Tight Thresholds for Cuckoo Hashing via XORSAT

Martin Dietzfelbinger*, Andreas Goerdt**, Michael Mitzenmacher***,
Andrea Montanari†, Rasmus Pagh‡, and Michael Rink*

Abstract. We settle the question of tight thresholds for offline cuckoo hashing. The problem can be stated as follows: we have n keys to be hashed into m buckets each capable of holding a single key. Each key has $k \geq 3$ (distinct) associated buckets chosen uniformly at random and independently of the choices of other keys. A hash table can be constructed successfully if each key can be placed into one of its buckets. We seek thresholds c_k such that, as n goes to infinity, if $n/m \leq c$ for some $c < c_k$ then a hash table can be constructed successfully with high probability, and if $n/m \geq c$ for some $c > c_k$ a hash table cannot be constructed successfully with high probability. Here we are considering the offline version of the problem, where all keys and hash values are given, so the problem is equivalent to previous models of multiple-choice hashing. We find the thresholds for all values of $k > 2$ by showing that they are in fact the same as the previously known thresholds for the random k-XORSAT problem. We then extend these results to the setting where keys can have differing number of choices, and provide evidence in the form of an algorithm for a conjecture extending this result to cuckoo hash tables that store multiple keys in a bucket.

1 Introduction

Consider a hashing scheme with n keys to be hashed into m buckets each capable of holding a single key. Each key has $k \geq 3$ (distinct) associated buckets chosen uniformly at random and independently of the choices of other keys. A hash table can be constructed successfully if each key can be placed into one of its buckets. This setting describes the offline load balancing problem corresponding to multiple choice hashing [1] and cuckoo hashing [14,26] with $k \geq 3$ choices. An open question in the literature (see, for example, the discussion in [24]) is to determine a tight threshold c_k such that if $n/m \leq c$ for some $c < c_k$ then a hash table can be constructed successfully with high probability, and if $n/m \geq c$ for some $c > c_k$ a hash table cannot be constructed successfully with high probability. In this paper, we provide these thresholds.

We note that, in parallel with this work, two other papers have similarly provided means for determining the thresholds [13,15]. Our work differs from these works in substantial ways. Perhaps the most substantial is our argument that, somewhat surprisingly, the thresholds we seek were actually essentially already known. We show that tight thresholds follow from known results in the literature, and in fact correspond exactly to the known thresholds for the random k-XORSAT problem. We describe the k-XORSAT problem and the means for computing its thresholds in more detail in the following sections. Our argument is somewhat indirect, although all of the arguments appear to rely intrinsically on the analysis of corresponding random hypergraphs, and hence the alternative arguments of [13,15] provide additional insight that may prove useful in further explorations.

* Fakultät für Informatik und Automatisierung, Technische Universität Ilmenau. Research supported by DFG grant DI 412/10-1. {martin.dietzfelbinger,michael.rink}@tu-ilmenau.de

** Fakultät für Informatik, Technische Universität Chemnitz. goerdt@informatik.tu-chemnitz.de

*** Harvard University, School of Engineering and Applied Sciences. Part of this work was done while visiting Microsoft Research New England. michaelm@eecs.harvard.edu

† Department of Electrical Engineering and Department of Statistics, Stanford University. Part of this work was done while visiting Microsoft Research New England. montanar@stanford.edu

‡ Efficient Computation group, IT University of Copenhagen. pagh@itu.dk

With this starting point, we extend our study of the cuckoo hashing problem in two ways. First, we consider *irregular cuckoo hashing*, where the number of choices corresponding to a key is not a fixed constant k but itself a random variable depending on the key. Our motivations for studying this variant include past work on irregular low-density parity-check codes [20] and recent work on alternative hashing schemes that have been said to behave like cuckoo hashing with "3.5 choices" [18]. Beyond finding thresholds, we show how to optimize irregular cuckoo hashing schemes with a specified average number of choices per key; for example, with an average of 3.5 choices per key, the optimal scheme is the natural one where half of the keys obtain 3 choices, and the other half obtain 4. Second, we consider the generalization to the setting where a bucket can hold more than one key. We provide a conjecture regarding the appropriate threshold behavior for this setting, and provide a simple algorithm that, experimentally, appears to perform remarkably close to the thresholds predicted by our conjecture.

Section 2 presents an exposition of known results on cores of random hypergraphs. Readers familiar with this material may want to skip directly to Section 3, which provides our proof that the thresholds for k-XORSAT and k-ary cuckoo hashing are identical. In Section 4 we extend the discussion of thresholds to the case where k is any real number greater than 2. Finally, Section 5 presents our simple algorithm to construct hash tables and presents experimental evidence that it is able to achieve load factors close to the thresholds. Further details appear in the appendices.

2 Technical background on cores

The key to our analysis will be the behavior of cores in random hypergraphs. We therefore begin by providing a review of this subject. To be clear, the results of this section are not new; the reader is encouraged to see [23, Ch. 18], as well as references [7,10,25] for more background.

We consider the set of all k-uniform hypergraphs with m nodes and n hyperedges $\mathcal{G}^k_{m,n}$. More precisely, each hypergraph G from $\mathcal{G}^k_{m,n}$ consists of n (labeled) hyperedges of a fixed size $k \geq 2$, chosen independently at random, with repetition, from the $\binom{m}{k}$ subsets of $\{1, \ldots, m\}$ of size k. This model will be regarded as a probability space. We always assume k is fixed, m is sufficiently large, and $n = cm$ for a constant c.

For $\ell \geq 2$, the *ℓ-core* of a hypergraph G is defined as the largest induced sub-hypergraph that has minimum degree ℓ or larger. It is well known that the ℓ-core can be obtained by the following iterative "peeling process": While there are nodes with degree smaller than ℓ, delete them and their incident hyperedges. By pursuing this process backwards one sees that the ℓ-core, conditioned on the number of nodes and hyperedges it contains, is a uniform random hypergraph that satisfies the degree constraint.

The fate of a fixed node a after a fixed number of h iterations of the peeling procedure is determined by the h-neighborhood of a, where the h-neighborhood of a is the sub-hypergraph induced on the nodes at distance at most h from a. For example, the 1-neighborhood contains all hyperedges containing a. In our setting where n is linear in m the h-neighborhood of node a is a hypertree of low degree (at most $\log \log m$) with high probability. We assume this in the discussion to come.

We can see whether a node a is removed from the hypergraph in the course of h iterations of the peeling process in the following way. Consider the hypertree rooted from a (so the children are nodes that share a hyperedge with a, and similarly the children of a node share a hyperedge with that node down the tree). First, consider the nodes at distance $h-1$ from a and delete them if they have at most $\ell-2$ child hyperedges; that is, their degree is at most $\ell-1$. Second, treat the nodes at distance $h-2$ in the same way, and so on, down to distance 1, the children of a. Finally, a is deleted if its degree is at most $\ell-1$.

The analysis of such random processes on trees has been well-studied in the literature. (See, for example, [4,19] for similar analyses.) We wish to determine the probability q_h that node a is deleted after h rounds of

the peeling process. For $j < h$ let p_j be the probability that a node at distance $h - j$ from a is deleted after j rounds of the peeling process. The discussion becomes easier for the binomial random hypergraph with an expected number of cm hyperedges: Each hyperedge is present with probability $k! \cdot c/m^{k-1}$ independently. It is well known that $\mathcal{G}^k_{m,n}$ and the binomial hypergraph are equivalent as far as asymptotic behavior of cores are concerned when c is a constant.

Let $\mathrm{Bin}(N, p)$ denote a random variable with a binomial distribution, and $\mathrm{Po}(\beta)$ a random variable with a Poisson distribution. Below we make use of the Poisson approximation of the binomial distribution and the fact that the number of child hyperedges of a node in the hypertree asymptotically follows the binomial distribution. This results in additive terms that tend to zero as m goes to infinity. We have $p_0 = 0$,

$$\begin{aligned} p_1 &= \Pr\left[\mathrm{Bin}\left(\binom{m-1}{k-1}\ ,\ k! \cdot \frac{c}{m^{k-1}}\right) \le \ell - 2\right] \\ &= \Pr[\mathrm{Po}(kc) \le \ell - 2] \pm o(1), \\ p_{j+1} &= \Pr\left[\mathrm{Bin}\left(\binom{m-1}{k-1}\ ,\ k! \cdot \frac{c}{m^{k-1}} \cdot (1-p_j)^{k-1}\right)\ \le \ell - 2\right] \\ &= \Pr[\mathrm{Po}(kc(1-p_j)^{k-1}) \le \ell - 2] \pm o(1),\ \text{for } j = 1, \ldots, h-2. \end{aligned}$$

The probability q_h that a itself is deleted is given by the following different formula:

$$q_h = \Pr[\mathrm{Po}(kc(1-p_{h-1})^{k-1}) \le \ell - 1] \pm o(1). \tag{1}$$

The p_j are monotonically increasing and $0 \le p_j \le 1$, so $p = \lim p_j$ is well-defined. The probability that a is deleted approaches p from below as h grows. Continuity of the functions involved implies that p is the smallest non-negative solution of

$$p = \Pr[\mathrm{Po}(kc(1-p)^{k-1}) \le \ell - 2].$$

Observe that 1 is always a solution. Equivalently, applying the monotone function $t \mapsto kc(1-t)^{k-1}$ to both sides of the equation, p is the smallest solution of

$$kc(1-p)^{k-1} = kc\left(1 - \Pr[\mathrm{Po}(kc(1-p)^{k-1}) \le \ell - 2]\right)^{k-1}. \tag{2}$$

Let $\beta = kc(1-p)^{k-1}$. It is helpful to think of β with the following interpretation:

Given a node in the hypertree, the number of child hyperedges (before deletion) follows the distribution $\mathrm{Po}(kc)$. Asymptotically, a given child hyperedge is not deleted with probability $(1-p)^{k-1}$, independently for all children. Hence the number of child hyperedges after deletion follows the distribution $\mathrm{Po}(kc(1-p)^{k-1})$. And β is the key parameter for the node giving the expected number of hyperedges containing it that could contribute to keeping it in the core.

Note that (2) is equivalent to

$$c = \frac{1}{k} \cdot \frac{\beta}{\left(\Pr[\mathrm{Po}(\beta)\ \ge \ell - 1]\right)^{k-1}}.$$

This motivates considering the function

$$g_{k,\ell}(\beta) = \frac{1}{k} \cdot \frac{\beta}{(\Pr[\mathrm{Po}(\beta) \ge \ell - 1])^{k-1}}, \tag{3}$$

which has the following properties in the range $(0, \infty)$: It tends to infinity for $\beta \to 0$, as well as for $\beta \to \infty$. Since it is convex there is exactly one global minimum. Let $\beta^*_{k,\ell} = \arg\min_\beta g_{k,\ell}(\beta)$ and $c^*_{k,\ell} = \min g_{k,\ell}(\beta)$. For $\beta > \beta^*_{k,\ell}$ the function $g_{k,\ell}$ is monotonically increasing. For each $c > c^*_{k,\ell}$ let $\beta(c) = \beta_{k,\ell}(c)$ denote the unique $\beta > \beta^*_{k,\ell}$ such that $g_{k,\ell}(\beta) = c$.

