Double Hashing Thresholds via Local Weak Convergence

Mathieu Leconte

Double Hashing Thresholds via Local Weak Convergence

M. Leconte
Technicolor - INRIA
mathieu.leconte@inria.fr

***Abstract*— A lot of interest has recently arisen in the analysis of multiple-choice "cuckoo hashing" schemes. In this context, a main performance criterion is the load threshold under which the hashing scheme is able to build a valid hashtable with high probability in the limit of large systems; various techniques have successfully been used to answer this question (differential equations, combinatorics, cavity method) for increasing levels of generality of the model. However, the hashing scheme analysed so far is quite utopic in that it requires to generate a lot of independent, fully random choices. Schemes with reduced randomness exists, such as "double hashing", which is expected to provide similar asymptotic results as the ideal scheme, yet they have been more resistant to analysis so far. In this paper, we point out that the approach via the cavity method extends quite naturally to the analysis of double hashing and allows to compute the corresponding threshold. The path followed is to show that the graph induced by the double hashing scheme has the same local weak limit as the one obtained with full randomness.**

I. Introduction

The hashing paradigm is as follows: we are given n items and m keys, and we want to assign a key to each item so as to be able to retrieve the items efficiently. In the most basic setting, we want to have exactly one key per item and at most one item per key. Critical performance metrics for such systems are the size of the hashtable (the number of keys) needed to hold a given number of items, and the time it takes to either retrieve or insert items. The *multiple-choice* hashing strategy is one that guaranties a constant look-up time. It consists in pre-defining a set of $d \leq m$ keys for each item, which is then only allowed to pick a key within that set. Of course, depending on the choices of the pre-defined sets of keys, it may be impossible to handle simultaneously some sets of items and inserting a new item may not be easy. As for the second issue, *cuckoo hashing* [1] is a simple, randomized way to search for a new valid assignement upon arrival of a new item: one first checks whether one of the keys pre-defined for the new item is available, in which case it suffices to pick one of these; otherwise, one of the pre-defined keys is chosen at random and re-allocated to the new item. The same procedure is then used for the item that has just been evicted (which is thus treated as a new item). In a fully random context, multiple-choice hashing has been shown to have good performance: if the set of d keys pre-defined for each item are chosen independently and uniformly at random among all sets of d keys, then there exists a threshold τ^* such that, if $n = \tau m$ with $\tau < \tau^*$, in the limit of $m, n \to \infty$ cukoo hashing will yield a valid hashtable with probability tending to 1 (which we refer to as w.h.p. for with high probability). Furthermore, the cuckoo hashing update procedure is also quite efficient [2], [3], in that the insertion time is polylogarithmic in the system size w.h.p. A recent refinement [4] of the update algorithm also achieves constant average insertion time w.h.p. when items are only inserted. The focus here being on the space utilization rather than on the insertion time, i.e. on the efficiency of multiple-choice hashing rather than on that of cuckoo hashing, we do not expand on this.

A major drawback of the fully random (multiple-choice) hashing scheme described above is the amount of randomness involved: the standard approach requires n independent, uniform choices of sets of d keys among m. Phrased differently, this means roughly $d \log m$ independent random bits per item. Generating unbiased, perfectly independent random bits does not come for free [5], [6], therefore a lot of effort has been put into reducing this amount of randomness needed (see [7] and references therein for an account of some directions investigated so far). A candidate alternative is *double hashing* [8], [9], which seems to have similar performances as the fully random one while requiring only roughly $2 \log m$ independent random bits per item: assume m is a prime number and label the m keys with the integers from 1 to m; for each item, independently draw two random numbers $f \in \{1, \dots, m\}$ and $g \in \{1, \dots, \frac{m-1}{2}\}$; the pre-defined set of keys associated with a couple (f, g) are the keys labeled $f + ig$ (mod m), for $i \in \{0, \dots d-1\}$. Although the reduced randomness of the double hashing scheme is the very reason this method could be prefered over fully random hashing, it also makes the theoretical analyzis of its performance more difficult. In this paper, we show that the load threshold under which double hashing succeeds w.h.p. is essentially the threshold τ^* of standard fully random multiple-choice hashing, thereby confirming the hypothesis that double hashing has very similar performance to fully random hashing. More specifically, we show that, if $n = \tau m$ with $\tau < \tau^*$, in the limit of $m, n \to \infty$ double hashing is able to handle all but $o(n)$ items w.h.p. In the case of fully random hashing, one would be able to state a similar result with $o(1)$ instead of $o(n)$. It is very likely that we could obtain the same result using an additional argument (as in [10], [11]), however we believe the current result is sufficient to demonstrate the efficiency of the approach and we do not cross this last gap. It is also of interest to note that there are many extensions of the basic setting considered in this paper (one key per item and at most one item per key). The approach used in this paper is also suited for the analysis of those extensions. For the sake of clarity we do not address them here, although the results go through with only minor modifications.

The remainder of this paper is structured as follows: in the next section, we point out some relevant previous work on the subject. In section III, we define the graphs and random graphs used to model the hashing techniques considered in this paper; in section IV, we recall the concept of local weak limit of random graphs, which is the basis of the cavity method approach for computing the load threshold τ^*. In section V, we show that the local weak limit of the random graphs induced by double hashing is identical to that of the random graphs induced by fully random hashing, which implies that the two methods have essentially the same load threshold.

II. Previous Work on Performance of Hashing Schemes for Load Balancing

The load threshold τ^* under which fully random hashing works w.h.p. can be computed by various techniques [12], [13], [14]. For $d \geq 3$, it is given by

$$\tau^* = \frac{\xi^*}{d(1-e^{-\xi^*})^{d-1}},$$

where ξ^* is the unique solution of the equation

$$d = \frac{\xi(1-e^{-\xi})}{1-e^{-\xi}-\xi e^{-\xi}}.$$

In the basic setting (one key per item and at most one item per key) as well as in extensions (more items allowed per key [15], [16], [17], more keys required per item [18], [10], restrictions on the number of times a couple (item,key) is used [11]), the load threshold τ^* is related to the emergence of a certain core of the bipartite graph with a large density. It can also be viewed as a problem of orientability of hypergraphs. In all the cases, the expression of the load threshold τ^* involves a solution of a fixed-point equation which can be traced back ot the cavity method, as shown in [19], [10], [11] as they are related to the fixed-point messages of the belief propagation message passing algorithm.

