Find A Recurrence Relation For The Number Of N Digit Ternary Sequences That Have Pattern 012 Occurring

Find A Recurrence Relation For The Number Of N Digit Ternary Sequences That Have Pattern 012 Occurring

Introduction

Understanding the combinatorial structure of sequences is a fundamental aspect of discrete mathematics and theoretical computer science. In particular, analyzing sequences over finite alphabets and counting those that contain specific patterns is a classic problem with applications in pattern matching, coding theory, and bioinformatics. This article focuses on ternary sequences—sequences composed of the symbols {0, 1, 2}—and aims to derive a recurrence relation for the number of such sequences of length n that contain the pattern "012" at least once.

Problem Statement and Objectives

Given an integer n, we seek to determine the number T_n of sequences of length n over the ternary alphabet {0, 1, 2} that contain the substring "012" at least once. Our goal is to find a recurrence relation that expresses T_n in terms of the counts for smaller sequence lengths, ideally with initial conditions, thereby enabling efficient computation.

Preliminary Concepts and Definitions

Sequences and Patterns

- A ternary sequence of length n is an ordered list $s_1 s_2 \cdots s_n$, where each $s_i \in \{0, 1, 2\}$.
- The pattern "012" is a contiguous substring consisting of the symbols 0, followed by 1, followed by 2.

Counting Sequences Based on Pattern Occurrence

- Let T_n be the total number of sequences of length n containing at least one

occurrence of "012".
- Alternatively, define S_n as the total number of sequences of length n over {0, 1, 2}, which is 3^n.

Our task involves focusing on the subset of sequences that contain "012" at least once, and devising a recurrence relation to count T_n efficiently.

Approach: Complementary Counting and State Machine Modeling

One effective method for such problems involves considering the complement: counting sequences that do not contain the pattern "012", and then subtracting from the total.

$$
T_n = 3^n - N_n,
$$
where N_n is the number of sequences of length n that avoid "012".

This approach simplifies the problem to counting sequences with forbidden pattern avoidance, which is often more manageable with state machine modeling.

Modeling Pattern Avoidance Using Finite Automata

To count sequences avoiding "012", we can model the process via a finite automaton (FA) that tracks the progress towards forming the pattern.

Constructing the Automaton

The automaton has states representing how much of "012" has been matched so far:

- State S0: No partial match; sequence does not currently match any prefix of "012".
- State S1: Last symbol was '0', indicating the start of the pattern.
- State S2: Last two symbols were '0' followed by '1', i.e., the prefix "01" has been matched, and the next symbol could complete the pattern.
- Forbidden State (S3): The pattern "012" has been matched; sequences reaching this are excluded from count of avoiding sequences.

Our goal is to count sequences that never reach S3.

Transition Rules

- From S0:
- On input '0': move to S1.
- On input '1' or '2': stay in S0.

- From S1:
- On '1': move to S2.
- On '0': stay in S1 (since '0' may start a new pattern).
- On '2': move back to S0 (pattern not progressing).

- From S2:
- On '2': move to S3 (pattern "012" found; sequences to be excluded).
- On '0': move to S1 (since '0' could start a new pattern).
- On '1': stay in S2 (partial pattern "01" remains).

Sequences that avoid "012" are those that never reach S3.

Defining Recurrence Relations for Avoidance Counts

Let:

- a_n = number of sequences of length n that avoid "012" and end in state S0.
- b_n = number of sequences of length n that avoid "012" and end in state S1.
- c_n = number of sequences of length n that avoid "012" and end in state S2.

Since sequences that reach S3 are forbidden, they are excluded from these counts.