Coming back to the fate of a under the peeling process, Equation (1) shows that a is deleted with probability approaching $\Pr[\mathrm{Po}(\beta(c)) \le \ell - 1]$. This probability is smaller than 1 if and only if $c > c^*_{k,\ell}$, which implies that the expected number of nodes that are *not* deleted is linear in n. As the h-neighborhoods of two nodes a and b are disjoint with high probability, by making use of the second moment we can show that in this case a linear number of nodes survive with high probability. (The sophisticated reader would use Azuma's inequality to obtain concentration bounds.)

Following this line of reasoning, we obtain the following results, the full proof of which is in [25]. (See also the related argument of [23, Ch. 18].) Note the restriction to the case $k + \ell > 4$, which means that the result does not apply to 2-cores in standard graphs; since the analysis of standard cuckoo hashing is simple, using direct arguments, this case is ignored in the analysis henceforth.

Proposition 1. *Let $k + \ell > 4$ and G be a random hypergraph from $\mathcal{G}^k_{m,n}$. Then $c^*_{k,\ell}$ is the threshold for the* appearance *of an ℓ-core in G. That is, for constant c and $m \to \infty$,*

(a) *if $n/m = c < c^*_{k,\ell}$, then G has an empty ℓ-core with probability $1 - o(1)$.*
(b) *if $n/m = c > c^*_{k,\ell}$, then G has an ℓ-core of linear size with probability $1 - o(1)$.*

In the following we assume $c > c^*_{k,\ell}$. Therefore $\beta(c) > \beta^*_{k,\ell}$ exists. Let $\hat{m}$ be the number of nodes in the ℓ-core and $\hat{n}$ be the number of hyperedges in the ℓ-core. We will find it useful in what follows to consider the *edge density* of the ℓ-core, which is simply the ratio of the number of hyperedges to the number of nodes.

Proposition 2. *Let $c > c^*_{k,\ell}$ and $n/m = c\,(1 \pm o(1))$. Then with high probability in $\mathcal{G}^k_{m,n}$*

$$\hat{m} = \Pr[\mathrm{Po}(\beta(c)) \ge \ell] \cdot m \pm o(m) \text{ and } \hat{n} = (\Pr[\mathrm{Po}(\beta(c)) \ge \ell - 1])^k \cdot n \pm o(m).$$

The bound for $\hat{m}$ follows from the concentration of the expected number of nodes surviving when we plug in the limit p for p_h in equation (1). The result for $\hat{n}$ follows similar lines: Consider a fixed hyperedge e that we assume is present in the random hypergraph. For each node of this hyperedge we consider its h-neighborhood modified in that e itself does not belong to this h-neighborhood. We have k disjoint trees with high probability. Therefore each of the k nodes of e survives h iterations of the peeling procedure independently with probability $\Pr[\mathrm{Po}(\beta(c)) \ge \ell - 1]$. Note that we use $\ell - 1$ here (instead of ℓ) because the nodes belong to e. Then e itself survives with $(\Pr[\mathrm{Po}(\beta(c)) \ge \ell - 1])^k$. Concentration of the number of surviving hyperedges again follows from second moment calculations or Azuma's inequality.

With this we have the information needed regarding the edge density of the ℓ-core.

Proposition 3. *If $c > c^*_{k,\ell}$ and $n/m = c\,(1 \pm o(1))$ then with high probability the edge density of the ℓ-core of a random hypergraph from $\mathcal{G}^k_{m,n}$ is*

$$\frac{\beta(c) \cdot \Pr[\mathrm{Po}(\beta(c)) \ge \ell - 1]}{k \cdot \Pr[\mathrm{Po}(\beta(c)) \ge \ell]} \pm o(1).$$

This follows directly from Proposition 2, where we have also used equation (3) to simplify the expression for $\hat{n}$.

We define $c_{k,\ell}$ as the unique c that satisfies

$$\frac{\beta(c) \cdot \Pr[\mathrm{Po}(\beta(c)) \ge \ell - 1]}{k \cdot \Pr[\mathrm{Po}(\beta(c)) \ge \ell]} = \ell - 1. \tag{4}$$

The values $c_{k,\ell}$ will prove important in the work to come; in particular, we next show that $c_{k,2}$ is the threshold for k-ary cuckoo hashing for $k > 2$. We also conjecture that $c_{k,\ell+1}$ is the threshold for k-ary cuckoo hashing when a bucket can hold ℓ keys instead of a single key.

The following table contains numerical values of $c_{k,\ell}$ for $\ell = 2, \ldots, 7$ and $k = 2, \ldots, 7$ (rounded to 10 decimal places). Some of these numbers are found or referred to in other works, such as [7, Sect. 5], [22, Sect. 4.4], [23, p. 423], [12], and [5].

$\ell \backslash k$	2	3	4	5	6	7
2	–	0.9179352767	0.9767701649	0.9924383913	0.9973795528	0.9990637588
3	1.7940237365	1.9764028279	1.9964829679	1.9994487201	1.9999137473	1.9999866878
4	2.8774628058	2.9918572178	2.9993854302	2.9999554360	2.9999969384	2.9999997987
5	3.9214790971	3.9970126256	3.9998882644	3.9999962949	3.9999998884	3.9999999969
6	4.9477568093	4.9988732941	4.9999793407	4.9999996871	4.9999999959	5.0000000000
7	5.9644362395	5.9995688805	5.9999961417	5.9999999733	5.9999999998	6.0000000000

3 Equality of thresholds for random k-XORSAT and k-ary cuckoo hashing

We now recall the random k-XORSAT problem and describe its relationship to cores of random hypergraphs and cuckoo hashing. The k-XORSAT problem is a variant of the satisfiability problem in which every clause has k literals and the clause is satisfied if the XOR of values of the literals is 1. Equivalently, since XORs correspond to addition modulo 2, and the negation of X_i is just 1 XOR X_i, an instance of the k-XORSAT problem corresponds to a system of linear equations modulo 2, with each equation having k variables (none of which is negated), and randomly chosen right hand sides. (In what follows we simply use the addition operator where it is understood we are working modulo 2 from context.)

For a random k-XORSAT problem, let $\Phi^k_{m,n}$ be the set of all sequences of n linear equations over m variables $x_1, \ldots, x_m$, where an equation is

$$x_{j_1} + \cdots + x_{j_k} = b_j,$$

where $b_j \in \{0, 1\}$ and $\{j_1, \ldots, j_k\}$ is a subset of $\{1, \ldots, m\}$ with k elements. We consider $\Phi^k_{m,n}$ as a probability space with the uniform distribution.

Given a k-XORSAT formula F, it is clear that F is satisfiable if and only if the formula obtained from F by repeatedly deleting variables that occur only once (and equations containing them) is satisfiable. Now consider the k-XORSAT formula as a hypergraph, with nodes representing variables and hyperedges representing equations. (The values b_j of the equations are not represented.) The process of repeatedly deleting all variables that occur only once, and the corresponding equations, is exactly equivalent to the peeling process on the hypergraph. Hence, after the peeling process, we obtain the 2-core of the hypergraph.

This motivates the following definition. Let $\Psi^k_{m,n}$ be the set of all sequences of n equations such that each variable appears at least twice. We consider $\Psi^k_{m,n}$ as a probability space with the uniform distribution.

Recall that if we start with a uniformly chosen random k-XORSAT formula, and perform the peeling process, then conditioned on the remaining number of equations and variables ($\hat{n}$ and $\hat{m}$), we are in fact left with a uniform random formula from $\Psi^k_{\hat{m},\hat{n}}$. Hence, the imperative question is when a random formula from $\Psi^k_{\hat{m},\hat{n}}$ will be satisfiable. In [10], it was shown that this depends entirely on the edge density of the corresponding hypergraph. If the edge density is smaller than 1, so that there are more variables than equations, the formula is likely to be satisfiable, and naturally, if there are more equations than variables, the formula is likely to be unsatisfiable. Specifically, we have the following theorem from [10].

Theorem 1. *Let $k > 2$ be fixed. For $n/m = \gamma$ and $m \to \infty$,*

(a) *if $\gamma > 1$ then a random formula from $\Psi^k_{m,n}$ is unsatisfiable with high probability.*
(b) *if $\gamma < 1$ then a random formula from $\Psi^k_{m,n}$ is satisfiable with high probability.*

The proof of Theorem 1 in Section 3 of [10] uses a first moment method argument for the simple direction (part (a)). Part (b) is significantly more complicated, and is based on the second moment method. Essentially the same problem has also arisen in coding theoretic settings; analysis and techniques can be found in for example [21]. It has been suggested by various readers of earlier drafts of this paper that previous proofs of Theorem 1 have been insufficiently complete, particularly for $k > 3$. We therefore provide a detailed proof in Appendix C for completeness.

We have shown that the edge density is concentrated around a specific value depending on the initial ratio c of hyperedges (equations) to nodes (variables). Let $c_{k,2}$ be the value of c such that the resulting edge density is concentrated around 1. Then Proposition 3 and Theorem 1 together with the preceding consideration implies:

Corollary 1. *Let $k > 2$ and consider $\Phi^k_{m,n}$. The satisfiability threshold with respect to the edge density $c = n/m$ is $c_{k,2}$.*

Again, up to this point, everything we have stated was known from previous work. We now provide the connection to cuckoo hashing, to show that we obtain the same threshold values for the success of cuckoo hashing. That is, we argue the following:

Theorem 2. *For $k > 2$, $c_{k,2}$ is the threshold for k-ary cuckoo hashing to work. That is, and with n keys to be stored and m buckets, with $c = n/m$ fixed and $m \to \infty$,*

(a) *if $c > c_{k,2}$, then k-ary cuckoo hashing does not work with high probability.*
(b) *if $c < c_{k,2}$, then k-ary cuckoo hashing works with high probability.*

Proof : Assume a set of n keys S is given, and for each $x \in S$ a random set $A_x \subseteq \{1, \ldots, m\}$ of size k of possible buckets is chosen.

To prove part (a), note that the sets A_x for $x \in S$ can be represented by a random hypergraph from $\mathcal{G}^k_{m,n}$. If $n/m = c > c_{k,2}$ and $m \to \infty$, then with high probability the edge density in the 2-core is greater than 1. The hyperedges in the 2-core correspond to a set of keys, and the nodes in the 2-core to the buckets available for these keys. Obviously, then, cuckoo hashing does not work.