More generally than hashing systems, this work is related to load balancing. A common formulation for the load balancing problem is that we have m bins, in which we are going to throw n balls. If we throw each balls uniformly at random independently of the previous ones, we end up with the most loaded bin holding $\frac{\log n}{\log\log n}(1+o(1))$ balls. However, if before throwing each ball we check d bins at random and throw the ball in the least loaded bin, then the most loaded bin will only hold $\frac{\log\log n}{\log d} + O(1)$ balls [20], which represents an exponential improvement. In the same setup, where bins have no limit capacity, double hashing achieves the same performances [21], [7] as fully random choices. However, in order to guaranty a worst-case constant look-up time, one cannot allow a key to refer to an unbounded number of items, which motivates the hashing setup presented in Section I.

III. Hashing Graph and Random Graph Models

A multiple-choice hashing system can be represented as a bipartite graph $G = (L \cup R, E)$ that we call the *hashing graph*, where the left part L represents the items and the right part R the keys; the edges of the graph indicate which keys can be assigned to which items. In the case of fully random hashing, the d edges adjacent to a left-vertex can be anything, while for double hashing they are restricted to having a certain structure. Given such a hashing graph G, it is possible to build a valid hashtable if we can associate each left-vertex (items) to a right-vertex (keys) without collision, i.e. if there exists a left-perfect matching of G. The items with which the hashing system is presented are random and thus the hashing graph is random, following a different distribution whether we consider fully random or double hashing. Thus, the goal is to determine whether the random graphs obtained have a left-perfect matching w.h.p. Note that for fully random hashing, the graph G is simply a uniform random graph with fixed degree d on the left.

For our analysis of double hashing, it is convenient to see its hashing graph as obtained in the following manner, where we merge all the items which are given the same choices of keys and keep track of the multiplicity of the vertices: we start from the bipartite graph $\widetilde{G} = (\widetilde{L} \cup R, E)$ which contains all the information about the choices of keys offered to every item that can potentially be inserted in the hashtable. We have $|R| = m$ and $|\widetilde{L}| = \binom{m}{2}$, with a left-vertex in $\widetilde{G}$ for each different neighborhood (different choice of keys) allowed by the local structure constraints of the hashing technique. If we label the vertices in R with the integers from 1 to m and the vertices in $\widetilde{L}$ with the couples (f,g) with $f \in \{1,\dots,m\}$ and $g \in \{1,\dots,\frac{m-1}{2}\}$, there is an edge in $\widetilde{G}$ between vertex $r \in R$ and $l = (f,g) \in \widetilde{L}$ if there exists $i \in \{0,\dots,d-1\}$ such that $r = f + ig \pmod{m}$. To build the hashing graph, f and g are drawn independently at random for each of the n items, which corresponds exactly to a sampling *with replacement* of n left-vertices in $\widetilde{L}$. We denote by $L' \subset \widetilde{L}$ the set of left-vertices obtained in this way, and by $Z'_l \sim \text{Bin}(n, 1/\binom{m}{2})$ the multiplicity of the left-vertices l. The induced graph $G' = \widetilde{G}[L' \cup R]$ is exactly the graph obtained via double-hashing. This main result is stated in Corollary 1.

As we are interested in the regime $n = \tau m \to \infty$, it is equivalent for our problem to suppose $n \sim \text{Poi}(\tau m)$ instead. Indeed, the Poi(τm) random variable is concentrated around τm with only logarithmic fluctuations for $m \to \infty$, therefore the existence and the value of a load threshold τ^* are identical in the two models. Thus, we consider instead the set L obtained by taking $Z_l \sim \text{Poi}(\frac{2\tau}{m-1})$ copies of each vertex in $\widetilde{L}$, independently for each vertex in $\widetilde{L}$, and keeping in L only those vertices with at least one copy, as we did before for L'. We will focus mainly on the random graph $G = \widetilde{G}[L \cup R]$; we denote by G_m a sample of this random graph for a particular value of m (τ being fixed and $n \sim \text{Poi}(\tau m)$).

IV. Local Weak Convergence and the Cavity Method

As we mentioned earlier, we intend to show that the approach via the cavity method for computing the value of the load threshold τ^* can be extended to double hashing. We will not explain here how or why the cavity method works; the interested reader can refer to the vast literature on that particular subject. However, we need to explain the notion of local weak convergence of a sequence of graphs, which is used in this paper and is at the basis of the cavity approach. The framework used here is that of [22].

First, we recall that a sequence of probability measures $(\rho_m)_{m\in\mathbb{N}}$ on a certain metric space S converges weakly to a probability measure ρ, which we denote by $\rho_m \rightsquigarrow \rho$, if for every bounded continuous function $f : S \to \mathbb{R}$, $\int f d\rho_m$ converges to $\int f d\rho$. We focus on locally finite rooted graphs (or rooted networks, if we want to include marks on the edges). A rooted graph (G, v) is a graph G together with a distinguished vertex v of G which is called the root. A rooted isomorphism of rooted graphs is a graph isomorphism which sends the root of one to the root of the other, i.e. a one-to-one mapping of the vertices of a graph to those of another graph which preserves the root, the edges (and their marks). We denote by $[G, v]$ the isomorphism class of (G, v) and by $\mathcal{G}_*$ the set of rooted isomorphism classes of rooted locally finite graphs. We endow $\mathcal{G}_*$ with the following metric: let the distance between $[G, v]$ and $[G', v']$ be $1/(1+\delta)$, where δ is the supremum of those $k \in \mathbb{N}$ such that there is some rooted isomorphism of the balls of graph-distance radius k around the roots of G and G'. With this metric on $\mathcal{G}_*$, the definition of weak convergence applies to probability measures on $\mathcal{G}_*$. Given a finite graph G, there is a natural way to form a probability measure on $\mathcal{G}_*$: we let $U(G)$ be the distribution over $\mathcal{G}_*$ obtained by choosing a uniform random vertex of G as a root. Then, we say a sequence $(G_m)_{m\in\mathbb{N}}$ converges locally weakly towards a measure ρ on $\mathcal{G}_*$ when the sequence of distributions $(U(G_m))_{m\in\mathbb{N}}$ converges weakly towards ρ.

