

Weighted Event-Growth Sequences: Enumeration and Bijections

Igor Kleiner*

Technion – Israel Institute of Technology, Haifa, Israel
Department of Data Science, HIT – Holon Institute of Technology
59 Eliezer Hoffien St., PO Box 305, Holon 5810201, Israel
igorkl@hit.ac.il
ORCID: 0000-0002-8361-8505

Abstract

Restricted-growth words are built by letting a prefix control which values may appear next. We introduce the weighted Event-Growth model, a three-parameter family based on three intrinsic relations between a new entry and its history: it may repeat the current value, return to another previously used value, or appear for the first time. Assigning nonnegative rewards to these Level, Return, and New events determines the future admissible range. Although admissibility depends on the entire prefix, we prove that all continuation multiplicities and state updates collapse to two integer coordinates. This yields a universal succession rule, a reversible Event-Slot encoding, a common differential system, and an integral formula for the shifted exponential generating functions. Four natural one-parameter families have exponential, Bell, Euler, and rational kernels. We construct direct bijections with colored increasing forests of height at most two, cycle-spider assemblies, partial injections, and Dowling partitions. Return-Growth and Reuse-Growth appear as neighboring points of the Euler family, placing several previously separate restricted-growth rules within one small-state and bijective framework.

Keywords. restricted growth words; generating trees; insertion encodings; Bell numbers; Euler numbers; partial injections; Dowling partitions.

2020 Mathematics Subject Classification. 05A05; 05A15; 05A19.

1 Introduction

Restricted-growth words encode a growing object by allowing the next integer to depend on the prefix already constructed. A broad statistic-controlled formulation was developed by Mansour and Vajnovszki [15]. The question here is more specific: what structure appears when the rule remembers only how the newest value relates to the history of the word?

There are exactly three possibilities. A new entry may equal the current last value, it may be an old value different from the current one, or it may not have appeared before. We call these events *Level*, *Return*, and *New*. For example, the four steps after the initial zero in

00203

*Corresponding author.

have types Level, New, Return, New. This trichotomy belongs to the word itself; it does not depend on any growth rule.

We introduce the *weighted Event-Growth model* by assigning fixed nonnegative integer rewards a, b, c to Level, Return, and New. If $L(w), R(w), N(w)$ are the accumulated event counts, put

$$g_{a,b,c}(w) = aL(w) + bR(w) + cN(w).$$

A weighted Event-Growth word satisfies

$$w_1 = 0, \quad 0 \leq w_i \leq 1 + g_{a,b,c}(w_1 \cdots w_{i-1}) \quad (i \geq 2).$$

Thus an event does not merely describe what has happened: its reward enlarges the interval of values available at later steps. The reward of the new entry is earned only after that entry has been admitted.

Two motivating cases show why this distinction is useful. If only Return earns one unit, the prefix 00 has earned no reward, so its next entry is at most 1. If both Level and Return earn one unit, the same prefix has already earned one unit, so the value 2 becomes legal. The resulting Return-Growth and Reuse-Growth rules are locally different, yet the present framework places them at the neighboring parameter points $(0, 1, 0)$ and $(1, 1, 0)$. One goal of the paper is to explain why such different local rules nevertheless share a common continuation structure.

The first answer is a small-state theorem. Although legality is defined from the whole prefix, future branching collapses to two nonnegative integers: the number of old values other than the current one and the number of admissible values not yet used. These are exactly the numbers of Return and New choices. Their updates depend only on the next event type, giving one succession rule for every reward triple (a, b, c) .

The paper develops this observation in four directions.

- (i) We prove the two-coordinate branching theorem and the resulting coordinatewise inclusion order on parameter space. In particular, the eight zero–one reward vectors form an ordered Boolean core rather than a disconnected list of examples.
- (ii) We construct a reversible Event-Slot encoding. It records the ranked choice made at each extension and therefore retains enough information to reconstruct the original word, not merely its state counts.
- (iii) The slot model leads to a master differential system and a common integral formula for shifted exponential generating functions. Fixing the Return and New rewards at zero or one produces four one-parameter families with exponential, Bell, Euler, and rational kernels.
- (iv) We give direct recursive bijections from important fibers to colored increasing forests of height at most two, cycle–spider assemblies, partial injections, and Dowling partitions. These correspondences turn the state variables and event counts into concrete structural statistics on familiar combinatorial objects.

Several parameter points have classical interpretations of their own. Adjacent-equality growth gives the repetition or level sequences of Callan and Mansour [7, 12]; growth by changes gives Arndt’s change sequences recorded in OEIS A000522 [3]; and growth by first appearances gives the usual restricted-growth encoding of set partitions. These examples are useful anchors for the parameter space. The main point of the weighted model is that they, together with Return-Growth, Reuse-Growth, and further weighted rules, inherit one continuation mechanism and can be compared along the same parameter directions.

Section 2 develops the definition through examples. Sections 3–5 establish the two-coordinate theorem, the reversible Event-Slot encoding, and the master counting equations. Sections 6–10 develop the four principal fibers and their structural realizations. Section 11 collects the eight zero–one specializations, and Sections 12–13 place the model in the surrounding literature and record open directions.

2 Weighted Event-Growth Words

We first separate the event that occurs, the reward assigned to it, and the interval of values made available by the accumulated reward. Keeping these three notions distinct makes it possible to check a word under different rules without changing its event history.

2.1 Three kinds of memory event

Let $w = w_1 \cdots w_n$ be a nonempty word over $\mathbb{Z}_{\geq 0}$ with $w_1 = 0$. For $2 \leq i \leq n$, exactly one of the following events occurs:

$$\begin{aligned} L_i &: w_i = w_{i-1}, \\ R_i &: w_i \neq w_{i-1} \text{ and } w_i \in \{w_1, \dots, w_{i-1}\}, \\ N_i &: w_i \notin \{w_1, \dots, w_{i-1}\}. \end{aligned}$$

We call these events Level, Return, and New. The subscripts specify individual positions; $L(w)$, $R(w)$, and $N(w)$ denote the numbers of positions of the three types. The initial zero has no event type and contributes to none of these counts.

In 0000, all three steps are Level. In 010, the final zero is Return, since it is old but differs from the immediately preceding value. In 012, both steps are New. These examples describe event types only; whether a word is legal will depend on the rewards introduced below.

Two statistics from those models will be compared with these counts. Their *return statistic* is

$$\text{ret}(w) = \#\{2 \leq i \leq n : w_i \neq w_{i-1} \text{ and } w_i \in \{w_1, \dots, w_{i-1}\}\},$$

and their *reuse statistic* is

$$\text{rep}(w) = \#\{2 \leq i \leq n : w_i \in \{w_1, \dots, w_{i-1}\}\},$$

the number of entries whose value has already appeared. Also let $d(w)$ be the number of distinct values occurring in w .

Proposition 2.1. *For every such word of length n ,*

$$L(w) + R(w) + N(w) = n - 1,$$

and

$$R(w) = \text{ret}(w), \quad L(w) + R(w) = \text{rep}(w) = n - d(w), \quad N(w) = d(w) - 1.$$

Proof. The three event types partition the $n - 1$ positions after the first. An entry is not a first occurrence precisely when it is Level or Return. Each distinct value has exactly one first occurrence, so the other $n - d(w)$ entries are counted by rep . Finally, the word starts with one distinct value and each New event introduces one more. $\square$

2.2 Rewards and the legal interval

Fix $a, b, c \in \mathbb{Z}_{\geq 0}$ and define

$$g_{a,b,c}(w) = aL(w) + bR(w) + cN(w).$$

We refer to this accumulated reward as the *growth budget*. It is not spent when an entry is appended. A Level event adds a , a Return event adds b , and a New event adds c to the budget for later steps. For fixed parameters we write $g(w)$ for $g_{a,b,c}(w)$.

