

10. Basics of recurrence relations

Combinatorics

Fall 2026

In earlier units, we counted configurations directly using the sum, product, difference, and division rules, or packaged an entire sequence of counts into an ordinary generating function. For many combinatorial problems, however, an explicit formula is not immediately apparent either because a closed formula does not exist or because it is not obvious how to derive it. Expressing the counts in terms of recurrence relations, when quantities associated to larger values of n depend on quantities for smaller values of n , is a way to help with bookkeeping, implement efficient counting algorithms, and often derive closed formulas for counts. Consider counting the ordered ways to write a positive integer n as a sum of 1s and 2s. For $n = 4$, the possibilities are

$$(1, 1, 1, 1), \quad (1, 1, 2), \quad (1, 2, 1), \quad (2, 1, 1), \quad (2, 2).$$

Because order matters, $(1, 1, 2)$ and $(2, 1, 1)$ are different outcomes. Listing every possibility by hand works for small values of n , but it quickly becomes error-prone for larger values of n .

Instead of constructing every list of size n from scratch, we look for an inductive structure: how can an allowed object of size n be decomposed into smaller objects of the same kind? This perspective leads to a **recurrence relation**, an equation that computes each term of a sequence from earlier terms. In this note, we examine how to formulate recurrences from combinatorial decompositions, analyze their order and algebraic classification, and connect them to ordinary generating functions.

After studying these notes, you will be able to do the following.

- Derive a recurrence relation by decomposing a set of combinatorial objects into mutually exclusive cases.
- Identify the order of a recurrence, specify its required initial conditions, and determine its valid index range.
- Classify a recurrence as linear or nonlinear, constant-coefficient or variable-coefficient, and homogeneous or nonhomogeneous.
- Convert a constant-coefficient linear recurrence into an algebraic equation for its ordinary generating function, accounting for initial conditions.
- Extract the recurrence relation and initial values satisfied by the sequence of coefficients of a rational generating function.
- Model combinatorial structures such as: ordered compositions, pattern-avoiding strings, and restricted walks using recurrences and auxiliary state sequences.

10.1 Decomposing ordered sums into smaller subproblems

Let a_n be the number of ordered lists of parts chosen from $\{1, 2\}$ whose entries sum to n . We include the empty list, written $()$, as the unique list with sum 0, so $a_0 = 1$.

For small values of n , we can find a_n by exhaustive listing:

n	Ordered lists of 1s and 2s with sum n	a_n
0	$()$	1
1	(1)	1
2	$(1, 1), (2)$	2
3	$(1, 1, 1), (1, 2), (2, 1)$	3
4	$(1, 1, 1, 1), (1, 1, 2), (1, 2, 1), (2, 1, 1), (2, 2)$	5

To determine a_n without listing all possibilities, let $n \geq 2$ and classify each list by its final entry. Every list of positive sum has at least one entry, and because the allowed parts are only 1 and 2, every list falls into exactly one of two cases:

- **Case 1: The list ends in 1.** A list in this case has the form $(s_1, \dots, s_{k-1}, 1)$. Removing the trailing 1 leaves $(s_1, \dots, s_{k-1})$, which is an ordered list of 1s and 2s with sum $n - 1$. Conversely, appending 1 to any ordered list with sum $n - 1$ produces a valid list of sum n ending in 1. Because removing the trailing 1 and appending 1 are mutual inverses, lists of sum n ending in 1 are in bijection with lists of sum $n - 1$. There are exactly a_{n-1} such lists.
- **Case 2: The list ends in 2.** A list in this case has the form $(s_1, \dots, s_{k-1}, 2)$. Removing the trailing 2 leaves $(s_1, \dots, s_{k-1})$, which is an ordered list of 1s and 2s with sum $n - 2$. Conversely, appending 2 to any ordered list with sum $n - 2$ produces a valid list of sum n ending in 2. These two operations are mutual inverses, so lists of sum n ending in 2 are in bijection with lists of sum $n - 2$. There are exactly a_{n-2} such lists.

Every valid list belongs to exactly one of these two cases because its final entry is either 1 or 2, never both. The sum rule therefore gives

$$a_n = a_{n-1} + a_{n-2} \quad (n \geq 2).$$

Starting from $a_0 = 1$ and $a_1 = 1$, we can run the recurrence one term at a time:

$$a_2 = 1 + 1 = 2, \quad a_3 = 2 + 1 = 3, \quad a_4 = 3 + 2 = 5, \quad a_5 = 5 + 3 = 8.$$

For $n = 5$, the recurrence organizes the eight lists into two groups. Highlighting the final part shows exactly which earlier list produced each one:

Final part	Ordered lists with sum 5	Count
1	$(1, 1, 1, 1, \mathbf{1}), (1, 1, 2, \mathbf{1}), (1, 2, 1, \mathbf{1}), (2, 1, 1, \mathbf{1}), (2, 2, \mathbf{1})$	$a_4 = 5$
2	$(1, 1, 1, \mathbf{2}), (1, 2, \mathbf{2}), (2, 1, \mathbf{2})$	$a_3 = 3$

Thus $a_5 = a_4 + a_3 = 5 + 3 = 8$. The recurrence provides a bookkeeping rule: reuse the complete lists for sums 4 and 3, append the appropriate final part, and obtain every list of sum 5 exactly once.

Notice why the condition $n \geq 2$ is necessary: if $n = 1$, the only list is (1) , which ends in 1. It cannot end in 2 because $1 - 2 = -1 < 0$, and lists with negative sum do not exist. The recurrence requires two previous values, so the values $a_0 = 1$ and $a_1 = 1$ cannot be produced by the recurrence; they must be provided as initial conditions.

Together with $a_0 = 1$ and $a_1 = 1$, the recurrence determines the entire sequence successively:

n	a_{n-1}	a_{n-2}	$a_n = a_{n-1} + a_{n-2}$
2	1	1	2
3	2	1	3
4	3	2	5
5	5	3	8
6	8	5	13
7	13	8	21

These counts are the Fibonacci numbers shifted by one index: $a_n = F_{n+1}$. The recurrence $a_n = a_{n-1} + a_{n-2}$ encodes the method of subdividing the problem at each step into subproblems associated to prior steps. The same idea applies when more part sizes are allowed.

Example 1 (Ordered sums using parts of size one, two, and three). Let b_n count the number of ordered

lists of parts chosen from $\{1, 2, 3\}$ whose sum is n . The initial values are

$$b_0 = 1, \quad b_1 = 1, \quad b_2 = 2,$$

because the lists for sums 0, 1, and 2 are $()$; (1) ; and $(1, 1), (2)$, respectively.