In the case of fully random hashing and with $n = \tau m$, the hashing graph is a uniform random graph with fixed degree d on the left. As $m \to \infty$, such a random graph, when rooted at a random right-vertex, is known to converge locally weakly to a two-step Galton-Watson tree with distributions Poi(τd) and Deterministic(d). A sample of such a (potentially infinite) tree is constructed as follows: the root (a key) has degree Poi(τ); every vertex at odd distance from the root (items) has exactly $d - 1$ children; every vertex at even distance from the root (keys) has a Poi(τd) number of children.

We should now explain how this notion of local weak convergence is used to obtain the load threshold τ^*. We noted previously that the existence of a valid hashtable is equivalent to that of a left-perfect matching of the hashing graph. Clearly, such a matching is of maximum size among all matchings and its size is exactly equal to n. In [19], [10], [11], the cavity method was used to compute the asymptotic density of a maximum-size matching for any sequence of random graphs converging in the local weak sense to a Galton-Watson tree. It is then possible to determine the threshold $\widetilde{\tau}^*$ under which a maximum-sized matching has size equivalent to $n = \tau m$ as $m \to \infty$. Having $\tau < \widetilde{\tau}^*$ thus guaranties the existence of a matching of size $n - o(n)$ as $m \to \infty$; therefore all but at most $o(n)$ items can be handled. In a second shorter step, it was shown that the two thresholds τ^* and $\widetilde{\tau}^*$ are actually equal for fully random hashing.

The advantage of the approach presented in this section is that the threshold $\widetilde{\tau}^*$ is already computed for all graphs converging in the local weak sense towards two-step Galton-Watson trees. Therefore, to compute the load threshold for another hashing scheme, one only needs to perform two smaller steps: (1) show the appropriate local weak convergence of the hashing graphs, (2) check the equality of the two threshold $\widetilde{\tau}^*$ and τ^* for the particular hashing scheme used. In the next section, we stop before this second step and show that the thresholds $\widetilde{\tau}^*$ are identical for the fully random hashing and double hashing, as both random graph models have the same two-step Galton-Watson tree as a local weak limit.

V. Local Weak Convergence of Double Hashing Graph

A. Main Results and Overview of the Proof

The proof consists in similar steps as in [23], with the difference that in our case the graphs are bipartite and have a more constrained local structure.

Let us call $G_m = (L_m \cup R_m, E_m)$ the random graph obtained at finite m and let ρ_{G_m} be the distribution of $[G_m, r_0]$, where r_0 is a random root in R_m; let $\overline{\rho}_m$ be the average of ρ_{G_m} over the random graph G_m. Let also ρ be the law of a two-step Galton-Watson tree with laws Poi(τd) and Deterministic(d), as defined in the previous section. In a first step, we will show that $\overline{\rho}_m$ converges weakly towards ρ as $m \to \infty$. Then, we will show that ρ_{G_m} is concentrated around $\overline{\rho}_m$ so that ρ_{G_m} almost surely converges weakly towards ρ. Explained differently, there are two sources of randomness involved in the sampling of a rooted graph $[G_m, r_0]$: the first one is artificial and is due to r_0 being a random right-vertex. It has been introduced to turn any fixed graph into a distribution over rooted graphs, which then allows to consider weak convergence of a sequence of such distributions. The second one is inherent to the hashing scheme used: it comes from the fact the couple (f, g) determining the choice of keys offered to each item is chosen at random upon arrival of the item, and it turns G_m itself into a random graph, in a way described in Section III. We first show that the measure ρ_m obtained by averaging over both sources of randomness at the same time converges weakly towards the law ρ of a two-step Galton-Watson, before showing that the averaging over the random choices offered to the items is not required for the convergence to hold, which means we do not need to average over different realization of a large hashtable for the results to hold.

We will thus show the following results: the first one is the intermediate result where we show local weak convergence with both types of averaging together.

Proposition 1: The average distribution $\overline{\rho}_m$ of the double hashing random rooted graph $[G_m, r_0]$ rooted at a random vertex $r_0 \in R_m$ converges weakly towards the distribution of a two-step Galton-Watson tree. In other words, we have

$$\overline{\rho}_m \rightsquigarrow \rho.$$

The result above, together with some results on concentration of measure, leads to our main theorem, which is almost sure local weak convergence of the hashing graph:

Theorem 1: The distribution ρ_{G_m} of the double hashing random graph G_m almost surely converges weakly towards the distribution of a two-step Galton-Watson tree. In other words,

$$\rho_{G_m} \rightsquigarrow \rho \text{ a.s.}$$

Finally, the announced result on the load threshold of double hashing is contained in the following corollary, which is a direct consequence of Theorem 1 and Theorem 2 from [10] or equivalently Theorem 2.1 from [11].

Corollary 1: The load threshold τ^* under which all items can be inserted in the hashtable for fully random hashing w.h.p. is also the load threshold under which all n items but $o(n)$ can be inserted for double hashing w.h.p.

