Module Title: INF1-CL
Exam Diet (Dec/April/Aug): Dec 2015
Brief notes on answers:

1. (a) For each of the following propositional expressions, use a truth table to determine whether it is tautologous, contradictory, or contingent. [*2 marks*]

   i. $((\neg B \to \neg A) \to B) \to B$

A	B	$\neg A$	$\neg B$	$\neg B \to \neg A$	$(\neg B \to \neg A) \to B$	$((\neg B \to \neg A) \to B) \to B$
$\bot$	$\bot$	$\top$	$\top$	$\top$	$\bot$	$\top$
$\bot$	$\top$	$\top$	$\bot$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\top$	$\bot$	$\top$	$\bot$
$\top$	$\top$	$\bot$	$\bot$	$\top$	$\top$	$\top$

   contingent

   ii. $((A \vee B) \to A) \to (B \to A)$

A	B	$A \vee B$	$((A \vee B) \to A)$	$(B \to A)$	$((A \vee B) \to A) \to (B \to A)$
$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\top$
$\bot$	$\top$	$\top$	$\bot$	$\bot$	$\top$
$\top$	$\bot$	$\top$	$\top$	$\top$	$\top$
$\top$	$\top$	$\top$	$\top$	$\top$	$\top$

   tautology

   (b) This part concerns the 256 possible truth valuations of the following eight propositional letters A, B, C, D, E, F, G, H. For each of the following expressions, say how many of the 256 valuations satisfy the expression, and briefly explain your reasoning. For example, the expression $\neg D$ is satisfied by half of the valuations, that is 128 of the 256, since for each valuation that makes D true there is a matching valuation that makes D false.

   i. A *128*

   ii. $E \oplus F$ *128*

   iii. $B \vee \neg D$ *192*

   iv. $\neg A \to H$ *192*

   v.
$$(A \to B) \wedge (B \to C) \wedge (C \to D)$$
   $80 = 2^4 \times 5$

   vi.
$$(A \to C) \wedge (B \to C) \wedge (C \to E) \wedge (D \to E)$$
   $88 = 2^3 \times 11$ *(6 with D true; 5 with D false)*

vii.

$$(A \to B) \land (B \to C) \land (C \to D) \land (D \to E)$$
$$\land (E \to G) \land (F \to G) \land (G \to H) \land (H \to E)$$

6

viii.

$$(A \to (B \land C)) \land (B \to D) \land (D \to E) \land (E \to F) \land (F \to G) \land (G \to H)$$

15 (7 with C true; 8 with C false)

[*8 marks*]

2. You are given the following inference rules, due to Gentzen:

$$\frac{}{\Gamma, A, B \vdash \Delta, A}\ (I)$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ (\wedge L) \qquad \frac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \vee B, \Delta}\ (\vee R)$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta}\ (\vee L) \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\ (\wedge R)$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\ (\to L) \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ (\to R)$$

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, \neg A \vdash \Delta}\ (\neg L) \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta}\ (\neg R)$$

Here, A and B are propositional expressions, Γ, Δ are finite sets of expressions, and Γ, A is shorthand for $\Gamma \cup \{A\}$.

An entailment $\Gamma \vdash \Delta$ is *valid* iff every valuation that makes each expression in Γ true makes some expression in Δ true.

(a) i. Explain what it means to say that a valuation V is a **counterexample** for an entailment, $\Gamma \vdash \Delta$. *V makes every expression in Γ True, and every expression in Δ False.* [*2 marks*]

ii. Show that the rule $(\to L)$ has the property that a valuation V is a counterexample for the rule's conclusion iff it is a counterexample to at least one of its assumptions. *A counterexample to the conclusion makes everything in Γ True, and everything in Δ False. It also makes $A \to B$ true, so it either makes A False or B True. In the first case, the valuation is a counterexample to the first assumption; in the second case, to the second assumption. Contrariwise, a counterexample to either assumption makes all of Γ True and all of Δ False. For the first assumption, it makes A False; for the second, it makes B True. So, in either case $A \to B$ is true, and so we have a counterexample to the conclusion.* [*4 marks*]

(b) Use the Gentzen rules to show that [*4 marks*]

$$(Q \to P) \to Q \vdash Q$$

$$\cfrac{\cfrac{\cfrac{}{Q \vdash P, Q}\ I}{\vdash Q \to P, Q}\ \to R \qquad \cfrac{}{Q \vdash Q}\ I}{(Q \to P) \to Q \vdash Q}\ \to L$$

(c) Use the Gentzen rules to build an attempted proof of [*4 marks*]

$$(P \wedge Q) \to R, P \vee Q \vdash R$$

$$\dfrac{\dfrac{\dfrac{\overline{P \vdash P, R}\,(I) \qquad P \vdash Q, R}{P \vdash P \wedge Q, R}\,(\wedge R) \qquad \dfrac{Q \vdash P, R \qquad \overline{Q \vdash Q, R}\,(I)}{Q \vdash P \wedge Q, R}\,(\wedge R)}{P \vee Q \vdash P \wedge Q, R}\,(\vee L) \qquad \overline{P \vee Q, R \vdash R}\,(I)}{(P \wedge Q) \to R, P \vee Q \vdash R}\,(\to L)$$

(d) Derive a counter-example from your attempted proof. (Show your working, and briefly justify your answer in terms of the particular properties of this set of rules.) [*2 marks*]

Each rule has the property shown for $(\to L)$ in 2(a)ii. So a counter-example to either of the open assumptions in the attempted proof will be a counter-example to the conclusion. Either $P\bar{Q}\bar{R}$ or $Q\bar{P}\bar{Q}$ provides a counterexample.

