Impatient PPSZ – A Faster Algorithm for CSP

Shibo Li
Shanghai Jiao Tong University, China

Dominik Scheder
Shanghai Jiao Tong University, China

Abstract

PPSZ is the fastest known algorithm for (d, k)-CSP problems, for most values of d and k. It goes through the variables in random order and sets each variable randomly to one of the d colors, excluding those colors that can be ruled out by looking at few constraints at a time.

We propose and analyze a modification of PPSZ: whenever all but 2 colors can be ruled out for some variable, immediately set that variable randomly to one of the remaining colors. We show that our new "impatient PPSZ" outperforms PPSZ exponentially for all k and all $d \geq 3$ on formulas with a unique satisfying assignment.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Randomized algorithms, Constraint Satisfaction Problems, exponential-time algorithms

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2021.33

Related Version *Full Version*: `https://arxiv.org/abs/2109.02795`

Funding Both authors acknowledge support from the National Natural Science Foundation of China under grant 61502300 and 11671258.

Acknowledgements Dominik Scheder wants to thank Timon Hertli, Isabelle Hurbain, Sebastian Millius, Robin A. Moser, and May Szedlák, his co-authors of [4]. The idea of impatient assignment came up when we were working on [4].

1 Introduction

A Constraint Satisfaction Problem, or CSP for short, consists of a finite set of variables $x_1, \ldots, x_n$, a domain $[d] := \{1, \ldots, d\}$ of potential values, called *colors*, and a set of constraints. A constraint is of the form $(x_{i_1}, \ldots, x_{i_k}) \in S$, where $S \subseteq [d]^k$. In analogy to CNF-SAT, we assume in this paper that $|S| = d^k - 1$, i.e., all but one possible assignments satisfy the constraint. We speak of a (d, k)-*CSP* if all constraints are over k variables. In a slight abuse of notation, we also use (d, k)-CSP to denote the associated *decision problem*: is there a way to assign values in $[d]$ to the variables that satisfies all constraints? This is NP-complete except when $d = 1$ or $k = 1$ or $k = d = 2$, so researchers focus on finding *moderately exponential-time algorithms*: algorithms of running time c^n for $c < d$. Examples include Beigel and Eppstein's randomized algorithm for $(d, 2)$-CSP with running time $O((0.4518d)^n)$ [1];Schöning' s random walk algorithm of running time $O^*((\frac{d(k-1)}{k})^n)$ [11]; Paturi, Pudlák, and Zane encoding-based randomized algorithm called PPZ [7] for k-SAT (i.e., $d = 2$), which runs in time $O(2^{(1-1/k)n})$. Paturi, Pudlák, Saks and Zane [6] improved PPZ by introducing a pre-processing step using small-width resolution. Both PPZ and PPSZ can be easily modified to work for (d, k)-CSP as well, as done by Scheder [8] for PPZ and Hertli et al. [4] for PPSZ. In both cases, several subtleties and technical difficulties arise, which are not present for k-SAT. Furthermore, [4] is the currently fastest algorithm for (d, k)-CSP when $k \geq 4$.

1.1 The PPSZ Algorithm

Let us give an informal description of PPSZ, first for SAT, and then for CSP. In either case, it chooses a random ordering π on the variables $x_1, \dots, x_n$. Then it goes through the variables one by one, in the order of π; when processing x_i, it fixes x_i randomly to `true` or `false`, unless there is a set of up to D clauses that logically implies $x_i = b$ for some $b \in \{0, 1\}$; in the latter case, we fix x_i to b and say that x_i has been inferred by D-implication. For CSP, the only difference is that when processing x_i, it checks (with brute force) for which colors $c \in [d]$ the statement $[x_i \neq c]$ can be inferred by a set of up to D constraints; if so, we say $[x_i \neq c]$ is *D-implied*, and color c is obviously ruled out. It then fixes x_i randomly to one of the colors not yet ruled out (or declares failure if all colors have been ruled out).

Unique-SAT versus general-SAT. A peculiar feature of PPSZ, as analyzed in the seminal paper [6], is that it performs better if the input instance F has a *unique* satisfying assignment. Certain properties, such as the existence of critical clause trees, break down once F has multiple solutions. In [6], the authors proposed a clever but technical workaround, which incurred an exponential overhead for $k = 3, 4$. In his 2011 breakthrough paper, Hertli [3] showed that this peculiarity is in fact an artifact of the analysis, and gave a very abstract and high-level proof that PPSZ on formulas with many solutions is indeed no worse. His proof was later simplified by Scheder and Steinberger [10]. The proofs in [3] and [10] work only provided that the internal machinery of the PPSZ algorithm (e.g., checking D-implication) is "not too good". Curiously, in [4] it turned out that, for $k = 2, 3$ and certain values of d, the PPSZ machinery is indeed "too good", and consequently their time complexity for the general case (multiple solutions) is worse than for the unique case (exactly one solution). For formulas with a unique solution, their analysis gives a running time of $O\left(d^{nS_{d,k}+o(n)}\right)$, for $S_{d,k}$ defined below. This is the best known running time for all d, k except for $(d, k) = (3, 2)$ and $(d, k) = (4, 2)$.

Improvements to PPSZ for k-SAT. Two recent results improve PPSZ. Hansen, Kaplan, Zamir, and Zwick [2] define a *biased* version of PPSZ and show that it achieves an improvement for all $k \geq 3$. Scheder [9] shows that PPSZ itself performs exponentially better than in the analysis of [6]. We would not be surprised if both improvements carry over to (d, k)-CSP, although to our knowledge, this has not been analyzed so far. The improvement presented in this work is of a different quality: it is not a generalization of some idea for k-SAT; in fact, the main idea only makes sense for $d \geq 3$ and thus is particular to (d, k)-CSP problems.

The time complexity of PPSZ for Unique (d, k)-CSP. A main result of [4] is that PPSZ solves Unique (d, k)-CSP in time $O\left(2^{S_{d,k}\, n+o(n)}\right)$, where $S_{d,k}$ is defined by the following random experiment: let T^∞ be the infinite rooted tree in which each node on even depth (which includes the root at depth 0) has $k-1$ children and every node on odd depth has $d-1$ children. Let $T_1, \dots, T_{d-1}$ be disjoint copies of T^∞, sample $p \in [0, 1]$ uniformly, and delete every odd-level node with probability p, independently. Let J_c be the indicator variable that is 1 if the root of T_c is contained in an infinite component after this deletion step. Then

$$S_{d,k} := \mathbb{E}[\log_2(J_1 + \cdots + J_{d-1} + 1)] \,. \tag{1}$$

Approximate values of $S_{d,k}$ for small values of d, k can be found in Hertli et al. [4][1] To understand $S_{d,k}$ for large values d, it is most advantageous to write $c_{d,k} := \log_2(d) - S_{d,k}$. Intuitively, $\log_2(d)$ is the number of bits required to specified a color and $S_{d,k}$ is the number of coin tosses used by PPSZ to determine it, so $c_{d,k}$ is the number of coin tosses per variable saved by PPSZ. Theorem 1.4 of [4] states that $\lim_{d\to\infty} c_{d,k} = -\int_0^1 \log_2(1 - r^{k-1})$.

Thus, for large d, PPSZ saves *a constant number of bits* per variable; this is somewhat unsatisfactory; we would like to save a *constant fraction of bits*, i.e., an algorithm of running time $d^{(1-c_k)n}$ for some $c_k > 0$ for arbitrary d. The existence of such an algorithm is posed as an open problem by Kouckỳ, Rödl, and Talebanfard [5]. In the same paper, they give such an algorithm for CSP formulas where every variable appears in a constant number of constraints. Formally, their algorithm runs in time $d^{(1-c_{k,r})n}$ on (d, k)-CSP formulas F where each of the n variables occurs in at most r constraints.

1.2 Our Contribution

In this work, we focus on the case that F has a unique satisfying assignment α^*, without loss of generality $\alpha^* := (d, d, \ldots, d)$. The idea behind our improvement is as follows: suppose x, y, z are variables appearing in the order y, x, z in π. Focus on the point in time when PPSZ processes x, and assume every assignment prior to x has been correct. For example, the variable y has already been replaced by the constant d. In other words, when PPSZ tries to infer statements like $[x \neq c]$ from small sets of constraints, it can use the information $[y = d]$. It cannot use $[z = d]$, however. Or can it? Maybe PPSZ can already infer $[z \neq 1], \ldots, [z \neq d-1]$; in this case, it can also infer $[z = d]$, and it would be safe to fix z to d. Let us propose the following rule:

> **Rule of One.** Whenever $[z = c]$ can be inferred by D-implication, fix z to c.

This rule is "uncontroversial" in the sense that it will never make a mistake. Indeed, the reader who is familiar with the literature about PPSZ, in particular with its original version using small-width resolution, will notice that resolution implicitly implements the above rule. We propose the following more aggressive rule:

> **Rule of Two.** Whenever $[z = c_1 \vee z = c_2]$ can be inferred by D-implication, i.e., if all but 2 colors can be ruled out, pick $c \in \{c_1, c_2\}$ uniformly at random and fix z to c.

Obviously, this rule can introduce mistakes. On the plus side, it might be very unlikely that the range of plausible (i.e., not ruled out) colors for z further decreases from 2 to 1. Better to bite the bullet now, decide on a value for z, hope that it is correct, and use that information for subsequent D-implications. For example, it might be that using the information $[z = d]$ lets us rule out additional colors for x, the variable currently being processed by PPSZ. Unfortunately, this rule does more bad than good: consider the variables coming towards the very end of π. For each of them, it is very likely that all but one color can be ruled out; thus, PPSZ would set them correctly with high probability; using our Rule of Two, this probability would go down from (almost) 1 to (roughly) 1/2 since we decide on a value once only two values are left. We propose a less impatient rule:

[1] In [4] they define $S_{d,k}$ with $\log_d$ instead of $\log_2$. We prefer the binary logarithm for purely cosmetic reasons: otherwise some of our expressions would contain the expression "$\log_d(2)$", but "1" is undoubtedly nicer.

Conservative Rule of Two. Apply the Rule of Two only to variables z that are among the first θn in π; don't apply it to the last $(1-\theta)n$ variables.