In the next subsection, we introduce a main tool for our proofs: a two-step breadth-first search (BFS) exploration of the hashing graph G_m; and illustrate its analysis through the simple example of upper-bounding the number of short cycles in G_m.

B. Breadth-First Search Exploration

When it is clear from the context, we simply write G instead of G_m. For any $k \in \mathbb{N}^*$ and any graph G^*, we let $\mathcal{N}^k_{G^*}(v)$ denote the k-hop neighborhood of vertex v in the graph G^*; to ease the notation, when $G^* = G$ we simply write $\mathcal{N}^k(v)$, and when $k = 1$ we write $\mathcal{N}_{G^*}(v)$.

We consider the breadth-first search (BFS) exploration of the graph G, starting from a random right-vertex $r_0 \in R$. At each step, we select an active right-vertex and explore all its left-neighbors as well as their own right-neighbors. We define the sets C^R_t and C^L_t of connected right- and left-vertices at time t, the list of active right-vertices A^R_t, and the sets of unexplored right- and left-vertices U^R_t and U^L_t. We have

$$\begin{array}{rcccl} C^R_0 & = & C^L_0 & = & \varnothing, \\ & & A^R_0 & = & (r_0), \\ & & U^R_0 & = & R \setminus \{r_0\}, \\ & & U^L_0 & = & \widetilde{L}. \end{array}$$

At each step t, we proceed as follows:

- we let r_t be the first element in the list A^R_t (so that it is the closest vertex to r_0 in A^R_t);
- if $A^R_t \neq \varnothing$, we do the following updates:

$$\begin{array}{rcl} C^R_{t+1} & = & C^R_t \cup \{r_t\}, \\ C^L_{t+1} & = & C^L_t \cup \mathcal{N}(r_t), \\ A^R_{t+1} & = & (A^R_t \setminus \{r_t\}, \mathcal{N}^2(r_t) \cap U^R_t), \\ U^R_{t+1} & = & U^R_t \setminus \mathcal{N}^2(r_t), \\ U^L_{t+1} & = & U^L_t \setminus \mathcal{N}_{\widetilde{G}}(r_t). \end{array}$$

We let X^R_t be the number of vertices added to $A^R_t \setminus \{r_t\}$ at step t, i.e. $X^R_t = |\mathcal{N}^2(r_t) \cap U^R_t|$, and X^L_t (resp. $\widetilde{X}^L_t$) be the number of left-vertices of G (resp. $\widetilde{G}$) explored at step t, i.e. $X^L_t = |\mathcal{N}(r_t) \cap U^L_t|$ and $\widetilde{X}^L_t = |\mathcal{N}_{\widetilde{G}}(r_t) \cap U^L_t|$, where we identify the vertices in G and $\widetilde{G}$ with the same label. We also let $\theta = \inf\{t \geq 1 : \ A_t \neq \varnothing\}$ be the time at which we complete the exploration of the connected component of r_t in G; θ is a stopping time for the filtration $\mathcal{F}_t = \sigma\Big(\big(C^R_i, C^L_i, A^R_i, U^R_i, U^L_i\big)_{0 \leq i \leq t} \Big)$. With these definitions and for all $t \leq \theta$, it is clear that

$$\begin{array}{rcl} |C^R_t| & = & t, \\ |C^L_t| & = & \sum_{i=0}^{t-1} X^L_i, \\ |A^R_t| & = & 1 + \sum_{i=0}^{t-1} (X^R_i - 1) \\ & \leq & 1 - t + (d-1) \sum_{i=0}^{t-1} X^L_i, \\ |U^R_t| & = & m - 1 - \sum_{i=0}^{t-1} X^R_i, \\ |U^L_t| & = & \binom{m}{2} - \sum_{i=0}^{t-1} \widetilde{X}^L_i \geq \frac{m-1}{2}(m - dt). \end{array}$$

To get used to reasoning on this model of random bipartite graphs with local structure constraints as well as to understand why double hashing graphs converge to trees, we start off by bounding the number of cycles of a given length in G:

Lemma 1: Let C_k be the number of cycles of length k in G. We have

$$\mathbb{E}\left[C_{2k}\right] \leq d^{2k} \tau^k + O(1/m)$$

Proof: Let $r_1, \ldots, r_k \in R$ be distinct vertices of R. For $r_1, \ldots, r_k$ to be on a cycle of size $2k$ of G in this order, there must exist distinct vertices $l_1, \ldots, l_k \in L$ such that, for all i, l_i connects r_{i-1} and r_i, where $r_0 = r_k$. Fixing r and r' distinct in R, the number of vertices connecting them in $\widetilde{G}$ is exactly $\binom{d}{2}$ (as setting the set of indices $\{i, i'\}$ such that $f + ig = r$ and $f + i'g = r'$ or $f + i'g = r$ and $f + ig = r'$ leaves exactly one possibility for the couple $(f, g) \in \widetilde{L}$). Therefore, the probability that r and r' are connected by a vertex in L is

$$\begin{aligned} &\mathbb{P}\Big(r \text{ and } r' \text{ are connected by a vertex in } L\Big) \\ &= 1 - e^{-\frac{2\tau}{m-1}\binom{d}{2}} \\ &= \tbinom{d}{2} \tfrac{2\tau}{m} + O(1/m^2). \end{aligned}$$

Furthermore, it is clear that the probability of existence of *distinct* vertices $l_1, \ldots, l_k$ in L such that, for each i, l_i connects r_{i-1} and r_i, is no larger than the product of the probabilities of existence of each l_i independently of the others:

$$\begin{aligned} &\mathbb{P}\Big(\forall i,\ \exists l_i \in L \text{ connecting } r_{i-1} \text{ and } r_i;\ l_i \neq l_j,\ \forall j \neq i\Big) \\ &\leq \textstyle\prod_i \mathbb{P}\Big(\exists l_i \in L \text{ connecting } r_{i-1} \text{ and } r_i\Big) \\ &\leq \left(\tfrac{\tau d^2}{m}\right)^k + O(1/m^{k+1}). \end{aligned}$$

We conclude by summing over all the $m \ldots (m{-}k{+}1) \leq m^k$ possible choices of $r_1, \ldots, r_k$. ■

C. Proofs

We now return to the proof of the main theorem. Lemma 2 shows that the joint distribution of the numbers of children of the left-vertices explored during the first t steps of BFS is asymptotically very close to that we would obtain in a two-step Galton-Watson tree.