Definition 2.2. For $n \geq 1$, a *weighted Event-Growth word* of length n is a word of nonnegative integers satisfying

$$w_1 = 0, \quad w_i \leq 1 + g(w_1 \cdots w_{i-1}) \quad (2 \leq i \leq n).$$

We write $\mathcal{G}_n(a, b, c)$ for this class. For a legal nonempty prefix w , its *legal interval* is the set

$$I(w) = \{0, 1, \dots, 1 + g(w)\}$$

of values that may be appended next.

There is a single word of length one, namely 0, and its budget is zero. Thus every model has the same two words of length two, 00 and 01. The rewards begin to distinguish the classes at the next step. Every nonempty prefix of a legal word is legal by definition. We do not need an empty-word budget.

2.3 A small gallery of models

Example 2.3 (Return-Growth: $(a, b, c) = (0, 1, 0)$). The word 01020 has event sequence N, R, N, R . Before appending its fourth entry, the prefix 010 has one Return, so $g = 1$ and $I(010) = \{0, 1, 2\}$. This is why the new value 2 is allowed. Before that Return, the bound was only 1. This is the Return-Growth model.

Example 2.4 (Reuse-Growth: $(a, b, c) = (1, 1, 0)$). The word 00203 has event sequence L, N, R, N . Both ways of reusing an old value earn one unit. Here is the complete check; the budget in each row belongs to the prefix before the indicated entry is appended.

prefix	budget	legal interval	next entry	event
0	0	$\{0, 1\}$	0	L
00	1	$\{0, 1, 2\}$	2	N
002	1	$\{0, 1, 2\}$	0	R
0020	2	$\{0, 1, 2, 3\}$	3	N

The final word has $L = 1, R = 1, N = 2$ and budget 2. Since $L + R = \text{rep}$, this is precisely the Reuse-Growth rule.

Example 2.5 (Ordinary restricted growth: $(0, 0, 1)$). Only New events earn reward. After a prefix with d distinct values, the next entry is at most d . In this model the used values are always $\{0, \dots, d-1\}$: this holds initially, and the only possible new value under the bound is d . This induction is what identifies the rule with the usual restricted-growth encoding of set partitions, rather than merely a rule involving the number of distinct values.

For example, 001202 has events L, N, N, R, R . Grouping positions with equal values gives the blocks $\{1, 2, 5\}$, $\{3\}$, and $\{4, 6\}$. Conversely, numbering blocks by increasing least element, starting at zero, recovers the word.

Example 2.6 (The full event model: $(1, 1, 1)$). Every event earns one unit. A prefix of length m has budget $m - 1$, so the next entry may be any value in $\{0, \dots, m\}$. These words are exactly the *subexcedant sequences*, defined by

$$0 \leq w_i \leq i - 1 \quad (1 \leq i \leq n).$$

There are $1 \cdot 2 \cdots n = n!$ of length n .

Example 2.7 (A weighted point outside the zero-one choices). Take $(a, b, c) = (2, 1, 3)$ and again $w = 00203$. The successive prefix budgets, including the initial and final prefixes, are

$$0, \quad 2, \quad 5, \quad 6, \quad 9.$$

The four entries appended are therefore checked against upper bounds 1, 3, 6, 7, respectively, and all are legal. The final counts $L = 1, R = 1, N = 2$ give

$$g_{2,1,3}(w) = 2 \cdot 1 + 1 \cdot 1 + 3 \cdot 2 = 9,$$

so the next legal interval is $\{0, 1, \dots, 10\}$. The same events now earn different amounts of future freedom.

2.4 The same word under different rewards

Table 1 compares the word 00203 across several choices. Its event sequence is always $LNRN$; its legality is not.

(a, b, c)	rewards assigned to	status of 00203
$(0, 1, 0)$	Return only	illegal: the third entry 2 exceeds the bound 1
$(1, 1, 0)$	Level and Return	legal
$(0, 0, 1)$	New only	illegal: the third entry 2 exceeds the bound 1
$(1, 0, 1)$	Level and New	legal
$(1, 1, 1)$	all three events	legal
$(2, 1, 3)$	the weighted example	legal

Table 1: The event history is fixed by the word, whereas the rewards determine whether it is legal.

The comparison in Table 1 reflects a general order on the parameter space. Increasing a reward cannot invalidate a word that was already legal.

Proposition 2.8 (Coordinatewise monotonicity). *Let $a, b, c, a', b', c' \in \mathbb{Z}_{\geq 0}$ satisfy*

$$a \leq a', \quad b \leq b', \quad c \leq c'.$$

Then, for every $n \geq 1$,

$$\mathcal{G}_n(a, b, c) \subseteq \mathcal{G}_n(a', b', c').$$

Proof. For every nonempty word u ,

$$g_{a',b',c'}(u) - g_{a,b,c}(u) = (a' - a)L(u) + (b' - b)R(u) + (c' - c)N(u) \geq 0.$$

If $w \in \mathcal{G}_n(a, b, c)$, then at each position $i \geq 2$ its defining inequality gives

$$w_i \leq 1 + g_{a,b,c}(w_1 \cdots w_{i-1}) \leq 1 + g_{a',b',c'}(w_1 \cdots w_{i-1}).$$

Hence the same word is legal for (a', b', c') . □

In particular, within the zero-one parameter cube, turning on one additional event reward gives a literal inclusion of word classes. The cube used later is therefore more than a catalogue of eight specializations: its coordinate directions record enlargements of the admissible classes.

2.5 A few immediate checks

If $(a, b, c) = (0, 0, 0)$, the budget remains zero and all entries after the first are chosen independently from $\{0, 1\}$. Hence $|\mathcal{G}_n(0, 0, 0)| = 2^{n-1}$. The full event model gives $|\mathcal{G}_n(1, 1, 1)| = n!$. Both follow from the same elementary product.

Proposition 2.9 (Diagonal weights). *For integers $r \geq 0$ and $n \geq 1$,*

$$|\mathcal{G}_n(r, r, r)| = \prod_{j=0}^{n-2} (2 + rj),$$

where the empty product for $n = 1$ is 1.

Proof. A prefix of length m has $m - 1$ events and budget $r(m - 1)$. It has exactly $2 + r(m - 1)$ legal continuations, independently of its values. Multiply these numbers for $m = 1, \dots, n - 1$. $\square$

The diagonal cases need no information beyond length. Off the diagonal, the next question is how the legal interval splits among the three event types.

Example 2.10 (One prefix, split into its legal types). For $(a, b, c) = (1, 0, 1)$ and $w = 00203$, we have $L = 1, R = 1, N = 2$ and $g = L + N = 3$. Thus

$$I(w) = \{0, 1, 2, 3, 4\}, \quad \underbrace{\{3\}}_{\text{Level}}, \quad \underbrace{\{0, 2\}}_{\text{Return}}, \quad \underbrace{\{1, 4\}}_{\text{New}}.$$

In particular, New does not mean larger than every previous value: 1 is still new because it has never appeared. Figure 1 records the same split.

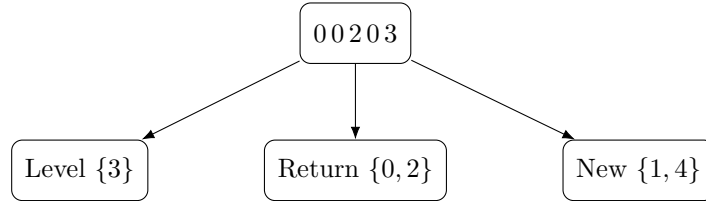


Figure 1: The legal values of Example 2.10 split into one Level value, two Return values, and two New values.

Before counting old choices in general, we need to know that an old value never becomes illegal.

Lemma 2.11 (Persistence of old labels). *Every value occurring in a legal prefix w belongs to $I(w)$.*

Proof. The value zero belongs to $I(w)$ directly. If a positive value x first occurs at position j , then $j \geq 2$ and its admission gives

$$x \leq 1 + g(w_1 \cdots w_{j-1}).$$

All rewards are nonnegative, so g is nondecreasing under extension. Consequently $x \leq 1 + g(w)$, as required. $\square$

3 The Two-Coordinate Growth Theorem

A Level continuation has only one candidate, the current last value. By Lemma 2.11, it is always legal. We therefore need to count two other kinds of choice: old values different from the current value, and legal values that are still unused.