We will show that for those early variables, it is extremely unlikely that the set of plausible colors gets narrowed down to only one color; and that it is somewhat more likely that the Rule of Two helps us rule out one additional colors for a variable. In particular, we prove the following theorem:

▶ **Theorem 1.** *For every $d \geq 3$ and $k \geq 2$, there is some $\epsilon > 0$ and a randomized algorithm solving (d,k)-CSP in time $2^{n(S_{d,k}-\epsilon)}\text{poly}(n)$, where $S_{d,k}$ is as defined in (1).*

Note that the Rule of Two does not make any sense in the Boolean context. For every variable z we can trivially 0-imply $[z = 0 \vee z = 1]$, and thus every variable immediately qualifies for impatient assignment. We thus lose all control over the ordering π in which we process the variables. The Rule of One does of course make sense in the Boolean setting and is implicitly implement if you are running "strong" PPSZ, i.e., with small-width resolution.

1.3 Notation

Let $V = \{x_1, \ldots, x_n\}$ be a set of variables and $[d] = \{1, \ldots, d\}$ be the set of possible colors. A *literal* is an expression $(x \neq c)$, where $x \in V, c \in [d]$. A *clause* is a disjunction of literals: $(v_1 \neq c_1 \vee v_2 \neq c_2 \vee ... \vee v_k \neq c_k)$. A (d,k)-CSP is a conjunction of clauses of size k each. An assignment α is a function $V \to [d]$. It satisfies a literal $(x \neq c)$ if $\alpha(v) \neq c$; it satisfies a clause if it satisfies at least one literal therein; it satisfies a (d,k)-CSP F if it satisfies all clauses in F. If $V' \subseteq V$ and $\alpha : V' \to [d]$, we call α a *partial assignment*; $\text{vbl}(\alpha)$ denotes its domain, i.e., V'. $F^{[\alpha]}$ is the simplified formula after setting all variables in V' according to α. We will write partial assignments like this: $[x \mapsto 2, y \mapsto 3, ...]$ and therefore $F^{[x \mapsto 2]}$ will denote the formula after replacing x with 2. For a set $V' \subseteq V$ of variables, we take the liberty to write $F^{[V' \mapsto d]}$, where $[V' \mapsto d]$ is the partial assignment $[x \mapsto d$ for each $x \in V']$. For a clause C and a (d,k)-CSP F, $\text{vbl}(C)$ and $\text{vbl}(F)$ denote the sets of variables in C and F, respectively. For a rooted tree T and a node v therein, the *subtree of T rooted at v* is the tree containing v (as root) and all its descendants. We use the notation [`statement`], which evaluate to 1 if `statement` holds, and to 0 otherwise.

1.4 PPSZ and impatient PPSZ

▶ **Definition 2** (D-implication [4])**.** *Let F be a (d,k)-CSP formula and u be a literal of F. We say F* implies *u and write $F \vDash u$ if all assignment satisfying F also satisfy u. We say F* D-implies *u and write $F \vDash_D u$ if there is some $G \subseteq F$ with $|G| \leq D$ and $G \vDash u$.*

For the rest of the paper, $D = D(n)$ will be some slowly growing function in n, so $F \models_D u$ can be checked in time $O(|F|^D \text{poly}(n))$, which is subexponential in n.

▶ **Definition 3** (Plausible values)**.** *Let F be a (d,k)-CSP formula and x a variable. We say color $c \in [d]$ is D-plausible for x in F if F does not D-imply $(x \neq c)$. Let* $\text{Plaus}(x, F, D)$ *denote the set of all colors that are D-plausible for x. We will drop the parameter D if it is understood from the context.*

Note that our code specifies π as an explicit input parameter; it is the responsibility of the "user" to make sure $\text{PPSZ}(F, \pi)$ is called with a random π; furthermore, we implicitly assume that PPSZ declares failure if the set $\text{Plaus}(x, F^{[\alpha]})$ in Line 4 is empty.

Algorithm 1 PPSZ algorithm.

procedure PPSZ(F, π)
  $\alpha \leftarrow$ the empty assignment
  for $x \in \mathrm{vbl}(F)$ in the order of π **do**
    choose $\iota \in \mathrm{Plaus}(x, F^{[\alpha]})$ uniformly at random
    $\alpha := \alpha \cup [x \mapsto \iota]$
  end for
  return α if it satisfies F, else failure
end procedure

From now on, we view π not as a permutation of the variables but as a *placement*, i.e., a function $V \to [0,1]$; note that if $\pi : V \to [0,1]$ is sampled uniformly at random, it will be an injection with probability 1; sorting V in ascending order by their π-value will give a permutation of V. Additionally, we fix two parameters θ (to be determined later) and $\zeta := 2 - \log_2(3)$, and mark every variable x as *eligible for impatient assignment* as follows:

▶ **Definition 4** (Eligible for impatient assignment). *For each variable x, define $\mathbb{I}_x \in \{0,1\}$ as follows. (1) If $\pi(x) \geq \theta$, set $\mathbb{I}_x := 0$; (2) if $\pi(x) < \theta$, set $\mathbb{I}_x := 1$ with probability ζ and to 0 with probability $1-\zeta$, independently of all other choices. If $\mathbb{I}_x = 1$ we say x is* eligible for impatient assignment.

The reasoning behind condition (1) is that a "late" variable x (one with $\pi(x) \geq \theta$) is likely to end up with a unique plausible value when PPSZ reaches it; setting it prematurely will do more bad than good; our decision to mark "early" variables with $\pi(x) < \theta$ as eligible only with probability ζ (and not with probability 1) has technical reasons: we want a certain function to become concave, and our choice of ζ is the largest value for which this happens.

Algorithm 2 Impatient PPSZ.

procedure ImpatientPPSZ(F, π)
  $\alpha :=$ the empty assignment
  for $x \in \mathrm{vbl}(F)$ in ascending order of π **do**
    while $\exists y \in \mathrm{vbl}(F) \setminus \mathrm{vbl}(\alpha)$ with $\mathbb{I}_y = 1$ and $|\mathrm{Plaus}(y, F^{[\alpha])}| \leq 2$ **do**
      choose $\iota \in \mathrm{Plaus}(y, F^{[\alpha]})$ uniformly at random
      $\alpha := \alpha \cup [y \mapsto \iota]$
    end while
    if $x \notin \mathrm{vbl}(\alpha)$ **then**
      choose $\iota \in \mathrm{Plaus}(x, F^{[\alpha]})$ uniformly at random
      $\alpha := \alpha \cup [x \mapsto \iota]$
    end if
  end for
  return α if it satisfies F, else failure
end procedure

Proof sketch of Theorem 1. As the analysis of PPSZ, our proof relies heavily on the concept of *critical clause trees*. Intuitively, those trees are a neat and tidy way to describe all ways how $[x = c]$ can be ruled out. For PPSZ, this can be described as an extinction event in a Galton-Watson process; for ImpatientPPSZ however, we need a more complicated notion. To make matters worse, while [6] and [4] showed that the worst case for their PPSZ analysis

happens if all trees are "nice", this fails for our algorithm. We have to come up with a (short) list of non-niceties, and for each of them show that $[x = c]$ is a bit more likely to be ruled out; we can then finally analyze the "nice" case. ◀

2 Conceptual Framework

Notation for sets of variables coming before variable x: V_x and V_x^{imp}. To analyze PPSZ and our variant ImpatientPPSZ, we need to talk about the point in time where the algorithm processes a variable x, and in particular, we need to talk about the set of variables that have already been assigned a value at this point. For PPSZ, this is easy: we define $V_x := \{y \in \text{vbl}(F) \mid \pi(y) < \pi(x)\}$. For ImpatientPPSZ, it's a bit more complicated: imagine we run ImpatientPPSZ but feed it the "correct" values in every assignment; that is, whenever a color c is chosen, make sure that $c = d$ (we manipulate this random source to always choose the correct color); pause the algorithm in the iteration when variable x is being processed, just after line 7, and look at the partial assignment α built so far. We set $V_x^{\text{imp}} := \text{vbl}(\alpha) \setminus \{x\}$. We remove x for purely technical reasons; if x happens to be already set at that time, then line 9 and 10 will be skipped by the algorithm anyway.

▶ **Observation 5.** *If line 9 is executed then c is chosen uniformly at random from the set* $\text{Plaus}(x, F^{[V_x^{\text{imp}} \mapsto d]})$.

We define the following indicator variables:

$$A_{x,c} := \begin{cases} 1 & \text{if } c \in \text{Plaus}(x, F^{[V_x \mapsto d]}, D) \\ 0 & \text{else.} \end{cases}$$

$$A_{x,c}^{\text{imp}} := \begin{cases} 1 & \text{if } c \in \text{Plaus}(x, F^{[V_x^{\text{imp}} \mapsto d]}, D) \\ 0 & \text{else.} \end{cases}$$

and $A_x := \sum_c A_{x,c}$ and $A_x^{\text{imp}} := \sum_c A_{x,c}^{\text{imp}}$. These are random variables in our random placement π. Note that $A_{x,d} = A_{x,d}^{\text{imp}} = 1$ because color d is always plausible; also, $A_{x,c}^{\text{imp}} \le A_{x,c}$ simply because $V_x \subseteq V_x^{\text{imp}}$, i.e., ImpatientPPSZ has at least as much information as PPSZ.

▶ **Lemma 6** ([4]). *For a fixed permutation π,* $\Pr[\text{PPSZ}(F, \pi) \text{ finds } \alpha^*] = \prod_x \frac{1}{A_x(\pi)}$. *For a random permutation,* $\Pr_\pi[\text{PPSZ}(F, \pi) \text{ finds } \alpha^*] \ge 2^{-\sum_x \mathbb{E}_\pi[\log_2 A_x(\pi)]}$.

The second statement follows from the first by Jensen's inequality. To obtain a similar formula for ImpatientPPSZ, we need to take into account that a variable x might be assigned in line 6 or in line 10.

