Artificial Intelligence
Homework no. 3

Alireza Rostami
Student Number: 9832090

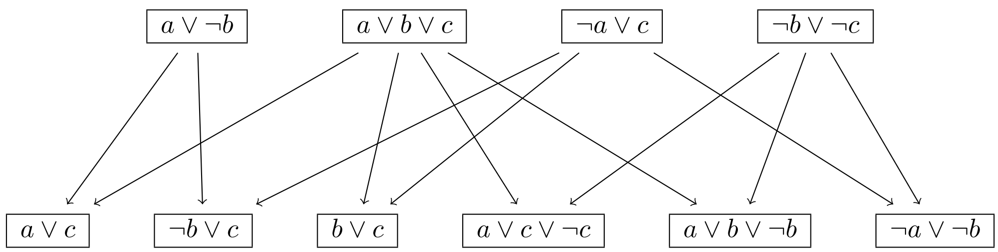
Find the LaTeX code of this masterpiece at my Github @ github.com/WellOfSorrows.

Question 1

1. *a*

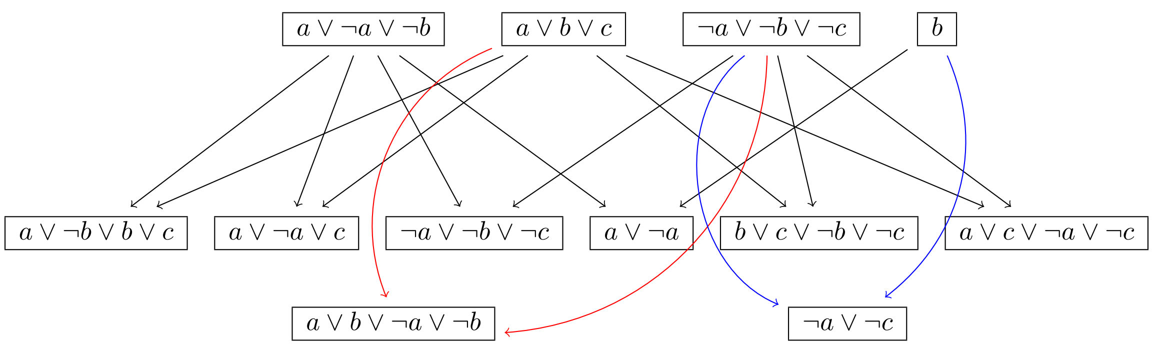
1. *a. i*

We would get:



1. *a. ii*

We would get:



1. *b*

To write the CNF form of the first and the third sentence, we eliminate $\Rightarrow$.

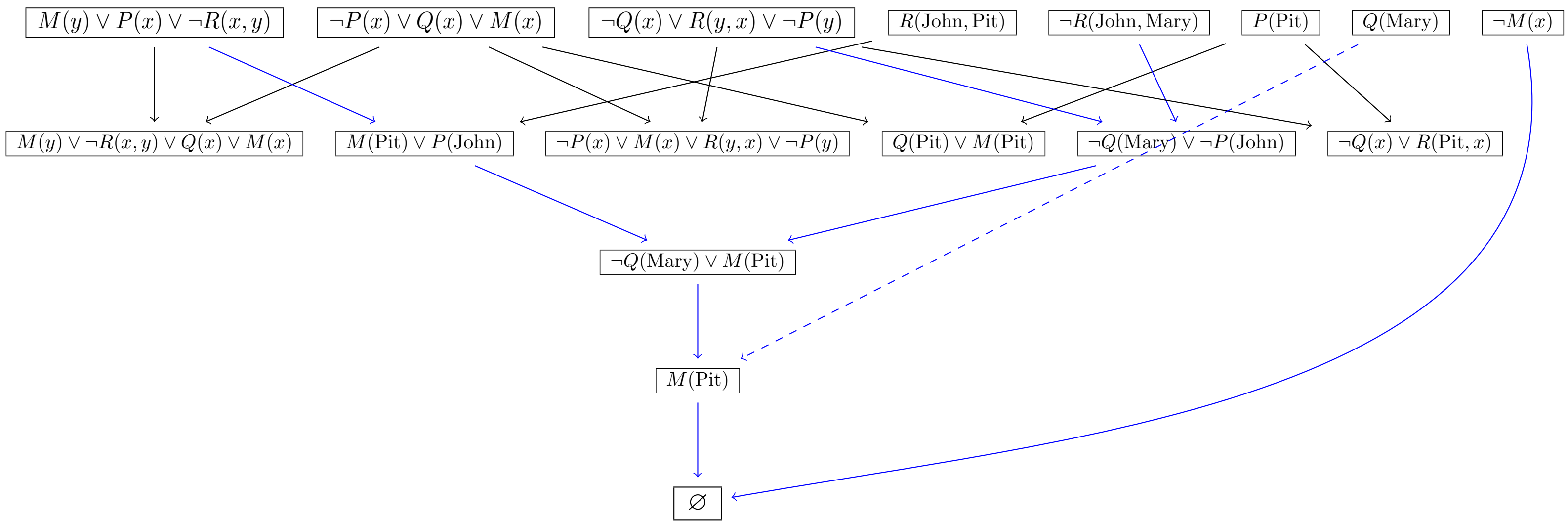
$$\begin{aligned} & P(x) \rightarrow Q(x) \vee M(x) \\ \text{is equivalent to } & \neg P(x) \vee Q(x) \vee M(x) \end{aligned}$$

