An Exploration into Distinct Multiplicity Fixed Point Partitions

Alejandro Montemayor, Eden Tripp

Advisor: Dr. Jane Friedman

Fletcher Jones Summer Scholars

University of San Diego

Herbert Wilf's 6th Question on Distinct Multiplicity

Let T (n) be the set of partitions of n for which the (nonzero) multiplicities of its parts are all different, and write f(n) = |T (n)|. See Sloane's sequence A098859 for a table of values. Find any interesting theorems about f(n). The mapping that sends a partition of n to another partition of n in which the roles of parts and multiplicities are interchanged is a well defined involution on T (n), which is how I arrived at the study of this problem.

Introduction

Over the course of our research, we studied integer partitions in which all nonzero multiplicities are distinct (often referred to as Wilf partitions). Specifically, we observed the frequency of fixed points in the involution **t(p)**: a mapping that takes a partition p of n and interchanges parts and their corresponding multiplicities. Our work expands from Doron Zeilberger's generating-function approach which addressed Herbert Wilf's 6th unsolved question (see right), where he uses inclusion-exclusion over "forbidden equal multiplicities" to define a generating function for distinct multiplicity partitions.

Let n = 10

Distinct Multiplicity Partitions of n ($\text{part}^{\text{multiplicity}}$):

10^1, $8^1 1^2$, $7^1 1^3$, $6^1 2^2$, $6^1 1^4$, $5^1 1^5$, $4^2 2^1$, $4^1 3^2$, $4^1 1^6$, $3^3 1^1$, $3^2 1^4$, $3^1 2^2 1^3$, 2^5, $2^4 1^2$, $2^3 1^4$, $2^2 1^6$, $2^1 1^8$, 1^{10}

The involution t(p) swaps a partition p's parts and corresponding multiplicities, fixed points in this involution have the following forms:

- Diagonals: $i^i \rightarrow (i,i)$
- Off Diagonals: $i^j j^i \rightarrow (i,j)(j,i)$
- Combination of the above: $k^k i^j j^i \rightarrow (k,k)(i,j)(j,i)$

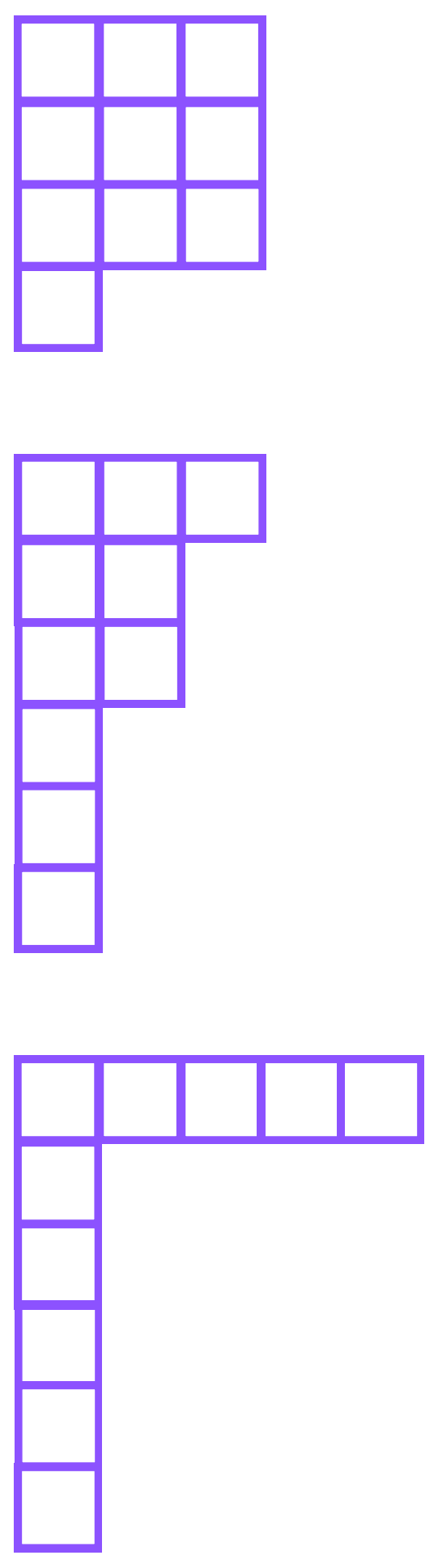
Distinct Multiplicity Fixed Points in the involution of p:

$3^3 1^1$, $3^1 2^2 1^3$, $5^1 1^5 \rightarrow \{(3,3),(1,1)\}, \{(3,1)(2,2)(1,3)\}, \{(5,1)(1,5)\}$

Notice if we swap the multiplicities and part :

$3^3 1^1$, $1^3 2^2 3^1$, $1^5 5^1 \rightarrow \{(3,3),(1,1)\}, \{(1,3)(2,2)(3,1)\}, \{(1,5)(5,1)\}$

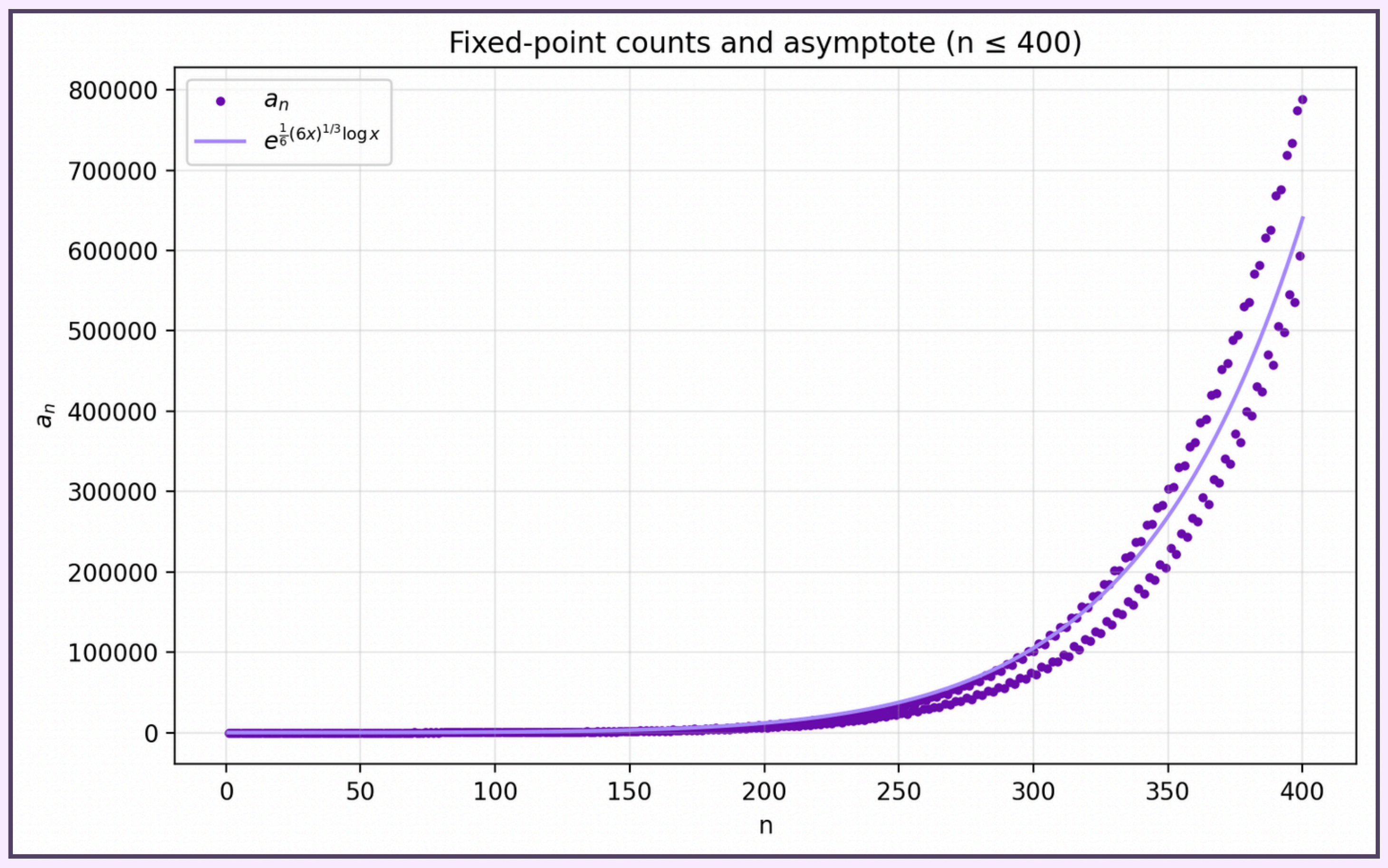
- Sets are unordered so the above collection of sets are equivalent