For a legal prefix w , define

$$p(w) = N(w), \quad q(w) = g(w) + 1 - N(w).$$

The first count equals the number of distinct values minus one. The second subtracts all used values from the legal interval, as the following argument shows.

Theorem 3.1 (Two-coordinate branching). *The legal next values consist of exactly one Level value, $p(w)$ Return values, and $q(w)$ New values. In particular, both coordinates are nonnegative integers.*

Proof. There are $N(w) + 1$ distinct old values. All are legal by Lemma 2.11; one is the current last value, leaving $N(w) = p(w)$ Return choices. The interval $I(w)$ contains $g(w) + 2$ values, so its unused values number

$$g(w) + 2 - (N(w) + 1) = q(w).$$

These sets partition $I(w)$. □

Call $(p(w), q(w))$ the *state* of w . A *succession rule* lists, for each state, the states of its children in the generating tree together with their multiplicities. The rule below has the same shape for every choice of rewards; only the increments change.

Corollary 3.2 (Succession rule). *For fixed a, b, c , a state (p, q) of a legal prefix has the following transitions, where the number before each arrow is its multiplicity:*

$$\begin{aligned} L : 1 &\longrightarrow (p, q + a), \\ R : p &\longrightarrow (p, q + b), \\ N : q &\longrightarrow (p + 1, q + c - 1). \end{aligned}$$

The initial state is $(0, 1)$. A transition of multiplicity zero is omitted.

Proof. Level and Return leave N unchanged and add a and b to g , respectively. New adds one to N and c to g . These changes give the displayed updates; their multiplicities come from Theorem 3.1. The initial word 0 has $g = N = 0$. □

The zero-multiplicity convention matters when $c = 0$. If $q = 0$, there is no New extension; the formula does not create a child with coordinate -1 . For instance, the binary prefix 01 at $(a, b, c) = (0, 0, 0)$ has only a Level and a Return continuation.

Example 3.3 (Counting with the succession rule). Take $(a, b, c) = (0, 1, 0)$, the Return-Growth point, where the rule reads $L: 1 \rightarrow (p, q)$, $R: p \rightarrow (p, q + 1)$, $N: q \rightarrow (p + 1, q - 1)$. Start from the single state $(0, 1)$ and record, for each length, how many words sit at each state:

n	states, with multiplicities	$ \mathcal{G}_n(0, 1, 0) $
1	$(0, 1)$	1
2	$(0, 1), (1, 0)$	2
3	$(0, 1), 2 \times (1, 0), (1, 1)$	4
4	$(0, 1), 3 \times (1, 0), 3 \times (1, 1), (1, 2), (2, 0)$	9

At $n = 3$ the state $(1, 0)$ is occupied by 001 and 011: both have used their only New value and can continue only by Level or Return, so each contributes two words to the next row. The word 010, at state $(1, 1)$, has earned one Return reward and contributes three. No word had to be listed to obtain the counts 1, 2, 4, 9, which agree with the separately studied Return-Growth enumeration.

The two coordinates are sufficient for all future branching because both the multiplicities and the updates use only the current state. These coordinates summarize the branching, not the word itself. For $(a, b, c) = (2, 1, 0)$, the legal words 002 and 003 both have state $(1, 2)$. Their admissible New values are $\{1, 3\}$ and $\{1, 2\}$, respectively. Thus the state determines the number of choices, not their numerical values. To recover individual words, we must also record the choices made within each event type.

4 Event-Slot Histories

We now build a structure that records those individual choices. A *slot* is an available position for a future insertion, not an integer value of the word. It has type L , R , or N . At each step one active slot is filled with the next time label $1, 2, \dots$, and new active slots are attached to that filled position. The time label records when a choice was made; it is unrelated to the numerical value appended to the word.

We retain the whole history: all filled positions with their time labels and incoming types, their ordered child slots, and the unfilled slots. Thus a history is a typed rooted forest with empty slots at its boundary, not just the populations of active types. Initially the forest consists of an L -slot and an N -slot. No time label is used for the initial word value zero.

To order the choices, give the two root slots addresses 0 and 1. An address is a finite tuple of nonnegative integers. When a slot at address α is filled, its children receive addresses $\alpha 0, \alpha 1, \dots$ in their specified order. Here concatenation appends an integer symbol, not a decimal digit. Active slots of a fixed type are ordered lexicographically by their addresses.

Definition 4.1 (Event-Slot replacement rules). Filling an L -slot creates one L -slot followed by a ordered N -slots. Filling an R -slot creates one R -slot followed by b ordered N -slots. Filling an N -slot creates one R -slot followed by c ordered N -slots. An *Event-Slot history* is any finite history obtained from the two initial slots by these rules, with consecutive increasing time labels.

Example 4.2 (Slot addresses). If the initial L -slot at address 0 is filled when $a = 2$, its children are the L -slot 00 and the N -slots 01, 02. The original N -slot 1 is still active. Its rank among the active N -slots is now three, since $01 < 02 < 1$. Filling 01 later creates an R -slot 010 and, if $c > 0$, the N -slots 011, $\dots$, 01 c . For $c = 0$ there are no new N -slots.

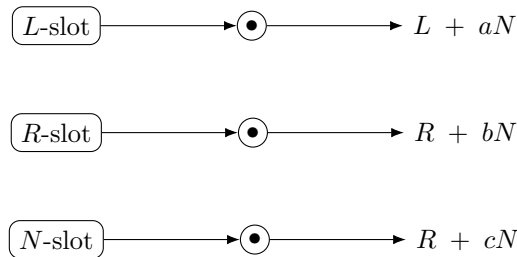


Figure 2: The three Event-Slot replacement rules. Filling a New slot consumes that slot, which accounts for the $q \mapsto q + c - 1$ update.

Several correspondences below are *recursive bijections*: maps defined by induction on the length of the word, in which each legal extension of a prefix is matched to one insertion into the object encoding that prefix, and the inverse removes the most recently inserted element. Such a map is a bijection once the two operations are shown to undo each other on every word and on every object of the target class.

Theorem 4.3 (Event-Slot rank bijection). *For every $a, b, c \in \mathbb{Z}_{\geq 0}$ and $n \geq 1$, there is a recursive bijection between $\mathcal{G}_n(a, b, c)$ and Event-Slot histories with time labels $1, \dots, n-1$. The active populations are*

$$1 \text{ } L\text{-slot}, \quad N(w) \text{ } R\text{-slots}, \quad g(w) + 1 - N(w) \text{ } N\text{-slots}.$$

Proof. The word 0 corresponds to the initial two unfilled slots. Suppose a prefix has already been encoded with active populations $1, p, q$. A Level extension fills the unique L -slot. For a Return extension, order its p legal values increasingly and match them to the p active R -slots in address order. For New, do the same with the q unused legal values and the active N -slots. Fill the matched slot with the next time label.

Each replacement has exactly the population update of Corollary 3.2. In particular, filling an N -slot consumes one such slot before creating c new ones. This proves inductively that all required choices exist and that the construction has the asserted populations.

For the inverse, note first that the active populations of an arbitrary history depend only on the sequence of types of its filled vertices: each replacement rule changes the populations by the fixed amounts of Corollary 3.2, whichever slot of that type was filled. Now take any nonempty history and locate its largest time label. It was inserted last, so every child slot it created is still unfilled. Remove that vertex and its child slots, and restore the incoming slot at its original address and with its original type. The remaining object is a shorter history. Recover its word recursively. In this restored history, find the rank of the restored slot among active slots of its type. For type R or N , choose the legal Return or New value of that rank for the recovered prefix; for type L , use its last value. The population invariant guarantees that the value exists. Appending it gives a legal word and refilling its matched slot recreates the deleted vertex. Induction proves both compositions are the identity, including the empty-history base case. $\square$

Example 4.4 (A full encoding and its last inverse step). Use $(a, b, c) = (1, 0, 1)$ and $w = 00203$. The table lists all active slots after each insertion. The first row is the initial configuration.

prefix	time label	event	filled address	L -slots	R -slots	N -slots
0	–	–	–	0	$\emptyset$	1
00	1	L	0	00	$\emptyset$	01, 1
002	2	N	1	00	10	01, 11
0020	3	R	10	00	100	01, 11
00203	4	N	11	00	100, 110	01, 111

For the second insertion, the new values are 1, 2, so choosing 2 fills the second N -slot, at address 1. For the last insertion the new values are 1, 3, so choosing 3 fills 11, again the second N -slot. Conversely, deleting time label 4 restores 11. After the shorter word 0020 has been recovered, its restored rank two identifies 3 among 1, 3. The final populations alone would not have supplied this information.