▶ **Lemma 7.** *For a fixed permutation π, we have* $\Pr[\text{ImpatientPPSZ}(F, \pi) \text{ finds } \alpha^*] \ge \prod_x \frac{1}{\max\left(1+\mathbb{I}_x, A_x^{\text{imp}}(\pi)\right)}$. *For a random permutation, the probability that ImpatientPPSZ succeeds is at least*

$$\Pr_\pi[\text{ImpatientPPSZ}(F, \pi) \text{ finds } \alpha^*] \ge 2^{-E_\pi\left[\sum_x \log_2\left(\max\left(1+\mathbb{I}_x, A_x^{\text{imp}}(\pi)\right)\right)\right]}.$$

Proof. If $\mathbb{I}_x = 0$ then x will be assigned in line 10 and thus its value will be correct with probability $1/A_x^{\text{imp}}(\pi)$, conditioned on all prior assignments being correct. If $\mathbb{I}_x = 1$ then either it is assigned in line 6, and is correct with probability $1/2$; or it is still assigned regularly in line 10, and is correct with probability $1/A_x^{\text{imp}}(\pi)$. This proves the first inequality. The second inequality in the lemma follows from the first by Jensen's inequality. ◀

2.1 Independence between colors

The crucial quantity in the analysis of ImpatientPPSZ is the random variable $A^{\text{imp}}_x = \sum_c A^{\text{imp}}_{x,c}$. The next lemma states that we can focus on analyzing the indicator variables $A^{\text{imp}}_{x,c}$ individually; that is, if we condition on $\pi(x) = p$, then the d indicator variables are independent in the worst case. More formally:

▶ **Lemma 8** (Independence between colors). *Let $\pi : V \to [0,1]$ be uniformly random and set $p := \pi(x)$. We sample d random variables $\tilde{A}^{\text{imp}}_{x,c} \in \{0,1\}$, $c = 1, \dots, d$ by setting each $\tilde{A}^{\text{imp}}_{x,c}$ to 1 with probability $\Pr[A^{\text{imp}}_{x,c} = 1 \mid \pi(x) = p]$, independently. Set $\tilde{A}^{\text{imp}}_x := \sum_c \tilde{A}^{\text{imp}}_{x,c}$. Then*

$$\mathop{\mathbb{E}}_{\pi}\left[\log_2\left(\max\left(1 + \mathbb{I}_x, A^{\text{imp}}_x(\pi)\right)\right)\right] \le \mathop{\mathbb{E}}_{\pi}\left[\log_2\left(\max\left(1 + \mathbb{I}_x, \tilde{A}^{\text{imp}}_x(\pi)\right)\right)\right] \tag{2}$$

Proof idea. We would like to prove this along the lines of Lemma 3.5 of [4]. The additional problem here is that although the function $f : t \mapsto \log(t)$ is concave, the function $g : t \mapsto \log(\max(2, t))$ isn't. This is why, if $\pi(x) < \theta$, we set $\mathbb{I}_x$ to 1 with probability ζ and to 0 with probability $1 - \zeta$. The convex combination $(1 - \zeta) \cdot f + \zeta \cdot g$ is concave[2] and the proof goes through just as for Lemma 3.5 in [4]. See Lemma 24 in the appendix for a complete proof. ◀

The upshot is that it is sufficient to bound $\Pr[A^{\text{imp}}_{x,c} = 1 \mid \pi(x) = p]$ from above, for each variable x and color c, individually.

2.2 Critical Clause Trees and Brief Analysis of PPSZ

In this section, we define critical clause trees and review some results from [4]. We assume that $\mathbf{d} = (d, \dots, d)$ is our unique satisfying assignment. Note that every unsatisfied literal is of the form $(y = d)$ and every satisfied literal is of the form $(y \neq c)$ for some $c \neq d$. Let $x \in \text{vbl}(F)$ and $c \in \{1, \dots, d-1\}$. The *critical clause tree* $T^h_{x,c}$ *of height* h has two types of nodes: a node u on an even level (which includes the root at level 0) is a *clause node*, has a *clause label* clauselabel(u) and an *assignment label* β_u; it has at most $k - 1$ children (in fact, one for each unsatisfied literal in clauselabel(u)). A node v on an odd level is a *variable nodes* and has a *variable label* varlabel(v); it has exactly $d - 1$ children. An edge (v, w) from a variable node v to a clause node w has an edge color $EC(e) \in [d-1]$. Thus, if varlabel$(v) = y$ and (v, w) edge color $EC(v, w) = i$, then this edge represents the alternative assignment $[y \mapsto i]$. The critical clause tree $T^h_{x,c}$ is constructed as in algorithm 3.

Let us assume h is always odd, so the lowest layer of $T^h_{x,c}$ consists of variable nodes. $T^h_{x,c}$ has two types of leaves: those variable nodes at height h; we call them *safe leaves*; and clause nodes whose clause label does not contain any literal of the form $(y \neq d)$; we call them *unsafe leaves.*

▶ **Proposition 9** ([4]).

1. *Suppose v is a clause node in $T^h_{x,c}$ with clause label C and $(y \neq i), i \in [d]$ is a literal in C. Then if $i = d$, v has a child whose variable label is y. If $i < d$ and $y \neq x$ then v has an ancestor node whose variable label is y.*
2. *No variable appears more than once as variable label on a path from root to a leaf.*

[2] The attentive reader might notice: it's not concave; however, if we change the definition of "log" in the definition of f and g from the usual log to "log on $\mathbb{N}$ and linear between integers, then it is concave, provided that $\zeta \le 2 - \log_2 3$.

Algorithm 3 BuildCCT(F, x, c, h).

Create a root node and set $\beta_{\text{root}} := \alpha[x = c]$
while $\exists$ clause node u of height less than $h-1$ without a clause label **do**
    Find a clause C which is not satisfied by β_u
    Set clauselabel$(u) := C$
    for each unsatisfied literal $(y \neq d)$ in C **do**
        Create a new child v of u
        varlabel$(v) := y$
        for $i \in [d-1]$ **do**
            Create a new child w of v
            Set $\beta_w := \beta_v[y = i]$
            Set $EC(v, w) = i$
        end for
    end for
end while
remove clause nodes at height $h+1$
return $T^h_{x,c}$

▶ **Definition 10** (labeled tree). *A labeled tree is a possibly infinite tree such that: (1) every node is either a variable node or a clause node; (2) a variable node u has a label* varlabel$(u) \in \mathbb{L}$ *in some label space $\mathbb{L} \supseteq V$; (3) they alternate, i.e., if a variable node has children, they are all clause nodes, and vice versa; (4) its degree is bounded: there is some $\Delta \in \mathbb{N}$ such that every node has at most Δ children. A leaf in a labeled tree is a safe leaf if it is a variable node; Otherwise, it is an unsafe leaf.*

Note that each subtree of a critical clause tree is a labeled tree. A *safe path* in a labeled tree is a path that starts at the root and is either infinite or ends at a safe leaf.

▶ **Definition 11** (Cut_p and Cut). *Let T be a labeled tree. The event* $\text{Cut}_p(T)$ *is an event in the probability space of all placements $\pi : \mathbb{L} \to [0,1]$ that happens if every safe path in T contains a node v with $\pi(\text{varlabel}(v)) < p$. Let x be the label of the root of T. We define* $\text{Cut}(T) := \text{Cut}_{\pi(x)}(T)$.

Suppose T is a labeled tree, and let $T_1, \ldots, T_l$ be the subtrees rooted at the l children of the root of T. Note that the T_i are themselves labeled trees. If the root of T is a clause node then $\text{Cut}_p(T) = \bigwedge_{i=1}^{l} \text{Cut}_p(T_i)$. If it is a variable node, let $y := \text{varlabel}(\text{root}(T))$, and observe that $\text{Cut}_p(T) = [\pi(y) < p]$ if root(T) itself is a safe leaf (i.e., if $l = 0$) and $\text{Cut}_p(T) = [\pi(y) < p] \vee \bigwedge_{i=1}^{l} \text{Cut}_p(T_i)$ else .

Next, we connect the notion of cuts to our notion of being a plausible color. For this, set $L := (d-1)(k-1)$ and observe that $T^h_{x,c}$ has at most L^i clause nodes at depth $2i$. Choose $\tilde{h}$ to be the largest integer for which $1 + L + L^2 + \cdots + L^{\tilde{h}} \leq D$ (recall D, our strength parameter in the definition of D-implication), and set $h := 2\tilde{h} + 1$. Then $T^h_{x,c}$ has at most D clause nodes and h is also a slowly growing function in n.

▶ **Lemma 12** ([4]). *Let $p = \pi(x)$. If* $\text{Cut}_p(T^h_{x,c})$ *happens then $A_{x,c} = 0$.*

Recall the infinite trees T^∞ and $T_1, \ldots, T_{d-1}$ and the indicator variables $J_1, \ldots, J_{d-1}$ defined above, just before (1), and observe that $J_c = 1$ iff $\text{Cut}_p(T_c)$ does *not* happen. Let T_∞ be the subtree of T^∞ rooted at the first child of the root. Define $Q(p) := \Pr[\text{Cut}_p(T^\infty)]$ and $R(p) := \Pr[\text{Cut}_p(T_\infty)]$. The next proposition is from [4], adapted for our purposes.

▶ **Proposition 13** ([4]). *Set $L = (k-1)(d-1)$. If $p \geq 1 - \frac{1}{L}$ then $Q(p) = R(p) = 1$; otherwise, $Q(p)$ and $R(p)$ are the unique roots in $[0,1]$ of the equations $Q = \left(p + (1-p)Q^{d-1}\right)^{k-1}$ and $R = p + (1-p)R^L$, respectively. Furthermore, $Q(p) = R(p)^{k-1}$.*

As our height parameter h grows (roughly logarithmic with our strength parameter D), the critical clause trees $T^h_{x,c}$ will look more and more like T^∞, and thus the cut probability will converge to $Q(p)$. Formally, let $\mathrm{error}(d,k,h,p)$ and $\mathrm{error}(d,k,h)$ stand for any functions that converge to 0 as $h \to \infty$.

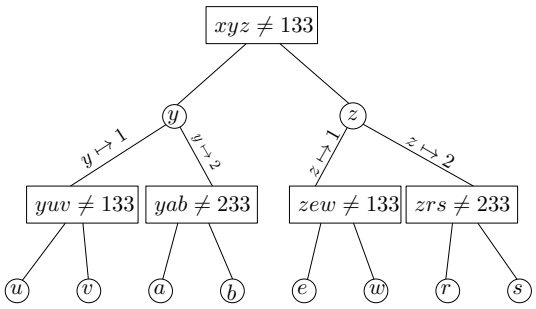
▶ **Proposition 14** (Lemma 3.6 in [4]). $\Pr[\mathrm{Cut}_p(T^h_{x,c})] \geq \Pr[\mathrm{Cut}_p(T_c)] - \mathrm{error}(d,k,p,h)$.

