text{ mod } 6, 6) \\
&= \gcd(0,6) \\
&= 6
\end{aligned}
$$

Example: Give a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$.

Solution: Use the reduction

$$\gcd(a,b) = \gcd(b \ \mathbf{mod}\ a, a)$$

and the condition $\gcd(0,b) = b$ when $b > 0$.

```
procedure gcd(a,b: nonnegative integers
             with a < b)
if  a = 0 then return b
          else  return  gcd (b mod  a, a)
{output is gcd(a, b)}
```

2. Recurrence Relations

Definition: A *recurrence relation for the sequence* $\{a_n\}$ is an equation that expresses a_n in terms of one or more of the previous terms, that is, $a_0, a_1, \ldots, a_{n-1}$.

A sequence is called a *solution* of a recurrence relation if its terms satisfy the recurrence relation.

Example

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ n + P(n-1) & \text{if } n > 0 \end{cases}$$

- A kind of recursive definitions
- Different from recursive algorithms
- Often complexity of recursive algorithms can be expressed as recurrence relations

Bottom-up evaluation

Find $P(5)$ using the recurrence relation (bottom-up).

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ n + P(n-1) & \text{if } n > 0 \end{cases}$$

Solution

Bottom-up evaluation

Find $P(5)$ using the recurrence relation (bottom-up).

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ n + P(n-1) & \text{if } n > 0 \end{cases}$$

Solution

P(0) = 1.

P(1) = 1 + P(0) = 1 + 1 = 2.

P(2) = 2 + P(1) = 2 + 2 = 4.

P(3) = 3 + P(2) = 3 + 4 = 7.

P(4) = 4 + P(3) = 4 + 7 = 11.

P(5) = 5 + P(4) = 5 + 11 = 16.

Top-down evaluation

Find $P(5)$ using the recurrence relation (top-down).

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ n + P(n-1) & \text{if } n > 0 \end{cases}$$

Solution

Top-down evaluation

Find $P(5)$ using the recurrence relation (top-down).

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ n + P(n-1) & \text{if } n > 0 \end{cases}$$

Solution

P(5) = 5 + P(4)

= 5 + (4 + P(3))

= 5 + (4 + (3 + P(2)))

= 5 + (4 + (3 + (2 + P(1))))

= 5 + (4 + (3 + (2 + (1 + P(0)))))

= 5 + (4 + (3 + (2 + (1 + 1))))

= 5 + (4 + (3 + (2 + 2)))

= 5 + (4 + (3 + 4))

= 5 + (4 + 7)

= 5 + 11 = 16

Recurrence relation for the size of a power set

Let X be a finite set with n elements. Find a recurrence relation $C(n)$ for the number of elements in the power set $\mathcal{P}(X)$.

Solution

Recurrence relation for the size of a power set

Let X be a finite set with n elements. Find a recurrence relation $C(n)$ for the number of elements in the power set $\mathcal{P}(X)$.

Solution

$$C(n) = \begin{cases} 1 & \text{if } n = 0 \\ 2 \cdot C(n-1) & \text{if } n > 0 \end{cases}$$

(Why?)

Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:

Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:

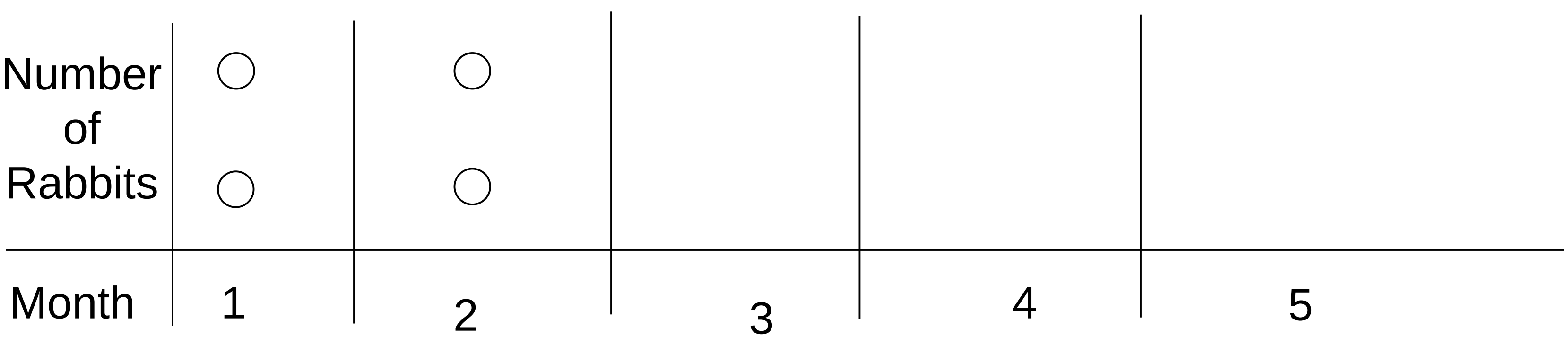
Number of Rabbits					
Month	1	2	3	4	5

Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:

