Horse Numbers, Rigging, and Scheduling

Benjamin Schreyer
benontheplanet@gmail.com

Abstract

The horse numbers, Fubini numbers, or Ordered Bell numbers count the total weak orderings $(<, >, =)$ on a set of n elements. *Constrained horse numbers* count orderings of n elements such that k are in a specific strong ordering relative to each other. Constrained horse numbers are expressed as a sum of horse numbers with weightings given by *the signed Stirling numbers of the first kind*. Considering the case of fully ordered constraint, a recurrence for the horse numbers is determined. Additionally, cardinality for unions and intersections of strong ordering or equivalence constraints on subsets is given. An application to enumeration of constrained task schedules is finally exemplified.

1 Introduction

The cardinality of totally ordered sets is considered. No incomparable element is allowed or calculated for.

1.1 Total Weak Ordering

If the finite set is $\{d, e, a, b, c \ldots\}$, then a weak ordering may be applied with symbols $<, >, =$, such as $a < d < e \cdots$, or $(c = d) < a \cdots$. The number of such orderings are known as the horse numbers, Fubini numbers, or ordered Bell numbers $H(n)$. A horse race is a combinatorial setting where ties may occur, hence horse numbers. Fubini's name is associated in relation to his theorem for integrals, where integrals over two variables may be taken as one (analogues to equality ordering), or performed separately in some strong order. Counting the ways to perform these integrals is an equivalent problem.

1.2 Total Strong Ordering

A strong ordering is a permutation of a set of elements. Permutation in the sense of only allowing the relations $<$ and $>$. Strong orderings are counted by the factorial.

1.3 Stirling Numbers of the First Kind

Given in a table of identities in *Concrete Mathematics* [1], the signed Stirling numbers of the first kind $s(l, m)$ give the coefficient on x^m in the falling factorial $(x)_l$ where $(x)_l = (x)(x-1)\cdots(x-l+1)$. The formula is $(x)_l = \sum_{m=0}^{l} s(l,m)x^m$. Usually, as is the case in *Concrete Mathematics* [1] these numbers appear when counting permutations with a set number of cycles. Here they appear for rigged horse races, and a recurrence for the ordered Bell numbers.

1.4 Constrained or Rigged Weak Ordering

Constrained horse numbers $B_k(n)$ count weak orderings such that k elements of the finite set are chosen, and constrained to follow a specific strong ordering. If the elements under total weak ordering are $x_1, x_2, \ldots, x_n$, such a strong ordering could be $x_1 < x_2 < \cdots < x_k$. Another type of constraint may be imposed by equality, for example $x_1 = x_2 = \cdots = x_k$. Both of these types will be counted, with equality being the easier case.

1.5 Example by Hand: Single Strong Ordering

The case $B_2(n)$ is given a short expression in terms of the horse numbers. $B_2(n)$ counts when $x_1 < x_2$, or two horses agree that one will forfeit to the other, and provides inspiration for the generalized case.

Theorem 1.

$$B_2(n) = \frac{H(n) - H(n-1)}{2} \tag{1}$$

Proof. The expression $[x_1 < x_2] \vee [x_1 > x_2] \vee [x_1 = x_2]$ is a tautology on over any set $x_1, x_2 \ldots x_n$ that is weakly ordered, so long as $n \geq 2$. The three conditions are mutually exclusive, so their union has cardinality that may be expressed as a sum of each of the three relation's cardinalities. Consider each case:

- Count for $x_1 < x_2$, this is $B_2(n)$.
- Count for $x_1 = x_2$, this is $H(n-1)$.
- Count for $x_1 > x_2$, this is $B_2(n)$ again by symmetry.

The sum of these counts is $H(n)$ because the expression as a whole is a tautology.

$$H(n) = 2B_2(n) + H(n-1) \tag{2}$$

Finally as was to be shown

$$B_2(n) = \frac{H(n) - H(n-1)}{2}$$

□

1.6 Shift Operators

Shift operators are used to formally show the number of orderings where $x_1, x_2, \ldots, x_k$ have a strong but not specific ordering can be expressed using $s(l, m)$, the Stirling numbers of the first kind.

Consider expressing the counting in terms of the left, right shift operators T_+, T_- on the sequence $H(0), H(1), \ldots, H(n+k)$. Shift operators are defined such that $T_+F(n) = F(n+1)$ and $T_-F(n) = F(n-1)$. Importantly $T_+T_- = T_-T_+ = I$ (I being the identity operation), so any product of shift operators may be abbreviated T_a where a is an integer and $T_aF(n) = F(n+a)$ so long as $F(n+a)$ is in the domain. In the case $F(n+a)$ is not part of the finite sequence $T_aF(n) = 0$. The shift operator is linear and commutes with integers acting as scalars.