5 Joint Enumeration and the Master Flow

The succession rule can now be used without keeping individual words. We first retain all three event counts. Introduce marking variables x, y, z and, with the parameters a, b, c fixed, define

$$F_n(x, y, z) = \sum_{w \in \mathcal{G}_n(a, b, c)} x^{L(w)} y^{R(w)} z^{N(w)}.$$

Thus x, y, z mark Level, Return, and New, respectively. Partial derivatives, written $\partial_x, \partial_y, \partial_z$, multiply a monomial by its corresponding exponent while lowering that exponent by one.

Theorem 5.1. *For $n \geq 1$,*

$$F_{n+1} = (x + z)F_n + axz \partial_x F_n + byz \partial_y F_n + (yz + (c - 1)z^2) \partial_z F_n, \quad F_1 = 1.$$

Proof. A word with counts (L, R, N) has one Level extension, N Return extensions, and

$$q = aL + bR + (c - 1)N + 1$$

New extensions. Their weights are the original monomial multiplied by x , by Ny , and by qz , respectively. Summing these three contributions gives the recurrence. $\square$

The combined coefficient q counts choices and is nonnegative on legal words even when $c = 0$; the negative derivative term then removes values that are no longer new. As a check,

$$F_2 = x + z, \quad F_3 = x^2 + (a + 2)xz + yz + cz^2.$$

In particular, $|\mathcal{G}_3(a, b, c)| = a + c + 4$.

For total enumeration it is more convenient to track the two state coordinates. Introduce variables u, v marking p, q , and define

$$P_n(u, v) = \sum_{w \in \mathcal{G}_n(a, b, c)} u^{p(w)} v^{q(w)}.$$

Proposition 5.2.

$$P_{n+1} = v^a P_n + uv^b \partial_u P_n + uv^c \partial_v P_n, \quad P_1 = v.$$

Proof. On a monomial $u^p v^q$, the three operators give $u^p v^{q+a}$, $pu^p v^{q+b}$, and $qu^{p+1} v^{q+c-1}$, exactly the three transitions. Differentiation makes the last contribution zero when $q = 0$. $\square$

In the Return model, $P_3(u, v) = v + 2u + uv$. The coefficient 2 records two different words in state $(1, 0)$; setting $u = v = 1$ gives the total 4. The polynomial packages the state multiplicities from Example 3.3.

Write

$$D_{a, b, c} = v^a + uv^b \partial_u + uv^c \partial_v.$$

The term v^a means multiplication by v^a , so $P_n = D_{a, b, c}^{n-1} v$. For example, $P_2 = v^{a+1} + uv^c$. This compact operator is a particular instance of the first-order combinatorial operators studied in [5, Section 9, Note 25, Proposition 24].

We use an *exponential generating function*, or EGF, in which the coefficient for size m is divided by $m!$. The relevant size for slot histories is the number $n - 1$ of inserted time labels. Accordingly, define the *shifted EGF*

$$H_{a, b, c}(t) = \sum_{n \geq 1} |\mathcal{G}_n(a, b, c)| \frac{t^{n-1}}{(n-1)!}. \quad (1)$$

All generating-series identities are formal in t , with rational coefficients or coefficients in the polynomial ring of the indicated marking variables. No analytic convergence assumption is required.

The slot interpretation gives a direct explanation for the counting equations. To turn the two-root slot construction into equations, it is convenient to separate the contribution of each slot type. Allow a history to start with a single slot rather than the original pair. Let $M(t), U(t), V(t)$ be the EGFs of histories starting with one L -, R -, or N -slot, respectively. Each includes the unfilled slot as its unique object of size zero. These auxiliary histories need not themselves encode words; the original pair does.

Theorem 5.3 (Master flow). *The single-slot series are the unique formal solutions of*

$$U' = UV^b, \quad V' = UV^c, \quad M' = MV^a, \quad U(0) = V(0) = M(0) = 1.$$

The word-counting series is

$$H_{a,b,c}(t) = M(t)V(t).$$

The system reduces to integrations and inversion of a formal series with nonzero linear coefficient.

Proof. In a nonempty single-slot history the root is filled by the smallest time label. Delete that root. Its ordered child slots carry histories of their respective types, on disjoint subsets of the remaining labels. After the root is removed, the remaining time labels are distributed among the ordered child histories; within each child history only their relative order matters. This is why the corresponding exponential generating functions multiply. Thus an L -root leaves one L -history and a distinguishable N -histories, an R -root leaves one R -history and b N -histories, and an N -root leaves one R -history and c N -histories. Removing the smallest label corresponds to differentiating the EGF; this is the boxed product of [9, Section II.6], and it gives the three equations. Their coefficients determine the formal solutions recursively from the constant terms.

The original two root slots are distinguishable and carry an L -history and an N -history on complementary label sets. Their EGF is therefore MV . Theorem 4.3 identifies its size- $n - 1$ coefficient with the length- n word count.

For the reduction, U and V are units and $V'(0) = 1$. Dividing the first two equations gives $dU/dV = V^{b-c}$. Put $\rho = b - c + 1$ and define, as a series about $v = 1$,

$$f(v) = \begin{cases} 1 + (v^\rho - 1)/\rho, & \rho \neq 0, \\ 1 + \log v, & \rho = 0. \end{cases}$$

Then $U(t) = f(V(t))$, and

$$t = \int_1^{V(t)} \frac{ds}{s^c f(s)}, \quad M(t) = \exp \left(\int_1^{V(t)} \frac{s^{a-c}}{f(s)} ds \right).$$

All denominators have nonzero constant term at $s = 1$; $\log s$ means its formal expansion there. The first integral has linear coefficient one in $V - 1$, so it can be inverted uniquely. This determines V , then U and M . $\square$

6 Four One-Parameter Families

The equations for U, V do not involve the Level reward. Fix b, c and allow a to range over the nonnegative integers. We call this family of word models the (b, c) -*fiber*. Its common single- N -slot

series $V_{b,c}(t)$ will be called its *kernel*. The terminology refers only to this parameter organization: a fiber is a family of models, and its kernel is the shared auxiliary series.

Figure 3 summarizes the two parameter geometries used below. For each fixed pair (b, c) , the vertical direction varies the Level reward a and gives one fiber. The levels $a = 0$ and $a = 1$ form the *Boolean core* of the parameter space: the eight zero–one reward vectors, collected in Section 11. By Proposition 2.8, every solid arrow that increases one coordinate is an inclusion of word classes; the dashed vertical rays continue the four fibers beyond the Boolean core.

