MTAT.07.004 — Complexity Theory	December 9th, 2011
Lecture 13	
Lecturer: Peeter Laud	*Scribe(s): Alisa Pankova, Ilya Kuzovkin*

Class #P

Definition 1 Function $f : \{0,1\}^* \to \mathbb{N}$ belongs to class #P if it can decide if DTM M has exactly K certificates on input x. DTM M checks certificates for $L \in NP$ in polynomial time. More formally

$$f(x) = |\{y \in \{0,1\}^{p(x)} \mid M(x,y) = true\}|$$

$\diamondsuit$

Definition 2 Class FP is class of all function $f : \{0,1\}^* \to \mathbb{N}$ computable in polynomial time. $\diamondsuit$

Theorem 1 *If FP = #P then P = NP*

Proof We want to show that if FP = #P then NP $\subseteq$ P. Solving any FP problem takes polynomial time. Our assumption implies that for any NP problem we can learn the number of certificates in polynomial time. So when we have some NP problem we will ask #P how many certificates it has and if number of certificates > 0 we will know that problem has a certificate in NP. ■

#P might be richer than NP

Let's have a look at CYCLE problem for a directed graph. The question it answers it "Does graph G contain simple (no repeating vertices or edges, except for starting vertex) cycle?". This problem can be solved in polynomial time CYCLE $\in$ P.

Next question to ask is how many simple cycles graph G has. This problem is referred as #CYCLE and #CYCLE $\in$ #P.

Theorem 2 *If #CYCLE $\in$ FP then P = NP*

Proof If #CYCLE is in FP, then for any graph we can find amount of cycles in polynomial time. Now assume we have graph G and we want to know if it is Hamiltonian. We introduce new graph G' which is constructed as is shown on Figure 1: for every pair of vertices we add $n \log n$ layers between them, so than there will be $2^{n \log n + 1}$ possible paths from u' to v'.

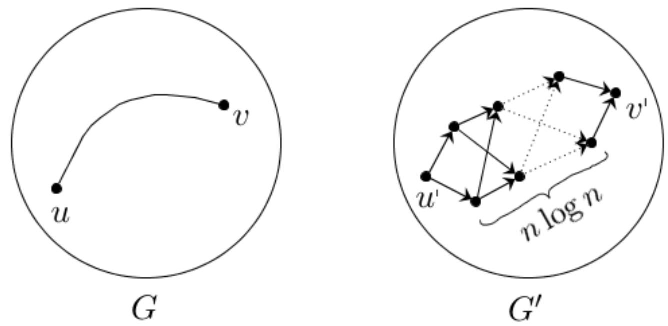

Figure 1: G' size is polynomial of G size

In G number of cycles of length m is at most

$$\frac{n(n-1)\dots(n-m+1)}{m}$$

as we know from the Figure 1 for each cycle of length m in G there are

$$2^{m(n\log n+1)}$$

cycles in G'. Now we can compute total number of cycles in G' as

$$\sum_{m=3}^{n-1}\frac{n(n-1)\dots(n-m+1)}{m}\cdot 2^{m(n\log n+1)}$$

Also we know that if G' is Hamiltonian it will have at least

$$2^{n(n\log n+1)}$$

cycles.

Problem of determining if graph is Hamiltonian is known to be NP-complete. Using our approach we can check if number of cycles in graph G' is larger or equal than $2^{n(n\log n+1)}$. If it is larger or equal, then G' and G are Hamiltonian, if it is smaller, then they are not Hamiltonian. Since all our computation was made in polynomial time in FP (by assumption) then it turns out that we can solve NP-complete problem in polynomial time, which will imply NP $\subseteq$ P (and P $\subseteq$ NP is obvious).

■

#P-completeness

Definition 3 Function $f : \{0,1\}^* \to \mathbb{N}$ is #P-complete if $f \in \#P$ and $FP^f = \#P$. In other words completeness is achieved when function belongs to the class (obviously) and some polynomial function $g \in$ FP, which uses f-Oracle can solve any problem in #P. $\diamondsuit$

Theorem 3 *#SAT is #P-complete.*

#SAT answers the question of how many satisfying valuations particular Boolean formula has.

