The Algorithmics of Solitaire-Like Games

Roland Backhouse, Wei Chen, João F. Ferreira[1]

School of Computer Science, University of Nottingham, Nottingham, NG8 1BB, England

---

Abstract

One-person solitaire-like games are explored with a view to using them in teaching algorithmic problem solving. The key to understanding solutions to such games is the identification of invariant properties of polynomial arithmetic. We demonstrate this via three case studies: solitaire itself, tiling problems and a novel class of one-person games.

The known classification of states of the game of (peg) solitaire into 16 equivalence classes is used to introduce the relevance of polynomial arithmetic. Then we give a novel algebraic formulation of the solution to a class of tiling problems. Finally, we introduce an infinite class of challenging one-person games, which we call "replacement-set games", inspired by earlier work by Chen and Backhouse on the relation between cyclotomic polynomials and generalisations of the seven-trees-in-one type isomorphism. We present an algorithm to solve arbitrary instances of replacement-set games and we show various ways of constructing infinite (solvable) classes of replacement-set games.

Keywords: Solitaire, tiling problems, cyclotomic polynomials, cyclotomic game, replacement-set game, seven-trees-in-one, algorithm derivation, invariants, type isomorphism

---

1. Introduction

Puzzles and games have been used for centuries to nurture problem-solving skills. Although often presented as isolated brain-teasers, the desire to know how to win makes games ideal examples for teaching algorithmic problem solving.

There are two concepts that are basic to all algorithms that process input values using some sort of iterative or recursive scheme: invariants and making progress. Although in principle making progress can involve quite complicated theories on well-founded relations, in practice the concept is easy for students to grasp. On the other hand, students may sometimes be given misleading information about invariants; they may be taught that invariants are (only) needed for *post-hoc* verification of program correctness and are very difficult to formulate. In reality, a good understanding of invariants is crucial to successful algorithm design.

In this paper, we study three types of one-person game with a view to using them as an aid to teaching algorithmic problem solving. We set the scene with two well-known examples: (peg) solitaire and a collection of tiling problems. The third type of game we study is novel. It is a class of (one-person) games played on a one-dimensional tape where the objective is to move a checker a certain distance along the tape according to a couple of replacement rules. We develop an algorithm to solve such "replacement-set" games in general and we show how to construct an infinite collection of (solvable) replacement-set games. The common theme in all three types of game is invariants of polynomial arithmetic.

Email addresses: `rcb@cs.nott.ac.uk` (Roland Backhouse), `wzc@cs.nott.ac.uk` (Wei Chen), `joao@joaoff.com` (João F. Ferreira)

[1] Funded by Fundação para a Ciência e a Tecnologia (Portugal) under grant SFRH/BD/24269/2005. Now at School of Computing, University of Teesside.

We begin the paper in section 2 with a brief summary of well-known properties of the game of solitaire. These properties are derived using the algebra of polynomials in a suitably chosen semiring; it is this algebra that is the basis for the novel applications that we discuss in later sections.

Section 3 is about a class of tiling problems. In section 3.1, we show how Golomb's [1] use of colours to solve one such problem is formulated algebraically. Our solution is simpler than the algebraic formulation proposed by Mackinnon [2]. The solution to the class of tiling problems is discussed in section 3.2.

The so-called "nuclear pennies" game [3] is an example of a game which, until now, has been of isolated interest. The game is based on the theorem attributed to Lawvere that "seven trees are one". That is, if T is the type of binary trees, the type T^7 (the cartesian product of T with itself 7 times) is isomorphic to T. The game involves moving a checker 6 squares to the right on a one-dimensional tape (from square 1 to square 7) following rules that reflect the recursive definition of unlabelled binary trees. In section 4, we formulate an infinite collection of games, each with different rules, where the goal is to move a checker a certain number of squares from its starting position on a one-dimensional tape. So far as we are aware, these games and their solution are original to this paper. The games were derived from our study of the problem: given a number n, invent an interesting type T such that T^n is isomorphic to T [4]. In [4], we showed that all such types are based on products of cyclotomic polynomials. The general class of "replacement-set" games is formulated in section 4.2 and an algorithm to solve them is formulated in section 4.3. Methods to construct an infinite collection of solvable replacement-set games are given in section 4.4. These methods are based on the theory of products of cyclotomic polynomials some of which is new to this paper. (Properties of individual cyclotomic polynomials are well-known but properties of products of cyclotomic polynomials of the sort we are interested in do not appear to have been studied elsewhere.) The conclusion (section 5) remarks on some open problems.

2. Solitaire

Solitaire is a well-known game. The game begins with a number of pegs stuck in holes in a board. The holes are arranged in a grid; the shape of the grid (which varies from country to country) is not relevant to our discussion. A move, shown diagrammatically in fig. 1, replaces two pegs by one and the game is to remove all pegs bar one, leaving the peg in a designated position. In this section, we show how invariants of polynomial arithmetic are used in the analysis of moves.

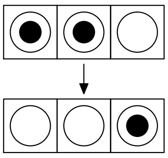


Figure 1: Move from left to right. Similar moves are allowed from right to left, from top to bottom, and from bottom to top.

2.1. Equivalent States

De Bruijn [5] shows that states in the game of solitaire can be divided into 16 equivalence classes in such a way that all moves are between equivalent states. (So the equivalence class of the state is an invariant of each move.) Here is a brief reformulation of De Bruijn's argument[2].

---

[2] Berlekamp *et al* [6, pp. 708–710] attribute the theorem to M. Reiss [7] but the argument is different. De Bruijn's argument is more relevant to later sections of this paper. We have not read Reiss's paper but assume that Berlekamp *et al* copy his presentation.

Suppose we assign non-negative integer coordinates (i,j) to each hole in the board. Suppose $R=(A,\mathbf{0},\mathbf{1},+,\cdot)$ is a semiring[3] and suppose p is an element of A. Assign to a peg at position (i,j) the *weight* p^{i+j}. The *total weight* of a state in the game is the sum of the weights of all the pegs on the board in that state. There are four types of move in the game — vertically up and down, and horizontally left and right. A vertical-up move replaces pegs with weights p^{i+j+0} and p^{i+j+1} by a peg with weight p^{i+j+2}. So, if p has the property that $p^0+p^1=p^2$, the total weight is invariant. Similarly, a horizontal-left move replaces pegs with weights p^{i+2+j} and p^{i+1+j} by a peg with weight p^{i+0+j}. So, if $p^2+p^1=p^0$, the total weight remains invariant. A similar analysis applies to the two other types of moves: if $p^2+p^1=p^0$, the total weight is invariant under vertical-down moves and, if $p^0+p^1=p^2$, the total weight is invariant under horizontal-right moves.

The carrier set of the field[4] $GF(4)$ has exactly 4 elements which can be named $\mathbf{0}$, $\mathbf{1}$, p and p^2. (See fig. 2 for the addition and multiplication tables of $GF(4)$.) Moreover, these elements have the property that $\mathbf{1}+\mathbf{1}=\mathbf{0}$, $\mathbf{1}+p=p^2$ and (hence) $p^2+p=\mathbf{1}$. Thus, if $GF(4)$ is used to compute the weights of states, the weight is invariant under all moves. This divides the states into four equivalence classes, and the initial and final states in the game must be in the same equivalence class.

$+$	$\mathbf{0}$	$\mathbf{1}$	p	p^2
$\mathbf{0}$	$\mathbf{0}$	$\mathbf{1}$	p	p^2
$\mathbf{1}$	$\mathbf{1}$	$\mathbf{0}$	p^2	p
p	p	p^2	$\mathbf{0}$	$\mathbf{1}$
p^2	p^2	p	$\mathbf{1}$	$\mathbf{0}$

(a) Addition

$\times$	$\mathbf{0}$	$\mathbf{1}$	p	p^2
$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$
$\mathbf{1}$	$\mathbf{0}$	$\mathbf{1}$	p	p^2
p	$\mathbf{0}$	p	p^2	$\mathbf{1}$
p^2	$\mathbf{0}$	p^2	$\mathbf{1}$	p

(b) Multiplication

Figure 2: Addition and Multiplication Tables for $GF(4)$

To complete the argument, a symmetrical weighting is used: assign to a peg at position (i,j) the weight p^{i-j}. The total weight is again the sum of the weights of all the pegs on the board in that state. (We call it a "symmetrical" weighting because it is equivalent to turning the board through $90°$.) The same analysis applies, classifying each state into one of 4 equivalence classes.

Combining the two[5], the states are divided into 4×4 equivalence classes in such a way that the equivalence class is invariant under moves. The game can be solved only if the initial and final states are in the same equivalence class.

2.2. The Solitaire Army

De Bruijn's weighting of a state in a game does not provide a sufficient condition for when it is possible to move from a given initial state to a given final state. Berlekamp *et al* [6, chap. 23] discuss in detail a number of problems, when they can be solved and when they cannot be solved. A tool in their analysis is the notion of a *pagoda function*, which computing scientists would recognise as being similar to a measure of progress. Specifically, a pagoda function is any real-valued function pag on peg-positions that has the property that if a move replaces pegs at positions r and s by a peg at position t then

$$pag.t \leq pag.s + pag.r$$

A much-celebrated problem —discussed in several other books and Internet pages— which Berlekamp *et al* [6, chap. 23] solve using a pagoda function is the "solitaire army" problem. This is how they describe the problem.

---

[3] That is, addition in R is associative and commutative and has unit $\mathbf{0}$, product is associative and has unit $\mathbf{1}$ and zero $\mathbf{0}$, and product distributes over addition.

[4] The fact that $GF(4)$ is a field and not just a semiring is not relevant to the argument.

[5] As pointed out to us by Diethard Michaelis, the combination of weights is an element of the semiring $GF(4)\times GF(4)$ where addition and product are defined componentwise. Note that $GF(4)\times GF(4)$ is not a field because it has divisors of zero.

> A number of Solitaire men stand initially on one side of a straight line beyond which is an infinite empty desert. How many men do we need to send a scout just 0, 1, 2, 3, 4 or 5 paces out into the desert?

The surprising fact is that for 5 paces (or more) there is no solution! The proof [6, chap. 23] uses a weighting function similar to the one used by De Bruijn[6]. Suppose peg positions are assigned cartesian coordinates so that the goal position is given the coordinates $(0,0)$ and the initial positions of the Solitaire men have coordinates (i,j) where i is an arbitrary integer and $j \geq n$, for some given number n. Suppose we assign to a peg at position (i,j) the weight $\sigma^{|i|+|j|}$, where σ is yet to be chosen. Note that $|i|+|j|$ is the Manhattan distance of (i,j) from $(0,0)$. The weight of any state in the game is the sum of the weights of all the pegs on the board in that state. It is easy to calculate that, if all positions (i,j) satisfying $j \geq n$ are filled by pegs, and $0 \leq \sigma < 1$, the weight of the state is $\sigma^n \times (1+\sigma) \times (1-\sigma)^{-2}$. (Note that this requires an infinite number of pegs.)

