Brian Salinas

CSC 389

Assignment #8

1. (Reading) Done
2. (Encoding in SAT)
   a) Vertex $x_{1,j} \vee x_{2,j} \vee \ldots \vee x_{n,j}$ is true for each j. The first vertex j in the path is where $i = 1$. OR the opposite Vertex $x_{i,1} \vee x_{i,2} \vee \ldots \vee x_{i,n}$ is true for each i. There can only be one first vertex because there is only one instance where a vertex n is in position $i = 1$.
   b) $\bar{x}_{i,j} \vee \bar{x}_{i,k}$ where $j \neq k$. No two vertices can occupy the same position i. No two vertices can be the first vertex. Similar to part a we can also flip this to make sure that a position i isn't tied to two different vertices.
   c) All vertices in graphs are not always connected to each other so to make sure that the path we are creating is actually possible we need to make a clause for this.
      If we're using $i = 1,2$ then we can have a clause that would look like $\bar{x}_{1,j} \vee \bar{x}_{2,k}$ where j, k are vertices in G. if they are adjacent then this will return True. If they are not adjacent than the truth assignments for the accepted state will return false in this clause.

3. (PSPACE)
   a) There are only ever two different possible truth assignments that can be used for each variable in HALF so there are only ever 2^n possible combinations for each possible clause. We only need to count the number of truth assignments that are in ψ and check if half of them are satisfying and we can do this in a $P(n)$ amount of tape.
   b) The smallest k for which we can show $HALF \in SPACE(n^k)$ is $k = 1$. n is arbitrary.
4. **(True or False)**
   a) **True**
      Since this a Non-deterministic TM running in time f(n) we enumerate through all possible options but since we are constrained to f(n) space we can work with our proof that $SAT \in SPACE(n)$ by simply erasing our tape for each truth assignment we have until we eventually get one that accepts.
   b) **True**
      $PSPACE$ is closed under union because we can simply check in order if TMs for A or B accepts the input. For the symmetric difference, however, we have to check if A accepts then B must reject and vice versa. Another idea would be if both A and B accept the input then the complement of that will also be in $PSPACE$ since the complement of a language is also in $PSPACE$.
   c) **True**
      The HALTing problem is a (HALTing set) which is non-decidable and only needs a P amount of space. Either HALT will take n amount of steps before it stops or it will continue to loop forever (in the context of a program) which will mean the it will only ever take n^x amount of space at most.