Proof The proof goes exactly as for "SAT is NP-complete" theorem from Lecture 3. Only difference is that now we are interested not only in the fact some of the branches will accept, but also in how many of them will accept. ■

Perfect matching in bipartite graphs

Definition 4 Perfect matching of undirected (not necessarily bipartite) graph G is set S of edges, such that each vertex of a graph is incident with exactly one edge in S. ♢

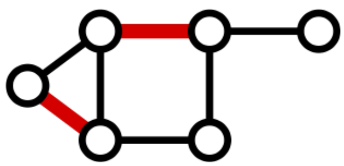

Figure 2: Not a perfect matching

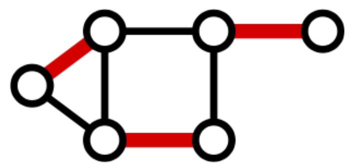

Figure 3: Perfect matching

Definition 5 Graph G is called bipartite is it's vertices set V can be divided into two disjoint subsets V_1 and V_2 such that there is no edge between any two vertices in V_1 and no edge between any two vertices in V_2 (all existing edges go from V_1 to V_2 or vice versa). $\diamondsuit$

We will consider bipartite graphs where $|V_1| = |V_2|$.

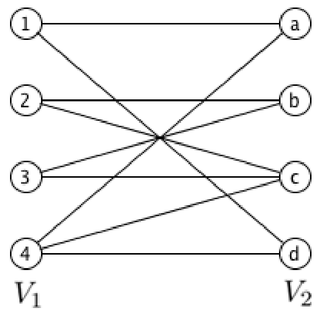

Figure 4: Bipartite graph

Graphs can be represented as adjacency matrices. Here is the representation of the graph from Figure 3.

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

Permanent of a matrix

Definition 6 Permanent of a matrix A is

$$\text{perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

where S_i is a set of a permutations $\{1, \ldots, n\}$.
In other words permanent is sum of multiplications of the elements taken in such way that all factors inside one summand come from different row and column. $\diamondsuit$

$$\text{perm} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} =$$

$$(1 \cdot 1 \cdot 1 \cdot 1) + (1 \cdot 1 \cdot 0 \cdot 1) + (1 \cdot 1 \cdot 1 \cdot 1) + (1 \cdot 1 \cdot 0 \cdot 0) +$$

$$(1 \cdot 0 \cdot 1 \cdot 1) + (1 \cdot 0 \cdot 1 \cdot 0) + (0 \cdot 0 \cdot 1 \cdot 1) + (0 \cdot 0 \cdot 0 \cdot 1) +$$

$$(0 \cdot 1 \cdot 0 \cdot 1) + (0 \cdot 1 \cdot 0 \cdot 1) + (0 \cdot 0 \cdot 0 \cdot 1) + (0 \cdot 0 \cdot 1 \cdot 1) +$$

$$
\begin{gathered}
(0 \cdot 0 \cdot 1 \cdot 1) + (0 \cdot 0 \cdot 0 \cdot 0) + (0 \cdot 1 \cdot 0 \cdot 1) + (0 \cdot 1 \cdot 0 \cdot 1) + \\
(0 \cdot 0 \cdot 0 \cdot 0) + (0 \cdot 0 \cdot 1 \cdot 1) + (1 \cdot 0 \cdot 1 \cdot 1) + (1 \cdot 0 \cdot 1 \cdot 0) + \\
(1 \cdot 1 \cdot 0 \cdot 1) + (1 \cdot 1 \cdot 1 \cdot 1) + (1 \cdot 1 \cdot 0 \cdot 0) + (1 \cdot 1 \cdot 1 \cdot 1) = \\
1 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 1 = \\
4
\end{gathered}
$$

Note If summand is 1 is show that there is a perfect matching, 0 means no matching. Permanent = number of perfect matchings for a graph.