Now σ is chosen so that the weighting of pegs is a pagoda function. Specifically, choose $\sigma = \frac{1}{2}(\sqrt{5}-1)$. This choice guarantees that $\sigma^0 = \sigma^1 + \sigma^2$ and $\sigma^2 < \sigma^1 + \sigma^0$. Thus the weight of a peg that moves towards $(0,0)$ remains constant and the weight of a peg that moves away from $(0,0)$ decreases. (Of course, the weight of a peg that moves but remains at the same distance from $(0,0)$ does not change.) Then, for the 5-pace problem, the goal is to reach a state with weight at least σ^{0+0} (i.e. 1) but this is impossible because the weighting is a pagoda function —its value is never increased by a move— and any initial state with a finite number of pegs has total weight strictly less than $\sigma^5 \times (1+\sigma) \times (1-\sigma)^{-2}$, which simplifies to 1.

Edsger W. Dijsktra [8] discusses a similar problem (and gives a similar solution).

3. Tiling Problems

Tiling problems involve covering a board without overlapping with a given collection of tiles. Traditionally their solution involves (seemingly ad hoc) colouring arguments. This section is essentially about how to formulate the colouring arguments algebraically.

3.1. The Chessboard Problem

Consider the problem of tiling a chessboard with twenty-one 3×1 rectangles and one 1×1 square. Index each square of the chessboard by a pair of natural numbers (i,j) in the obvious way. For concreteness, we assume that the bottom-left corner is given the label $(0,0)$.

Suppose a chessboard is partially tiled by 3×1 rectangles. As in De Bruijn's analysis of solitaire, give to the square (i,j) two "weights": the *forward* weight is p^{i-j} and the *backward* weight is p^{i+j}, where p is a generator of the field $GF(4)$. Two weights are assigned to the chessboard as follows: the *forward weight* of the chessboard is the sum (in $GF(4)$) of the forward weights of all the individual squares that are tiled, and the *backward weight* of the chessboard is the sum of the backward weights of all the individual squares that are tiled.

Recall (fig. 2) that the elements of $GF(4)$ are $\mathbf{0}$, $\mathbf{1}$, p and p^2 and that $\mathbf{1}+\mathbf{1}=\mathbf{0}$ and $\mathbf{1}+p=p^2$. It follows that

$$(1) \qquad \mathbf{0} = \mathbf{1}+p+p^2 \quad .$$

In particular, $p^3 = \mathbf{1}$, the forward weight of square (i,j) is $p^{(i-j) \bmod 3}$ and its backward weight is $p^{(i+j) \bmod 3}$. Thus the weights are identical on forward and backward diagonals of the board, respectively. (This is the explanation for our choice of nomenclature.)

It is easily checked that when a 3×1 tile is placed on a chessboard, both the forward and backward weights of the chessboard do not change; they are *invariants* of the tiling process. (For

---

[6] At the time of writing, the Wikipedia entry entitled "Conway's Soldiers" attributes the problem and its solution to John Horton Conway (one of the authors of [6]) in 1961, which pre-dates De Bruijn's publication; however, neither Wikipedia nor Berlekamp *et al* [6] cite any publication by Conway on this topic.

example, if a $3{\times}1$ tile is placed horizontally on the board with leftmost square at position (i,j), the weight $p^{i+j}{\times}(\mathbf{1}{+}p{+}p^2)$ is added to the weight of the board. Because $\mathbf{1}{+}p{+}p^2$ equals $\mathbf{0}$, adding or subtracting this weight has no effect on the total weight.) This is the basis for the choice of $GF(4)$ in weighing squares: it is the simplest possible semiring that satisfies (1) in a non-trivial way.

The forward and backward weights of a completely tiled chessboard are 1 and p, respectively. In order to tile the chessboard completely with twenty-one $3{\times}1$ rectangles and one $1{\times}1$ square, the $1{\times}1$ square must therefore be placed on a square with forward weight 1 and backward weight p. The former are the squares (i,j) with $(i{-}j\equiv 0) \operatorname{mod} 3$ and the latter are the squares (i,j) with $(i{+}j\equiv 1) \operatorname{mod} 3$. Combined with the requirement that i and j are natural numbers each of which is at most 7, there are just 4 solutions to this pair of equations, namely $(i,j)=(2,2)$, $(i,j)=(5,5)$, $(i,j)=(2,5)$, and $(i,j)=(5,2)$.

The above argument is a simpler presentation of an "algebraic" proof given by Mackinnon [2]. (Our formulation is simpler because Mackinnon takes for p a complex solution of equation (1) — the field of complex numbers is, of course, much more complicated than $GF(4)$.) If colours —say red, white and blue— are assigned to the non-zero elements of $GF(4)$, the argument is essentially the same as Golomb's [1] "colouring" proof. Specifically, Golomb's proof has two components, the colouring of the squares and rotational symmetry of the board. The colouring of the squares is just the assignment of three different values to the squares; this is chosen so that the net "count" of colours on the board —whereby three differently coloured squares "count" as zero— is one. The rotational symmetry is expressed algebraically by the two weights given to squares of the board. The colouring and algebraic proofs are thus in essence identical.

3.2. The Generalisation

In order to demonstrate the effectiveness of the algebraic formulation, let us consider a generalisation. Suppose we have an $m{\times}m$ board, an unlimited supply of n-ominoes and one 1-omino. (An n-omino is an $n{\times}1$ board, i.e. a strip of n squares each of which is the same size as a square of the given $m{\times}m$ board.) In order to eliminate the trivial case, we assume that $1<m$. We prove that it is possible to cover the $m{\times}m$ board with the supplied n-ominoes and one 1-omino, without overlapping, precisely when[7]

$$(2) \qquad 1\leq n<m \;\wedge\; (n\backslash(m{-}1)\vee n\backslash(m{+}1)) \quad .$$

An obvious necessary condition is

$$1\leq n<m \;\wedge\; n\backslash(m^2{-}1) \quad .$$

This, however, is not equivalent. For example, it is satisfied by $m=11$ and $n=8$ but it is not the case that $8\backslash(11{-}1)\vee 8\backslash(11{+}1)$.

3.2.1. Invariant. Arbitrary Semiring

First, we show that (2) is necessary. Consider any semiring $R=(A,\mathbf{0},\mathbf{1},+,\cdot)$ with an element $x{\in}A$ that has the property that[8]

$$(3) \qquad \langle \Sigma i : 0\leq i<n : x^i \rangle \;=_R\; \mathbf{0} \quad .$$

---

[7] The "$\backslash$" symbol denotes the divides relation on integers (more precisely, the converse of the "is-a-multiple-of" relation). Although the notation $m|n$ is traditional, we prefer to use an asymmetric symbol such as the backward slash to denote an asymmetric relation. As the authors of [9, p. 102] point out, vertical bars are overused and $m\backslash n$ gives an impression that m is the denominator of an implied ratio.

[8] We use a systematic notation for quantified expressions which has the form $\langle \bigoplus bv\colon\ range\colon\ term\rangle$. There are five components to the notation, which we explain in turn. The first component is the quantifier, in this case $\bigoplus$. The second component is the dummy bv. The third component is the range of the dummy, a boolean-valued expression that determines a set of values of the dummy. The fourth component is the term. The final component of the notation is the angle brackets; they serve to delimit the scope of the dummy. For more details, see [10, chapter 11] and [11, chapter 8].

(We give examples of such semirings later. The subscript on the equality symbol is necessary later to avoid the confusion that can be caused by overloading.) Now let us assign to each square (i,j) the *weight* x^{i+j} if it is covered and the weight $\mathbf{0}$ if it is not covered. The weight of a (partially) tiled board is defined to be the sum of the weights of the tiled squares.

On account of (3) above, the placement of an n-omino on the board does not change the weight of the board. Since there is exactly one 1-omino on a completely covered board, a necessary condition is that

$$\left\langle \exists k :: \left\langle \Sigma\, i,j : 0 \leq i < m \wedge 0 \leq j < m : x^{i+j} \right\rangle =_R x^k \right\rangle \quad .$$

Equivalently (calculation left to the reader),

$$(4) \qquad \left\langle \exists k :: \left\langle \Sigma i : 0 \leq i < m : x^i \right\rangle^2 =_R x^k \right\rangle \quad .$$

We show that (4) implies $n \backslash (m-1) \vee n \backslash (m+1)$. Our calculations exploit the following immediate consequences of (3):

$$(5) \qquad \left\langle \forall j :: x^j =_R x^{j \,\mathsf{mod}\, n} \right\rangle \quad ,$$

and, hence,

$$(6) \qquad \left\langle \forall j :: \left\langle \Sigma i : 0 \leq i < j : x^i \right\rangle =_R \left\langle \Sigma i : 0 \leq i < j \,\mathsf{mod}\, n : x^i \right\rangle \right\rangle \quad .$$

3.2.2. Invariant. Polynomials over $GF(2)$

To complete our argument, we fix the semiring R to be

$$GF(2)[x] \;/\; \left\langle \Sigma i : 0 \leq i < n : x^i \right\rangle$$

That is, R is the set of polynomials in the indeterminate x with coefficients in $GF(2)$ (which is conventionally denoted by $GF(2)[x]$) modulo the polynomial $\left\langle \Sigma i : 0 \leq i < n : x^i \right\rangle$. Thus, in R we have the property (3).

This choice of R is motivated by our goal. Note first the squaring in (4); $GF(2)$ is the simplest example of a semiring in which squaring distributes through addition. This property is easily seen to be inherited by $GF(2)[x]$. That is, for all j,

$$(7) \qquad \left\langle \Sigma i : 0 \leq i < j : x^i \right\rangle^2 =_{GF(2)[x]} \left\langle \Sigma i : 0 \leq i < j : x^{2i} \right\rangle \quad .$$

Hence, the equality also holds in R. Also, the semiring R has 2^{n-1} distinct elements since each element in R has two representations as a polynomial in $GF(2)[x]$ with degree less than n, and there are 2^n such polynomials. (For example, $\mathbf{0}$ is represented by the two polynomials $\left\langle \Sigma i : 0 \leq i < n : 0 \times x^i \right\rangle$ and $\left\langle \Sigma i : 0 \leq i < n : 1 \times x^i \right\rangle$.) In particular (cf (4))

$$x^k =_R \left\langle \Sigma i : 0 \leq i < n \wedge i \neq k : x^i \right\rangle \quad .$$

Consequently, if the function $\#$ of type $GF(2)[x] \rightarrow \mathbb{N}$ counts the number of non-zero coefficients in a given polynomial, then for all k and all P in $GF(2)[x]$ with degree less than n,

$$(8) \qquad (P =_R x^k) \;\Rightarrow\; (\#P = 1) \vee (\#P = n-1) \quad .$$

(It is at this point that the subscript R on the equality sign becomes essential; the left and right side of the equation denote P and x^k, respectively, after injection into the semiring $GF(2)[x]$ modulo the polynomial $\left\langle \Sigma i : 0 \leq i < n : x^i \right\rangle$.)