For $n \geq 3$, every list ends in exactly one of 1, 2, or 3. Removing that final part leaves an ordered list with sum $n - 1$, $n - 2$, or $n - 3$, respectively. These three disjoint cases give

$$b_n = b_{n-1} + b_{n-2} + b_{n-3} \quad (n \geq 3).$$

For example,

$$b_3 = b_2 + b_1 + b_0 = 2 + 1 + 1 = 4,$$

$$b_4 = b_3 + b_2 + b_1 = 4 + 2 + 1 = 7,$$

$$b_5 = b_4 + b_3 + b_2 = 7 + 4 + 2 = 13.$$

The four lists counted by b_3 are $(1, 1, 1), (1, 2), (2, 1), (3)$.

Example 2 (Tilings with multiple tile varieties of the same size). How many ways can a $1 \times n$ board be tiled using squares of length 1, dominoes of length 2, red triominoes of length 3, and blue triominoes of length 3? Let d_n denote the number of tilings.

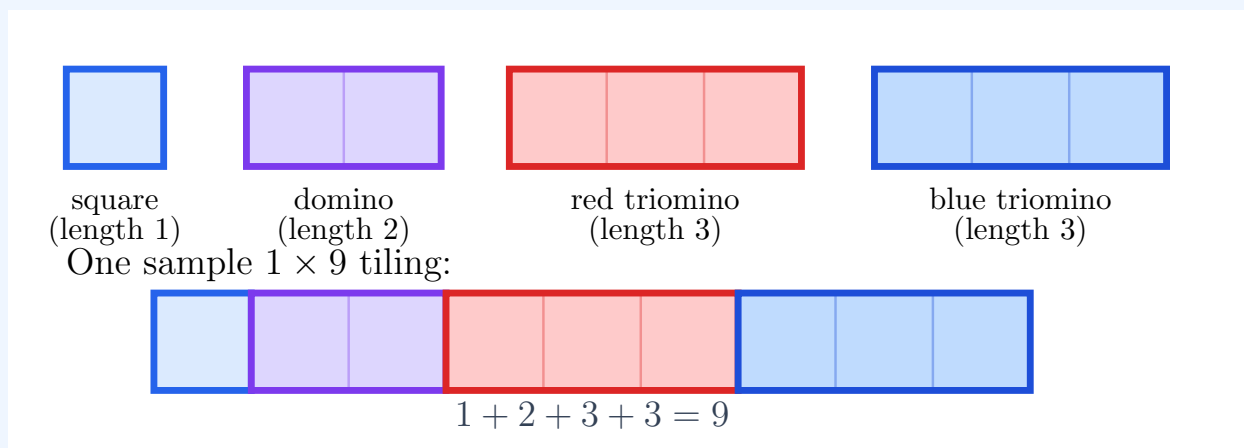


Figure 1: The four available tiles and one sample tiling. The red and blue triominoes have the same length but remain distinct choices.

Figure description: The first row shows a length-one square, a length-two domino, a red length-three triomino, and a blue length-three triomino. The second row shows a length-nine board tiled, from left to right, by one square, one domino, one red triomino, and one blue triomino.

Solution: The initial values are

$$d_0 = 1, \quad d_1 = 1, \quad d_2 = 2.$$

For $n \geq 3$, classify a tiling by its rightmost tile. A 1×1 tile leaves a board of length $n - 1$, a domino leaves one of length $n - 2$, and either color of triomino leaves one of length $n - 3$. Hence

$$d_n = d_{n-1} + d_{n-2} + 2d_{n-3} \quad (n \geq 3).$$

Thus the complete recurrence relation is

$$d_0 = 1, \quad d_1 = 1, \quad d_2 = 2, \quad d_n = d_{n-1} + d_{n-2} + 2d_{n-3} \quad (n \geq 3).$$

10.2 Recurrences, order, and initial conditions

A recurrence relation is a propagation rule for continuing a sequence. By itself, the rule does not identify a unique sequence; starting values are an indispensable part of the definition.

Definition 3 (Recurrence relation, order, and initial conditions). A complete **recurrence relation of order** r for a sequence $(a_n)_{n \geq 0}$ consists of an update rule together with r initial conditions:

$$\begin{cases} a_0 = \alpha_0, a_1 = \alpha_1, \dots, a_{r-1} = \alpha_{r-1}, & \text{initial conditions,} \\ a_n = F(n, a_{n-1}, a_{n-2}, \dots, a_{n-r}), & n \geq r. \end{cases}$$

The update function F must genuinely depend on a_{n-r} ; thus a_{n-r} is the earliest term used and the order is r . The initial conditions start the computation, and the update rule then determines every term $a_r, a_{r+1}, \dots$ uniquely.

The *order* measures how far into the past the recurrence reaches. For instance, the recurrence

$$a_n = a_{n-1} + a_{n-3} \quad (n \geq 3)$$

has order $n - (n - 3) = 3$, even though only two terms appear on the right-hand side. Computing a_3 requires a_2 and a_0 ; thus, three initial values (a_0, a_1, a_2) must be known before the recurrence can run.

Example 4 (The same recurrence with different initial conditions). Consider the order-2 recurrence $s_n = s_{n-1} + s_{n-2}$ for $n \geq 2$. Different initial conditions select entirely different sequences:

- With $a_0 = 1$ and $a_1 = 1$, we obtain the shifted Fibonacci numbers:

$$(a_n)_{n \geq 0} = (1, 1, 2, 3, 5, 8, 13, 21, \dots).$$

- With $b_0 = 2$ and $b_1 = 1$, we obtain the numbers (they have a name: "Lucas numbers"):

$$\begin{aligned} b_2 &= 1 + 2 = 3, \\ b_3 &= 3 + 1 = 4, \\ b_4 &= 4 + 3 = 7, \\ b_5 &= 7 + 4 = 11, \end{aligned} \quad (b_n)_{n \geq 0} = (2, 1, 3, 4, 7, 11, 18, 29, \dots).$$

- With $c_0 = 0$ and $c_1 = 0$, every term is zero: $(c_n)_{n \geq 0} = (0, 0, 0, 0, \dots)$.

The recurrence provides the propagation mechanism, the initial conditions anchor the sequence.

For the recurrences studied in this unit, three structural properties describe the form of the equation.

Definition 5 (Linear recurrence relation). An order- r recurrence relation is **linear** if it can be written as

$$a_n = c_1(n)a_{n-1} + c_2(n)a_{n-2} + \dots + c_r(n)a_{n-r} + g(n) \quad (n \geq r),$$

where the coefficients $c_1(n), \dots, c_r(n)$ and the term $g(n)$ depend only on the index n , and not on any terms of the sequence (a_n) .

