Eligibility Discriminates Among Theorems and Not Among the Constants They Rest On

Vince Gonzalez
ORCID 0009-0005-3640-014X
Draft — 2026-08-10

Abstract

Separating a theorem's statement dependencies from its proof dependencies bounds how much classical dependence a formal library could shed. Across Mathlib, 13.1% of theorems have a choice-free statement and a choice-dependent proof; nothing outside that band can be removed under any argument. The measure is informative because most theorems fail it — 78.6% of the theorems that reach Classical.choice are bound by what they say.

This note reports that the same measure stops discriminating exactly where it would be used. Applied to the 327,981 constants that dominate those theorems in the dependency graph, 22.88% are eligible overall — but the rate is not uniform in the thing that matters. Among constants dominating a single theorem it is 16.04%. Among constants dominating two or more it is above 99%, rising monotonically with size, and among the 25 constants dominating 500 theorems or more it is 100%.

The transition is a cliff between one and two, not a gradient. A constant that is load-bearing for more than one theorem is eligible with near-certainty, so eligibility cannot be used to choose which load-bearing constant to attack.

The reason is structural. A constant many theorems route through is a general-purpose lemma or instance, general-purpose declarations are stated in general terms, and general terms are choice-free. The property that makes a constant important is the property that makes it eligible. The clearest witness sits at the top of the ranking: Classical.propDecidable : Decidable a has a choice-free type, is therefore eligible, dominates 91,858 theorems, and cannot be made constructive, because deciding an arbitrary proposition is the classical assumption.

---

1. Eligibility, and the population it works on

For a declaration t, write A(t) for the axioms its proof term reaches and S(t) for the union of A(c) over the constants c in t's type, both from Lean.collectAxioms inside the environment. t is statement-bound in an axiom x when $x \in A(t)$ and $x \in S(t)$, and eligible when $x \in A(t)$ and $x \notin S(t)$.

Eligibility is an upper bound on what could be removed, not a claim that removal is possible: a statement whose constants are individually choice-free may still require excluded middle.

Measured over Lean 4.32.1 with a contemporaneous Mathlib — 766,564 declarations, 530,348 of them theorems, 29,469,817 dependency edges:

	theorems	share of all theorems
reach Classical.choice	324,510	61.2%
statement-bound in it	254,957	48.1%
eligible	69,553	13.1%

78.6% of the theorems that reach choice are statement-bound, so the eligible set is a genuine restriction. §5 reconciles these figures with an earlier measurement made by different code.

2. Constants, and the sites among them

A theorem's classical dependence enters somewhere. In the dependency graph reversed and rooted at Classical.choice, a constant c dominates a theorem t when every path from t to the axiom passes through c; severing c frees exactly the theorems it dominates. 398,967 declarations lie on some path to the axiom.

A run of constants each dominating the next, with no theorems lost between them, is one site: severing any member frees the same set, so counting them separately counts one thing many times. Collapsing at a 0.97 weight ratio and naming each site by its most readable member gives 327,981 sites.

3. The result

Applying §1's test to every site:

	sites	share
eligible	75,041	22.88%
statement-bound	252,940	77.12%

Read alone this looks like the measure working. It is not. Broken out by how many theorems each site dominates:

dominates	sites	eligible	statement-bound	eligible %
1	301,062	48,298	252,764	16.04%
2–4	20,151	20,011	140	99.31%
5–9	4,032	4,008	24	99.40%
10–49	2,324	2,313	11	99.53%
50–499	387	386	1	99.74%
500+	25	25	0	100.00%

The aggregate 22.88% is carried almost entirely by the 301,062 sites that dominate exactly one theorem, which are 91.8% of all sites and where the measure still rejects 84% of candidates. The moment a constant is load-bearing for two theorems rather than one, it passes with probability above 0.99, and the rate climbs monotonically across every remaining bucket to 100%.

Among the 2,736 sites dominating ten theorems or more — the population anyone

choosing a target would actually look at — 2,724 are eligible and 12 are not. The largest statement-bound site, Filter.HasAntitoneBasis.iInf_principal, dominates 112 theorems; the next, Perfection.subsemiring._proof_2, dominates 35.