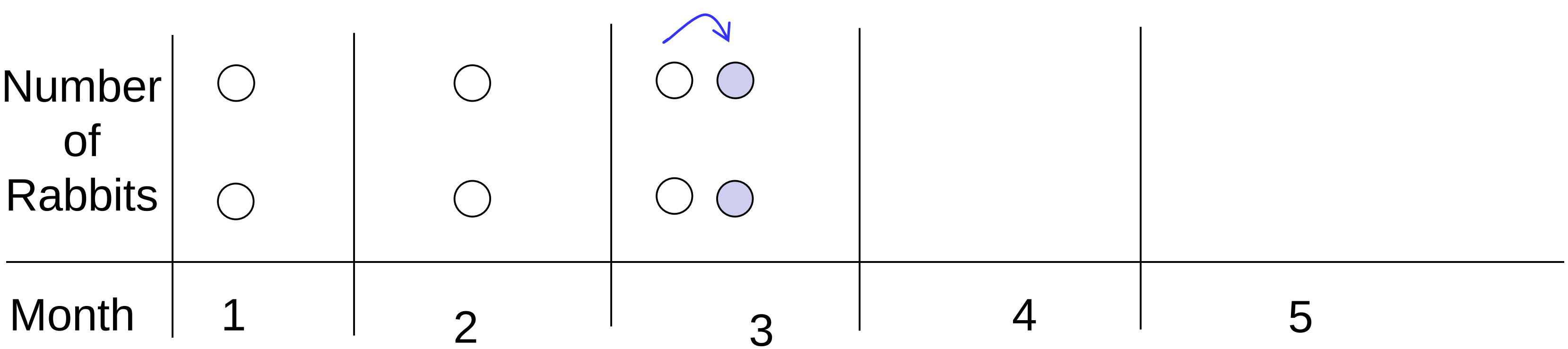


Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:

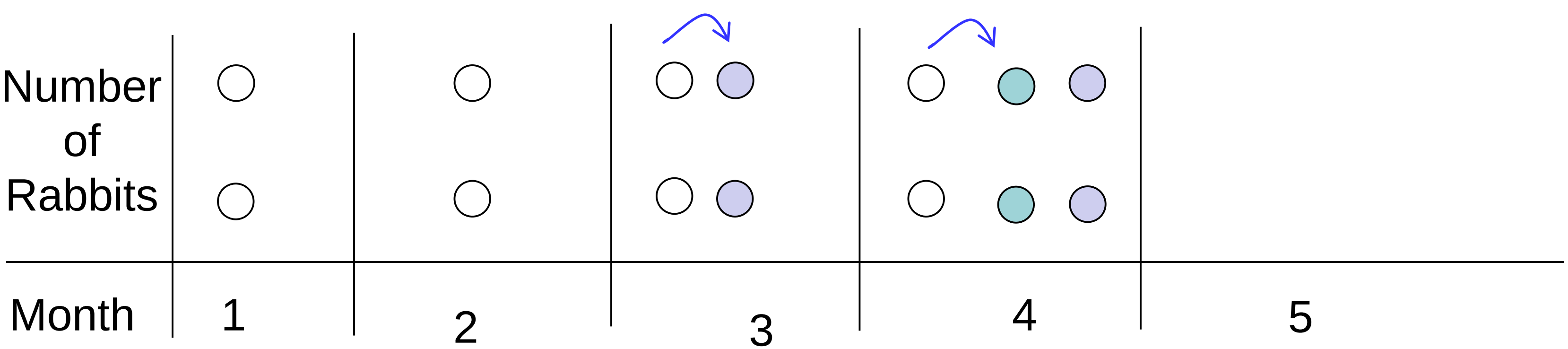


Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:

