COM3105 Advanced Algorithms

Homework 1

Joffray Hargreaves

Questions

1 Question 1

Find an equivalent CNF representation for the following formula:

$$\bigvee_{1 \leq i < j \leq n} (x_i \wedge x_j)$$

Try to make it as short as possible. Prove that your CNF has the smallest possible size. (3 points)

Answer:

Equivalent CNF

The formula is true when at least two of the variables $x_1, \ldots, x_n$ are true.
An equivalent CNF:

$$\bigwedge_{i=1}^{n} \Big(\bigvee_{j \neq i} x_j \Big)$$

i.e. for each i, the clause $C_i = (x_1 \vee \cdots \vee x_{i-1} \vee x_{i+1} \vee \cdots \vee x_n)$.

Proof of Minimised Size

To prove that this CNF is minimum size we need to know the definition of size:

$$\text{size} = \text{number of clauses} \times \text{number of variables in each clause}$$

in the CNF provided this is equivalent to:

$$\text{size} = n \times (n-1) = n(n-1)$$

Therefore, we need to prove that the number of clauses cannot be less than n and the number of variables in each clause cannot be less than $n-1$.

1. Number of Clauses:

- Each clause C_i corresponds to a variable x_i where:

$$C_i = (x_1 \vee \cdots \vee x_{i-1} \vee x_{i+1} \vee \cdots \vee x_n)$$

- If we assume an assignment $x_p = 1$, and all other $x_i = 0$ for $i \neq p$, then all clauses except C_p are satisfied.
- Therefore, the overall CNF is not satisfied, which is correct, since only one variable is true.
- Now, if we assume that we remove a clause, it could be clause C_p.
- In this case, the assignment $x_p = 1$ and all other $x_i = 0$ for $i \neq p$ would satisfy all remaining clauses, making the CNF true.
- This is incorrect since only one variable is true, so we have a contradiction.
- Therefore, we cannot remove any clauses.
- Since there are n variables, there must be at least n clauses.

2. Number of Variables in Each Clause:

- Each x_i appears in all the clauses but 1 (the clause C_i).
- Assuming a satisfying assignment to our CNF is $x_p = 1$ and $x_q = 1$ and all other $x_i = 0$ for $p \neq q$, then all clauses, including C_p and C_q, are satisfied.
- Now if we assume that there is another clause (as well as C_p) where x_p is missing, call this C_r, then we would have a clause with $(n-1)-1$ variables.
- So, C_r contains all variables, except for x_p and some other variable x_r.
- Using this assumption, there exists a scenario where $x_r = x_q$, hence $C_r = C_q$.
- In this scenario there would be no true variable in C_r, since neither or the true variables (x_p or x_q) are present, making the CNF false.
- Therefore, each clause must contain at least $n-1$ variables.

As we have proved that both the number of clauses and the number of variables in each clause cannot be less than their respective lower bounds, we have shown that the proposed CNF has the minimum size.

---

2 Question 2

Let F be an unsatisfiable formula. Prove that there is a derivation of the empty formula $\perp$ from F using resolution. (3 points)

Answer:
Let F be an unsatisfiable formula in CNF form, i.e. $F = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ where each C_i is a clause.
As there is no satisfying assignment for F, we can conserve unsatisfiability by also expressing it as:
Note: I don't believe this is strictly allowed, but my idea was to conserve unsatisfiability.

$$F \implies x \wedge \neg x$$

which is a simple unsatisfiable formula.
Defining the resolution rule:

$$Resolution = (C \vee x) \wedge (\neg x \vee D) \implies (C \vee D)$$

Using the resolution rule on the simplified F:

$$Res(F) \implies Res((x) \wedge (\neg x)) \implies \emptyset = \perp$$

Therefore, we have proved that we can derive the empty formula $\perp$ from the unsatisfiable formula F using resolution.

---

3 Question 3

Recall the idea of using a maximal set of pairwise disjoint clauses to get a non-trivial 3-SAT algorithm from 2-SAT. Extend this idea of k-CNFs and give a k-SAT algorithm running in time $O(c^n)$ for some c. What is the value of c? (4 points)

Answer:
Remembering the non-trivial 3-SAT algorithm discussed in class, the disjoint collection of variables is defined by the set T.
Here we explain an algorithm to reduce a k-CNF to a $(k-1)$-CNF, by using the set T of disjoint clauses:

- We create the set T.
- $|T| \leq n/k$, as we take one variable from each clause, in the worst case.
- We set the variables in T to all possible assignments.
- In the worst case, we may have some clauses where all of the clauses' variables are in T.
- In this case, there are a total of $2^k - 1$ possible assignments, as we can ignore the assignment where all variables in a clause are false.
- We exponentiate this by $|T|$ to get the total number of assignments to check, which is $(2^k - 1)^{|T|}$.

Therefore, the runtime of the algorithm is:

$$(2^k - 1)^{n/k}$$

We can repeat this process until we reach 2-CNF, which we can solve in polynomial time.
As the longest runtime is the first reduction from k-CNF to $(k-1)$-CNF, the runtime of the overall algorithm is:

$$O(((2^k - 1)^{1/k})^n)$$

Hence:

$$c = (2^k - 1)^{1/k}$$

Proof that this is smaller than 2 for $k \geq 2$:

- Assume we don't include -1 in the numerator.
- We get the equation $c = (2^k)^{1/k} = 2$.
- As this is always equivalent to 2, including the -1 in the numerator will always make c smaller than 2 for all $k \geq 2$.
