Two hours

Examination definition sheet is available on pages 6 to 10 of this examination paper.

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Logical Reasoning and Applications

Date: Thursday 19th January 2012
Time: 09:45 - 11:45

Please answer any THREE questions from the FOUR questions provided

For full marks your answers should be concise as well as accurate.
Marks will be awarded for reasoning and method as well as being correct.

This is a CLOSED book examination

The use of electronic calculators is NOT permitted
1. (Orderings, propositional resolution, LTL)
   
a) Give an example of: i) a well-founded ordering; ii) a non well-founded ordering. 
   Explain your answers. (2 marks)

b) Let $\succ^3_{\text{lex}}$ be the lexicographic ordering on triples based on $a \succ b \succ c \succ d$.
   
i) Order the following triples in $\succ^3_{\text{lex}}$:
      - $\langle c,d,a \rangle$
      - $\langle a,d,d \rangle$
      - $\langle a,a,c \rangle$
   
ii) Is $\succ^3_{\text{lex}}$ a total ordering?
   
iii) How many triples are there, which are smaller than $\langle d,c,b \rangle$?
   
Justify your answer (4 marks)

c) Describe how equivalence of two propositional formulas $F$ and $G$ can be checked using the resolution calculus. (3 marks)

d) Check whether the following set of clauses is satisfiable using the propositional resolution calculus (BRR). Apply subsumption elimination (SE) and tautology elimination (TE) whenever possible. Use the linear notation to write inferences.

   1. $\neg q \lor r$
   2. $\neg p \lor \neg q \lor \neg r$
   3. $q \lor \neg r$
   4. $p \lor \neg q \lor \neg r$
   5. $q \lor r$

   (8 marks)

e) Draw computation paths which satisfy the following LTL formulas:

   i) $\lozenge(\neg q \land p)$
   
   ii) $\lozenge \Box p$
   
   iii) $\Box(p \rightarrow \lozenge \neg p) \land \Box(\neg p \rightarrow \lozenge p)$

   (3 marks)
2. (CNF transformation, formalisation, unit propagation, DPLL)

a) What are two main differences between structural CNF transformation and syntactic CNF transformation (based on equivalence rules)?

(4 marks)

b) i) Apply the DPLL algorithm to the following set of clauses. Start with a decision literal \( p^d \) and apply unit propagation eagerly. Write whether this set is satisfiable or not.

\[
\begin{array}{l}
\neg p \lor s \lor q \\
p \lor q \lor \neg s \\
s \lor \neg q \\
\neg p \lor \neg s \\
s \lor p \\
\end{array}
\]

(7 marks)

ii) Consider a set of clauses over \( n \) variables. At most how many steps of unit propagation are possible, without applications of other rules of the DPLL algorithm.

(2 marks)

iii) For which fragment of propositional logic is unit propagation alone a decision procedure?

(1 mark)

iv) What are two main optimisations of the DPLL algorithm. Briefly describe one them.

(3 marks)

c) For the set of propositional variables \( \{p_1, p_2, p_3\} \) express the following properties in propositional logic:

i) at least one variable is true.

ii) at most one variable is true.

iii) exactly two variables are true.

(3 marks)
3. (Transformation to clausal form, model construction, orderings, redundancy)

(a) The transformation of any first-order formula to a set of clauses we discussed in class uses this sequence of steps.

1. Transform to PNF
2. Skolemise
3. Transform to CNF
4. Clausify

i. Give a brief explanation of each step. In each case use a small example in your explanation and briefly say what your example illustrates.

ii. Give an example of a formula where first performing transformation to CNF and then transformation to PNF, Skolemisation and clausification will not work.

(b) Let \( N \) be the following set of ground clauses.

1. \( A_4 \lor A_4 \)
2. \( A_5 \lor \neg A_2 \lor A_4 \lor \neg A_2 \)
3. \( A_2 \)
4. \( \neg A_3 \lor \neg A_1 \lor \neg A_1 \)
5. \( A_1 \lor A_1 \lor \neg A_2 \lor \neg A_1 \)
6. \( \neg A_2 \lor \neg A_5 \)

i. Let the ordering on atoms be defined by \( A_5 \succ A_4 \succ A_3 \succ A_2 \succ A_1 \). Sort the clauses in \( N \) with respect to \( \succ_C \).

ii. Compute the candidate model \( I_N^{\succ} \) for \( N \) as described in lectures. Is it a model of \( N \)?

