More thoughts after talking with Prof O (last friday)

- Is the distribution of displacement the same as the distribution of attempts? (attempts is the parking spot perspective for version of displacement. So for an occupied spot, how many cars attempt to park in that spot. Would need to do a little thinking on how this should extend to defect d preference lists) If I wanted to phrase this as a generating function question, it would look like $D(x_0, x_1, \cdots, x_m) = A(x_0, x_1, \cdots, x_m)$ where D is a generating function for displacement and A is a generating function for attepts? (this might not be quite right, but I'm moving on for now)
- Add probabalistic elements into code
- skim paper from prof o
- read paper by prof harris (quicksort, lucky - does this relate to connections to trees in Stanley and Yin?)
- Look through database!
- Write description of other map from parking functions to trees
- Make triangle for p-prime parking function thing - see if it's in the oeis

Notes from Reading [2]

- Main idea: number of preference lists with lucky $= n - 1$ is the same as the number of comparisons to perform quicksort
- One open note in the paper is to somehow understand this relation using underlying structures/relation between objects... In [4], in the generating function equality relating trees and parking functions, unlucky was related to nonleaders. If there was a map between the parking functions and trees that respected that generating function in some nice way, I wonder if you could somehow leverate the relation of lucky to leaders in the tree (with some factor of 2)
- Interesting that the number of parking functions with $n - 1$ lucky cars is the same as the number of non-parking function preference lists with $n - 1$ lucky cars.

Progress on code stuff

I made some random sampling methods for parking functions and prime parking functions, made random walk code in Testing file, worked on Iterate stats class so that it has all of the stats

Note: From experimental investigation - passed and displacement have the same sum since their sums both calculate total displacement, but they have different distributions. Example: 42111 has displacement distribution 30101 but passed distribution 12200

Worked on counting p-prime parking functions

There are 2 distinct ways that you can go about counting these, which correspond to the two ways that you can go about sampling from them. I know how to carry out the count for method 1. For method 2, there's still a missing piece between the equation I know I should get (via rearranging the result from method 1), and the counting strategies that I currently have.

- **Method 1**: Use $s = n - p$ possible spots to generate a preference list, and toss out the preference lists which are not prime parking functions.

  This gives the expression

  $$s^n - (s-1)^n - \sum_{i=1}^{s-1} \binom{n}{i} (i-1)^{i-1}(s-i)^{n-i}$$

  Note that s^n is the total number of choices, the $-(s-1)^n$ term is there because a parking function must have a 1 in it, and the ith term of the summation removes preference lists with the first break point at i (if the first break point was at i, the parking function would be a shuffle of a prime parking function of length i, and cars with preferences after i).

  For annotated with explanation version of this equation, see the whiteboard. For examples that helped me get to this expression see tablet notes on the right in green/pink

  Note that you can use Abel's binomial identity [1] (page 8) to rearrange this equation into a new form, which seems reminiscent of the other method of sampling from prime parking functions and inclusion/exclusion.

  $$\sum_{i=0}^{p} \binom{n}{i} (n-i-1)^{n-i-1}(d-i)^i(-1)^i$$

  Note that the $(i-d)^i$ part of this equation relates to oeis sequence A009999.

- **Method 2**: Sample from prime parking functions, and toss out the preference lists which are not p-prime.

  Note, counting wise, this is a headache, and there is a common error that I kept making. I'll try to explain the inclusion exclusion and the related count that I kept mistaking it for. It would be interesting to see if there was a straightforward way to modify the related count to properly count the correct thing. For notes see blue/pink on left of tablet notes, and whiteboard notes. Both places have the error.

  Ok. Enough caveats. We want to remove any prime parking functions that have the last p choices. Let $m_k(\pi)$ be the number of times that the preference k appears in π. Let us build a set of coefficients for inclusion exclusion

  $$E_C^V = \{\pi \in PPF_n | m_{v_i}(\pi) = c_i \text{ for each } i\}$$

This is just irritating notation for saying that $E_{V,C}$ is the set of prime parking functions where for each value in v_i in V, there are exactly c_i preferences for the spot v_i.

