CS F364

Design & Analysis of Algorithms

ALGORITHMS - COMPLEXITY

Complexity Classes

- NP-Completeness Via Reductions
    - Examples: SAT

$\mathbb{NP}$-Completeness Via Reductions

- Claim:
  - If π_1 is $\mathbb{NP}$-hard and $\pi_1 \precsim \pi_2$
  - then π_2 is $\mathbb{NP}$-hard.
- Proof:
  - Since π_1 is $\mathbb{NP}$-hard
    - for all π in $\mathbb{NP}$ $\pi \precsim \pi_1$
  - By transitivity
    - for all π in $\mathbb{NP}$ $\pi \precsim \pi_2$
  - i.e. π_2 is $\mathbb{NP}$-hard

NP-Completeness Via Reductions

- Claim: SAT is NP-Complete.
- Proof:
  - SAT is in NP
    - Given a formula F of length n in Boolean logic, a certificate for satisfiability would be:
      - a Boolean assignment to its variables for which F will evaluate to true.
    - The length of this certificate is the number of variables (i.e. <= n).
    - The time taken for verifying this certificate is
  - CIRCUIT-SAT $\lesssim$ SAT

NP-Completeness Via Reductions

- Claim: CIRCUIT-SAT $\precsim$ SAT
- Proof Argument:
  - The classic technique of constructing an equivalent formula given a circuit does not work:
    - The time for constructing the table is 2^n (why?), given n inputs to the circuit.
      - The size of the circuit need not be exponential in n i.e. the size of the table may be exponential in the size of the circuit.
      - **Exercise**:
        - Construct an example circuit for which this is true.
    - Can one walk the graph and extract the formula?
      - *When **fan-out** is unlimited, the number of paths walked may be exponential*.

NP-Completeness Via Reductions - SAT

- Proof [contd.]: CIRCUIT-SAT $\precsim$ SAT
  - Given a circuit C:
    - assume each input line is marked with a variable a_j
    - mark each line connecting the output of a gate to the input of another with a variable $b_{i,}$ i>0
    - mark the final output as b_0
  - Construct a formula:
    - **$(\text{AND}_i\ g_i)$ AND b_0** where each **g_i** is a formula corresponding to a gate:
      - **b_{i1} op b_{i2} <--> b_{i3}** where **b_{i1}** and **b_{i2}** are variables corresponding to input lines and **b_{i3}** is the output of the gate

CIRCUIT-SAT $\precsim$ SAT

- We need to map every circuit C to a formula F such that
  - *C is satisfiable* iff *F is satisfiable*
- Given a circuit C:
  - assume each input line is marked with a variable a_j
  - mark each output line of a gate with a variable $b_{i,}$ i>0
  - mark the final output line as b_0
- Construct a formula F:
  - **(AND$_i$ g_i) AND b_0** where each g_i is a formula for gate i:
    - **b_{i1} op b_{i2} <--> b_{i3}** if the gate is binary
    - **b_{i1} <--> b_{i3}** if the gate is unary where
      - **b_{i1}** and **b_{i2}** are variables corresponding to input lines
      - **b_{i3}** corresponds to the output line of the gate, and
      - **op** is the operator of the gate.

NP-Completeness Via Reductions – SAT [2]

- Proof [contd.]: CIRCUIT-SAT $\precsim$ SAT
  - (*see previous slide*)
    - we have a mapping of every combinational Boolean circuit C to some Boolean formula F.
  - Claims:
    - F is satisfiable iff C is satisfiable [Why?]
      - i.e. our *mapping is a reduction*.
    - Length of F is linearly proportional to that of C.
      - i.e. our *mapping is a polynomial time reduction*.