Electronic Colloquium on Computational Complexity, Report No. 179 (2018)

PPSZ on CSP Instances with Multiple Solutions

Dominik Scheder

October 31, 2018

Abstract

We study the success probability of the PPSZ algorithm on (d, k)-CSP formulas. We greatly simplify the analysis of Hertli, Hurbain, Millius, Moser, Szedlák, and myself for the notoriously difficult case that the input formula has more than one satisfying assignment.

1 Introduction

A (d, k)-CSP formula F over a variable set V is a bunch of constraints $C_1 \wedge \cdots \wedge C_m$, where each C_i is an arbitrary constraint on k variables from V. Each variable takes on a value from $[d] := \{1, \ldots, d\}$, and F is *satisfied* if all constraints are satisfied. We assume that d and k are constants, so each C_i can be represented explicitly, by its truth table, for example. We are interested in the decision problem (d, k)-CSP, which asks whether a given (d, k)-CSP formula is satisfiable. For $d = 2$, this is the well-known k-SAT problem. Note that $(d, 2)$-CSP contains d-colorability as a special case. The problem is NP-complete except for the trivial cases $k = 1$ or $d = 1$ and the non-trivial case of $(2, 2)$-CSP, which is 2-SAT and has a polynomial-time algorithm.

All other cases are NP-complete, and much effort has been invested into finding *moderately exponential* algorithms, that is, algorithms solving (d, k)-CSP in time c^n for some $c < d$. For example, Schöning's famous random walk algorithm [11] runs in time $O\left(\left(\frac{d(k-1)}{k}\right)^n \cdot \text{poly}(n)\right)$, or more precisely runs in polynomial time and returns a satisfying assignment, if there is one, with probability at least $O\left(\left(\frac{d(k-1)}{k}\right)^{-n} / \text{poly}(n)\right)$. Another example is PPZ [8], named after its authors Paturi, Pudlák, and Zane. Originally stated only

ISSN 1433-8092

for k-SAT, it runs in polynomial time and has success probability at least $2^{-n+n/k}$. It has a straightforward generalization to (d,k)-CSP. For $(d,2)$-CSP, it has been analyzed by Feder and Motwani [2]. For general (d,k)-CSP, Li, Li, Liu, and Xu [6] give a sub-optimal analysis, which has later been improved by myself [9].

Concerning (conditional) lower bounds, Patrick Traxler [12] showed that under the Exponential Time Hypothesis [5], $(d,2)$-CSP takes time at least d^{cn}, for some $c > 0$. That is, an algorithm of running time $f(d) \cdot 100^n$ solving $(d,2)$-CSP for all d is impossible under ETH. This stands in contrast to d-colorability, which is a special case of $(d,2)$-CSP and can be solved in $O(2^n \text{poly}(n))$ by a clever application of the Exclusion-Inclusion technique [1].

In this paper, we study the famous PPSZ algorithm [7] by Paturi, Pudlák, Saks, and Zane. Originally, this algorithm has been designed for k-SAT. Hertli, Hurbain, Millius, Moser, Szedlák, and myself give a version for (d,k)-CSP and analyze its success probability. Here, we greatly simplify their analysis for the notoriously difficult case that the input formula has more than one satisfying assignment. This is itself a generalization of a recent paper by Steinberger and myself [10], who simplify Timon Hertli's breakthrough analysis of PPSZ for k-CNF formulas (i.e., $d = 2$) that have more than one satisfying assignment.

1.1 The Algorithm

The algorithm PPSZ, named after its inventors Paturi, Pudlák, Saks, and Zane [7], is the currently fastest algorithm for solving k-SAT. It is a randomized algorithm, and analyzing its success probability is famously difficult. However, it is very easy to state informally. Let us sketch the algorithm when applied to a (d,k)-CSP formula F:

> **PPSZ Algorithm, Informally.** Process the variables of F in random order; when processing variable x, check for each color $c \in [d]$ whether $F|_{x=c}$ is "obviously unsatisfiable". If so, remove it from $[d]$. Let $P_x(F)$ be the remaining set of colors, i.e., those $c \in [d]$ for which $F|_{x=c}$ is not obviously unsatisfiable. Select $c \in P_x(F)$ uniformly at random and set $F := F|_{x=c}$. Then iterate.