This lets you can build an inclusion exclusion kind of argument that runs over partitions rather than integers. Note that for prime parking functions, at most i cars can have preferences larger than $n-i$ (ex: no cars can prefer the last spot, at most 2 cars can have preferences larger than $n-2$). So, for example,

$$|PPF_{n,2}| = |PPF_n| - |E_1^{n-1}|$$

$$\begin{aligned} |PPF_{n,3}| = |PPF_n| &- |E_1^{n-1}| - |E_1^{n-2}| + |E_{1,1}^{n-1,n-2}| && \text{Largest part is 1} \\ &- |E_2^{n-2}| && \text{Largest part is 2} \end{aligned}$$

$$\begin{aligned} |PPF_{n,4}| = |PPF_n| &- |E_1^{n-1}| - |E_1^{n-2}| - |E_1^{n-3}| \\ &+ |E_{1,1}^{n-1,n-2}| + |E_{1,1}^{n-2,n-3}| + |E_{1,1}^{n-1,n-3}| \\ &- |E_{1,1,1}^{n-1,n-2,n-3}| && \text{Largest part is 1} \\ &- |E_2^{n-2}| - |E_2^{n-3}| \\ &+ |E_{2,1}^{n-2,n-3}| + |E_{2,1}^{n-3,n-2}| + |E_{2,1}^{n-3,n-1}| && \text{Largest part is 2} \\ &- |E_3^{n-3}| && \text{Largest part is 3} \end{aligned}$$

So this inclusion exclusion argument works, but the point of inclusion exclusion is to make life simpler, and this doesn't look simpler yet. This simplification is the step that I got tripped up on.

Note that when the subset whose behavior is defined forms a parking function, then the count is a shuffle of the number of arrangements of this subset with the number of prime parking functions for the rest of the list. Because of the constraint on the number of cars with preferences larger than $n-i$ in prime parking functions, this is the case for a lot of the time for small cases. The first time when it is not the case appears in $PPF_{n,4}$: $|E_2^{n-2}|$. Take $n=5$. Then we have to include all rearrangements of $11122, 11223, 11224$ (please excuse the 0 indexing in all of my notes). The count for the others using just a shuffle would only include rearrangements of $11122, 11224$ since the index 3 is included in the segment for which cars are already parked given the information you have fixed.

This is exactly where my error was. If you **incorrectly** assume that for any set E_C^V, you can count the number of values using a shuffle, you end up with the expression

$$\sum_{i=0}^{p} \binom{n}{i} (n-i-1)^{n-i-1} \left(\sum_{\pi \in PF_{n=p-i,m=i}} (-1)^{\textit{distinct entries in } \pi} \right)$$