- It has **constant coefficients** if each coefficient is a constant number $c_i(n) = c_i$ independent of n , with $c_r \neq 0$.
- It is **homogeneous** if $g(n) = 0$ for all $n \geq r$.

- It is **nonhomogeneous** (or **inhomogeneous**) if $g(n)$ is not identically zero. The function $g(n)$ is called the **driving term** or **inhomogeneous term**.

Example 6 (Classifying recurrence relations). We classify each of the following recurrences by its order, linearity, coefficient type, and homogeneity:

- $a_n = 3a_{n-1} - 2a_{n-2}$: linear, order 2, constant coefficients ($c_1 = 3, c_2 = -2$), homogeneous.
- $a_n = 3a_{n-1} - 2a_{n-2} + n$: linear, order 2, constant coefficients, nonhomogeneous with driving term $g(n) = n$.
- $a_n = na_{n-1} + a_{n-2}$: linear, order 2, variable coefficients (since $c_1(n) = n$ depends on n), homogeneous.
- $a_n = a_{n-1} + a_{n-3}$: linear, order $n - (n - 3) = 3$, constant coefficients ($c_1 = 1, c_2 = 0, c_3 = 1$), homogeneous.
- $a_n = a_{n-1}a_{n-2} + 1$: nonlinear of order 2, because the earlier sequence terms a_{n-1} and a_{n-2} are multiplied together.
- $C_n = \sum_{i=0}^{n-1} C_i C_{n-1-i}$: nonlinear recurrence of variable order, as seen for the Catalan numbers in [Unit 2](#).

Warning (A recurrence must specify initial conditions and an index range). An equation alone does not define a sequence. Consider the relation $a_n = 2a_{n-1} - a_{n-2}$.

- With initial conditions $a_0 = 1$ and $a_1 = 1$, the sequence is constant:

$$a_2 = 2(1) - 1 = 1, \quad a_3 = 2(1) - 1 = 1, \quad \dots \implies a_n = 1 \text{ for all } n \geq 0.$$

- With initial conditions $a_0 = 1$ and $a_1 = 2$, the sequence grows linearly:

$$a_2 = 2(2) - 1 = 3, \quad a_3 = 2(3) - 2 = 4, \quad \dots \implies a_n = n + 1 \text{ for all } n \geq 0.$$

- If only $a_0 = 1$ is specified, the term $a_2 = 2a_1 - 1$ cannot be evaluated because a_1 is undetermined. An order-2 recurrence cannot begin without two initial values.

Similarly, the valid range of indices is essential. Applying $a_n = a_{n-1} + a_{n-2}$ at $n = 1$ for a sequence indexed on $n \geq 0$ would require a_{-1} , an undefined quantity. Always state the initial conditions and the range of n for which the recurrence is valid.

10.3 From a recurrence to an ordinary generating function

Let $A(x) = \sum_{n=0}^{\infty} a_n x^n$ be the ordinary generating function of $(a_n)_{n \geq 0}$. Multiplying $A(x)$ by x^k shifts the sequence of coefficients by k places:

$$x^k A(x) = \sum_{n=0}^{\infty} a_n x^{n+k} = \sum_{n=k}^{\infty} a_{n-k} x^n = a_0 x^k + a_1 x^{k+1} + a_2 x^{k+2} + \dots$$

Because each term $c_k a_{n-k} x^n$ in a constant-coefficient linear recurrence corresponds to a shifted series $c_k x^k A(x)$, we can translate the recurrence into an algebraic equation for $A(x)$.

10.3.1 The procedure for second-order recurrences

To understand how the algebra works, consider an order-2 linear homogeneous recurrence with constant coefficients:

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} \quad (n \geq 2),$$

with initial conditions a_0 and a_1 . We convert this recurrence into an algebraic equation for $A(x)$ using three steps.

1. **Multiply and sum over the valid index range.** Multiply the recurrence by x^n and sum over all indices $n \geq 2$ where the recurrence holds:

$$\sum_{n=2}^{\infty} a_n x^n = c_1 \sum_{n=2}^{\infty} a_{n-1} x^n + c_2 \sum_{n=2}^{\infty} a_{n-2} x^n.$$

2. **Express each sum in terms of $A(x)$.** The sum on the left-hand side is missing the first two terms of $A(x)$:

$$\sum_{n=2}^{\infty} a_n x^n = A(x) - a_0 - a_1 x.$$

For the first sum on the right-hand side, factor out x to align the exponent of x with the index of a :

$$c_1 \sum_{n=2}^{\infty} a_{n-1} x^n = c_1 x \sum_{n=2}^{\infty} a_{n-1} x^{n-1} = c_1 x \sum_{m=1}^{\infty} a_m x^m = c_1 x (A(x) - a_0).$$

For the second sum, factor out x^2 :

$$c_2 \sum_{n=2}^{\infty} a_{n-2} x^n = c_2 x^2 \sum_{n=2}^{\infty} a_{n-2} x^{n-2} = c_2 x^2 \sum_{k=0}^{\infty} a_k x^k = c_2 x^2 A(x).$$

Substituting these three representations back into the summed recurrence gives

$$A(x) - a_0 - a_1 x = c_1 x (A(x) - a_0) + c_2 x^2 A(x).$$

3. **Collect $A(x)$ and solve.** Move all terms containing $A(x)$ to the left-hand side and all initial condition terms to the right-hand side:

$$(1 - c_1 x - c_2 x^2) A(x) = a_0 + (a_1 - c_1 a_0) x.$$

Dividing by $1 - c_1 x - c_2 x^2$ yields the generating function:

$$A(x) = \frac{a_0 + (a_1 - c_1 a_0) x}{1 - c_1 x - c_2 x^2}.$$

Notice the structure of the result:

- The denominator $1 - c_1 x - c_2 x^2$ is determined entirely by the recurrence relation.
- The numerator $a_0 + (a_1 - c_1 a_0) x$ is determined by the initial conditions.

Example 7 (Generating function for ordered sums of ones and twos). Let a_n count the ordered lists of 1s and 2s with sum n , as in [Section 10.1](#). We found

$$a_0 = 1, \quad a_1 = 1, \quad a_n = a_{n-1} + a_{n-2} \quad (n \geq 2).$$

Thus $c_1 = c_2 = 1$. The denominator determined by the recurrence is

$$1 - c_1x - c_2x^2 = 1 - x - x^2,$$

and the numerator determined by the initial conditions is

$$a_0 + (a_1 - c_1a_0)x = 1 + (1 - 1 \cdot 1)x = 1.$$

Therefore, the ordinary generating function is

$$A(x) = \sum_{n=0}^{\infty} a_n x^n = \frac{1}{1 - x - x^2}.$$

In [Unit 2](#), we obtained the same result by building an ordered list from parts with generating polynomial $x + x^2$:

$$\sum_{k=0}^{\infty} (x + x^2)^k = \frac{1}{1 - (x + x^2)} = \frac{1}{1 - x - x^2}.$$

The recurrence strips off the final part, while the geometric series builds the list by appending parts. These complementary decompositions produce the same generating function.