Of course, to make this a formal algorithm, we have to state what it means to be "obviously unsatisfiable". First of all, let $S_x(F)$ be the set of all $c \in [d]$ such that $F|_{x=c}$ is satisfiable. Obviously $1 \le |S_x(F)| \le d$ if F is a satisfiable (d,k)-CSP formula. Second, suppose the algorithm has access to a proof heuristic P which, for every (d,k)-CSP formula F and variable x, outputs a set $P_x(F)$ with $S_x(F) \subseteq P_x(F) \subseteq [d]$. Note that P is *sound*, meaning that $c \notin P_x(F)$ implies that $F|_{x=c}$ is unsatisfiable. It might, however, be *incomplete*, namely if $S_x(F) \subsetneq P_x(F)$. We call $S_x(F)$ the set of *satisfiable colors* and $P_x(F)$ the set of *plausible colors*.

Algorithm 1 Generic PPSZ Procedure for Fixed Permutation

1: **procedure** `ppsz_fixed`(F, P, π)
2:     $\beta :=$ the empty assignment on V
3:     **for** $x \in V$ in the order of π **do**
4:         Select $c \in P_x(F|_\beta)$ uniformly at random
5:         $\beta(x) := c$
6:     **end for**
7:     **return** β
8: **end procedure**

Algorithm 2 Generic PPSZ Procedure

1: **procedure** `ppsz`(F, P)
2:     $\pi :=$ a uniformly random permutation of $\text{var}(F)$
3:     **return** `ppsz_fixed`(F, P, π)
4: **end procedure**

It is understood that `ppsz_fixed` returns `failure` if the set $P(F|_\beta, x)$ is empty at any point in Line 4, or if β does not satisfy F. Let $\text{var}(F)$ denote the set of variables in F and $\text{sat}(F)$ the set of satisfying assignments. Let π a permutation of $\text{var}(F)$, $x \in \text{var}(F)$, and $\alpha \in \text{sat}(F)$. We write $F|_{\pi,x,\alpha}$ to denote the formula that arises if we fix y to $\alpha(y)$ for all variables y that appear before x in π. Write $S_x(F,\pi,\alpha) := S_x(F|_{\pi,x,\alpha})$ and $P_x(F,\pi,\alpha) := P_x(F|_{\pi,x,\alpha})$. If F is understood, we might write $S_x(\pi,\alpha)$ and $P_x(\pi,\alpha)$ for brevity. We can give an exact formula for the success probability of PPSZ in terms of these expressions:

Observation 1. *Let α be a satisfying assignment of F. The probability that* `ppsz_fixed`(F, P, π) *returns α is* $\prod_{x \in \text{var}(F)} \frac{1}{|P_x(F,\pi,\alpha)|}$.

Using this observation, one calculates:

$$\begin{aligned}
\Pr[\texttt{ppsz}(F,P)=\alpha] &= \mathop{\mathbb{E}}_{\pi}\left[\prod_{x\in\mathrm{var}(F)}\frac{1}{|P_x(F,\pi,\alpha)|}\right] \\
&= \mathop{\mathbb{E}}_{\pi}\left[2^{-\sum_{x\in\mathrm{var}(F)}\log|P_x(F,\pi,\alpha)|}\right] \\
&\geq 2^{-\sum_{x\in\mathrm{var}(F)}\mathbb{E}_{\pi}[\log|P_x(F,\pi,\alpha)|]} \ . \qquad (1)
\end{aligned}$$

Next, we define a way to gauge the power of P.

Definition 2. *Let P be a proof heuristic. We say P has* error rate at most γ against (d,k)-CSP formulas *if $\mathbb{E}_{\pi}[\log|P_x(F,\pi,\alpha)|] \leq \gamma$ for all (d,k)-CSP formulas F, all $\alpha \in \mathrm{sat}(F)$, and all $x \in \mathrm{var}(F)$ for which $|S_x(F)| = 1$.*

We need the condition $|S_x(F)| = 1$ since otherwise we cannot state anything useful about $P_x(F,\pi,\alpha)$. For example, it might simply be that $S_x(F) = [d]$ and $P_x(F,\pi,\alpha) = [d]$ for all π and α. Of course, one might be able to bound $\mathbb{E}_{\pi}[\log|P_x(F,\pi,\alpha)|]$ if $|S_x(F)| = 2$, say. However, we do not know how to use such a bound.

Observation 3. *If F is a (d,k)-CSP formula with a unique satisfying assignment α, and P has error rate at most γ against (d,k)-CSP formulas, then $\Pr[\texttt{ppsz}(F,P)=\alpha] \geq 2^{-\gamma n}$.*

XXX move to top: This follows from (1) since $|S_x(F)| = |\{\alpha(x)\}| = 1$ for all variables x.