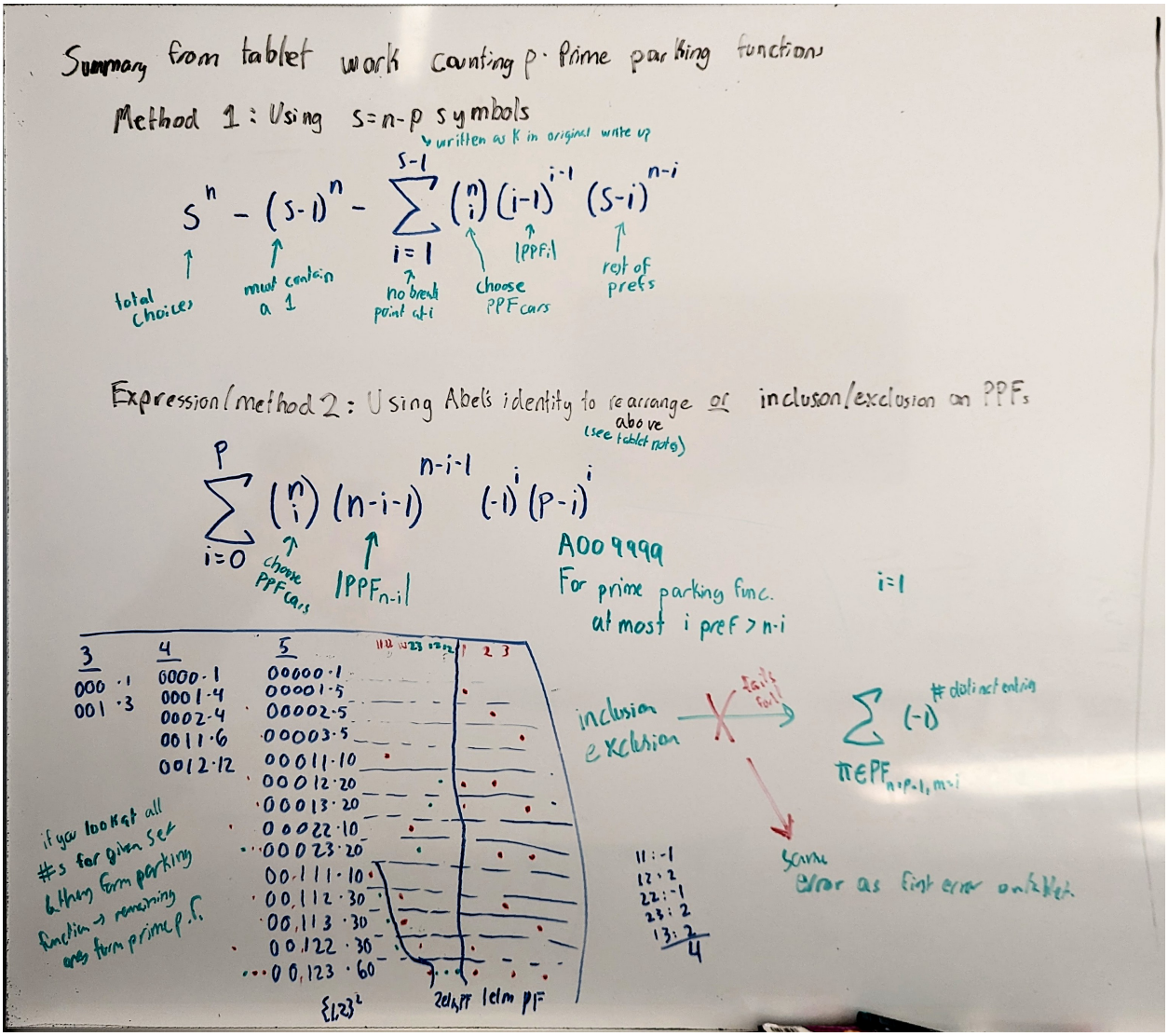
This is NOT a correct count. But I'm including it because it was a way of thinking about things that was close-ish, and I don't want to make the same mistake if I ever come back to these ideas.

If I were to think about using Method 2 and inclusion exclusion to arrive at the same count again, there are 2 different strategies I could try:

- Think about how to identify the special cases and take them into account to tweak the incorrect expression above (idk if that would go well)
- Look into the OEIS sequence A009999 more closely and see if there's a way of framing the inclusion exclusion that way

---

relevant notes:



6/26

I want to make the triangle!

p- Prime Parking Functions

length ↓ \ p-prime →	1 $(n-1)^{n-1}$	2 $(n-1)^{n-1}\cdot n(n-2)^{n-2}$	3 $(n-1)^{n-1}\cdot 2n(n-1)^{n-2} + \binom{n}{2}(n-3)^{n-3}$	4	5
2	1				
3	4	1			
4	27	11	1		
5	256	121	26	1	
6	3125	1589	453	57	1
7	46656	24781	8282	1576	120	1