4. Why, and what it means

The failure is not a threshold that could be tuned. A site is by construction a declaration many theorems route through, which is to say a general-purpose lemma or instance. General-purpose declarations are stated in general terms, and general terms are choice-free: Decidable a, a ≤ b → a < b ∨ a = b, Category (C => D). The property that makes a constant load-bearing is the same property that makes it eligible, so the two are positively correlated exactly where a discriminating test needs them not to be. That is the cliff at two: the sites dominating a single theorem are largely a theorem's own auxiliaries, whose types mention what the theorem's type mentions, and everything above that is general.

Classical.propDecidable is the witness worth stating plainly. Its type is (a : Prop) → Decidable a; no constant in it reaches Classical.choice, so it is eligible. It dominates 91,858 theorems, more than any other constant in the library. It is also the definition of classical reasoning in Lean's library and cannot be made constructive. Eligibility ranks it first among removal candidates and is wrong about it in the strongest available sense.

This is the same mismatch that defeats tactic-level attribution, one level up. There, a rate conditioned on a property of the theorem could not recover a property of the proof term. Here, a property of a constant's type cannot recover a property of what the constant asserts. In both cases the measured quantity and the target quantity are properties of different objects.

5. Reconciliation

The dump and analysis used here were written for this note, not reused from the earlier study. Three published figures are reproduced:

	this note	published
theorems	530,348	532,605
reach Classical.choice	324,510 (61.2%)	324,808 (61.0%)
eligibility ceiling	13.1%	13.1%
dependency edges	29,469,817	30,015,601

No figure moves by more than 3%, consistent with Mathlib version drift between the runs.

The dominator computation is validated independently before use. Severing a constant's outgoing edges and recomputing which theorems still reach Classical.choice frees 91,858 theorems for Classical.propDecidable, 23,550 for Classical.byContradiction, 2,018 for lt_or_eq_of_le, 1,221 for eq_or_lt_of_le, 583 for LE.le.lt_or_eq and 476 for Classical.em. Each equals the corresponding dominator subtree size exactly.

6. What this does not say

It does not say eligibility is a bad measure. §1 is evidence that it works on the population it was defined for, and the 13.1% ceiling stands.

It does not say the eligible sites are unfixable. It says the test does not distinguish among them.

It does not identify what does distinguish among them. On a sample of twenty sites that spend Classical.propDecidable directly, re-synthesising the Decidable instance inside the declaration's own local context found an axiom-free instance for 8 of the 21 occurrences that were testable, out of 114 occurrences in total; the remaining 93 sit under inner binders and cannot be synthesised in isolation by that method. All 8 were arithmetic propositions over Nat or Int — x % 2 = 0, b < c, ↑n = 0 and similar. That is consistent with the previously reported pattern of a classical instance being supplied where a constructive one exists on those types, but 8 resolved occurrences is a sample, not a rate, and no claim is made here about how far it generalises.

Substitution of this kind is also expensive: it requires elaboration per site. A ranking is what makes it affordable, and dominator subtree size supplies one. Eligibility does not.

7. Reproduction

The declaration graph is extracted from the Lean environment with statement and proof dependencies in separate columns. ConstantInfo.value? must be passed allowOpaque := true or theorem proofs read as empty, and every statement-versus-proof figure computed from such a dump is wrong rather than imprecise. Dominators are computed by the iterative algorithm of Cooper, Harvey and Kennedy, which converges in three passes on this graph.

Environment: Lean 4.32.1, Mathlib v4.32.1.

References

[1] Gonzalez, V. (2026). *Where Formal Libraries Spend Their Axioms: A Cross-Foundation Measurement, and an Avoidable Classical Dependency in Lean's omega.* Zenodo. https://doi.org/10.5281/zenodo.21769846

[2] Gonzalez, V. (2026). *Why Tactic-Level Rates Cannot Attribute Classical Dependencies in Lean.* Zenodo. https://doi.org/10.5281/zenodo.21853489

[3] K. D. Cooper, T. J. Harvey and K. Kennedy, "A Simple, Fast Dominance Algorithm", Rice University, 2001.

[4] T. Lengauer and R. E. Tarjan, "A fast algorithm for finding dominators in a flowgraph", ACM TOPLAS 1(1), 121–141, 1979.

[5] R. Mendoza-Smith, "Geometric Measurements of the Axiom of Choice in Neural Proof Embeddings", arXiv:2606.28572, 2026.