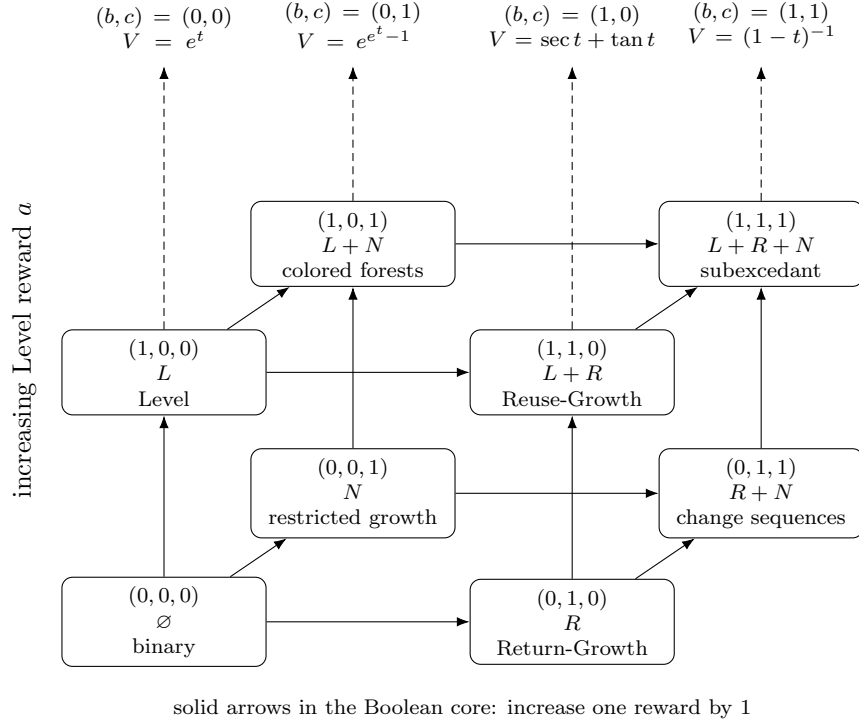


Figure 3: Parameter space of the four zero–one (b, c) fibers. The two displayed levels $a = 0, 1$ are the Boolean core. Solid arrows increase one reward and therefore represent class inclusions by Proposition 2.8; dashed rays continue each fixed- (b, c) fiber to $a \geq 2$. The labels above the rays are the common kernels.

The four choices $(b, c) \in \{0, 1\}^2$ have particularly simple kernels. To retain Level counts, introduce a separate marking variable ℓ and define

$$H_{a,b,c}(t; \ell) = \sum_{n \geq 1} \left(\sum_{w \in \mathcal{G}_n(a,b,c)} \ell^{L(w)} \right) \frac{t^{n-1}}{(n-1)!}.$$

Here ℓ marks Level events. Setting $\ell = 1$ gives (1).

Theorem 6.1. *The four kernels are*

$$V_{0,0}(t) = e^t, \quad V_{0,1}(t) = e^{e^t - 1}, \quad V_{1,0}(t) = \sec t + \tan t, \quad V_{1,1}(t) = \frac{1}{1-t}.$$

For every fixed pair $b, c \in \mathbb{Z}_{\geq 0}$ and every $a \in \mathbb{Z}_{\geq 0}$,

$$H_{a,b,c}(t; \ell) = V_{b,c}(t) \exp \left(\ell \int_0^t V_{b,c}(s)^a ds \right). \quad (2)$$

Proof. If $b = 0$, the equation for U gives $U = e^t$. For $c = 0$, integrating $V' = e^t$ with $V(0) = 1$ gives $V = e^t$. For $c = 1$, integrating $V'/V = e^t$ gives $V = e^{e^t-1}$.

For $(b, c) = (1, 0)$, the reduction in Theorem 5.3 gives $U = (1 + V^2)/2$, so $V' = (1 + V^2)/2$, $V(0) = 1$. The series $\sec t + \tan t$ satisfies this equation and initial condition. For $(b, c) = (1, 1)$, the equations for U, V agree, so $U = V$ and $V' = V^2$, giving $V = (1 - t)^{-1}$.

Only L -slot insertions contribute a Level event. Marking them by ℓ changes their equation to $M' = \ell M V^a$ and leaves U, V unchanged. Integrating $M'/M = \ell V^a$ and multiplying by the initial N -slot factor V proves (2). $\square$

(b, c)	kernel $V_{b,c}(t)$	name	specialization section
$(0, 0)$	e^t	exponential	Section 10
$(0, 1)$	e^{e^t-1}	Bell	Section 7
$(1, 0)$	$\sec t + \tan t$	Euler	Section 9
$(1, 1)$	$(1 - t)^{-1}$	rational	Section 8

Table 2: Four fibers obtained by fixing the two zero-one rewards b, c and varying a . The names describe their kernels, not necessarily their full counting series.

For example, $V_{1,1}$ is rational, whereas $H_{0,1,1} = e^t/(1 - t)$ is not. The integral factor in (2) must not be omitted when interpreting the family.

The next sections turn the state counts into concrete structures. In the Bell fiber, the three events become vertices at specified depths in a forest. In the rational and exponential fibers, Level is distinguished from the combined class of Return and New, and the resulting choices become component or block insertions. The Euler fiber then places Return-Growth and Reuse-Growth at consecutive values of the Level reward.

7 The Bell Fiber: Colored Height-Two Forests

Fix $(b, c) = (0, 1)$. Then $q = aL + 1$: each Level event provides a further choices of where a New event can occur, while each New event provides one Return choice. A forest makes these two stages visible.

For $n \geq 1$, write $[n] = \{1, \dots, n\}$, and set $[0] = \emptyset$. A rooted forest on $[n]$ is *increasing* if a parent's label is smaller than its child's. Children are not ordered. Roots have depth zero; height at most two means that every vertex has depth 0, 1, or 2. In the forests used here, vertex 1 is the distinguished root. Edges from it to depth 1 are uncolored. Every edge from an ordinary root to depth 1 independently receives one of the colors $1, \dots, a$, and edges from depth 1 to depth 2 are uncolored. There is no bound on the number of children of a vertex. When $a = 0$, ordinary roots must be isolated.

Theorem 7.1. *There is a recursive bijection between $\mathcal{G}_n(a, 0, 1)$ and the increasing colored forests just defined, of height at most two with vertex 1 distinguished. Under the bijection,*

$$L = \#\{\text{ordinary roots}\}, \quad N = \#\{\text{depth-1 vertices}\}, \quad R = \#\{\text{depth-2 vertices}\}.$$

Proof. Start with the word 0 and the single distinguished vertex 1. At each extension add the next largest vertex. Level creates an ordinary root. For New, the available attachments are the distinguished root, followed by the ordinary roots in increasing label order, with colors $1, \dots, a$ at

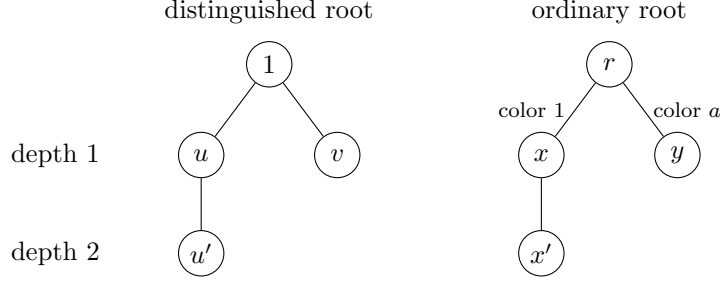


Figure 4: Geometry of the Bell fiber for $a \geq 1$. Level events create ordinary roots, New events create depth-1 vertices by choosing a root and, for ordinary roots, one of a colors, and Return events create depth-2 vertices.

each ordinary root in that order. There are $1 + aL = q$ such choices. Match them by rank to the increasingly ordered legal New values. For Return, order the depth-1 vertices by label and match them to the legal Return values in increasing order. There are $N = p$ choices, and the new vertex is attached at depth 2.

Each insertion produces a permitted increasing forest and preserves the three displayed counts. Conversely, in any such forest the largest vertex is a leaf or an isolated ordinary root. Delete it. An ordinary root records Level; a depth-1 vertex records New, together with its parent and, if applicable, its color; a depth-2 vertex records Return and its parent. The smaller forest stays in the same class, since its distinguished vertex 1 is never deleted unless it is the base object. Recover the smaller word recursively. In the attachment order of that smaller forest, the recorded parent and color or parent label supply the appropriate rank, hence the appended word value. These operations undo one another, proving bijectivity. $\square$

At $a = 1$, the colors carry no choices, and distinguishing vertex 1 adds no choice either: it is a root in every increasing forest. The target is therefore the known class of increasing forests of height at most two. This class, its root/depth insertion recurrence, and its counting series are recorded in OEIS entry A187761, with contributions from 2013 [2]. The correspondence above identifies the three word events with its vertex depths and extends the construction to the specified colors.