$3^n - 2^{n-1}(n+2) - \binom{n}{3}$ $\quad 2^n - (n+1)$ $\quad 1$

For 2 row partitions

$1+5+10+10$

$(x+y)^n$ coeff w/ $x^i, i \geq 2$

000000 · 1
00000 1 · 6
0000 11 · 15
000 111 · 20
00 1111 · k = 2

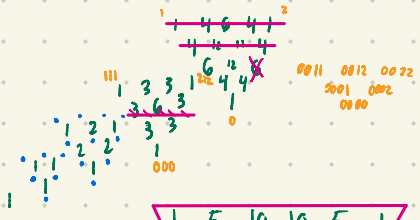
pascals Δ row sum $= 2^n$

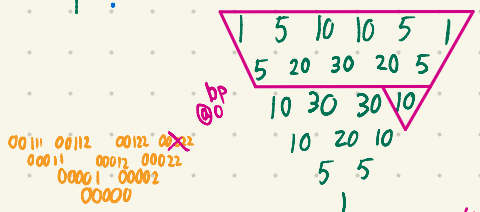
so for 2 part partitions $\Rightarrow 2^n - (n+1) = 2^n - 1^{n-1}(n+1)$

For 3 row partitions

$(x+y+z)^n$ coeff w/ $x^i, i \geq 2$ → like plane in pascal cube thing

$3^n -$





Total: $3^n - (2^n + 2^{n-1}\cdot n) - \binom{n}{3}$

$3^n - 2^{n-1}(2+n) - \binom{n}{3}$

Prime PFs not incl. last pel. : $(n-1)^{n-1} - \binom{n}{1}n(n-2)^{n-2} + \binom{n}{2}n(n-1)(n-3)^{n-3} - \binom{n}{3}n(n-1)(n-2)(n-4)^{n-4}$

$- 2\binom{n}{2}(n-3)^{n-3} + 3\binom{n}{2}(n-3)(n-4)^{n-4} - \binom{n}{3}(n-4)^{n-4}$

$+9\binom{n}{3} \quad -6\binom{n}{3}$

this expression is incorrect. need additional $-3\binom{n}{3}(n-4)^{n-4}$

for P=4

\# prime parking

- # exactly 2 n-1 $\quad n \cdot PPF_{n-1}$
- # exactly 1 n-2
- # exactly 1 n-3
- + # exactly 1, exactly 1 $\binom{n-1}{2}$ $\binom{n-1}{3}$ $\quad n\cdot(n-1)\,PPF_{n-2}$
- # exactly 1 3 $\quad n(n-1)(n-2)PPF_{n-3}$
- # exactly 2 n-2 $\quad \binom{n}{2} PPF_{n-2}$
- # exactly 2 n-3 $\quad \binom{n}{1} PPF_{n-2}$ → this line is a source of the error.
- + # exactly 2 n-2, 1 n-3 $\quad \binom{n}{2}(n-2)PPF_{n-3}$
- + # exactly 2 n-3, 1 n-2 $\quad \binom{n}{2}(n-2)PPF_{n-3}$
- + # exactly 2 n-3, 1 n-1 $\quad \binom{n}{2}(n-2)PPF_{n-3}$
- # exactly 3 n-3 $\quad \binom{n}{3} PPF_{n-3}$

For 4 row partitions

0,1,2,3, ... $(2,1,1,0) \to 0012$
$(2,1,0,1) \to 0013$

must include 2 x 0s

given 0s. if 2 x 0s: must include 1

given 0s, 1s. if 000 or 001: must include 2

total: 4^n

$-\left[\sum_{|\lambda|=3,\lambda\neq n}\binom{n}{0,\lambda} + \sum_{|\lambda|=3,\lambda=n}\binom{n}{1,\lambda}\right]$

