AM 106/206: Applied Algebra		**Prof. Salil Vadhan**
	Problem Set 4	
Assigned: Fri. Oct. 8, 2010		Due: Fri. Oct. 15, 2010 (2:10 PM sharp)

- You may submit your problem sets in the AM106 in the Maxwell–Dworkin basement, or electronically by email to `am106-hw@seas.harvard.edu`. If you use LaTeX, please submit both the source (`.tex`) and the compiled file (`.pdf`). Name your files `PS4-yourlastname`.
- Aim for clarity and conciseness in your solutions, emphasizing the main ideas over low-level details. Justify your answers except when otherwise specified.
- Problems marked [AM106] or [AM106-X] are for AM106 students (though AM206 students should confirm that they know how to do them), and those marked [AM206-X] are for AM206 students. However, AM106 students can do a problem marked [AM206-X] instead of one marked [AM106-X] (for the same value of X) if they wish (out of interest, or for a challenge). If you wish to keep the option of staying in either AM106 or AM206 open until add/drop date, then you should do all problems marked [AM106] and all problems marked [AM206-X].

Problem 1. (Orders of Permutations) What are all the possible orders for elements of S_8 and of A_8? Justify your answers.

Problem 2. (Generating S_n [AM106-A]) For a group G and elements $g_1, \ldots, g_n \in G$, the *subgroup generated by* $g_1, \ldots, g_n$ is defined to be the set of all elements we can obtain by multiplying the g_i's and their inverses together any number of times. Formally:

$$\langle g_1, \ldots, g_n \rangle = \left\{ g_{i_1}^{k_1} g_{i_2}^{k_2} \cdots g_{i_t}^{k_t} : t \in \mathbb{N}, i_1, \ldots, i_t \in \{1, \ldots, n\}, k_1, \ldots, k_t \in \mathbb{Z} \right\}.$$

(Note that a *cyclic* subgroup is a subgroup generated by a *single* generator g. Here we allow multiple generators, so these subgroups need not be cyclic.)

Prove that for $n \geq 2$, $S_n = \langle (12), (12 \cdots n) \rangle$. (Hint: repeatedly use conjugation to obtain all the transpositions.)

Problem 3. (Isomorphisms of Specific Groups) For each of the following pairs of groups (G, H), determine whether or not they are isomorphic. If not, determine whether one is isomorphic to a subgroup of the other. Justify your answers.

1. [AM106-B] $\mathbb{Z}_5$ vs. S_5.
2. $\mathbb{Z}_8^*$ vs. $\mathbb{Z}_{12}^*$.
3. $\mathbb{R}^*$ vs. $\mathbb{C}^*$.
4. [AM206-B] $\mathbb{R}$ vs. $GL_2(\mathbb{R})$.

Problem 4. (From Cayley to Lagrange, Gallian 6.46)

1. Recall that in the proof of Cayley's Theorem, the isomorphism from a group G to a subgroup of $Sym(G)$ takes an element $g \in G$ to the permutation $T_g(x) = gx$. Show that for finite G, the disjoint cycle notation for T_g consists entirely of cycles of length equal to the order of g.

2. Deduce the following corollary of Lagrange's Theorem: the order of an element $g \in G$ divides the order of the group G.

Problem 5. (Parallelism vs. Memory via Group Theory [AM206-A]) In this you will use algebraic properties of the group S_5 to prove an equivalence between two finite computational models for evaluating functions $f : \{0,1\}^n \to \{0,1\}$:

- Small-depth Boolean Formulas: These are defined by induction. A *depth 0 boolean formula* F on n variables is of the form $F(x_1, \ldots, x_n) = x_i$ for some $i \in [n]$. A *depth* $d+1$ *boolean formula* is of the form $F = (G \wedge H)$ or $F = \neg G$, where G and H are formulas of depth at most d, $\wedge$ denotes logical AND, and $\neg$ denotes logical negation. Interpreting 1 as TRUE and 0 as FALSE, every such formula $F(x_1, \ldots, x_n)$ can be interpreted as a function $f : \{0,1\}^n \to \{0,1\}$. For example, the formula $F = (\neg(x_1 \wedge x_2) \wedge \neg(\neg x_1 \wedge \neg x_2))$ is a depth 4 formula computing the function $f : \{0,1\}^2 \to \{0,1\}$ where $f(00) = f(11) = 0$ and $f(01) = f(10) = 1$ (i.e. f=XOR). Depth d boolean formulas capture those functions that can be computed by digital circuits in "parallel time" $\Theta(d)$.

- Small-space Computations: A *branching program* P of *width* w and *length* ℓ on n *variables* consists of a start state $s_0 \in [w]$ (where $[w] = \{1, \ldots, w\}$), a set of accept states $A \subseteq [w]$, a sequence of ℓ indices $i_1, \ldots, i_\ell \in [n]$, and ℓ transition functions $T_1, \ldots, T_\ell : [w] \times \{0,1\} \to [w]$. On an input $x \in \{0,1\}^n$, the program computes its output $P(x)$ as follows: it computes states $s_1, \ldots, s_\ell$ iteratively using the rule $s_j = T_j(s_{j-1}, x_{i_j})$, and outputs 1 if $s_\ell \in A$ and 0 otherwise. The width of a branching program measures the amount of memory the program requires (beyond a time counter), and the length measures the amount of time it requires.

It can be shown that that for any constant w, every function computed by a branching programs of width w and length ℓ can also be computed by a boolean formula of depth $O(\log \ell)$. You will show the converse: every function computed by a boolean formula of depth d can be computed by a width 5 branching program of length at most 4^d.

To do this, you will use an intermediate algebraic computational model. An S_5*-product program* of *length* ℓ on n *variables* consists of a sequence of ℓ triples $(i_1, \sigma_1^{(0)}, \sigma_1^{(1)}), (i_2, \sigma_2^{(0)}, \sigma_2^{(1)}), \ldots, (i_\ell, \sigma_\ell^{(0)}, \sigma_\ell^{(1)}) \in [n] \times S_5 \times S_5$, as well as an *accept permutation* $\alpha \in S_5 \setminus \{\varepsilon\}$. We say such a program *computes* a function $f : \{0,1\}^n \to \{0,1\}$ if for every input $x = x_1 \ldots x_n \in \{0,1\}^n$, the product $\sigma_1^{(x_{i_1})} \sigma_2^{(x_{i_2})} \cdots \sigma_\ell^{(x_{i_\ell})}$ equals the identity ε if $f(x) = 0$ and equals α if $f(x) = 1$.

1. Show that there are $\beta, \gamma \in S_5$ such that β, γ, and $\beta\gamma\beta^{-1}\gamma^{-1}$ are all 5-cycles.

2. Show that if α, α' are conjugates and there is an S_5-product program of length ℓ computing a function f with accept permutation α, then there is also such a program whose accept permutation is α'.

3. Prove by induction on d that if a function is computable by a boolean formula of depth d, then it is computable by an S_5-product program of length at most 4^d with an accept permutation that is a 5-cycle.
4. Prove that every function computable by an S_5-product program of length ℓ is also computable by a width 5 branching program of length ℓ.