Rui Ji
1000340918

Problem Set 1

1. $\oplus$ *introduction rules*

   Left $\frac{A,\Gamma,\to\Delta,B \quad B,\Gamma\to\Delta,A}{(A\oplus B),\Gamma\to\Delta}$

   Right $\frac{\Gamma\to\Delta,A,B \quad A,B,\Gamma\to\Delta}{\Gamma\to\Delta,(A\oplus B)}$

   Let see how we get those rules first.
   For left introduction, we know $A \oplus B \iff (A \wedge \neg B) \vee (B \wedge \neg A)$.

$$
\dfrac{\dfrac{(\neg\ left)\dfrac{(\wedge\ left)\dfrac{A,\Gamma\to\Delta,B}{A,\neg B,\Gamma\to\Delta}}{A\wedge\neg B,\Gamma\to\Delta} \quad \dfrac{\dfrac{B,\Gamma\to\Delta,A}{B,\neg A,\Gamma\to\Delta}(\neg\ left)}{B\wedge\neg A,\Gamma\to\Delta}(\wedge\ left)}{(A\wedge\neg B)\vee(B\wedge\neg A),\Gamma\to\Delta}(\vee\ left)}{(A\oplus B),\Gamma\to\Delta}
$$

   For right introduction, we know $A \oplus B \iff (A \vee B) \wedge \neg(A \wedge B)$.

$$
\dfrac{\dfrac{(\vee\ right)\dfrac{\dfrac{\Gamma\to\Delta,A,B}{\Gamma\to\Delta,A,B}}{\Gamma\to\Delta,A\vee B} \quad \dfrac{\dfrac{A,B,\Gamma\to\Delta}{A\wedge B,\Gamma\to\Delta}(\wedge\ left)}{\Gamma\to\Delta,\neg(A\wedge B)}(\neg\ right)}{\Gamma\to\Delta,(A\vee B)\wedge\neg(A\wedge B)}(\wedge\ right)}{\Gamma\to\Delta,(A\oplus B)}
$$

   Proof by truth table:

   right introduction

$\square$

2. Suppose the size of the set S of propositional clauses is n which means there are n clauses in S, from the question we know that every clause can have at most 2 literals. Then, there are at most $2n$ different literals.

   We know that each execution of the general steps results either adding a new distinct literal to the stack, or adding a new clause to the list S'. Since every clause can have at most 2 literals, then every time we add a new a resultant clause R to list S', the resultant clause R can have at most 1 literal; hence, we have at most $2n+1$ options of adding resultant clause.

   Since we only pop out literal after adding a resultant clause to S', then in the worst case, every time the procedure add a new resultant clause to list S', it has to add $2n$ literals to the stack first and after adding that new a resultant clause to list S', the procedure pop out all literals. Also, in the worst case the procedure added all possible resultant clauses to list S', before it generates a satisfying assignment for S or a resolution refutation. Since in the worst case, it takes $\mathcal{O}(n)$ to add a resultant clause to S'; therefore, the worst case run time is in order $\mathcal{O}(n^2)$.

   The procedure halts in time polynomial in the size of input S.

$\square$

3. (a)

   According to the definition of $3-colouring$ we know that G_n has a $3-colouring$ iff every node only has one colour. Also, every two nodes sharing a same edge have different colours. Then based on those rules, we get our formula as follows.
   propositional formula :

$$A_n = \bigwedge_{i \in V_n} (\neg R_i \wedge \neg B_i \wedge Y_i) \vee (\neg R_i \wedge B_i \wedge \neg Y_i) \vee (R_i \wedge \neg B_i \wedge \neg Y_i) \wedge \bigwedge_{(x,y)\in E_n} (R_x \wedge \neg R_y) \vee (B_x \wedge \neg B_y) \vee (Y_x \wedge \neg Y_y)$$

   (b)

   First, let's define a set of propositional formulas Φ as follow:

$$\Phi = \{ \bigwedge_{i \in V} (\neg R_i \wedge \neg B_i \wedge Y_i) \vee (\neg R_i \wedge B_i \wedge \neg Y_i) \vee (R_i \wedge \neg B_i \wedge \neg Y_i) \} \cup \bigwedge_{\substack{x,y \in V \wedge \\ (x,y)\in E}} (R_x \wedge \neg R_y) \vee (B_x \wedge \neg B_y) \vee (Y_x \wedge \neg Y_y)$$

   The size of Φ is based on V, also Φ is a infinite set iff graph $G(V,E)$ is a infinte graph.

   Let $V_n = \{0, 1, ...n-1\}$ be a finite subset of V, $E_n = E - (V \setminus V_n)$ be the corresponding edge set, then $G_n(V_n, E_n)$ is a induced subgraph of G on the vertex set V_n.

   Based on the question, we know that G_n has a $3-colouring$, then let $c : V_n \to \{R, B, Y\}$ be a $3-colouring$ of G_n

   Then, clearly that

$$\Phi_n = \{ \bigwedge_{i \in V_n} (\neg R_i \wedge \neg B_i \wedge Y_i) \vee (\neg R_i \wedge B_i \wedge \neg Y_i) \vee (R_i \wedge \neg B_i \wedge \neg Y_i) \} \cup \bigwedge_{\substack{(x,y)\in E_n \\ \wedge x \in V_n \wedge y \in V_n}} (R_x \wedge \neg R_y) \vee (B_x \wedge \neg B_y) \vee (Y_x \wedge \neg Y_y)$$

   is satisfied by c.

   Notice, it's clear that Φ is satisfiable iff graph $G(V,E)$ is $3-colourable$, also every finite subset of Φ is associated with a induced subgraph of G. By *Propositional Compactness Theorem* we know Φ is satisfiable; hence, G has a $3-colouring$.

$\square$

4. Think R as negation, that is $R(x,y)$ means $x = \neg y$. Then, based on this general thought, let's build our sentence.
   Define sentence A as follow,

   (a) $\forall x, y : R(x,y) \Rightarrow R(y,x)$
   (b) $\forall x, y, z : R(y,x) \wedge R(z,x) \Rightarrow y = z$
   (c) $\forall x, \exists y : R(y,x) \wedge \neg(y = x)$

   Then, if M is a model for A, for any element a in M based on (c) there exist a pair (a,b) such that $R(b,a)$ and $a \neq b$; hence, every element in M has a pair. Therefore, we can treat M as a set of pairs and every pair has 2 distinct elements.
   For any two pairs (a,b) and (c,d), which means $R(b,a)$ and $R(d,c)$,

   If $a = c$, we have $R(b,a) = R(b,c)$, then by (b) we have $b = d$, which means $(a,b) = (c,d)$.

   If $a = d$, we have $R(b,a) = R(b,d) = R(d,b)$ then we have $b = c$, which means $(a,b) = (c,d)$.

   Same for $b = c$ or $b = d$, we can also get $(a,b) = (c,d)$.

   On the other hand, if $(a,b) \neq (c,d)$, then $a \neq b \neq c \neq d$. In others words, every element in any two distinct pairs are distinct; hence, M has even number of element.

   $\square$