1.2 Which Error Rates Are Possible

In [4], Hertli, Hurbain, Millius, Moser, Szedlák, and myself have analyzed the "usual" proof heuristic P_D: it checks whether there is a set $G \subseteq F$ of D constraints for which $G|_{x=c}$ is unsatisfiable (and includes c into $P_x(F)$ if this is not the case). The parameter D is usually taken to be some slowly increasing function $D(n)$, slowly enough so that P_D can be implemented to run in time $2^{o(n)}$. It was shown in [4] that P_D has error rate at most $\gamma_{d,k} + o_D(1)$. Here, $\gamma_{d,k}$ can be defined by the following random experiment (next paragraph copied almost literally from [4]):

> Let T be an infinite rooted tree in which every even-level vertex (this includes the root, which has level 0) has $k-1$ children, and every odd-level vertex has $d-1$ children (there are no leafs). Take $d-1$ disjoint copies of T, choose a value $p \in [0,1]$ uniformly

at random, and delete each odd-level vertex of the $d-1$ trees with probability p, independently. Let Y be the number of trees in which this deletion still leaves an infinite path starting at the root. Obviously, Y is a random variable and $0 \le Y \le d-1$. Define $\gamma_{d,k} := \mathbb{E}[\log(1+Y)]$.

Lemma 4 (Lemma 2.5 from [4])**.** *P_D has error rate at most $\gamma_{d,k}+o(1)$ against (d,k)-CSP formulas.*

Observation 3 immediately implies the following theorem:

Theorem 5 (Theorem 1.1 from [4])**.** *Let F be a (d,k)-CSP formula over n variables. If F has a unique satisfying assignment, then PPSZ outputs it with probability at least $2^{-\gamma_{d,k}n-o(n)}$.*

The theorem, as stated, raises the obvious question whether we can remove the condition that F have a unique satisfying assignment. For $d=2$ (the Boolean case, k-SAT), and $k \ge 5$, this has already been proved by Paturi, Pudlák, Saks, and Zane [7]. Unfortunately, their proof ceases to work for $k=3,4$ and has to look inside P_D in great detail. In a breakthrough paper, Hertli [3] gave a more abstract proof that also works for $d=3,4$, and works for almost any proof heuristic P. Still, his proof is very technical and somewhat obscure. Also, it only works provided that P is "not too strong", i.e., only as long as its error rate γ is at least $1-\frac{\log(e)}{2}$. Since this holds for the concrete heuristic P_D, this is a purely hypothetical limitation to his proof. We simply don't know of any proof heuristic that runs in subexponential time and has an error rate less than $1-\frac{\log(e)}{2}$. In [4], Hertli, Hurbain, Millius, Moser, Szedlák, and myself generalize Hertli's result to (d,k)-CSP problems:

Definition 6. *A proof heuristic P is* monotone *if $P_x(F|_{y=c}) \subseteq P_x(F)$, for all formulas F, variables $x \neq y$, and colors c.*

Intuitively, this very mild condition states that adding information can never hurt the heuristic. The reader is welcome to verify that P_D is monotone.

Theorem 7 (Theorem 1.2 from [4])**.** *Let F be a satisfiable (d,k)-CSP and P a monotone proof heuristic of error rate at most γ against (d,k)-CSP formulas. Then* `ppsz`*(F,P) finds a satisfying assignment of F with probability at least $2^{-\gamma' n-o(n)}$, where $\gamma' = \max(\gamma, \theta_d)$ and $\theta_d := \log(d) - \frac{\log(e)}{2}$.*

Lemma 1.3 of [4] states that for $k \ge 4$ it holds that $\gamma_{d,k} \ge \theta_d$, and thus the bound of Theorem 7 matches that of Theorem 5, when using P_D as proof heuristic. The same is true for $d=2$ (the Boolean case). However, if $k=2$ and $d \ge 3$ or if $k=3$ and $d \ge 6$, it happens that $\gamma_{d,k} < \theta_d$, and thus Hertli's hypothetical limitation of too strong a proof heuristic suddenly becomes real.

1.3 Our Contribution

In this paper, we reprove Theorem 7, giving a proof that is simpler, shorter, and more accessible than the original one from [4]. This builds upon work from Steinberger and myself [10], who simplified Hertli's proof for $d = 2$. However, the non-Boolean case $d \geq 3$ comes with additional subtleties that need to be addressed. Our proof also highlights where the mysterious θ_d comes from.

1.4 Open Questions

We suspect that Theorem 7 holds true when we replace γ' by γ. That is, the limitation $\gamma \geq \theta_d$ is simply an artifact of the proof. However, we currently cannot prove this, not even for the Boolean case $d = 2$.