iii. Which clauses in \( N \) are redundant with respect to \( \succ_C \)? Justify why these clauses are redundant in \( N \).
4. (Herbrand interpretations, unification, ordered resolution)

(a) For each set of clauses $N$, write down a Herbrand interpretation $I$ such that $I \models N$. Justify your answer in each case. (6 marks)

i. $N = \{1. \ p(a), \\
2. \ \neg p(a) \lor q(b)\}$

ii. $N = \{1. \ p(a), \\
2. \ \neg p(x) \lor p(b)\}$

iii. $N = \{1. \ p(a), \\
2. \ \neg p(f(x)) \lor p(x)\}$

iv. $N = \{1. \ p(a), \\
2. \ \neg p(x) \lor p(f(x))\}$

(b) Apply the basic unification algorithm based on the $\Rightarrow_U$-rules to the following problem in order to determine whether it is unifiable. If yes, give a most general unifier. (6 marks)

\[\{P(x, f(y)) \Rightarrow P(x, f(x)), P(x, f(x)) \Rightarrow P(u, v)\}\]

$x, y, u, v$ denote variables.

(c) Let $\succ$ be a total and well-founded ordering on ground atoms such that, if the atom $A$ contains more symbols than $B$, then $A \succ B$. Let $N$ be the following set of clauses.

1. $\neg P(x)$
2. $P(y) \lor Q(y, b)$
3. $\neg Q(f(f(z)), b) \lor \neg Q(a, b) \lor P(z)$

i. For each clause determine which literals are strictly maximal in it. Give justifications in each case. (3 marks)

ii. Derive $\bot$ from $N$ by using the rules of $Res_{\succ}$, where $\succ$ is defined as above and $S$ is the empty selection function for all clauses. Indicate the maximal literals in every clause in the derivation. Justify each step in your derivation. (5 marks)
Examination definition sheet

**Structural transformation.** Lemma: $F[G]$ is satisfiable $\iff F[n_G] \land (n>G) \iff G$ is satisfiable, provided $n_G$ is a (fresh) propositional variable not occurring in $F[G]$. $n_G$ can be seen as a name for $G$. **Structural CNF Transformation**: introduce names recursively for every non-literal subformula in the original formula.

**Lexicographic Combination.** Let $(X_1, \succ_1), (X_2, \succ_2)$ be two orderings. **Lexicographic combination** of $(X_1, \succ_1), (X_2, \succ_2)$ is an ordering: $\succ_{\text{lex}} = (\succ_1, \succ_2)_{\text{lex}}$ on $X_1 \times X_2$ such that $\langle x_1, x_2 \rangle \succ_{\text{lex}} \langle y_1, y_2 \rangle$ if and only if (i) $x_1 \succ_1 y_1$, or else (ii) $x_1 = y_1$ and $x_2 \succ_2 y_2$. We combine an ordering $(X, \succ)$ with itself $n$ times obtaining $\succ^n_{\text{lex}}$. That is $\langle x_1, \ldots, x_n \rangle \succ^n_{\text{lex}} \langle y_1, \ldots, y_n \rangle$ if and only if $x_1 = y_1, \ldots, x_{i-1} = y_{i-1}$ and $x_i \succ y_i$ for some $i$, $1 \leq i \leq n$.

**DPLL rules.**

- **Unit Propagate (UP):**
  \[
  U \parallel S, \quad \Rightarrow_{\text{UP}} \quad U\ell \parallel S
  \]
  if \[
  \begin{cases}
  I_U \models \neg C, \text{ for } C \lor \ell \in S \\
  \ell \text{ is undefined in } I_U
  \end{cases}
  \]

- **Decide (D):**
  \[
  U \parallel S \Rightarrow_{D} \quad U\ell^d \parallel S
  \]
  if \[
  \begin{cases}
  \ell \text{ is undefined in } I_U
  \end{cases}
  \]

- **Backtrack (B):**
  \[
  U\ell^dV \parallel S \Rightarrow_{B} \quad U\ell \parallel S
  \]
  if \[
  \begin{cases}
  I_{U\ell^dV} \models \neg C, \text{ for } C \in S, \\
  V \text{ contains no decision literals}
  \end{cases}
  \]

- **Unsat (⊥):**
  \[
  U \parallel S \Rightarrow_{\perp} \quad \perp \parallel S
  \]
  if \[
  \begin{cases}
  I_U \models \neg C, \text{ for } C \in S, \\
  U \text{ contains no decision literals}
  \end{cases}
  \]