We now have:

$$\begin{array}{ll} & (4) \\ = & \{ \quad (6) \text{ and } (7) \quad \} \\ & \left\langle \exists k :: \left\langle \Sigma i : 0 \leq i < m \,\mathsf{mod}\, n : x^{2i} \right\rangle =_R x^k \right\rangle \quad . \end{array}$$

In order to apply (8), we conduct a case analysis on $m \,\mathsf{mod}\, n$ and on n. There are three cases to consider:

(a) $2(m \,\mathsf{mod}\, n) < n$.

This is the easiest case. The degree of $\langle \Sigma i : 0 \le i < m \,\mathsf{mod}\, n : x^{2i} \rangle$ is less than n so, applying (8), we have:

$$(4) \;\Rightarrow\; (m \,\mathsf{mod}\, n = 1) \vee (m \,\mathsf{mod}\, n = n{-}1) \quad .$$

(b) $2(m \,\mathsf{mod}\, n) \ge n \;\wedge\; \mathsf{even}.n$.

Suppose $n = 2q$. The goal is to reduce $\langle \Sigma i : 0 \le i < m \,\mathsf{mod}\, n : x^{2i} \rangle$ to a polynomial with degree less than n. This is done in the following calculation. (Equalities are in R, i.e. in $GF(2)[x] \;/\; \langle \Sigma i : 0 \le i < n : x^i \rangle$.)

$$\begin{array}{ll}
 & \langle \Sigma i : 0 \le i < m \,\mathsf{mod}\, n : x^{2i} \rangle \\
=_R & \{ \quad \text{assumption, } 2(m \,\mathsf{mod}\, n) \ge n \quad \} \\
 & \langle \Sigma i : 0 \le i < q : x^{2i} \rangle \;+\; \langle \Sigma i : q \le i < m \,\mathsf{mod}\, n : x^{2i} \rangle \\
=_R & \{ \quad (5),\; n = 2q \quad \} \\
 & \langle \Sigma i : 0 \le i < q : x^{2i} \rangle \;+\; \langle \Sigma i : q \le i < m \,\mathsf{mod}\, n : x^{2(i-q)} \rangle \\
=_R & \{ \quad \text{quantifier calculus, } \langle \forall a \;:\; a \in GF(2) \;:\; a + a = 0 \rangle, \\
 & \quad\quad m \,\mathsf{mod}\, n < n = 2q \quad \} \\
 & \langle \Sigma i \;:\; m \,\mathsf{mod}\, n - q \le i < q \;:\; x^{2i} \rangle \quad .
\end{array}$$

The last line above is a polynomial in $GF(2)$ with degree less than n. Hence, applying (8) to it, we have:

$$(4) \;\Rightarrow\; (2q - m \,\mathsf{mod}\, n = 1) \vee (2q - m \,\mathsf{mod}\, n = n{-}1) \quad .$$

Simplifying, using $n = 2q$ (and symmetry of disjunction),

$$(4) \;\Rightarrow\; (m \,\mathsf{mod}\, n = 1) \vee (m \,\mathsf{mod}\, n = n{-}1) \quad .$$

(c) $2(m \,\mathsf{mod}\, n) \ge n \;\wedge\; \mathsf{odd}.n$.

Suppose $n = 2q - 1$. Then, by a similar calculation, we have:

$$\begin{array}{ll}
 & \langle \Sigma i : 0 \le i < m \,\mathsf{mod}\, n : x^{2i} \rangle \\
=_R & \{ \quad \text{assumption, } 2(m \,\mathsf{mod}\, n) \ge n, \\
 & \quad\quad (5),\; n = 2q - 1 \quad \} \\
 & \langle \Sigma i : 0 \le i < q : x^{2i} \rangle \;+\; \langle \Sigma i : q \le i < m \,\mathsf{mod}\, n : x^{2(i-q)+1} \rangle \\
=_R & \{ \quad \text{quantifier calculus} \quad \} \\
 & \langle \Sigma i : 0 \le i < m \,\mathsf{mod}\, n : x^{i} \rangle \quad .
\end{array}$$

The last line above is a polynomial in $GF(2)$ with degree less than n. Hence, applying (8) to it, we again get:

$$(4) \;\Rightarrow\; (m \,\mathsf{mod}\, n = 1) \vee (m \,\mathsf{mod}\, n = n{-}1) \quad .$$

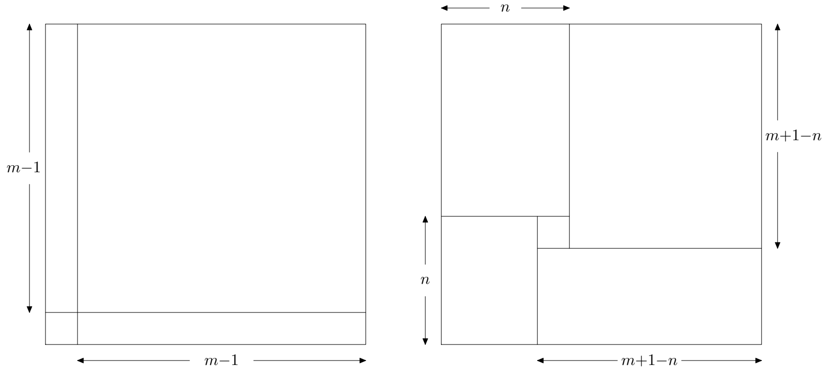
We conclude that, in all cases, (4) implies that

$$(m \,\mathsf{mod}\, n \;=\; 1) \;\vee\; (m \,\mathsf{mod}\, n \;=\; n{-}1) \quad .$$

Equivalently,

$$n\backslash(m{-}1) \;\vee\; n\backslash(m{+}1) \quad .$$

Figs. 3(a) and 3(b) show that this condition is also sufficient. In both figures, the small square identifies where the 1-omino is placed; the remaining rectangles each have at least one side whose length is divisible by n (the side that is labelled by a length) and so can easily be tiled by n-ominoes.



(a) $n\backslash(m{-}1)$ (b) $n\backslash(m{+}1)$

Figure 3: $n\backslash(m{-}1) \vee n\backslash(m{+}1)$ is sufficient

4. Games On Cyclotomic Polynomials

In this section, we propose and solve a novel class of games played on a one-dimensional tape. We call the games *replacement-set* games. The general class is considered in section 4.2; the so-called "nuclear pennies game" [12] based on the "seven trees in one" property [13, 14] is used in section 4.1 to introduce the solution method, which is presented in section 4.3. Section 4.4 shows how to construct an infinite collection of instances of replacement-set games. The construction is based on the theory of cyclotomic polynomials relevant parts of which are summarised in section 4.4.1.

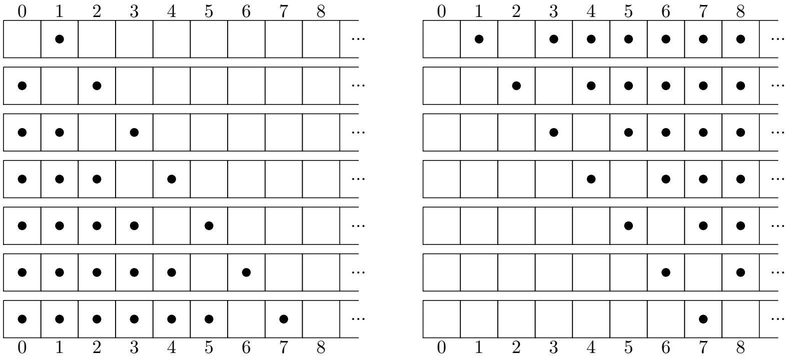
4.1. Seven-Trees-in-One and the Nuclear Pennies Game

Consider the definition of (unlabelled) binary trees — a binary tree is an empty tree or a pair of binary trees. Let us use symbols $+$ and $\times$ to denote disjoint union and cartesian product respectively and let $\mathbb{1}$ denote the unit type. The type T of binary trees can be characterised by the type isomorphism $T \cong \mathbb{1}{+}T{\times}T$. Surprisingly, it can be shown that there is an isomorphism between seven-tuples of binary trees and binary trees. That is, $T^7 \cong T$. This has been dubbed "seven trees in one" by Blass [13] who attributes the isomorphism to a remark made by Lawvere [15]. (See also [16].)

The isomorphism has been turned into a game with checkers called the "nuclear pennies game" [3]. The game is played on an infinite one-dimensional board divided into squares. A checker is placed on one of the squares and the goal is to move the checker six squares to the right. An atomic move is to replace a checker in a square numbered $n{+}1$ by two checkers, one on each of the two adjacent squares n and $n{+}2$, or vice-versa, two checkers, one on square n and one on square $n{+}2$ for some n , are replaced by a checker on square $n{+}1$. (There is no limit to the number of checkers that can be placed on any one square.) The connection with seven-trees-in-one is easy to see if one views a move as replacing $T^n{\times}T$ by $T^n{\times}(\mathbb{1}{+}T{\times}T)$ or vice-versa.

The nuclear-pennies game has an easy solution if one exploits the left-right symmetry of the problem (moving a coin 6 squares to the right is the same as moving a coin 6 squares to the left). The problem is decomposed into first ensuring that there is a checker on the square 6 squares to the right of the starting square and, symmetrically, there is a checker on the square 6 squares to the left of the finishing square.

Achieving this first stage is easy. Fig. 4 shows how it is done. First, six moves are needed to ensure that a checker is added six squares to the right of the starting square. (This is shown in fig. 4(a) using dots to indicate checkers on a square. A blank indicates no checker on the square.) Symmetrically (fig. 4(b)), working from bottom to top, six moves are needed to ensure that a checker is added six squares to the left of the finishing square.



(a) Initial Expansion Phase (b) Final Contraction Phase

Figure 4: Seven Trees in One — Initial and Final Phases

Now the goal is to connect these two intermediate states (the bottom state in fig. 4(a) and the top state in fig. 4(b)). An appropriate (symmetrical) sequence of states is shown in fig. 5 . (For the reader's convenience, the last and first states in figs. 4(a) and 4(b) are repeated as the top and bottom states in fig. 5.)

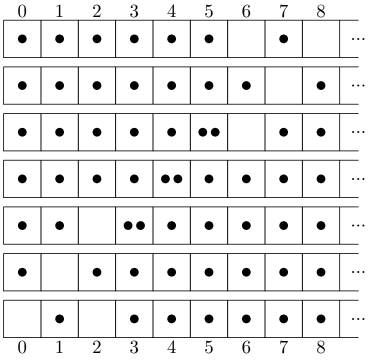


Figure 5: Seven Trees in One — Linking Phase

The first and last moves make the number of checkers in the leftmost and rightmost squares equal. Then a small amount of creativity is needed to identify the two (symmetrical) moves to the (symmetrical) middle state. (In fact, no creativity is needed, as we show later.)