To summarize: conditioned on $\pi(x) = p$, the sum $A_x = A_{x,1} + \cdots + A_{x,d}$ has the worst behavior if all $A_{x,c}$ are independent (Lemma 8); furthermore, $A_{x,c} \leq J_c$ except with probability $\mathrm{error}(d,k,p,h)$, for all $c \leq d-1$, and therefore:

▶ **Lemma 15** ([4]). $\mathbb{E}_\pi[\log_2(A_x)] \leq \mathbb{E}[\log_2(J_1 + \cdots + J_{d-1} + 1)] + \mathrm{error}(d,k,h) = S_{d,k} + \mathrm{error}(d,k,h)$.

3 The Probability of Inferring $x \neq c$

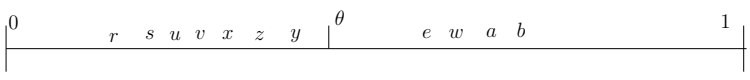
Just as [4] analyzes PPSZ by studying the random variables $A_{x,c}$, we have to study $A^{\mathrm{imp}}_{x,c}$. We can always resort to the "old" analysis via $A^{\mathrm{imp}}_{x,c} \leq A_{x,c}$. However, the whole point of this work is to show that this inequality is often strict. To understand how and when this might happen, we discuss an example for $d = 3$.



This is $T^3_{x,1}$, the critical clause tree for x and 1 built up to height 3. The formula F in question contains the constraints shown as clause labels, but of course contains many more constraints. Suppose that u, v, a, b, z come before x in π, and e, w, r, s, y come later. In the normal PPSZ, we have already set $u, v, a, c, z \mapsto 3$ when considering x, and thus the clauses of F will have shrunk:

- $(yuv \neq 133)$ shrinks to $(y \neq 1)$;
- $(yab \neq 233)$ shrinks to $(y \neq 2)$;
- $(zew \neq 133)$ and $(zrs \neq 233)$ don't shrink but disappear: they are satisfied by $z \mapsto 3$;
- $(xyz \neq 133)$ shrinks to $(xy \neq 13)$.

Together, the three shrunk clauses $(y \neq 1)$, $(y \neq 2)$, and $(xy \neq 13)$ imply $(x \neq 1)$; since $D \geq 3$ this means that $x = 1$ can be ruled out, i.e., $A_{x,1} = 0$. Next, suppose π, viewed as a placement $\pi : V \to [0,1]$, looks like this:



and assume for simplicity that all variables l with $\pi(l) < \theta$ in $\mathrm{vbl}(F)$ are eligible for impatient assignment (i.e., have $\mathbb{I}_l = 1$). Note that $\mathrm{Cut}(T^3_{x,1})$ does not happen. Namely, the path from root to c contains two variable labels, y and b, and $\pi(y), \pi(b) \geq \pi(x)$. Analogously, the alternative assignment $\alpha^*[x \mapsto 1, y \mapsto 2, b \mapsto 2]$ satisfies all clauses in the figure above, and thus the algorithm cannot infer $x \neq 1$ from those clauses alone, and $A_{x,1} = 1$. Observe now what happens in ImpatientPPSZ:

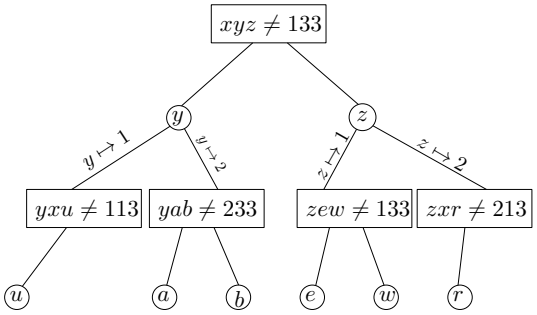
- $r, s, u, v \mapsto 3$ before x is even considered;
- $(yuv \neq 133)$ shrinks to $(y \neq 1)$, and thus $\mathrm{Plaus}(y, F^{[\alpha]})$ shrinks to $\{2, 3\}$;
- y is assigned a value in line 6;
- the analogous thing happens to z;
- $r, s, u, v \in V_x$, and $r, s, u, v, y, z \in V^{\mathrm{imp}}_x$;
- $(xyz \neq 133)$ shrinks to $(x \neq 1)$ and thus $A^{\mathrm{imp}}_{x,1} = 0$.

We can now try to work out a formula for the probability that $x = c$ is ruled out in this manner; however, our above example and analysis contains two silent assumptions that cannot be taken for granted in general:

1. All variable labels in $T^3_{x,c}$ are distinct.

2. All clause labels of $T^3_{x,c}$ are critical clauses, i.e., $k-1$ of its literals are of the form $y \neq d$.

The original PPSZ paper [6] addresses Point 1 by using the FKG inequality to show that having multiple labels can never hurt us. But now we are talking about a more complicated event; it is not clear whether an FKG-like result applies. Point 2 is more troublesome. Consider the alternative scenario that $T^3_{x,c}$ looks like this:



and consider the same π as above: $r, s, u, v, x, z, y, \theta, e, w, a, b$. After setting $r, s, u, v \mapsto 3$, the shrunk clauses are $(yx \neq 11)$, $(yab \neq 233)$, $(zew \neq 133)$, and $(zx \neq 21)$. Neither for y nor for z can we rule out any color, and therefore our impatient mechanism will not kick in. We will have $V_x = V^{\mathrm{imp}}_x = \{r, s, u, v\}$. In other words, non-critical clauses seem useless for ImpatientPPSZ. But looking at the above example tree, we see what comes to the rescue: the right-most clause node is missing a child; it has at most $k-2$ children instead of $k-1$. This alone will be enough to improve our success probability by a bit. It is time for some formal definitions.

▶ **Definition 16** (Privileged variables). *A variable x is* privileged *if there is some color $c \in \{1, \ldots, d-1\}$ such that*

1. *$T^h_{x,c}$ has fewer than $(k-1)^2(d-1)$ variable nodes at level 3* or

2. *$T^3_{x,c}$ has two variable nodes u and w with* $\mathrm{varlabel}(u) = \mathrm{varlabel}(w)$.

▶ **Proposition 17.** *There is an $\epsilon_{\mathrm{privileged}} > 0$ for variable x which is* privileged*, depending only on d and k, such that*

$$\mathbb{E}\left[\log_2(A_x)\right] \leq S_{d,k} - \epsilon_{\mathrm{privileged}} + \mathrm{error}(d, k, h) \ ,$$

for every privileged variable x in F.

See Proposition 25 in the appendix for a proof.

▶ **Corollary 18.** *For every privileged variable* $x \in \mathrm{vbl}(F)$ *it holds that* $\mathbb{E}\left[\log_2(\max(1+\mathbb{I}_x, A_x^{\mathrm{imp}}))\right] \leq S_{d,k} - \epsilon_{\mathrm{privileged}} + c\theta + \mathrm{error}(d,k,h)$

Proof. Since $\max(a,b) \leq a \cdot b$ when $a, b \geq 1$, we get

$$\mathbb{E}\left[\log_2(\max(1+\mathbb{I}_x, A_x^{\mathrm{imp}}))\right] \leq \mathop{\mathbb{E}}_{\pi}[\log_2(1+\mathbb{I}_x)] + \mathop{\mathbb{E}}_{\pi}[\log_2(A_x^{\mathrm{imp}})] \ .$$

The first term equals $\Pr[\mathbb{I}_x = 1] = c\theta$; the second is at most $\mathbb{E}\left[\log_2(A_x)\right]$, which by Proposition 17 is at most $S_{d,k} - \epsilon_{\mathrm{privileged}} + \mathrm{error}(d,k,h)$. This concludes the proof. ◀

▶ **Lemma 19.** *There is a constant* $\epsilon > 0$*, depending only on* d *and* k*, such that*

$$\mathbb{E}\left[\log_2(\max(1+\mathbb{I}_x, A_x^{\mathrm{imp}}))\right] \leq S_{d,k} - 0.1699\left(\frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)\right) + \mathrm{error}(d,k,h) \ .$$

for all non-privileged variables x*. The constant factor hidden in the* $O(\cdot)$ *depends only on* d *and* k*.*

By choosing θ sufficiently small, we can make sure that the bounds in Lemma 19 and Corollary 18 are both at most $S_{d,k} - \epsilon_{d,k} + \mathrm{error}(d,k,h)$, for some $\epsilon_{d,k}$ depending only on d and k. Together with Lemma 7, this proves Theorem 1.

Proof of Lemma 19. For a color $1 \leq c \leq d-1$, fix the critical clause tree $T_{x,c}^h$ and let us introduce a bit of notation. The root of $T_{x,c}^h$ has a label

$$C_{\mathrm{root}} = (x \neq c \vee y_1 \neq d \cdots \vee y_{k-1} \neq d) \ .$$

It has $k-1$ children $v_1, \ldots, v_{k-1}$, whose respective variable labels are $y_1, \ldots, y_{k-1}$. Let T_i denote the subtree of $T_{x,c}^h$ rooted at v_i. Each y_i in turn has $d-1$ children; each such level-2 node v has a clause label C_v; note that C_v is a *critical clause*, i.e., $k-1$ of its literals are of the form $(z \neq d)$, since otherwise it would have fewer than $k-1$ children, and $T_{x,c}^h$ would have fewer than $(k-1)^2(d-1)$ nodes at level 3; in other words, x would be privileged.

We need to define an event $\mathrm{ImpCut}_p(T_{x,c}^h)$ which, analogous to $\mathrm{Cut}_p(T_{x,c}^h)$, describes the event $A_{x,c}^{\mathrm{imp}} = 0$ in terms of $T_{x,c}^h$ only. Going for a full such characterization is possible but messy, and it is not clear what the worst-case structure of such $T_{x,c}^h$ will be; this is the reason why we, when considering our impatient assignment mechanism, will look only up to depth 3 in $T_{x,c}^h$. For each node w of $T_{x,c}^h$ at level 1, 2, or 3, we define event $\mathrm{LocalImpCut}_p(v)$ as follows:

1. If v is at level 3 of $T_{x,c}^h$ then $\mathrm{LocalImpCut}_p(v)$ happens if $\pi(\mathrm{varlabel}(v)) < p$.
2. If v is at level 2 of $T_{x,c}^h$ then $\mathrm{LocalImpCut}_p(v)$ happens if $\mathrm{LocalImpCut}_p(w)$ happens for the $k-1$ children w of v (recall that $\mathrm{clauselabel}(v)$ is a critical clause and therefore v has exactly $k-1$ children);
3. If v is at level 1, set $y := \mathrm{varlabel}(v)$; $\mathrm{LocalImpCut}_p(v)$ happens if
   a. $\pi(y) < p$ *or*
   b. $\mathbb{I}_y = 1$ *and* $\mathrm{LocalImpCut}_p(v)$ happens for at least $d-2$ of the $d-1$ children of v.