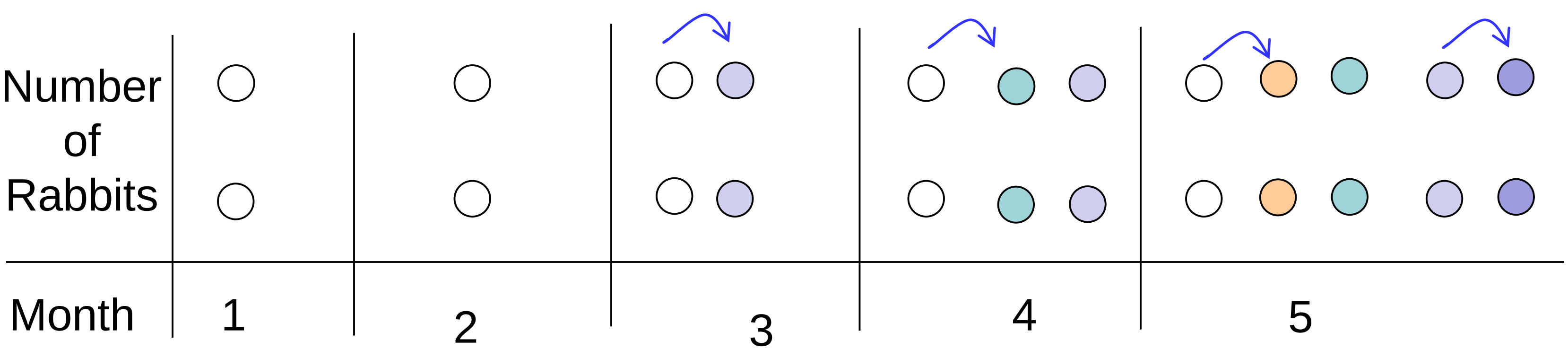


Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:



Fibonacci's rabbits

In the early thirteenth century, the Italian mathematician Leonardo Pisano Fibonacci asked:

> *A certain man put a pair of rabbits in a place surrounded on all sides by a wall. How many pairs of rabbits can be produced from that pair in a year if it is supposed that every month each pair begets a new pair which from the second month on becomes productive?*

Recurrence relation answer:

$$F(n) = \begin{cases} 1 & \text{if } n = 1 \text{ or } n = 2 \\ F(n-1) + F(n-2) & \text{if } n > 2 \end{cases}$$

The sequence $F(1), F(2), F(3), \ldots$ is called the *Fibonacci sequence*.

3. Closed Form Solutions

Complexity of recursive algorithms expressed with recurrence relations can be directly known by finding their closed-form solutions

The recurrence relation

$$P(n) = \begin{cases} 1 & \text{if } n = 1 \\ n + P(n-1) & \text{if } n > 1 \end{cases}$$

and the formula

$$P(n) = \frac{n(n+1)}{2}$$

give exactly the same answers for all values of n. (Mathematical Induction can be used to prove this) The latter, non-recursive, formula is called a *closed-form solution* to the recurrence relation.

Guessing a closed-form solution: finding patterns

Evaluating

$$M(n) = \begin{cases} 500 & \text{if } n = 0 \\ 1.10 \cdot M(n-1) & \text{if } n > 0 \end{cases}$$

gives

$$\begin{aligned} M(0) &= 500 &&= 500\,(1.10)^0 \\ M(1) &= 500 \cdot 1.10 &&= 500(1.10)^1 \\ M(2) &= 500 \cdot 1.10 \cdot 1.10 &&= 500(1.10)^2 \\ M(3) &= 500 \cdot 1.10 \cdot 1.10 \cdot 1.10 &&= 500(1.10)^3 \\ M(4) &= 500 \cdot 1.10 \cdot 1.10 \cdot 1.10 \cdot 1.10 &&= 500(1.10)^4 \end{aligned}$$

The pattern suggests that $M(n) = 500(1.10)^n$ is a closed-form solution to the recurrence relation.

$$H(n) = \begin{cases} 1 & \text{if } n = 1 \\ H(n-1) + 6n - 6 & \text{if } n > 1 \end{cases}$$

How to find a closed-form solution?

