Augmented Integer Linear Recurrences (AILRs)

Aaron Snook *asnook@andrew.cmu.edu*
Advisor: Manuel Blum *mblum@cs.cmu.edu*

***Abstract*—This paper proposes a new sequence type, the Augmented Integer Linear Recurrence. This sequence type finds patterns in sequences with runs of the same integer present. AILRs use linear recurrence inference as a base, and uses invertible sequence transforms to transform sequences into others and recursively infers the post-transformation sequences.**

I. INTRODUCTION

A common technique used in solving mathematical problems is to solve the problem for simple cases and then extrapolate the results to more complex cases using sequence inference. This technique is frequently used by students to solve word problems. However, there are a considerable number of sequences that have a relatively simple pattern that do not fit common definitions of sequences, such as linear recurrences or polynomials. This abstract proposes a new kind of sequence definition, the Augmented Integer Linear Recurrence (AILR) as a solution to this problem. One of the main features of an AILR is the ability to recognize runs of the same integer repeated in a sequence and to see patterns in these runs.

II. DEFINITIONS

Integer linear recurrence: A linear recurrence in which every term is an integer.
Let S = $\{a_i | i \in N\}$ be a sequence.
The difference sequence of S is the sequence $\{a_{i+1} - a_i | i \in N\}$
The mode sequence and the underlying sequences of S are obtained by first partitioning S into a set of subsequences $\{A_i | i \in N\}$ such that for all $i.j \in N$ if $a_i \in A_j$ and $a_i = a_{i+1}$ then $a_{i+1} \in A_j$ and otherwise $a_{i+1} \in A_{j+1}$.
The mode sequence is the sequence of the cardinalities of each A_i in order.
The underlying sequence is the sequence of the unique integer in each A_i in order (the integer may appear many times in each A_i).
Note that the difference sequence along with the first term of the original sequence is enough to reconstruct the original. Also, the mode and the underlying sequences together contain enough information to construct the original sequence.
This allows us to formally define an AILR:
A sequence S is an AILR iff:

(1) S is an integer linear recurrence
(2) The difference sequence of S is an AILR
(3) The mode and underlying sequences of S are both AILRs

III. EXAMPLE

Consider the sequence

1 1 2 1 2 3 1 2 3 4 1 2 3 4 5…

Its difference sequence is

0 1 -1 1 1 -2 1 1 1 -3 1 1 1 1…

The mode sequence of its difference sequence is

1 1 1 2 1 3 1 4…

The underlying sequence of its difference sequence is

0 1 -1 1 -2 1 -3…

Both the mode and underlying sequences are integer linear recurrences; therefore, they both are AILRs by (1). Therefore by (3) the difference sequence of S is an AILR. Therefore, by (2) the original sequence is an AILR.

IV. Algorithm

This definition allows for an algorithm to find such sequences. Our algorithm's goal, given a finite sequence of integers, is to extend that sequence by some finite number of terms. This algorithm can be modified to return a continuation of the sequence inferred instead. We say we have "inferred a sequence" if we accomplish either of these.

Our algorithm first tests a finite set of integers to see if it is consistent with an integer linear recurrence; if it is, S is extended as a linear recurrence and the extension is returned. Otherwise, we take the difference, mode, and underlying sequences of the input, and recursively attempt to extend those sequences. If we obtain an extension of the difference sequence, or an extension of the mode and underlying sequences, we construct an extension of the original sequence using the extended sequences.

V. Example of Algorithm

Suppose the following sequence were input into the algorithm above:

1 1 2 1 2 3 1 2 3 4 1 2 3 4 5

The algorithm would attempt to infer this as an integer linear recurrence and fail.

The algorithm would also attempt to recursively infer the mode and underlying sequences of this sequence, but this will be fruitless in this algorithm, and we will not focus on this.

The algorithm would then obtain the difference sequence

D: 0 1 -1 1 1 -2 1 1 1 -3 1 1 1 1

The algorithm would attempt to infer this as an integer linear recurrence and fail.

The algorithm would then determine the mode and underlying sequences of this:

Mode: 1 1 1 2 1 3 1 4

Underlying: 0 1 -1 1 -2 1 -3

which would be inferred and extended as linear recurrences – in this case, we extend them by 2:

Mode: 1 1 1 2 1 3 1 4 1 5

Underlying: 0 1 -1 1 -2 1 -3 1 -4

We then combine these to extend the difference sequence D:

0 1 -1 1 1 -2 1 1 1 -3 1 1 1 1 -4 1 1 1 1 1

We then use this sequence to construct an extension of the original sequence:

1 1 2 1 2 3 1 2 3 4 1 2 3 4 5 1 2 3 4 5 6

which completes the extension.

VI. Runtime

The runtime of the algorithm described is $O(3^m m^3)$ where n is the maximum recursive depth allowed, and m is the number of sequence terms inputted. To see this, note that the space of sequences checked is a tree with a root of the original sequence. Each node has branching factor 3 (its mode sequence, its underlying sequence, and its difference sequence). Furthermore, note that for any given sequence S, the mode, underlying, and difference sequences have strictly fewer terms than the original sequence (unless the original sequence contains no runs of length greater than one, which can be detected and discarded easily.) At each point in the tree, Gaussian elimination is used to determine the integer linear sequence and whether or not it is integer valued. This algorithm is $O(m^3)$ where m is the number of sequence terms. Note that since we cannot gain terms this bound applies at all points in the tree. This tree can be pruned using several methods (an example would be to not attempt to interpret the mode and underlying sequences of a sequence with no consecutive terms identical). Also, a depth parameter n can be set to the tree to limit its height, trimming the runtime to $O(3^n m^3)$.

VII. Number of terms used

The more important metric in calculating the efficiency of this algorithm in practice is the number of sequence terms used. For linear recurrences, it takes at least 2n terms to calculate a linear recurrence of degree n[1]. Furthermore, mode and underlying sequences may be arbitrarily smaller than their originals, preventing any mathematical guarantees on number of sequence terms required to infer such a sequence.

VIII. Conflicts, Integer Constraints

The reason why our definition of AILR includes the requirement that sequence terms be integers is that this filters out several interpretations. Without the integer requirement, any finite sequence of numbers (of length ≥ 2) is an AILR since all such sequences can be interpreted as the start of a linear recurrence[1] The integer requirement does not filter out all undesired interpretations, however, and so multiple differing interpretations can be reached. In many cases, the conflict can be resolved with more sequence terms; however, this cannot be guaranteed to be the case. More constraints or an ordering on possible interpretations that favor "better" interpretations are under development.

IX. Future Work

There is significant room for improvement in conflict resolution strategy for this algorithm, as this is the greatest obstacle from guaranteeing a particular result from this algorithm.
Furthermore, the framework of AILRs suggests a more general sequence-inferring technique.
One could propose the following:
A set F of algorithms that either infer a sequence or report failure (in the case of AILRs, F contained only the integer linear recurrence solve).
A set of transformations T (in the case of AILRs, T contained the difference, mode, and underlying transformations), partitioned into sets $\{T_1, T_2 \ldots T_t\}$ where each T_i is a set of transforms which together can be used to deduce the original sequence.
Then, one could construct a inference function G in the following way:
Let S be a given sequence. Attempt to apply all functions in F to S to find an extension. Otherwise recursively apply all transformations in T to S. If there is any i such that all transforms in T_i transform S into a inferable function by G, extend each of the transformed sequences and use them to construct an inference for the original sequence.

References

[1] G. O. Young, "Synthetic structure of industrial plastics (Book style with paper title and editor)," in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.