Analyzing Zeilbergers Work

We began our research by first familiarizing ourselves with generating functions. They become relevant in our research as they encode sequences as the coefficients of x^n in a formal power series. Doron Zeilberger's paper *Using Generatingfunctionology to Enumerate Distinct-Multiplicity Partitions* does exactly this, he finds first a recursive formula for $f_m(n;S)$ which is defined as the number of partitions of n with parts <= m, where all its multiplicities are distinct and none of the multiplicities belong to the set S. He then developed a closed generating function from this recurrence and Maple code to enumerate partitions in which all multiplicities are distinct. His approach involved adapting classical partition recurrences, introducing "forbidden multiplicity" sets, and using inclusion-exclusion on graphs to build explicit generating functions.

Our next step was translating Zeilberger's Maple code into SageMath/Python. This not only reproduced his enumerations, but also provided a flexible platform for us to build algorithms off of and visualize results.



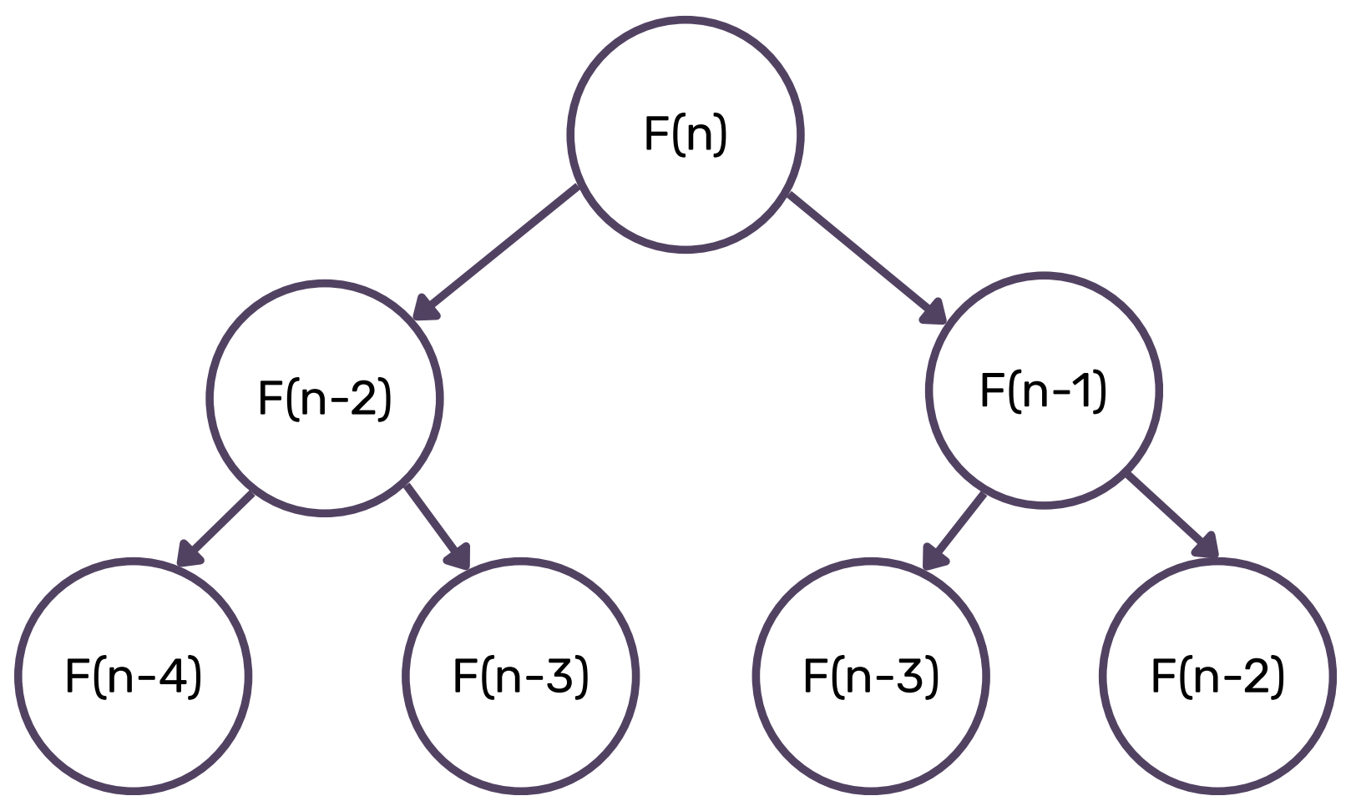
Fixed Points

Building on Zeilberger's process, we shifted focus to distinct multiplicity fixed points. We derived a recursive formula for the number of fixed points, implemented it in Sage, and verified its correctness against small cases using the online encyclopedia of integer sequences(A217605). While this recursion is effective and allows us to expand Sloane's sequence by approximately fifty terms, finding a closed-form generating function or optimizing the code would enable us to discover a multitude of additional terms.

$$G(S;x) = G(S \cup \{i\};x) + x^{i^2} \cdot G(S \cup \{i\};x) + \sum_{\substack{j=i+1 \\ j \notin S}}^{\lfloor n/(2i) \rfloor} x^{2ij} \cdot G(S \cup \{i\};x)$$

