

Applications of Chernoff's Inequality

16.1. QuickSort is Quick

We revisit **QuickSort**. We remind the reader that the running time of **QuickSort** is proportional to the number of comparisons performed by the algorithm. Next, consider an arbitrary element u being sorted. Consider the i th level recursive subproblem that contains u , and let S_i be the set of elements in this subproblem. We consider u to be *successful* in the i th level, if $|S_{i+1}| \leq \frac{3}{4} |S_i|$. Namely, if u is successful, then the next level in the recursion involving u would include a considerably smaller subproblem. Let X_i be the indicator variable which is 1 if u is successful.

We first observe that if **QuickSort** is applied to an array with n elements, then u can be successful at most $T = \lceil \log_{4/3} n \rceil$ times, before the subproblem it participates in is of size one, and the recursion stops. As noted, the indicator variables X_i are independent, and $\mathbb{P}[X_i = 1] \geq 1/2$. Indeed, when a pivot is chosen uniformly at random from S_i , it falls in the middle half of S_i with probability at least $1/2$. To this end, conceptually divide S_i into four equal parts (in sorted order of its elements). If the pivot falls in middle quarters, then both subproblems have size at most $\frac{3}{4} |S_i|$. And this probability is regardless of which subproblem of S_i contains u .

If u participates in v levels, we have the random variables $X_1, \dots, X_v$. To make things simpler, we extend this series by adding independent random variables, such that $\mathbb{P}[X_i = 1] = 1/2$ for $i > v$. Thus, we have an infinite sequence of independent random 0/1 variables, each equal to 1 with probability at least $1/2$. The question is how many elements in the sequence we need to read until we get T ones.

Lemma 16.1.1. *Let $X_1, X_2, \dots$ be an infinite sequence of independent random 0/1 variables with $\mathbb{P}[X_i = 1] \geq 1/2$. Let M be an arbitrary parameter. Then the probability that we need to read more than $6M$ variables of this sequence until we collect M ones is at most $2 \exp(-M)$.*

Proof: Consider the random variable $Y = \sum_{i=1}^L X_i$, where $L = 6M$. Its expectation is $3M$, and using the **Chernoff inequality**, we get

$$\begin{aligned} \alpha &= \mathbb{P}[Y \leq M] \leq \mathbb{P}[|Y - 3M| \geq 2M] \leq 2 \exp\left(-\frac{2(2M)^2}{6M}\right) \\ &\leq 2 \exp(-M). \end{aligned} \quad \textcolor{blue}{\mathfrak{Z}}$$

Lemma 16.1.2. *For any $c \geq 4$, the probability that **QuickSort** performs more than $6cn \ln n$ comparisons is smaller than $2/n^{c-1}$.*

Proof: For **QuickSort**, we have that if we sort n elements, for $M = c \ln n$ with $c \geq 4$ (so that $M \geq T$), the probability that an element u will participate in more than $6M$ rounds is at most $2 \exp(-M) = 2/n^c$, by **Lemma 16.1.1**. There are n elements being sorted, and as such, by the union bound, the probability that any element would participate in more than $6c \ln n$ recursive calls is smaller than $2/n^{c-1}$, by **Lemma 16.1.1**. Since each element participates in at most one comparison per recursive call it is involved in (as a non-pivot compared against the chosen pivot of its subproblem), the total number of comparisons performed is at most $\sum_u 6c \ln n = 6cn \ln n$. $\textcolor{blue}{\mathfrak{Z}}$

16.2. The minimums they are a-changin?

Let $\Pi = \pi_1 \dots \pi_n$ be a random permutation of $\{1, \dots, n\}$. Let $\mathcal{E}_i$ be the event that π_i is the minimum number seen so far as we read Π ; that is, $\mathcal{E}_i$ is the event that $\pi_i = \min_{k=1}^i \pi_k$. Let X_i be the indicator variable that is one if $\mathcal{E}_i$ happens. We have already seen, and it is easy to verify, that $\mathbb{E}[X_i] = 1/i$. We are interested in how many times the minimum might change^{16.1}; that is, $Z = \sum_{i=1}^n X_i$, and how concentrated the distribution of Z is. The following is perhaps surprising.