4.2. Replacement-Set Games

Although the nuclear-pennies game is an interesting exercise in the exploitation of symmetry in problem decomposition, it has until recently been an isolated example and appears to have attracted relatively little attention[9]. Chen and Backhouse [4] posed the problem of, given an arbitrary n, is it possible to invent an "interesting" type T such that $T \cong T^{n+1}$. Likewise, given an arbitrary n, is it possible to invent an "interesting" nuclear-pennies-like game in which the task is to move a checker n squares to the right using a sequence of pre-defined atomic moves. They gave an affirmative solution to the first question and a partial solution to the second, both answers being based on the use of so-called cyclotomic polynomials. (The solution to the second problem is partial in the sense that games were invented for an unbounded number of values of n but not all numbers n.) In this section, we present this class of "cyclotomic" games and their solution. (The paper [4] predicts that all cyclotomic games are solvable but does not give an explicit solution.)

Although our original investigation concerned type isomorphisms, in this paper we abandon that application in order to focus on the algorithmics of games with checkers. This allows us to be more inventive in the games we propose.

The games we consider are called *replacement-set* games. Each instance is specified by a multiset R of integers and a natural number n; the multiset R is called the *replacement set* and the number n is called the *displacement*. The games are played on an infinite one-dimensional tape divided into squares. There is an unbounded supply of checkers each of which occupies one square. When playing the game, stacking several checkers on the same square is allowed.

Given a multiset[10] R and displacement n, play begins with one checker on one of the squares. All other squares are empty. The objective is to displace the checker by n squares in a sequence of moves. There are two types of moves: *expansions* and *contractions*. An *expansion* involves choosing an occupied square numbered k, say. One checker on square k is removed and one checker is added on square $k{+}i$ for each occurrence of element i in the multiset R. For example, if the multiset R is $\{\!|1{*}(-1), 2{*}0, 1{*}1|\!\}$, a move would replace one checker on square k by one checker on square $k{-}1$, two checkers on square k and one checker on square $k{+}1$. A *contraction* is the reverse of an expansion. That is, if there is at least one checker on square $k{+}i$ for each occurrence of element i in the multiset R, these checkers are removed and replaced by one checker on square k. Henceforth, we refer to an expansion or contraction *at square* k.

Table 1 lists a number of examples of solvable replacement-set games. The first row, labelled $E_{\{(3,1)\},1}$, is the one just discussed; it is an instance of a simple, general subclass of solvable replacement-set games. Example $E_{\{(2,8)\},3}$ in the fourth row is an instance of the same subclass.

Name	Replacement Multiset	Displacement
$E_{\{(3,1)\},1}$	$\{\!|-1, 2{*}0, 1|\!\}$	3
$D_{2,3}$	$\{\!|-1, 1|\!\}$	6
$M1$	$\{\!|-3, -2, 2, 3|\!\}$	12
$E_{\{(2,8)\},3}$	$\{\!|-3, 0, 5|\!\}$	16
$D_{2,5}$	$\{\!|-3, -1, 1, 3|\!\}$	30
$D_{3,5}$	$\{\!|-7, -4, -2, -1, 1, 2, 4, 7|\!\}$	105
$M2$	$\{\!|-2, 1, 3, 4, 7, 8, 10, 13|\!\}$	120

Table 1: Example Replacement-Set Games

Example $D_{2,3}$ is the so-called "nuclear-pennies game", the game based on seven-trees-in-one; examples $D_{2,3}$, $D_{2,5}$ and $D_{3,5}$ are also instances of a simple, general subclass of solvable

---

[9] The website [12] gives one other example, dubbed the "thermonuclear pennies game". Marcelo Fiore identified an infinite class of examples —see section 4.4.2— at the time of his work on type isomorphisms [14] but did not publish them [private communication, 2010].

[10] We denote a multiset by listing its elements together with their multiplicities within multiset brackets. Specifically, $m{*}a$ denotes m *occurrences* of the *element* a. For brevity, the multiplicity m is sometimes omitted if it is 1. For example, $\{\!|-1, 2{*}0, 1|\!\}$ is an abbreviation of $\{\!|1{*}(-1), 2{*}0, 1{*}1|\!\}$. Both denote a multiset with four elements: -1 and 1 occur once whilst 0 occurs twice.

replacement-set games. The subclasses D and E are discussed further in section 4.4. We have as yet been unable to identify a simple, general proper subclass of the solvable replacement-set games that has example $M1$ as an instance. The same goes for $M2$; " M " stands for "miscellaneous". Example $E_{\{(3,1)\},1}$ is easy to solve: generally, it is easy to construct non-challenging examples of solvable replacement-set games like $E_{\{(3,1)\},1}$ but harder to construct challenging examples.

4.2.1. Quotient Polynomial

Suppose (R,n) is a replacement-set game. Let $-m$ be the smallest element of R . (Typically R has both positive and negative elements. So m is positive. If the elements of R all have the same sign, the game is either unsolvable or trivial.) Then we model the game by the equation:

$$(9) \qquad T^m = \Lambda \times T^m$$

where $\Lambda \times T^m$ is a polynomial in $\mathbb{N}[T]$ (that is, for all i , the coefficients of T^i are non-zero only when $0 \leq i$ and all coefficients are natural numbers). Specifically, the coefficient $\left[T^i\right]\Lambda$ of T^i in Λ is the multiplicity of the element i in the multiset R . For example, the nuclear-pennies game is the replacement-set game ($\{\!|-1\,,1|\!\}\,,6$) and is modelled by the equation

$$(10) \qquad T^1 \;=\; (T^{-1}{+}T^1) \times T^1 \quad .$$

The expansions in a game modelled by (9) are to replace a checker on square k by $\left[T^i\right]\Lambda$ checkers on each of the squares $i{+}k$; this corresponds to the use of the equation (9) as a left-to-right replacement rule. The contractions are the reverse: if for each integer i there are at least $\left[T^i\right]\Lambda$ checkers on square $i{+}k$, remove the $\left[T^i\right]\Lambda$ checkers on the square $i{+}k$ and place one checker on square k . The task is to move from an initial state in which there is just one checker on square m to a final state where there is just one checker on square $m{+}n$.

Example $M1$ is modelled by the equation:

$$(11) \qquad T^3 \;=\; (T^{-3} + T^{-2} + T^2 + T^3) \times T^3 \quad .$$

Playing the game is equivalent to showing that $T^3 = T^{15}$ assuming only that addition and multiplication enjoy the type-isomorphism properties of disjoint sum and cartesian product (that is, addition and multiplication are separately commutative monoids and multiplication distributes through addition).

A necessary condition for a game based on (9) to be solvable is easily determined. Suppose we represent any state of the board by a polynomial in the indeterminate T . Then, the moves are so designed that the polynomial modulo $(\Lambda{-}1) \times T^m$ is an invariant. The initial state is represented by T^{m+0} and the desired final state is represented by T^{m+n} . A necessary condition is thus that

$$(12) \qquad T^{m+0} \,\mathsf{mod}\, ((\Lambda{-}1) \times T^m) \;=\; T^{m+n} \,\mathsf{mod}\, ((\Lambda{-}1) \times T^m) \quad .$$

Excluding trivial games where there is only ever one checker on the board, it is thus necessary that the polynomial $(\Lambda{-}1) \times T^m$ is a divisor (in $\mathbb{Z}[T]$) of $T^n{-}1$.

Note that the introduction of the multiplicative factor T^m ensures that all powers of T in the polynomial $(\Lambda{-}1) \times T^m$ are at least 0 and, hence, that modulo computation is well-defined. This is why the factor is introduced. Where modulo computation is not relevant, we allow polynomials with negative exponents in order to model replacements more directly.

In the next section, we establish that this condition is also sufficient. The proof is constructive: we give an algorithm that solves the given replacement-set game.

4.3. An Algorithm Solving Replacement-Set Games

In this section, we assume that the replacement-set game (R,n) is given. We make extensive use (here and elsewhere) of the (1–1) correspondence between multisets S of natural numbers and polynomials Ψ in the indeterminate T given by

$$(13) \qquad \left\langle \Sigma i :: mult.(i,S) \times T^i \right\rangle \qquad \leftrightarrow \qquad \{\!| i :: (\left[T^i\right]\Psi) * i |\!\}$$

where $mult.(i,S)$ is the multiplicity of i in the multiset S and $\left[T^i\right]\Psi$ is the coefficient of T^i in Ψ. We assume that $-m$ is the smallest element of R and that $(\Lambda{-}1){\times}T^m$ is a divisor of $T^n{-}1$, where Λ is the polynomial such that the coefficient $\left[T^i\right]\Lambda$ of T^i is the multiplicity of the value i in the multiset R. We show how to construct an algorithm that solves the game (R,n).

Underlying the algorithm are three datatypes: multisets of (relative) positions[11], sequences of (relative) positions and polynomials. We switch frequently between the three datatypes, with polynomials playing the central role. As detailed in section 4.2.1, we use $\mathbb{N}[T]$ to represent the state space of replacement-set games. Specifically, a state is represented by a polynomial such that the number of checkers on square i is the coefficient of T^i. We also need to map sequences of replacements to polynomials. If js is a sequence of relative positions, and Λ is the polynomial corresponding to the replacement set R, the polynomial $poly.js$ is defined by

$$poly.[\,] \ = \ 0$$

and

$$poly.(j{:}js) \ = \ (\Lambda{-}1){\times}T^j + poly.js \quad .$$

Suppose s is a polynomial in T, possibly with negative exponents. (Think of s as representing a state of the game.) The sequence js is said to be a *valid expansion sequence from state* s if applying the sequence of expansions specified by js starting from state s is a valid sequence of replacements in the game. Formally,

$$valid.(s\,,\,[\,]) \ = \ (s\in\mathbb{N}[T])$$

and

$$valid.(s\,,\,j{:}js) \ = \ ((s \in \mathbb{N}[T]) \wedge valid.(s+(\Lambda{-}1){\times}T^j \ ,\ js)) \quad .$$

Note that, for all sequences js and states s,

$$valid.(s,js) \ \Rightarrow \ (s\in\mathbb{N}[T]) \wedge (s+poly.js \in \mathbb{N}[T]) \quad .$$

The sequence js is said to be a *valid contraction sequence from state* s if the reverse of js is a valid expansion sequence from state $s-poly.js$.

Occasionally our algorithm leaves the order in which replacements are executed unspecified. In such cases, the replacements are specified by a polynomial, p say. We use the terminology *expanding from square* j *in state* s *according to* p and *contracting from square* j *in state* s *according to* p. Define the ordering $\leq$ on $\mathbb{N}[T]$ by $q\leq p$ equivales $\langle\forall i :: \left[T^i\right] q \leq \left[T^i\right] p\rangle$. Then, an expansion from square j in state s according to p is valid if, for all q in $\mathbb{N}[T]$ such that $q\leq p$,

$$s+(\Lambda{-}1){\times}T^j{\times}q \ \in \ \mathbb{N}[T] \quad .$$

A contraction from square j in state s according to p is valid if, for all q in $\mathbb{N}[T]$ such that $q\leq p$,

$$s-(\Lambda{-}1){\times}T^j{\times}q \ \in \ \mathbb{N}[T] \quad .$$

We begin the development of the algorithm in section 4.3.1 by reviewing the solution to the nuclear pennies game. What makes the nuclear pennies game easy to solve is the fact that $\{-1\,,1\}$ is a *subset* of the replacement set. (In fact, in the nuclear pennies game, the replacement set *equals* $\{-1\,,1\}$ but equality is not necessary.) The structure of our algorithm, which we explain in section 4.3.2, is modelled on the solution to the nuclear pennies game. However, whereas the expansions/contractions in the nuclear pennies game are each single applications of the replacement rule, the expansions/contractions in the general algorithm typically involve a sequence of replacements. The construction of an appropriate sequence of replacements is detailed in section 4.3.3.