2 Proof of Theorem 7

For a CSP formula F and $x \in \text{var}(F)$, recall that $S_x(F)$ denotes the set of satisfying colors, i.e., the set of all $c \in [d]$ such that $F|_{x=c}$ is satisfiable. Let $S(F)$ be the set $\{(x, c) \in \text{var}(F) \times [d] \mid c \in S_x(F)\}$. If F is unsatisfiable then $S(F)$ is obviously empty. Otherwise, $|S(F)|$ is at least n and at most dn. Recall that $F|_{\pi,x,\alpha}$ denotes the formula that arises if we fix y to $\alpha(y)$ for all variables y that appear before x in π, and $S_x(F, \pi, \alpha) := S_x(F|_{\pi,x,\alpha})$ and $P_x(F, \pi, \alpha) := P_x(F|_{\pi,x,\alpha})$. If F is understood, we might write $S_x(\pi, \alpha)$ and $P_x(\pi, \alpha)$ for brevity.

For a satisfiable formula F, consider the following process R_F: Start with $F' := F$. Sample a pair $(x, c) \in S(F)$ uniformly at random and set $F' := F'|_{x=c}$. Then continue until all variables have been set. Let π denote the order in which the variables have been chosen and let α denote the resulting satisfying assignment. This defines a probability distribution R_F, or simply R, if F is understood.

Observation 8. $R(\pi, \alpha) = \prod_{x \in \text{var}(F)} \frac{1}{|S(F|_{\pi,x,\alpha})|}$.

As an aside, suppose we call `ppsz`(F, S), i.e., we give it access to a complete proof heuristic (which is computationally intractable, of course). Since S is complete, it will never output `failure`. Denote by $Q(\pi, \alpha)$ the probability that PPSZ chooses π and outputs α. One checks that $Q(\pi, \alpha) = \frac{1}{n!} \cdot \prod_{x \in \text{var}(F)} \frac{1}{|S_x(\pi,\alpha)|}$. The reader is invited to check three things: first, Q and R are indeed different probability distributions. Second, and less obviously, Q and R induce the same marginal distribution on $\text{sat}(F)$ if $d = 2$, i.e., in the Boolean case. This is mentioned, without proof, in Scheder and Steinberger [10]. Third, once $d \geq 3$, however, those marginal distributions are not the same in general.

One of the difficulties in proving Theorem 7 is that, if F is multiple satisfying assignments, the bound in (1) can be exponentially smaller than $2^{-\gamma n}$. In fact, it can be exponentially small even for the complete proof heuristic ($\gamma = 0$). Steinberger and myself [10] exhibit a CNF formula with this behavior. This problem disappears once we do not apply Jensen's inequality to the uniform distribution over π (as in the derivation of (1), but "skew" that distribution using the R defined above. In what follows, we treat $P_x(\pi, \alpha)$ and $S_x(\pi, \alpha)$ as random variables over the probability space $\mathrm{Sym}(\mathrm{var}(F)) \times \mathrm{sat}(F)$ and sometimes drop the argument, i.e., write P_x and S_x if π, α are understood from the context.

$$
\begin{aligned}
\Pr[\text{success}] &= \sum_{\alpha} \mathop{\mathbb{E}}_{\pi}\left[\prod_{x} \frac{1}{|P_x(\pi,\alpha)|}\right] = \sum_{\pi,\alpha} \frac{1}{n!} \prod_{x} \frac{1}{|P_x|} \\
&= \mathop{\mathbb{E}}_{(\pi,\alpha)\sim R}\left[\frac{1}{n! R(\pi,\alpha)} \prod_{x} \frac{1}{|P_x|}\right] \\
&\geq 2^{-\mathbb{E}_R[\log(n! R(\pi,\alpha))] - \sum_x \mathbb{E}_R[\log |P_x|]} . \qquad (2)
\end{aligned}
$$

To complete the proof, it suffices to prove the following lemma.

Lemma 9. $\mathbb{E}_R[\log(n!R(\pi, \alpha))] + \sum_x \mathbb{E}_R[\log |P_x|] \leq \gamma' n$.

Proof. Recall that $S_x = S_x(\pi, \alpha)$ is the set of satisfying colors for x at the point in time when x is processed. Let L_x be the indicator variable that is 1 if $[S_x| \geq 2$ and 0 otherwise. We call x *frozen in* F if $|S_x(F)| = 1$, and *liquid* otherwise. Thus, L_x is the indicator variable of x being liquid. Recall that P has error rate at most γ, and $\gamma' := \max\left(\gamma, \log(d) - \frac{\log(e)}{2}\right)$. Let $\epsilon := \log(d) - \gamma' = \min\left(\log(d) - \gamma, \frac{\log(e)}{2}\right)$.