(e) i. What does it mean to say a set of rules is **complete**? [*2 marks*]

It means that any valid sequent is derivable.

ii. Is this a complete set of rules? Briefly justify your answer. [*2 marks*]

Yes. The rules can be used to eliminate connectives until either all assumptions are discharged, or we are left with a set of open assumptions of the form $\Gamma \vdash \Delta$, where both Gamma and Δ are set of propositional letters, and $\Gamma \cap \Delta = \emptyset$. Any such assumption is refutable.

3. (a) Express each of the following expressions in clausal form. Write your answers in the table provided for Q3c. (Please order the literals in each clause alphabetically.) *[4 marks]*

   i. $P \to (Q \vee R)$

   ii. $(P \wedge Q) \to \neg R$

   iii. $\neg P \to (R \to Q)$

   iv. $(Q \wedge \neg P) \to R$

(b) Use resolution to determine whether there is a valuation satisfying the conjunction of these four expressions.

$$\begin{array}{r|c|}
\neg P, Q, R & Q, R, \neg R (= \top) \\
\neg P, \neg Q, \neg R & Q, \neg Q, R (= \top) \\
P, Q, \neg R & Q, \neg Q, \neg R (= \top) \\
P, \neg Q, R & \neg Q, \neg R, R (= \top)
\end{array}$$

Resolve first on P. All resolvants are trivial. We cannot derive the empty clause, so the conjunction is satisfiable. *[6 marks]*

(c) Using the table provided record the search for a satisfying valuation for these clauses, using the one watched literal algorithm.

Start with the empty valuation, watching the first literal in each clause. At each step of the search use one column of the table provided to record in the first 3 squares the truth values assigned to the atoms, P, Q, R, and in the remaining four, any changed position (1,2,3) of the watched literal for each of the four clauses. When you need to find a new literal to watch always choose the first one available (reading each expression left-to-right). Place an $\times$ in the watched literal square when no suitable literal is available.

P		$\top$	$\top$	$\top$	$\top$						
Q			$\top$	$\top$	$\top$						
R				$\top$	$\bot$						
(i) $\neg P, Q, R$	*1*	*2*									
(ii) $\neg P, \neg Q, \neg R$	*1*	*2*	*3*	$\times$							
(iii) $P, Q, \neg R$	*1*										
(iv) $P, \neg Q, R$	*1*										

Satisfying valuation: $P \wedge Q \wedge \neg R$ *[10 marks]*

(d) What is the invariant property of the watched literals that has to be maintained?
No watched literal is false. Every watched literal is either True or Undefined [*2 marks*]

(e) What must you do when it is not possible to maintain the invariant? *Backtrack or fail.*

(f) Sketch the search tree explored by this search. [*4 marks*]

(g) What invariant must be maintained for the 2-watched literals algorithm?
If one of the watched literals is set to False, then the other must be True [*2 marks*]

(h) What advantage is gained by watching two literals? *Unit clauses are propagated* [*2 marks*]

4. A cruise control system for a car communicates with the engine to maintain a set speed.

   The engine periodically provides **inputs**

   - c: correct
   - s: slow
   - f: fast

   to the controller and the controller matches these with appropriate **outputs**

   - a: acc
   - d: dec

   to indicate that the engine should accelerate or decelerate.

   Requirements In addition to this function, the controller must interact with the driver.

   - The driver should always be able to turn the system off.
   - The driver should be able to request the system to maintain the current speed.
   - If the driver brakes the system should go into standby mode.
   - The system should allow the driver to travel faster than the set speed by using the accelerator.

   The system has the following **additional inputs** to sense the user's actions and commands

onoff: A button that toggles the system on and off.

set: set the cruise speed to the current speed

brake: the brake has been pressed

accP: the accelerator has been pressed

accR: the accelerator has been released

resume: resume travelling at the set speed

States The system has 5 states:

Off: The system is not operational.