Approach 1)

H(k) = H(k-1) + 6k – 6
= {H(k-2) + 6(k-1) -6} + 6k -6
= {[H(k-3) + 6(k-2) -6] + 6(k-1)-6}+ 6k-6
= H(k-3) + 6x{(k-2)+(k-1)+k} - 3x6
. . .
= H(1) + 6x{2+…+(k-2)+(k-1)+k} – (k-1)x6 - - pattern has been found - (A)
= 1 + 6x(k-1)(k+2)/2 – (k-1)x6 - - pattern has been simplified - (B)
= 1+ 3x(k-1)(k+2) – (k-1)x6
= 1 + 3x(k-1)k
= $3k^2$-3k +1

$$H(n) = \begin{cases} 1 & \text{if } n = 1 \\ H(n-1) + 6n - 6 & \text{if } n > 1 \end{cases}$$

How to find a closed-form solution?

Approach 2)

$\Sigma_{n=1}^{k} (H(n) - H(n-1)) = \Sigma_{n=1}^{k} (6n - 6)$ -- (C)

$H(k) - H(1) = 6 (\Sigma_{n=1}^{k} n - \Sigma_{n=1}^{k} 1)$
$= 6 (k(k+1)/2 - k)$
$= 6 (k(k-1)/2)$
$= 3k^2 - 3k$ -- (D)

Since $H(1) = 1$, $H(k) = 3k^2 - 3k + 1$.

Sequences of differences

Approach 3)

$$H(n) = \begin{cases} 1 & \text{if } n = 1 \\ H(n-1) + 6n - 6 & \text{if } n > 1 \end{cases}$$

The first six values are:

$$\begin{array}{ccccccccccc}
1 & & 7 & & 19 & & 37 & & 61 & & 91 \\
& \vee & & \vee & & \vee & & \vee & & \vee & \\
& 6 & & 12 & & 18 & & 24 & & 30 & \\
& & \vee & & \vee & & \vee & & \vee & & \\
& & 6 & & 6 & & 6 & & 6 & &
\end{array}$$

The second sequence of differences is constant. This suggests that the sequence may have a formula of the form
$H(n) = An^2 + Bn + C$.

Sequences of differences

H(n) = An² + Bn + C -- (E)

Solving the system of equations

$$
\begin{aligned}
\textbf{n = 1:} \quad 1 &= A + B + C \\
\textbf{n = 2:} \quad 7 &= 4A + 2B + C \\
\textbf{n = 3:} \quad 19 &= 9A + 3B + C
\end{aligned}
$$

gives $H(n) = 3n^2 - 3n + 1$

Inductively verifying a solution: general template

Given:

$$
\begin{aligned}
R(n) &= \text{some recurrence relation} \\
f(n) &= \text{hypothesized closed-form formula}
\end{aligned}
$$

We wish to prove that $R(n) = f(n)$ for all **$n \geq 0$.** A proof by induction has the following parts:

Base Case: Verify that **$R(0) = f(0)$.**

$$\boxed{\begin{array}{l}\textbf{Inductive Hypothesis: } \text{Let } \mathbf{k \geq 0} \text{ be some (unspecified) integer.} \\ \text{Suppose as } \textit{inductive hypothesis} \text{ that } R(k-1) = f(k-1).\end{array}}$$

Part of Inductive Step

Inductive Step: Prove that $R(k) = f(k)$.

Why bother proving anything?

Let $P(n)$ be defined by the following recurrence relation.

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ 3 \cdot P(n-1) - (n-1)^2 & \text{if } n > 0 \end{cases}$$

Is f(n) $= (n+2) \cdot 2^{n-1}$ **the solution for P(n) ?**

$f(0) = 2 \times 2^{-1} = 1 \quad = \quad P(0) = 1$

$f(1) = 3 \times 2^{0} = 3 \quad = \quad P(1) = 3 \times P(0) - 0^2 = 3$

$f(2) = 4 \times 2^{1} = 8 \quad = \quad P(2) = 3 \times P(1) - 1^2 = 8$

$f(3) = 5 \times 2^{2} = 20 \quad = \quad P(3) = 3 \times P(2) - 2^2 = 20$

Why bother proving anything?

Let $P(n)$ be defined by the following recurrence relation.

$$P(n) = \begin{cases} 1 & \text{if } n = 0 \\ 3 \cdot P(n-1) - (n-1)^2 & \text{if } n > 0 \end{cases}$$

Is f(n) $= (n+2) \cdot 2^{n-1}$ **the solution for P(n) ?**

$f(0) = 2 \times 2^{-1} = 1$ = $P(0) = 1$

$f(1) = 3 \times 2^{0} = 3$ = $P(1) = 3 \times P(0) - 0^2 = 3$

$f(2) = 4 \times 2^{1} = 8$ = $P(2) = 3 \times P(1) - 1^2 = 8$

$f(3) = 5 \times 2^{2} = 20$ = $P(3) = 3 \times P(2) - 2^2 = 20$

$f(4) = 6 \times 2^{3} = 48$ $\neq$ $P(4) = 3 \times P(3) - 3^2 = 51$

Verifying closed form solutions – Example 1

Let $C(n)$ be defined by the following recurrence relation (giving the size of the power set of a set with n elements).

$$C(n) = \begin{cases} 1 & \text{if } n = 0 \\ 2 \cdot C(n-1) & \text{if } n > 0 \end{cases}$$

Prove that $C(n) = 2^n$ for all $n \geq 0$.

