CS280, Spring 2001 **Handout 1**
January 22, 2001

1. Welcome to CS280!

2. Reading for Lecture 1: K. Rosen *Discrete Mathematics and Its Applications*, 1.1, 1.2

3. The main message of the first lecture:

> **Truth values *false* and *true* in logic correspond to bits 0 and 1: no wonder that the same set of boolean operations serves equally well in logic and computations. Correct laws of logic (tautologies) are true under any truth assignments of its atomic propositions. Each proposition has a truth table, each truth table is realized by a proposition.**

Definition 1.1. Proposition - a statement which can be assigned some truth value *true* and *false*. Propositions:

Rome is the capital of Italy,
Rome is not the capital of Italy,
John loves Mary, *CS280 is fun*,
$2 + 2 = 4$, $2 \times 2 = 5$.

Not propositions:

"*Rome*", "*John*", "2" (they are names of objects)
"*loves*", "*is*", "*is not*", "=" (they are relation symbols)
"+", "×" (operation symbols), etc.

In logic **atomic propositions** are denoted by letters $p, q, r, \ldots$, from which we can build **(compound) propositions** by the following rules

1. each atomic proposition and a boolean constant **F** or **T** is a proposition
2. if φ, ψ are propositions then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, $(\varphi \leftrightarrow \psi)$, $\neg\varphi$ are also propositions.

Here $\varphi \wedge \psi$ denotes "φ *and* ψ",

$\varphi \vee \psi$ – "φ *or* ψ", $\varphi \to \psi$ – "*if* φ *then* ψ",
$\varphi \leftrightarrow \psi$ – "φ *if and only if* ψ", $\neg\varphi$ – "*not* φ".

Examples of propositions (the outermost parentheses are suppressed for brevity): $\neg\neg p$, $(p \to \mathbf{F})$, $(\neg p \vee q) \leftrightarrow \neg(\mathbf{T} \wedge \neg q)$, etc. Note that for particular propositions the operations $\wedge, \vee, \to, \neg$ may well connect semantically unrelated sentences, like

"$2 \times 2 = 4$" *vee* *"the Moon is made of a Swiss cheese"*

For the time being we did not even care whether a proposition was true or not.

Atomic propositions may be assigned **truth values** `T` (*true*) or `F` (*false*). Boolean constants **F** and **T** have preassigned truth values `F` and `T` respectively. Under each such an assignment the truth value of a compound proposition can be computed by the following **boolean truth tables**:

φ	ψ	$\varphi \wedge \psi$	$\varphi \vee \psi$	$\varphi \to \psi$	$\varphi \leftrightarrow \psi$	$\neg\varphi$
F	F	F	F	T	T	T
F	T	F	T	T	F	T
T	F	F	T	F	F	F
T	T	T	T	T	T	F

Example 1.2. Calculating truth values of propositions

p	q	$(\neg p$	$\vee$	$q)$	$\rightarrow$	$(p$	$\wedge$	$\neg q)$
F	F	T	T		F		F	T
F	T	T	T		F		F	F
T	F	F	F		T		T	T
T	T	F	T		F		F	F

Other propositions having the same truth function: $\neg(p \rightarrow q)$, $\neg(\neg p \vee q)$, $(p \wedge \neg q)$, etc.

Definition 1.3. Tautology is a proposition which is true under all truth values of its atomic propositions. Examples: $(p \rightarrow p)$, $(p \vee \neg p)$, $(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$, $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$. Tautologies sometimes are also called the boolean logical laws.

Recognizing tautologies is a hard computational task, since the straightforward algorithm tries to exhaust all possible combinations of truth values of atomic propositions. Suppose we have a (moderate size) proposition depending on 100 atomic ones. The number of truth value combinations here is 2^{100} which is greater then 10^{30}. Imagine trying to exhaust all those combinations on a super fast computer checking a trillion (i.e. 10^{12}) combinations per second. Then you have to run such a device for $10^{30}/10^{12} = 10^{30-12} = 10^{18}$ seconds, which is more than 3×10^{10} (thirty billion) years without errors, power outages, etc. The problem of whether there exists an algorithm recognizing tautologies substantially faster than by an exhausting search is one of the central theoretical problems in both Computer Science and Mathematics.

Definition 1.4. Propositions φ, ψ are **logically equivalent** is they have the same truth function (notation: $\varphi \Leftrightarrow \psi$). Obviously, $\varphi \Leftrightarrow \psi$ holds if and only if a proposition $\varphi \leftrightarrow \psi$ is a tautology. Examples: $((\neg p \vee q) \rightarrow (p \wedge \neg q)) \Leftrightarrow \neg(p \rightarrow q) \Leftrightarrow \neg(\neg p \vee q) \Leftrightarrow (p \wedge \neg q)$. For a table of useful logical equivalences and simple examples see Rosen p.17.

Definition 1.5. Truth values can be represented by **bits**, i.e. 0 (for F) and 1 (for T).

A truth table of three atomic propositions p, q, r consists of $2^3 = 8$ lines from (FFF) to (TTT). In each line we can independently pick a corresponding truth value which makes $2^8 = 256$ possible non-equivalent truth tables. Likewise, a truth table of n distinct atomic propositions has 2^n lines, therefore, there are 2^{2^n} such tables.

Theorem 1.6. *Each truth table can be realized by a proposition with connectives* $\wedge, \vee, \neg$.
Proof (by an example which provides a general algorithm of recovering a proposition given its proof table). Let a table of p, q, r contain T's in the lines FFF and TFT, and F's in all the other six lines. Step one: build *elementary conjunctions* corresponding to T-lines: $\neg p \wedge \neg q \wedge \neg r$ and $p \wedge \neg q \wedge r$ (specific grouping of conjuncts is not essential). The key observation: each of the elementary conjunctions is true *only in the corresponding line* (FFF and TFT respectively) and false in any other line of the table. Step two: take the disjunction of all elementary conjunctions: here it is $(\neg p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg q \wedge r)$. The resulting proposition is true in each T-line of the table and false in each F-line.

Homework assignments (due Friday 01/26).

A. 1.1: 25cd, 42b

B. 1.2: 12, 14, 28, 29

C. Find a proposition $Q(p, q, r)$ with in true on FFT and TTF and false on all other strings of truth values.