To prove part (b), consider the case where $n/m = c < c_{k,2}$ and $m \to \infty$. Picking for each x a random $b_x \in \{0, 1\}$, the sets A_x, $x \in S$, induce a random system of equations from $\Phi^k_{m,n}$. Specifically, $A_x = \{j_1, \ldots, j_k\}$ induces the equation $x_{j_1} + \cdots + x_{j_k} = b_x$.

By Corollary 1 a random system of equations from $\Phi^k_{m,n}$ is satisfiable with high probability. This implies that the the matrix M made up from the left-hand sides of these equations consists of linearly independent rows with high probability. This is because a given set of left-hand sides with dependent rows is only satisfiable with probability at most $1/2$ when we pick the b_x at random.

Therefore we have an $n \times n$-submatrix in M with a nonzero determinant. The expansion of the determinant of this submatrix as a sum of products by the Leibniz formula must contain a product with all factors being variables x_{i_j} (as opposed to 0). This product term corresponds to a permutation mapping keys to buckets, showing that cuckoo hashing is indeed possible. □

We make some additional remarks. We note that the idea of using the rank of the key-bucket matrix to obtain lower bounds on the cuckoo hashing threshold is not new either; it appears in [11]. There the authors

use a result bounding the rank by Calkin [6] to obtain a lower bound on the threshold, but this bound is not tight in this context. More details can be found by reviewing [6, Theorem 1.2] and [23, Exercise 18.6]. Also, Batu et al. [3] note that 2-core thresholds provide an upper bound on the threshold for cuckoo hashing, but fail to note the connection to work on the k-XORSAT problems.

4 Non-integer choices

The analysis of k-cores in Section 3 and the correspondence to k-XORSAT problems extends nicely to the setting where the number of choices for a key is not necessarily a fixed number k. This can be naturally accomplished in the following way: when a key x is to be inserted in the cuckoo hash table, the number of choices of location for the key is itself determined by some hash function; then the appropriate number of choices for each key x can also be found when performing a lookup. Hence, it is possible to ask about for example cuckoo hashing with 3.5 choices, by which we would mean an average of 3.5 choices. Similarly, even if we decide to have an average of k choices per key, for an integer k, it is not immediately obvious whether the success probability in k-ary cuckoo hashing could be improved if we do not fix the number of possible positions for a key but rather choose it at random from a cleverly selected distribution.

Let us consider a more general setting where for each $x \in U$ the set A_x is chosen uniformly at random from the set of all k_x-element subsets of $[m]$, where k_x follows some probability mass function ρ_x on $\{2, \dots, m\}$.[1] Let $\kappa_x = E(k_x)$ and $\kappa^* = \frac{1}{n}\sum_{x \in S} \kappa_x$. Note that κ^* is the average (over all $x \in S$) worst case lookup time for successful searches. We keep κ^* fixed and study which sequence $(\rho_x)_{x \in S}$ maximizes the probability that cuckoo hashing is successful.

We fix the sequence of the expected number of choices per key $(\kappa_x)_{x \in S}$ and therefore κ^*. Furthermore we assume $\kappa_x \leq n - 2$, for all $x \in S$; obviously this does not exclude interesting cases. For compactness reasons, there is a system of probability mass functions ρ_x that maximizes the success probability. We will show the following:

Proposition 4. *Let $(\rho_x)_{x \in S}$ be an optimal sequence. Then we have, for all $x \in S$:*

$$\rho_x(\lfloor \kappa_x \rfloor) = 1 - (\kappa_x - \lfloor \kappa_x \rfloor), \text{ and } \rho_x(\lfloor \kappa_x \rfloor + 1) = \kappa_x - \lfloor \kappa_x \rfloor.$$

That is, the success probability is maximized if for each $x \in S$ the number of choices k_x is concentrated on $\lfloor \kappa_x \rfloor$ and $\lfloor \kappa_x \rfloor + 1$ (when the number of choices is non-integral). Further, in the natural case where all keys x have the same expected number κ^* of choices, the optimal assignment is concentrated on $\lfloor \kappa^* \rfloor$ and $\lfloor \kappa^* \rfloor + 1$. Also, if κ_x is an integer, then a fixed degree $k_x = \kappa_x$ is optimal. This is very different from other similar scenarios, such as erasure- and error-correcting codes, where irregular distributions have proven beneficial [20].

The proof is given in Appendix A.

4.1 Thresholds for non-integral degree distributions

We now describe how to extend our previous analysis to derive thresholds for the case of a non-integral number of choices per key; equivalently, we are making use of thresholds for XORSAT problems with an irregular number of literals per clause.

[1] We could in principle also consider the possibility of keys having only a single choice. However, this is generally not very interesting since even a small number of keys with a single choice would make an assignment impossible whp., by the birthday paradox. Hence, we restrict our attention to at least two choices.

Following notation that is frequently used in the coding literature, we let Λ_k be the probability that a key obtains k choices, and define $\Lambda(x) = \sum_k \Lambda_k x^k$. Clearly, then, $\Lambda'(x) = \sum_k \Lambda_k k x^{k-1}$, and $\Lambda'(1) = \kappa^*$. (We assume henceforth that $\Lambda_0 = \Lambda_1 = 0$ and $\Lambda_k = 0$ for all k sufficiently large for technical convenience.)

We now follow our previous analysis from Section 2; to see if a node a is deleted after h rounds of the peeling process, we let p_j be the probability that a node at distance $h - j$ from a is deleted after j rounds. We must now account for the differing degrees of hyperedges. Here, the appropriate asymptotics is given by a mixture of binomial hypergraphs, with each hyperedge of degree k present with probability $k! \cdot c\Lambda_k / m^{k-1}$ independently.

The corresponding equations are then given by $p_0 = 0$,

$$\begin{aligned} p_1 &= \Pr\left[\sum_k \mathrm{Bin}\left(\binom{m-1}{k-1},\ k! \cdot \frac{c\Lambda_k}{m^{k-1}}\right) \le \ell - 2\right] \\ &= \Pr\left[\sum_k \mathrm{Po}(kc\Lambda_k) \le \ell - 2\right] \pm o(1), \\ &= \Pr[\mathrm{Po}(c\Lambda'(1)) \le \ell - 2] \pm o(1), \end{aligned}$$

$$\begin{aligned} p_{j+1} &= \Pr\left[\sum_k \mathrm{Bin}\left(\binom{m-1}{k-1},\ k! \cdot \frac{c\Lambda_k}{m^{k-1}} \cdot (1-p_j)^{k-1}\right) \le \ell - 2\right] \\ &= \Pr\left[\sum_k \mathrm{Po}(kc\Lambda_k(1-p_j)^{k-1}) \le \ell - 2\right] \pm o(1), \text{ for } j = 1, \dots, h-2, \\ &= \Pr[\mathrm{Po}(c\Lambda'(1-p_j)) \le \ell - 2] \pm o(1), \text{ for } j = 1, \dots, h-2. \end{aligned}$$

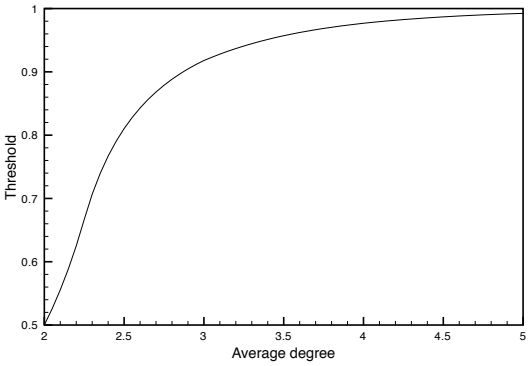
Note that we have used the standard fact that the sum of Poisson random variables is itself Poisson, which allows us to conveniently express everything in terms of the generating function $\Lambda(x)$ and its derivative. As before we find $p = \lim p_j$, which is now given by the smallest non-negative solution of

$$p = \Pr[\mathrm{Po}(c\Lambda'(1-p)) \le \ell - 2].$$

When given a degree distribution $(\Lambda_k)_k$, we can proceed as before to find the threshold load that allows that the edge density of the 2-core remains greater than 1; using the approach of Appendix C, this can again be shown to be the required property for the corresponding XORSAT problem to have a solution, and hence for there to be a permutation successfully mapping keys to buckets. Notice that this argument works for all degree distributions (subject to the restrictions given above), but in particular we have already shown that the optimal thresholds are to be found by the simple degree distributions that have all weight on two values, $\lfloor \kappa^* \rfloor$ and $\lfloor \kappa^* \rfloor + 1$. Abusing notation slightly, let $c_{\kappa^*,2}$ be the unique c such that the edge density of the 2-core of the corresponding mixture is equal to 1, following the same form as in Proposition 3 and equation (4). The corresponding extension to Theorem 2 is the following:

Theorem 3. *For $\kappa^* > 2$, $c_{\kappa^*,2}$ is the threshold for cuckoo hashing with an average of κ^* choices per key to work. That is, with n keys to be stored and m buckets, with $c = n/m$ fixed and $m \to \infty$,*

(a) *if $c > c_{\kappa^*,2}$, for any distribution on the number of choices per key with mean κ^*, cuckoo hashing does not work with high probability.*
(b) *if $c < c_{\kappa^*,2}$, then cuckoo hashing works with high probability when the distribution on the number of choices per key is given by $\rho_x(\lfloor \kappa_x \rfloor) = 1 - (\kappa_x - \lfloor \kappa_x \rfloor)$ and $\rho_x(\lfloor \kappa_x \rfloor + 1) = \kappa_x - \lfloor \kappa_x \rfloor$, for all $x \in S$.*



κ^*	$c_{\kappa^*,2}$
2.25	0.6666666667
2.50	0.8103423635
2.75	0.8788457372
3.00	0.9179352767
3.25	0.9408047937
3.50	0.9570796377
3.75	0.9685811888
4.00	0.9767701649

κ^*	$c_{\kappa^*,2}$
4.25	0.9825693463
4.50	0.9868637629
4.75	0.9900548807
5.00	0.9924383913
5.25	0.9942189481
5.50	0.9955692011
5.75	0.9965961383
6.00	0.9973795528

Fig. 1. Thresholds for non-integral κ^*-ary cuckoo hashing, with optimal degree distribution. The values in the tables are rounded to the nearest multiple of 10^{-10}.

We have determined the thresholds numerically for a range of values of κ^*. The results are shown in Figure 1. One somewhat surprising finding is that the threshold for $\kappa^* \leq 2.25$ appears to simply be given by $c = 0.5/(3 - \kappa^*)$. Consequently, in place of using 2 hash functions per key, simply by using a mix of 2 or 3 hash functions for a key, we can increase the space utilization by adding 33% more keys with the same (asymptotic) amount of memory.