$-\sum_{|\lambda|=2}\binom{n}{2,0,\lambda}$ $\quad \lambda \to n-2$

$-\binom{n}{3,0,0,n-3} - \binom{n}{2,1,0,n-3}$

$= -\binom{n}{n-3}(1+3)$

$= 4^n$

$= -3^n + n\cdot 3^{n-1}$

$= -\binom{n}{2}\cdot 2^{n-2}$

$= -\binom{n}{3}(1PPF_3 1)$

For K row partitions

$$K^n - (K-1)^n - \sum_{i=1}^{K-1}\binom{n}{i}|PPF_i|(K-i)^{(n-i)}$$

- K^n: total choices
- $(K-1)^n$: must contain a 0
- $\sum_{i=1}^{K-1}\binom{n}{i}|PPF_i|(K-i)^{(n-i)}$: no break point at i

---

This expression gives a way to recover the results from Conrad

$(K=n-1)$

$$K^n - (K-1)^n - \sum_{i=1}^{K-1}\binom{n}{i}|PPF_i|(K-1)^{(n-i)} \quad \text{of } K^{(i-1)^{i-1}} \text{ terms}$$

2 options for random:

Use partition which is S long : works

or

Use random prime parking Func. : works $\quad K^{K+d} - [K(K-1)^{K+d}] - \sum_{i=1}^{K-1}\binom{K+d}{i}(-1+i)^{i-1}(K-i)^{K+d-i}$ of $(K+d-1)^{K+d-1}$

std

$$K^{K+d} - (K-1)^{K+d} + \sum_{i=0}^{K+d}\binom{K+d}{i}(-1)(-1+i)^{i-1}(K-i)^{K+d-i} - \left[K^{K+d} + \sum_{i=K}^{K+d}\binom{K+d}{i}(-1)(-1+i)^{i-1}(K-i)^{K+d-i}\right]$$

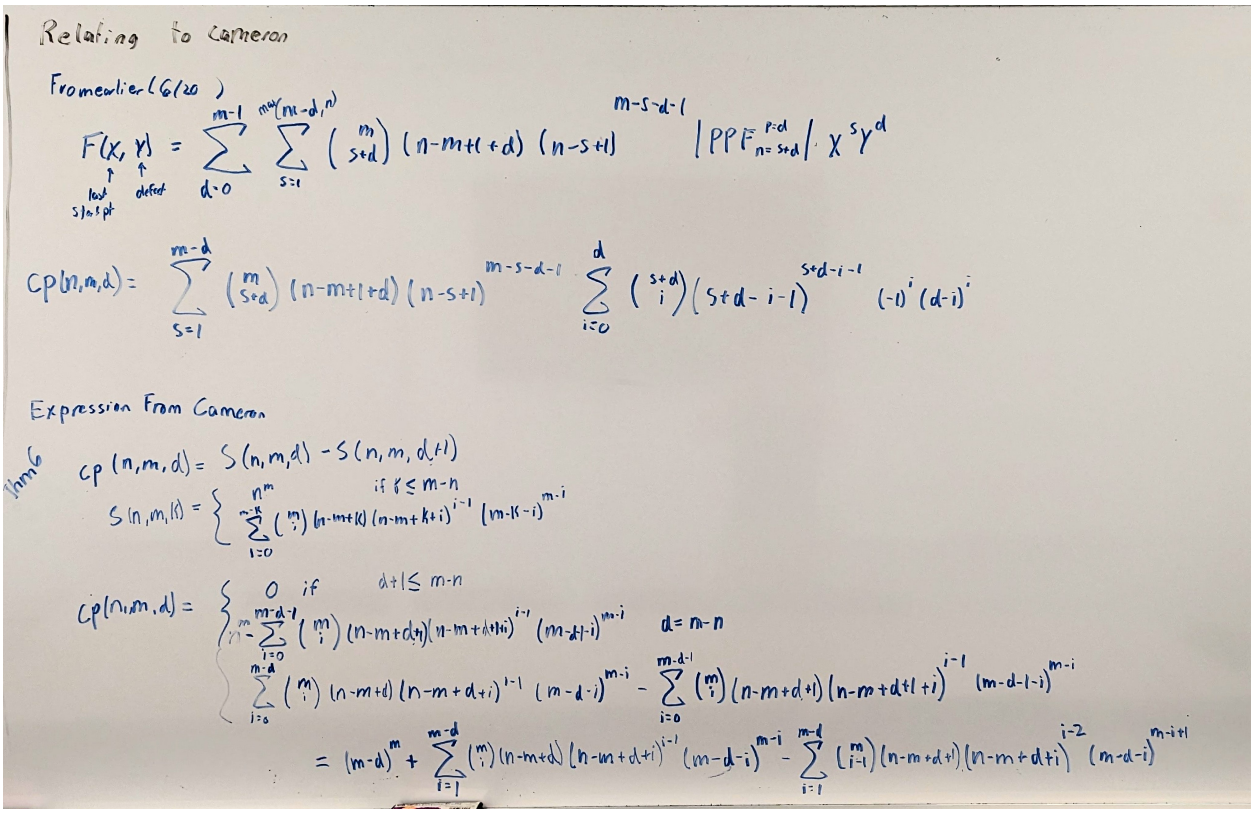
$(K-1)^{K+d}$ (labels, see comment, pg8)