Lemma 10. $\mathbb{E}_R[\log |P_x|] \leq \gamma' + \mathbb{E}_R[\epsilon L_x]$.

Proof. Let the R process run until one of two things happens: (1) x freezes, i.e., $|S_x(F')|$ drops to 1; (2) x is selected as the next variable in π. If (2) happens first, then $L_x = 1$. Since $|P_x| \leq d$ always, the left-hand side is at most $\log(d)$ and the right hand side is $\gamma' + \epsilon = \log(d)$. If (1) happens first, then $L_x = 0$. Let F' be the restricted formula after (1) has happened, and U the uniform distribution over permutations of $\mathrm{var}(F')$. We know that $\mathbb{E}_{\pi\sim\mathcal{U}}[\log |P_x(F', \pi, \alpha)|] \leq \gamma$, for every fixed $\alpha \in \mathrm{sat}(F')$, since P has error rate at most γ by assumption. However, this is not exactly what we need. We need to show that

$$\mathop{\mathbb{E}}_{(\pi,\alpha)\sim R}[\log|P_x(F,\pi\alpha)| \mid (1) \text{ and } F'] \leq \gamma \ ,$$

where the condition means that (1) happens first, and F' is the restricted formula just after (1) happens. Sampling from R conditioned on this specific past is basically the same as running the R-process, as defined above, on F' instead of F. That is, we have to show that

$$\mathop{\mathbb{E}}_{(\pi,\alpha)\sim R_{F'}}[\log|P_x(F',\pi,\alpha)|] \leq \gamma \ . \tag{3}$$

Note that F' is again a (d,k)-CSP formula, and thus it suffices to show the following lemma and apply it to F.

Lemma 11. *Let F be a satisfiable (d,k)-CSP formula and R the distribution over pairs (π,α) defined above. Then $\mathbb{E}_{(\pi,\alpha)\sim R}[\log|P_x(F,\pi,\alpha)|] \leq \gamma$.*

This lemma is non-obvious since we take the expectation over R, where both α and π are random, and π is typically not uniform, whereas in the definition of error rate, we take a fixed α and a uniform π. This lemma is, in a way, the heart of the overall proof, and also justifies the definition of R. We will prove it in the next section. To summarize, we have showed that if (2) happens first, then $L_x = 1$ and $\log|P_x| \leq \log d$ trivially, and the claimed bound holds. If (1) happens first, then $L_x = 0$, and by Lemma 11, $\mathbb{E}_R[\log|P_x| \mid (1)] \leq \gamma \leq \gamma'$. Thus, the claimed bound holds in both cases, which concludes the proof of Lemma 10, except for Lemma 11, a proof of which we give in the next section. □

Plugging the bound of Lemma 10 into the left-hand side of Lemma 9, it suffices to show that

$$\mathbb{E}[\log(n!R(\pi,\alpha))] + \epsilon \sum_x \mathbb{E}[L_x] \leq 0 \ . \tag{4}$$

As an aside, note that the veracity of the above inequality depends solely on $\text{sat}(F)$ as a subset of $[d]^n$, and neither on the way this set is presented as a (d,k)-CSP formula nor on the proof heuristic P. Let us now change perspective. Rather than summing over individual variables x, let us sum / multiply over the steps taken by the R-process. Consider the i^{th} step, let F_i be the formula at the beginning of step i, let $n_i := n - i + 1$ be the number variables in F_i, let $s_i := |S(F_i)| = \sum_{x\in\text{var}(F_i)} |S_x(F_i)|$ be the number of satisfying literals. Let x_i be the variable chosen in step i and $L_i := L_{x_i}$

the indicator variable which is 1 if this variable is liquid and 0 if frozen. All of these are random variables with respect to the underlying probability distribution R (except n_i, of course, which is known beforehand). Note that $R(\pi, \alpha) = \prod_{i=1}^{n} \frac{1}{s_i}$. Indeed, there is, in every step, exactly one choice out of s_i many to produce the pair (π, α). Consequently, $\log(n! R(\pi, \alpha)) = \sum_{i=1}^{n} \log\left(\frac{n_i}{s_i}\right)$, and (4) is equivalent to

$$\sum_{i=1}^{n} \mathbb{E}\left[\log\left(\frac{s_i}{n_i}\right) - \epsilon L_i\right] \geq 0 \ . \tag{5}$$

In the next lemma, we will show that every summand is non-negative. This concludes the proof of Lemma 9. □