10.3.2 The general order- r theorem

The same three-step procedure applies to linear recurrences of any finite order r , including nonhomogeneous equations.

Theorem 8 (Generating function of a linear recurrence with constant coefficients). Let $(a_n)_{n \geq 0}$ satisfy the order- r linear recurrence

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_r a_{n-r} + g_n \quad (n \geq r),$$

where $c_1, \dots, c_r$ are constants with $c_r \neq 0$. Define

$$A(x) = \sum_{n=0}^{\infty} a_n x^n \quad \text{and} \quad G(x) = \sum_{n=r}^{\infty} g_n x^n.$$

Then

$$(1 - c_1x - c_2x^2 - \cdots - c_r x^r)A(x) = I(x) + G(x),$$

where $I(x)$ is a polynomial of degree at most $r - 1$ determined by the initial conditions:

$$I(x) = \sum_{j=0}^{r-1} \left(a_j - \sum_{i=1}^r c_i a_{j-i} \right) x^j.$$

Consequently,

$$A(x) = \frac{I(x) + G(x)}{1 - c_1x - c_2x^2 - \cdots - c_r x^r}.$$

Proof. Multiply the recurrence by x^n and sum over all indices $n \geq r$:

$$\sum_{n=r}^{\infty} a_n x^n - \sum_{i=1}^r c_i \sum_{n=r}^{\infty} a_{n-i} x^n = \sum_{n=r}^{\infty} g_n x^n = G(x).$$

On the left side, the first sum is

$$\sum_{n=r}^{\infty} a_n x^n = A(x) - \sum_{j=0}^{r-1} a_j x^j.$$

For each $i \in \{1, \dots, r\}$, the shifted sum satisfies

$$\sum_{n=r}^{\infty} a_{n-i} x^n = x^i \sum_{n=r}^{\infty} a_{n-i} x^{n-i} = x^i \left(A(x) - \sum_{k=0}^{r-i-1} a_k x^k \right).$$

Substituting these expressions into the equation gives

$$\left(1 - \sum_{i=1}^r c_i x^i \right) A(x) - \sum_{j=0}^{r-1} a_j x^j + \sum_{i=1}^r c_i x^i \sum_{k=0}^{r-i-1} a_k x^k = G(x).$$

Moving the finite sums of degree less than r to the right side defines $I(x)$:

$$I(x) = \sum_{j=0}^{r-1} a_j x^j - \sum_{i=1}^r \sum_{k=0}^{r-i-1} c_i a_k x^{i+k}.$$

In the double sum, let $j = i + k$. The exponent j ranges from 1 to $r - 1$, and for each fixed j , the index i ranges from 1 to j with $k = j - i$. Grouping by powers of x^j gives the coefficient of x^j as $a_j - \sum_{i=1}^j c_i a_{j-i}$. This establishes the formula for $I(x)$. Dividing by $1 - c_1 x - \dots - c_r x^r$ completes the proof. $\square$

Corollary 9 (Homogeneous linear recurrences have rational generating functions). If the recurrence in [Theorem 8](#) is homogeneous, then $g_n = 0$ for all $n \geq r$, so $G(x) = 0$. In this case,

$$A(x) = \frac{I(x)}{1 - c_1 x - c_2 x^2 - \dots - c_r x^r}$$

is a rational function whose denominator has degree r and constant term 1, and whose numerator $I(x)$ has degree at most $r - 1$ and is determined by the initial data. More generally, whenever the driving series $G(x)$ is a rational function, $A(x)$ is rational.

Warning (Do not omit the numerator correction for initial conditions). A frequent student error when converting an order-2 recurrence $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ to a generating function is to write

$$A(x) \stackrel{\text{wrong}}{=} \frac{a_0}{1 - c_1 x - c_2 x^2}.$$

This expression is correct only in the special circumstance that $a_1 = c_1 a_0$. If $a_1 \neq c_1 a_0$, expanding $\frac{a_0}{1 - c_1 x - c_2 x^2} = a_0 + c_1 a_0 x + \dots$ forces the degree-1 coefficient to be $c_1 a_0$ instead of a_1 . The term $(a_1 - c_1 a_0)x$ in the numerator $I(x) = a_0 + (a_1 - c_1 a_0)x$ is the exact correction required to match the independent initial condition a_1 .

Problem 10 (Applying the theorem to an order-3 recurrence). Let $(a_n)_{n \geq 0}$ satisfy

$$a_n = 2a_{n-1} + a_{n-2} - a_{n-3} \quad (n \geq 3),$$

with $a_0 = 1$, $a_1 = 3$, and $a_2 = 8$. Use [Theorem 8](#) to find $A(x) = \sum_{n=0}^{\infty} a_n x^n$. First determine the denominator from the recurrence. Then write the unknown initial polynomial as $I(x) = u + vx + wx^2$ and determine u, v, w from the initial data.

Solution:

1. **Find the denominator.** Here $c_1 = 2$, $c_2 = 1$, and $c_3 = -1$, so the theorem gives

$$1 - c_1x - c_2x^2 - c_3x^3 = 1 - 2x - x^2 + x^3.$$

2. **Fit the initial polynomial to the initial data.** Because the recurrence is homogeneous, the theorem gives

$$A(x) = \frac{u + vx + wx^2}{1 - 2x - x^2 + x^3}.$$

Equivalently,

$$u + vx + wx^2 = (1 - 2x - x^2 + x^3)(a_0 + a_1x + a_2x^2 + \dots).$$

Using only the coefficients through degree 2,

$$\begin{aligned} u &= a_0 = 1, \\ v &= a_1 - 2a_0 = 3 - 2 = 1, \\ w &= a_2 - 2a_1 - a_0 = 8 - 6 - 1 = 1. \end{aligned}$$

Therefore,

$$A(x) = \frac{1 + x + x^2}{1 - 2x - x^2 + x^3}.$$

10.4 From a rational generating function to a recurrence

We can also read the connection between rational generating functions and recurrences in reverse. Multiplying a rational generating function by its denominator produces a polynomial, whose coefficients must eventually vanish.