Lemma 16.2.1. *The events $\mathcal{E}_1, \dots, \mathcal{E}_n$ are independent (as such, the variables $X_1, \dots, X_n$ are independent).*

Proof: Exercise. $\textcolor{blue}{\mathfrak{Z}}$

^{16.1}The answer, my friend, is blowing in the permutation.

Theorem 16.2.2. *Let $\Pi = \pi_1 \dots \pi_n$ be a random permutation of $1, \dots, n$, and let Z be the number of times that π_i is the smallest number among $\pi_1, \dots, \pi_i$, for $i = 1, \dots, n$. Then, for $t \geq 2e$, we have that*

$$\mathbb{P}[Z > 2t + t \ln n] \leq 1/n^{t/2}.$$

Proof: Follows readily from Chernoff's inequality, as $Z = \sum_i X_i$ is a sum of independent indicator variables, and, by linearity of expectation, we have

$$\begin{aligned} \mu &= \mathbb{E}[Z] = \sum_i \mathbb{E}[X_i] = \sum_{i=1}^n \frac{1}{i} \leq 1 + \int_{x=1}^{n+1} \frac{1}{x} dx = 1 + \ln(n+1) \\ &\leq 2 + \ln n, \end{aligned}$$

and similarly $\mu \geq \ln n$. Next, setting $\delta = t - 1$, and [Lemma 16.7.3](#), yields

$$\mathbb{P}[Z > t(\ln n + 2)] \leq \mathbb{P}[Z > (1 + \delta)\mu] < 2^{-\mu(1+\delta)} < 2^{-t \ln n} \leq n^{-t/2}. \quad \text{🐼}$$

16.3. Routing in a parallel computer

Let G be a graph of a network, where every node is a processor. The processors communicate by sending packets on the edges. Let $[0, \dots, N - 1]$ denote the vertices (i.e., processors) of G , where $N = 2^n$, and G is the hypercube. As such, each processor is identified by a binary string $b_1 b_2 \dots b_n \in \{0, 1\}^n$. Two nodes are connected if their binary strings differ only in a single bit. Namely, G is the binary *hypercube* over n bits.

We want to investigate the best routing strategy for this network topology. We assume that every processor needs to send a message to a single other processor. This is represented by a permutation π , and we would like to figure out how to send the messages encoded by the permutation while creating minimum delay/congestion.

Specifically, in our model, every edge has a FIFO queue^{16.1} of the packets it has to transmit. At every clock tick, the message in front of the queue gets sent. All the processors start sending their packets (to the destination specified by the permutation) at the same time.

^{16.1}First in, first out queue. I sure hope you already knew that.

A routing scheme is *oblivious* if every node that has to forward a packet, inspects the packet, and depending only on the content of the packet, decides how to forward it. That is, such a routing scheme is local in nature, and does not take into account other considerations. Oblivious routing is of course a bad idea—it ignores congestion in the network, and might insist on routing things through regions of the hypercube that are “gridlocked”.

Theorem 16.3.1 ([KKT91]). *For any deterministic oblivious permutation routing algorithm on a network of N nodes each of out-degree n , there is a permutation for which the routing of the permutation takes $\Omega(\sqrt{N/n})$ units of time (i.e., ticks).*

Proof: (Sketch) The above is implied by an averaging argument: construct, for every possible destination, the routing tree of all packets to this specific node. Argue that there must be many edges in this tree that are highly congested (which is not the permutation routing we are looking for). Now, by averaging, there must be a single edge that is congested in “many” of these trees. Pick a source-destination pair from each one of these trees that uses this edge, and complete it into a full permutation in the natural way. Clearly, the congestion of the resulting permutation is high. For the exact details see [KKT91]. 