---

[11] "Relative" positions are integers; they are used to refer to a position relative to a given position.

4.3.1. Reviewing Nuclear Pennies

The solution to the nuclear pennies game acts as a model for the general algorithm. Recall that the solution consisted of three phases, the “initial expansion” phase, the “linking” phase and the “final contraction” phase: see figs. 4(a), 4(b) and 5. The initial expansion and final contraction phases are symmetric: in the initial expansion phase, checkers on squares 1 thru 7 are replaced in turn and, in the final contraction phase, the reverse of replacing checkers on squares 7 thru 1 in turn is executed. (This description is slightly different from earlier: for the purposes of generalisation, the expansions at square 7 and the contraction at square 1 are also included in the expansion and contraction phases.) The linking phase links the expansion and contraction phases via a middle state — which in this case is symmetrical.

The steps needed to solve the nuclear pennies game are summarised below. The linking phase comprises the two steps labelled “expand at squares 5 and 6” and “contract at squares 2 and 3”. The variable s denotes the state of the tape, expressed as a polynomial in T. The replacement set is expressed by the polynomial $(T^{-1}+T^{1})\times T^{1}$; an expansion at square k is expressed by the addition to s of $(T^{-1}-T^{0}+T^{1})\times T^{k}$ and a contraction at square k is expressed by the subtraction from s of $(T^{-1}-T^{0}+T^{1})\times T^{k}$. Note that the middle state is expressed in two ways, via two syntactically different but nevertheless equal polynomials.

$\{\quad s=T^{1}\quad\}$

initial expansion phase: expand at squares 1 thru 7;

$\{\quad s=T^{1}+\langle\Sigma i:1\leq i\leq 7:T^{i}\times(T^{-1}-T^{0}+T^{1})\rangle\quad\}$

expand at squares 6 and 5;

$\{\quad s=T^{1}+\langle\Sigma i:1\leq i\leq 7:T^{i}\times(T^{-1}-T^{0}+T^{1})\rangle+(T^{5}+T^{6})\times(T^{-1}-T^{0}+T^{1})\quad\}$

$\{\quad s=T^{7}+\langle\Sigma i:7\geq i\geq 1:T^{i}\times(T^{-1}-T^{0}+T^{1})\rangle+(T^{2}+T^{3})\times(T^{-1}-T^{0}+T^{1})\quad\}$

contract at squares 2 and 3;

$\{\quad s=T^{7}+\langle\Sigma i:7\geq i\geq 1:T^{i}\times(T^{-1}-T^{0}+T^{1})\rangle\quad\}$

final contraction phase: contract at squares 7 thru 1

$\{\quad s=T^{7}\quad\}$.

We remarked in section 4.1 that the steps to and from the middle state required a “small amount of creativity”. In fact, no creativity is required. The steps are based on the (unique) factorisation of $T^{6}-1$ by $T^{2}-T+1$:

$$T^{6}-1\;=\;((T^{4}+T^{3})-(T+1))\times(T^{2}-T+1)\;\;.$$

From this factorisation, we easily calculate that, for all γ,

$$T^{1}+\gamma+(T^{5}+T^{6})\times(T^{-1}-1+T^{1})\;\;=\;\;T^{7}+\gamma+(T^{2}+T^{3})\times(T^{-1}-1+T^{1})\;\;.$$

In the solution, γ is chosen to be

$$\langle\Sigma i:1\leq i\leq 7:T^{i}\times(T^{-1}-T^{0}+T^{1})\rangle$$

corresponding to expanding at squares 1 thru 7, or equally

$$\langle\Sigma i:7\geq i\geq 1:T^{i}\times(T^{-1}-T^{0}+T^{1})\rangle$$

corresponding to contracting at squares 7 thru 1. This choice of γ is dictated by the requirements that

(a) The initial expansion and final contraction phases are both valid.

(b) In the linking phase, there are checkers on squares 6, 5, 2 and 3. (This ensures that the expansions and contractions at these squares are valid.)

4.3.2. Algorithm Decomposition

We now consider how to generalise the solution to the nuclear pennies game to the solution of an arbitrary replacement-set game (R,n). Recall that $-m$ is the smallest element of R and that (12) holds, where Λ is the polynomial in $\mathbb{N}[T]$ such that the coefficient $\left[T^i\right]\Lambda$ is the multiplicity of the value i in the multiset R. We assume that $0<m<n$. (It is easy to see that if this condition is not satisfied, the game is either trivial or unsolvable.)

Without loss of generality, we may also assume that $\mathsf{gcd}.R=1$. This is because

$$\Lambda \;=\; \lambda[T:=T^{\mathsf{gcd}.R}]$$

where, for all i,

$$\left[T^i\right]\lambda \;=\; \left[T^{i\times\mathsf{gcd}.R}\right]\Lambda \quad .$$

In a game played with a replacement set R such that $\mathsf{gcd}.R>1$, only squares i for which $i\,\mathsf{mod}\,(\mathsf{gcd}.R)$ equals $m\,\mathsf{mod}\,(\mathsf{gcd}.R)$ are ever occupied by checkers. (This assumption is not used in our algorithm but it does help to simplify our presentation. Chen's forthcoming PhD thesis will include all such details.)

Suppose $T^n-1 \;=\; \delta\times((\Lambda-1)\times T^m)$ where $\delta\in\mathbb{Z}[T]$. Split δ in the obvious way into $\alpha-\beta$ where α and β are both in $\mathbb{N}[T]$, both have degree less than n and

$$\langle\forall i\;::\;\neg(0<\left[T^i\right]\alpha\wedge 0<\left[T^i\right]\beta)\rangle \quad .$$

Then

$$(14)\qquad T^n-1 \;=\; (\alpha-\beta)\times((\Lambda-1)\times T^m) \quad .$$

Our algorithm supposes that α and β have been calculated. The algorithm is, of course, the long-division algorithm for polynomials, so we do not consider it further.

Rewriting (14) as an equation between polynomials in $\mathbb{N}[T]$,

$$(15)\qquad 1+\alpha\times(\Lambda\times T^m)+\beta\times T^m \;=\; T^n+\alpha\times T^m+\beta\times(\Lambda\times T^m) \quad ,$$

and exploiting the assumption that $0<m<n$, we deduce that β and $\Lambda\times T^m$ are monic (i.e. $1=\beta[T:=0]=(\Lambda\times T^m)[T:=0]$) and $\alpha[T:=0]=0$. (For example, in the case of the nuclear pennies game α is T^3+T^4 and β is $1+T$.)

Now, for all γ in $\mathbb{N}[T]$,

$$(16)\qquad T^m+(\gamma+\alpha\times T^{2m})\times(\Lambda-1) \;\;=\;\; T^{m+n}+(\gamma+\beta\times T^{2m})\times(\Lambda-1) \quad .$$

The left and right sides of this equation express the "middle state": the left side is the postcondition of the expansion phase and the right side is the precondition of the contraction phase. Beginning in the state T^m, the expansion phase "expands" the state initially to $T^m+\gamma\times(\Lambda-1)$ and then to $T^m+(\gamma+\alpha\times T^{2m})\times(\Lambda-1)$. The contraction phase does the reverse. Beginning in the state $T^{m+n}+(\gamma+\beta\times T^{2m})\times(\Lambda-1)$ (which by (16) is the postcondition of the expansion phase), the state is "contracted" to $T^{m+n}+\gamma\times(\Lambda-1)$ and finally to T^{m+n}. Recall that the process of expanding the state from some state s to $s+\omega\times(\Lambda-1)$ is called expanding the state *according to* ω. Similarly, a contraction from $s+\omega\times(\Lambda-1)$ to s is called contracting the state *according to* ω. Expanding the state according to ω means executing $\left[T^i\right]\omega$ replacements at square i, for each i. In order for this to be valid irrespective of the order in which the replacements are executed, it is sufficient that in the state s there are at least $\left[T^i\right]\omega$ checkers on square i, for each i. (This is because a replacement at any square does not decrease the number of checkers on other squares.)

As in the case of the nuclear pennies game, there are two requirements on γ:

(a) The initial expansion and final contraction phases are both valid.

(b) In the linking phase, there are sufficient checkers to be able to expand the state $T^m+\gamma\times(\Lambda-1)$ according to $\alpha\times T^{2m}$ and to be able to expand the state $T^{m+n}+\gamma\times(\Lambda-1)$ according to $\beta\times T^{2m}$.

In order to meet these requirements, we choose γ to have the form

$$k\times\left\langle \Sigma i : m\le i\le m{+}n : T^i\times\Psi\right\rangle$$

where k is the maximum of the largest coefficient of α and the largest coefficient of β, and $\Psi = poly.L$ for some sequence L that is valid starting from state T^0 and guarantees that $0 < \left[T^{-1}\right](T^0+\Psi\times(\Lambda{-}1))$ and $0 < \left[T^{1}\right](T^0+\Psi\times(\Lambda{-}1))$. (In effect, Ψ corresponds to a replacement set that contains $\{-1\,,1\}$.) The value of k is easily computed. So our algorithm takes the following form, where the definition of Ψ still has to be formulated.

```
{  $s = T^m$  }
for $k$ times
do  for $m \le i \le m{+}n$ (in order of increasing $i$ )
    do  for $j$ in $L$ (in order)
        do  expand at square $i{+}j$ ;
{  $s = T^m + k\times\langle\Sigma i : m\le i\le m{+}n : T^i\times\Psi\rangle\times(\Lambda{-}1)$  }
expand according to $\alpha\times T^{2m}$;
{  $s = T^m + (k\times\langle\Sigma i : m\le i\le m{+}n : T^i\times\Psi\rangle + \alpha\times T^{2m})\times(\Lambda{-}1)$  }
{  $s = T^{m+n} + (k\times\langle\Sigma i : m{+}n\ge i\ge m : T^i\times\Psi\rangle + \beta\times T^{2m})\times(\Lambda{-}1)$  }
contract according to $\beta\times T^{2m}$;
{  $s = T^{m+n} + k\times\langle\Sigma i : m{+}n\ge i\ge m : T^i\times\Psi\rangle\times(\Lambda{-}1)$  }
for $k$ times
do  for $m{+}n \ge i \ge m$ (in order of decreasing $i$ )
    do  for $j$ in $L$ (in order)
        do  contract at square $i{+}j$
{  $s = T^{m+n}$  }  .
```