For $a = 1$, the running word 00203 gives roots 1 and 2 and edges

$$2 \longrightarrow 3, \quad 3 \longrightarrow 4, \quad 2 \longrightarrow 5.$$

The Level step created root 2; the two New steps attached vertices 3 and 5 to it; and the Return step attached vertex 4 below vertex 3. The counts are $L = 1, N = 2, R = 1$, as in the word. Deleting vertex 5 records its parent 2, the second root-attachment choice in the smaller forest, and recovers the last New value 3.

For a genuinely colored example, take $a = 2$. After the prefix 00, the New values are 1, 2, 3. They correspond, in order, to the distinguished root, the ordinary root 2 with color 1, and the ordinary root 2 with color 2. Thus 002 and 003 produce the same underlying forest, but the edge $2 \rightarrow 3$ receives different colors.

8 The Rational Fiber: Cycle-Spider Assemblies

Fix $(b, c) = (1, 1)$. Here Return and New earn the same reward, so we combine them when matching insertions. The target objects are *assemblies*: unordered collections of components on disjoint label sets whose union is $[m]$. The assembly on $[0]$ is empty.

There are two component types. A directed cycle is a cyclic order of its labels, identified up to rotation but not reversal; a singleton cycle is a loop. An a -arm *list-spider* consists of its minimum label as a center and an ordered tuple of a lists containing the remaining labels exactly once. Arms may be empty, and entries within an arm need not increase. When $a = 0$, the only spider is a singleton. A singleton spider and a singleton cycle are different component types.

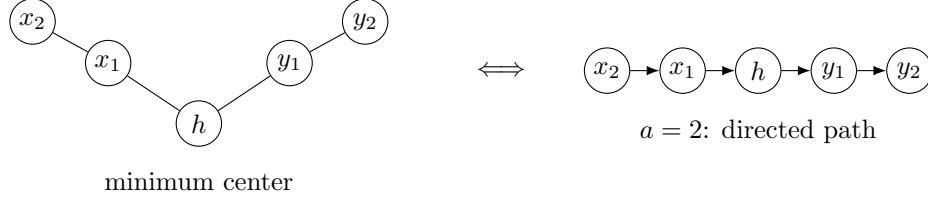


Figure 5: A two-arm list-spider with minimum center h becomes a directed path: reverse the first arm, pass through h , and then read the second arm.

Theorem 8.1. *For $a, m \in \mathbb{Z}_{\geq 0}$, the words in $\mathcal{G}_{m+1}(a, 1, 1)$ are in recursive bijection with assemblies on $[m]$ of directed cycles and a -arm list-spiders. The statistic L is the number of spider components.*

Proof. The word 0 corresponds to the empty assembly. Suppose m labels have been inserted and there are k spiders. Since $L = k$ and $L + R + N = m$, the word budget is $m + (a - 1)k$. There is one Level value, and there are

$$m + (a - 1)k + 1$$

non-Level values. Level creates a singleton spider with the new largest label.

For the other extensions there is one site creating a singleton cycle. A cycle of size s has s sites, one on each directed edge, including the loop when $s = 1$. A spider of size s with arm lengths $\lambda_1, \dots, \lambda_a$ has $\sum_j (\lambda_j + 1) = s + a - 1$ list-insertion sites, since $\sum_j \lambda_j = s - 1$. For $a = 0$, the spider is a singleton and has no such sites. Summing over components, and adding the new-cycle site, gives $m + (a - 1)k + 1$.

Order the sites as follows: the new-cycle site first, then existing components by increasing minimum label. In a cycle, order edge sites by the label preceding the edge. In a spider, order by arm number and then by gap position, starting with the gap nearest the center. Match these sites to the legal non-Level values in increasing order. Inserting the new largest label preserves the minimum-center condition and all component definitions.

For the inverse, delete the largest label. If it is a singleton spider, remove that component and record Level. If it is a singleton cycle, remove the cycle and record the new-cycle site. Otherwise it lies on a cycle edge or in a spider arm. Remove it and restore the corresponding edge or gap; deleting from a two-cycle restores a singleton loop. In a nonsingleton spider the largest label cannot be its minimum center, so the spider remains valid. Recover the smaller word recursively. The restored site has a unique rank in the smaller assembly, identifying the non-Level value. These are inverse operations on every target assembly. $\square$

Because Return and New are deliberately merged into one ordered set of non-Level choices, the assembly records L directly as the number of spider components. The inverse history still recovers every Return/New choice, even though the two event counts are not separate static component statistics of the assembly.

At $a = 1$ a spider is simply a list that begins with its minimum, and there are $(s - 1)!$ of these on s labels, as many as directed cycles. Both component types therefore have exponential generating

function $-\log(1-t)$, so the assemblies on $[m]$ number $(m+1)!$, in agreement with the subexcedant count $|\mathcal{G}_{m+1}(1,1,1)| = (m+1)!$ of Example 2.6.

At $a = 2$ and $m = 2$, there are seven target objects. Four use two singleton components, each independently a cycle or a spider; one is the two-cycle; and two are spiders with center 1 and label 2 on the first or second arm. Their number agrees with the seven words of length three for these rewards. This count would be wrong if the two singleton types were identified.

A concrete word shows how the ordered sites encode the actual choice. For $a = 2$, the prefix 00 creates the singleton spider centered at 1. Its non-Level values 1, 2, 3 correspond to a new singleton cycle, the first arm, and the second arm. Appending 2 puts label 2 in the first arm. For the prefix 002, the non-Level values are 0, 1, 3, 4; their sites are a new cycle, before 2 in the first arm, after 2 in that arm, and the empty second arm. Appending 4 puts label 3 in the second arm. The resulting spider is $(1; [2], [3])$, hence the directed path $2 \rightarrow 1 \rightarrow 3$. Deleting label 3 recovers the fourth site and therefore the final word value 4.

Corollary 8.2. *At $a = 2$, the assemblies in Theorem 8.1 are in bijection with partial injections on $[m]$, that is, injective maps from a subset of $[m]$ into $[m]$.*

Proof. Read a two-arm spider by reversing its first list, then reading its minimum center, then its second list. This gives a directed path. Conversely, split a directed path at its minimum label and reverse the portion before that minimum to recover the first arm. The empty portions give empty arms, and an isolated vertex gives a singleton spider.

The directed graph of a partial injection has indegree and outdegree at most one at every vertex. Each component is therefore a directed cycle or a directed path, with isolated vertices included as paths. Conversely such a graph specifies a unique partial injection by its directed edges. Loops remain cycles, distinct from isolated vertices. This classical path-cycle decomposition is described in [10]. $\square$

9 The Euler Fiber: Return and Reuse

The Euler fiber has $(b, c) = (1, 0)$. Write

$$S(t) = \sec t + \tan t = \sum_{j \geq 0} E_j \frac{t^j}{j!},$$

where this identity fixes the normalization of the Euler zigzag numbers E_j . The fiber formula becomes

$$H_{a,1,0}(t; \ell) = S(t) \exp \left(\ell \int_0^t S(s)^a ds \right).$$

Two consecutive values of a recover the motivating Return and Reuse models.

At $a = 0$, the budget is $R = \text{ret}$, so

$$H_{0,1,0}(t) = e^t S(t).$$

Thus the master formula recovers Return-Growth as the $a = 0$ specialization of the Euler fiber and places its enumeration under the same kernel that governs the whole family.

At $a = 1$, the budget is $L + R = \text{rep}$, giving Reuse-Growth. Since

$$S' = \frac{1 + S^2}{2}, \quad S'' = SS', \quad S'(0) = 1,$$

we have $(\log S')' = S$ and hence

$$H_{1,1,0}(t) = S(t)S'(t) = S''(t).$$

The coefficient normalization in (1) therefore gives

$$|\mathcal{G}_n(1, 1, 0)| = E_{n+1}.$$

Reuse-Growth is thus the neighboring $a = 1$ specialization of the same Euler fiber.