$$= \sum_{i=K}^{K+d}\binom{K+d}{i}(-1+i)^{i-1}(K-i)^{K+d-i} = \sum_{i=0}^{d}\binom{n}{i}(n-i-1)^{n-i-1}(i-d)^i$$

↳ relates to A009999

Comparing Results to [1]

This is a fun result because it means that now I have a way to count defect d parking functions. I was curious if there was a nice way to show that it was the same count as the count in Cameron. In terms of the expressions... they are full of summations and have indices running around everywhere, so I stopped trying to find a way to explicitly show they were equal.



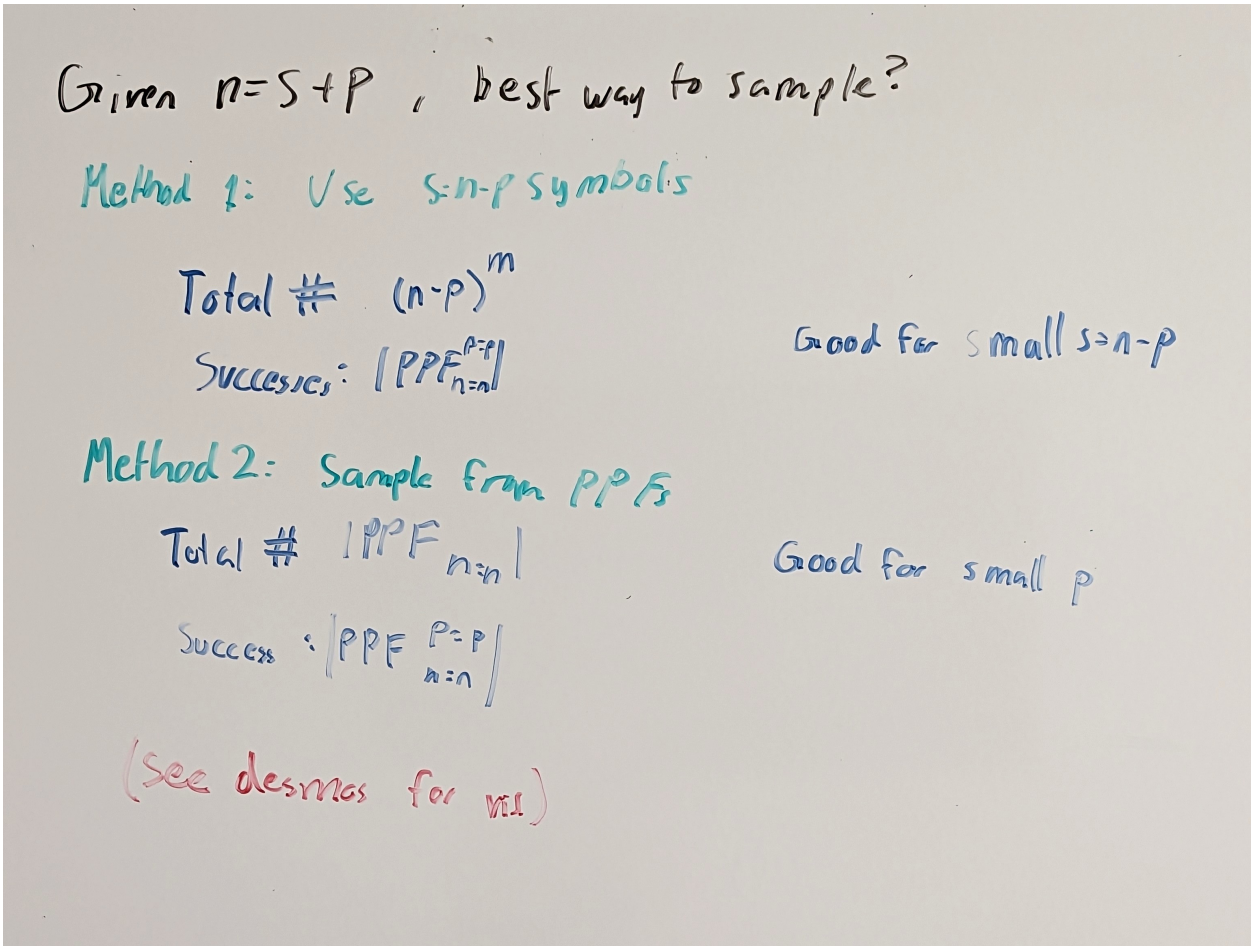
Instead I made a big table of values, and started poking around some of their asymptotic results too.