The total number of avoiding sequences of length n:

$$
N_n = a_n + b_n + c_n.
$$

Recurrence Relations

Based on the transition rules:

- For sequences ending in S0:

$$
a_{n} = a_{n-1} \times 1 + b_{n-1} \times 1 + c_{n-1} \times 0,
$$

but considering the transition rules:

- From S0:
- '0' keeps us in S1 (so contributes to b_{n})
- '1' or '2' keep us in S0 (contribute to a_{n})

Similarly, for a_{n}:

$$
a_{n} = a_{n-1} \times 2 + b_{n-1} \times 1,
$$
since:

- From S0, choosing '1' or '2' (2 options) keeps in S0.
- From S1, choosing '2' (1 option) moves back to S0.

Actually, to be precise:

- a_n counts sequences ending in S0:

$$
a_n = a_{n-1} \times 2 + b_{n-1} \times 1,
$$
because:

- Staying in S0: choosing '1' or '2' from sequences ending in S0.
- From S1, choosing '2' goes to S0.

Similarly, for b_n:

- From sequences ending in S0, choosing '0' moves to S1:

$$
b_n = a_{n-1} \times 1 + b_{n-1} \times 0 + c_{n-1} \times 0,
$$
but since only sequences ending in S0 can move to S1:

$$
b_n = a_{n-1}.
$$

For c_n:

- From S1, choosing '1' remains in S2:

$$
c_n = b_{n-1} \times 1,
$$
- From S2, choosing '0' moves to S1, which is already accounted for; choosing '2' would reach forbidden state (excluded).

Thus, the recursions are:

$$
a_n = 2 a_{n-1} + b_{n-1},
$$
$$
b_n = a_{n-1},
$$
$$
c_n = b_{n-1}.
$$

Initially, for $n=0$ (empty sequence):

$$
a_0 = 1, \quad b_0=0, \quad c_0=0,
$$
since the empty sequence is in S0.

Deriving the Recurrence for N_n

Using the relations:

$$
b_n = a_{n-1},
$$
$$
c_n = b_{n-1} = a_{n-2}.
$$

Substitute $b_{n-1} = a_{n-2}$:

$$
a_n = 2 a_{n-1} + a_{n-2}.
$$

This is a second-order linear recurrence for a_n.

Since total avoiding sequences:

$$
N_n = a_n + b_n + c_n,
$$
and

$$
b_n = a_{n-1}, \quad c_n = a_{n-2},
$$

we have:

$$
N_n = a_n + a_{n-1} + a_{n-2}.
$$

Given the recurrence:

$$
a_n = 2 a_{n-1} + a_{n-2},
$$

we can compute a_n, then N_n.

Computing Initial Conditions and Solving the Recursion

Initial conditions:

- For $n=0$:

$$
a_0=1, \quad b_0=0, \quad c_0=0,
$$

so

$$
N_0 = a_0 + b_0 + c_0=1+0+0=1,
$$
which corresponds to the empty sequence.

For $n=1$:

- $a_1 = 2 a_0 + a_{-1}$. Since a_{-1} is undefined

Frequently Asked Questions

How can I define a recurrence relation for the number of n-digit ternary sequences containing the pattern '012'?

You can define the recurrence by considering whether the sequence ends with '012' or not, and tracking states based on partial matches. Typically, this involves setting up state variables for sequences that have matched parts of '012' and deriving relations based on appending digits while avoiding or creating the pattern.

What initial conditions are necessary for the recurrence relation counting sequences with '012'?

Initial conditions include the counts for small sequence lengths, such as for n=1 and n=2, where the pattern '012' cannot occur, so the counts are based on sequences without '012'. For example, for n=1 and n=2, all sequences are valid, so counts are 3 and 9 respectively, depending on the specific approach.

How do I handle sequences that already contain '012' when constructing the recurrence?

Once a sequence contains '012', all extensions will also contain the pattern. To manage this, introduce a state variable indicating whether '012' has occurred, and once it does, count all subsequent sequences as part of the total. The recurrence then accumulates counts based on whether the pattern has been formed or not.

Can the recurrence relation be simplified to a closed-form expression for the total number of sequences?

In general, recurrence relations for pattern-based sequences can be complex, but with proper state definitions, they can sometimes be solved to obtain closed-form formulas, often involving linear recurrence solutions or generating functions. However, the specific pattern '012' may require detailed algebraic work to derive such closed forms.