Lemma 2: On an enlarged probability space, there exists a sequence $(Y_t^L)_{t\geq 0}$ of i.i.d. $\mathrm{Poi}(\tau d)$ variables such that

$$\begin{aligned}&\mathbb{P}\Big((X_0^L,\dots,X_{(t-1)\wedge\theta}^L)\neq(Y_0^L,\dots,Y_{(t-1)\wedge\theta}^L)\Big)\\&\leq \tfrac{1}{m-1}(\tau d^2t^2+\tau^2dt+2\tau t).\end{aligned}$$

We denote by $\mathrm{d_{TV}}(p,q)$ the distance in total variation between the two distributions p and q. To ease the notation, we will also use $\mathrm{d_{TV}}$ for random variables, which will refer to the distance between their distributions. The maximal coupling inequality says that, over an enlarged probability space, there exists a coupling of X and Y such that $\mathbb{P}(X\neq Y)=\mathrm{d_{TV}}(X,Y)$.

Proof: For any $t\leq\theta$, X_t^L is a binomial random variable of parameters $|U_t^L\cap\mathcal{N}_{\widetilde{G}}(r_t)|$ and $1-e^{-\frac{2\tau}{m-1}}$. We have that $|\mathcal{N}_{\widetilde{G}}(r_t)|=\frac{m-1}{2}d$ and, as any two vertices in R (and in particular r_i and r_t for any $i<t$) are connected by exactly $\binom{d}{2}$ vertices in $\widetilde{L}$, we obtain the bound $\frac{m-1}{2}d-\binom{d}{2}t\leq|U_t^L\cap\mathcal{N}_{\widetilde{G}}(r_t)|\leq\frac{m-1}{2}d$. Furthermore, $\frac{2\tau}{m-1}-\frac{2\tau^2}{(m-1)^2}\leq 1-e^{-\frac{2\tau}{m-1}}\leq\frac{2\tau}{m-1}$. It follows

$$\begin{aligned}&\mathbb{P}\Big((X_0^L,\dots,X_{(t-1)\wedge\theta}^L)\neq(Y_0^L,\dots,Y_{(t-1)\wedge\theta}^L)\Big)\\&\leq\mathbb{E}\Big[\textstyle\sum_{i=0}^{(t-1)}\mathbf{1}(i\leq\theta)\mathbb{P}(X_i^L\neq Y_i^L\big|\mathcal{F}_i)\Big]\end{aligned}$$

and

$$\begin{aligned}&\mathbb{E}\Big[\mathrm{d_{TV}}(X_i^L,Y_i^L)|\mathcal{F}_i\Big]\\&\leq\mathbb{P}\Big(\mathrm{Bin}(\tbinom{d}{2}i,1-e^{-\frac{2\tau}{m-1}})\neq 0\Big)\\&+\mathrm{d_{TV}}\Big(\mathrm{Bin}(\tfrac{m-1}{2}d,1-e^{-\frac{2\tau}{m-1}}),\mathrm{Bin}(\tfrac{m-1}{2}d,\tfrac{2\tau}{m-1})\Big)\\&+\mathrm{d_{TV}}\Big(\mathrm{Bin}(\tfrac{m-1}{2}d,\tfrac{2\tau}{m-1}),\mathrm{Poi}(\tau d)\Big)\\&\leq\tfrac{2\tau i}{m-1}\tbinom{d}{2}+\tfrac{\tau^2d}{m-1}+\tfrac{2\tau}{m-1}.\end{aligned}$$

We conclude by using the maximal coupling inequality and then summing over i. ■

Next, Lemma 3 asserts that w.h.p. the graph explored in the first t steps of BFS is a tree.

Lemma 3: For an integer $t\leq\theta$, the portion of the graph G explored until step t becomes a tree w.h.p. as $m,n\to\infty$. More precisely,

$$\mathbb{P}\Big(G[(C_t^R\cup A_t^R)\cup C_t^L]\text{ not a tree}\Big)\leq\frac{\tau^2d^4t+\tau^2d^4t^2}{m-1}.$$

Proof: Loops get formed in the portion of the graph explored at step t when one of the following events occur:

- there exists a vertex $r\in U_t^R$ connected to r_t by at least two distinct vertices in L;
- there exists a vertex $l\in U_t^L\cap L$ which connects r_t and some vertex $r\in A_t^R$.

Indeed, assuming $G[(C_t^R\cup A_t^R)\cup C_t^L]$ is a tree and none of the two events above occur, it is easy to see that $G[(C_{t+1}^R\cup A_{t+1}^R)\cup C_{t+1}^L]$ is still a tree. At step t, if a loop is created, it must contain a newly explored left-vertex (i.e. a vertex in $C_{t+1}^L\setminus C_t^L$), otherwise it would not involve any new vertex or edge and hence would also be a loop in $G[(C_t^R\cup A_t^R)\cup C_t^L]$; furthermore, we can always assume that loop contains r_t. If the second event does not occur, the newly added vertices are connected to those in $(C_t^R\cup A_t^R)\cup C_t^L$ only through r_t, and thus any new loop must be contained in $G[\{r_t\}\cup(A_{t+1}^R\setminus A_t^R)]$, which is prevented since the first event does not occur either. Let us call E_t and E_t' the two events considered.

