[Knu11, Chapter 7] and [Knu22, Chapter 7.2.2.1]

Langford Pairings and Exact Covers

Anakin

Outline

Langford Pairings

Characterization and Existance

Enumeration

Exact Covers

Section 1

Langford Pairings

Langford Pairings

Consider the following list, called a "Langford pairing"

$$[2, 3, 1, 2, 1, 3] \tag{1}$$

It has a very peculiar property. Each pair of the same digits k has exactly k numbers between them

- There is exactly **1** number between both **1**'s
- There is exactly **2** numbers between both **2**'s
- There is exactly **3** numbers between both **3**'s

Exercise: Consider the list of digits $[1, 1, \ldots, n, n]$. Creating such a number as in Equation 1 is impossible for $n = 1$ or 2. We just saw it's possible for $n = 3$. Come up with a pairing for $n = 4$.
Answer: $[4, 1, 3, 1, 2, 4, 3, 2]$ or $[2, 3, 4, 2, 1, 3, 1, 4]$.

Existence of Langford Pairs

- So these Langford pairs for $[1, 1, \ldots, n, n]$ exist sometimes
  - Trivially[1], it exists for $n = 0$
  - No such pairing exists for $n = 1$ or $n = 2$ (try it yourself)
  - We just saw pairings exist for $n = 3$ and $n = 4$
  - Can we characterize for exactly which n we can find pairings?

[1] or perhaps stupidly, depending on your perspective

Section 2

Characterization and Existance

A characterization of n

We are going to characterize the set of n that have at least one Langford pairing. In doing so, we will find a formula to construct these pairings.

Theorem [Dav59]: *The numbers* $[1, 1, \ldots, n, n]$ *can be arranged in a Langford pairing if and only if* n *is a multiple of* 4 *or one less than a multiple of* 4

Proof of the Theorem

- Suppose $[1, 1, \ldots, n, n]$ can be arranged into some sort of Langford pairing.
- Consider the numbers in such a pairing. Let a_r be equal to the index of the first time r appears in the sequence
  - Then note that $a_r + r + 1$ is the index of the second time r appears
- These a_r and $a_r + r + 1$ are just some arrangement of the indices 1 through $2n$

Proof of the Theorem

Since the a_r and $a_r + r + 1$ are just some arrangement of the indices 1 through $2n$

$$\begin{aligned} \sum_{r=1}^{n} a_r + \sum_{r=1}^{n} (a_r + r + 1) &= 2\sum_{r=1}^{n} a_r + \sum_{r=1}^{n} r + \sum_{r=1}^{n} 1 \\ &= 2\sum_{r=1}^{n} a_r + \frac{n(n+1)}{2} + n \end{aligned}$$

Proof of the Theorem

But the indices in total must sum to

$$\sum_{i=1}^{2n} i = \frac{2n(2n+1)}{2} = 2n^2 + n$$

This implies that

$$2\sum_{r=1}^{n} a_r + \frac{n(n+1)}{2} + n = 2n^2 + n$$

which in turn implies that

$$\sum_{r=1}^{n} a_r = \frac{3n^2 - n}{4}$$

Proof of the Theorem

All the a_r are integers which means that $\sum_{r=1}^{n} a_r$ is an integer. Thus $\frac{3n^2-n}{4}$ must be an integer

If n is an integer, than n is either $4m$, $4m+1$, $4m+2$, or $4m+3$

Plugging in all possible options into $\frac{3n^2-n}{4}$ yields that $n = 4m$ or $4m+3 = 4(m+1)-1$. Thus n is a multiple of 4 or one less than a multiple of 4

Formula for general n

These formulas are from [Dav59]. The terms hidden by . . .'s are consecutive even / odd terms. Ex: $(2, 4, 8, \ldots)$, $(1, 3, 5, \ldots)$

The case $n = 4m$: $4m-4, \ldots, 2m, 4m-2, 2m-3, \ldots, 1, 4m-1,$
$1, \ldots, 2m-3, 2m, \ldots, 4m-4, 4m, 4m-3, \ldots, 2m+1, 4m-2,$
$2m-2, \ldots, 2, 2m-1, 4m-1, 2, \ldots, 2m-2, 2m+1, \ldots, 4m-3, 2m-1, 4m$

The case $n = 4m-1$: $4m-4, \ldots, 2m, 4m-2, 2m-3, \ldots, 1, 4m-1,$
$1, \ldots, 2m-3, 2m, \ldots, 4m-4, 2m-1, 4m-3, \ldots, 2m+1, 4m-2,$
$2m-2, \ldots, 2, 2m-1, 4m-1, 2, \ldots, 2m-2, 2m+1, \ldots, 4m-3$

Exercise: Convince yourself these formulas work by writing a program that generates Langford pairings using these formulas

Section 3

Enumeration

Enumeration

- For $n = 4m$ or $n = 4m - 1$, Langford pairings exist
- For $n = 0, 3, 4$ the solution is unique. What about larger n?
- There are many pairings for larger n
  - Can we enumerate them?
- Let L_n denote the number of Langford pairings. We will count a pairing and it's reverse as the same.
- The state of the matter is that it is quite hard to compute L_n
- John Miller has a wonderful online history on enumerating Langford pairings for various n