Finally, we define

$$\mathrm{ImpCut}_p(T_{x,c}^h) := \bigwedge_{i=1}^{k-1} \left(\mathrm{Cut}_p(T_i) \vee \mathrm{LocalImpCut}_p(v_i)\right) \tag{3}$$

The next lemma is the "impatient analog" of Lemma 12.

▶ **Lemma 20.** *Let $p = \pi(x)$. If $\mathrm{ImpCut}_p(T^h_{x,c})$ happens then $A^{\mathrm{imp}}_{x,c} = 0$.*

The proof is very similar to that of Lemma 12, just taking into account the impatient assignment mechanism. We restate and prove it as Lemma 26 in the appendix. Next, we prove a lower bound on $\Pr[\mathrm{ImpCut}_p(T^h_{x,c})]$. For $q \in [0,1]$ and $l \in \mathbb{N}$, define

$$\mathrm{abamo}(q,l) := q^l + l(1-q)q^{l-1} \ . \tag{4}$$

The name abamo is the acronym of "all but at most one" and is indeed the probability that, among l independent events of probability q each, all or all but one happen. Recall the definition of $Q(p) := \Pr[\mathrm{Cut}_p(T^\infty)]$ just before Proposition 13.

▶ **Lemma 21.** *If $p < \theta$ then $\Pr[\mathrm{ImpCut}_p(T^h_{x,c}) \mid \pi(x) = p]$ is at least*

$$\left(p + c(\theta-p)\mathrm{abamo}(p^{k-1}, d-1) + (1-p-c(\theta-p))Q(p)^{d-1}\right)^{k-1} - \mathrm{error}(d,k,h) \ .$$

If $p \geq \theta$ then it is at least $Q(p) - \mathrm{error}(d,k,h)$.

Proof sketch. For each subtree T_i of $T^h_{x,c}$, either $\mathrm{Cut}_p(T_i)$ or $\mathrm{LocalImpCut}_p(v_i)$ must happen. Now this happens if either (1) $\pi(y) < p$, which explains the first term of the sum in the parentheses; (2) $\pi(y) \geq p$ and $\mathbb{I}_{y_i} = 1$ and $\mathrm{LocalImpCut}_p(v_i)$, which is the second term; or (3) $\pi(y) \geq p$ and $\mathbb{I}_{y_i} = 0$ and $\mathrm{Cut}_p(T_i)$, which is the third term. See Lemma 27 for a complete proof. ◀

Let us summarize our reasoning so far. Define an ensemble $J^{\mathrm{imp}}_1, \ldots, J^{\mathrm{imp}}_{d-1}$ of random variables in $\{0,1\}$ as follows: set $p := \pi(x)$; then independently set each J^{imp}_c to 0 with probability W^{k-1} and 1 with probability $1 - W^{k-1}$, where

$$W = W(p) := \begin{cases} p + c(\theta-p)\mathrm{abamo}(p^{k-1}, d-1) + (1-p-c(\theta-p))Q(p)^{d-1} & \text{if } p < \theta \\ R(p) & \text{else.} \end{cases}$$

One checks that $W(p)$ is continuous at $p = \theta$ since $R(p) = p + (1-p)R(p)^{(k-1)(d-1)} = p + (1-p)Q(p)^{d-1}$. Set $J^{\mathrm{imp}} := J^{\mathrm{imp}}_1 + \cdots + J^{\mathrm{imp}}_{d-1} + 1$. We have shown so far that

$$\begin{aligned} \mathbb{E}\left[\log_2 \max(1 + \mathbb{I}_x, A^{\mathrm{imp}}_{x,c})\right] &\leq \mathbb{E}\left[\log_2 \max(1 + \mathbb{I}_x, J^{\mathrm{imp}})\right] + \mathrm{error}(d,k,h) \\ &= \Pr[J^{\mathrm{imp}} = 1 \wedge \mathbb{I}_x] + \mathbb{E}\left[\log_2(J^{\mathrm{imp}})\right] + \mathrm{error}(d,h,k) \ . \end{aligned} \tag{5}$$

▶ **Proposition 22.** $\Pr[J^{\mathrm{imp}} = 1 \wedge \mathbb{I}_x] \leq \frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)$.

▶ **Proposition 23.** $\mathbb{E}\left[\log_2(J^{\mathrm{imp}})\right] - S_{d,k} \leq (d-1)\log_2(1-1/d) \cdot \left(\frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)\right)$.

We prove the two propositions in Section E in the appendix. Together with (5), they imply that $\mathbb{E}\left[\log_2 \max(1 + \mathbb{I}_x, A^{\mathrm{imp}}_{x,c})\right] - S_{d,k}$ is at most

$$\left(\frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)\right)\left(1 + (d-1)\log_2(1-1/d)\right) + \mathrm{error}(d,k,h) \ .$$

The expression in the first parenthesis is positive for sufficiently small θ; in fact, we have to choose θ small enough to beat the hidden constant in the $O(\cdot)$, which in turn depends only on d and k. The expression in the second parenthesis, $1 + (d-1)\log_2(1-1/d)$, is negative for all $d \geq 3$. It is maximized for $d = 3$, where it becomes $2 - 2\log_2(3) < -0.1699$. Thus, we can choose θ such that the whole expression is at most $S_{d,k} - \epsilon_{d,k} + \mathrm{error}(d,k,h)$ for some $\epsilon_{d,k} > 0$ depending only on d and k. This concludes the proof of Lemma 19. ◀

4 Future Work

In the analysis of PPSZ, the worst case happens if all everything looks "nice": all variable nodes in $T_{x,1}, \dots, T_{x,d-1}$ have different labels; all clause labels are critical clauses.

In this scenario, our analysis for impatient assignment could go deeper than level 3; we could define a more powerful event ImpCut and obtain much better bounds on the running time. Indeed, future work hopefully will identify the worst-case shape of the $T^h_{x,c}$ and allow us to analyze the full power impatient assignment.

The condition $|\mathrm{Plaus}(y, F^{[\alpha]})| \leq 2$ in Line 4 in Algorithm 2 is arbitrary. Why "≤ 2"? Why not "≤ 3"? For large d, what would the optimal cut-off value be?

References

1 Richard Beigel and David Eppstein. 3-coloring in time $O(1.3289^n)$. *J. Algorithms*, 54(2):168–204, 2005. doi:10.1016/j.jalgor.2004.06.008.

2 Thomas Dueholm Hansen, Haim Kaplan, Or Zamir, and Uri Zwick. Faster k-SAT algorithms using biased-PPSZ. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 578–589. ACM, 2019. doi:10.1145/3313276.3316359.

3 Timon Hertli. 3-SAT faster and simpler – unique-SAT bounds for PPSZ hold in general. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science – FOCS 2011*, pages 277–284. IEEE Computer Soc., Los Alamitos, CA, 2011. doi:10.1109/FOCS.2011.22.

4 Timon Hertli, Isabelle Hurbain, Sebastian Millius, Robin A Moser, Dominik Scheder, and May Szedlák. The PPSZ algorithm for constraint satisfaction problems on more than two colors. In *International Conference on Principles and Practice of Constraint Programming*, pages 421–437. Springer, 2016.

5 Michal Koucký, Vojtech Rödl, and Navid Talebanfard. A separator theorem for hypergraphs and a CSP-SAT algorithm. *CoRR*, abs/2105.06744, 2021. arXiv:2105.06744.

6 Ramamohan Paturi, Pavel Pudlák, Michael E Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *Journal of the ACM (JACM)*, 52(3):337–364, 2005.

7 Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 566–574. IEEE, 1997.

8 Dominik Scheder. PPZ for more than two truth values-an algorithm for constraint satisfaction problems. *arXiv preprint*, 2010. arXiv:1010.5717.

9 Dominik Scheder. PPSZ is better than you think. *Electron. Colloquium Comput. Complex.*, 28:69, 2021. URL: https://eccc.weizmann.ac.il/report/2021/069.

10 Dominik Scheder and John P. Steinberger. PPSZ for General k-SAT - making Hertli's analysis simpler and 3-SAT faster. In Ryan O'Donnell, editor, *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, volume 79 of *LIPIcs*, pages 9:1–9:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.CCC.2017.9.

11 Uwe Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 410–414. IEEE Computer Society, Los Alamitos, CA, 1999. doi:10.1109/SFFCS.1999.814612.

A Independence between colors

▶ **Lemma 24** (Lemma 8, restated). *Let $\pi : V \to [0,1]$ be uniformly random and set $p := \pi(x)$. We sample d random variables $\tilde{A}^{\mathrm{imp}}_{x,c} \in \{0,1\}$, $c = 1, \dots, d$ by setting each $\tilde{A}^{\mathrm{imp}}_{x,c}$ to 1 with probability $\Pr[A^{\mathrm{imp}}_{x,c} = 1 \mid \pi(x) = p]$, independently. Set $\tilde{A}^{\mathrm{imp}}_x := \sum_c \tilde{A}^{\mathrm{imp}}_{x,c}$. Then*

$$\mathop{\mathbb{E}}_{\pi}\left[\log_2\left(\max\left(1+\mathbb{I}_x, A^{\mathrm{imp}}_x(\pi)\right)\right)\right] \leq \mathop{\mathbb{E}}_{\pi}\left[\log_2\left(\max\left(1+\mathbb{I}_x, \tilde{A}^{\mathrm{imp}}_x(\pi)\right)\right)\right] \tag{6}$$

Proof. We prove (6) conditioned on $\pi(x) = p$. Let $\mathbf{Z} \in \{0,1\}^{V\setminus\{x\}}$ be defined by $Z_y := [\pi(y) \geq p]$. Note that each Z_y is 1 with probability $1-p$, independently. Next, observe that each $A^{\text{imp}}_{x,c}$ is a monotone increasing Boolean function $f_c(Z)$: moving some $\pi(y)$ above p can only increase $A^{\text{imp}}_{x,c}$. Let $\mathbf{Z}^{(1)}, \dots, \mathbf{Z}^{(d)}$ be d independent copies of $\mathbf{Z}$; that is, each has the same distribution as $\mathbf{Z}$ but they are independent. Conditioned on $\pi(x) = p$, we have $(f_1(\mathbf{Z}), \dots, f_d(\mathbf{Z})) \sim (A^{\text{imp}}_{x,1}, \dots, A^{\text{imp}}_{x,d})$ and $(f_1(\mathbf{Z}^{(1)}), \dots, f_d(\mathbf{Z}^{(d)})) \sim (\tilde{A}^{\text{imp}}_{x,1}, \dots, \tilde{A}^{\text{imp}}_{x,d})$, where $A \sim B$ means that the random variables A and B have the same distribution.