2 General Rigged Orderings

2.1 Strong Ordering of $k \leq n$ Elements

Theorem 2.

$$B_k(n) = \frac{1}{k!} \sum_{j=0}^{k} s(k, k-j) H(n-j) \tag{3}$$

Proof. First remove or ignore the counting of k elements to be counted in strong ordering $x_1, x_2, x_3, \ldots, x_k$: $T_{-k}H(n)$.

- Reintroduce the element x_1 by increasing the argument to $H(n-k+1)$, then subtract any case where $x_1 \in \emptyset$. That is $(T_+)T_{-k}H(n)$
- Reintroduce the element x_2 by increasing the argument, then subtract any case where $x_2 \in \{x_1\}$. That is $(T_+ - 1I)T_{-k}T_+H(n)$
- Reintroduce the element x_3 by increasing the argument, then subtract any case where $x_3 \in \{x_1, x_2\}$. That is $(T_+ - 2I)(T_+ - 1I)T_+T_{-k}H(n)$
- $\cdots$
- Reintroduce the element x_k by increasing the argument, then subtract any case where $x_k \in \{x_1, x_2, \ldots, x_{k-1}\}$. The total is $(T_+ - (k-1)I) \cdots (T_+ - 2I)(T_+ - 1I)T_+T_{-k}H(n)$.

Now all elements are counted, but those from $x_1, x_2, \ldots, x_k$ have had mutual equalities removed from the count, such that any counted ordering has $x_1, x_2, \ldots, x_k$ strongly ordered. The falling factorial appears with argument T_+ and k terms.

$$(T_+ - (k-1)I)\cdots(T_+ - 2I)(T_+ - 1I)T_+T_{-k}H(n) \tag{4}$$

By commuting shift operators and integer scalars, the count is:

$$T_{-k}(T_+)_k H(n) \tag{5}$$

Where $(x)_n$ is the falling factorial of x with n terms. As discussed in the introduction 1.3, Stirling number $s(n,a)$ expresses the integer coefficient of x^a in $(x)_n$. The count is now expressed as follows.

$$T_{-k}[\sum_{j=0}^{k} s(k,j)T_j]H(n) \tag{6}$$

The formula applies because T_a commute and repetition of T_a may be treated as would exponentiation of a polynomial variable. The effect of the shift operators is now trivial upon $H(n)$.

$$\sum_{j=0}^{k} s(k,j)H(n-k+j) \tag{7}$$

By reindexing the sum.

$$\sum_{j=0}^{k} s(k,k-j)H(n-j) \tag{8}$$

The number of arrangements of $x_1, x_2, \ldots, x_n$ where $x_1, x_2, \ldots, x_k$ are strongly ordered have been counted. Given the strongly ordered subset count it is straightforward to determine $B_k(n)$ by dividing by $k!$, since there are $k!$ total strong orderings of $x_1, x_2, \ldots, x_k$, and only one is desired per the definition of $B_k(n)$.

$$B_k(n) = \frac{1}{k!}\sum_{j=0}^{k} s(k,k-j)H(n-j)$$

□

Note constrained horse numbers may be efficiently computed by applying a discrete Fourier domain convolution. This is the case because $B_k(n)$ has an expression as a weighted sum of $H(m)$, $m \leq n$, with constant coefficients for fixed k.

Another interesting notation is given using the relation between falling factorials and binomial coefficients on the equation for strong orderings (5) after it has been divided by $k!$ to pick a specific ordering. The notation is inspired by the following manipulations.

$$B_k(n) = \frac{T_+(T_+ - 1)\cdots(T_+ - k + 1)}{k!} H(n-k) \tag{9}$$

$$B_k(n) = \frac{T_+!}{k!(T_+ - k)!} H(n-k) \tag{10}$$

$$B_k(n) = \binom{T_+}{k} H(n-k) \tag{11}$$

If a useful interpretation can be given to this expression, it should be undertaken, to qualify the meaning of division here. There is ambiguity between inversion of a linear operation or division in the sense of counting. For now this is only notational.