Proposition 11 (Rational generating functions determine eventual recurrences). Suppose the ordinary generating function of a sequence $(a_n)_{n \geq 0}$ is a rational function

$$A(x) = \sum_{n=0}^{\infty} a_n x^n = \frac{P(x)}{1 - c_1x - c_2x^2 - \dots - c_r x^r},$$

where $P(x)$ is a polynomial and $c_r \neq 0$. Then for all $n > \deg P$ with $n \geq r$, the sequence satisfies the constant-coefficient linear recurrence

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \dots + c_r a_{n-r}.$$

Proof. Apply [Theorem 8](#) in reverse. Multiplying by the denominator gives

$$(1 - c_1x - c_2x^2 - \dots - c_r x^r)A(x) = P(x).$$

The theorem says that an identity of this form corresponds to the recurrence

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \dots + c_r a_{n-r} + g_n,$$

where $g_n = [x^n]P(x)$. For every $n > \deg P$, we have $g_n = 0$. Hence the recurrence is homogeneous for all $n > \deg P$ with $n \geq r$. $\square$

The condition $n > \deg P$ explains why the recurrence is called *eventual*. If $\deg P < r$, the condition $n > \deg P$ holds automatically for all $n \geq r$, so the recurrence holds from $n = r$ onward without exception. If $\deg P \geq r$, the early coefficients for $r \leq n \leq \deg P$ receive nonzero contributions from $[x^n]P(x)$, behaving like nonhomogeneous driving terms. Once n exceeds the degree of the numerator, the homogeneous recurrence takes over permanently.

Example 12 (Reading a recurrence and initial values from a rational function). Consider the rational generating function

$$B(x) = \frac{1 + 2x}{1 - 3x + 2x^2} = \sum_{n=0}^{\infty} b_n x^n.$$

Determine the recurrence relation, its valid index range, and the initial conditions satisfied by $(b_n)_{n \geq 0}$.

Solution:

1. **Find the recurrence and its valid range.** Write the denominator in the standard form

$$1 - 3x + 2x^2 = 1 - c_1 x - c_2 x^2.$$

Thus $c_1 = 3$, $c_2 = -2$, and the recurrence has order 2. Since the numerator $P(x) = 1 + 2x$ has degree 1, [Proposition 11](#) applies at every $n \geq 2$:

$$b_n = 3b_{n-1} - 2b_{n-2} \quad (n \geq 2).$$

2. **Find the initial conditions.** Use the identity

$$(1 - 3x + 2x^2)(b_0 + b_1 x + b_2 x^2 + \dots) = 1 + 2x.$$

Equating the coefficients of x^0 and x^1 gives

$$b_0 = 1, \quad b_1 - 3b_0 = 2.$$

Therefore $b_0 = 1$ and $b_1 = 5$.

Using the recurrence $b_n = 3b_{n-1} - 2b_{n-2}$ with initial values $b_0 = 1$ and $b_1 = 5$, we compute the next few terms:

$$b_2 = 3b_1 - 2b_0 = 3(5) - 2(1) = 13,$$

$$b_3 = 3b_2 - 2b_1 = 3(13) - 2(5) = 29,$$

$$b_4 = 3b_3 - 2b_2 = 3(29) - 2(13) = 61.$$

As an independent check, we can expand $B(x)$ using partial fractions (as studied in [Unit 2](#)). The denominator factors as $(1 - x)(1 - 2x)$. Setting

$$\frac{1 + 2x}{(1 - x)(1 - 2x)} = \frac{C_1}{1 - x} + \frac{C_2}{1 - 2x},$$

we find $C_1 = -3$ and $C_2 = 4$. Expanding both geometric series gives

$$B(x) = \frac{-3}{1 - x} + \frac{4}{1 - 2x} = -3 \sum_{n=0}^{\infty} x^n + 4 \sum_{n=0}^{\infty} 2^n x^n = \sum_{n=0}^{\infty} (4 \cdot 2^n - 3) x^n.$$

The explicit formula is $b_n = 4 \cdot 2^n - 3$. Evaluating this formula for small n gives

$$b_0 = 4(1) - 3 = 1, \quad b_1 = 4(2) - 3 = 5, \quad b_2 = 4(4) - 3 = 13, \quad b_3 = 4(8) - 3 = 29, \quad b_4 = 4(16) - 3 = 61,$$

confirming the recurrence computation at every step.

10.5 Pattern-avoiding strings

A string over an alphabet Σ **avoids** a pattern w if w does not appear as a contiguous substring in any position. For instance, the binary string 0010 contains 00 in its first two positions and therefore does not avoid 00.

Pattern avoidance is a global condition, not merely a restriction on how a string ends. The string 0011 ends in 11, but it contains 00 at its start, so it fails to avoid 00.

10.5.1 Binary strings avoiding repeated zeroes

Let a_n count the number of length- n binary strings over $\{0, 1\}$ that avoid 00.

- For $n = 0$, the empty string contains no characters and vacuously avoids 00, so $a_0 = 1$.
- For $n = 1$, both length-1 strings 0 and 1 avoid 00, so $a_1 = 2$.

For $n \geq 2$, every valid string ends in either 1 or 0.

- **Case 1: The string ends in 1.** A string in this case has the form $s_1 \dots s_{n-1} 1$. Because the added final character is 1, it cannot complete a forbidden block 00. Therefore, $s_1 \dots s_{n-1} 1$ avoids 00 if and only if the prefix $s_1 \dots s_{n-1}$ avoids 00. There are a_{n-1} such strings.
- **Case 2: The string ends in 0.** Because $n \geq 2$ and the string avoids 00, the character immediately preceding the final 0 cannot be 0. It must be 1. Thus, every valid string ending in 0 actually ends in the two-character block 10:

$$s_1 \dots s_{n-2} 10.$$

Because the block 10 begins with 1, appending it to any valid string of length $n - 2$ cannot create 00. Therefore, $s_1 \dots s_{n-2} 10$ avoids 00 if and only if the prefix $s_1 \dots s_{n-2}$ avoids 00. There are a_{n-2} such strings.

Because every valid string ends in either 1 or 0, the two cases partition the set of valid strings:

$$a_n = a_{n-1} + a_{n-2} \quad (n \geq 2).$$

With $a_0 = 1$ and $a_1 = 2$, the sequence begins:

$$a_2 = 2 + 1 = 3, \quad a_3 = 3 + 2 = 5, \quad a_4 = 5 + 3 = 8, \quad a_5 = 8 + 5 = 13.$$

For $n = 2$, among the $2^2 = 4$ binary strings, only 00 is forbidden, leaving 01, 10, 11, which confirms $a_2 = 3$. Using our generating function formula for order-2 recurrences:

$$A(x) = \frac{a_0 + (a_1 - c_1 a_0)x}{1 - c_1 x - c_2 x^2} = \frac{1 + (2 - 1 \cdot 1)x}{1 - x - x^2} = \frac{1 + x}{1 - x - x^2}.$$

Compare this with the ordered-sum generating function $\frac{1}{1-x-x^2}$ from [Example 7](#). The recurrence relation $a_n = a_{n-1} + a_{n-2}$ is identical, but the initial conditions differ ($a_1 = 2$ instead of $a_1 = 1$). That difference in initial conditions produces the numerator $1 + x$ rather than 1.