Here is a Desmos where you can see the proportion of all preference lists plotted for a bunch of defect values as $n = m$ grows. You can also play with c which is the difference between n and m. It's kind of surprising that each defect d seems to have a value of n for which it peaks in terms of the proportion of preference lists, and they trade off which defect has the highest proportion. For example:

Defect with highest proportion	Range for $n = m$	Range for $n = m + 1$
0	0-4	0-14
1	5-17	15-25
2	18-38	26-51
3	39-68	52-85
4	69-105	86-126

Also, to see counts that match Cameron's paper, just remove the divided by n^m in the expression for $P(n, m, d)$

For a given n and p, which way of sampling is better?



Here is another desmos which shows the proportion of the time that you will be successful for each sampling method. One of the main takeaways is that Method 1 continues to work fairly well for a while, while Method 2 falls off a lot faster. In terms of sampling, this means that the hardest lists to sample from will be those with relatively small defect (but not super small) since both sampling methods struggle.

Also one note, it seems that the first explicit formula for computing defect (from method 1) tends to behave better computationally for more of the time as well. This somewhat parallels the result for which behaves better sampling wise as well.

Also in this desmos are some expressions for the distribution that you pull s from, normalized a bit now that I have a more general expressions. Also, good news is that for small s, Method 1 works well, and that's one of the more commmon values of s. Bad news is for somewhat large s and medium small defect, this is the worst case scenario for sampling from p-prime parking functions.

Reading Ian's thesis - chapter on partition algebras

- Things kept reminding me of parking functions.
- I wonder if the Double Centralizer Theorem (pg 46) could be used to understand the representation theory of parking modules... In this context E would be $\mathbb{C}[n]^m$. Then I *think* the set of endomrphisms $\mathrm{End}_{\mathbb{C}}E = \mathbb{C}\, S_n \wr S_m$ (Thats also at the heart of why Prof O said that that is really the right group to think about! i think)

  If you let A be $\mathbb{C}S_m$, I wonder if the resulting centralizer/comutant is the diagonal subgroup. Is that more generally true if $End_{\mathbb{C}}E \cong \mathbb{C}H \wr G$? I think so? Also does this relate to Gelfand pairs?