Now if $p > \theta$ and therefore $\mathbb{I}_x = 0$, then the function $\log_2(\max(1+\mathbb{I}_x, \cdot))$ in (6) becomes $\log_2(\cdot)$ and we can directly apply the Concave Correlation Lemma (Lemma A.1 of the full version of [4]).

If $\mathbb{I}_x = 1$, the trouble is that the function $t \mapsto \log_2(\max(2,t))$ is not concave anymore. However, note that if $\pi(x) < \theta$, we set $\mathbb{I}_x$ to 1 with probability ζ and 0 with probability $1-\zeta$. Conditioned on $\pi(x) = p$, the randomness in (6) comes from two sources: (1) the choice of $\mathbb{I}_x$; (2) the randomness in $\mathbf{Z}$ (or $\mathbf{Z}^{(1)}, \dots, \mathbf{Z}^{(d)}$ for the right-hand side). We can break down both sides of (6) as follows:

$$\mathop{\mathbb{E}}_{\mathbf{Z},\mathbb{I}_x}\left[\log_2(\max(1+\mathbb{I}_x, A^{\text{imp}}_x))\right] = \mathop{\mathbb{E}}_{\mathbf{Z}}\left[\zeta \log_2(\max(2, A^{\text{imp}}_x)) + (1-\zeta)\log_2(A^{\text{imp}}_x)\right] \ , \tag{7}$$

where $\zeta = 2 - \log_2(3)$. Now the function $t \mapsto \zeta\log_2(\max(2,t)) + (1-\zeta)\log_2(t)$ is still not concave. However, note that the arguments of $\log_2(t)$ in (6) and (7) are integers; define $g(t)$ to be the function that equals $\log_2(t)$ if t is an integer, and is linear between integers. Now g is concave and $t \mapsto \zeta g(\max(2,t)) + (1-\zeta)g(t)$ is concave, too. In fact, this function is linear on $[1,3]$ and agrees with g for $t \geq 3$. Now the lemma again follows by the Concave Correlation Lemma (Lemma A.1 of [4]). ◀

B PPSZ for privileged variables

▶ **Proposition 25** (Proposition 17, restated). *Suppose $x \in \text{vbl}(F)$ is a priviledged variable. Then there is an $\epsilon_{\text{privileged}} > 0$, depending only on d and k, such that*

$$\mathbb{E}\left[\log_2(A_x)\right] \leq S_{d,k} - \epsilon_{\text{privileged}} + \text{error}(d,k,h) \ ,$$

for every privileged variable x in F.

Proof. This proof is similar in spirit and also technical details to the proof of Lemma 19 in [9], except that the latter is concerned with SAT (i.e., the case $d = 2$).

Note that a variable x can be privileged for two reasons: first, there is some color c such that the critical clause tree $T^h_{x,c}$ has fewer than $(k-1)L$ leaves at level 3; in other words, some clause node v at level 2 has fewer than $k-1$ children (note that the nodes at level 0 and 1 have the "right" numer of children; the clause label of 0 is a critical clause, and therefore the root has always $k-1$ children; an odd-level node always has $d-1$ children). The second reason would be that, for some color c, level 1 and 3 of the critical clause tree $T^h_{x,c}$ contain nodes u and v with varlabel(u) = varlabel(v).

It is easy to see that the first kind of privilege is stronger: let v be the level-2 node with fewer than $k-1$ children. We can add "fictitious" subtrees until v has $k-1$ children, and make sure that one of the added children shares its variable label with an already-existing level-3 node. The result of this operation, $T'_{x,c}$, exhibits a privilege of the second kind, and $\text{Cut}_p(T^h_{x,c}) \supseteq \text{Cut}_p(T'_{x,c})$.

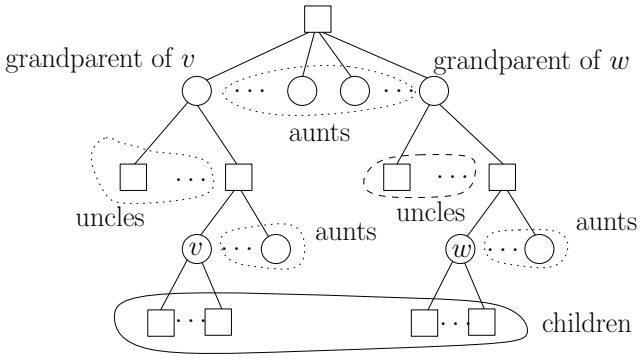
Thus, let us assume that x is privileged because $T^h_{x,c}$ contains two nodes v and w with varlabel(v) = varlabel$(w) = z$ and the depths of v and w are in $\{1,3\}$. Analogous to the proof of Proposition 14 (Lemma 3.5 in [4], we start with iteratively assign fresh labels to variable

nodes; as shown in [4], this never increases $\Pr[\mathrm{Cut}_p(T^h_{x,c})]$. We apply this to all variable nodes except v and w, and obtain a new tree T. We make sure that there are no "missing children" in T, i.e., that every clause has $k-1$ children; this can be achieved by attaching fictitious subtrees, which does not increase $\Pr[\mathrm{Cut}_p(T)]$. Also, we will for convenience assume that T is infinite, i.e., has no safe leaves (and no unsafe leaves, either). This does increase $\Pr[\mathrm{Cut}_p]$, but by at most $\mathrm{error}(d,k,h)$. In T we still have $\mathrm{varlabel}(v) = \mathrm{varlabel}(w) = z$, but all other labels are distinct. Let T' be the tree where v and w receive fresh labels z_v, z_w. We already know that $\Pr[\mathrm{Cut}_p(T')] = Q(p)$. It remains to show that $\Pr[\mathrm{Cut}_p(T)]$ is substantially larger than $\Pr[\mathrm{Cut}_p(T')]$. For this, let $\mathbb{L}$ be the set of variable labels appearing in T and T', and let $\tau : \mathbb{L} \setminus \{z, z_v, z_w\} \to [0,1]$. We will analyze the difference

$$\Pr[\mathrm{Cut}_p(T) \mid \tau] - \Pr[\mathrm{Cut}_p(T') \mid \tau] \tag{8}$$

for fixed τ. Introduce the three Boolean variables $a := [\pi(z) < p]$, $a_v := [\pi(z_v) < p]$, and $a_w := [\pi(z_w) < p]$. Note that under τ, the event $\mathrm{Cut}_p(T')$ reduces to $f_\tau(a_v, a_w)$ for some monotone Boolean function and $\mathrm{Cut}_p(T')$ reduces to $f_\tau(a,a)$, for the same function f_τ. There are only six possible such functions: $f_\tau(a_v,a_w)$ is either 0, 1, a_v, a_w, $a_v \wedge a_w$, or $a_v \vee a_w$. If it is one of the first four, then $\Pr[f_\tau(a_v,a_w)] = \Pr[f_\tau(a,a)]$ and (8) is 0. It cannot be $a_v \vee a_w$: the nodes v and w are not ancestors of each other. Finally, if $f_\tau(a_v,a_w) = a_v \wedge a_w$ then we call τ *pivotal* and observe that (8) becomes $p - p^2$.

From here on, our plan is to lower bound the probability that τ is pivotal. We give a necessary and sufficient criterion for τ to be pivotal.[3] It is best illustrated with a figure.



Squares are the clause nodes and circles are the variable nodes. Note that we assume that v and w are both on level 3, and their lowest common ancestor is the root. In the other cases, the picture and the subsequent calculation will be slightly different. To ease notation, we adopt the notation $\mathrm{Cut}_p(u) := \mathrm{Cut}_p(T_u)$, where T_u is the subtree of T' rooted at u (note that T' and T have the same node set, only some labels differ). In the case depicted in the figure, τ is pivotal if and only if

1. $\mathrm{Cut}_p(u)$ happens for all aunts and uncles u;
2. $\mathrm{Cut}_p(u)$ does not happen for all children u of v; neither for all children u of w.
3. $\pi(\text{grandparent of } v), \pi(\text{grandparent of } w) \geq p$.

[3] Actually, it is sufficient for our purposes that the criterion be sufficient, and not necessary that it be necessary.