The expansion in the initial phase is valid for the following reasons. First, provided Ψ is chosen so that

$$0 < \left[T^{1}\right](T^0+\Psi\times(\Lambda{-}1)) \quad , \tag{17}$$

each time a checker at square $i{+}j$ is replaced, a checker is added at square $i{+}j{+}1$ ready for the next replacement specified by L; also, provided Ψ is chosen so that

$$0 < \left[T^{-1}\right](T^0+\Psi\times(\Lambda{-}1)) \quad , \tag{18}$$

a checker is added at square m when i equals $m{+}1$ ready for any subsequent iteration of executing the expansions specified by L. (Note that $0<n$ so that $m{+}1\le m{+}n$). A similar argument justifies the validity of the contraction phase (with the roles of the two provisos on Ψ reversed). The expansions and contractions according to $\alpha\times T^{2m}$ and $\beta\times T^{2m}$ are then valid because, by inspection of (14) and the fact that $0<m$,

$$m < \mathsf{cod}.(\alpha\times T^{2m}) \le \mathsf{deg}.(\alpha\times T^{2m}) < (n{-}m)+2m \quad ,$$

and

$$m < \mathsf{cod}.(\beta\times T^{2m}) \le \mathsf{deg}.(\beta\times T^{2m}) < (n{-}m)+2m \quad .$$

(For polynomial p, $\mathsf{deg}.p$ is the degree of p: the largest power of T that has a non-zero coefficient; similarly, $\mathsf{cod}.p$ is the smallest power of T that has a non-zero coefficient.) The provisos on the choice of Ψ guarantee the presence of checkers at squares m thru $m{+}n{-}1$ after the initial expansion phase and at squares $m{+}1$ thru $m{+}n$ before the final contraction phase.

The following section documents how to construct Ψ.

4.3.3. The Intermediate State

In this section, we show how to construct a valid sequence of (relative) replacement positions L that begins with (relative) position 0 and guarantees (17) and (18), where $\Psi = poly.L$.

We begin by constructing two multisets A and B. The elements of A and B are non-zero elements of the replacement set R. That is,

$$\mathsf{set}.A \cup \mathsf{set}.B \subseteq \mathsf{set}.R - \{0\} \quad .$$

(The function set coerces a multiset to a set.) On termination of the algorithm, ΣA (the sum of all the elements of the multiset A; for example, $\Sigma\{\!|2{*}(-3)\,,1{*}2\,,1{*}3|\!\} = -1$) is $-(\mathsf{gcd}.R)$ and ΣB is $\mathsf{gcd}.R$. The algorithm is an adaptation of Euclid's algorithm that takes account of the presence of negative as well as positive values in R. It reduces the set R to the set $\{-(\mathsf{gcd}.R), \mathsf{gcd}.R\}$ by repeatedly removing the minimum value x and maximum value y in R, replacing them by either $-(\mathsf{gcd}.\{x,y\})$ or $\mathsf{gcd}.\{x,y\}$, the choice being made in such a way as to guarantee that R always has at least one negative element and at least one positive element. Since we may assume that $\mathsf{gcd}.R = 1$, the set R is thus reduced to $\{-1\,,1\}$.

```
{ $\mathsf{min}.R < 0 < \mathsf{max}.R$ }
$x\,,y \;:=\; \mathsf{min}.R, \mathsf{max}.R$ ;
$Q\,,A\,,B \;:=\; \mathsf{set}.R - \{x,0,y\}\;,\; \{\!|x|\!\}\;,\; \{\!|y|\!\}$ ;
{ Invariant:
        $\Sigma A \,=\, x \;\wedge\; \Sigma B \,=\, y \;\wedge\; x < 0 < y$
    $\wedge$   $Q \subseteq \mathsf{set}.R \;\wedge\; \mathsf{gcd}.(Q \cup \{x,y\}) \,=\, \mathsf{gcd}.R$
    Bound: $|Q|$ }
do $x \neq -(\mathsf{gcd}.R) \vee y \neq \mathsf{gcd}.R$ →
        do $-x < y$ → $y\,,B \;:=\; x{+}y\,, A \uplus B$
        □ $y < -x$ → $x\,,A \;:=\; x{+}y\,, A \uplus B$
        od ;
        { $-x = y \;\wedge\; \mathsf{gcd}.(Q \cup \{x,y\}) \,=\, \mathsf{gcd}.R$ }
        if $Q \neq \emptyset \wedge \mathsf{min}.Q < 0$ → $x \;:=\; \mathsf{min}.Q$ ; $Q\,,A \;:=\; Q{-}\{x\}\,, \{\!|x|\!\}$
        □ $Q \neq \emptyset \wedge \mathsf{max}.Q > 0$ → $y \;:=\; \mathsf{max}.Q$ ; $Q\,,B \;:=\; Q{-}\{y\}\,, \{\!|y|\!\}$
        □ $Q = \emptyset$ → skip
        fi
od
{ $\Sigma A \,=\, -(\mathsf{gcd}.R) \;\wedge\; \Sigma B \,=\, \mathsf{gcd}.R$ }
```

Although not formally documented above, it is important to note that all elements of A and B are non-zero. (The multiset R may contain 0 but the algorithm exploits the fact that 0 is the unit of greatest common divisor: see the initialisation of Q.)

From the multisets A and B we construct a sequence L of relative positions. This list is a valid sequence of positions that guarantees that if there is a checker on some square i and expansions are made at each of the squares in L in order relative to i, on termination there will be at least one checker on squares $i - \mathsf{gcd}.R$ and $i + \mathsf{gcd}.R$.

Any sequence that is formed from the multiset A by ordering its elements and then forming the corresponding sequence of partial sums is valid. (For example, $\{\!|2{*}(-3)\,,2{*}2|\!\}$ might be ordered as the sequence $-3\,,2\,,-3\,,2$; the sequence of partial sums is then $0\,,-3\,,-1\,,-4\,,-2$.) Moreover, starting from a state where there is a checker on square i, such a sequence of partial sums results in a checker on square $i-\mathsf{gcd}.R$ (because ΣA is $-(\mathsf{gcd}.R)$). This is easily proven by induction on the length of the sequence of partial sums; the basis is the assumption that there is a checker on square i and the induction step exploits the fact that $\mathsf{set}.A$ is a subset of $\mathsf{set}.R$. Similarly, any sequence formed from the multiset B in the same way results in a checker on square $i+\mathsf{gcd}.R$. The complication is combining the two sequences; it is not possible, for example, to simply join the sequences because there is no guarantee of there being a checker at relative position 0 after this checker is replaced in the first step.

We resolve the problem by splitting the multisets A and B into positive and negative elements. The head of L is 0 . Let A^- denote the sub-multiset of A consisting of the negative elements of A and let A^+ denote the sub-multiset of A consisting of the positive elements of A . Suppose we sequence the elements of A^- arbitrarily (making sure that the number of times each element appears in the sequence equals its multiplicity in A^-). Let psA^- denote the sequence of partial sums of A^- , omitting the head element, 0 . Note that the last element of this sequence is ΣA^- . Let $\Sigma A^-{+}psA^+$ denote the sequence formed by adding ΣA^- to each element of the sequence of partial sums of A^+ , again omitting the head element 0 . Define $\Sigma B^+{+}psB^-$ similarly. Then the sequence L is given by:

$$L \;=\; [0] +\!\!+\, psA^- +\!\!+\, (\Sigma A^-{+}psA^+) +\!\!+\, psB^+ +\!\!+\, (\Sigma B^+{+}psB^-) \quad .$$

This is a valid sequence starting from an arbitrary state T^i because each of the relative positions in psA^- is negative and so too is each of the relative positions in $\Sigma A^-{+}psA^+$ —the last element of this sequence is ΣA which equals $-(\mathsf{gcd}.R)$ and it is the largest element of the sequence—; similarly, each of the relative positions in psB^+ is positive and so too is each of the relative positions in $\Sigma B^+{+}psB^-$.

This completes the development of the algorithm.

4.4. Constructing Solvable Replacement-Set Games

The algorithm we have derived in section 4.3 would not be of any interest were it not possible to construct many examples of replacement-set games. Fortunately, there is an abundance of such examples. In this section, we show how to construct several infinite classes of replacement-set games; we believe that, except for the very simplest examples in each class, all the games would be very challenging to solve without knowledge of the theory that we have developed.

A well-known result is that, in $\mathbb{Z}[T]$, a polynomial is a divisor of $T^n{-}1$ equivales it is a product of so-called *cyclotomic polynomials*. Specifically, suppose $\Phi.k$ denotes the k th cyclotomic polynomial. Then, using the symbol " $\backslash$ " to denote both the divides relation on polynomials in $\mathbb{Z}[T]$ and the divides relation on numbers

$$\Psi \backslash (T^n{-}1) \quad \equiv \quad \langle \exists D : D \subseteq \mathbb{N} : \Psi = \langle \Pi k : k{\in}D \wedge k\backslash n : \Phi.k \rangle \rangle$$

(The theory of cyclotomic polynomials is very well known. Wikipedia, [9] and [17] are the sources we have used for information on cyclotomic polynomials.)

For brevity, a polynomial in $\mathbb{Z}[T]$ that can be written as $\langle \Pi k : k{\in}D : \Phi.k \rangle$ for some non-empty set D of strictly positive natural numbers will be called a *cyclotomic product*. Note that $\langle \Pi k : k{\in}D : \Phi.k \rangle$ is a divisor of $T^{\mathsf{lcm}.D}{-}1$. (The function lcm returns the lowest common multiple of a set of numbers.) Note also that a cyclotomic product is uniquely defined by the set D . (This is because cyclotomic polynomials are irreducible and $\mathbb{Z}[T]$ is a unique factorisation domain.) Given a cyclotomic product, p , we call the set D of strictly positive natural numbers such that $p = \langle \Pi k : k{\in}D : \Phi.k \rangle$ the *natural factors* of p . (The adjective "natural" is used to avoid confusion with the cyclotomic polynomials $\Phi.k$, where $k{\in}D$, which would be called the *factors* of p .)

The construction of an infinite class of games is straightforward: it suffices to consider just two classes of cyclotomic products:

- **(a)** Each element of the class is a polynomial in $\mathbb{Z}[T]$ such that exactly one of the coefficients is -1 and all other coefficients are in $\mathbb{N}$.
- **(b)** Each element of the class is a polynomial in $\mathbb{N}[T]$.

We call replacement-set games derived from cyclotomic products in class (a) *true* replacement-set games. This is because, for this class of games, a move always truly “replaces” a checker in the sense that the number of checkers on one square strictly decreases (by one) in an “expansion” move: the checker is replaced by checkers elsewhere on the tape. For this class of games, the number m is the exponent of the single negative coefficient in the cyclotomic product.