Example 13 (Binary strings avoiding three consecutive zeroes). Let b_n count the number of length- n binary strings that avoid 000. For $n \geq 3$, every valid string ends in exactly one of the blocks:

$$1, \quad 10, \quad 100.$$

Removing the trailing block leaves an arbitrary valid prefix of length $n - 1$, $n - 2$, or $n - 3$, respectively.

Therefore,

$$b_n = b_{n-1} + b_{n-2} + b_{n-3} \quad (n \geq 3).$$

All binary strings of lengths 0, 1, and 2 avoid 000, so

$$b_0 = 1, \quad b_1 = 2, \quad b_2 = 4.$$

The initial polynomial is

$$I(x) = b_0 + (b_1 - b_0)x + (b_2 - b_1 - b_0)x^2 = 1 + (2 - 1)x + (4 - 2 - 1)x^2 = 1 + x + x^2.$$

Hence,

$$B(x) = \frac{1 + x + x^2}{1 - x - x^2 - x^3}.$$

10.5.2 A nonhomogeneous recurrence: avoiding one-zero

Let c_n count length- n binary strings that avoid 10. In any valid string, once a 1 appears, every subsequent character must also be 1; a 0 appearing anywhere after a 1 would create a 10 transition. Every valid string therefore consists of k zeroes followed by $n - k$ ones:

$$\underbrace{00 \cdots 0}_{k \text{ zeroes}} \underbrace{11 \cdots 1}_{n-k \text{ ones}}$$

for some integer $k \in \{0, 1, \dots, n\}$. Because there are $n + 1$ choices for k , there are exactly $c_n = n + 1$ valid strings.

We can also derive this result by setting up a recurrence based on the final character: For $n \geq 1$:

- If the string ends in 1, removing that 1 leaves an arbitrary valid string of length $n - 1$. (Appending 1 can never create 10). This yields c_{n-1} strings.
- If the string ends in 0, it cannot contain a 1 in any earlier position, because any earlier 1 followed eventually by this final 0 would require at least one 10 transition. Thus, all preceding $n - 1$ characters must be 0. This yields exactly 1 string: $00 \cdots 0$.

Because these two cases are disjoint and cover all valid strings, we obtain

$$c_n = c_{n-1} + 1 \quad (n \geq 1),$$

with $c_0 = 1$. This is a nonhomogeneous linear recurrence with constant coefficient $c_1 = 1$ and constant driving term $g(n) = 1$.

We convert it to a generating function using [Theorem 8](#). Multiply by x^n and sum for $n \geq 1$:

$$\sum_{n=1}^{\infty} c_n x^n = \sum_{n=1}^{\infty} c_{n-1} x^n + \sum_{n=1}^{\infty} x^n.$$

Expressing each sum in terms of $C(x) = \sum_{n=0}^{\infty} c_n x^n$:

$$C(x) - c_0 = xC(x) + \frac{x}{1 - x}.$$

Substituting $c_0 = 1$ and rearranging:

$$(1 - x)C(x) = 1 + \frac{x}{1 - x} = \frac{1}{1 - x}.$$

Dividing by $1 - x$:

$$C(x) = \frac{1}{(1-x)^2}.$$

Extracting the coefficient using the negative binomial expansion from Unit 2:

$$[x^n] \frac{1}{(1-x)^2} = \binom{n+2-1}{n} = \binom{n+1}{1} = n+1,$$

which matches the direct count.

10.5.3 Alphabet size and subtraction techniques

When the alphabet contains more than two symbols, the number of choices for allowed final characters scales the recurrence coefficients.

Example 14 (Ternary strings avoiding repeated zeroes). Let p_n count the number of length- n strings over $\{0, 1, 2\}$ that avoid 00. For $n = 0$, $p_0 = 1$. For $n = 1$, all three characters are valid, so $p_1 = 3$. For $n \geq 2$, classify a valid string by its final symbol:

- If it ends in 1 or 2: the preceding prefix can be any valid string of length $n - 1$. There are 2 choices for the final symbol, contributing $2p_{n-1}$ strings.
- If it ends in 0: the preceding symbol cannot be 0, so it must be 1 or 2. The string ends in 10 or 20. The prefix of length $n - 2$ can be any valid string. There are 2 choices for the block, contributing $2p_{n-2}$ strings.

Combining the cases:

$$p_n = 2p_{n-1} + 2p_{n-2} \quad (n \geq 2).$$

With $p_0 = 1$ and $p_1 = 3$, we compute $p_2 = 2(3) + 2(1) = 8$. (Among all $3^2 = 9$ strings of length 2, only 00 is forbidden). The generating function is

$$P(x) = \frac{p_0 + (p_1 - 2p_0)x}{1 - 2x - 2x^2} = \frac{1 + (3 - 2)x}{1 - 2x - 2x^2} = \frac{1 + x}{1 - 2x - 2x^2}.$$

The difference rule from Unit 1 is particularly effective when an allowed ending is described by subtracting an exception from all valid prefixes.

Example 15 (Ternary strings avoiding one-zero via subtraction). Let q_n count length- n strings over $\{0, 1, 2\}$ that avoid 10. For $n = 0$, $q_0 = 1$. For $n = 1$, $q_1 = 3$. For $n \geq 2$, consider the final symbol:

- If the final symbol is 1 or 2: appending 1 or 2 to any valid length- $(n - 1)$ string can never create 10. This produces $2q_{n-1}$ valid strings.
- If the final symbol is 0: we may append 0 to any valid length- $(n - 1)$ prefix *except* those that end in 1. How many valid length- $(n - 1)$ strings end in 1? A valid string of length $n - 1$ ending in 1 is formed uniquely by appending 1 to an arbitrary valid string of length $n - 2$. (Appending 1 never introduces 10). Thus, exactly q_{n-2} of the q_{n-1} valid prefixes end in 1. By the difference rule, the number of valid prefixes that do not end in 1 is $q_{n-1} - q_{n-2}$.