Lemma 12. *For every $1 \leq i \leq n$, it holds that $\mathbb{E}\left[\log\left(\frac{s_i}{n_i}\right) - \epsilon L_i\right] \geq 0$.*

Proof. Let us imagine the R-process has already finished the first $i-1$ steps, and interpret $\mathbb{E}$ as conditioned on this past. Then s_i becomes a constant. Let us write s and n instead of s_i and n_i, to simplify no tation. Let f be the number of frozen variables in F_i. Note that $\mathbb{E}[L_i] = 1 - \frac{f}{s}$. Also, every liquid variable has at least 2 possible colors, thus $s \geq f + 2(n-f) = 2n - f$ and $f \geq 2n - s$. Therefore,

$$\mathbb{E}[L_i] = 1 - \frac{f}{s} \leq 1 - \frac{2n-s}{s} = \frac{2s-2n}{s} \ .$$

Note that this upper bound can easily be larger than 1, in which case it is of course trivial. Next, we will bound $\log\left(\frac{s}{n}\right)$ from below:

$$\begin{aligned} \log\left(\frac{s}{n}\right) &= -\log\left(\frac{n}{s}\right) = -\log\left(1 - \frac{s-n}{s}\right) \\ &= -\log(e) \ln\left(1 - \frac{s-n}{s}\right) \\ &\geq \log(e) \cdot \frac{s-n}{s} \ . \end{aligned}$$

Combining these two bounds, we obtain

$$\begin{aligned} \mathbb{E}\left[\log\left(\frac{s_i}{n_i}\right) - \epsilon L_i\right] &\geq \log(e) \cdot \frac{s-n}{s} - \epsilon \cdot \frac{2s-2n}{s} \\ &= \frac{s-n}{s} \cdot (\log(e) - 2\epsilon) \ . \end{aligned}$$

Obviously $\frac{s-n}{s} \geq 0$. Note that $\log(e) - 2\epsilon$ is non-negative since $\epsilon \leq \frac{\log(e)}{2}$. □

3 Bounding $\mathbb{E}_R[\log |P_x|]$—Proof of Lemma 11

We have to show that $\mathbb{E}_{(\pi,\alpha)\sim R}[\log |P_x(F,\pi,\alpha)|] \leq \gamma$. For a fixed $\alpha \in \mathrm{sat}(F)$, let $R_\alpha(\pi) := R(\pi|\alpha)$. So R_α is a distribution over permutations of variables. We can rewrite our expression as

$$\mathop{\mathbb{E}}_{(\pi,\alpha)\sim R}[\log |P_x|] = \mathop{\mathbb{E}}_{\alpha\sim R}\left[\mathop{\mathbb{E}}_{\pi\sim R_a}[\log |P_x|]\right] , \tag{6}$$

and we will show that this is at most γ by showing that

$$\mathop{\mathbb{E}}_{\pi\sim R_a}[\log |P_x|] \leq \gamma \ , \text{ for every fixed } \alpha \in \mathrm{sat}(F) \ . \tag{7}$$

Since the R-process samples π and α in a very intertwined way, it is by no means clear how R_α behaves, and how (7) should be proved.

3.1 Informal Proof Outline

Let us first give an intuitive proof outline. Let π be some permutation and consider $\log |P_x|$. Suppose we move x towards the back of π, creating a new permutation π'. This can make P_x smaller. In fact, $P_x(\pi', \alpha) \subseteq P_x(\pi, \alpha)$. This should be clear: the later x is processed, the more information the heuristic P has about x, the more colors can be excluded as non-satisfying.

We claim that in $\pi \sim R_\alpha$, the variable x tends to come later than in a uniform permutation (in a sense we make precise soon). This means that P_x tends to be smaller under R_α than under the uniform distribution, thus $\mathbb{E}_{\pi\sim R_\alpha}[\log |P_x|] \leq \mathbb{E}_{\pi\sim\mathcal{U}}[\log |P_x|]$, and the latter expectation is at most γ by definition of the error rate of the proof heuristic.

What is an intuitive reason that x tends to come later under R_α? When we remove the condition on α and only consider R, then a variable y is chosen first with probability $\frac{|S(F,y)|}{|S(F)|}$. Since x is frozen, we have $|S_x(F)| = 1$ and thus it is least likely to come first. The same argument applies to every step of the R-process. However, this is not what we want—we want to show x tends to come late under R_α, not R. Bayes' formula tells us that we have to correct $\frac{|S(F,y)|}{|S(F)|}$ by a factor that measures how the probability of α changes when conditioning on y being chosen first. A minute of thought shows that choosing y first changes α's probability to something at least $\frac{1}{|S(F,y)|}$ times what it was before; furthermore, if y is frozen, it does not change it at all. That is, under R_α, x comes first with probability $\frac{1}{|S(F)|}$, while some other y comes first with probability at least $\frac{|S(F,y)|}{|S(F)|} \times \frac{1}{|S(F,y)|}$. So x is still least likely to come first.

