Claude Bolduc
Jules Desharnais
Béchir Ktari (Eds.)

LNCS 6120

Mathematics of Program Construction

10th International Conference, MPC 2010
Québec City, Canada, June 2010
Proceedings

Springer

Lecture Notes in Computer Science 6120

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Claude Bolduc Jules Desharnais
Béchir Ktari (Eds.)

Mathematics of Program Construction

10th International Conference, MPC 2010
Québec City, Canada, June 21-23, 2010
Proceedings

Springer

Volume Editors

Claude Bolduc
Jules Desharnais
Béchir Ktari

Université Laval, Département d'informatique et de génie logiciel
Pavillon Adrien-Pouliot, 1065 Avenue de la Médecine
Québec, QC, G1V 0A6, Canada
E-mail: {Claude.Bolduc, Jules.Desharnais, Bechir.Ktari}@ift.ulaval.ca

Library of Congress Control Number: 2010927075

CR Subject Classification (1998): F.3, D.2, F.4.1, D.3, D.2.4, D.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-13320-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13320-6 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180

Preface

This volume contains the proceedings of MPC 2010, the 10th International Conference on the Mathematics of Program Construction. The biennial MPC conferences aim to promote the development of mathematical principles and techniques that are demonstrably practical and effective in the process of constructing computer programs, whether implemented in hardware or software. The focus is on techniques that combine precision with conciseness, enabling programs to be constructed by formal calculation. Within this theme, the scope of the series is very diverse, including programming methodology, program specification and transformation, program analysis, programming paradigms, programming calculi, programming language semantics, security, and program logics.

The conference took place during June 21–23 in Lac-Beauport, a suburb of Québec City, Canada, prior to AMAST 2010 (June 23–26). The previous nine conferences were held in 1989 in Twente, The Netherlands (LNCS 375); in 1992 in Oxford, UK (LNCS 669); in 1995 in Kloster Irsee, Germany (LNCS 947); in 1998 in Marstrand near Göteborg, Sweden (LNCS 1422); in 2000 in Ponte de Lima, Portugal (LNCS 1837); in 2002 in Dagstuhl, Germany (LNCS 2386); in 2004, in Stirling, UK (LNCS 3125); in 2006 in Kuressaare, Estonia (LNCS 4014); and in 2008 in Marseille-Luminy, France (LNCS 5133).

The volume contains one invited paper, the abstracts of two invited talks, and 19 papers selected for presentation by the Program Committee from 37 submissions. Each paper was refereed by at least three reviewers, and on average by four. We are grateful to the members of the Program Committee and the external referees for their care and diligence in reviewing the submitted papers. The review process and compilation of the proceedings were greatly helped by Andrei Voronkov's EasyChair system that we highly recommend to every Program Chair.

June 2010

Claude Bolduc
Jules Desharnais
Béchir Ktari

Conference Organization

Program Chair

Jules Desharnais	Université Laval, Canada

Program Committee

Philippe Audebaud	École Normale Supérieure de Lyon, France
Ralph-Johan Back	Åbo Akademi University, Finland
Eerke Boiten	University of Kent, UK
Sharon Curtis	Oxford Brookes University, UK
Jeremy Gibbons	University of Oxford, UK
Lindsay Groves	Victoria University of Wellington, New Zealand
Ian J. Hayes	University of Queensland, Australia
Eric Hehner	University of Toronto, Canada
Zhenjiang Hu	National Institute of Informatics, Japan
Johan Jeuring	Utrecht University, The Netherlands
Christian Lengauer	Universität Passau, Germany
Bernhard Möller	Universität Augsburg, Germany
Shin-Cheng Mu	Academia Sinica, Taiwan
David Naumann	Stevens Institute of Technology, USA
José Nuno Oliveira	Universidade do Minho, Portugal
Alberto Pardo	Universidad de la República, Uruguay
Christine Paulin-Mohring	INRIA-Université Paris-Sud, France
Steve Reeves	University of Waikato, New Zealand
Tim Sheard	Portland State University, USA
Georg Struth	Sheffield University, UK
Tarmo Uustalu	Institute of Cybernetics, Estonia

External Reviewers

José Bacelar Almeida	Han-Hing Dang	Makoto Hamana
Luís Barbosa	Simon Doherty	Christoph Herrmann
Bruno Barras	Facundo Domínguez	Richard Jones
Daniel Calegari	Brijesh Dongol	Hiroyuki Kato
Liang-Ting Chen	Steve Dunne	Heiko Mantel
Olaf Chitil	Jean-Christophe Filliâtre	Clare Martin
Robert Colvin	Roland Glück	Kazutaka Matsuda
Alcino Cunha	Sergei Gorlatch	Conor McBride