What is the significance of tracking partial matches in the recurrence relation for sequences containing '012'?

Tracking partial matches allows for a systematic way to count sequences that are close to forming the pattern or have just formed it. This approach helps in building the recurrence by understanding how appending each new digit affects the potential to complete or avoid the pattern, thereby enabling precise count calculations.

Are there existing formulas or algorithms for efficiently counting ternary sequences with a specific pattern like '012'?

Yes, pattern-avoidance and pattern-incorporation problems are well-studied in combinatorics. Algorithms involving automata, generating functions, or dynamic programming are often used to efficiently compute the number of sequences with or without specific patterns, including for ternary sequences containing '012'.

Additional Resources

Find A Recurrence Relation For The Number Of N Digit Ternary Sequences That Have Pattern 012 Occurring

When analyzing sequences over a ternary alphabet (digits 0, 1, 2), a common problem involves counting the number of sequences of a given length that contain a specific pattern—in this case, the pattern "012". Such combinatorial problems are fundamental in the study of pattern avoidance and pattern occurrence within sequences, with applications spanning computer science, coding theory, and combinatorics. The challenge lies in deriving a recurrence relation that succinctly relates the count of valid sequences of length n to those of smaller lengths, thereby enabling efficient computation and deeper understanding of the sequence structure.

Understanding the Problem: Ternary Sequences and Pattern 012

Before delving into the recurrence relation, it's essential to understand the basic elements of the problem:

- Ternary sequences: These are sequences composed of three symbols {0, 1, 2}. For a fixed length n, there are 3^n total sequences.
- Pattern 012: A contiguous substring "012" within a sequence. The problem asks for the number of sequences of length n that contain at least one occurrence of "012".
- Counting goal: Determine $f(n)$, the number of sequences of length n over {0,1,2} that have the pattern "012" at least once.

This problem is a classic example of pattern occurrence counting in sequences, which can be approached via various combinatorial techniques, including direct counting, inclusion-exclusion, and recursive methods.

Approach Overview: Recursion and State Modeling

To find the recurrence relation for $f(n)$, a common approach is to model the problem using states that encode the "progress" towards forming the pattern "012".

Key idea:
Define a set of states representing how much of the pattern "012" has been matched so far at the end of the sequence. The sequence can be viewed as evolving through these states as new symbols are appended.

States Definition

Let's define the following states:

- State S0: No partial match of "012" at the end of the sequence.
- State S1: The last symbol matches the first symbol of the pattern ("0"), indicating a partial match "0".
- State S2: The last two symbols are "01", indicating a partial match "01".
- State S3: The pattern "012" has been found at least once; once in this state, the pattern is already present, and all subsequent sequences are considered "success" sequences.

Our goal is to count sequences that reach State S3 at any point, with the initial sequence starting at length 0.

Formulating the Recurrence Relation

Using the states, we can define recurrence relations based on how sequences extend from length $n-1$ to length n.

Let:

- $A(n)$: Number of sequences of length n ending in state S0.
- $B(n)$: Number of sequences of length n ending in state S1.
- $C(n)$: Number of sequences of length n ending in state S2.
- $D(n)$: Number of sequences of length n that have already contained "012" (state S3, absorbing state).

Since once the pattern is found, all subsequent sequences remain in that "success" state, $D(n)$ accumulates all sequences containing "012".

Initial Conditions:

- At $n=0$:
$A(0) = 1$ (the empty sequence, no partial match)
$B(0) = 0$
$C(0) = 0$
$D(0) = 0$

Transition Equations

At each step, for each state, appending a symbol from {0, 1, 2} leads to a new state:

- From S0 (no partial match):
- Append 0 → S1 (matching first symbol "0")
- Append 1 or 2 → stay in S0 (no progress towards pattern)

- From S1 ("0" matched):
- Append 1 → S2 ("01") partial match
- Append 0 or 2 → stay in S1 or revert to S0 depending on the symbol

But since the pattern is "012", only appending 1 advances the pattern; appending 0 or 2 resets the partial match unless carefully considered.