5 Algorithm for computing a placement

In this section, we describe an algorithm for finding a placement for the keys using k-ary cuckoo hashing when the set S of keys is given an advance. The algorithm is an adaptation of the "selfless algorithm" proposed by Sanders [27], for the case $k = 2$, and analyzed in [5], for orienting standard undirected random graphs so that all edges are directed and the maximum indegree of all nodes is at most ℓ, for some fixed $\ell \geq 2$. We generalize this algorithm to hypergraphs, including hypergraphs where hyperedges can have varying degrees.

Of course, maximum matching algorithms can solve this problem perfectly. However, there are multiple motivations for considering our algorithms. First, it seems in preliminary experiments that the running times of standard matching algorithms like the Hopcroft-Karp algorithm [17] will tend to increase significantly as the edge density approaches the threshold (the details of this effect are not yet understood), while our algorithm has linear running time which does not change in the neighborhood of the threshold. This proves useful in our experimental evaluation of thresholds. Second, we believe that algorithms of this form may prove easier to analyze for some variations of the problem.

We first describe the generalized selfless algorithm for bucket size $\ell = 1$. A description in pseudocode is given as Algorithm 1. The algorithm can deal with arbitrary hypergraphs, uniform or not. The aim is to "orient" the hyperedges of the hypergraph G, i. e., associate a node $v \in e$ to each hyperedge e so that at most one hyperedge is directed towards any one node v. Initially, all hyperedges are unoriented. Nodes that have an hyperedge directed towards them are saturated and are not considered further, and similarly hyperedges once oriented are fixed. At each step, if there is a node v that is incident to only one undirected hyperedge e, we direct v to e, breaking ties arbitrarily. (In the pseudocode, this is realized by giving such nodes the highest *priority*, which is 0. Note that this rule entails that the algorithm starts by carrying out the peeling process for the 2-core. But the rule is also applied when hyperedges from the 2-core have already

been treated.) If there are no such nodes, every unoriented hyperedge is assigned as its *weight* the number of unsaturated nodes it contains. (Intuitively, a smaller weight means a higher need to direct the hyperedge.) The priority of a node v then is the sum of the inverses of the weights of the hyperedges that contain v. This corresponds to the expected number of hyperedges v would have directed toward it if all its unoriented hyperedges were directed to one of their nodes at random. Now a vertex v of smallest (highest) priority is chosen, again breaking ties at random. If this priority is larger than 1, then the algorithm stops and reports "failure". This is because the sum of all priorities is the number of undirected hyperedges, so if the smallest priority is bigger than 1, the number of undirected hyperedges is larger than the number of unsaturated nodes, and it is impossible to complete the process of directing the hyperedges. Otherwise the algorithm directs the minimum weight incident hyperedge of v toward v, breaking ties randomly. (Intuitively, this means that the algorithm tries to continue the peeling process "on average".) This step is repeated until all hyperedges have been oriented or failure occurs.

Algorithm 1: $(k, 1)$-Generalized Selfless

Input: Hypergraph $G = (V, E)$ with m nodes and n hyperedges.
Purpose: Direct all hyperedges such that the maximum indegree is at most 1.
`// ` $\mathcal{U}(v)$ `set of undirected hyperedges incident to node` v
for $t \leftarrow 1$ **to** n **do**
  $V_0 \leftarrow \{v \in V : |\mathcal{U}(v)| > 0\}$; $E_0 \leftarrow \{e \in E : e$ undirected$\}$;
  forall the $e \in E_0$ **do** `// calculate edge` *weight* $\omega(e)$
    $\omega(e) \leftarrow |\{v \in e :$ no hyperedge is directed towards $v\}|$;
  forall the $v \in V_0$ **do** `// calculate` *priority* $\pi(v)$
    if *a hyperedge is directed towards* v **then** $\pi(v) \leftarrow 2$ `// saturated` ;
    else
      if $|\mathcal{U}(v)| = 1$ **then** $\pi(v) \leftarrow 0$ **else** $\pi(v) \leftarrow \sum_{e \in \mathcal{U}(v)} \frac{1}{\omega(e)}$;
      ;
  find $v \in V_0$ with smallest priority (break ties by randomization);
  if $\pi(v) > 1$ **then return failure**;
  choose minimum weight hyperedge $e \in \mathcal{U}(v)$ (break ties by randomization);
  direct e towards v

We ran the generalized selfless algorithm for hypergraphs with 10^5 and 10^6 nodes and tabulated the failure rate around the theoretical threshold values $c_{k,2}$ for $k = 3, 4, 5$. Results demonstrate that the generalized selfless algorithm achieves results quite near the threshold; more details and figures are given in Appendix B.

5.1 A conjecture, with evidence from a generalized selfless algorithm

Now consider a situation in which buckets have a capacity of $\ell > 1$ keys. There is as yet no rigorous analysis of the appropriate thresholds for cuckoo hashing for the cases $k > 2$ and $\ell > 1$. However, our results of Section 2 suggest a natural conjecture:

Conjecture 1. For k-ary cuckoo hashing with bucket size ℓ, it is *conjectured* that cuckoo hashing works with high probability if $n/m = c > c_{k,\ell+1}$, and does not work if $n/m = c < c_{k,\ell+1}$, i. e., that the threshold is at the point where the $(\ell + 1)$-core of the cuckoo hypergraph starts having edge density larger than ℓ.

In order to provide evidence for this conjecture, we generalize our algorithm further so that it can deal with bucket size $\ell > 1$. The pseudocode is given as Algorithm 2. In hypergraph language, we are now looking for an orientation of the hyperedges of G so that every node has at most ℓ hyperedges directed toward it. Now a node is saturated if it has ℓ edges pointing to it. As long as there are nodes v such that the number of hyperedges directed toward v and the number of undirected hyperedges containing v taken together does not exceed ℓ, one such node is chosen and its undirected edges are directed toward it. Again, the effect of this rule is that the algorithm starts by carrying out the peeling process that finds the $(\ell + 1)$-core. Otherwise, the algorithm assigned weights and priorities as before, and if all priorities exceed ℓ, the algorithm stops and reports failure. If the smallest (highest) priority is at most ℓ, a vertex of smallest priority is chosen and one of the incident undirected hyperedges of minimum weight is directed toward it. The process is carried out until all hyperedges have been directed or failure occurs.

Algorithm 2: (k, ℓ)-Generalized Selfless

Input: Hypergraph $G = (V, E)$ with m nodes and n hyperedges.
Purpose: Direct all hyperedges such that the maximum indegree is at most ℓ.
// $\mathcal{D}(v)$ set of hyperedges directed towards node v
// $\mathcal{U}(v)$ set of undirected hyperedges incident to node v
for $t \leftarrow 1$ **to** n **do**
  $V_0 \leftarrow \{v \in V : |\mathcal{U}(v)| > 0\}$; $E_0 \leftarrow \{e \in E : e$ undirected$\}$;
  forall the $e \in E_0$ **do** // calculate edge *weight* $\omega(e)$
    $\omega(e) \leftarrow |\{v \in e : |\mathcal{D}(v)| < \ell\}|$;
  forall the $v \in V_0$ **do** // calculate *priority* $\pi(v)$
    if $|\mathcal{U}(v)| + |\mathcal{D}(v)| \leq \ell$ **then** $\pi(v) \leftarrow 0$;
    else $\pi(v) \leftarrow \sum_{e \in \mathcal{U}(v)} \frac{1}{\omega(e)} + |\mathcal{D}(v)|$;
  find $v \in V_0$ with smallest priority (break ties by randomization);
  if $\pi(v) > \ell$ **then return failure**;
  choose minimum weight hyperedge $e \in \mathcal{U}(v)$ (break ties by randomization);
  direct e towards v

Experiments with Algorithm 2 corroborate Conjecture 1, in that they show that the failure rate of the algorithm changes from 0 to 1 very close to the possible threshold values suggested in Conjecture 1. Again, numerical results are given in Appendix B.

6 Conclusion

We have found tight thresholds for cuckoo hashing with 1 key per bucket, by showing that the thresholds are in fact the same for the previous studied k-XORSAT problem. We have generalized the result to irregular cuckoo hashing where keys may have differing numbers of choices, and have conjectured thresholds for the case where buckets have size larger than 1 based on an extrapolation of our results.

References

1. Y. Azar, A. Broder, A. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999.

2. R.N. Bhattacharya and R. Rao, *Normal approximation and asymptotic expansions*, Wiley, New York, (1976).
3. T. Batu, P. Berenbrink, and C. Cooper. Balanced allocations: Balls-into-bins revisited and chains-into-bins. CDAM Research Report LSE-CDAM-2007-34.
4. A. Broder, A. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-CNF formulas. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 322–330, 1993.
5. J. A. Cain, P. Sanders, and N. Wormald. The random graph threshold for k-orientiability and a fast algorithm for optimal multiple-choice allocation. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algrorithms*, pp. 469–476, 2007.
6. N. Calkin. Dependent Sets of Constant Weight Binary Vectors. *Combinatorics, Probability, and Computing*, 6(3):263-271, 1997.
7. C. Cooper. The size of the cores of a random graph with a given degree sequence. *Random Structures and Algorithms*, 25(4):353–375, 2004.
8. N. Creignou and H. Daudé. Smooth and sharp thresholds for random k-XOR-CNF satisfiability. *Theoretical Informatics and Applications*, 37(2):127–147, 2003.
9. N. Creignou and H. Daudé. The SAT-UNSAT transition for random constraint satisfaction problems. *Discrete Mathematics*, 309, No 8 (2009), 2085-2099.
10. O. Dubois and J. Mandler. The 3-XORSAT threshold, In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, pp. 769–778, 2002.
11. M. Dietzfelbinger and R. Pagh. Succinct data structures for retrieval and approximate membership. In *Proceedings of the 35th ICALP*, pp. 385–396, 2008.
12. D. Fernholz and V. Ramachandran. The k-orientability thresholds for $G_{n,p}$. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algrorithms*, pp. 459–468, 2007.
13. N. Fountoulakis and K. Panagiotou. Sharp load thresholds for cuckoo hashing. *CoRR*, abs/0910.5147, 2009. Submitted on 27 Oct 2009.
14. D. Fotakis, R. Pagh, P. Sanders, and P. Spirakis. Space efficient hash tables with worst case constant access time. *Theory of Computing Systems*, 38(2):229–248, 2005.
15. A. M. Frieze and P. Melsted. Maximum matchings in random bipartite graphs and the space utilization of cuckoo hashtables. *CoRR*, abs/0910.5535, 2009. Submitted on 29 Oct 2009 (v1), revised 11 Nov 2009 (v2).
16. Brian Gough. *GNU Scientific Library Reference Manual - Third Edition*. Network Theory Ltd., 2009. online: http://www.gnu.org/software/xGSl/manual/.
17. J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.
18. E. Lehman and R. Panigrahy. 3.5-Way cuckoo hashing for the price of 2-and-a-bit. In *Proceedings of the 17th Annual European Symposium on Algorithms*, pp. 671–681, 2009.
19. M. Luby, M. Mitzenmacher, and M.A. Shokrollahi. Analysis of random processes via and-or tree evaluation. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 364–373, 1998.
20. M. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D. Spielman. Efficient erasure correcting codes. *IEEE Transactions on Information Theory*, 47(2):569–584, 2001.
21. C. Méasson, A. Montanari, and R. Urbanke. Maxwell construction: the hidden bridge between iterative and maximum a posteriori decoding. *IEEE Transactions on Information Theory*, 54(12):5277–5307, 2008.
22. M. Mézard, F. Ricci-Tersenghi, and R. Zecchina. Two solutions to diluted p-spin models and XORSAT problems. *J. Statist. Phys.* 111(3/4): 505–533, 2003.
23. M. Mézard and A. Montanari. Information, Physics, and Computation. Oxford University Press, 2009.
24. M. Mitzenmacher. Some open questions related to cuckoo hashing. In *Proceedings of the 17th Annual European Symposium on Algorithms*, pp. 1–10, 2009.
25. M. Molloy. Cores in random hypergraphs and Boolean formulas. *Random Structures and Algorithms*, 27(1):124–135, 2005.
26. A. Pagh and F. Rodler. Cuckoo hashing. *Journal of Algorithms*, 51(2):122–144, 2004.
27. P. Sanders. Algorithms for Scalable Storage Servers. In *Proceedings of SOFSEM 2004*, pp. 82–101, 2004.