How do we send a packet? We use *bit fixing*. Namely, the packet from node i always goes to the adjacent node that has the first different bit as we scan the destination string $d(i)$. For example, a packet from (0000) going to (1101) would pass through (1000), (1100), and (1101).

The routing algorithm. We assume each edge has a FIFO queue. The routing algorithm is depicted in **Figure 16.1**.

16.3.1. Analysis

We analyze only step 1 in the algorithm, as step 3 follows from the same analysis. In the following, let ρ_i denote the route taken by v_i in step 1.

Exercise 16.3.2. A packet v_j traveling along a path ρ_j cannot leave a path ρ_i and then rejoin it later. Namely, $\rho_i \cap \rho_j$ is a (possibly empty) contiguous path.

```

RandomRoute(  $v_0, \dots, v_{N-1}$ )
  //  $v_i$ : Packet at node  $i$  to be routed to node  $d(i)$ .
  (i) Pick a random intermediate destination  $\sigma(i)$  from
       $[0, \dots, N-1]$ . Packet  $v_i$  travels to  $\sigma(i)$ .

      // Random sampling done with replacement.
  (ii) Wait until all the packets arrive at their intermediate
       destination.
  (iii) Packet  $v_i$  travels from  $\sigma(i)$  to its destination  $d(i)$ .

```

Figure 16.1: The routing algorithm

Lemma 16.3.3. *Let the route of a message $\mathbf{c}$ follow the sequence of edges $\pi = (e_1, e_2, \dots, e_k)$. Let S be the set of packets whose routes pass through at least one of $(e_1, \dots, e_k)$. Then, the delay incurred by $\mathbf{c}$ is at most $|S|$.*

Proof: A packet in S is said to leave π at that time step at which it traverses an edge of π for the last time. If a packet is ready to follow edge e_j at time t , we define its *lag* at time t to be $t - j$. The lag of $\mathbf{c}$ is initially zero, and the delay incurred by $\mathbf{c}$ is its lag when it traverses e_k . We will show that each step at which the lag of $\mathbf{c}$ increases by one can be charged to a distinct member of S .

We argue that if the lag of $\mathbf{c}$ reaches $\ell + 1$, some packet in S leaves π with lag ℓ . When the lag of $\mathbf{c}$ increases from ℓ to $\ell + 1$, there must be at least one packet (from S) that wishes to traverse the same edge as $\mathbf{c}$ at that time step, since otherwise $\mathbf{c}$ would be permitted to traverse this edge and its lag would not increase. Thus, S contains at least one packet whose lag reaches the value ℓ .

Let τ be the last time step at which any packet in S has lag ℓ . Thus there is a packet $\mathbf{d}$ ready to follow edge e_μ at τ , such that $\tau - \mu = \ell$. We argue that some packet of S leaves π at time τ —this establishes the lemma since once a packet leaves π , it would never join it again and as such will never again delay $\mathbf{c}$.

Since $\mathbf{d}$ is ready to follow e_μ at time τ , some packet ω (which may be $\mathbf{d}$ itself) in S traverses e_μ at time τ . Now ω must leave π at time τ —if not, some packet will follow $e_{\mu+1}$ at time $\tau + 1$ with lag ℓ . But this violates the maximality of τ . We charge to ω the increase in the lag of $\mathbf{c}$ from ℓ to $\ell + 1$. Since ω leaves π , it will never be charged again. Thus, each

member of S whose route intersects π is charged for at most one delay, establishing the lemma. 

Let H_{ij} be an indicator variable that is 1 if ρ_i and ρ_j share an edge, and 0 otherwise. The total delay for v_i is at most $\sum_{j \neq i} H_{ij}$.