Theorem 4 *Perm is #P-complete (even for 0,1-matrices)*

Proof

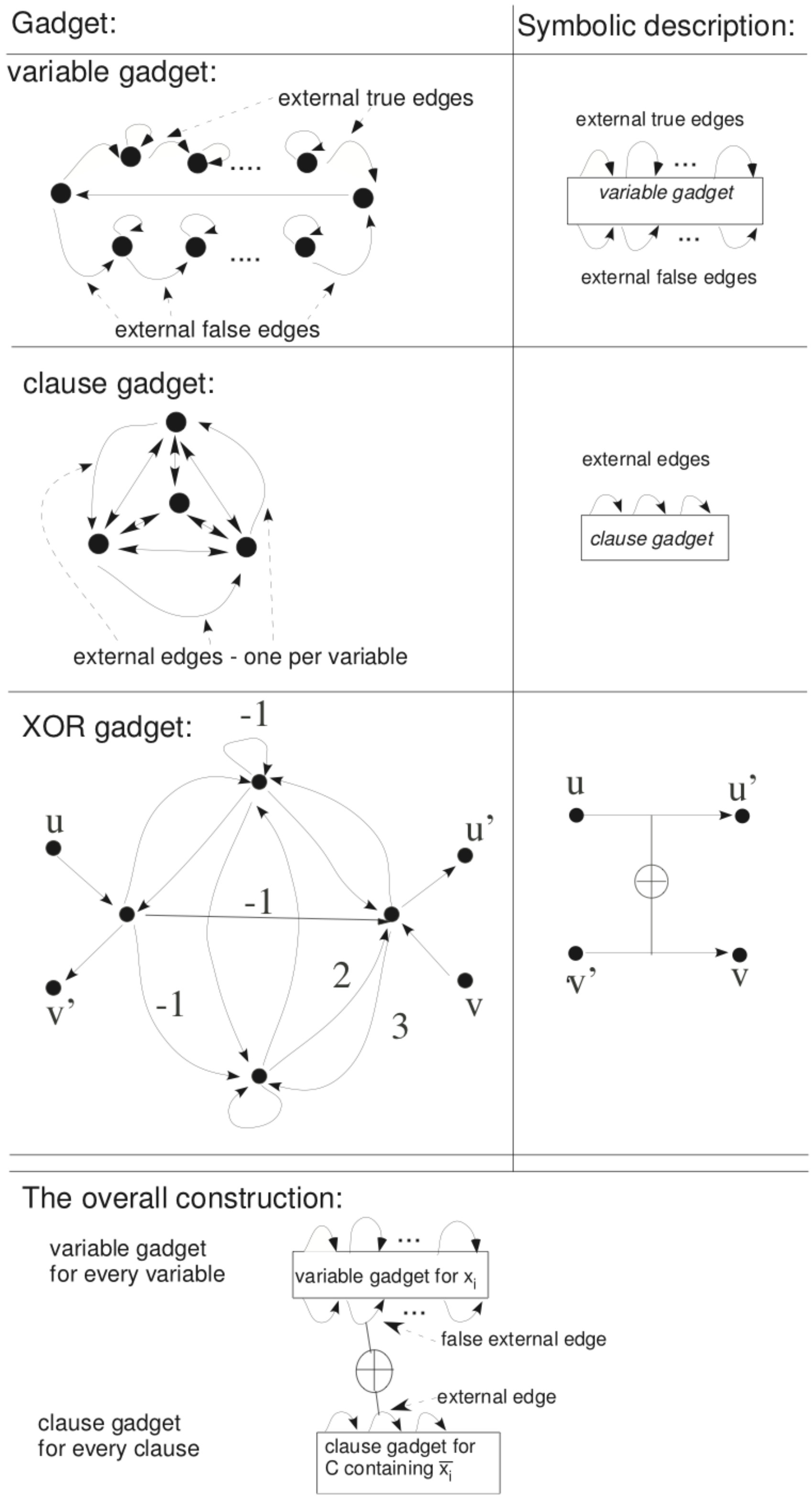

Figure 5: Graph representation of SAT formula, it's variables and clauses.

■

Optimization

Approximability

Various complexity classes