2.2 Union and Intersection of Disjoint Strong Constraints

Corollary 3. *The cardinality of weakly ordered arrangements of the elements $x_1, x_2, \ldots, x_n$ under condition $[x_{a_1} < x_{a_1+1} < \cdots < x_{a_1+A_1-1}] \wedge [x_{a_2} < x_{a_2+1} < \cdots < x_{a_2+A_2-1}] \wedge \cdots \wedge [x_{a_N} < x_{a_N+1} < \cdots < x_{a_N+A_N-1}]$ of N specifically strongly ordered subsets of sizes $A_1, A_2, \ldots, A_N$ where $\{x_{a_j}, x_{a_j+1}, \ldots, x_{a_j+A_j-1}\} \cap \{x_{a_i}, x_{a_i+1}, \ldots, x_{a_i+A_i-1}\} = \emptyset \ \forall i, j$:*

$$\frac{1}{A_1!A_2!\cdots A_N!}[(T_+)_{A_1}(T_+)_{A_2}\cdots(T_+)_{A_N}]T_{-(\sum_{j=1}^{N} A_j)}H(n) \tag{12}$$

This is equivalent to the following by the notation used earlier (11).

$$[\prod_{j=1}^{N}\binom{T_+}{A_j}]H(n - \sum_{j=1}^{N} A_j) \tag{13}$$

Proof. The procedure of the proof of the second theorem (2) may be repeated for any amount of disjoint subsets which are strongly ordered, because counting only involves the number of elements from the respective subset already reintroduced, and the total count. The subsets being disjoint allows this to remain trivial. An additional provision of allowing $H(0), H(1), \ldots, H(n + (\sum_{j=1}^{N} A_j))$ under shift operation is needed. Factorial division for the size of each strongly ordered subset again accomplishes specifying a strong ordering, rather than over counting all strong orderings of a subset. The resulting formula is given above (13). □

The cardinality for union of conditions immediately follows by the inclusion-exclusion principle.

2.3 Union and Intersection of Equality Constraints

Corollary 4. *The first theorem exemplified that dealing with conditions $x_a = x_b$ is a simple reduction in the effective number of elements being ordered, for $x_a = x_b$, $a \neq b$, this is $H(n-1)$. For an intersection of such equalities, the number of elements that are removed from counting is determined by counting the number of equivalence classes introduced by the intersection of equality constraints, k, the cardinality is then $H(n+k-m)$, where m is the number of elements included non-trivially ($x_a = x_a$ is trivial) in the intersection of equality constraints. The cardinality for union of equality constraints is easily expressed via the inclusion-exclusion principle.*

3 Horse Numbers

3.1 A Complete Alternating Recurrence

Corollary 5.

$$H(n) = n! - \sum_{j=1}^{n} s(n, n-j) H(n-j) \tag{14}$$

Proof. $B_n(n) = 1$, if the set is $x_1, x_2, \ldots, x_n$, and $x_1 < x_2 < \cdots < x_n$ the number of arrangements is 1. It follows from the second theorem (2) :

$$1 = \frac{1}{n!} \sum_{j=0}^{n} s(n, n-j) H(n-j) \tag{15}$$

Which may be rearranged after the substitution $s(n, n) = 1$ to the result.

$$H(n) = n! - \sum_{j=1}^{n} s(n, n-j) H(n-j)$$

□

4 Remarks

Problems of optimal job shop scheduling or production lines may be more easily characterized in the size of their search space using the given mathematical tools. Consider scheduling n jobs, to be executed in a large shop, but g sub-groupings of the jobs have a required strong order. For example testing of an engine may only occur after some fuel pumps are repaired, this is a strong ordering in the schedule. Other shop tasks allow for equality ordering, say painting a chassis or producing spare parts, these tasks are independent and may be scheduled to co-occur, not having any prerequisite steps at the model scale considered.

4.1 Worked Example: Finite Factory Schedule

Production problems are usually interesting in the case where the production capacity per time step m is smaller than the demanded total production over multiple time steps. Consider a factory floor with n tasks to complete, each taking up one of the m sets of equipment available. A second constraint is introduced. A group of tasks of size k must occur in a strictly enforced order as described above. A computer scientist is trying to optimize the factory schedule, and wants to know the number of possible scheduling plans they must search through. Translation operator based counting is extremely useful.

First count the production plans where there are at most m tasks executed in a given time step. To do this, remove all tasks, then add them back one by one, subtracting those in which a variable takes on equality (same time step scheduling) with more than $m-1$ other task. Tasks are weakly ordered because they may be co-occurring, so the base counting function is $H(n)$.