Crucially, for a fixed i , the variables $H_{i1}, \dots, H_{iN}$ are almost independent. Formally, they are conditionally independent given the path ρ_i (i.e., fix ρ_i and then evaluate these variables). Indeed, imagine first picking the destination of v_i , and let the associated path be ρ_i . Now, pick the destinations of all the other packets in the network. Since the sampling of destinations is done with replacement, whether or not the path ρ_j of v_j intersects ρ_i , is independent of whether ρ_k intersects ρ_i . Of course, the probabilities $\mathbb{P}[H_{ij} = 1]$ and $\mathbb{P}[H_{ik} = 1]$ are probably different. Confusingly, however, $H_{11}, \dots, H_{NN}$ are not independent. Indeed, imagine k and j being close vertices on the hypercube. If $H_{ij} = 1$, then intuitively it means that ρ_i is traveling close to vertex j (the source of packet v_j), and as such there is a higher probability that $H_{ik} = 1$.

Let

$$\rho_i = (e_1, \dots, e_k),$$

and let $T(e_j)$ be the *extra load* of e_j . That is, $T(e_j)$ is the number of packets (i.e., paths) that pass through e_j (in addition to the i th packet). We have that

$$\sum_{j \neq i} H_{ij} \leq \sum_{m=1}^k T(e_m) \quad \text{and thus} \quad \mathbb{E} \left[\sum_{j \neq i} H_{ij} \right] \leq \mathbb{E} \left[\sum_{m=1}^k T(e_m) \right].$$

Because of symmetry, the variables $T(e)$ have the same distribution for all the edges of G . On the other hand, the expected length of a path is $n/2$, there are N packets, and there are $Nn/2$ edges^{16.1}. We conclude

$$\mathbb{E}[T(e)] = \frac{\text{Total length of paths}}{\# \text{ of edges in graph}} = \frac{N(n/2)}{N(n/2)} = 1.$$

Claim 16.3.4. *The probability that the delay of the i th packet exceeds $7n$ is at most 2^{-7n} .*

^{16.1}Indeed, the hypercube has N vertices, all of degree n . As such, the number of undirected edges is $Nn/2$. Note that the network has Nn directed edges, but the calculation here considers undirected edges.

Proof: For a fixed path ρ_i of length $k = |\rho_i|$, let $\mu_i = \mathbb{E}[\sum_{j \neq i} H_{ij} \mid \rho_i]$. We have

$$\begin{aligned} \mu_i &= \mathbb{E}[\sum_{j \neq i} H_{ij} \mid \rho_i] \leq \mathbb{E}[\sum_{m=1}^k T(e_m) \mid \rho_i] \\ &= \sum_{m=1}^k \mathbb{E}[T(e_m) \mid \rho_i] = \sum_{m=1}^k 1 = k \leq n. \end{aligned}$$

(In expectation over ρ_i , $\mathbb{E}[\mu_i] \leq \mathbb{E}[k] = n/2$.) For a fixed ρ_i , by the Chernoff inequality, specifically [Lemma 16.7.3](#), we have

$$\mathbb{P}\left[\sum_{j \neq i} H_{ij} > 7n \mid \rho_i\right] = \mathbb{P}\left[\sum_{j \neq i} H_{ij} > \frac{7n}{\mu_i} \mu_i \mid \rho_i\right] < 2^{-(7n/\mu_i)\mu_i} \leq 2^{-7n},$$

since $1 + \delta = 7n/\mu_i \geq 7 > 2e$, so $\delta > 2e - 1$. Taking the expectation over all choices of ρ_i yields the unconditional probability $\mathbb{P}[\sum_{j \neq i} H_{ij} > 7n] < 2^{-7n}$. 

Since there are $N = 2^n$ packets, we know that with probability at least $1 - 2^{-6n}$, all packets arrive at their temporary destination with a delay of at most $7n$. Specifically, by the union bound across all $N = 2^n$ packets, the probability of any packet suffering delay $> 7n$ is at most $N \cdot 2^{-7n} = 2^n \cdot 2^{-7n} = 2^{-6n}$.

Theorem 16.3.5. *All packets arrive at their (real) destination in $\leq 16n$ time steps, with probability at least $1 - 1/N$ (note that this is very conservative).*

Proof: In each phase, a packet traverses a path of length at most n with delay at most $7n$, taking at most $8n$ time steps. Summing over both phases gives at most $16n$ time steps (or $14n$ if delay is bounded by $6n$). By the union bound over both phases, the probability that all packets arrive within $16n$ steps is at least $1 - 2 \cdot 2^{-6n} = 1 - 2^{-(6n-1)} \geq 1 - 1/N$. 