A Optimality of degree distribution

We present here the proof of Proposition 4. Specifically, we show that if $(\rho_x)_{x\in S}$ is an optimal sequence, then for all $x \in S$:

$$\rho_x(\lfloor\kappa_x\rfloor) = 1 - (\kappa_x - \lfloor\kappa_x\rfloor), \text{ and } \rho_x(\lfloor\kappa_x\rfloor + 1) = \kappa_x - \lfloor\kappa_x\rfloor.$$

Proof : We consider a random bipartite graph G_S with left node set S, right node set $[m]$ and an edge between two nodes $x \in S$ and $a \in [m]$ if and only if $a \in A_x$. Let the sequence $(\kappa_x)_{x\in S}$ be fixed. For each $x \in S$ we want to obtain a distribution ρ_x for the degree k_x (or, equivalently, the cardinality of A_x), such that we have $E(k_x) = \kappa_x$ and the following quantity is maximized:

$$\Pr(\text{“success”}) := \Pr((A_x)_{x\in S} \text{ admits a left-perfect matching}^2 \text{ in } G_S)\ . \tag{5}$$

We study the sequence $(\rho_x)_{x\in S}$ that realizes the maximum. Let z be an arbitrary but fixed element of S with probability mass function ρ_z. To prove Proposition 4 it is sufficient to show that if there exist two numbers l and k with $l < \kappa_z < k$ and $k - l \geq 2$ as well as $\rho_z(l) > 0$ and $\rho_z(k) > 0$ then (5) cannot be maximal.

We start by fixing k_x and A_x for each $x \in S - \{z\}$ and consider the corresponding bipartite graph $G_{S-\{z\}}$. Let $B \subseteq [m]$ be the set of right nodes in $G_{S-\{z\}}$ that are matched in every matching. Then there is a matching for the whole key set S in G_S if and only if $A_z \not\subseteq B$. Note that $0 \leq |B| < m$, i. e., there must be at least one right node that is not matched. Let $p = \min\{\rho_z(l), \rho_z(k)\} > 0$ and $|B| = b$. We will show that changing ρ_z to

$$\begin{aligned} \rho'_z(l) &:= \rho_z(l) - p & \rho'_z(k) &:= \rho_z(k) - p \\ \rho'_z(l+1) &:= \rho_z(l+1) + p & \rho'_z(k-1) &:= \rho_z(k-1) + p, \end{aligned}$$

with $\rho'_z(j) = \rho_z(j)$ for $j \notin \{l, k\}$, increases (5), while leaving κ_z unchanged. This is the case if and only if

$$p \cdot \frac{\binom{b}{l}}{\binom{m}{l}} + p \cdot \frac{\binom{b}{k}}{\binom{m}{k}} \geq p \cdot \frac{\binom{b}{l+1}}{\binom{m}{l+1}} + p \cdot \frac{\binom{b}{k-1}}{\binom{m}{k-1}}, \tag{6}$$

and the strict inequality holds for at least one value b that occurs with positive probability. The left sum of (6) is the $2 \cdot p$ fraction of the failure probability (by $\rho_z(l)$ and $\rho_z(k)$) before the change of ρ_z under the condition that B has cardinality b; the right sum is the corresponding fraction of the failure probability after the change. Depending on b we have to distinguish several cases.

Case 1: $b = m-1$. In this case both sides of (6) are equal, i. e., the modification we do to ρ_z will not change the success probability.

Case 2: $k \leq b < m-1$. Canceling p and subtracting $\binom{b}{k}/\binom{m}{k}$ and $\binom{b}{l+1}/\binom{m}{l+1}$ from both sides of (6) shows that the strict inequality holds if and only if

$$\frac{b\cdots(b-l+1)}{m\cdots(m-l+1)} - \frac{b\cdots(b-l)}{m\cdots(m-l)} > \frac{b\cdots(b-k+2)}{m\cdots(m-k+2)} - \frac{b\cdots(b-k+1)}{m\cdots(m-k+1)}\ . \tag{7}$$

Factoring out $\frac{b\cdots(b-l+1)}{m\cdots(m-l+1)}$ on the left side and $\frac{b\cdots(b-k+2)}{m\cdots(m-k+2)}$ on the right side gives

$$\frac{b\cdots(b-l+1)}{m\cdots(m-l)} > \frac{b\cdots(b-k+2)}{m\cdots(m-k+1)} \Leftrightarrow \frac{m\cdots(m-k+1)}{m\cdots(m-l)} > \frac{b\cdots(b-k+2)}{b\cdots(b-l+1)}\ . \tag{8}$$

[2] In the following “matching” and “left-perfect matching” are used synonymously.

Since $l \leq k-2$, this is equivalent to

$$(m-l+1)\cdot(m-l)\cdots(m-k+1) > (b-l)\cdot(b-l-1)\cdots(b-k+2)\,, \tag{9}$$

which is true for $m-1 > b$.

Case 3: $l \leq b < k$. Calculations along the lines of case 2 show that the strict inequality of (6) also holds in this case. Note that $\binom{b}{k}$, $\binom{b}{k-1}$ and $\binom{b}{l+1}$ can be zero.

Case 4: $0 \leq b < l$. In this case both sides of (6) are zero, i. e., the modifications we do to ρ_z will not change the success probability.

Since in cases 1 and 4 above there was no change in the success probability, to show that (5) cannot be maximal when $k-l \geq 2$ as we are considering, it remains to show that at least one of the Cases 2 and 3 occurs with positive probability. We construct a situation in which one of these cases applies, and which occurs with positive probability.

Choose degrees k_x for all elements $x \in S-\{z\}$ such that $k_x \leq \kappa_x$ and $\rho_x(k_x) > 0$. Consider a permutation of the elements $x \in S-\{z\}$ such that these degrees are ordered, i. e., $k_{x_1} \leq k_{x_2} \leq \ldots \leq k_{x_{n-1}}$. Choose the first element x_i with $i \geq l$ and $k_{x_i} \leq i$. Such an element must exist, since we assume $k_x \leq \kappa_x \leq n-2$, in particular we have $l < n-2$. Arrange that $A_{x_j} \subseteq [i], 1 \leq j \leq i$, such that there is a matching in $G_{\{x_1,\ldots,x_i\}}$. This implies $b \geq l$. Then arrange that $|A_{x_j} - \bigcup_{1\leq j'<j} A_{x_{j'}}| = 1$, for all $i < j \leq n-2$, as well as $|A_{x_{n-1}} - \bigcup_{1\leq j'<n-1} A_{x_{j'}}| = 2$, which implies $b < m-1$. This finishes the proof of Proposition 4. □

B Performance results for the generalized selfless algorithm

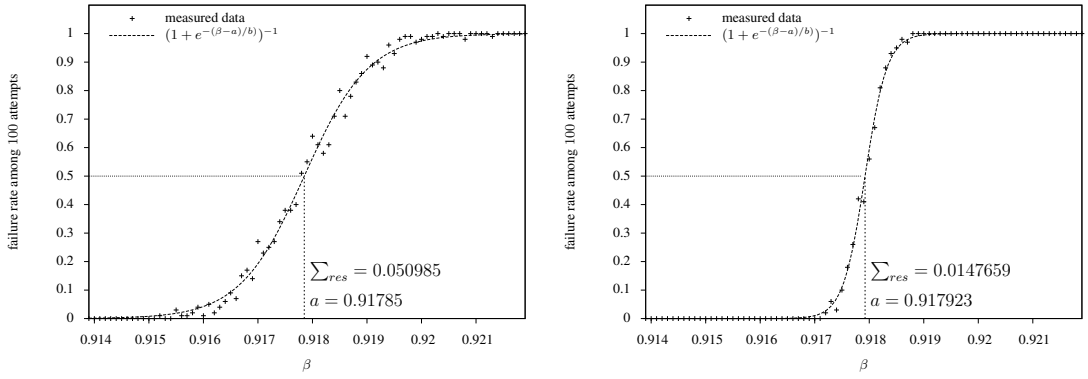
We present some performance results for the generalized selfless algorithm We ran the generalized selfless algorithm for hypergraphs with 10^5 and 10^6 nodes and tabulated the failure rate around the theoretical threshold values $c_{k,2}$ for $k = 3, 4, 5$. For each pair (m, k) we considered 81 edge densities $c = \frac{n}{m}$, spaced apart by 0.0001, thus covering an interval of length 0.008, which encloses the theoretical threshold value for the particular parameter pair (m, k). The hyperedges of the hypergraphs were randomly chosen via pseudo random number generator MT19937 "Mersenne Twister" of the GNU Scientific Library [16]. We measured the average failure rate of the algorithm over 100 random hypergraphs for each combination (m, n, k) within the parameter space. To get an estimation of the threshold, i. e., the rate c where the algorithm switches from success to failure, we fit the sigmoid function

$$\sigma(c; a, b) = \frac{1}{1+e^{-(c-a)/b}} \tag{10}$$

to the measured failure rate (via gnuplot[3]), using the method of least squares. We determined the parameters a, b that lead to a (local) minimum of the sum of squares of the 81 residuals, denoted by $\sum_{res}$. The parameter a is the inflection point of (10) and therefore the approximation of the threshold of the generalized selfless algorithm. Figures 2, 3 and 4 show the results of the experiments.