- From S2 ("01" matched):
- Append 2 → Pattern "012" found → D(n) sequences (success state)
- Append 0 or 1 → partial match resets to S1 or S0 depending on the symbol

- Once in D, remain in this success state for subsequent steps.

Simplified transition equations:

$$
\begin{cases}
A(n) = 2A(n-1) + 2B(n-1) \\
B(n) = A(n-1) \\
C(n) = B(n-1) \\
D(n) = D(n-1) + C(n-1)
\end{cases}
$$

But, more precisely, to account for the pattern "012," and transitions:

$$
\begin{cases}
A(n) = 2A(n-1) + 2B(n-1) \quad \text{(appending 1 or 2)} \\
B(n) = A(n-1) \quad \text{(appending 0)} \\
C(n) = B(n-1) \quad \text{(appending 1 to "0" gives "01")} \\
D(n) = D(n-1) + C(n-1) \quad \text{(pattern "012" found when appending 2 to "01")}
\end{cases}
$$

Note that sequences in success state $D(n)$ accumulate all sequences that have already contained "012".

Deriving the Recurrence for $f(n)$

Our ultimate goal is $f(n)$, the total number of sequences of length n that contain "012". This can be expressed as:

$$
f(n) = D(n) + \text{sequences that reach success at length } n
$$

Given the relations above, and initial conditions:

- $A(0) = 1$, $B(0) = 0$, $C(0) = 0$, $D(0) = 0$.

From the transition equations:

$$
\begin{cases}
A(n) = 2A(n-1) + 2B(n-1) \\
B(n) = A(n-1) \\
C(n) = B(n-1) \\
D(n) = D(n-1) + C(n-1)
\end{cases}
$$

Substituting $B(n)$ and $C(n)$:

$$
A(n) = 2A(n-1) + 2A(n-2)
$$
(since $B(n-1) = A(n-2)$ and $C(n-1) = B(n-2) = A(n-3)$, depending on the shifting).

Similarly, the recurrence for $D(n)$:

$$
D(n) = D(n-1) + C(n-1) = D(n-1) + B(n-2) = D(n-1) + A(n-3)
$$

This suggests that $f(n) = D(n) + \text{sequences that just formed "012" at step } n$. But to simplify, note that sequences that contain "012" are exactly those counted in $D(n)$ plus those that newly contain "012" at step n.

Final recurrence:

$$
f(n) = f(n-1) + \text{number of sequences that newly contain "012" at step } n
$$

The number of sequences that newly contain "012" at step n is:

\[

\text{Number of sequences ending with "01" at length } n-1 \text{, then appending 2}
\]

which is \(C(n-1)\). From earlier, \(C(n-1) = B(n-2) = A(n-3)\).

Putting all together:

\[
f(n) = f(n-1) + A(n-3)
\]

where \(A(n)\) satisfies:

\[
A(n) = 2A(n-1) + 2A(n-2)
\]

with initial conditions:

\[
A(0) = 1, \quad A(1) = 2 \times A(0) + 2 \times 0 = 2, \quad A(2) = 2A(1) + 2A(0) = 2 \times 2 + 2 \times 1 = 6
\]

and

\[
f(0) = 0, \quad f(1) = 0, \quad f(2) = 0
\]

since "012" can't

Find A Recurrence Relation For The Number Of N Digit Ternary Sequences That Have Pattern 012 Occurring

Find A Recurrence Relation For The Number Of N Digit Ternary Sequences That Have Pattern 012 Occurring

Related Articles

- I Really Need Help With ThisA Recurring Symbol Throughout The Novel To Kill A Mockingbird Is The Story
- In January 2017 New York Times Estimated That An Individual Residing In A City Viewed Up To __________
- In January 2022, The Management Of Carla Vista Company Concludes That It Has Sufficient Cash To Permit