Larissa Meinicke
Akimasa Morihata
Chris Okasaki
Olga Pacheco
Viorel Preoteasa
Wolfgang Scholz
Luis Sierra
Kim Solin
Martin Vechev
Janis Voigtländer
Yingfei Xiong

Local Organizers

Claude Bolduc, Jules Desharnais, and Béchir Ktari (Université Laval, Canada)

Sponsoring Institutions

- Université Laval, Québec, Canada, http://www.ulaval.ca
- Centre de recherches mathématiques, Université de Montréal, Montréal, Canada, http://www.crm.umontreal.ca

Table of Contents

Invited Talks

Contributed Talks

The Algorithmics of Solitaire-Like Games

Roland Backhouse, Wei Chen, and João F. Ferreira*

School of Computer Science
University of Nottingham
Nottingham NG8 1BB, England
rcb@cs.nott.ac.uk, wzc@cs.nott.ac.uk, joao@joaoff.com

Abstract. 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. With this in mind, this paper explores one-person solitaire-like games.

The key to understanding solutions to solitaire-like games is the identification of invariant properties of polynomial arithmetic. We demonstrate this via three case studies: solitaire itself, tiling problems and a collection of novel 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 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 show how to derive algorithms to solve these games.

Keywords: Solitaire, invariants, tiling problems, polynomials, games on cyclotomic polynomials, seven-trees-in-one, nuclear pennies, algorithm derivation.

1 Introduction

There are two concepts that are basic to all algorithms that process input values using some sort of iterative 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 are often given misleading information about invariants; they are often taught that invariants are (only) needed for *post-hoc* verification of program correctness and very difficult to formulate. In reality, a good understanding of invariants is crucial to successful algorithm design.

For the last seven years, the module Algorithmic Problem Solving has been taught to first-year Computer Science students at the University of Nottingham.

* Funded by Fundação para a Ciência e a Tecnologia (Portugal) under grant SFRH/BD/24269/2005.

C. Bolduc, J. Desharnais, and B. Ktari (Eds.): MPC 2010, LNCS 6120, pp. 1–18, 2010.

The module aims to introduce students to effective problem-solving techniques, particularly in the context of solving problems that demand an algorithmic solution; the first technique that is presented is the use of invariants. At a later stage, two-person games are studied in some depth; games are inherently algorithmic in nature (after all, the goal is to win, which means formulating some sort of algorithm) and require little motivation.

There are many puzzles and games in the mathematical literature which have been used for centuries to nurture problem-solving ability. Many are presented as isolated brain-teasers but, for our pedagogical purposes, it is important that they have two qualities. First, any problem that is studied must have a substantial number of variations which can be used to test students' understanding and, second, the solution method should demonstrate effective algorithmic problem-solving rather than being ad hoc or magic.

Recently, we have been studying one-person solitaire-like games in order to try to extract useful examples for study in the module. In this paper, we present our findings so far, including an infinite class of challenging games that we have invented based on insights from type theory.

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 places to the right on a one-dimensional tape (from position 1 to position 7) following rules that reflect the recursive definition of 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 positions 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].

2 Solitaire and Variations

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 which is not relevant to the current 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 in the game of solitaire and in a variation on solitaire called the solitaire army game.

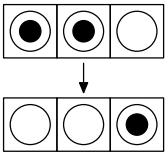


Fig. 1. Move from left to right. Similar moves are allowed from right to left, from top to bottom, and from bottom to top.

2.1 Solitaire

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[1].

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[2] 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[3] $GF(4)$ has exactly 4 elements which can be named $\mathbf{0}$, $\mathbf{1}$, p and p^2. 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.

[1] 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.

[2] 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.

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

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 90^0.) The same analysis applies, classifying each state into one of 4 equivalence classes.

Combining the two[4], the states are divided into $4{\times}4$ 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 they solve using a pagoda function is the "solitaire army" problem. This is how they describe the problem.