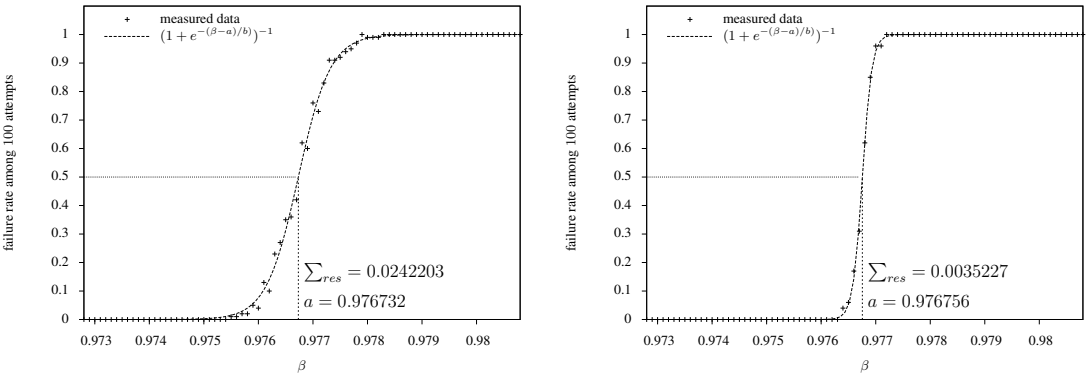
One observes that this simple algorithm is able to construct the placements for edge densities quite close to the calculated thresholds $c_{k,2}$. The slope of the sigmoid curve increases and $\sum_{res}$ decreases with growing m and k, leading to a sharp transition from total success to total failure. Clearly the algorithm can fail on hypergraphs that admit a matching. Experimental comparisons with a perfect matching algorithm [17] showed that this is very unlikely for random hypergraphs. An example is given in Figure 5, which shows

[3] gnuplot, an interactive plotting program, version 4.2, http://www.gnuplot.info



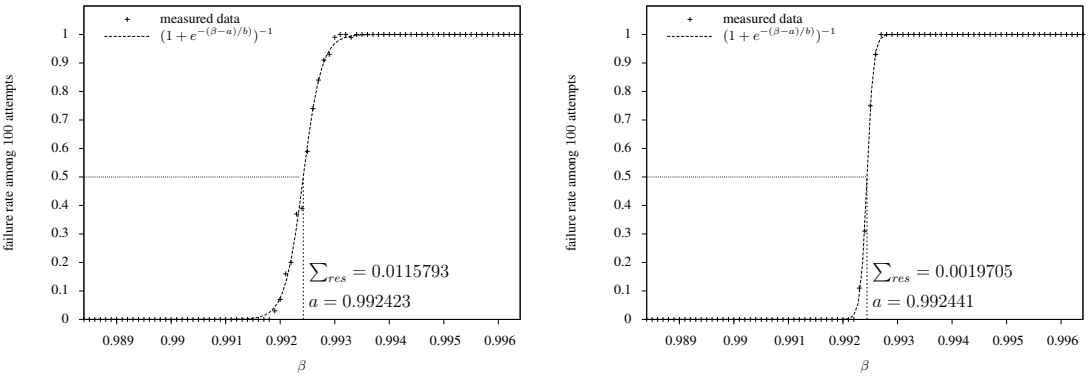
(a) $m = 10^5$ (b) $m = 10^6$

Fig. 2. edge size $k = 3$; theoretical threshold $c_{k,2} \approx 0.91794$



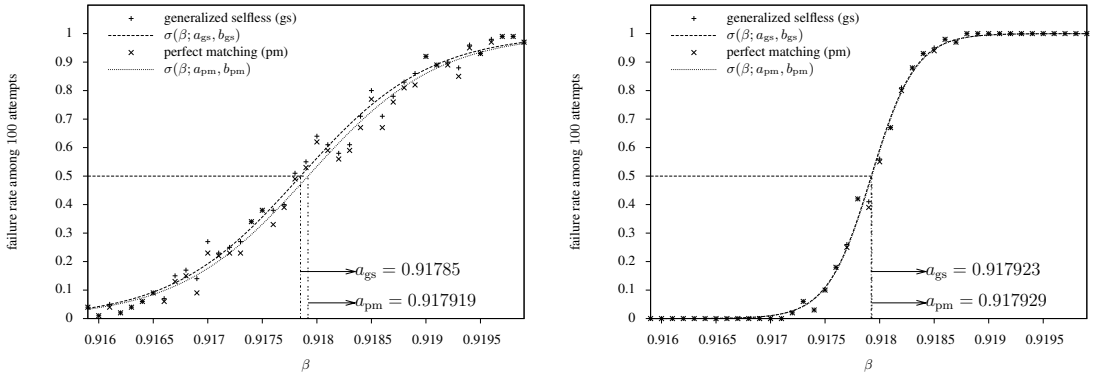
(a) $m = 10^5$ (b) $m = 10^6$

Fig. 3. edge size $k = 4$; theoretical threshold $c_{k,2} \approx 0.97677$



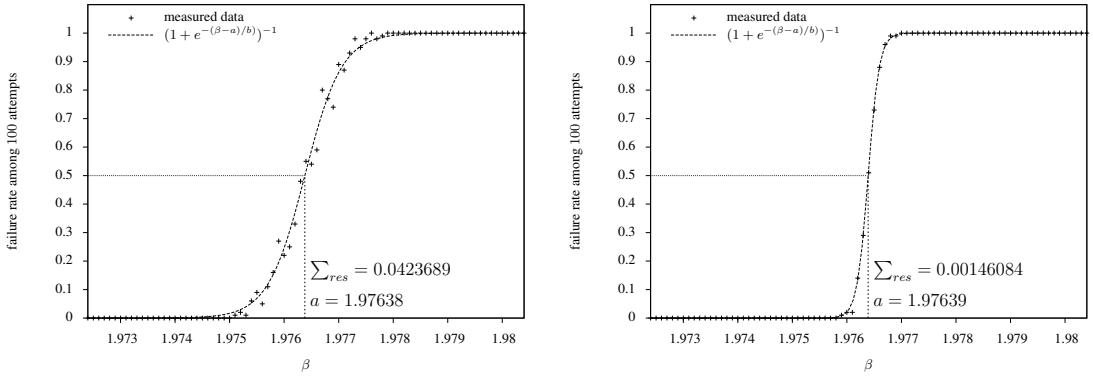
(a) $m = 10^5$ (b) $m = 10^6$

Fig. 4. edge size $k = 5$; theoretical threshold $c_{k,2} \approx 0.99244$



(a) $m = 10^5$ (b) $m = 10^6$

Fig. 5. $k = 3$; theoretical threshold $c_{k,2} \approx 0.91794$, interval size 0.004



(a) $m = 10^5$ (b) $m = 10^6$

Fig. 6. $k = 3$, $\ell = 2$; *conjectured* threshold value $c_{k,\ell+1} \approx 1.97640$

the failure rate of perfect matching in comparison to the generalized selfless algorithm. Note that the plot shows an interval of size 0.004, i. e., 41 data points instead of 81. The differences in the failure rates of the algorithms become very small as m grows.

Similarly, we find our generalized algorithm for the case where the bucket size ℓ is greater than 1 has similar behavior. For an example see Figure 6.

C Proof of the threshold for k-XORSAT

In this section we give a full proof of the threshold for k-XORSAT (Corollary 1). The proof employs the notation and facts developed in Sections 2 and 3, especially Propositions 2 and 3, and the following fact (known as "Friedgut's Theorem" for k-XORSAT [8,9]).

Fact 1. *For every $k \geq 3$ there exists a function $c_k(m) \leq 1$ such that, for every $\varepsilon > 0$ and a random formula F (system $Ax = b$ of equations) from $\Phi^k_{m,n}$ we have the following:*

$$\lim_{m\to\infty} \Pr[F \text{ is satisfiable}] = \begin{cases} 1, \text{ if } n = c_k(m)(1-\varepsilon)m \\ 0, \text{ if } n = c_k(m)(1+\varepsilon)m\,. \end{cases}$$

Recall from Section 3 that $\Phi^k_{m,n}$ can be regarded as a probability space whose elements are pairs (A, b) where A is an $n \times m$ matrix with entries in $\{0, 1\}$, each row containing k 1's, and $b \in \{0, 1\}^n$. Alternatively, A can be regarded as a node-edge incidence matrix A_G of a k-uniform hypergraph $G \in \mathcal{G}^k_{m,n}$. Via the obvious correspondence we identify $\mathcal{G}^k_{m,n}$ with the set of bipartite graphs G with n left nodes ("check nodes") and m right nodes ("variable nodes") and degree k at each left node. Similarly, $\Psi^k_{\hat{m},\hat{n}}$ is the probability space whose elements are pairs $(\hat{A}, \hat{b})$, $\hat{b} \in \{0, 1\}^{\hat{n}}$, where $\hat{A}$ is either the incidence matrix of a k-uniform hypergraph H with $\hat{m}$ nodes, $\hat{n}$ edges, and minimum degree 2 or the adjacency matrix $\hat{A}_H$ of a bipartite graph H with $\hat{n}$ left nodes and $\hat{m}$ right nodes, with degree k at each left node and minimum degree 2 at each right node. We use the same notation for both and let $\mathcal{H}^k_{\hat{m},\hat{n}}$ be the set of all these graphs.[4] The following lemma is central.

Lemma 1. *For any $\delta > 0$ there exists $\varepsilon = \varepsilon(\delta) > 0$ such that the following happens. Let $H \in \mathcal{H}^k_{\hat{m},\hat{n}}$ be uniformly random with $\hat{n} < \hat{m}(1-\delta)$ and denote by Z_H the number of solutions of the linear system $\hat{A}_H x = 0$ (over $\mathrm{GF}[2]$). Then*

$$\Pr[Z_H = 2^{\hat{m}-\hat{n}}] \geq \varepsilon(\delta) > 0\,. \tag{11}$$

We note that a full proof of this lemma for the special case $k = 3$, with $\Pr[Z_H = 2^{\hat{m}-\hat{n}}] = 1 - o(1)$, was given in [10].

Proof of Corollary 1 (assuming Lemma 1)**:** Consider the following two cases.

