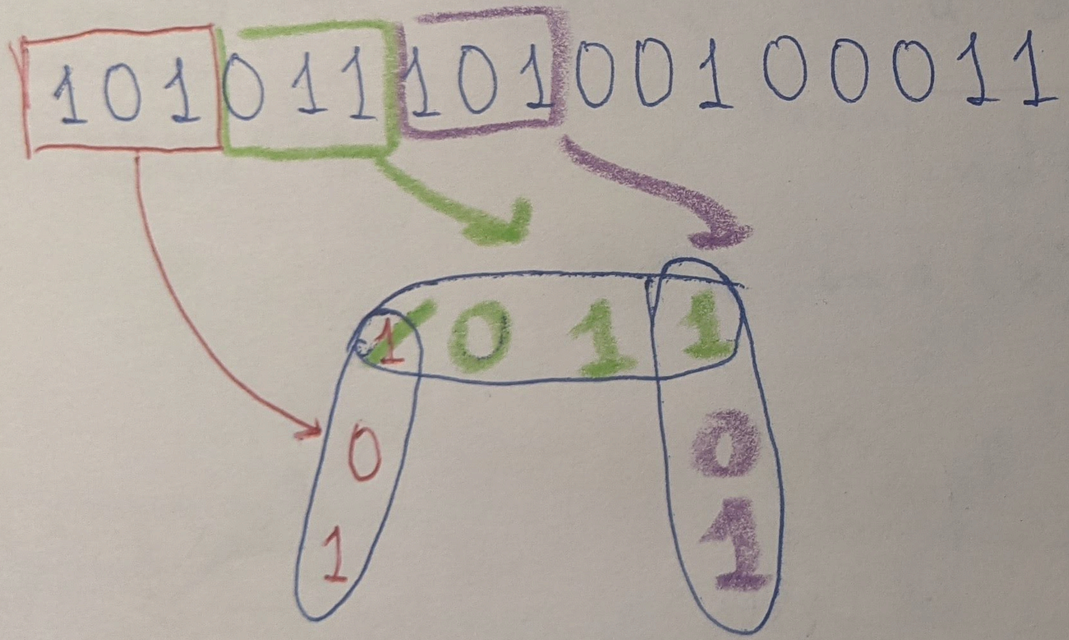
Lecture 19: Entropy Compression



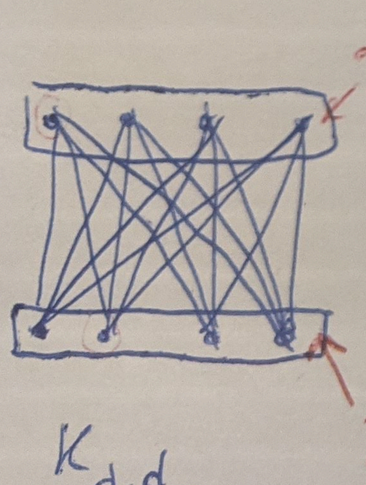
But First:

Thm 9.11: Shearer's Lemma. $X = (X_1, \ldots, X_n)$, let $X_A = (X_i)_{i \in A}$.
If $A_1, A_2, \ldots, A_m \subseteq [n]$ are such that every element of $[n]$ belongs to at least k of them. then

$$k\,H(X) \leq \sum_{i \in [m]} H(X_{A_i}).$$

Thm 9.14: (Kahn, Zhao) If G is a d-regular graph on n vertices then $\text{ind}(G) \leq \text{ind}(K_{d,d})^{n/2d} = (2^{d+1}-1)^{n/2d}$.

Where $\text{ind}(G)$ denotes the number of independent sets in G.

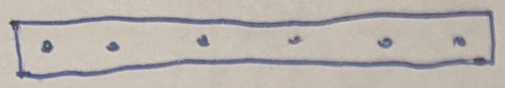


Proof for bipartite G:

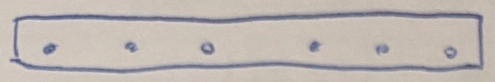
(to prove it for general graphs; Zhao proved

$$(\text{ind}(G))^2 \leq \text{ind}(G \times K_2)$$

↓ d-regular bipartite.



A $|A| = |B| = \frac{n}{2}$.



B

Let X be a random variable uniformly distributed on independent sets of G.

$$\log_2 \text{ind}(G) = H(X) \overset{(2)}{\leq} \frac{|B|}{d} \cdot \log_2 \text{ind}(K_{d,d})$$

$X_S \rightarrow$ restriction of X to $S \subseteq V(G)$.

$\rightarrow$ is random variable supported on independent sets of $G[S]$ (subgraph of G induced by S)

$$d\,H(X) = d\,H(X_A, X_B) = d\,H(X_A) + d\,H(X_B \mid X_A)$$

$H(X_A)$: sum of entropies

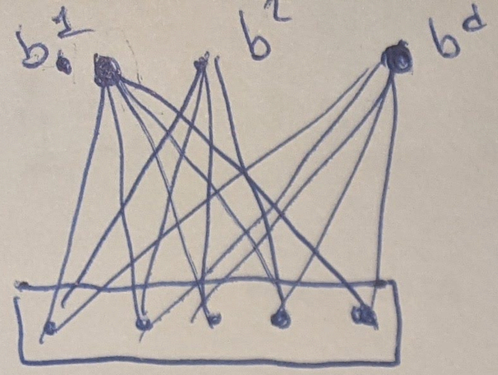
$$\overset{\text{Shearer}}{\leq} \sum_{b \in B} H(X_{N(b)}) + d \sum_{b \in B} H(X_b \mid X_A)$$

$N(b)$: set of neighbors of b.

dropping conditioning

$$\leq \sum_{b \in B} \left(H(X_{N(b)}) + d\,H(X_b \mid X_{N(b)}) \right)$$

Remains to show: (*) $H(X_{N(b)}) + d\,H(X_b \mid X_{N(b)}) \leq \log_2 \left(\text{ind}(K_{d,d})\right)$



$|N(b)| = d$

$$d\,H(X_b \mid X_{N(b)}) = H(X_{b^1}, X_{b^2}, \ldots, X_{b^d} \mid X_{N(b)})$$

X_{b^i} are conditionally independent copies of X_b.

