Latin Square

Alexander Bentkamp

May 26, 2024

Abstract

A theory about Latin Squares following [1]. A Latin Square is a $n \times n$ table filled with integers from 1 to n where each number appears exactly once in each row and each column. A Latin Rectangle is a partially filled $n \times n$ table with r filled rows and $n - r$ empty rows, such that each number appears at most once in each row and each column. The main result of this theory is that any Latin Rectangle can be completed to a Latin Square.

Contents

theory *Latin-Square*
imports *Marriage.Marriage*
begin

This theory is about Latin Squares. A Latin Square is a $n \times n$ table filled with integers from 1 to n where each number appears exactly once in each row and each column.

As described in "Das Buch der Beweise" a nice way to describe these squares by a $3 \times n$ matrix. Each column of this matrix contains the index of the row r, the index of the column c and the number in the cell (r,c). This $3 \times n$ matrix is called orthogonal array ("Zeilenmatrix").

I thought about different ways to formalize this orthogonal array, and came up with this: As the order of the columns in the array does not matter at all and no column can be a duplicate of another column, the orthogonal array is in fact a set of 3-tuples. Another advantage of formalizing it as a set is that it can easily model partially filled squares. For these 3-tuples I decided against 3-lists and against $nat \times nat \times nat$ (which is really $(nat \times nat) \times nat$) in favor of a function from a type with three elements to nat.

Additionally I use the numbers 0 to $n - 1$ instead of 1 to n for indexing the rows and columns as well as for filling the cells.

datatype *latin-type* = *Row* | *Col* | *Num*

latin_type is of sort enum, needed for "value" command

instantiation *latin-type* :: *enum*
begin
definition *enum-latin-type* == [*Row*, *Col*, *Num*]
definition *enum-all-latin-type* (*P*:: *latin-type* ⇒ *bool*) = (*P Row* ∧ *P Col* ∧ *P Num*)
definition *enum-ex-latin-type* (*P*:: *latin-type* ⇒ *bool*) = (∃ *x*. *P x*)

instance
⟨*proof*⟩

end

Given a latin_type t, you might want to reference the other two. These are "next t" and "next (next t)":

definition [*simp*]:*next t* ≡ (*case t of Row* ⇒ *Col* | *Col* ⇒ *Num* | *Num* ⇒ *Row*)

lemma *all-types-next-eqiv*:(∀ *t*. *P* (*next t*)) ⟷ (∀ *t*. *P t*)
⟨*proof*⟩

We call a column of the orthogonal array a latin_entry:

type-synonym *latin-entry* = *latin-type* ⇒ *nat*

This function removes one element of the 3-tupel and returns the other two as a pair:

definition *without* :: *latin-type* ⇒ *latin-entry* ⇒ *nat* × *nat* **where**
[*simp*]:*without t* ≡ λ*e*. (*e* (*next t*), *e* (*next* (*next t*)))

value *without Row* (λ*t*. *case t of Row* ⇒ *0* | *Col* ⇒ *1* | *Num* ⇒ *2*) — returns (1,2)

abbreviation *row-col* ≡ *without Num*

returns row and column of a latin_entry as a pair.

abbreviation *col-num* ≡ *without Row*

returns column and number of a latin_entry as a pair.

abbreviation *num-row* ≡ *without Col*

returns number and row of a latin_entry as a pair.

A partial latin square is a square that contains each number at most once in each row and each column, but not all cells have to be filled. Equivalently we can say that any two rows of the orthogonal array contain each pair of two numbers at most once. This can be expressed using the inj_on predicate:

definition *partial-latin-square* :: *latin-entry set* ⇒ *nat* ⇒ *bool* **where**
partial-latin-square s n ≡

$(\forall t.\ inj\text{-}on\ (without\ t)\ s) \wedge$ — numbers are unique in each column (t=Row), numbers are unique in each row (t=Col), rows-column combinations are specified unambiguously (t=Num)
$(\forall e{\in}s.\ \forall t.\ e\ t < n)$ — all numbers, column indices and row indices are <n

value *partial-latin-square* {
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 1 \mid Num \Rightarrow 0)$,
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 1 \mid Col \Rightarrow 0 \mid Num \Rightarrow 1)$
} *2* — True

value *partial-latin-square* {
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 0 \mid Num \Rightarrow 1)$,
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 1 \mid Col \Rightarrow 0 \mid Num \Rightarrow 1)$
} *2* — False, because 1 appears twice in column 0

Looking at the orthogonal array a latin square is given iff any two rows of the orthogonal array contain each pair of two numbers at exactly once:

definition *latin-square* :: *latin-entry set* ⇒ *nat* ⇒ *bool* **where**
latin-square s n ≡
$(\forall t.\ bij\text{-}betw\ (without\ t)\ s\ (\{0..<n\}\times\{0..<n\}))$

value *latin-square* {
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 0 \mid Num \Rightarrow 1)$, $(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 1 \mid Num \Rightarrow 0)$,
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 1 \mid Col \Rightarrow 0 \mid Num \Rightarrow 0)$, $(\lambda t.\ case\ t\ of\ Row \Rightarrow 1 \mid Col \Rightarrow 1 \mid Num \Rightarrow 1)$
} *2* — True

value *latin-square* {
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 0 \mid Num \Rightarrow 1)$, $(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 1 \mid Num \Rightarrow 0)$,
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 1 \mid Col \Rightarrow 0 \mid Num \Rightarrow 0)$, $(\lambda t.\ case\ t\ of\ Row \Rightarrow 1 \mid Col \Rightarrow 1 \mid Num \Rightarrow 0)$
} *2* — False, because 0 appears twice in Col 1 and twice in Row 1