(i) $c^*_{k,2} < c < c_{k,2}$. Let a system $A_G x = b$ be chosen at random from $\Phi^k_{m,n}$. Reducing G to its 2-core H leads to a system $\hat{A}_H x = \hat{b}$ with $\hat{m}$ variables, $\hat{n}$ equations, and $\mathrm{rank}(\hat{A}_H) = \mathrm{rank}(A_G) - (m - \hat{m})$. The graph H is random in $\mathcal{H}^k_{\hat{m},\hat{n}}$. By Propositions 2 and 3, with high probability $\hat{n} \leq (1-\delta)\hat{m}$ for some $\delta = \delta(c)$, and $\hat{m} = \Theta(m)$. By Lemma 1, for m large enough, we get $\Pr[Z_H = 2^{\hat{m}-\hat{n}}] \geq \varepsilon(\delta) > 0$. This implies $\Pr[A_G x = b \text{ is satisfiable}] \geq \Pr[A_G \text{ has full row rank}] = \Pr[Z_G = 2^{m-n}] \geq \varepsilon(\delta)$.

(ii) $c > c_{k,2}$. Let $A_G x = b$ and its reduced version $\hat{A}_H x = \hat{b}$ be as in (i). By Propositions 2 and 3, with high probability $\hat{n} \geq (1+\delta)\hat{m}$ for some $\delta = \delta(c)$, and $\hat{m} = \Theta(m)$. We have $\mathrm{rank}(\hat{A}) \leq \hat{m}$, and by the randomness of $\hat{b}$ we have $\Pr[\hat{A}_H x = \hat{b} \text{ is satisfiable}] \leq 2^{\hat{m}-\hat{n}} \leq 2^{-\delta\hat{m}}$.

Combining parts (i) and (ii) with Friedgut's Theorem (Fact 1) shows that $\lim_{m\to\infty} c_k(m) = c_{k,2}$, which implies Corollary 1. □

We now move to the proof of Lemma 1, which focuses on the 2-core H of the graph G, and we condition on its number of nodes. With a slight abuse of notation we will drop the "hat" from our notations. In other words, we now let H be a uniformly random graph from $\mathcal{H}^k_{m,n}$ and let $\gamma = n/m$ (see Theorem 1).

[4] For simplicity we assume that for each left node a sequence of k right nodes is chosen at random, allowing and ignoring repetitions. The difference from k-uniform hypergraphs is negligible.

It is convenient to introduce some additional notation. Given a formal series $p(z)$, $\mathrm{coeff}[p(z), z^r]$ denotes the coefficient of z^r in $p(z)$. We further introduce the notations

$$q(z) = (e^z - 1 - z)\,, \qquad Q(z) = \frac{zq'(z)}{q(z)}\,, \tag{12}$$

$$p_k(z) = \frac{1}{2}(1+z)^k + \frac{1}{2}(1-z)^k\,, \qquad P_k(z) = \frac{zp_k'(z)}{p_k(z)}\,. \tag{13}$$

It is easy to see that $z \mapsto Q(z)$ is a strictly increasing function with $\lim_{z\to 0} Q(z) = 2$, and $\lim_{z\to\infty} Q(z) = \infty$. Further $z \mapsto P_k(z)$ is strictly increasing with $\lim_{z\to 0} P_k(z) = 0$, and $\lim_{z\to\infty} P_k(z) = k$ for k even, and $k-1$ otherwise.

Further we define the domain sets

$$\mathcal{D}_{m,n} = \big\{(w,l) \in \mathbb{Z}^2 \,:\, 0 \le w \le m\,, 2w \le l \le kn - 2(m-w)\,, l \text{ even}\,\big\}\,, \tag{14}$$

$$\mathcal{D}_{\gamma}(\varepsilon) = \big\{(\omega,\lambda) \in \mathbb{R}^2 \,:\, \varepsilon \le \omega \le 1-\varepsilon\,, \tfrac{2\omega}{k\gamma} + \varepsilon \le \lambda \le 1 - \tfrac{2(1-\omega)}{k\gamma} - \varepsilon\big\}\,, \tag{15}$$

$$\mathcal{D}_{m,n}(\varepsilon) = \big\{(w,l) \in \mathcal{D}_{m,n} \,:\, (\tfrac{w}{m}, \tfrac{l}{kn}) \in \mathcal{D}_{\gamma}(\varepsilon)\big\}\,, \tag{16}$$

$$\overline{\mathcal{D}}_{m,n}(\varepsilon) = \mathcal{D}_{m,n} \setminus \mathcal{D}_{m,n}(\varepsilon)\,. \tag{17}$$

The assertion of Lemma 1 now follows from the following sequence of lemmas, to be proven in the subsections below.

Lemma 2. *Let Z_H be the number of solutions of the linear system $A_H x = 0$. Then*

$$\mathrm{E}[Z_H] = \frac{1}{N_0} \sum_{(w,l)\in\mathcal{D}_{m,n}} N(w,l)\,, \tag{18}$$

where we define

$$N_0 = (kn)!\ \mathrm{coeff}[(e^z - 1 - z)^m, z^{kn}]\,, \tag{19}$$

$$N(w,l) = \binom{m}{w} l!(kn-l)!\ \mathrm{coeff}[(e^z-1-z)^w, z^l]\ \mathrm{coeff}[(e^z-1-z)^{m-w}, z^{kn-l}]\ \mathrm{coeff}[p_k(z)^n, z^l]\,. \tag{20}$$

Lemma 3. *For any $\delta > 0$ there exists $\varepsilon > 0$ such that, if $n \le m(1-\delta)$, then*

$$\frac{1}{N_0} \cdot \sum_{(w,l)\in\overline{\mathcal{D}}_{m,n}(\varepsilon)} N(w,l) \le 2^{m\delta}\,. \tag{21}$$

Lemma 4. *For any $\delta > 0, \varepsilon > 0$ there exists $C = C(\delta,\varepsilon)$ such that, if $m\delta \le n \le m(1-\delta)$ and $(w,l) \in \mathcal{D}_{m,n}(\varepsilon)$, then*

$$\frac{N(w,l)}{N_0} \le \frac{C}{m}\,\exp\Big(m\,\psi\big(\tfrac{w}{m}, \tfrac{l}{kn}\big)\Big)\,, \tag{22}$$

where, letting $h(z) = -z\log z - (1-z)\log(1-z)$,[5] *we define*

$$\begin{aligned}\psi(\omega,\lambda) = {} & h(\omega) - k\gamma\, h(\lambda) - \log q(s) + k\gamma \log s \\ & + \omega \log q(a) - k\gamma\lambda \log a + (1-\omega)\log q(b) - k\gamma(1-\lambda)\log b \\ & + \gamma \log p_k(c) - k\gamma\lambda \log c\,.\end{aligned} \tag{23}$$

Finally, $a = a(\omega,\lambda)$, $b = b(\omega,\lambda)$, $c = c(\omega,\lambda)$, *and* s *are the unique non-negative solutions of*

$$Q(s) = k\gamma\,, \qquad Q(a) = \frac{k\gamma\lambda}{\omega}\,, \qquad Q(b) = \frac{k\gamma(1-\lambda)}{(1-\omega)}\,, \tag{24}$$

$$P_k(c) = k\lambda\,. \tag{25}$$

Lemma 5. *For any* $\gamma < 1$, *the function* $\psi : \mathcal{D}_\gamma(0) \to \mathbb{R}$ *achieves its unique global maximum at* $(\omega,\lambda) = (1/2, 1/2)$, *with* $\psi(1/2,1/2) = (1-\gamma)\log 2$.

Further, there exists $\xi > 0$ *such that* $-\mathrm{Hess}_\psi(1/2,1/2) \succeq \xi\, I_2$.[6]

Finally, let us recall a well known fact about lattice sums (see for instance [2]).

Lemma 6. *Let* D *be an open domain in* $\mathbb{R}^d$, *and* $F : D \to \mathbb{R}$ *be continuously differentiable, achieving its unique maximum in* $z_* \in D$, *with* $\mathrm{Hess}_F(z_*) \succeq \xi I_d$ *for some* $\xi > 0$. *Then there exists* $C > 0$ *such that, for any* $\delta \geq 0$

$$\sum_{x\in\mathbb{Z}^d\,:\,x\,\delta\in D} \exp\left(\tfrac{1}{\delta}F(\delta x)\right) \leq \frac{C}{\delta^{d/2}}\, \exp\left(\tfrac{1}{\delta}F(z_*)\right). \tag{26}$$

Proof of Lemma 1: The proof is simply obtained by putting together Lemmas 2, 3, 4, 5 and using Lemma 6 (with $F(x_1,x_2) = \psi(x_1, 2x_2/(k\gamma))$, $d = 2$ and $\delta = 1/n$) to bound the sum. □

C.1 Proof of Lemma 2

Clearly N_0 is the number of graphs in $\mathcal{H}^k_{m,n}$. Indeed it is the number of way of putting nk distinct balls in m bins in such a way that each bin contains at least 2 balls.

The claim follows by proving that, for each $(w,l) \in \mathcal{D}_{m,n}$, $N(w,l)$ is the number of couples (H,x) where $H \in \mathcal{H}^k_{m,n}$ and $x \in \{0,1\}^m$ with $A_H x = 0 \mod 2$, such that x has w ones and H has l edges incident on variable (right) nodes i such that $x_i = 1$. Indeed, $\binom{m}{w}$ gives the number of ways of choosing the ones. Paint by red the l edges incident on these nodes, and by blue the other $(kn-l)$ edges. The coefficient factors give the number of ways of attributing red/blue edges to nodes on the two sides. The factorials give the number of ways of matching edges of the same color on the two sides. □

C.2 Proof of Lemma 4

Let us start by proving a lower bound on N_0. For any $s > 0$, we have

$$N_0 = (kn)!\frac{q(s)^m}{s^{kn}}\, \Pr_s\Big[\sum_{i=1}^m X_i = kn\Big]\,, \tag{27}$$

[5] log means logarithm to the base e

[6] Hess_ψ denotes the Hessian matrix of ψ and I_2 the 2×2 unit matrix

where $X_1, \dots, X_m$ are i.i.d. Poisson random variable (with parameter s) conditioned to $X_i \geq 2$, i.e., for any $q \geq 2$,

$$\Pr_s[X_i = q] = \frac{1}{e^s - 1 - s} \frac{s^q}{q!} . \tag{28}$$

By assumption s is chosen such that $\mathrm{E}_s[X_i] = Q(s) = k\gamma \in (k\delta, k(1-\delta))$. By the local central limit theorem for lattice random variables of [2, Corollary 22.3], we have $\Pr_s\left[\sum_{i=1}^m X_i = kn\right] \geq C'/\sqrt{m}$ for sone constant $C'(\delta)$, whence, using Stirling's formula

$$N_0 \geq C_1(\delta) \left(\frac{kn}{e}\right)^{kn} \frac{q(s)^m}{s^{kn}} . \tag{29}$$

Consider now $N(w, l)$. By the central limit theorem for the sum of Bernoulli random variables, for any $m\delta \leq w \leq m(1-\delta)$, we have

$$\binom{m}{w} \leq \frac{C_2(\delta)}{\sqrt{m}} e^{mh(w/m)} . \tag{30}$$

Treating the coefficient terms as above, and using Stirling's formula for l, $(kn - l) = \Theta(m)$, we get

$$N(w, l) \leq \frac{C_3(\delta)}{m} e^{mh(w/m)} \left(\frac{l}{e}\right)^l \left(\frac{knl}{e}\right)^{(kn-l)} \frac{q(a)^w}{a^l} \frac{q(b)^{m-w}}{b^{kn-l}} \frac{p_k(c)^n}{c^l} . \tag{31}$$

The claim is proved by taking the ratio of the bounds (31) and (29). □

C.3 Proof of Lemma 5

For $(\omega, \lambda) = (1/2, 1/2)$, Eqs. (24), (25) admit the unique solution $a = b = s$ and $c = 1$. A straightforward calculation yields $\psi(1/2, 1/2) = (1 - \gamma) \log 2$.