(*) ~~rewritest~~ is equal to

$$H(\underbrace{X_{b^1}, X_{b^2}, \ldots, X_{b^d}, X_{N(b)}}) \leq \log_2(\mathrm{ind}(K_{d,d}))$$

distributed on independent sets of $K_{d,d}$

$\leq$ by uniform bound.

Aside: Agenda for the remainder of the semester

March 24, 29, 31: Remaining material:
Entropy compression, Dependent Random Choice,
Hypergraph containers.
Presentations by students.

↓

Choose the paper,
prefered date soon.
Discuss presentation
with me beforehand
if possible.

April 7, 12, 14:
(15 if needed)

10. Entropy compression

Robin Moser, 2009, term coined by Terry Tao.

Symmetric LLL:

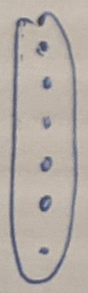
$A_1, \ldots, A_n$ events. s.t.

- $Pr[A_i] \le p$
- each A_i is independent of all the other events except $\le d$
- $e \cdot p \cdot (d+1) \le 1$

Then with positive probability none of A_i occur.

2-coloring hypergraphs:

Let G be a k-uniform hypergraph every edge intersects $\le d$ others.



$p = \left(\frac{1}{2}\right)^{k-1}$

- probability that uniformly random 2 coloring makes an edge monochromatic

By LLL if

$$e \cdot \frac{1}{2^{k-1}} (d+1) \le 1$$

then G is 2-colorable.

i.e. $d \le \frac{2^k}{2e} - 1$.

Satisfiability problem: $x_1, x_2, \ldots, x_n \in \{\text{true}, \text{false}\} = \{0,1\}$.

CNF form of a boolean formula: $\underbrace{x_{i_1} \vee x_{i_2} \vee \ldots \vee (\neg x_{j_1}) \vee (\neg x_{j_\ell})}_{\text{clause}}$

Let S be a family of clauses.

Satisfiability problem asks whether there exist values of $x_1, \ldots, x_n$ s.t. all clauses are satisfied.

2-coloring: variables $x_1, x_2, \ldots, x_n$ for vertices.

for each edge $S = \{i_1, \ldots, i_k\}$.

clauses: $x_{i_1} \vee x_{i_2} \vee \ldots \vee x_{i_k}$

$(\neg x_{i_1}) \vee (\neg x_{i_2}) \vee \ldots \vee (\neg x_{i_k})$

family of these clauses for all edges encode 2-colorability.

Similarly by LLL:

If all clauses have size k

If every clause shares a variable with $\leq \frac{2^k}{e} - 1$ other clauses then there is a {true, false} assignment satisfying all.

Can we find this assignment efficiently?

Naive procedure:

- Randomly choose values for all variables.
- If one of the clauses is not satisfied reset all the variables in it and choose again.

How can we show that "typically" such a procedure finds a satisfying assignment of variables?

We will show that if the procedure continues for a while then we can reduce the entropy of the input stream of random assignments (i.e. we will encode all intermediate assignments in a more compact manner).

which is impossible.

(Eventually this argument boils down to the pigeonhole principle).

Theorem 10.1: There exists $\varepsilon > 0$ such that for every k and every set S of clauses of size k s.t. each clause shares variables with $\leq \varepsilon \cdot 2^k$ others there exists a randomized algorithm which w.h.p. finds an assignment satisfying S in polynomial time.

In particular, S is satisfiable.

Proof: Let R be a sequence of zeros & ones of length M. $R \in \{0,1\}^M$.

We will implement the random procedure described above by using R to obtain choices of variables x_i.

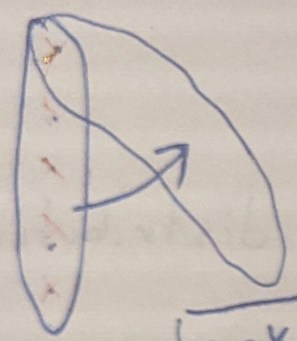
Let $x_1, \ldots, x_n$ be the variables involved, use the first n bits of R to assign their values.

Then we will repeatedly Fix violated clauses in S.

If s is a violated clause. the ~~[illegible]~~ subroutine Fix(s) does the following:

- uses the first k unused bits of R to give new values to variable in clause s.
- If one of the clauses that ~~shares~~ uses a newly chosen variable is violated, say s', run Fix(s')

  } we might need to do this repeatedly.
- otherwise, return.



~~Repeatedly~~ run Fix(s) for all clauses violated in the initial assignment.

If all terminate, then we succeeded in finding an assignment with no violated clauses.

During the algorithm we will maintain its history by recording the initial assignment, & for each recursive call of Fix(s') inside Fix(s)

~~We keep the history of the run of the algorithm~~ we record the index of s' with respect to s. (Importantly, index only uses $k + \lceil \log_2 \varepsilon \rceil$ bits).