16.4. Faraway Strings

Consider the Hamming distance between binary strings. It is natural to ask how many strings of length n one can have, such that any pair of

them is of Hamming distance at least t from each other. Consider two random strings, generated by picking each bit randomly and independently. Thus, $\mathbb{E}[d_H(x, y)] = n/2$, where $d_H(x, y)$ denotes the Hamming distance between x and y . In particular, using the Chernoff inequality, specifically [Corollary 16.7.1](#), we have that

$$\mathbb{P}[d_H(x, y) \leq n/2 - \Delta] \leq \exp(-2\Delta^2/n).$$

Next, consider generating M such strings, where the value of M will be determined shortly. Clearly, the probability that some pair of strings is at distance at most $n/2 - \Delta$, is

$$\alpha \leq \binom{M}{2} \exp(-2\Delta^2/n) < M^2 \exp(-2\Delta^2/n).$$

If this probability is smaller than one, then there is some probability that all the M strings are of distance at least $n/2 - \Delta$ from each other. Namely, there exists a set of M strings such that every pair of them is far. We used here the fact that if an event has probability strictly greater than zero, then there exists an outcome satisfying it. Thus, set $\Delta = n/4$, and observe that

$$\alpha < M^2 \exp(-2n^2/16n) = M^2 \exp(-n/8).$$

Thus, for $M = \exp(n/16)$, we have that $\alpha < 1$. Indeed, $M = \lfloor \exp(n/16) \rfloor$, and $\alpha \leq \binom{M}{2} \exp(-n/8) < \frac{M^2}{2} \exp(-n/8) \leq 1/2 < 1$. We conclude:

Lemma 16.4.1. *There exists a set of $\exp(n/16)$ binary strings of length n , such that any pair of them is at Hamming distance at least $n/4$ from each other.*

[Remark: Since M is defined as $\lfloor \exp(n/16) \rfloor$, the set cardinality is $\lfloor \exp(n/16) \rfloor$, making the integer quantity explicit.]

This is our first introduction to the beautiful technique known as the probabilistic method—we will hear more about it later in the course.

This result also has an interesting interpretation in the Euclidean setting. Indeed, consider the sphere $\mathbb{S}$ of radius $\sqrt{n}/2$ centered at

$$(1/2, \dots, 1/2) \in \mathbb{R}^n.$$

Clearly, all the vertices of the binary hypercube $\{0, 1\}^n$ lie on this sphere. As such, let P be the set of points on $\mathbb{S}$ as generated by [Lemma 16.4.1](#). A pair p, q of points of P has Euclidean distance at least $\sqrt{d_H(p, q)} = \sqrt{n/4} = \sqrt{n}/2$ from each other. We conclude:

Lemma 16.4.2. *Consider the unit hypersphere in $\mathbb{R}^n$, formally defined as $\mathbb{S}^{n-1} = \{u \in \mathbb{R}^n \mid \|u\| = 1\}$. The sphere $\mathbb{S}^{n-1}$ contains a set Q of points, such that each pair of points is at (Euclidean) distance at least one from each other, and $|Q| \geq \exp(n/16)$.*

Proof: Translate the point set P by $-(1/2, \dots, 1/2)$ so that the sphere is centered at the origin, and then scale it down by a factor of $\sqrt{n}/2$. 

16.5. Bibliographical notes

[Section 16.3](#) is based on Section 4.2 in [\[MR95\]](#). A similar result to [Theorem 16.3.5](#) is known for the case of the wrapped butterfly topology (which is similar to the hypercube topology but every node has a constant degree, and there is no clear symmetry). The interested reader is referred to [\[MU05\]](#). The randomized two-phase routing scheme via intermediate random destinations is due to Valiant [\[Val82\]](#). The high-probability bound for [QuickSort](#) using Chernoff bounds is standard; see Hoare [\[Hoa62\]](#) for the original algorithm and Motwani and Raghavan [\[MR95\]](#) for the analysis. The existence of sets of faraway binary strings is closely related to the Gilbert-Varshamov bound in coding theory [\[Gil52\]](#).