Call $\Psi(\omega, \lambda; a, b, c)$ the right hand side of Eq. (23). Notice that the derivatives of Ψ with respect to a, b, c vanish by Eqs. (24), (25). Therefore it is easy to compute the partial derivatives

$$\frac{\partial \psi}{\partial \omega} = \log \frac{1-\omega}{\omega} + \log \frac{q(a)}{q(b)} , \tag{32}$$

$$\frac{\partial \psi}{\partial \lambda} = -k\gamma \log \frac{1-\lambda}{\lambda} - k\gamma \log \frac{a}{b} - k\gamma \log c . \tag{33}$$

Using the fact that $a = b = s$ and $c = 1$ at $(\omega, \lambda) = (1/2, 1/2)$, we get that the gradient of ψ vanishes at $(1/2, 1/2)$, and again, $\psi(1/2, 1/2) = (1 - \gamma) \log 2$.

By a somewhat longer calculation, we obtain the following second derivatives

$$\left.\frac{\partial^2 \psi}{\partial \omega^2}\right|_{1/2,1/2} = -4\left(1 + \frac{(k\gamma)^2}{s^2 C}\right) , \tag{34}$$

$$\left.\frac{\partial^2 \psi}{\partial \lambda \partial \omega}\right|_{1/2,1/2} = 4\frac{(k\gamma)^2}{s^2 C} , \tag{35}$$

$$\left.\frac{\partial^2 \psi}{\partial \omega^2}\right|_{1/2,1/2} = -4\frac{(k\gamma)^2}{s^2 C} , \tag{36}$$

with

$$C = \frac{q''(s)}{q(s)} - \frac{q'(s)^2}{q(s)^2} + \frac{k\gamma}{s^2} > 0\,. \tag{37}$$

It is easy to deduce that $-\operatorname{Hess}_\psi(1/2, 1/2)$ is positive definite.

The function $\psi : \mathcal{D}_\gamma(0) \to \mathbb{R}$ is continuous in $\mathcal{D}_\gamma(0)$ and differentiable in its interior. Further, we have the following asymptotic behaviors (first two at fixed λ, second two at fixed ω):

$$\lim_{\omega\to 0} \frac{\partial\psi}{\partial\omega} = +\infty\,, \qquad \lim_{\omega\to 1} \frac{\partial\psi}{\partial\omega} = -\infty\,, \tag{38}$$

$$\lim_{\lambda\to 2\omega/(k\gamma)} \frac{\partial\psi}{\partial\lambda} = +\infty\,, \qquad \lim_{\lambda\to 1-2(1-\omega)/(k\gamma)} \frac{\partial\psi}{\partial\lambda} = -\infty\,. \tag{39}$$

Therefore any global maximum of ψ must be a stationary point in the interior of $\mathcal{D}_\gamma(0)$. We next will prove that $(1/2, 1/2)$ is the only such point.

Notice that $\Psi(\omega, \lambda; a, b, c)$ is convex with respect to a, b, c. As a consequence

$$\psi(\omega, \lambda) = \min_{a,b,c} \Psi(\omega, \lambda; a, b, c)\,. \tag{40}$$

We will construct an upper bound on ψ by choosing a, b, c appropriately. The first remark is that

$$\Psi(1-\omega, 1-\lambda; b, a, 1/c) = \Psi(\omega, \lambda; a, b, c) - \gamma\, \log \frac{p_k(c)}{c^k p_k(1/c)}\,. \tag{41}$$

Since, for $c \in [0, 1]$ (which is guaranteed by Eq. (25) for $\lambda \in [0, 1/2]$) we have $p_k(c) \geq c^k p_k(1/c)$, we can restrict without loss of generality to $\lambda \leq 1/2$ (whence $c \in [0, 1]$).

Next notice that, maximizing Ψ over ω, we get $\Psi(\omega, \lambda; a, b, c) \leq \Psi_1(\lambda; a, b, c)$, where

$$\begin{aligned}\Psi_1(\lambda; a, b, c) = &\log\big(q(a) + q(b)\big) - k\gamma\, h(\lambda) - \log q(s) + k\gamma \log s \\ &- k\gamma\lambda \log a - k\gamma(1-\lambda)\log b + \gamma \log p_k(c) - k\gamma\lambda \log c\,.\end{aligned} \tag{42}$$

Next fix $c = c(\lambda) = b\lambda/(a - a\lambda)$. Since this transformation is invertible, we can as well keep c as a free parameter, and let $\lambda = ac/(ac + b)$. If we let $\Psi_2(a, b, c) = \Psi_1(ac/(ac + b); a, b, c)$, we get

$$\begin{aligned}\Psi_2(a, b, c) = &\log\big(q(a) + q(b)\big) - \log q(s) + \gamma \log p_k(c) \\ &-k\gamma \log(ac + b) + k\gamma\lambda \log s\,.\end{aligned} \tag{43}$$

Also, without loss of generality, we can rescale a by a factor s, and set $b = s$, therefore defining $\Psi_3(a, c) = \Psi_2(sa, s, c)$. If we introduce the notation

$$\Lambda_s(x) = \frac{q(sx)}{q(s)} = \frac{e^{sx} - 1 - sx}{e^s - 1 - s}\,, \tag{44}$$

we get the expression

$$\Psi_3(a, c) = -k\gamma \log(1 + ac) + \log\big(1 + \Lambda_s(a)\big) + \gamma \log p_k(c)\,. \tag{45}$$

By the above derivation we have the following relation with $\psi(\omega, \lambda)$:

$$\psi(\omega, \lambda) \le \Psi_4(c)|_{c=\lambda/a_*(c)(1-\lambda)} \,, \tag{46}$$

$$\Psi_4(c) = \Psi_3(a_*(c), c) \,, \qquad a_*(c) = \arg\min_{a \ge 0} \Psi_3(a, c) \,. \tag{47}$$

A direct calculation shows that $a_*(1) = 1$ and $\Psi_3(1, 1) = (1 - \gamma) \log 2$. This point corresponds to $(\omega, \lambda) = (1/2, 1/2)$ through the above derivation. We will show that $c = 1$ is indeed the global maximum of $\Psi_4(c)$ for $c \in [0, 1]$, which implies the assertion.

Maximizing $\Psi_3(a, c)$ with respect to c implies $a_*(c)$ to be the unique non-negative solution of the stationarity condition

$$a = \frac{(1 + c)^{k-1} - (1 - c)^{k-1}}{(1 + c)^{k-1} + (1 - c)^{k-1}} \,. \tag{48}$$

On the other hand, the stationarity condition with respect to a yields

$$c = \frac{\lambda_s(a)}{1 + \Lambda_s(a) - a\lambda_s(a)} \,. \tag{49}$$

where we used the fact that $\Lambda_s'(1) = k\gamma$ and defined $\lambda_s(x) = \Lambda_s'(x)/\Lambda_s'(1)$.

Equations (48) and (49) admit the solutions $a = c = 0$ and $a = c = 1$, and is easy to check that these are both local maxima of Ψ_4. We will show that they admit only one more solution with $c \in (0, 1)$, that necessarily is a local minimum of Ψ_4. Indeed, if we let $a = \tanh x$, $c = \tanh y$, Eq. (48) becomes

$$x = (k - 1)y \,. \tag{50}$$

Our claim is therefore implied by Lemma 7 below. □

Lemma 7. *For $s > 0$, let*

$$\Lambda_s(t) = \frac{e^{st} - 1 - st}{e^s - 1 - s} \,, \qquad \lambda_s(t) = \frac{e^{st} - 1}{e^s - 1} \,. \tag{51}$$

Define $F_s : \mathbb{R} \to \mathbb{R}$ by

$$F_s(x) = \operatorname{atanh}\big(f(\tanh x)\big) \,, \qquad f_s(t) = \frac{\lambda_s(t)}{1 + \Lambda_s(t) - t\lambda_s(t)} \,. \tag{52}$$

Then F_s is convex on $[0, \infty)$.

Proof : This can be seen simply by graphing $F_s(x)$, or by some calculus which we omit. □

C.4 Proof of Lemma 3

The proof is analogous to the one of Lemma 4. We have just to be careful to the values of w, l near the boundary of the domain $\mathcal{D}_{m,n}$. Luckily we only need a loose upper bound. Equation (29) remains true in the present case (as it only hinges on $n = \Theta(m)$). On the other hand using $\binom{m}{w} \le \exp(mh(w/m))$, $\text{coeff}[f(x)^k, x^l] \le f(a)^k/a^l$ and $m! \le \sqrt{2\pi}\,(m/e)^{m+1/2}$, we get

$$N(w, l) \le 2\pi\, e^{-kn-1} e^{mh(w/m)}\, l^{l+1/2} (kn - l)^{(kn-l+1/2)}\, \frac{q(a)^w}{a^l}\, \frac{q(b)^{m-w}}{b^{kn-l}}\, \frac{p_k(c)^n}{c^l} \,. \tag{53}$$

for any $a, b, c > 0$. Taking the ratio, and bounding polynomial factors $\sqrt{l(kn-l)} \leq Cm$ we get

$$\frac{N(w,l)}{N_0} \leq C\, m\, \exp\left(m\psi(w/m, \lambda/kn)\right), \tag{54}$$

whence

$$\frac{1}{N_0} \sum_{(w,l)\in\overline{\mathcal{D}}_{m,n}(\varepsilon)} N(w,l) \leq Cm^3\, \exp\left(m \sup\{\psi(\omega,\lambda) :\, (\omega,\lambda) \in \mathcal{D}_\gamma(0) \setminus \mathcal{D}_\gamma(\varepsilon)\}\right) \tag{55}$$

with $\psi(\omega, \lambda)$ defined as in Eq. (23). Notice that $\psi : \mathcal{D}_\gamma(\delta) \to \mathbb{R}$ is a continuous function. It is therefore sufficient to show that it is strictly smaller than $(1-\gamma)\log 2$ on the boundaries of its domain. This indeed follows from Lemma 5. □