```
Compute fixed-point counts up to n = 400
How many times would you like to run the program: 3
loop_haf(400) – avg over 3 run(s): 82.9247s (min 82.3590s)
Running prefix check against precomputed values...
Compared terms 0..88: Everything matches!
```

Top Down Dynamic Programming



Optimizing the Code

After writing a basic recursion which models and counts all possible fixed points up to an integer N, we quickly realized that this approach is far too computationally expensive to feasibly produce large sequences. As a consequence of this, we opted for a Dynamic Programming approach, which splits the larger problem into smaller sub-problems whose results are combined at the end to produce our final desired answer. High level languages such as Python, which we wrote in, who feature built in data structures like dictionaries make Dynamic Programming more accessible, however provide more overhead relative to low-level languages. So, our next step in code optimization is to write this in a lower-level language such as C.

Sources

Zeilberger, D. (2012). *Using generatingfunctionology to enumerate distinct-multiplicity partitions*. arXiv:1201.4093.

Fill, J. A., Janson, S., & Ward, M. D. (2012). *Partitions with distinct multiplicities of parts: On an "unsolved problem" posed by Herbert Wilf*. The Electronic Journal of Combinatorics, 19(2), P18.

Wagner, S. (2013). *The number of fixed points of Wilf's partition involution*. The Electronic Journal of Combinatorics, 20(4), P13.

Pattern of Fours

When graphing a visual representation of the number of fixed points in t(p) corresponding and up to a number N, we noticed that the number of fixed points in this involution generally decreases in chunks of 4 (see below). This is something we plan to explore more in the future.