Summing the allowed endings:

$$q_n = 2q_{n-1} + (q_{n-1} - q_{n-2}) = 3q_{n-1} - q_{n-2} \quad (n \geq 2).$$

With $q_0 = 1$ and $q_1 = 3$:

$$\begin{aligned} q_2 &= 3(3) - 1 = 8, \\ q_3 &= 3(8) - 3 = 21. \end{aligned}$$

For $n = 2$, among the $3^2 = 9$ strings, only 10 is forbidden, leaving $9 - 1 = 8$. For $n = 3$, among the $3^3 = 27$ strings, exactly 6 contain 10 (100, 101, 102, 010, 110, 210), leaving $27 - 6 = 21$.

The generating function has denominator $1 - 3x + x^2$ and numerator

$$q_0 + (q_1 - 3q_0)x = 1 + (3 - 3)x = 1.$$

Therefore,

$$Q(x) = \frac{1}{1 - 3x + x^2}.$$

Subtraction avoids the need to divide the prefixes into separate positive cases based on their endings.

10.6 Walks with forbidden reversals: a lucky simplification

Suppose a robot moves on the integer lattice $\mathbb{Z}^2$. At each second it takes one of three steps:

$$U = (0, 1) \quad (\text{up}), \quad L = (-1, 0) \quad (\text{left}), \quad R = (1, 0) \quad (\text{right}).$$

To prevent the robot from wasting a move, it may not reverse horizontal direction immediately: the consecutive blocks LR and RL are forbidden. Let t_n be the number of valid step sequences of length n . The empty walk gives $t_0 = 1$, and the three one-step walks give $t_1 = 3$.

To extend a valid walk of length $n - 1$, the number of choices depends on its final step:

- If it ends in U, all three next steps are allowed.
- If it ends in L or R, exactly two next steps are allowed.

Let u_{n-1} be the number of valid length- $(n - 1)$ walks ending in U. The remaining $t_{n-1} - u_{n-1}$ walks end horizontally. Counting all valid one-step extensions gives

$$t_n = 3u_{n-1} + 2(t_{n-1} - u_{n-1}) = 2t_{n-1} + u_{n-1}.$$

We can evaluate the exceptional count u_{n-1} without creating a system of recurrences. Appending U to any valid walk of length $n - 2$ produces a unique valid walk of length $n - 1$ ending in U, and removing the final U reverses this operation. Therefore,

$$u_{n-1} = t_{n-2}.$$

Substitution gives the scalar recurrence

$$t_0 = 1, \quad t_1 = 3, \quad t_n = 2t_{n-1} + t_{n-2} \quad (n \geq 2).$$

The sequence begins

$$\begin{aligned} t_2 &= 2(3) + 1 = 7, \\ t_3 &= 2(7) + 3 = 17, \\ t_4 &= 2(17) + 7 = 41. \end{aligned}$$

For $n = 2$, among the $3^2 = 9$ two-step walks, only LR and RL are forbidden, confirming $t_2 = 9 - 2 = 7$. The generating function has denominator $1 - 2x - x^2$ and numerator

$$t_0 + (t_1 - 2t_0)x = 1 + (3 - 2)x = 1 + x.$$

Therefore,

$$T(x) = \frac{1+x}{1-2x-x^2}.$$

Here we were lucky: the walks receiving three extensions—those ending in $\mathbf{U}$ —are in immediate bijection with all valid walks one step shorter. For more complicated local restrictions, each ending type may have its own continuation rules, and no exceptional count collapses so neatly to an earlier total. We will need a reliable method, based on state variables and matrix recurrence relations, to model that more difficult logic.

10.7 Boundary choices: decomposing from the start versus the end

In the examples developed so far, we decomposed objects by examining their **final** element—the last part of a sum, the trailing symbol of a string, or the final step of a walk. Decomposing by the **first** element is equally valid and can sometimes simplify the argument.

10.7.1 Initial character decomposition in pattern avoidance

Consider again the sequence c_n of length- n binary strings that avoid 10 from [Section 10.5.2](#). When we decomposed by the final character, strings ending in 0 required an argument showing that all previous symbols had to be 0.

Let us instead classify a valid string of length $n \geq 1$ by its **first** character:

- **Case 1: The string begins with 0.** Prepending 0 to any valid string of length $n - 1$ cannot create 10, because the newly added character is 0. Therefore, any valid string of length $n - 1$ can serve as the suffix. This yields c_{n-1} strings.
- **Case 2: The string begins with 1.** If the string begins with 1, it can never contain a 0 in any subsequent position, because a 0 anywhere after this leading 1 would create a 10 transition. Thus, all remaining $n - 1$ characters must be 1. This yields exactly 1 string: $11 \cdots 1$.

Because every valid string begins with either 0 or 1, the sum rule immediately reproduces

$$c_n = c_{n-1} + 1 \quad (n \geq 1).$$

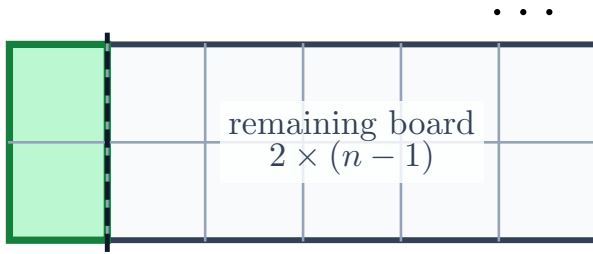
Looking at the start of the string makes the uniqueness of the second case immediate.

10.7.2 A geometric model: tiling a grid with dominoes

Decomposing from a boundary is also the fundamental tool for counting tilings of two-dimensional regions.

Example 16 (Tiling a $2 \times n$ board with dominoes). How many ways are there to tile a $2 \times n$ board using 1×2 dominoes? Dominoes may be placed horizontally (1×2) or vertically (2×1).

Case 1: one vertical domino



Case 2: two horizontal dominoes

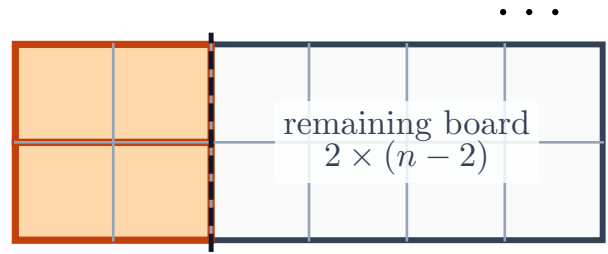


Figure 2: The top-left square is covered either by one vertical domino or by a horizontal domino that forces a second horizontal domino below it. The two cases leave smaller complete rectangles.

Figure description: Two schematic two-row boards. In the first, one vertical domino fills the first column and leaves a two-by-(n minus 1) rectangle. In the second, two horizontal dominoes fill the first two columns and leave a two-by-(n minus 2) rectangle.