Varying the nonnegative integer a changes only the reward assigned to Level, while Return continues to earn one unit and New earns none. The common state updates, Event-Slot histories, and integral formula therefore interpolate directly between the Return and Reuse points and continue beyond them to every $a \geq 0$.

10 The Exponential Fiber and Dowling Structures

Fix $(b, c) = (0, 0)$ and an integer $a \geq 1$. Only Level earns reward. Combining Return and New again leaves an insertion rule involving the number of Level events alone. The target class is a classical group-colored partition structure [8], which we describe explicitly.

Write $C_a = \mathbb{Z}/a\mathbb{Z}$ with its residues ordered as $0, \dots, a-1$. A *Dowling partition* of $[m]$ for C_a consists of a distinguished zero block Z , which may be empty, and a set partition of $[m] \setminus Z$ into nonzero blocks. Each nonzero block has a map to C_a , considered modulo adding the same group element to all colors in that block. We use the unique representative in which its minimum label has color zero. Elements of the zero block have no colors. The unique object on $[0]$ has an empty zero block and no nonzero blocks.

Theorem 10.1. *There is a recursive bijection between $\mathcal{G}_n(a, 0, 0)$ and Dowling partitions of $[n-1]$ for C_a , under which L is the number of nonzero blocks.*

Proof. Start from 0 and the empty partition. If $L = k$, there is one Level extension and $ak + 1$ non-Level extensions. Map Level to creation of a singleton nonzero block, with the new largest label colored zero.

The insertions preserving k are insertion into the zero block and insertion into one of the k nonzero blocks with one of a colors. Order them with the zero-block choice first, then the nonzero blocks by increasing minimum, then colors $0, \dots, a-1$ within each block. Match this order to the legal non-Level values in increasing order. Since the inserted label is largest, the minimum color-zero normalization of an existing block is unchanged.

For the inverse, delete the largest label. A singleton nonzero block records Level. Otherwise record whether the label was in the zero block or in a specified nonzero block with its normalized color. In the latter case the block has at least two elements, so its minimum and normalization survive deletion. The smaller partition remains in the class. After recursively recovering its word, the recorded insertion choice has a unique rank and identifies the deleted non-Level value. Inserting it again restores the partition. This proves both directions of the bijection. $\square$

For example, with $a = 2$, the word 0013 corresponds to a partition of $[3]$ with zero block $\{2\}$ and one nonzero block $\{1, 3\}$, colored 0, 1 respectively. The first Level creates the block $\{1\}$; the next value chooses the zero block; the final value is the third non-Level choice, selecting color 1 in the existing block.

Corollary 10.2. *For integers $a \geq 1$,*

$$H_{a,0,0}(t) = \exp\left(t + \frac{e^{at} - 1}{a}\right).$$

Proof. In (2), substitute $V_{0,0} = e^t$ and $\ell = 1$, then integrate e^{as} . $\square$

The case $a = 0$ is the binary model $H_{0,0,0} = e^{2t}$ and requires no cyclic group of order zero. At $a = 1$, colors disappear. Adjoin a new label 0 to the zero block and then increase every label by one, turning the zero block into the first block of a partition of $[n]$. Under this identification, the recursive correspondence above is Callan’s repetition-to-partition bijection [7, Section 2]: a Level append creates a singleton block, and each non-Level append selects an existing block by rank. Thus the Bell enumeration at this point comes with a previously known explicit map.

11 The Boolean Core

If each reward is zero or one, choosing the reward vector is the same as choosing a subset of event types that earn one unit. The eight choices form the Boolean core of the model, shown as the two levels $a = 0, 1$ of Figure 3. Proposition 2.8 orients every cube edge from the lower reward vector to a containing word class. Put $B(t) = e^{e^t - 1}$, the Bell kernel, and use $S(t) = \sec t + \tan t$ from Section 9. Table 3 records the consequences of (2) and the elementary cases in Section 2. Its counts correspond to $n = 1, \dots, 5$, so they are the coefficients of $t^{n-1}/(n-1)!$, not of $t^n/n!$.

(a, b, c)	rewarded events	counts for $n = 1, \dots, 5$	shifted EGF
$(0, 0, 0)$	$\emptyset$	1, 2, 4, 8, 16	e^{2t}
$(1, 0, 0)$	L	1, 2, 5, 15, 52	$e^t B(t)$
$(0, 1, 0)$	R	1, 2, 4, 9, 24	$e^t S(t)$
$(0, 0, 1)$	N	1, 2, 5, 15, 52	$e^t B(t)$
$(1, 1, 0)$	$L + R$	1, 2, 5, 16, 61	$S''(t)$
$(1, 0, 1)$	$L + N$	1, 2, 6, 23, 106	$B(t) \exp(\int_0^t B(s) ds)$
$(0, 1, 1)$	$R + N$	1, 2, 5, 16, 65	$e^t/(1-t)$
$(1, 1, 1)$	$L + R + N$	1, 2, 6, 24, 120	$(1-t)^{-2}$

Table 3: The Boolean core. The counts use word length n and follow from the displayed shifted EGFs.

The two Bell rows have different word rules. The N -row directly labels blocks by first appearance, whereas the L -row uses Callan’s partition interpretation, recovered here as the $a = 1$ Dowling specialization. Composing the two interpretations gives a bijection between $\mathcal{G}_n(1, 0, 0)$ and $\mathcal{G}_n(0, 0, 1)$ through set partitions of $[n]$. The $R + N$ -row rewards precisely the changes $w_i \neq w_{i-1}$ and is Arndt’s change-sequence model recorded in OEIS A000522 [3]. The table therefore organizes classical and newer specializations inside one ordered family and makes their common continuation mechanism explicit.

12 Related Work and Positioning

Mansour and Vajnovszki [15] study restricted-growth words controlled by a prefix statistic, providing the natural background for the present construction. Qiu, Sawada, and Williams consider the still broader (s, f, c) formulation [16, Section 1.2, Eq. (3)], in which the next upper bound may

be an arbitrary nonnegative function of the prefix together with a position-dependent positive offset. Weighted Event-Growth specializes this broad admissibility template to the intrinsic statistic $aL + bR + cN$ and develops the additional structure created by that choice: a two-coordinate branching theorem, a reversible Event-Slot realization, a common master flow, and parameter-specific bijections.

Several vertices of the Boolean core already have established names and interpretations. Callan’s repetition sequences use adjacent equalities [7]; Mansour studies the same objects as level sequences and classifies their length-three pattern avoidance [12]. They are exactly $\mathcal{G}_n(1, 0, 0)$. Arndt’s OEIS comment for A000522 records restricted-growth strings whose next entry is bounded by one plus the number of adjacent changes, hence $\mathcal{G}_n(0, 1, 1)$ [3]. The same comment points toward the classical ascent-sequence setting [6]. Ordinary restricted-growth words and subexcedant sequences occur at $(0, 0, 1)$ and $(1, 1, 1)$. In the present parameter space these familiar points serve as anchors for comparison, while the common state rule explains how neighboring reward vectors enlarge the admissible classes.

Return-Growth and Reuse-Growth motivated the weighted framework. Here their unrestricted forms emerge as the consecutive Euler-fiber points $(0, 1, 0)$ and $(1, 1, 0)$. The two-coordinate description and Event-Slot rules apply uniformly to them and to every nonnegative integer reward triple, so the comparison does not depend on model-specific pattern-avoidance or permutation results.

Insertion encodings and rank matching are established combinatorial methods. Bean, Bell, and Ollson develop horizontal and vertical insertion encodings for ordinary restricted growth functions and study regularity of pattern languages [4]. Their horizontal encoding records promises about future entries in a completed word. Event-Slot histories are prefix-driven instead: the three active slot types have fixed replacement rules determined by (a, b, c) , and the finite history already contains the ranked choices needed to reconstruct the prefix. This prefix-local replacement system is what produces the small-state theorem and the master flow used here.