$$\begin{aligned}\mathbb{P}(E_t)&\leq\mathbb{E}\Big[\textstyle\sum_{r\in U_t^R}\mathbb{P}(|\mathcal{N}(r)\cap\mathcal{N}(r_t)|\geq 2\big|\mathcal{F}_t)\Big]\\&\leq\tfrac{\tau^2d^4}{m-1},\\\mathbb{P}(E_t')&\leq\mathbb{E}\Big[\sum_{r\in A_t^R\setminus\{r_t\}}\mathbb{P}(\mathcal{N}(r)\cap\mathcal{N}(r_t)\neq\varnothing\big|\mathcal{F}_t)\Big]\\&\leq\tfrac{\tau d^2}{m-1}(\mathbb{E}[|A_t^R|]-1).\end{aligned}$$

We can bound $\mathbb{E}[|A_t^R|]$ as follows

$$\mathbb{E}[|A_t^R|]-1\leq -t+(d-1)\sum_{i=0}^{t-1}\mathbb{E}[X_i^L]\leq\tau d^2t.$$

Summing over t yields the desired result. ■

The hashing graph G obtained may be a tree, but remember that we merged all the items with the same choices of keys into a single left-vertex of G. Lemma 4 shows that each left-vertex explored in the first t steps of BFS actually correspond to a single item, i.e. the items explored were given different choices of keys, w.h.p.

Lemma 4: For an integer $t\leq\theta$, no left-vertex explored until step t has two copies or more (i.e. $Z_l\leq 1$ for all $l\in C_t^L$) w.h.p. as $m,n\to\infty$. We have

$$\mathbb{P}\Big(\exists l\in C_t^L\,:\,Z_l\geq 2\Big)\leq\frac{4\tau^3dt}{(m-1)^2}.$$

Proof: The result follows straightforwardly from the union bound:

$$\begin{aligned}&\mathbb{P}\Big(\exists l\in C_t^L\,:\,Z_l\geq 2\Big)\\&\leq\mathbb{E}[|C_t^L|](1-\mathbb{P}\big(\mathrm{Poi}(\tfrac{2\tau}{m-1})\leq 1)))\\&=\textstyle\sum_{i=0}^{t-1}\mathbb{E}[X_i^L]\big(1-e^{-\frac{2\tau}{m-1}}-\frac{2\tau}{m-1}e^{-\frac{2\tau}{m-1}}\big)\\&\leq\tau dt\tfrac{4\tau^2}{(m-1)^2}.\end{aligned}$$

■

We are now ready to show that the average distribution $\overline{\rho}_m$ of the k-hop neighborhood of a random root-vertex r_0 in a random graph G_m tends to the distribution of a two-step Galton-Watson tree cut at depth k, for any $k\in\mathbb{N}$ and as $m\to\infty$ (Proposition 1).

Proof: [Proof of Proposition 1] Let $k\in\mathbb{N}$. For any finite tree T of depth at most k (or more generally for rooted graphs of radius at most k) and any collection of such trees $\mathcal{T}$, let $A_T=\{[G]\in\mathcal{G}_*,\ (G)_k\simeq T\}$, where $\simeq$ refers to rooted isomorphism, and $A_{\mathcal{T}}=\cup_{T\in\mathcal{T}}A_T$. For every $\epsilon>0$ and any finite k, there exists a finite set $\mathcal{T}$ of trees of depths at most k in which every odd-depth vertex has degree d and such that $\rho(A_{\mathcal{T}})=\sum_{T\in\mathcal{T}}\rho(A_T)\geq 1-\epsilon$. Let t be the maximum size of the trees in $\mathcal{T}$; it means that, with probability at least $1-\epsilon$, a BFS run during t steps on a two-step Galton-Watson tree sampled from ρ will explore at least the k-hop neighborhood of the root.

Let r_0 be a random right-vertex in the random graph G_m. According to Lemmas 3 and 4 and with probability at least $1 - \frac{\tau^2 d^4 t + \tau^2 d^4 t^2}{m-1} - \frac{4\tau^3 dt}{(m-1)^2} \underset{m\to\infty}{\to} 1$, the graph induced by $[G_m, r_0]$ on the vertices explored during the first t steps of BFS is a tree, with all left-vertices having exactly one copy. Hence, in particular the offspring of each left-vertex in this tree is of size $d-1$. Furthermore, according to Lemma 2 and with probability at least $1 - \frac{\tau d^2 t^2 + \tau^2 dt + 2\tau t}{m-1} \underset{m\to\infty}{\to} 1$, there is a coupling between the numbers of newly explored left-vertices in the graph $[G_m, r_0]$ during the first t steps of BFS and a sequence of i.i.d. Poi(τd) random variables. Therefore, on an event of probability tending to 1 as $m \to \infty$, we can couple the exploration during the first t steps of BFS on $[G_m, r_0]$ and on a two-step Galton-Watson tree sampled from ρ. On the event that these two exploration can be coupled and for any $T \in \mathcal{T}$ such that the BFS exploration on T will have explored at least the k-hop neighborhood of the root within t steps, it follows the BFS exploration on $[G_m, r_0]$ will also have explored at least the k-hop neighborhood of r_0. Then, for any $T \in \mathcal{T}$, we have

$$\begin{aligned} &|\mathbb{P}((G_m, r_0)_k \simeq T) - \rho(A_T)| \\ &\leq \tfrac{\tau^2 d^4 t + \tau^2 d^4 t^2 + \tau d^2 t^2 + \tau^2 dt + 2\tau t}{m-1} + \tfrac{4\tau^3 dt}{(m-1)^2}. \end{aligned}$$

Then, it follows $\lim_{m\to\infty} \overline{\rho}_m(A_T) = \rho(A_T)$ for any $T \in \mathcal{T}$.