> 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 (attributed in Wikipedia to John Horton Conway, 1961) resembles De Bruijn's analysis. 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 negative coordinates. 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 distance of (i, j) from $(0, 0)$ along a shortest path taken by a peg in moves of the game. The weight of any state in the game is the sum of the weights of all the pegs on the board in that state. Now σ is chosen so that the weighting of pegs is a pagoda function. Specifically, choose $\sigma = \frac{1}{2}(\sqrt{5}-1)$ so that $\sigma^2+\sigma = 1$. (This guarantees that the weighting of pegs remains constant when a peg is moved along a shortest path to $(0, 0)$ and decreases for other moves.) Then the maximum weight of an initial state in which a finite number

[4] 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.

of Solitaire men is at a distance at least n from $(0,0)$ is less than σ^{n-5}. For the 5-pace problem, the goal is to reach a state with weight at least σ^{5-5} (i.e. 1) but this is impossible because the weighting is a pagoda function — its value is never increased by a move.

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 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

$$\mathbf{0}=\mathbf{1}+p+p^2 \quad . \tag{1}$$

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 example, if a 3×1 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 3×1 rectangles and one 1×1 square, the 1×1 square must therefore be

placed on a square with forward weight 1 and backward weight p. The former are the squares (i,j) with $(i-j \equiv 0) \bmod 3$ and the latter are the squares (i,j) with $(i+j \equiv 1) \bmod 3$. 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

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

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 $8 \backslash (11-1) \vee 8 \backslash (11+1)$.

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

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

(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 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

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

Equivalently (calculation left to the reader),

$$\left\langle \exists k :: \langle \Sigma i : 0 \le i < m : x^i \rangle^2 =_R x^k \right\rangle \quad . \tag{4}$$

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

$$x^j =_R x^{j \bmod n} \quad , \tag{5}$$

and, hence, for all j,

$$\langle \Sigma i : 0 \le i < j : x^i \rangle =_R \langle \Sigma i : 0 \le i < j \bmod n : x^i \rangle \quad . \tag{6}$$

Invariant. Polynomials over $GF(2)$. To complete our argument, we fix the semiring R to be

$$GF(2)[x] \ / \ \langle \Sigma i : 0 \le i < n : x^i \rangle$$

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 $\langle \Sigma i : 0 \le i < n : x^i \rangle$. 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,

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

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 $\langle \Sigma i : 0 \le i < n : 0 \times x^i \rangle$ and $\langle \Sigma i : 0 \le i < n : 1 \times x^i \rangle$.) In particular (cf (4))

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

Consequently, if the function $\#$ of type $GF(2)[x] \to \mathbb{N}$ 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,

$$(P =_R x^k) \ \Rightarrow \ (\#P = 1) \vee (\#P = n-1) \quad . \tag{8}$$

(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 $\langle \Sigma i : 0 \le i < n : x^i \rangle$).

We now have:

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

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

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

$$(4) \ \Rightarrow \ (m \bmod n = 1) \vee (m \bmod n = n-1) \quad .$$

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

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

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

$$(4) \ \Rightarrow \ (2q - m \bmod n = 1) \vee (2q - m \bmod n = n-1) \quad .$$

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

$$(4) \ \Rightarrow \ (m \bmod n = 1) \vee (m \bmod n = n-1) \quad .$$

(c) $2 \times (m \bmod n) \ge n \ \wedge \ \mathsf{odd}.n$.
Suppose $n = 2q - 1$. Then, by a similar calculation, we have:

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

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

$$(4) \quad \Rightarrow \quad (m \bmod n = 1) \vee (m \bmod n = n{-}1) \quad .$$

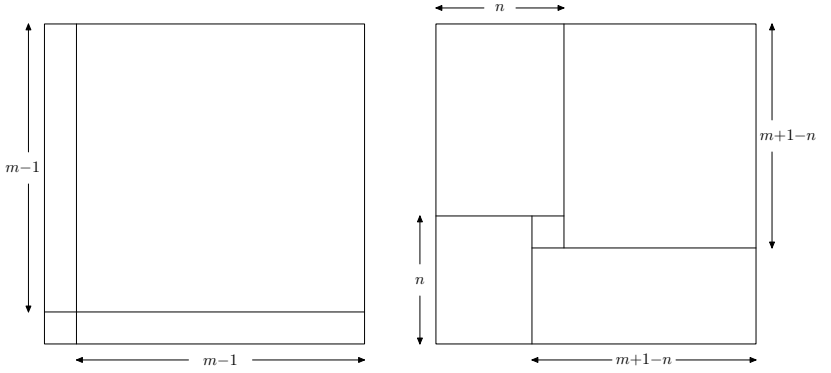
We conclude that, in all cases, (4) implies that

$$(m \bmod n \;=\; 1) \;\vee\; (m \bmod n \;=\; n{-}1) \quad .$$

Equivalently,

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

Figs. 2(a) and 2(b) show that this condition is also sufficient.