Proof:

We use induction on n. Let $f(n) = 2^n$.

Base Case: If $n = 0$, the recurrence relation says that $C(0) = 1$, and the formula says that $f(0) = 2^0 = 1$, so $C(0) = f(0)$.

Inductive Hypothesis: Let $k > 0$. Suppose as inductive hypothesis that

$$C(k-1) = 2^{k-1}.$$

Part of inductive step

Inductive Step: Using the recurrence relation,

$$\begin{aligned} C(k) &= 2 \cdot C(k-1), \text{ by the second part of the recurrence relation} \\ &= 2 \cdot 2^{k-1}, \text{ by inductive hypothesis} \\ &= 2^k \end{aligned}$$

so, by induction, $C(n) = 2^n$ for all $n \geq 0$. $\square$

Verifying closed form solutions – Example 2

Recall the Fibonacci numbers:

$$F(n) = \begin{cases} 1 & \text{if } n = 1 \text{ or } n = 2 \\ F(n-1) + F(n-2) & \text{if } n > 2 \end{cases}$$

Theorem

For $n \geq 1$, the nth Fibonacci number is

$$\mathbf{g(n)} = \frac{\alpha^n - \beta^n}{\alpha - \beta} \quad \text{--- (1)}$$

where

$$\alpha = \frac{1+\sqrt{5}}{2} \text{ and } \beta = \frac{1-\sqrt{5}}{2}. \quad \text{--- (2)}$$

Proof:

It is easy to check that α and β are the solutions to the following equation.

$$x^2 = x + 1$$

Thus we can use this equation as an identity for both α and β. We proceed by induction on n.

$$\alpha^2 = \alpha + 1$$

$$\beta^2 = \beta + 1$$

Proof: (continued)

Base Case: If $n = 1$ or $n = 2$, the recurrence relation says that $F(1) = 1 = F(2)$. The closed-form formula gives

$$F(1) = \frac{\alpha^1 - \beta^1}{\alpha - \beta} = \frac{\alpha - \beta}{\alpha - \beta} = 1$$

and

$$\begin{aligned} F(2) &= \frac{\alpha^2 - \beta^2}{\alpha - \beta} \\ &= \frac{(\alpha + 1) - (\beta + 1)}{\alpha - \beta}, \text{ using the identity} \\ &= \frac{\alpha - \beta}{\alpha - \beta} \\ &= 1 \end{aligned}$$

so the result holds for $n = 1$ and $n = 2$.

Proof: (continued)

Inductive Hypothesis: Let $k > 2$. Suppose as inductive hypothesis that

$$F(i) = \frac{\alpha^i - \beta^i}{\alpha - \beta}$$

for all i such that $1 \leq i < k$.

Part of inductive step

Inductive Step: Using the recurrence relation,

$$\begin{aligned}
F(k) &= F(k-1) + F(k-2) \\
&= \frac{\alpha^{k-1} - \beta^{k-1}}{\alpha - \beta} + \frac{\alpha^{k-2} - \beta^{k-2}}{\alpha - \beta}, \text{ by inductive hypothesis} \\
&= \frac{\alpha^{k-2}(\alpha + 1) - \beta^{k-2}(\beta + 1)}{\alpha - \beta} \\
&= \frac{\alpha^{k-2}(\alpha^2) - \beta^{k-2}(\beta^2)}{\alpha - \beta} = \frac{\alpha^k - \beta^k}{\alpha - \beta}
\end{aligned}$$

as required. □

Quiz 16-1

Which of the following is NOT true?

(a) A recurrence relation is a kind of recursive definition.

(b) A recursive algorithm is a kind of algorithm.

(c) The complexity of an algorithm can be expressed as a recurrence relation.

(d) The complexity of a recursive algorithm can be expressed as a recurrence relation.

(e) A closed form solution of a recurrence relation should be verified for its correctness.