We call replacement-set games derived from cyclotomic products in class (b) *monotonic* replacement-set games. They are instances of replacement-set games but, in an expansion move, the number of checkers on each square does not decrease. For products in class (b), the choice of m is an arbitrary number that is greater than 0 and less than the degree of the product: if Ψ is a cyclotomic product in $\mathbb{N}[T]$, we take Λ to be $1+\Psi{\times}T^{-m}$.

In this section, we show how to construct an unbounded number of (solvable) true replacement-set games (section 4.4.3) and an unbounded number of (solvable) monotonic games (section 4.4.2). We begin by giving a brief overview of properties of products of cyclotomic polynomials.

4.4.1. Properties of Cyclotomic Products

Cyclotomic polynomials are irreducible factors of $T^n{-}1$ for positive natural numbers n. The properties of cyclotomic polynomials are frequently discussed in textbooks (for example [9]) and, so, we state such properties without proof. The properties of non-trivial *products* of cyclotomic polynomials —which is what we are interested in— do not appear to have been documented, so we are obliged to provide proofs.

The mth cyclotomic polynomial is defined by

$$\Phi.m \;=\; \left\langle \Pi k : 0 \le k < m \wedge k \perp m : e^{\frac{2k\pi}{m}i} \right\rangle$$

where $k\perp m$ means that natural numbers k and m are coprime. From this definition, we have that for all positive natural numbers n,

$$(19) \qquad T^n{-}1 \;=\; \langle \Pi k : 1 \le m \le n \wedge m\backslash n : \Phi.m \rangle \quad .$$

Equation (19) is a recurrence relation in Φ. Solving, we get the inductive definition:

$$(20) \qquad \Phi.m \;=\; \begin{cases} 1, & \text{if } m=0\ ; \\ T-1, & \text{if } m=1\ ; \\ \frac{(T^m-1)}{(T-1)} \times \langle \Pi\ k\ :\ 1<k<m\ \wedge\ k\backslash m\ :\ \frac{1}{\Phi.k} \rangle, & \text{if } m \ge 2\ . \end{cases}$$

With the aid of Möbius function μ, which is defined by the recurrence relation:

$$\langle \Sigma k : 1 \le k \le m\ \wedge\ k\backslash m : \mu.k \rangle \;=\; [m{=}1] \quad ,$$

Φ can be calculated by the following formula:

$$(21) \qquad \Phi.m \;=\; \left\langle \Pi k\ :\ 1 \le k \le m\ \wedge\ k\backslash m\ :\ (T^k{-}1)^{\mu.(\frac{m}{k})} \right\rangle \quad .$$

From the definition of μ, it is easy to show that, for all m, $\mu.m$ is either $+1$, -1 or 0. (See, for example, [9].) It is 0 if the exponent of at least one prime number in the prime factorisation of m is greater than 1. Otherwise, m is said to be *square-free* and $\mu.m$ is $+1$ if the number of prime factors of m is even and -1 if the number of prime factors of m is odd. In general, we have

$$(22) \qquad \Phi.(m{\times}a^2) \;=\; (\Phi.(m{\times}a))[T := T^a] \quad .$$

Using this formula, computing the polynomial form of $\Phi.m$, for all m, is reduced to computing the polynomial form of $\Phi.m$ for all square-free m. For square-free m, formula (21) becomes:

$$(23) \qquad \Phi.m \;=\; \frac{\langle \Pi k\ :\ 1 \le k \le m\ \wedge\ k\backslash m\ \wedge\ \mu.m{=}\mu.k\ :\ T^k{-}1 \rangle}{\langle \Pi k\ :\ 1 \le k \le m\ \wedge\ k\backslash m\ \wedge\ \mu.m{\neq}\mu.k\ :\ T^k{-}1 \rangle} \quad .$$

When $m \ge 2$, the number of terms in the numerator of (23) equals the number of terms in its denominator.

4.4.2. Constructing Monotonic Replacement-Set Games

Monotonic replacement-set games are based on cyclotomic products that are elements of $\mathbb{N}[T]$. In this section, we show how to construct such products.

From (20) we deduce that $(\Phi.1)[T:=0]=-1$ and $(\Phi.m)[T:=0]=1$ for all $m\geq 2$. It follows that a cyclotomic product is an element of $\mathbb{N}[T]$ (i.e. all its coefficients are non-negative) only if 1 is not one of its natural factors. From (22) and (23), any non-trivial cyclotomic product that does not have 1 as a natural factor can be written in the form

$$\left\langle \Pi ab \;:\; ab\in PN \;:\; \frac{T^{\mathsf{fst}.ab}-1}{T^{\mathsf{snd}.ab}-1}\right\rangle$$

for some set PN of pairs of numbers. However, not all such products are cyclotomic products. Nevertheless, this is the basis for one method of constructing cyclotomic products in $\mathbb{N}[T]$.

Suppose we define $\Gamma.(a,b)$ by

$$(24) \qquad \Gamma.(a,b) \;=\; \frac{T^{a\times b}-1}{T^{b}-1} \quad .$$

Then

$$(25) \qquad \Gamma.(a,b) \;=\; \langle \Sigma k \;:\; 0\leq k<a \;:\; T^{k\times b}\rangle$$

and, by combining (19) and (24),

$$(26) \qquad \Gamma.(a,b) \;=\; \langle \Pi k \;:\; k\backslash(a\times b) \wedge \neg(k\backslash b) \;:\; \Phi.k\rangle$$

That is, for all a and b such that $a\geq 2$ and $b\geq 1$, $\Gamma.(a,b)$ is a cyclotomic product in $\mathbb{N}[T]$. This suggests a systematic way of constructing monotonic replacement-set games: for any set PN of pairs of numbers such that for all $ab\in PN$, $\mathsf{fst}.ab\geq 2$ and $\mathsf{snd}.ab\geq 1$,

$$(27) \qquad \langle \Pi ab \;:\; ab\in PN \;:\; \Gamma.ab\rangle$$

is a polynomial in $\mathbb{N}[T]$. It is also a cyclotomic product if

$$\begin{array}{ll} \langle \forall\, ab,cd,a,b,c,d,k & \\ \qquad\qquad : & ab\in PN \wedge cd\in PN \wedge ab\neq cd \wedge ab=(a,b) \wedge cd=(c,d) \\ \qquad\qquad : & \neg(k\backslash(a\times b) \wedge \neg(k\backslash b) \wedge k\backslash(c\times d) \wedge \neg(k\backslash d)) \\ \rangle \quad . & \end{array}$$

That is, if (27) is expanded into a product of cyclotomic polynomials with the aid of (26), no cyclotomic polynomial is repeated. Note that the test on k can be simplified to

$$k\backslash((a\times b)\,\nabla\,(c\times d)) \;\;\Rightarrow\;\; k\backslash b \vee k\backslash d$$

where “ ∇ ” is the gcd operator. Summarising, we have:

Theorem 28. Suppose PN is a set of pairs of numbers with the above property and f is any number such that

$$0<f<\langle \Sigma\, ab : ab\in PN : \mathsf{deg}.(\Gamma.ab)\rangle \quad .$$

Let the replacement-set R be the multi-set defined by the polynomial

$$\langle \Pi ab : ab\in PN : \Gamma.ab\rangle \times T^{-f} \;+\; 1$$

and let n equal $\langle LCM\, ab : ab\in PN : \mathsf{fst}.ab \times \mathsf{snd}.ab\rangle$. Then (R,n) is a solvable replacement-set game, which we denote by $E_{PN,f}$.

$\square$

Examples of such games were given in section (4.2):

$$E_{\{(3,1)\},1} \;=\; (\lbrace\!\vert 1{*}(-1)\,,2{*}0\,,1{*}1 \vert\!\rbrace\,,\,3)$$

and

$$E_{\{(2,8)\},3} \;=\; (\lbrace\!\vert 1{*}(-3)\,,1{*}0\,,1{*}5 \vert\!\rbrace\,,\,16) \quad .$$

In both examples, the set of pairs has just one element so there is nothing to check.

Lots of other examples can be constructed. For example, given a square-free number n, let $prf.n$ denote the set of prime factors of n. (For example, $prf.6 = \{2{,}3\}$.) Then we can choose PN to be $\{p : p{\in}prf.n : (p,1)\}$. Choose any offset f such that $0 < f < \langle\Sigma p : p{\in}prf.n : p{-}1\rangle$. Then the game $(E_{PN,f}, n)$ is solvable. The replacement set R is the multiset corresponding to $(\Psi{+}T^f){\times}T^{-f}$ where $\Psi = \langle\Pi p : p{\in}prf.n : \Gamma.(p,1)\rangle$. For example, taking n to be 6, the polynomial Ψ is

$$1 + 2{\times}T + 2{\times}T^2 + T^3 \quad .$$

Choosing the offset to be 2, we get the (solvable) game

$$(\lbrace\!\vert 1{*}(-2)\,,2{*}(-1)\,,3{*}0\,,1{*}1 \vert\!\rbrace\,,\,6) \quad .$$

Table 2 uses variables n, f, a, b, c and d to define several subclasses of monotonic games; concrete examples follow the description of each subclass. In each case, the requirement on f is that the replacement set contain both strictly positive and strictly negative elements. The requirement on a, b, c and d in the class $E_{\{(a,b),(c,d)\},f}$ is

$$\langle\forall k : k\backslash((a{\times}b)\,\nabla\,(c{\times}d)) : k\backslash b \vee k\backslash d\rangle \quad .$$

The class $E_{\{(2,2n)\},1}$ was previously identified by Marcelo Fiore [private communication, 2010]. The class $E_{\{(a,a^{n-1})\},1}$ was studied by the authors in the earlier paper [18]. The final example illustrates that the multiplicity of several elements of the replacement set may be greater than 1.

Name	Replacement (Multi-)Set	Displacement
$E_{\{(n,1)\},f}$	$\lbrace\!\vert i\text{:}\ 0 \le i < n\text{:}\ i{-}f \vert\!\rbrace \uplus \lbrace\!\vert 0 \vert\!\rbrace$	n
$E_{\{(5,1)\},2}$	$\lbrace\!\vert -2\,,-1\,,2{*}0\,,1\,,2 \vert\!\rbrace$	5
$E_{\{(2,2n)\},f}$	$\lbrace\!\vert -f\ ,\ 0\ ,\ 2n - f \vert\!\rbrace$	$4n$
$E_{\{(2,8)\},5}$	$\lbrace\!\vert -5\,,0\,,3 \vert\!\rbrace$	16
$E_{\{(a,a^{n-1})\},f}$	$\lbrace\!\vert i\text{:}\ 0 \le i < a\text{:}\ i{\times}a^{n-1} - f \vert\!\rbrace \uplus \lbrace\!\vert 0 \vert\!\rbrace$	a^n
$E_{\{(3,3^2)\},7}$	$\lbrace\!\vert -7\,,0\,,2\,,11 \vert\!\rbrace$	27
$E_{\{(a,b),(c,d)\},f}$	$\lbrace\!\vert i\text{:}\ 0 \le i < a \wedge 0 \le j < b\text{:}\ i{\times}b + j{\times}d - f \vert\!\rbrace \uplus \lbrace\!\vert 0 \vert\!\rbrace$	$\mathsf{lcm}.\{a{\times}b\,,c{\times}d\}$
$E_{\{(3,1),(2,8)\},6}$	$\lbrace\!\vert -6\,,-5\,,-4\,,0\,,2\,,3\,,4 \vert\!\rbrace$	48
$E_{\{(2,5),(3,7),(2,14)\},9}$	$\lbrace\!\vert -9\,,-4\,,-2\,,0\,,3\,,2{*}5\,,2{*}10\,,12\,,17\,,19\,,24 \vert\!\rbrace$	210

Table 2: Monotonic Replacement-Set Games

Not every monotonic replacement-set game has the form $E_{PN,f}$ for some PN and some f. The game $M2$ in table 1 is an example.