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

Fig. 2. $n \backslash (m{-}1) \vee n \backslash (m{+}1)$ is sufficient

4 Games on Cyclotomic Polynomials

In this section, we solve a novel class of games played on a one-dimensional tape. The general class is considered in section 4.2; the so-called "nuclear pennies game" [9] based on the "seven trees in one" property [10,11] is used in section 4.1 to introduce the solution method.

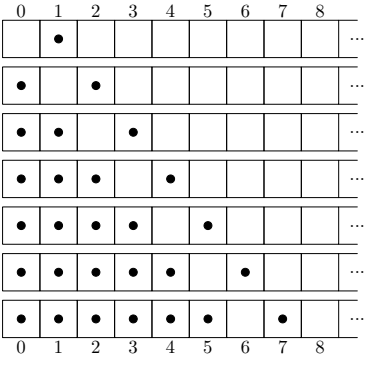
4.1 Seven-Trees-in-One and the Nuclear Pennies Game

Consider the definition of binary trees — a binary tree is an empty tree or an element together with 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 [10] who attributes the isomorphism to a remark made by Lawvere [12].

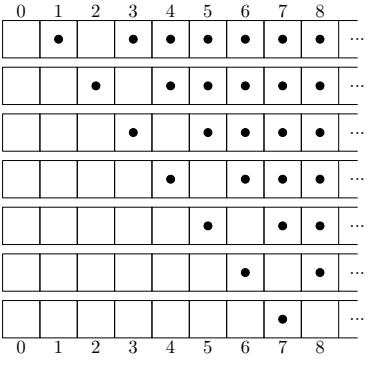
The isomorphism has been turned into a game with checkers called the "nuclear pennies game" [3]. The game is played on a one-dimensional board of infinite extent. A checker is placed on one of the squares and the goal is to move the checker six places to the right. An atomic move is to replace a checker in a square numbered $n{+}1$ by two checkers, one in each of the two adjacent squares n and $n{+}2$, or vice-versa, two checkers, one in square n and one in square $n{+}2$ for some n, are replaced by a checker in square $n{+}1$. 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 places to the right is the same as moving a coin 6 places to the left). The problem is decomposed into first ensuring that there is a checker in the square 6 places to the right of the starting position and, symmetrically, there is a checker in the square 6 places to the left of the finishing position.

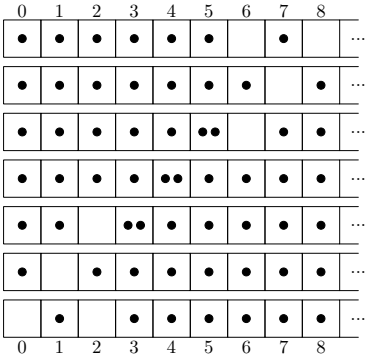
Achieving this first stage is easy. Below we show how it is done. First, six moves are needed to ensure that a checker is added six places to the right of the starting position. (This is shown below using dots to indicate checkers on a square. A blank indicates no checker on the square).



Symmetrically, working from bottom to top, six moves are needed to ensure that a checker is added six places to the left of the finishing position.



Now the goal is to connect these two intermediate states (the bottom state in the top diagram and the top state in the bottom diagram). An appropriate (symmetrical) sequence of states is as follows. (For the reader's convenience, the last and first states in the above figures are repeated as the top and bottom states in the figure below).



The first and last moves make the number of checkers in the leftmost and rightmost positions equal. Then a small amount of creativity is needed to identify the two (symmetrical) moves to the (symmetrical) middle state.

4.2 Cyclotomic Polynomials

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. (The website [9] gives one other example, dubbed the "thermonuclear pennies game".) In a recently submitted paper, 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 places 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).

The games we consider in this section are all based on an equation of the form

$$T^1 = T^1 + \Gamma \tag{9}$$

where Γ is a polynomial in T with positive integer coefficients. The atomic moves in such a game are to supplement a checker at position $i{+}1$ by Γ_k additional checkers at positions $i{+}k$ where Γ_k is the coefficient of T^k in the polynomial Γ —this corresponds to the use of the equation (9) as a left-to-right