A latin rectangle is a partial latin square in which the first m rows are filled and the following rows are empty:

definition *latin-rect* :: *latin-entry set* ⇒ *nat* ⇒ *nat* ⇒ *bool* **where**
latin-rect s m n ≡
$m \leq n \wedge$
partial-latin-square s n ∧
bij-betw row-col s $(\{0..<m\}\times\{0..<n\}) \wedge$
bij-betw num-row s $(\{0..<n\}\times\{0..<m\})$

value *latin-rect* {
$(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 0 \mid Num \Rightarrow 1)$, $(\lambda t.\ case\ t\ of\ Row \Rightarrow 0 \mid Col \Rightarrow 1 \mid Num \Rightarrow 0)$
} *1 2* — True

value *latin-rect* {
(λ*t*. *case t of Row* ⇒ *0* | *Col* ⇒ *0* | *Num* ⇒ *1*), (λ*t*. *case t of Row* ⇒ *0* | *Col* ⇒ *1* | *Num* ⇒ *0*),
(λ*t*. *case t of Row* ⇒ *1* | *Col* ⇒ *0* | *Num* ⇒ *0*), (λ*t*. *case t of Row* ⇒ *1* | *Col* ⇒ *1* | *Num* ⇒ *1*)
} *1 2* — False

There is another equivalent description of latin rectangles, which is easier to prove:

lemma *latin-rect-iff*:
m≤*n* ∧ *partial-latin-square s n* ∧ *card s* = *n*∗*m* ∧ (∀ *e*∈*s*. *e Row* < *m*) ⟷ *latin-rect s m n*
⟨*proof*⟩

A square is a latin square iff it is a partial latin square with all n^2 cells filled:

lemma *partial-latin-square-full*:
partial-latin-square s n ∧ *card s* = *n*∗*n* ⟷ *latin-square s n*
⟨*proof*⟩

Now we prove Lemma 1 from chapter 27 in "Das Buch der Beweise". But first some lemmas, that prove very intuitive facts:

lemma *bij-restrict*:
assumes *bij-betw f A B* ∀ *a*∈*A*. *P a* ⟷ *Q* (*f a*)
shows *bij-betw f* {*a*∈*A*. *P a*} {*b*∈*B*. *Q b*}
⟨*proof*⟩

lemma *cartesian-product-margin1*:
assumes *a*∈*A*
shows {*p*∈*A*×*B*. *fst p* = *a*} = {*a*}×*B*
⟨*proof*⟩

lemma *cartesian-product-margin2*:
assumes *b*∈*B*
shows {*p*∈*A*×*B*. *snd p* = *b*} = *A*×{*b*}
⟨*proof*⟩

The union of sets containing at most k elements each cannot contain more elements than the number of sets times k:

lemma *limited-family-union*: *finite B* ⟹ ∀ *P*∈*B*. *card P* ≤ *k* ⟹ *card* (⋃ *B*) ≤ *card B* ∗ *k*
⟨*proof*⟩

If f hits each element at most k times, the domain of f can only be k times bigger than the image of f:

lemma *limited-preimages*:
assumes ∀ *x* ∈ *f* ‘ *D*. *card* ((*f* −‘ {*x*})∩*D*) ≤ *k finite D*
shows *card D* ≤ *card* (*f* ‘ *D*) ∗ *k*

⟨*proof*⟩

Let $A_1, \ldots, A_n$ be sets with $k > 0$ elements each. Any element is only contained in at most k of these sets. Then there are more different elements in total than sets A_i:

lemma *union-limited-replicates*:
assumes *finite I* $\forall\, i{\in}I.$ *finite* $(A\ i)$ $k{>}0$ $\forall\, i{\in}I.$ *card* $(A\ i) = k$ $\forall\, i{\in}I.$ $\forall\, x{\in}(A\ i).$ *card* $\{i{\in}I.\ x{\in}A\ i\} \leq k$
shows *card* $(\bigcup i{\in}I.\ (A\ i)) \geq$ *card I* ⟨*proof*⟩

In a $m \times n$ latin rectangle each number appears in m columns:

lemma *latin-rect-card-col*:
assumes *latin-rect s m n* $x{<}n$
shows *card* $\{e\ Col | e.\ e{\in}s \wedge e\ Num = x\} = m$
⟨*proof*⟩

In a $m \times n$ latin rectangle each column contains m numbers:

lemma *latin-rect-card-num*:
assumes *latin-rect s m n* $x{<}n$
shows *card* $\{e\ Num | e.\ e{\in}s \wedge e\ Col = x\} = m$
⟨*proof*⟩

Finally we prove lemma 1 chapter 27 of "Das Buch der Beweise":

theorem
assumes *latin-rect s* $(n{-}m)$ *n* $m{\leq}n$
shows $\exists\, s'.\ s{\subseteq}s' \wedge$ *latin-square* s' *n*
⟨*proof*⟩

end

References

[1] M. Aigner and G. Ziegler. *Das Buch der Beweise*. Springer, 2004.