$$\begin{aligned} & \neg M(y) \rightarrow \neg(\neg P(x) \wedge R(x,y)) \\ \text{is equivalent to } & \neg M(y) \rightarrow P(x) \vee \neg R(x,y) \\ \text{is equivalent to } & M(y) \vee P(x) \vee \neg R(x,y) \end{aligned}$$

Thus, our CNF-normal KB is:

$$\begin{aligned} & M(y) \vee P(x) \vee \neg R(x,y) \\ & \neg P(x) \vee Q(x) \vee M(x) \\ & \neg Q(x) \vee R(y,x) \vee \neg P(y) \\ & R(\text{John}, \text{Pit}) \\ & \neg R(\text{John}, \text{Mary}) \\ & P(\text{Pit}) \\ & Q(\text{Mary}) \end{aligned}$$

To use resolution algorithm, we must show $KB \wedge \neg M(x)$ leads to contradiction.



We reached contradiction; thus, the value of $M(x)$ is $True$.

Question 2

2. *a*

2. *a.* *i*

$$Mother(Mary, Charles) \Rightarrow \exists x \; Loves(x, Charles)$$

2. *a.* *ii*

$$\exists d \forall c \exists b \;\; Dog(d) \wedge Cat(c) \wedge Bird(b) \wedge Eat(c, b) \;\Rightarrow\; Hate(d, c)$$

2. *b*

2. *b.* *i*

If a number is less than zero, then the cube of the number is also less than zero.

2. *b.* *ii*

There is some student that would teach every student who did not learn.

2. *b.* *iii*

Every student has at least two distinct friends.

Question 3

3. *a*

One version of such assertion is:

$$KB = (\neg X_{1,2} \wedge X_{2,2} \wedge X_{2,1}) \vee (X_{1,2} \wedge \neg X_{2,2} \wedge X_{2,1}) \vee (X_{1,2} \wedge X_{2,2} \wedge \neg X_{2,1})$$

To translate the above DNF into CNF normal form, we can do the following. We know CNF normal form is equivalent to *product-of-sums(POS)* form and DNF to *sum-of-products(SOP)* form. Using these equivalences, we translate DNF form of our KB into SOP form:

$$KB \equiv \overline{X_{1,2}} \cdot X_{2,2} \cdot X_{2,1} + X_{1,2} \cdot \overline{X_{2,2}} \cdot X_{2,1} + X_{1,2} \cdot X_{2,2} \cdot \overline{X_{2,1}}$$

Using $X_{1,2}$ as MSB, and considering we have three variables, we would get:

$$KB \equiv \sum minterms(3, 5, 6) \equiv \prod Maxterms(0, 1, 2, 4, 7)$$

Using the POS, we write the CNF form of our KB as:

$$\begin{aligned} KB \equiv\, & (\neg X_{1,2} \vee \neg X_{2,2} \vee \neg X_{2,1}) \\ & \wedge (\neg X_{1,2} \vee \neg X_{2,2} \vee X_{2,1}) \\ & \wedge (\neg X_{1,2} \vee X_{2,2} \vee \neg X_{2,1}) \\ & \wedge (X_{1,2} \vee \neg X_{2,2} \vee \neg X_{2,1}) \\ & \wedge (X_{1,2} \vee X_{2,2} \vee X_{2,1}) \end{aligned}$$