3.2 The Formal Proof

For two strings σ, π, we write $\sigma \preceq \pi$ if σ is a prefix of π. A permutation π on set V of size n can be viewed as a string in V^n without repeated letters. A string $\sigma \in V^*$ without repeated letters is called a *partial permutation.* If D is a distribution over permutations on V and σ is a partial permutation, we write $D(\sigma) := \Pr_{\pi \sim D}(\sigma \preceq \pi) = \sum_{\pi : \sigma \preceq \pi} D(\pi)$.

Definition 13. *Let D be a distribution over permutations on V, and let $x \in V$. We say D* delays *x if for all $y \in V$ and all partial permutations σ not containing x or y, it holds that $D(\sigma x) \leq D(\sigma y)$.*

Lemma 11 will follow from the next two lemmas.

Lemma 14. *Let x be a frozen variable. Then the distribution R_α delays x*

Lemma 15 (Lemma 21 from [10]). *Let V be a finite set, $x \in V$, D a distribution over permutations of V that delays x, and $f : V \to \mathbb{R}$ a monotone function, meaning $f(U_1) \leq f(U_2)$ whenever $U_1 \subseteq U_2$. Denote by $W = W(\pi)$ the set of elements coming after x in π. Then*

$$\mathop{\mathbb{E}}_{\pi \sim D}[f(W)] \leq \mathop{\mathbb{E}}_{\pi \sim \mathcal{U}}[f(W)] \ ,$$

where $\mathcal{U}$ is the uniform distribution over permutations.

To finish the proof of Lemma 11, let W be the set of variables occurring after x in π, and note that P_x indeed only depends on W: it depends on which variables have been set yet; the order in which they have been set is irrelevant. Also, it is monotone. The fewer variables come after x, the more information the heuristic P has about x, the more colors can be excluded as obviously non-satisfying, and the smaller P_x becomes. Thus, $f : W \mapsto \log |P_x|$ is monotone in the sense of Lemma 15. Lemma 11 now follows since R_α delays x and thus can play the role of D in Lemma 15.

Lemma 15 can be proved by a simple coupling argument. A full formal proof can be found in [10]. It remains to prove Lemma 14. The proof is similar to that of Lemma 21 in [10].

Proof of Lemma 14. Let x and y be variables and let σ be a partial permutation not containing x or y. We have to show that $R_\alpha(\sigma x) \leq R_\alpha(\sigma y)$. Multiplying both sides with $R(\alpha)$, this is equivalent to $R(\sigma x, \alpha) \leq R(\sigma y, \alpha)$. Note that $R(\sigma, \alpha)$ means $\sum_{\pi : \sigma \preceq \pi} R(\pi, \alpha)$. This is good, since we removed the hard-to-grasp conditioning on α.

We argue that it is enough to show this inequality for empty σ, i.e., $R(x,\alpha) \le R(y,\alpha)$. Indeed, if σ is non-empty, one can imagine running the R-process according to α and σ for the first $|\sigma|$ steps, arriving at a formula F', and then appeal to the case of empty prefix, for the new formula F'.

To prove $R(x,\alpha) \le R(y,\alpha)$, we consider an alternative way to sample $(\pi,\alpha) \sim R$. First, recall that $S(F)$ is the set of all (v,c) such that $F|_{v=c}$ is satisfiable. Choose a random permutation on $S(F)$, namely $\tau = (v_1,c_1),(v_2,v_2),\dots,(v_s,c_s)$. Start with $F' = F$ and, for $i = 1,\dots,s$, check whether (1) v_i has not been set yet and (2) $F'|_{v_i=c_i}$ is satisfiable. If so, apply this assignment, i.e., set $F' := F'|_{v_i=c_i}$, and output (v_i,s_i). Otherwise, do nothing. Proceed to $i+1$. The output sequence has length n and defines a permutation π and a satisfying assignment α. We say τ *leads to* π and α. It is easy to see that (π,α) follows distribution R.