For any bounded uniformly continuous function f, there exists $k \in \mathbb{N}$ such that $|f((G)_k) - f(G)| \leq \epsilon$ for all rooted graphs $G \in \mathcal{G}_*$. For this k we can define a finite collection of trees $\mathcal{T}$ as before, such that $\rho(\mathcal{T}) \geq 1 - \epsilon$. For m large enough, we have $\overline{\rho}_m(\mathcal{T}) \geq 1 - 2\epsilon$, and

$$\begin{aligned} \left| \int f \mathrm{d}\overline{\rho}_m - \int f \mathrm{d}\rho \right| &\leq \epsilon(1 + 3||f||_\infty) \\ &+ \textstyle\sum_{T\in\mathcal{T}} f(T) \big|\overline{\rho}_m(A_T) - \rho(A_T)\big|. \end{aligned}$$

Letting $m \to \infty$ and then $\epsilon \to 0$ completes the proof. ■

We will now prove *almost sure* local weak convergence of the random graph G_m towards the two-step Galton-Watson tree (Theorem 1). To that end, we use the Azuma-Hoeffding measure-concentration inequality:

Proposition 2: Let $M = (M_t)_{0\leq t\leq m}$ be a martingale with respect to a filtration $\mathcal{F} = (\mathcal{F}_t)_{0\leq t\leq m}$. Suppose there exists constants $c_1, \ldots, c_m$ such that, for all $1 \leq t \leq m$, the following holds:

$$\big|M_t - M_{t-1}\big| \leq c_t.$$

Then, for all $\epsilon > 0$,

$$\mathbb{P}\Big(\big|M_m - M_0\big| \geq \epsilon\Big) \leq 2e^{-2\epsilon^2 / \sum_{0\leq t\leq m} c_m^2}.$$

Proof: [Proof of Theorem 1] Let $k \in \mathbb{N}^*$ and H be a rooted graph of radius at most k. We define A_H as in the proof of Proposition 1. We let h be the number of right-vertices in H; we focus on m large enough such that $\ln m \geq h$. Recall that $\rho_{G_m}(A_H) = \frac{1}{m}\sum_{r_0\in R_m} \mathbf{1}\left((G_m, r_0)_k \simeq H\right)$.

For any $r \in R_m = \{1, \ldots, m\}$, we define

$$\zeta_r = (Z_l :\ l \in \mathcal{N}_{\widetilde{G}}(r) \text{ and } l \notin \mathcal{N}_{\widetilde{G}}(r'), \forall r' < r)$$

and we let $\mathcal{F}_t = \sigma\Big((\zeta_r)_{1\leq r\leq t}\Big)$. It is easy to obtain from Chernoff's bound that

$$\mathbb{P}(|\zeta_r| \geq \ln m) \leq \mathbb{P}(\sum_{l\in\mathcal{N}_{\widetilde{G}}(r)} Z_l \geq \ln m) \leq \frac{me^{-\tau d}}{m^{\ln\frac{\ln m}{\tau d}}},$$

so that

$$\mathbb{P}(\exists r \in R_m \text{ such that } |\zeta_r| \geq \ln m) \leq \frac{m^2 e^{-\tau d}}{m^{\ln\frac{\ln m}{\tau d}}} \underset{m\to\infty}{\to} 0.$$

We say that G_m is valid if $|\zeta_r| < \ln m$ for all r, and that ζ_r is valid if $|\zeta_r| < \ln m$. We will sometimes write $G_m(\zeta)$ and $\rho_{G_m(\zeta)}(A_H)$ to avoid confusion. We define $\overline{\rho}_m(A_H|G_m \text{ valid})$ as the expectation of $\rho_{G_m}(A_H)$ over the random graph G_m conditionally on G_m valid.

Now, we let $M_t = \mathbb{E}[\rho_{G_m}(A_H)|\mathcal{F}_t, G_m \text{ valid}]$. We have $\mathbb{E}[M_{t+1}|\mathcal{F}_t] = \mathbb{E}[\rho_{G_m}(A_H)|\mathcal{F}_t, G_m \text{ valid}] = M_t$, thus M is indeed a martingale with respect to $\mathcal{F}$. Note that $M_0 = \overline{\rho}_m(A_H|G_m \text{ valid})$ and, for G_m valid, $M_m = \rho_{G_m}(A_H)$. Consider two sequences $\zeta = (\zeta_r)_{1\leq r\leq m}$ and $\zeta' = (\zeta'_r)_{1\leq r\leq m}$ differing only in one value, say $\zeta_r \neq \zeta'_r$. There are at most $(1+|\zeta_r|(d-1))h$ right-vertices r_0 in $G_m(\zeta)$ such that $(G_m(\zeta), r_0)_k \simeq H$ and r has a 2-hop neighbor in $R \cap (G_m(\zeta), r_0)_k$. Assuming ζ_r and ζ'_r are valid, we obtain $|\rho_{G_m(\zeta)}(A_H) - \rho_{G_m(\zeta')}(A_H)| \leq \frac{2dh\ln m}{m}$. Therefore, assuming ζ_t is valid for $t < r$, we have

$$\begin{aligned} &|M_r - M_{r-1}| \\ &\leq \mathbb{E}\Big[\max_{\zeta_r,\zeta'_r}\big|\rho_{G_m(\zeta)}(A_H) - \rho_{G_m(\zeta')}(A_H)\big|\Big|\mathcal{F}_{r-1}, G_m \text{ valid}\Big] \\ &\leq \tfrac{2dh\ln m}{m}. \end{aligned}$$

Then, the Azuma-Hoeffding inequality yields

$$\begin{aligned} &\mathbb{P}\Big(\big|\rho_{G_m}(A_H) - \overline{\rho}_m(A_H|G_m \text{ valid})\big| \geq \epsilon\Big|G_m \text{ valid}\Big) \\ &\leq 2e^{-m\epsilon^2/(2h^2d^2\ln^2 m)}. \end{aligned}$$