4.4.3. Constructing True Replacement-Set Games

We now consider the construction of “true” replacement-set games. Recall that these are games derived from a cyclotomic product with one coefficient that is -1 and all other coefficients are positive.

The construction of true replacement-set games seems to be considerably more difficult than the construction of monotonic replacement-set games. (Of course, it is always possible to use a brute-force search of all cyclotomic products of appropriate degree, but we do not regard this as an acceptable construction method.) In this section, we present one class of true replacement-set games, albeit constructed in a somewhat ad hoc fashion. The class of games is formulated in the following theorem.

Lemma 29. For all distinct primes p and q,

$$\Phi.(p{\times}q)\times\Gamma.(p{\times}q-p-q\ ,\ 1) \tag{30}$$

is a cyclotomic product with one coefficient that is -1 and all other coefficients are positive. Moreover, it is a factor of $T^n{-}1$ where n is the least common multiple of $p{\times}q$ and $p{\times}q-p-q$.

Proof Since $p{\times}q$ is greater than $p{\times}q-p-q$, we have that $p{\times}q$ is not a divisor of $p{\times}q-p-q$. By (26), the formula (30) can be rewritten as:

$$\langle\Pi k\ :\ k=p{\times}q\ \vee\ (k\backslash(p{\times}q-p-q)\wedge k\neq 1)\ :\ \Phi.k\rangle\quad .$$

Further, by (19), (30) is a factor of $T^n{-}1$ where n is the least common multiple of $p{\times}q$ and $p{\times}q-p-q$.

We now show that the formula (30) has only one negative coefficient. We begin by observing that

$$
\begin{array}{ll}
 & \Phi.(p{\times}q)\times\Gamma.(p{\times}q-p-q\ ,\ 1)+T^{p\times q\,-\,p\,-\,q} \\
= & \{\qquad (23)\text{ and }(24)\quad\} \\
 & \dfrac{T^{p\times q}{-}1}{T^{p}{-}1}\times\dfrac{T{-}1}{T^{q}{-}1}\times\dfrac{T^{p\times q\,-\,p\,-\,q}{-}1}{T{-}1}\ +\ T^{p\times q\,-\,p\,-\,q} \\
= & \{\qquad \text{polynomial arithmetic}\quad\} \\
 & \dfrac{T^{(q-1)\times p}{-}1}{T^{p}{-}1}\times\dfrac{T^{(p-1)\times q}{-}1}{T^{q}{-}1} \\
= & \{\qquad \text{definition of }\Gamma\quad\} \\
 & \Gamma.(q{-}1\,,p)\times\Gamma.(p{-}1\,,q)\quad .
\end{array}
$$

That is,

$$\Phi.(p{\times}q)\times\Gamma.(p{\times}q-p-q\ ,\ 1)+T^{p\times q\,-\,p\,-\,q}\ \in\ \mathbb{N}[T]$$

and if the ($p{\times}q-p-q$)th coefficient of $\Gamma.(q{-}1\,,p)\times\Gamma.(p{-}1\,,q)$ is 0,

$$\Gamma.(q{-}1\,,p)\times\Gamma.(p{-}1\,,q)\ -\ T^{p\times q\,-\,p\,-\,q}$$

(which equals (30)) has one coefficient that is -1 and all other coefficients are positive.

By (25), powers of monomials in $\Gamma.(q{-}1\,,p)\times\Gamma.(p{-}1\,,q)$ are $j{\times}p+k{\times}q$ for $0\leq j<q{-}1$ and $0\leq k<p{-}1$. But

$$
\begin{array}{ll}
 & j{\times}p+k{\times}q\ =\ p{\times}q-p-q \\
= & \{\qquad \text{arithmetic}\quad\} \\
 & (j{+}1){\times}p+(k{+}1){\times}q\ =\ p{\times}q \\
= & \{\qquad p\text{ and }q\text{ are distinct primes}\quad\} \\
 & (j{+}1=q\ \wedge\ k{+}1=0)\ \vee\ (j{+}1=0\ \wedge\ k{+}1=p) \\
= & \{\qquad 0\leq j<q{-}1\ \wedge\ 0\leq k<p{-}1\quad\} \\
 & \mathsf{false}\quad .
\end{array}
$$

That is, the ($p{\times}q-p-q$)th coefficient of $\Gamma.(q{-}1\,,p)\times\Gamma.(p{-}1\,,q)$ is indeed 0.
□

Theorem 31. Let Ψ denote

$$\Phi.(p{\times}q) \times \Gamma.(p{\times}q - p - q\ ,\ 1) + T^{p\times q\,-\,p\,-\,q}\quad .$$

Define the multiset R to be

$$\{\!|i\!:\ 0 \leq i\!:\ ([T^i]\ \Psi){*}(i - (p{\times}q - p - q))|\!\}$$

and define n to be the least common multiple of $p{\times}q$ and $p{\times}q - p - q$. Then (R, n) is a solvable (true) replacement-set game.

Proof Follows from lemma 29 and the algorithm given in section 4.3.
□

Examples of games constructed using theorem 31 are the games $D_{2,3}$, $D_{2,5}$ and $D_{3,5}$ given in table 1. The game $M1$ in table 1 is also a true replacement-set game but is not an instance of the class D . (It is derived from $\Phi.3 \times \Phi.12$.)

5. Conclusion

Games are a well-established medium for illustrating mathematical insights. They are particularly useful for developing algorithmic-problem-solving skills because games are inherently about algorithms for winning and they require little motivation.

The common theme of all our examples is the exploitation of invariant properties of polynomials. The class of tiling problems discussed in section 3 assumes that all “ominoes” are straight. Golomb [1] extends his colouring argument to other problems where the ominoes are not straight. We haven’t explored these problems. It would be interesting to see whether the algebraic formulation can be extended to such problems in a uniform way.

The primary novel contribution of this paper is the class of “replacement-set games” and their solution. A vital aspect of this class is the ability to construct abundantly many challenging instances — isolated examples of games have very little pedagogical value. Although we have achieved this goal, our investigation has left several open problems. A particular challenge is to identify necessary and sufficient conditions for a cyclotomic product to be an element of $\mathbb{N}[T]$ or to have exactly one coefficient that equals -1 — conditions that admit simple algorithms to construct arbitrary instances.

We believe that, with sufficient imagination, the replacement-set games offer a rich source of examples for the further development of algorithmic-problem-solving skills. We have implemented our algorithm (in Haskell) in order to try to avoid transcription errors in the preparation of this paper but we have put no effort into trying to develop an optimal algorithm and we are well aware that our algorithm is suboptimal in many cases. For example, in our earlier conference paper [18], we developed a more efficient algorithm for a particular subclass of replacement-set games. A useful educational exercise would be to explore improved algorithms for solving particular subclasses of replacement-set games, exploiting the specific properties of the subclass. For example, the cyclotomic products defining true replacement-set games are invariably palindromic (that is, if the coefficients are written in order they form a palindrome) but we have not attempted to exploit this property in any way. It is also very likely that our algorithm can be improved even in the general case.

Acknowledgement. Our thanks to Diethard Michaelis for his comments on the conference paper which formed the basis for the current paper.

References

[1] S. W. Golomb, Polyominoes, George Allen & Unwin Ltd, 1965.
[2] N. Mackinnon, An algebraic tiling proof, The Mathematical Gazette 73 (465) (1989) 210–211.
[3] D. Piponi, Arboreal isomorphisms from nuclear pennies, blog post available at http://blog.sigfpe.com/2007/09/arboreal-isomorphisms-from-nuclear.html (September 2007).

[4] W. Chen, R. Backhouse, From seven-trees-in-one to cyclotomics, unpublished (2010).
URL `http://cs.nott.ac.uk/~wzc`

[5] N. G. de Bruijn, A Solitaire game and its relation to a finite field, J. of Recreational Math 5 (1972) 133–137.

[6] E. R. Berlekamp, J. H. Conway, R. K. Guy, Winning Ways, Vol. I and II, Academic Press, 1982.

[7] M. Reiss, Beiträge zur Theorie des Solitär-Spiels, Journal für die reine und angewandte Mathematik (Crelles Journal) 54 (1857) 344–379.

[8] E. W. Dijkstra, The checkers problem told to me by M.O. Rabin, available from http://www.cs.utexas.edu/users/EWD/ewd11xx/EWD1134.PDF (September 1992).

[9] R. L. Graham, D. E. Knuth, O. Patashnik, Concrete Mathematics : a Foundation for Computer Science, 2nd Edition, Addison-Wesley Publishing Company, 1994.

[10] R. Backhouse, Program Construction. Calculating Implementations From Specifications, John Wiley & Sons, 2003.

[11] D. Gries, F. B. Schneider, A Logical Approach to Discrete Math, Springer-Verlag, 1993.

[12] D. Piponi, Using thermonuclear pennies to embed complex numbers as types, blog post available at http://blog.sigfpe.com/2007/10/using-thermonuclear-pennies-to-embed.html (October 2007).

[13] A. Blass, Seven trees in one, Journal of Pure and Applied Algebra 103 (1) (1995) 1–21.

[14] M. Fiore, Isomorphisms of generic recursive polynomial types, in: Proceedings of the 31st Annual ACM SIGPLAN-SIGACT Symposium on the Principles of Programming Languages, ACM Press, New York, NY, USA, 2004, pp. 77–88.

[15] F. W. Lawvere, Some thoughts on the future of category theory, Lecture Notes in Mathematics 1488 (1991) 1–13.

[16] F. W. Lawvere, Left and right adjoint operations on spaces and data types, Theoretical Computer Science 316 (1-3) (2004) 105 – 111, Recent Developments in Domain Theory: A collection of papers in honour of Dana S. Scott.

[17] I. Isaacs, Algebra: a graduate course, Graduate studies in mathematics, American Mathematical Society, 1994.

[18] R. Backhouse, W. Chen, J. Ferreira, The algorithmics of solitaire-like games, in: C. Bolduc, J. Desharnais, B. Ktari (Eds.), Proceedings of 10th International Conference on Mathematics of Program Construction, Vol. 6120 of LNCS, Springer, 2010, pp. 1–18.