Furthermore, note that $\Pr[\mathrm{Cut}_p(u)]$ equals $Q(p)$ if u is an uncle and $R(p)$ if u is an aunt. Therefore,

$$\begin{aligned}\Pr[\mathrm{Cut}_p(T)] - \Pr[\mathrm{Cut}_p(T')] \geq (p-p^2)\cdot \Pr[\tau \text{ is pivotal}] = \\ (p-p^2)Q(p)^{\text{uncles}}\cdot R(p)^{\text{aunts}}\cdot \left(1-Q(p)^{d-1}\right)^2 (1-p)^2 \\ =: \delta(p)\ .\end{aligned}$$

It is clear that $\delta(p) > 0$ for $0 < p < 1 - 1/N$ and $\delta(p) = 0$ for $p \geq 1 - 1/N$. Recalling the definition of $S_{d,k} = \mathbb{E}[\log(J_1 + \cdots + J_{d-1} + 1)]$ comparing it to $\mathbb{E}[\log_2(A_x)] = \mathbb{E}[\log_2(A_{x,1} + \cdots + A_{x,d-1} + 1)]$, we can couple the ensembles $\mathbf{A} := (A_{x,c})_{c=1}^{d-1}$ and $\mathbf{J} := (J_c)_{c=1}^{d-1}$ such that $\mathbf{A} \leq \mathbf{J}$ except with probability $\mathrm{error}(d,k,p,h)$, and $A_{x,c} = 0, J_c = 1$, conditioned on $\pi(x) = p$, happens with probability at least $\delta(p) - \mathrm{error}(d,k,p,h)$. In fact, let us ignore the term $\mathrm{error}(d,k,h)$ for now and simply assume that $\mathbf{A} \leq \mathbf{J}$ (more rigorously, we would have to replace every $T^h_{x,c}$ by the appropriate infinite version; we decide to simply ignore $\mathrm{error}(d,k,h)$ in the following, lest we overload the reader with our notation). Set $\Delta := J - A_x$, and observe that $\Delta \geq 0$ and $\Pr[\Delta \geq 1 \mid \pi(x) = p] \geq \delta(p)$.

$$\begin{aligned}\mathbb{E}[\log_2(J)] - \mathbb{E}[\log_2(A_x)] &= -\mathbb{E}\left[\log_2\left(\frac{J-\Delta}{J}\right)\right] \\ &= -\mathbb{E}\left[\log_2\left(1-\frac{\Delta}{J}\right)\right] \\ &\geq -\mathbb{E}\left[\log_2\left(1-\frac{\Delta}{d}\right)\right] \\ &\geq \frac{\log_2(e)}{d}\,\mathbb{E}\left[\Delta\right] \\ &\geq \frac{\log_2(e)}{d}\int_0^1 \delta(p)\,dp =: \epsilon_{\text{privileged}}\ .\end{aligned}$$

This is some positive number, and it depends only on d and k. ◀

C Local reasoning for ImpatientPPSZ

▶ **Lemma 26** (Lemma 20, restated). *Suppose $x \in \mathrm{vbl}(F)$ is non-priviledged. Let $p = \pi(x)$. If $\mathrm{ImpCut}_p(T^h_{x,c})$ happens then $A^{\mathrm{imp}}_{x,c} = 0$.*

Proof. We will prove the contrapositive: assume that $A^{\mathrm{imp}}_{x,c} = 1$ and show that $\mathrm{ImpCut}_p(T^h_{x,c})$ does not happen. Let $F(T^h_{x,c})$ denote the set of clause labels appearing in $T^h_{x,c}$. Since $A^{\mathrm{imp}}_{x,c} = 1$ by assumption, the formula $F^{[V^{\mathrm{imp}}_x \mapsto d]}$ does not D-imply $(x \neq c)$. In particular, $|F(T^h_{x,c})| \leq D$ and therefore $F(T^h_{x,c})^{[V^{\mathrm{imp}} \mapsto d]}$ does not imply $(x \neq c)$. This means that there is an assignment γ that (1) satisfies $F(T^h_{x,c})$, (2) $\gamma(x) = c$, (3) $\gamma(y) = d$ for all $y \in V^{\mathrm{imp}}_x$.

As a first step, we will show that $\mathrm{Cut}_p(T^h_{x,c})$ does not happen. For this, we will construct a sequence of clause nodes $u_0, u_1, \ldots$, with u_0 being the root andn u_{i+1} being a grandchild of u_i, keeping the following invariant:

Invariant. For every clause node u in the sequence, $\beta_u(y) \neq d \Rightarrow \gamma(y) = \beta_u(y)$.

Note that the invariant is satisfied for the root: x is the only variable with $\beta_{\mathrm{root}}(x) \neq d$, and $\gamma(x) = c = \beta_{\mathrm{root}}(x)$. To find u_{i+1} from u_i, let C_i be the clause label of u_i, and write C_i as

$$C_i = (y_1 \neq c_1 \vee \cdots \vee y_l \neq c_l \vee z_{l+1} \neq d \vee \cdots \vee z_{k-1} \neq d)\ ,$$

where $c_1, \dots, c_l \neq d$. By construction, β_{u_i} violates C_i, and therefore $\beta_{u_i}(y_j) = c_j$ for $1 \leq j \leq l$; by the invariant, $\gamma(y_j) = c_j$, too. But γ satisfies C_i (it satisfies every clause label in $T^h_{x,c}$), and therefore $\gamma(z_j) = c \neq d$ for some $l+1 \leq j \leq k-1$. In particular, u_i has children. Let v be the child of u_i with variable label z_j. If v is a leaf (a safe leaf), terminate the process and call the path from root to v the *witness path*. Otherwise, and let u_{i+1} be the child of v with $EC(v, u_{i+1}) = c$. Note that u_{i+1} satisfies the invariant.

Since $T^h_{x,c}$ is finite, this process terminates with a witness path. Note that $\gamma(y) \neq d$ for all variable labels y appearing on that path. In particular, this means that $y \notin V^{\text{imp}}_x$, thus $y \notin V_x$, thus $\pi(y) \geq \pi(x)$. In other words, $\text{Cut}_p(T^h_{x,c})$ does not happen.

Without loss of generality, let v_1 be the level-1-node on the witness path, and T_1 be the tree rooted at v_1, and $y_1 := \text{varlabel}(v_1)$. Observe that $\text{Cut}_p(T_1)$ does not happen. We will now show that $\text{LocalImpCut}_p(v_1)$ does not happen, either. Assume, for the sake of contradiction, that $\text{LocalImpCut}_p(v_1)$ happens. Does it happen because of Point 3a in the definition? Certainly not: $\gamma(y_1) \neq d$ since v_1 is on the witness path, and thus $\pi(y_1) \geq p$. So it happens because of Point 3b, and $\mathbb{I}_{y_1} = 1$; without loss of generality, this means that $\text{LocalImpCut}_p(v_1)$ happens for the first $d-2$ children $w_1, \dots, w_{d-2}$ of v_1; let $C_1, \dots, C_{d-2}$ be the respective clause labels. All those C_i are critical clauses (x is non-priviledged, remember), and have $k-1$ children each. So LocalImpCut_p happens for the first $(k-1)(d-2)$ of the $(k-1)(d-1)$ grandchildren of v_1. In other words, all their variable labels z have $\pi(z) < p$ and thus $z \in V_x$. Under the assignment $[V_x \mapsto d]$, each of C_i reduces to a unit clause; this unit clause is still violated by β_{w_i} and is therefore either $(y_1 \neq i)$ or $(x \neq c)$. If it was $(x \neq c)$ then $F(T^h_{x,c})^{[V_x \mapsto d]}$ would imply $(x \neq c)$ and therefore $A_{x,c} = A^{\text{imp}}_{x,c} = 0$, contradicting our assumption. So it is $(y_1 \neq i)$. In other words, $F(T^h_{x,c})^{[V_x \mapsto d]}$ contains the unit clauses $(y_1 \neq 1), \dots, (y_1 \neq d-2)$; thus, when x is being processed by ImpatientPPSZ, the set of plausible values for y has been reduced to at most two values: $d-1$ and d; since $\mathbb{I}_{y_1} = 1$, the algorithm will assign y_1 a value in Line 6, and $y_1 \in V^{\text{imp}}_x$. This is again a contradiction: $\gamma(y_1) \neq d$ since v_1 is on the witness path; $\gamma(y_1) = d$ since $y_1 \in V^{\text{imp}}_x$. This concludes the proof. ◀

D ImpCut probability

Suppose $x \in \text{vbl}(F)$ is non-priviledged and $T^h_{x,c}$ is a critical clause tree for x and $c \in [d]$.

▶ **Lemma 27** (Lemma 21, restated). *If $p < \theta$ then* $\Pr[\text{ImpCut}_p(T^h_{x,c}) \mid \pi(x) = p]$ *is at least*

$$\left(p + c(\theta - p)\text{abamo}(p^{k-1}, d-1) + (1 - p - c(\theta - p))Q(p)^{d-1}\right)^{k-1} - \text{error}(d,k,h)\ .$$

If $p \geq \theta$ then it is at least $Q(p) - \text{error}(d,k,h)$.

Proof. If $p \geq \theta$ then this is obvious since already $\text{Cut}_p(T^h_{x,c})$ has probability at least $Q(p) - \text{error}(d,k,h)$, by Proposition 14. Thus we assume $p < \theta$. The root of $T^h_{x,c}$ has $k-1$ children $v_1, \dots, v_{k-1}$, whose respective variable labels are $y_1, \dots, y_{k-1}$. Let T_i denote the subtree of $T^h_{x,c}$ rooted at v_i.

$$\begin{aligned}\Pr\left[\text{ImpCut}_p(T^h_{x,c})\right] &= \Pr\left[\bigwedge_{i=1}^{k-1}\left(\text{Cut}_p(T_i) \vee \text{LocalImpCut}_p(v_i)\right)\right] \\ &\geq \prod_{i=1}^{k-1}\left(\Pr[\text{Cut}_p(T_i) \vee \text{LocalImpCut}_p(v_i)]\right)\ . \qquad \text{(FKG inequality)}\end{aligned}$$

We can apply the FKG inequality because each event $\mathrm{Cut}_p(T_i) \vee \mathrm{LocalImpCut}_p(v_i)$ is a monotone increasing Boolean function in the variables $[\pi(z) < p]$ and $\mathbb{I}_{y_i}$. It remains to show that, for each $1 \le i \le k-1$, the event $\mathrm{Cut}_p(T_i) \vee \mathrm{LocalImpCut}_p(v_i)$ happens with probability at least

$$p + c(\theta - p)\mathrm{abamo}(p^{k-1}, d-1) + (1 - p - c(\theta - p))Q(p)^{d-1} - \mathrm{error}(d,k,h) \tag{9}$$

For this, let us abbreviate $T := T_i$, $v := v_i$ its root, and $y := \mathrm{varlabel}(v) = y_i$; also, we define the events $A := \mathrm{LocalImpCut}_p(v)$ and $B := \mathrm{Cut}_p(T)$. We distinguish three cases:

(i) if (1) $\pi(y) < p$ then the desired event $A \vee B$ happens;
(ii) if $\pi(y) \ge p$ and $\mathbb{I}_y = 1$ (which implies $\pi(y) < \theta$) then we ignore B and focus on A;
(iii) if $\pi(y) \ge p$ and $\mathbb{I}_y = 0$, then A does not happen, so focus on B.

Formally,

$$\Pr[A \vee B] \ge \Pr[(i)] + \Pr[(ii)] \cdot \Pr[A \mid (ii)] + \Pr[(iii)] \cdot \Pr[B \mid (iii)]$$

Next, let us look at each case.