Furthermore, it is easy to check that

$$\begin{aligned} \big|\overline{\rho}_m(A_H) - \overline{\rho}_m(A_H|G_m \text{ valid})\big| &\leq 2\mathbb{P}(G_m \text{ not valid}) \\ &\leq \tfrac{2m^2e^{-\tau d}}{m^{\ln\frac{\ln m}{\tau d}}}, \end{aligned}$$

and it follows that

$$\begin{aligned} \mathbb{P}\Big(\big|\rho_{G_m}(A_H) - \overline{\rho}_m(A_H)\big| \geq \epsilon\Big) &\leq 2e^{-\frac{m\epsilon^2}{2h^2d^2\ln^2 m}} \\ &+ \tfrac{2m^2e^{-\tau d}}{m^{\ln\frac{\ln m}{\tau d}}}. \end{aligned}$$

The term on the right-hand side is the general term of a convergent series, hence we can conclude using the Borel-Cantelli lemma and then the same argument as for Proposition 1. ■

VI. Conclusion

We have shown that the approach via the cavity method for computing load thresholds of multiple-choice hashing extends naturally from fully random hashing to double hashing. The interest of this approach is that the results on the maximum size of matchings in random graphs can be fully re-used once one proves that the hashing graphs have the same local weak limit for the schemes considered, which leaves only smaller, scheme-dependent steps to check.

The hashing graph of double hashing differs from that of fully random hashing through its local structure, which is constrained in a simple way. However, more complex constraints on the local structure may perhaps be handled via the same approach.

References

[1] R. Pagh and F. F. Rodler, "Cuckoo hashing," in *ESA*, ser. Lecture Notes in Computer Science, F. Meyer auf der Heide, Ed., vol. 2161. Springer, 2001, pp. 121–133.

[2] A. Frieze, P. Melsted, and M. Mitzenmacher, "An analysis of random-walk cuckoo hashing," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2009, pp. 490–503.

[3] N. Fountoulakis, K. Panagiotou, and A. Steger, "On the insertion time of cuckoo hashing," *CoRR*, vol. abs/1006.1231, 2010.

[4] M. Khosla, "Balls into bins made faster," in *ESA*, ser. Lecture Notes in Computer Science. Springer, 2013.

[5] D. Zuckerman, "Simulating bpp using a general weak random source," *Algorithmica*, vol. 16, no. 4-5, pp. 367–391, 1996.

[6] S. Vembu and S. Verdú, "Generating random bits from an arbitrary source: Fundamental limits," *Information Theory, IEEE Transactions on*, vol. 41, no. 5, pp. 1322–1332, 1995.

[7] M. Mitzenmacher, "Balanced allocations and double hashing," *CoRR*, vol. abs/1209.5360, 2012.

[8] L. J. Guibas and E. Szemeredi, "The analysis of double hashing," *Journal of Computer and System Sciences*, vol. 16, no. 2, pp. 226–274, 1978.

[9] G. Lueker and M. Molodowitch, "More analysis of double hashing," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*. ACM, 1988, pp. 354–359.

[10] M. Lelarge, "A new approach to the orientation of random hypergraphs," in *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '12. SIAM, 2012, pp. 251–264. [Online]. Available: http://dl.acm.org/citation.cfm?id=2095116.2095139

[11] M. Leconte, M. Lelarge, and L. Massoulié, "Convergence of multivariate belief propagation, with applications to cuckoo hashing and load balancing," in *SODA*, S. Khanna, Ed. SIAM, 2013, pp. 35–46.

[12] N. Fountoulakis and K. Panagiotou, "Orientability of random hypergraphs and the power of multiple choices," in *Automata, Languages and Programming*. Springer, 2010, pp. 348–359.

[13] M. Dietzfelbinger, A. Goerdt, M. Mitzenmacher, A. Montanari, R. Pagh, and M. Rink, "Tight thresholds for cuckoo hashing via xorsat," in *Proceedings of the 37th international colloquium conference on Automata, languages and programming*, ser. ICALP'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 213–225. [Online]. Available: http://dl.acm.org/citation.cfm?id=1880918.1880943

[14] A. Frieze and P. Melsted, "Maximum matchings in random bipartite graphs and the space utilization of cuckoo hash tables," *Random Structures & Algorithms*, vol. 41, no. 3, pp. 334–364, 2012.

[15] M. Dietzfelbinger and C. Weidling, "Balanced allocation and dictionaries with tightly packed constant size bins," in *ICALP*, ser. Lecture Notes in Computer Science, L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, Eds., vol. 3580. Springer, 2005, pp. 166–178.

[16] 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 eighteenth annual ACM-SIAM symposium on Discrete algorithms*, ser. SODA '07. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 469–476. [Online]. Available: http://dl.acm.org/citation.cfm?id=1283383.1283433

[17] D. Fernholz and V. Ramachandran, "The k-orientability thresholds for gn, p," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, ser. SODA '07. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 459–468. [Online]. Available: http://dl.acm.org/citation.cfm?id=1283383.1283432

[18] P. Gao and N. C. Wormald, "Load balancing and orientability thresholds for random hypergraphs," in *Proceedings of the 42nd ACM symposium on Theory of computing*, ser. STOC '10. New York, NY, USA: ACM, 2010, pp. 97–104. [Online]. Available: http://doi.acm.org/10.1145/1806689.1806705

[19] C. Bordenave, M. Lelarge, and J. Salez, "Matchings on infinite graphs," *Probability Theory and Related Fields*, pp. 1–26, 2012.

[20] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations," *SIAM J. Comput.*, vol. 29, no. 1, pp. 180–200, 1999.

[21] M. Mitzenmacher and J. Thaler, "Peeling arguments and double hashing," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*. IEEE, 2012, pp. 1118–1125.

[22] D. Aldous and R. Lyons, "Processes on unimodular random networks," *Electron. J. Probab.*, vol. 12, pp. no. 54, 1454–1508, 2007.

[23] C. Bordenave, "Lecture notes on random graphs and probabilistic combinatorial optimization!! draft in construction!!" 2012.