Let T be the set of all such sequences, i.e., $T = \mathrm{Sym}(S(F))$. Let $T_{v,\alpha}$ be the set of all $\tau \in T$ leading to α and some π in which v comes first. With this notation, $R(v,\alpha) = \frac{|T_{v,\alpha}|}{|T|}$. Thus, we have to show that $|T_{x,\alpha}| \le |T_{y,\alpha}|$. We show this by defining an injection from the former set into the latter as follows. Consider a sequence $\tau \in T_{x,\alpha}$. The first pair in τ must be $(x,\alpha(x))$, and the pair $(y,\alpha(y))$ must appear somewhere in τ. Let $\varphi(\tau)$ be the sequence arising from exchanging these two pairs. Clearly φ is injective, and we claim that $\varphi(\tau) \in T_{y,\alpha}$. Indeed, since τ leads to α, we can move the pair $(x,\alpha(x))$ to any position, and the new sequence still leads to α. This is because x is frozen, and thus $\alpha(x)$ is the only satisfying value for x anyway, and it does not matter when we actually apply the restriction $x = \alpha(x)$. So $\varphi(\tau)$ leads to α, as well, and its first pair is $(y,\alpha(y))$, so clearly y comes first in $\varphi(\tau)$, thus $\varphi(\tau) \in T_{y,\alpha}$.

□

4 Conclusion

We would like to find a proof that works for all monotone proof heuristics, for arbitrarily small error rates γ. More modestly, it would be nice if we could extend the proof of this paper to cover all values (d,k) for the concrete proof heuristic P_D, or at least more than it currently does. There are two points in the proof where we bound things too generously: first, in the proof of Lemma 11 we bound $\log|P_x| \le \log(d)$, which is of course true. However, it is too pessimistic on expectation. For example, suppose $|S_x(F)| = 2$. Then $|P_x(\pi,\alpha)|$ should still be less than d, on expectation. Indeed, for any unsatisfiable color c, there is a certain chance that P_D "catches" it and thus excludes it from $P_x(\pi, \alpha)$. This probability is not formalized in the definition of γ, the error rate. However, we surely can bound it for the concrete heuristic P_D. However, there are some difficulties: if $|S_x(F)| \geq 2$, then Lemma 14 does not apply anymore, and R_α quite possibly does not delay x. Thus, $\mathbb{E}_{\pi \sim R_\alpha}[\log |P_x(\pi, \alpha)|]$ might be significantly larger than under uniform π.

The other point in the proof where we possibly give away a lot is in the proof of Lemma 12. We argue that "every liquid variable has at least 2 possible colors. This might well be tight: $\text{sat}(F)$ might well be contained in the set $\{1, 2\}^n$, for example. However, this is an extreme case, and the pessimistic scenario that $|S_x(F)| \leq 2$ for all (or even most) x might turn out to be beneficial at some other point in the analysis.

References

[1] A. Björklund and T. Husfeldt. Inclusion–exclusion algorithms for counting set partitions. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 575–582. IEEE Computer Society, 2006.

[2] T. Feder and R. Motwani. Worst-case time bounds for coloring and satisfiability problems. *J. Algorithms*, 45(2):192–201, 2002.

[3] T. 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.

[4] T. Hertli, I. Hurbain, S. Millius, R. A. Moser, D. Scheder, and M. Szedlák. The PPSZ algorithm for constraint satisfaction problems on more than two colors. In M. Rueher, editor, *Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings*, volume 9892 of *Lecture Notes in Computer Science*, pages 421–437. Springer, 2016.

[5] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity. *J. Comput. System Sci.*, 63(4):512–530, 2001. Special issue on FOCS 98 (Palo Alto, CA).

[6] L. Li, X. Li, T. Liu, and K. Xu. From k-sat to k-csp: Two generalized algorithms. *CoRR*, abs/0801.3147, 2008.

[7] R. Paturi, P. Pudlák, M. E. Saks, and F. Zane. An improved exponential-time algorithm for k-SAT. *J. ACM*, 52(3):337–364 (electronic), 2005.

[8] R. Paturi, P. Pudlák, and F. Zane. Satisfiability coding lemma. *Chicago J. Theoret. Comput. Sci.*, pages Article 11, 19 pp. (electronic), 1999.

[9] D. Scheder. Ppz for more than two truth values - an algorithm for constraint satisfaction problems. *CoRR*, abs/1010.5717, 2010.

[10] D. Scheder and J. P. Steinberger. PPSZ for General k-SAT - making Hertli's analysis simpler and 3-SAT faster. In R. 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.

[11] U. 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.

[12] P. Traxler. The time complexity of constraint satisfaction. In M. Grohe and R. Niedermeier, editors, *Parameterized and Exact Computation, Third International Workshop, IWPEC 2008, Victoria, Canada, May 14-16, 2008. Proceedings*, volume 5018 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 2008.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il