1. $\Pr[(i)] = p$; this explains the first term in (9).
2. $\Pr[(ii)] = c(\theta - p)$. Furthermore, if if (ii) happens, then A happens if and only if for at least $d-2$ of the children $w_1, \dots, w_{d-1}$, the event $A_j := \mathrm{LocalImpCut}_p(w_j)$ happens. Each A_j happens with probability $\rho := p^{k-1}$; they are independent since all $(d-1)(k-1)$ grandchildren of v have distinct labels. Therefore,

$$\begin{aligned}\Pr[A \mid (ii)] &= \Pr[A_1 \wedge \dots \wedge A_{d-1}] + \sum_{j^*=1}^{d-1} \Pr[\neg A_{j^*} \wedge \bigwedge_{j \ne j^*} A_j] \\ &= \rho^{d-1} + (d-1)(1-\rho)\rho^{d-2} = \mathrm{abamo}(p^{k-1}, d-1)\ .\end{aligned}$$

   This explains the second term in (9).
3. $\Pr[(iii)] = 1 - p - c(\theta - p)$. If (iii) happens, then B happens if and only if $\mathrm{Cut}_p(T')$ happens for each of the $d-1$ subtrees of T. By Proposition 14, this happens with probability $(Q(p) - \mathrm{error}(d,k,h))^{d-1}$. This explains the third and fourth term in (9).

This concludes the proof. ◀

E Bounding losses and gains. Proofs of Propositions 22 and 23

First, we need some good-enough estimates for our probabilities $R(p)$, $Q(p)$, and $W(p)$. Note that $R(p)$ and $Q(p)$ are the roots of certain polynomials, and we do not have an explicit formula for them. The bounds in Proposition 28 are somewhat crude but sufficient for our purposes.

▶ **Proposition 28.** *$R(p) \le p + 4\,p^L$; $Q(p) \le \left(p + 4\,p^L\right)^{k-1}$; and $W(p) \le p + O(\theta p^{(d-2)(k-1)})$. The hidden constant in the O depends on d and k only.*

Proof. One checks that $R(p)$ is convex on the interval $[0, 1 - 1/L]$. To see this, note that for $p \le 1 - 1/L$, $R(p)$ is the unique solution in $[0,1]$ of the equation

$$R = p + (1-p)R^L\ ,$$

by Proposition 13. We can solve explicitly for p and check that $p(R)$ is concave, by elementary calculus. Since R is convex, $R(0) = 0$, and $R(1 - 1/L) = 1$, the graph of $R(p)$ is below the line from $(0, 0)$ to $(1 - 1/L, 1)$, and therefore $R(p) \leq \frac{L}{L-1}p$. This is not enough yet, but applying the equation of R to this estimate gives

$$R = p + (1-p)R^L \leq p + (1-p)\left(\frac{L}{L-1}\,p\right)^L \leq p + 4\,p^L\ .$$

The upper bound for Q follows directly from $Q(p) = R(p)^{k-1}$. It remains to prove the upper bound on $W(p)$:

$$\begin{aligned}
W(p) &= p + c(\theta - p)\mathrm{abamo}(p^{k-1}, d-1) + (1 - p - c(\theta - p))Q(p)^{d-1} \\
&\leq p + \theta\,\mathrm{abamo}(p^{k-1}, d-1) + Q(p)^{d-1} \\
&= p + \theta p^{(k-1)(d-1)} + \theta(d-1)(1-p^{k-1})p^{(k-1)(d-2)} + Q(p)^{d-1} \\
&\leq p + (d-1)\theta p^{(d-2)(k-1)} + R^L \\
&\leq p + (d-1)\theta p^{(d-2)(k-1)} + (p + 4\,p^L)^L \\
&\leq p + O\left(\theta p^{(d-2)(k-1)}\right)\ .
\end{aligned}$$

◀

▶ **Proposition 29** (Proposition 22, restated). $\Pr[J^{\mathrm{imp}} = 1 \wedge \mathbb{I}_x] \leq \frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)$.

Proof. Recall that if $\pi(x) < \theta$ then $\mathbb{I}_x$ is 1 with probability c and 0 with probability $1 - c$. If $\pi(x) \geq \theta$ then $\mathbb{I}_x = 0$. Also, $J^{\mathrm{imp}} = 1$ if and only if $J_1^{\mathrm{imp}} = \cdots = J_{d-1}^{\mathrm{imp}} = 0$. Therefore,

$$\begin{aligned}
\Pr[J^{\mathrm{imp}} = 1 \wedge \mathbb{I}_x] &= c \cdot \int_0^\theta \Pr[J^{\mathrm{imp}} = 1 \mid \pi(x) = p]\,dp = c \cdot \int_0^\theta W^{(d-1)(k-1)}\,dp \\
&= c \cdot \int_0^\theta (p + O(\theta p^{(d-2)(k-1)}))^L\,dp \leq c \cdot \int_0^\theta p^L(1 + O(\theta))\,dp \\
&\qquad\qquad \text{(since } (d-2)(k-1) \geq 1\text{)} \\
&= \frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)
\end{aligned}$$

This proves the proposition. ◀

▶ **Proposition 30** (Proposition 23, restated). $\mathbb{E}\left[\log_2(J^{\mathrm{imp}})\right] - S_{d,k} \leq (d-1)\log_2(1 - 1/d) \cdot \left(\frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)\right)$.

Proof. Recall the definition of $S_{d,k}$: sample random variables $J_1, \ldots, J_{d-1}$ by setting $p := \pi(x)$ and setting each J_c to 0 with probability $Q(p)$ and to 1 with probability $1 - Q(p)$, and $J = J_1 + \cdots + J_{d-1} + 1$. So the J_c are independent conditioned on $\pi(x) = p$. Then $S_{d,k} = \mathbb{E}[\log_2(J)]$. Set $\Delta_c := J_c - J_c^{\mathrm{imp}}$ and $\Delta = \sum_c \Delta_c$. Note that all Δ_c have the same distribution.

▶ **Proposition 31.** $\mathbb{E}[\Delta_1 \mid \pi(x) = p] \geq c(\theta - p)L\left(p^{L-1} - O(p^L)\right)$ *for all* $1 \leq c \leq d-1$.

In particular, if $p < \theta$ and θ is sufficiently small then $\mathbb{E}[\Delta_1] \geq 0$. Therefore, $\mathbb{E}[J_c] \leq \mathbb{E}[J_c^{\mathrm{imp}}]$ and we can couple the ensemble $(J_1, \ldots, J_{d-1})$ and $(J_1^{\mathrm{imp}}, \ldots, J_{d-1}^{\mathrm{imp}})$ on a common probability space on which $J_c \leq J_c^{\mathrm{imp}}$, always, and thus $\Delta \geq 0$. We therefore see that

$\mathbb{E}\left[\log_2(J^{\text{imp}})\right] - S_{d,k}$ is

$$\begin{aligned}
&\mathbb{E}[\left[\log_2(J^{\text{imp}}) - \log_2(J)\right] = \mathbb{E}\left[\log_2\left(1-\frac{\Delta}{J}\right)\right] \\
&\leq \mathbb{E}\left[\log_2\left(1-\frac{\Delta}{d}\right)\right] \leq \mathbb{E}\left[\log_2\left(\left(1-\frac{1}{d}\right)^{\Delta}\right)\right] \\
&= \mathbb{E}[\Delta]\log_2\left(1-\frac{1}{d}\right) \ .
\end{aligned}$$

Conditioned on $\pi(x) = p$ and using Proposition 31, this is at most

$$c(\theta-p)L\left(p^{L-1} - O(p^L)\right)(d-1)\log_2\left(1-\frac{1}{d}\right) \ .$$

We integrate this over p to get rid of the condition $\pi(x) = p$ and see that

$$\mathbb{E}\left[\log_2(J^{\text{imp}})\right] - S_{d,k} \leq (d-1)\log_2\left(1-\frac{1}{d}\right)\cdot\left(\frac{c}{L+1}\theta^{L+1} + O\left(\theta^{L+2}\right)\right) \ .$$

This concludes the proof of Proposition 30. ◀

It remains to prove Proposition 31.

Proof of Proposition 31.

$$\begin{aligned}
\mathbb{E}[\Delta_1 \mid \pi(x)=p] = \mathbb{E}[J_c - J_c^{\text{imp}} \mid \pi(x)=p] \quad &= (1-Q) - (1-W^{k-1}) = W^{k-1} - R^{k-1} \\
&\geq (k-1)(W-R)R^{k-2} \ ,
\end{aligned}$$

where the last inequality follows because $W^{k-1} = (R+W-R)^{k-1} = R^{k-1}\left(1+\frac{W-R}{R}\right)^{k-1} \geq R^{k-1}\left(1+\frac{(k-1)(W-R)}{R}\right) = R^{k-1} + (k-1)(W-R)R^{k-2}$. Now let us bound $W-R$ from below. If $p \geq \theta$ then $W(p) = R(p)$ and $W - R = 0$. If $p < \theta$, we expand $R(p)$ as follows:

$$R = p + (1-p)Q^{(d-1)} = p + c(\theta-p)Q^{d-1} + (1-p-c(\theta-p))Q^{d-1}$$

and therefore

$$\begin{aligned}
W - R &= c(\theta-p)\left(\text{abamo}(p^{k-1}, d-1) - Q^{d-1}\right) \\
&= c(\theta-p)\left(p^{(k-1)(d-1)} + (d-1)\left(1-p^{k-1}\right)p^{(k-1)(d-2)} - Q^{d-1}\right) \\
&\geq c(\theta-p)\left(p^L + (d-1)p^{(k-1)(d-2)} - (d-1)p^L - (p+O(p^2))^L\right) \\
&\geq c(\theta-p)(d-1)\left(p^{(k-1)(d-2)} - O(p^L)\right) \ .
\end{aligned}$$

Next, combining the previous two calculations, we see that

$$\begin{aligned}
\mathbb{E}[\Delta_1 \mid \pi(x)=p] &\geq (k-1)(W-R)R^{k-2} \geq (k-1)(W-R)p^{k-2} \\
&\geq (k-1)c(\theta-p)(d-1)\left(p^{(k-1)(d-2)} - O(p^L)\right)p^{k-2} \\
&\geq c(\theta-p)L\left(p^{L-1} - O(p^L)\right) \ .
\end{aligned}$$

◀