3. *b*

The DNF form of such assertion is easy; since we have k bombs in n neighbors, thus, assuming neighbors of a each square are named $Y_1, Y_2, \ldots, Y_n$, in each sentence S_i, k variables are $True$ and other are $False$. Since we have $t = \binom{n}{k}$ choices, we have t sentences. Thus:

$$\bigvee_{i=1}^{t} S_i; \quad \text{where } t = \binom{n}{k}$$

To find the *CNF* of *KB*, we must find all the minterms of our *KB*.
To give an example, suppose S_1 is the following:

$$S_1 \equiv \overline{Y_n} \cdot \overline{Y_{n-1}} \ \ldots \ \overline{Y_{k+1}} \cdot Y_k \ \ldots \ Y_1$$

The sentence S_1 corresponds to the following binary sequence:

$$1 + 2 + \cdots + 2^{k-1} = \sum_{i=0}^{k-1} 2^i = 2^k - 1$$

Thus, S_1 corresponds to the minterm number $2^k - 1$.
Doing such process for each S_i, we find all the minterms. Suppose m_i corresponds to S_i. Thus:

$$KB \equiv \sum minterms(m_1, \ldots, m_t) \equiv \prod Maxterms(M_1, \cdots, M_{2^n - t})$$

Letting M_i represent the CNF-normal sentence S'_i and t' as $2^n - t$, we arrive at:

$$\bigwedge_{i=1}^{t'} S'_i; \quad \text{where } t' = 2^n - \binom{n}{k}$$

3. *c*

For each cell we probe, the game gives a number n. We must construct a sentence with $\binom{n}{8}$ disjunct (supposing each cell has 8 neighbors, which is not always the case.) Conjoining all sentences together, we can use *DPLL* in order to ascertain whether this sentence entails $X_{i,j}$ for any pair of i, j we are concerned about.

3. *d*

To modify our model in order for it to fit the global constraint, we construct a $\binom{M}{N}$ disjuncts, where each has N variables. Since

$$\binom{M}{N} = \frac{M}{M!(M-N)!}$$

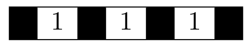
a Minesweeper game of 100 cells and 20 mines would contain 10^{39} disjuncts which is a bizarrely large number that no computer can process. To overcome such obstacle, we can append the global constraint to the DPLL itself. To achieve such goal, we can add min and max to DPLL algorithm, indicating the minimum and the maximum number of unassigned symbols that must be $True$ in the given model respectively. If there are no constraints, then we set $min = 0$ and $max = N$. Since Minesweeper is constrained, we cannot use such values. Thus, we set both $min = M$ and $max = M$ initially in the DPLL; and each time we call DPLL, we update min and max by subtracting one when assigning a $True$ value to a symbol. Within such process, if min becomes less than the number of remaining symbols required to be $True$, since this is a contradiction, then we will exit the model immediately; or, if the max becomes less than 0, which is also a contradiction, we end the model.

3. *e*

There is no conclusions being invalidated via appending this ability to DPLL algorithm and taking the global constraint into consideration by using it.

3. *f*

Consider the following configuration.



There are two possible models: either mines are under every even-numbered black cell, or under every odd-numbered black cell. Making a probe at either end will entail whether cells at the far end are empty or contain mines.

Question 4

4. *a*

$$\exists d\ Parent(Joan, d) \land Female(d)$$

4. *b*

$$\begin{aligned} &\exists d\ Female(d) \land Parent(Joan, d) \land (\forall e\ (Female(e) \land e \neq d) \Rightarrow \neg Parent(Joan, e)) \\ \equiv\ &\exists^1 d\ Parent(Joan, d) \land Female(d) \end{aligned}$$

4. *c*

$$\exists^1 d\ Female(d) \land Parent(Joan, d) \land (\forall e\ \neg Female(e) \Rightarrow \neg Parent(Joan, e))$$

4. *d*

$$\exists^1 c\ Parent(Joan, c) \land Parent(Kevin, c)$$

4. *e*

$$\exists c\ Parent(Joan, c) \land Parent(Kevin, c) \land (\forall p\ (p \neq Kevin) \Rightarrow \nexists d\ (Parent(Joan, d) \land Parent(p, d))$$