(1) $L = \{ x \mid x$ contains exactly two 1's $\}$

Proof idea:

1. Given an input of size n, pick 2-bits out of n at any time. i.e. $\binom{n}{2}$

2. Have negated literals ~~Check~~ all the bits other than the 2-bits chosen in step 1.
these two bits have as positive literals

3. AND ~~the~~ all these literals ~~bits~~, both chosen 2 ~~bits~~ and the inverted ~~bits~~.

4. Repeat the Steps 1, 2 & 3 ~~for~~ for all Combinations

5. OR all the AND's to get the result.

~~For ex:-~~
~~Consider the input $n = 1011$~~
~~following the above sequence, the output of the circuit will be FALSE.~~
~~The output will be true for say $n = 1001$.~~

At each stage we need to remember only (i, j) of our chosen literals both require $\log n$ spaces. Hence it is log space uniform.

2. Let L be the language

$$L = \{ 1^{\langle M, x \rangle} \mid M \text{ on input } x \text{ halts} \}$$

This language is not in P because it is not recursive. But it is in P/Poly ~~iff~~ as we can construct a p-time machine M that decides L ~~using the set of advice strings~~ ~~$\{A_0, A_1, \ldots A_n\}$ where $A_k = 1^k$ for any k~~

The advice string at a given length is of the form 1^z or 0^z where 1^z indicates that z is an encoding of a pair $\langle M, x \rangle$ and that M accepts. 0^z indicates either z is not such an encoding or M does not accept x. Given this string a P-time machine can easily determine for strings of length z whether it is in L. (i.e., if the string is not all 1's it is not in L; if the advice string was 0^z then an input of length z is not in L; only if the advice string was 1^z is 1^z in L).

3) BPP $\subseteq$ P/poly

The Simulation in P/poly of a language ~~&~~ L in BPP ~~[illegible]~~ circuit ~~[illegible]~~ family $c_0 \dots c_n$ with hard-coded ~~each~~ advice strings $A = (a_1 \dots a_m)$ $\rightarrow \approx 12n+1$

where each a_i has length $P(n)$. If $P(n) = n^2$ ~~such that~~ then ~~there are~~ the length of the advice strings ~~[illegible]~~ longer ~~enough to simulate BPP~~ ~~since n^2 [illegible] $P \subseteq BPP$ there are [illegible] languages that are larger than n^2~~

than $n^3 n^2$ and so we wouldn't be able to given all of the advice needed if we were restricted to be P/n^2 algorithm.

4.

f is a one-way function if the following hold of f

① f is one-to-one, and for all strings x,

$$|x|^{1/k} \leq |f(x)| \leq |x|^{k}.$$

② f is in FP

③ f^{-1} is not in FP

example:

given $p < q$ primes, the function $f(p,q) = p \times q$ is suspected to be one way.

5) A machine M in UP will have atmost 1 accepting computational path, but to determine if a boolean formula is in Unique-SAT or not M cannot tell if there is more than 1 true assignment.

Therefore, Unique-SAT is not obviously in UP

6 Suppose S is a sparse NP-hard language.

(Not on test)

Since S is NP-hard $\exists R$, a logspace reduction so that

$$x \in SAT \text{ iff } R(x) \in S.$$

Let $S' = \{\langle 0^K, y\rangle \mid K \geq 0 \wedge \exists x \in SAT\ [K \geq |x| \wedge R(x) = y]\}$

Given a string $\langle 0^m, z\rangle$ we can nondeterministically guess a string x of length at most m and check if $R(x) = z$. So S' is in NP.

It is also sparse since S is sparse.
~~there at most $p(n)$ y's such 0^k is within a polynomial factor of these~~

To see this let $p(n)$ bound number of strings of length n in S. How many strings of length at most n are in S'? There are at most $p(n)$ many different y choices and at most n choices for 0^K. So the total is $np(n)$ w/c is still polynomial.

A reduction from SAT to S' is given by

$$x \longrightarrow \langle 0^{|x|}, R(x)\rangle$$

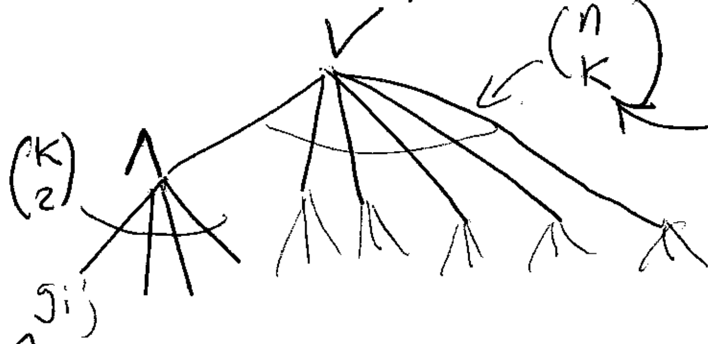
So S' is NP-complete & sparse.

7. Let $B = CVP$. Then since CVP is P-complete under logspace reductions we know $L^{CVP} = P$. On the other hand, $P^{CVP} = P$ as well because if had an $L \in P^{CVP}$ recognized by M^{CVP} then could simulate it in p-time by simulating M then if M queries the oracle we instead directly evaluate the circuit value instance on the query tape. As $CVP \in P$ this whole simulation will be P-time. So $L^{CVP} = P = P^{CVP}$ as desired.

8. The oracle B from class used to show $NP^B \neq P^B$ did it by forcing $L = \{0^n \mid \exists x, |x| = n \; x \in B\}$ to not be in P^B (it is in NP^B). But L above is sparse since the number of strings in L of length $\leq n$ is at most n.

9

CLIQUE$_{n,k}$



n = nodes in graph

k - size of clique

Here we are cycling over all possible k-cliques

Run over choices of input gates where i, j are from the k-subset of n we started with.

10 A <u>sunflower</u> is a family of p sets $\{P_1, \ldots, P_p\}$ where P_i are called <u>petals</u>, each P_i of cardinality $\leq l$, such that all pairs of sets in the family have the same intersection. (called the core).

In the clique lower bound, we approximate the a monotone circuit for $\text{clique}_{n,k}$ by a crude circuit. This is done gate by gate start with the 1st gate. The approximation of both an AND and an OR gate by a crude circuit both involve replacing sunflowers by cores (plucking) if the number of sets in the crude circuit grows bigger than M.