Some Formulas

Mike Godfrey[2] in 2002 came up with the following formula. For a derivation, see Exercise 6a of [Knu11, Chapter 7]

$$\text{Let } f(x_1, \ldots, x_{2n}) = \prod_{k=1}^{n} \left(x_k x_{n+k} \sum_{j=1}^{2n-k-1} x_j x_{j+k+1} \right)$$

$$\text{Then } \sum_{x_1, \ldots, x_{2n} \in \{-1, 1\}} f(x_1, \ldots, x_{2n}) = 2^{2n+1} \cdot L_n$$

[Pan21] conjectures some asymptotic approximations for L_n

[2] http://dialectrix.com/langford/godfrey/method.html

Section 4

Exact Covers

Exact Cover Problems

- Langford Pairings are a special case of a type of problem called *Exact Cover*
- In 1972, Richard Karp proved that Exact Cover, among 20 other problems, is NP-Complete
  - Easy to verify solutions in polynomial time
  - Hard to solve, best known solutions run in exponential time
  - Can simulate (or reduce) other problems in NP using Exact Cover
- The goal of Exact Cover is to “cover” a list of items using different given subsets, and select each item exactly one time

An Example of Exact Cover

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

We can abstract this to *options* containing *items*

$$\begin{array}{lll} 1\colon [3,5] & 2\colon [1,4,7] & 3\colon [2,3,6] \\ 4\colon [1,4,6] & 5\colon [2,7] & 6\colon [4,5,7] \end{array}$$

Answer: Select options 1, 4, and 5

Solving Exact Cover Problems

In trying to solve the previous problem, you may have naturally found a recursive algorithm to find a solution

```
FindCover(Options, Cover, i):
  if Cover is a cover:
    terminate successfully
  if no option in Options contains i:
    terminate unsuccessfully

  I ← options in Options that contain i
  Options ← Options \ I
  for each O in I:
    j ← an item still not covered
    FindCover(Options, Cover ∪ { O }, j)
```

Non-recursive Algorithms

- In [Knu22, Chapter 7.2.1.1], Knuth talks about algorithms which solve exact cover problems
- He does so using method involving doubly linked lists
  - He colorfully calls these *dancing links*
- His AlgorithmX uses dancing links to solve exact cover problems

Langford Pairings as an Exact Cover

- Let's model finding a Langford Pairing as an exact cover problem
- Suppose $n = 4$, then we want to place $[1, 1, \ldots, 4, 4]$ in a list of size 8
- Our items can be slots in the list: $l_1, l_2, \ldots, l_8$
- Our options can be modeled as such

$1\colon [l_1, l_3]$ $1\colon [l_2, l_4]$ $1\colon [l_3, l_5]$ $1\colon [l_4, l_6]$ $1\colon [l_5, l_7]$ $1\colon [l_6, l_8]$

$2\colon [l_1, l_4]$ $2\colon [l_2, l_5]$ $2\colon [l_3, l_6]$ $2\colon [l_4, l_7]$ $2\colon [l_5, l_8]$

$3\colon [l_1, l_5]$ $3\colon [l_2, l_6]$ $3\colon [l_3, l_7]$ $3\colon [l_4, l_8]$

$4\colon [l_1, l_6]$ $4\colon [l_2, l_7]$ $4\colon [l_3, l_8]$

Langford Pairings as an Exact Cover

- We can generalize this
- For general n, what items do we have?
  - $l_1, \ldots, l_{2n}$
- For some $1 \leq i \leq n$, what j, k work to form an option $i\colon [l_j, l_k]$? Say $j < k$ to avoid duplicates
  - $1 \leq j < k \leq 2n$
  - $k = j + i + 1$
- So all of our options take the form

$$i\colon [l_j, l_k], \qquad \text{for } 1 \leq j < k \leq 2n, \qquad k = j + i + 1, \qquad 1 \leq i \leq n.$$

- We can use our algorithm FindCover to (perhaps slowly) find all solutions for general n

Questions?

Combinatorics is special. Most mathematical topics which can be covered in a lecture course build towards a single, well-defined goal, such as the Prime Number Theorem. Even if such a clear goal doesn't exist, there is a sharp focus (e.g. finite groups). By contrast, combinatorics appears to be a collection of unrelated puzzles chosen at random. Two factors contribute to this. First, combinatorics is broad rather than deep. Second, it is about techniques rather than results.

— PETER J. CAMERON (1995)

Questions!

$$i\colon [l_j, l_k], \qquad \text{for } 1 \le j < k \le 2n, \qquad k = j + i + 1, \qquad 1 \le i \le n.$$

- Exercise 15 of [Knu22, Chapter 7.2.2.1]: Recall our formulation of finding Langford Pairings as an exact cover. Running FindCover on this will produce a pairing and it's reverse. Modify our formulation to only produce half of the Langford Pairings for n, where the missing half is the reversals of the solutions we find.

- Use the formulation of Langford Pairings stated before, or the one you find in the previous exercise, to write a program that finds all Langford Pairings for a given n. Try your algorithm out for $n = 7$ (there are 26, not including reversals).

Bibliography

Roy O. Davies.
On langford's problem (ii).
The Mathematical Gazette, 43(346):253–255, 1959.

Donald E. Knuth.
The Art of Computer Programming, Volume 4A: Combinatorial Algorithms, Part 1.
Addison-Wesley Professional, 2011.

Donald E. Knuth.
The Art of Computer Programming, Volume 4B: Combinatorial Algorithms, Part 2.
Addison-Wesley Professional, 1st edition, 2022.

Zan Pan.
Conjectures on the number of langford sequences.
preprint online at `https://eprint.arxitics.com/articles/langford.pdf`, March 2021.