16.6. Exercises

Exercise 16.6.1 (More binary strings. More!). [Lemma 16.4.1](#) is somewhat silly, as one can prove a better bound by direct argumentation. Indeed, for a fixed binary string x of length n , show a bound on the number of strings in the Hamming ball around x of radius $n/4$ (i.e., binary strings of distance at most $n/4$ from x). (Hint: interpret the special case of the Chernoff inequality as an inequality over binomial coefficients.)

Next, argue that the greedy algorithm which repeatedly picks a string that is at distance $\geq n/4$ from all strings picked so far, stops after picking at least $\exp(n/8)$ strings.

16.7. From previous lectures

Corollary 16.7.1. Let $X_1, \dots, X_n$ be n independent coin flips, such that $\mathbb{P}[X_i = 0] = \mathbb{P}[X_i = 1] = \frac{1}{2}$, for $i = 1, \dots, n$. Let $Y = \sum_{i=1}^n X_i$. Then, for any $\Delta > 0$, we have $\mathbb{P}[|Y - n/2| \geq \Delta] \leq 2 \exp(-2\Delta^2/n)$.

Theorem 16.7.2. Let $X_1, \dots, X_n$ be n independent variables, where $\mathbb{P}[X_i = 1] = p_i$ and $\mathbb{P}[X_i = 0] = q_i = 1 - p_i$, for all i . Let $X = \sum_{i=1}^n X_i$. $\mu = \mathbb{E}[X] = \sum_i p_i$. For any $\delta > 0$, we have

$$\mathbb{P}[X > (1 + \delta)\mu] < \left(e^\delta / (1 + \delta)^{1+\delta} \right)^\mu.$$

Lemma 16.7.3. Let $X_1, \dots, X_n$ be n independent Bernoulli trials, where $\mathbb{P}[X_i = 1] = p_i$, and $\mathbb{P}[X_i = 0] = 1 - p_i$, for $i = 1, \dots, n$. Let $X = \sum_{i=1}^n X_i$, and $\mu = \mathbb{E}[X] = \sum_i p_i$. For $\delta > 2e - 1$, we have $\mathbb{P}[X > (1 + \delta)\mu] < 2^{-\mu(1+\delta)}$.

Lemma 16.7.4. Let $X_1, \dots, X_n$ be n independent Bernoulli trials, where $\mathbb{P}[X_i = 1] = p_i$, and $\mathbb{P}[X_i = 0] = 1 - p_i$, for $i = 1, \dots, n$. Let $X = \sum_{i=1}^n X_i$, and $\mu = \mathbb{E}[X] = \sum_i p_i$. For $\delta \in (0, 1)$, we have

$$\mathbb{P}[X > (1 + \delta)\mu] < \exp(-\mu\delta^2/3).$$

References

- [Gil52] E. N. Gilbert. [A comparison of signalling alphabets](#). *Bell System Technical Journal*, 31(3): 504–522, 1952.
- [Hoa62] C. A. R. Hoare. [Quicksort](#). *The Computer Journal*, 5(1): 10–16, 1962.
- [KKT91] C. Kaklamanis, D. Krizanc, and T. Tsantilas. [Tight bounds for oblivious routing in the hypercube](#). *Mathematical Systems Theory*, 24(4): 223–232, 1991.
- [MR95] R. Motwani and P. Raghavan. [Randomized Algorithms](#). Cambridge University Press, 1995.
- [MU05] M. Mitzenmacher and E. Upfal. [Probability and Computing: Randomized Algorithms and Probabilistic Analysis](#). Cambridge University Press, 2005.

- [Val82] L. G. Valiant. **A scheme for fast parallel communication.** *SIAM Journal on Computing*, 11(2): 350–361, 1982.