  I suppose the result would be fun consequences for understanding the representation theory of parking functions as a $S_n \otimes S_m$ module maybe (pg 46-47) (where S_n is the diagonal subgroup, and the tensor is over $\mathbb{C}$ I think)? The representation theory of parking functions as an S_m module is something I've thought about but the representation theory of parking functions as an S_n module is something I haven't thought about as much. Also that leads to the question, is there some sort of diagram like the one on page 55 if you think about the diagonal subgroup where C_n is the one doing the acting? That should expand the other corresponding pair to be bigger than S_n... What else ends up in there?
- Page 50. Ian mentions that Gaetz and Ryba tweak the correspondence using elements which lump together all of the corsenings, and this makes the map that didn't used to be a module homomorphism into a module homomorphism. Why does this work??? Could something similar be applied to the map from preference lists to partial permutations? I'm skeptical but curious

Next steps:

Thinking for random walks:

- Make a list/partial list of random walks of interest. Focus on random walks with few fourier coefficients, statistics which are constant on cosets of the diagonal subgroup (esp anything to do with displacement distribution, other options include anything to do with passed distribution, anything to do with length of prime components), and initial distributions which are either really simple or also constant on cosets of the diagonal subgroup.
- (this is no longer a to do item, I realized I could do it while I was writing it down) Question from meeting with Prof O that I didn't have the braincells to answer in our meeting: If you have an indicator function on preference lists for when the ith car is lucky, is this compressible? The answer is yes! (and in a way that helps explain why nice basis is so nice) The indicator function is constant on cosets of the diagonal subgroup, so nonzero fourier coefficients are indexed by group elements whose entries sum to n. Also, the preferences of the cars after the ith car don't matter at all, so all of the nonzero fourier coefficients will be indexed by group elements with a 0 in the spot corresponding to cars after the ith car. This also behaves nicely with the action of the symmetric group S_{i-1} which rearranges the first $i-1$ cars since this will not change which spots are occupied upon the arrival of the ith car, so the fourier coeffecients are constant under the action of S_{i-1} rearranging the first $i-1$ cars. Command line code to see some nice outputs:

  ```
  CnmStat(6,6,get_ind(6,6,1)).print_by_value())
  ```

- Keep in mind how you could potentially use the extra information about the kinds of coefficients that you have (ie constant on reordering for statistics like total displacement)
- Read!

Code things: (also just see TODOs in code)

- Test some random walks that might be of interest by making plots. (along the way, it would be good to write a list of random walks of interest. Read for inspiration: Ian's thesis, and Hultman)
- Would be good to write code to sample from defect d preference lists

Questions/gaps relating to sampling from defect d things

- I still haven't proved for myself that $|PPF_n| = (n-1)^{n-1}$. I kind of just took Stanley's word for it that there is exactly one per diagonal coset in C^n_{n-1} after verifying a few examples

- It would be interesting if there was a way to see the expression resulting from Abel's identity as an inclusion exclusion expression relating to method 2 of sampling
- It would be nice to be able to see how the two explicit expressions counting prime parking functions relate to eachother... But that's a lot of indices. On the other hand, the fact that the expressions are somewhat hard to immediately reconcile does mean that I came up with the expression in a way which was meaningfully different I suppose. Also the Cameron paper references [3] on page 21, which I suppose I could also look at, but that's a lot of building context/understanding of polytopes
- Some part of me thinks the triangle enumerating p-prime parking functions is "of general interest" enough for the oeis, and it's not currently there. Maybe in addition to writing something about how to sample from defect d parking functions, I could also write out an entry about p-prime parking functions in the format that OEIS describes. Maybe someday I could even submit said sequence if some other people also think that the sequence is of general interest enough

References

[1] Peter J. Cameron, Daniel Johannsen, Thomas Prellberg, and Pascal Schweitzer. Counting defective parking functions, 2008.

[2] Pamela E. Harris, Jan Kretschmann, and J. Carlos Martínez Mori. Lucky cars and the quicksort algorithm, 2023.

[3] Jim Pitman and Richard Stanley. A polytope related to empirical distributions, plane trees, parking functions, and the associahedron, 1999.

[4] Richard P. Stanley and Mei Yin. Some enumerative properties of parking functions, 2023.