- **Backjumping (BJ):**
  \[
  U\ell^dV \parallel S \Rightarrow_{BJ} \quad Ue \parallel S
  \]
  if \[
  \begin{cases}
  I_{U\ell^dV} \models \neg C, \text{ for } C \in S, \\
  U \land S \models e, \\
  e \text{ is undefined in } U
  \end{cases}
  \]

- **Lemma Learning (LL):**
  \[
  U \parallel S \Rightarrow_{LL} \quad U \parallel S \cup \{C\}
  \]
  if \[
  \begin{cases}
  S \models C \\
  \text{C is set-reduced}
  \end{cases}
  \]
LTL semantics.

Let $\pi = s_0, s_1, s_2 \ldots$ be a sequence of states and $F$ be an LTL formula. $F$ is true on $\pi$, denoted by $\pi \models F$, defined by induction on $F$ as follows. For all $i = 0, 1, \ldots$ denote by $\pi_i$ the sequence of states $s_i, s_{i+1}, s_{i+2} \ldots$ (note that $\pi_0 = \pi$).

- $\pi \models \top$ and $\pi \not\models \bot$.
- $\pi \models x = v$ if $s_0 \models x = v$.
- $\pi \models F_1 \land \ldots \land F_n$ if for all $j = 1, \ldots, n$ we have $\pi \models F_j$.
- $\pi \models F_1 \lor \ldots \lor F_n$ if for some $j = 1, \ldots, n$ we have $\pi \models F_j$.
- $\pi \models \neg F$ if $\pi \not\models F$.
- $\pi \models F \rightarrow G$ if either $\pi \not\models F$ or $\pi \models G$.
- $\pi \models F \leftrightarrow G$ if either both $\pi \not\models F$ and $\pi \not\models G$ or both $\pi \models F$ and $\pi \models G$.
- $\pi \models \Diamond F$ if for some $i = 0, 1, \ldots$ we have $\pi_i \models F$.
- $\pi \models \Box F$ if for all $i = 0, 1, \ldots$ we have $\pi_i \models F$.
- $\pi \models F \mathcal{U} G$ if for some $k = 0, 1, \ldots$ we have $\pi_k \models G$ and $\pi_0 \models F, \ldots, \pi_{k-1} \models F$.

Two LTL formulas $F$ and $G$ are called equivalent, denoted $F \equiv G$, if for every path $\pi$ we have $\pi \models F$ if and only if $\pi \models G$. 
Herbrand models. The Herbrand universe (over $\Sigma$), denoted $T_\Sigma$, is the set of all ground terms over $\Sigma$.

A Herbrand interpretation (over $\Sigma$), denoted $I$, is a set of ground atoms over $\Sigma$.

Truth in $I$ of ground formulae is defined inductively by:

\[
\begin{align*}
    I \models \top & \quad I \not\models \bot \\
    I \models A & \iff A \in I, \text{ for any ground atom } A \\
    I \models \neg F & \iff I \not\models F \\
    I \models F \land G & \iff I \models F \text{ and } I \models G \\
    I \models F \lor G & \iff I \models F \text{ or } I \models G
\end{align*}
\]

Truth in $I$ of any quantifier-free formula $F$ with free variables $x_1, \ldots, x_n$ is defined by:

\[
I \models F(x_1, \ldots, x_n) \iff I \models F(t_1, \ldots, t_n), \text{ for every } t_i \in T_\Sigma
\]

Truth in $I$ of any set $N$ of clauses is defined by:

\[
I \models N \iff I \models C, \text{ for each } C \in N
\]

Construction of candidate models. Let $N, \succ$ be given.

For all ground clauses $C$ over the given signature, the sets $I_C$ and $\Delta_C$ are inductively defined with respect to the clause ordering $\succ$ by:

\[
\begin{align*}
    I_C & := \bigcup_{D \succ C} \Delta_D \\
    \Delta_C & := \begin{cases} 
        \{A\}, & \text{if } C \in N, \ C = C' \lor A, \ A \succ C' \text{ and } I_C \not\models C \\
        \emptyset, & \text{otherwise}
    \end{cases}
\end{align*}
\]

We say that $C$ produces $A$, if $\Delta_C = \{A\}$.