Remove all tasks:

$$T_{-n}H(n) \tag{16}$$

Add back the first m tasks:

$$T_m T_{-n} H(n) \tag{17}$$

- Now add back the $m+1$'th task, and subtract out any case where it is equal to m or more tasks.

$$(T_+ - \binom{m}{m}) T_m T_{-n} H(n) \tag{18}$$

- Now add back the $m+2$'th, subtracting out cases where it is equal to at least m tasks.

$$(T_+ - \binom{m+1}{m} - \binom{m+1}{m+1})(T_+ - \binom{m}{m}) T_m T_{-n} H(n) \tag{19}$$

- $\cdots$

- Now add back the n'th, removing cases where it is equal to m or more tasks.

$$(T_+ - [\sum_{l=m}^{n-1} \binom{n-1}{l}]) \cdots (T_+ - \binom{m+1}{m} - \binom{m+1}{m+1})(T_+ - \binom{m}{m}) T_m T_{-n} H(n) \tag{20}$$

The partial sums of binomial coefficients are clearly involved, to count ways which the introduced task may violate the m co-scheduling limit. Following this pattern to its conclusion:

$$[\prod_{j=m}^{n-1}(T_+ - [2^j - \sum_{l=0}^{m-1}\binom{j}{l}])]H(m) \tag{21}$$

Note the sum of binomial coefficients has been written in complement form to reduce computations, since m is considered smaller than n in interesting cases.

For use in more general cases this could also be written as follows.

$$\hat{C}_{n,m}H(n) \tag{22}$$

Where $\hat{C}_{n,m}$ is defined to match the first expression (21).

$$\hat{C}_{n,m} = [\prod_{j=m}^{n-1}(T_+ - [2^j - \sum_{l=0}^{m-1}\binom{j}{l}])]T_{m-n} \tag{23}$$

Now the counting operator for a constrained subset of k is applied, which has been given in a less expanded form before in the formal section (11) as:

$$\hat{B}_k = \binom{T_+}{k}T_{-k} \tag{24}$$

Note both of these operators contain T_0, this should not be surprising since at some point, all manipulated elements should be included, when dealing with these two constraints.

The total count of possible schedules for a factory limited to m concurrent units, with a required strong ordering on k specified tasks out of n total tasks is then:

$$\hat{B}_k\hat{C}_{n,m}H(n) \tag{25}$$

A clean result for a nontrivial constraint that may have sent some straight to taking the term enumerative combinatorics too seriously. Clean means computation and size of both operators are calculations on polynomials of length order n. Multiplication may be expedited by discrete Fourier methods for polynomial multiplication. Adding more strongly ordered tasks of length $k', k'' \ldots$ is trivial by using more $\hat{B}_{k'}, \hat{B}_{k''} \ldots$ operators. Processes that are dependent on two or more steps may be added, as all inequalities may be broken into unions and intersections of pairwise relations, but this can require the inclusion-exclusion principle.

4.2 Tools Developed

It is interesting to consider where else in combinatorics the shift operator may yield simplified understanding or proofs. Interpreting the binomial form (11) is expected to be useful to this end. Here the treatment revealed an over arching structure allowed the existing counting properties of $H(n)$ to be utilized, without considering a recurrence or explicit definition for $H(n)$. More concretely, $B_k(n)$ has some recurrence that could be shown to yield the second theorem (2), by defining $B_k(n)$ then making a connection to a recurrence for $H(n)$ or $s(l, m)$. The possibility to make adjustments to counting allowed by the shift operator means the second theorem (2) is revealed without such low level considerations.

The shift operator also allows for certain recurrences such as the discrete coupled oscillators problem of physics to be solved as a linear differential equation after separation of variables is done. This occurs by letting the shift operator act like a differential operator on indices n as $T_{\pm} = e^{\pm \frac{d}{dn}}$, revealing the dispersion relation of the system. Are there interesting combinatorial problems that can be solved with tools of differential equations by such an abuse of notation?

The number of multiplicative partitions may also be exposed for cases where

This new recurrence can prove $H(n)$ eventually periodic.

5 Acknowledgments

1. William Gasarch, bringing the rigged horse race problem to my attention, and providing critique of proofs, writing, and format.

2. Nathan Constantinides, for checking the single strong ordering case 1 numerically for small n.

3. Elan Fisher, for discussion of applications.

References

[1] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik, Concrete Mathematics, Second Edition, *Pearson Education* (1994), 259-264.

2020 *Mathematics Subject Classification*: 06A05.

Keywords: Shift operator, Ordered Bell numbers, Fubini numbers, Stirling numbers of the first kind, Weak ordering, Constrained weak ordering, Scheduling.

(Concerned with sequence A000670.)