Solution: Let d_n be the number of valid tilings of a $2 \times n$ board. For $n = 0$, the 2×0 board is empty and has 1 tiling, so $d_0 = 1$. For $n = 1$, the 2×1 board can only be covered by a single vertical domino, so $d_1 = 1$.

For $n \geq 2$, focus on the top-left square of the board (column 1, row 1). In any valid tiling, this square must be covered by either a vertical domino or a horizontal domino:

- **Case 1: The top-left square is covered by a vertical domino.** A vertical domino occupies both squares of column 1 (row 1 and row 2). Removing this domino leaves an uncovered $2 \times (n-1)$ board. Conversely, placing a vertical domino in column 1 next to any valid tiling of a $2 \times (n-1)$ board produces a valid tiling of the $2 \times n$ board. Thus, there are d_{n-1} tilings in this case.
- **Case 2: The top-left square is covered by a horizontal domino.** A horizontal domino occupies the top squares of columns 1 and 2. Now consider the bottom square of column 1 (row 2, column 1). It cannot be covered by a vertical domino, because the square above it is already occupied. It cannot be left empty. Therefore, it must be covered by another horizontal domino, which occupies the bottom squares of columns 1 and 2. Together, these two horizontal dominoes completely fill columns 1 and 2, forming a 2×2 block. Removing both horizontal dominoes leaves an uncovered $2 \times (n-2)$ board. Conversely, placing two horizontal dominoes in the first two columns next to any valid tiling of a $2 \times (n-2)$ board produces a valid tiling of the $2 \times n$ board. Thus, there are d_{n-2} tilings in this case.

Every valid tiling covers the top-left square in exactly one of these two ways, and the two cases leave independent sub-boards. The sum rule therefore gives

$$d_n = d_{n-1} + d_{n-2} \quad (n \geq 2).$$

With $d_0 = 1$ and $d_1 = 1$, this is again the Fibonacci recurrence:

$$(d_n)_{n \geq 0} = (1, 1, 2, 3, 5, 8, 13, 21, \dots).$$

The generating function is

$$D(x) = \frac{1}{1 - x - x^2}.$$

The domino tiling problem demonstrates that whether objects are represented as algebraic lists, character

strings, or geometric tiles, the underlying combinatorial technique is identical:

1. Isolate an indispensable boundary element (such as the final part, first symbol, or a fixed corner square).
2. Classify the mutually exclusive ways to cover or choose that boundary element.
3. Establish bijections between each case and smaller instances of the original problem to determine the recurrence relation and initial conditions.

10.8 Two next questions

Now that we covered the basics of recurrence relations, we still need to understand two questions.

1. **How do we solve the recurrence?**
2. **How do we model more complex recurrence problems?**

Problem 17 (A recurrence that is difficult to guess). Let f_n be the number of tilings of a $3 \times n$ board by 1×2 dominoes. Try to find a recurrence relation for (f_n) by examining the tiles that meet the left boundary.

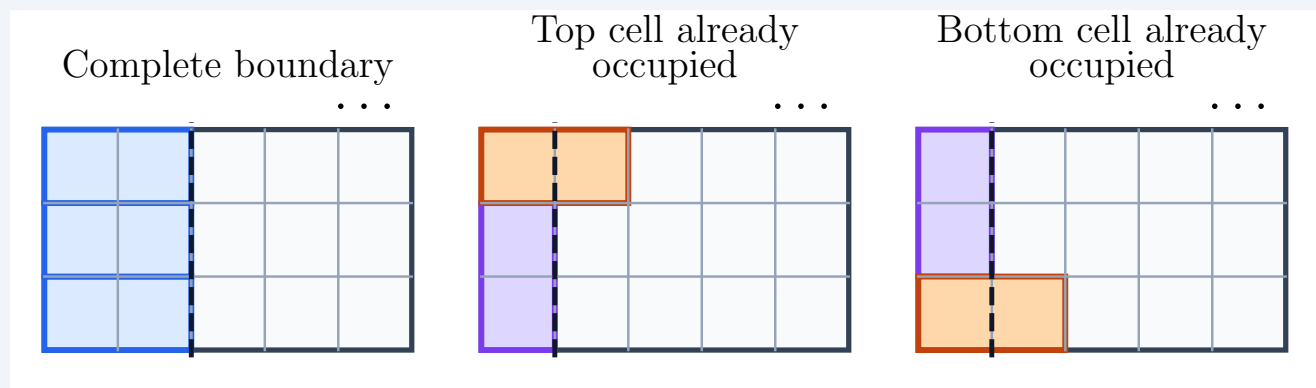


Figure 3: A cut through a $3 \times n$ tiling need not leave a smaller rectangle. A horizontal domino crossing the cut can leave the top or bottom square of the next column already occupied.

Figure description: Three schematic three-row boards. The first has a straight cut after two completely tiled columns and leaves a smaller rectangle. The second has a horizontal domino crossing the cut in the top row, so the top square beyond the cut is occupied. The third is the reflected case with the bottom square beyond the cut occupied.

The pictures show the obstacle. After removing tiles near the left boundary, the remaining region may be a smaller complete rectangle, but it may instead have a top or bottom boundary square already occupied by a horizontal domino. The number f_k counts only complete $3 \times k$ rectangles, so it does not distinguish these partial boundary shapes.

Consequently, a recurrence involving only the totals f_n is difficult to derive directly or guess reliably. A systematic solution must record which boundary squares are already occupied, assign a count to each possible boundary state, and track how those states change when a column is added. Matrix recurrence relations will organize those simultaneous transitions.

Problem 18 (Overlapping forbidden substrings). Let w_n be the number of length- n strings over the alphabet $\{a, b, c, d\}$ that avoid all three substrings

abca, bcab, cabc.

Find a recurrence relation and initial conditions for (w_n) .

This recurrence is difficult to guess from the totals $w_0, w_1, w_2, \dots$ alone. Whether a new letter is legal depends on the current suffix: for example, suffixes **abc**, **bca**, and **cab** forbid different next letters. These suffixes also overlap—the ending of one nearly forbidden pattern can be the beginning of another. A reliable model therefore needs states recording the longest relevant suffix of the current string, rather than only the total number of valid strings.

10.9 External References

10.9.1 Additional material (optional)

- [Discrete Math II - 8.1.1 Applications of Recurrence Relations](#)
- [Discrete Math II - 8.1.2 Modeling with Recurrence Relations](#)
- [Counting Rocks! - Introduction to Recurrence Relations](#)
- [TrevTutor - Recurrence Relations: The Basics](#)
- [Discrete Math II - 8.4.2 Generating Functions for Recurrences](#)