The candidate model for $N$ (wrt. $\succ$) is given as

\[
I_N^- := \bigcup_{C \in N} \Delta_C.
\]

We also simply write $I_N$, or $I$, for $I_N^-$, if $\succ$ is either irrelevant or known from the context.
**Orderings.** Let \((X, \succ)\) be an ordering. The *multi-set extension* \(\succ_{\text{mul}}\) of \(\succ\) to (finite) multi-sets over \(X\) is defined by

\[
S_1 \succ_{\text{mul}} S_2 \iff S_1 \neq S_2 \text{ and } \forall x \in S_2 \setminus S_1, \exists y \in S_1 \setminus S_2. y \succ x
\]

Suppose \(\succ\) is a total and well-founded ordering on ground atoms. \(\succ_L\) denotes the *ordering on ground literals* and is defined by:

\[
\neg A \succ_L \neg B, \text{ if } A \succ B
\]

\[
\neg A \succ_L A
\]

\(\succ_C\) denotes the *ordering on ground clauses* and is defined by the multi-set extension of \(\succ_L\), i.e. \(\succ_C = (\succ_L)_{\text{mul}}\).

**Maximal literals.** Let \(\succ\) be a total and well-founded ordering on ground atoms.

A ground literal \(L\) is called [strictly] maximal wrt. a ground clause \(C\) iff for all \(L'\) in \(C\): \(L \succeq L' [L \succ L']\).

A non-ground literal \(L\) is [strictly] maximal wrt. a (ground or non-ground) clause \(C\) iff there exists a ground substitution \(\sigma\) such that for all \(L'\) in \(C\): \(L\sigma \succeq L'\sigma [L\sigma \succ L'\sigma]\).

If \(L\) is [strictly] maximal wrt. a clause \(C\) then we say that \(L\) is [strictly] maximal in \(L \lor C\).

**The basic unification algorithm, based on rules.**

**Trivial:**

\[
t \doteq t, E \Rightarrow_U E
\]

**Decomposition:**

\[
f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n), E \Rightarrow_U s_1 \doteq t_1, \ldots, s_n \doteq t_n, E
\]

**Disagreement/Clash:**

\[
f(\ldots) \doteq g(\ldots), E \Rightarrow_U \bot
\]

**Substitution:**

\[
x \doteq t, E \Rightarrow_U x \doteq t, E \{x/t\}
\]

if \(x \in \text{var}(E), x \not\in \text{var}(t)\)

**Occur-check:**

\[
x \doteq t, E \Rightarrow_U \bot
\]

if \(x \in \text{var}(t), x \neq t\)

**Orientation:**

\[
t \doteq x, E \Rightarrow_U x \doteq t, E
\]

if \(t \not\in X\)
Ordered resolution with selection calculus $\text{Res}_S^\succ$. Let $\succ$ be an atom ordering and $S$ a selection function.

**Ordered resolution with selection rule:**

\[
\frac{C \lor A}{(C \lor D)\sigma} \quad \frac{\neg B \lor D}{(C \lor D)\sigma}
\]

provided $\sigma = \text{mgu}(A, B)$ and

(i) $A\sigma$ strictly maximal wrt. $C\sigma$;

(ii) nothing is selected in $C$ by $S$;

(iii) either $\neg B$ is selected, or else nothing is selected in $\neg B \lor D$ and $\neg B\sigma$ is maximal wrt. $D\sigma$.

**Ordered factoring rule:**

\[
\frac{C \lor A \lor B}{(C \lor A)\sigma}
\]

provided $\sigma = \text{mgu}(A, B)$ and

(i) $A\sigma$ is maximal wrt. $C\sigma$ and

(ii) nothing is selected in $C$.

**Redundancy.** Let $N$ be a set of ground clauses and $C$ a ground clause.

$C$ is called redundant wrt. $N$, if there exist $C_1, \ldots, C_n \in N$, $n \geq 0$, such that

(i) all $C_i \prec C$, and

(ii) $C_1, \ldots, C_n \models C$.

A general clause $C$ is called redundant wrt. $N$, if all ground instances $C\sigma$ of $C$ are redundant wrt. $G_2(N)$.

$N$ is called saturated up to redundancy (wrt. $\text{Res}_S^\succ$) iff every conclusion of an $\text{Res}_S^\succ$-inference with non-redundant clauses in $N$ is in $N$ or is redundant (i.e.

\[\text{Res}_S^\succ (N \setminus \text{Red}(N)) \subseteq N \cup \text{Red}(N),\]

where $\text{Red}(N)$ denotes the set of clauses redundant wrt. $N$).