PSPACE Problems

Space Complexity: If an algorithm A solves a problem X by using $O(f(n))$ bits of memory where n is the size of the input we say that $X \in \text{SPACE}(f(n))$.

The Class PSPACE

Def: $X \in \text{PSPACE}$ if and only if $X \in \text{SPACE}(n^k)$ for some k.

PSPACE Problems are interesting since:

- They form the first interesting class potentially greater than NP.

- The problem of finding winning strategies is in PSPACE.

P $\subseteq$ PSPACE

Assume X$\in$ P and there is a Turing Machine that decides X in time $O(n^k)$. This algorithm can use at most $O(n^k)$ bits of memory. So we get $X \in$ P $\Rightarrow X \in$ PSPACE.

In the other direction

Assume $Y \in$ PSPACE and that a Turing Machine M uses cn^k bits of memory. If we have 3 possible symbols $(0, 1, \#)$ on the input tape there are 3^{cn^k} possible contents on the tape and cn^k possible positions for the head. No possible combination of content/position can be repeated. (Since the machine then would be looping.) This shows that the machine must stop after at most $O(n^k 3^{cn^k})$ steps. So the time complexity cannot be worse than exponential, i.e. $Y \in$ EXPTIME.

The game (GENERALIZED) GEOGRAPHY

Let G be a directed graph with a start vertex v.

Let us assume that we have two players I and II.

I makes the first move. Then the players take turns and make moves.

The moves allowed are moves from a vertex x to an adjacent vertex y *which has not been visited before*.

The first player that cannot move loses the game.

Input: A graph G and a start vertex v.

Goal: Is there a winning strategy for player I?

GEOGRAFI $\in$ PSPACE

We will look at a sketch of an algorithm which decides if there is a winning strategy for the first player in GEOGRAPH.

Given the start configuration $< G, v >$ we let G_1 be G with v and all edges going from v removed.

Let $v_1, v_2, \ldots, v_k$ be the neighbors of v.

Test if $< G_1, v_1 >, < G_1, v_2 >, \cdots < G_1, v_k >$ recursively. If any of these problems does not have a winning strategy we return Yes, otherwise we return No.

It is easy to see that this algorithm can be implemented so that it uses polynomial size memory.

Savitch' Theorem

Given a graph G with n vertices and two vertices a, b there is an algorithm with space complexity $O((\log n)^2)$ which decides if there is a path between a and b or not.

We define

Path(x, y, L)

if $L = 1$ and $x = y$ or $(x, y) \in E(G)$
    return 1
if $L > 1$
    Enumerate all vertices with a counter using $\log n$ bits of memory
    foreach $z \in V(G)$
        Compute $Path(x, z, \lceil \frac{L}{2} \rceil)$. Erase used memory and return value
        Compute $Path(z, y, \lceil \frac{L}{2} \rceil)$. Erase used memory and return value
        save all returned values
        if both computations returns 1
            return 1
return 0

Compute $Path(a, b, n)$. If the answer is 1 we know that there is a path $a \to b$.

In each recursive step we store the information x, y, L. That takes $3 \log n$ bits of memory. The recursion depth is at most $\log n$. The space complexity is $O((\log n)^2)$.

NP $\subseteq$ PSPACE

We know that 3-SAT is NP-Complete. So we just have to show that 3-SAT $\in$ PSPACE.

Given ϕ with n variables we run true all 2^n possible value assignments one at a time. The amount of space needed is $\log 2^n = n$ to keep count of the number of the assignment and $+k$ extra bits of memory.. This gives us space complexity $O(n)$.

Different Complexity Classes

We now have the classes

$$\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE} \subseteq \mathsf{EXPTIME}$$

where EXPTIME is the class of problems that can be decided in TIME(c^{n^k}) for some numbers c, k. It is possible to show that P $\neq$ EXPTIME. No other inequalities are known. This means that no inequalities like P $\neq$ NP eller NP $\neq$ PSPACE are shown to be true.

PSPACE Complete Problems

A problem is PSPACE-Complete if

1. $A \in$ PSPACE

2. Every problem $B \in$ PSPACE can be reduced to A, i.e. $B \leq_P A$.

The problem QSAT

A QSAT-formula is of the form

$$\exists x_1 \forall x_2 \exists x_3 \ldots \forall x_{n-1} \exists x_n \phi(x_1, \ldots, x_n)$$

where ϕ is in 3-SAT-form.

possible values for the variables are $\{0, 1\}$.

$\exists x_1 \forall x_2 \phi(x_1, x_2)$ means that there is a value for x_1 (0 or 1) such that $\phi(x_1, x_2)$ Is true for all values for x_2 (0 och 1).

We want to decide if a formula of this kind are *valid* or not.