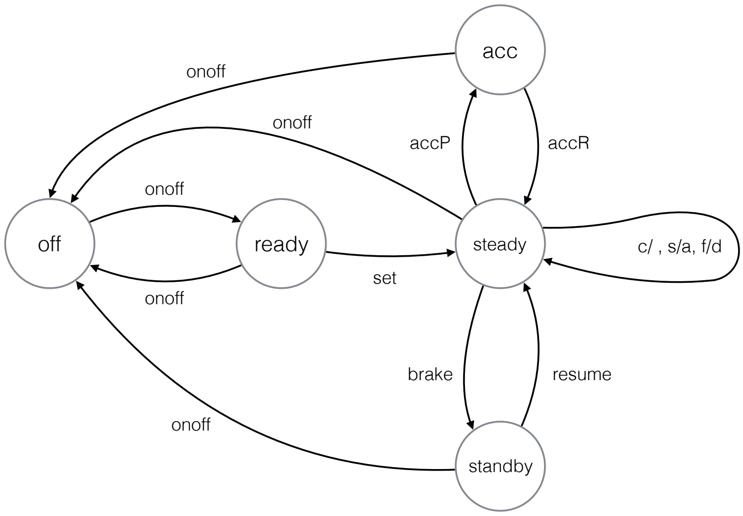
Ready: The system is switched on but so far no speed has been set to cruise at.

Steady: A cruise speed has been set and the system is maintaining it.

Standby: The system has a set cruise speed but at some time the driver used the brake and caused the system to wait until the resume button is pressed to bring the car back into cruise control.

Acc: The system has a set cruise speed but the accelerator has been pressed to override cruise control until the accelerator is released.

(a) Design a transducer-style finite state machine to model this system. [*8 marks*]



(b) Justify your design by explaining how it meets each of the requirements. If any of the requirements is ambiguous or unclear, explain how you have resolved the ambiguity. [*12 marks*]

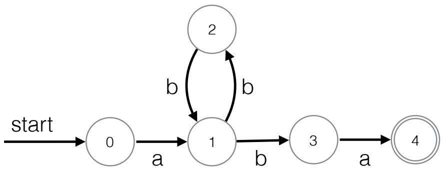
i. The driver should always be able to turn the system off. *There is an onoff transition to off from each other state.*

ii. The driver should be able to request the system to maintain the current speed. *This solution allows current speed to be set when system is on and neither accelerator nor brake is pressed.*

iii. If the driver brakes the system should go into standby mode. *If the accelerator has been pressed, it is unclear whether we should allow for brake before accR.*

iv. The system should allow the driver to travel faster than the set speed by using the accelerator. *When the accelerator is pressed the system moves to state acc, with no automated engine control.*

5. (a) Which of the following strings are accepted by the NFA in the diagram? (The start state is indicated by an arrow and the accepting state by a double



border.)

i. abb *No*

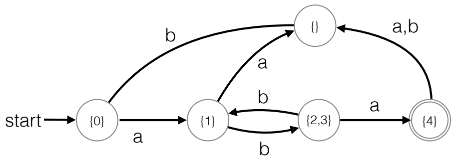
ii. aba *Yes*

iii. abbbbbba *No*

iv. abbbbbbba *Yes* [*4 marks*]

(b) Write a regular expression for the language accepted by this NFA. *a(bb)*ba* [*2 marks*]

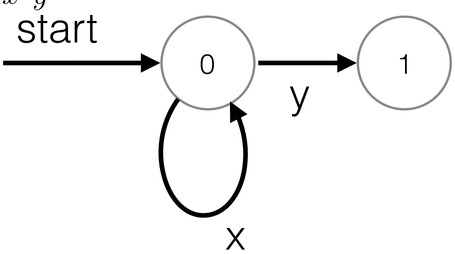
(c) Draw a DFA that accepts the same language. Label the states of your DFA to make clear their relationship to the states of the original NFA.



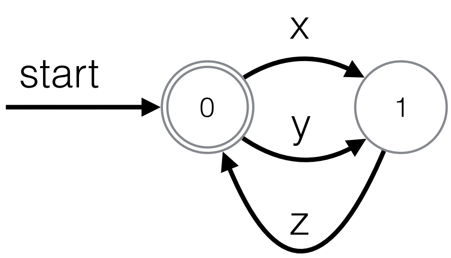
[*5 marks*]

(d) For each of the following regular expressions, draw a non-deterministic finite state machine that accepts the language described by the regular expression.

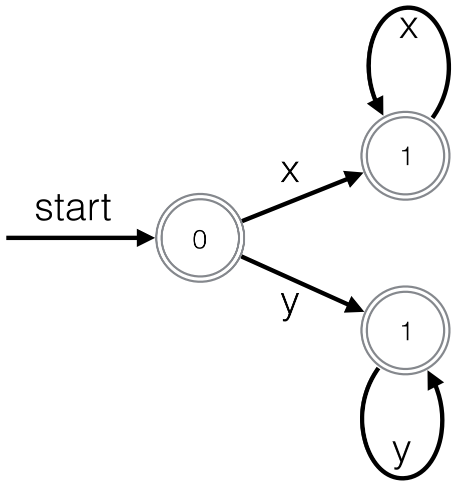
i. x^*y



ii. $((x|y)z)*$



iii. $(x^*|y^*)$



[9 marks]