The differential equations also sit naturally beside the multitype increasing-tree and differential-operator machinery of Blasiak and Flajolet [5, Section 9, Note 25, Proposition 24]. For Event-Growth the single-slot decomposition gives the equations directly, and their particular form reduces to $dU/dV = V^{b-c}$, yielding a uniform integral formula over each fixed (b, c) fiber.

The target objects in the special fibers have classical foundations. Riordan studies label-increasing trees and forests [17], and the uncolored height-two class at $a = 1$ is recorded in A187761 [2]. Theorem 7.1 supplies the explicit Event-Growth encoding and simultaneously identifies Level, New, and Return with root, depth-1, and depth-2 statistics. The path-cycle decomposition of partial injections is classical [10]; Theorem 8.1 reaches it directly from the $a = 2$ cycle-spider model. Likewise, the Level refinements of the rational and exponential fibers match specializations of the generalized Stirling numbers of Hsu and Shiue [11]; compare [14, Theorem 1, Eq. (13)]. The contribution here is the word-to-object correspondence and the way these arrays arise from one event-based growth rule.

Finally, Dowling’s group-colored partition lattices are classical [8], and signed restricted-growth encodings for type- B set partitions have also been studied [1, 13]. Theorem 10.1 gives a recursive correspondence from the nonnegative Event-Growth words at $(a, 0, 0)$ to Dowling partitions, with Level events becoming nonzero blocks. This completes the structural realization of the exponential fiber alongside the forest and assembly descriptions of the Bell and rational fibers.

13 Discussion and Open Problems

Weighted Event-Growth starts from a local trichotomy—Level, Return, or New—and turns it into a uniform theory for history-dependent restricted growth. The key mechanism is the persistence of old labels: once a value has appeared, nonnegative rewards keep it inside every later legal interval. As a result, the apparent dependence on the whole prefix collapses to the two counts

$$p = N, \quad q = g + 1 - N,$$

which are exactly the numbers of Return and New continuations. This small-state description drives every later result in the paper.

The Event-Slot bijection adds the information that state counts alone discard. By matching legal values with canonically ordered active slots, it records each ranked extension and admits an explicit inverse. The same replacement rules then produce the master differential system and the fiber formula. In this way the enumeration is not separate from the bijective picture: both come from the same insertion mechanism.

The special fibers show how much structure this mechanism carries. The Bell fiber turns the three events into vertex depths in colored increasing forests; the rational fiber turns non-Level choices into insertions in cycle-spider assemblies and yields partial injections at $a = 2$; the exponential fiber gives Dowling partitions; and the Euler fiber places Return-Growth and Reuse-Growth at neighboring parameter values. Coordinatewise monotonicity simultaneously organizes the zero-one models into an inclusion cube. Together these results explain why several apparently different restricted-growth rules belong to one family.

This framework leaves a number of natural directions. Which integer triples give elementary, algebraic, or D-finite total counting series? Here D-finite means satisfying a nonzero linear differential equation with polynomial coefficients. The reduction in Theorem 5.3 gives a common starting point for this classification. A second question is which further parameter choices admit direct object interpretations that retain all three event statistics as transparently as the forest correspondence does. In the assembly and Dowling models, Return and New remain recoverable through the inverse map; identifying equally simple static target statistics would sharpen those bijections. Finally, Proposition 2.8 makes the whole parameter space an ordered family. Describing how its inclusions act on the target objects—especially along the concrete edge from Return-Growth $(0, 1, 0)$ to Reuse-Growth $(1, 1, 0)$ —would add a genuinely cross-parameter layer to the theory.

Statements and Declarations

Funding. The author received no funding for this work.

Conflicts of Interest. The author declares no conflicts of interest.

Data Availability Statement. No new data were generated or analyzed in this study. Data sharing is not applicable to this article.

Code availability. Not applicable.

Author contributions. Igor Kleiner is the sole author and is responsible for the conception, formal analysis, investigation, and writing of the manuscript.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process. During the preparation of this work, the author used OpenAI ChatGPT for language editing, manuscript organization, and computational cross-checking. The author reviewed and verified the resulting content and takes full responsibility for the content of the article.

References

- [1] Acharyya, A.: Type B set partitions, an analogue of restricted growth functions. *Ars Combin.* 162, 191–204 (2025). <https://doi.org/10.61091/ars162-14>
- [2] Arndt, J.: Sequence A187761. The On-Line Encyclopedia of Integer Sequences (2013). <https://oeis.org/A187761>. Accessed 5 Sep 2026
- [3] Arndt, J.: Comment on sequence A000522. The On-Line Encyclopedia of Integer Sequences (2013). <https://oeis.org/A000522>. Accessed 6 Sep 2026
- [4] Bean, C., Bell, P.C., Ollson, A.: The Insertion Encoding of Restricted Growth Functions. *Ann. Comb.* (2026). <https://doi.org/10.1007/s00026-026-00832-y>
- [5] Blasiak, P., Flajolet, P.: Combinatorial Models of Creation-Annihilation. *Sém. Lothar. Combin.* 65, B65c, 1–78 (2011)
- [6] Bousquet-Mélou, M., Claesson, A., Dukes, M., Kitaev, S.: $(2 + 2)$ -free posets, ascent sequences and pattern avoiding permutations. *J. Combin. Theory Ser. A* 17(7), 884–909 (2010). <https://doi.org/10.1016/j.jcta.2009.12.007>
- [7] Callan, D.: On Ascent, Repetition and Descent Sequences. arXiv:1911.02209 (2019). <https://doi.org/10.48550/arXiv.1911.02209>
- [8] Dowling, T.A.: A class of geometric lattices based on finite groups. *J. Combin. Theory Ser. B* 14(1), 61–86 (1973); erratum 15, 211 (1973). [https://doi.org/10.1016/S0095-8956\(73\)80007-3](https://doi.org/10.1016/S0095-8956(73)80007-3)
- [9] Flajolet, P., Sedgewick, R.: *Analytic Combinatorics*. Cambridge University Press, Cambridge (2009)
- [10] Ganyushkin, O., Mazorchuk, V.: Combinatorics and distributions of partial injections. *Australas. J. Combin.* 34, 161–186 (2006). arXiv:math/0511431
- [11] Hsu, L.C., Shiue, P.J.-S.: A Unified Approach to Generalized Stirling Numbers. *Adv. Appl. Math.* 20(3), 366–384 (1998). <https://doi.org/10.1006/aama.1998.0586>
- [12] Mansour, T.: Wilf Classes for Level Sequences Avoiding Patterns of Length Three. *Mathematics* 14(11), 1983 (2026). <https://doi.org/10.3390/math14111983>
- [13] Mansour, T., Safadi, A.: Levels, ascents, and descents in restricted-growth words of type B. *Online J. Anal. Comb.* 21, Paper 6, 1–22 (2026). <https://doi.org/10.61091/ojac21-06>
- [14] Mansour, T., Schork, M., Shattuck, M.: The Generalized Stirling and Bell Numbers Revisited. *J. Integer Seq.* 15, Article 12.8.3 (2012). <https://cs.uwaterloo.ca/journals/JIS/VOL15/Schork/schork2.html>
- [15] Mansour, T., Vajnovszki, V.: Efficient generation of restricted growth words. *Inf. Process. Lett.* 113(17), 613–616 (2013). <https://doi.org/10.1016/j.ipl.2013.05.008>
- [16] Qiu, Y., Sawada, J., Williams, A.: Maximize the Rightmost Digit: Gray Codes for Restricted Growth Strings. In: *WALCOM: Algorithms and Computation, Lecture Notes in Computer Science*, vol. 15411, pp. 296–311. Springer (2025). https://doi.org/10.1007/978-981-96-2845-2_19

- [17] Riordan, J.: Forests of label-increasing trees. *J. Graph Theory* 3(2), 127–133 (1979).
<https://doi.org/10.1002/jgt.3190030204>