Technische Universiteit Eindhoven University of Technology

Student name:

Student number:

Examination cover sheet

(to be completed by the examiner)

Course name: Algorithms for Model Checking — Course code: 2IMF35

Date: 10-04-2017

Start time: 13:30 — End time : 16:30

Number of pages: 2

Number of questions: 4

Maximum number of points/distribution of points over questions:100

Method of determining final grade: divide total of points by 10

Answering style: open questions

Exam inspection: With the lecturer

Other remarks:

Instructions for students and invigilators

Permitted examination aids (to be supplied by students):

- ☐ Notebook
- ☐ Calculator
- ☐ Graphic calculator
- ☑ Lecture notes/book
- ☐ One A4 sheet of annotations
- ☐ Dictionar(y)(ies). If yes, please specify:
- ☑ Other: Notes, sheets of annotations and other written material

Important:

- examinees are only permitted to visit the toilets under supervision
- it is not permitted to leave the examination room within 15 minutes of the start and within the final 15 minutes of the examination, unless stated otherwise
- examination scripts (fully completed examination paper, stating name, student number, etc.) must always be handed in
- the house rules must be observed during the examination
- the instructions of examiners and invigilators must be followed
- no pencil cases are permitted on desks
- examinees are not permitted to share examination aids or lend them to each other

During written examinations, the following actions will in any case be deemed to constitute fraud or attempted fraud:

- using another person's proof of identity/campus card (student identity card)
- having a mobile telephone or any other type of media-carrying device on your desk or in your clothes
- using, or attempting to use, unauthorized resources and aids, such as the internet, a mobile telephone, etc.
- using a clicker that does not belong to you
- having any paper at hand other than that provided by TU/e, unless stated otherwise
- visiting the toilet (or going outside) without permission or supervision

Examination Algorithms for Model Checking (2IMF35)

10 April, 2017, 13:30 – 16:30

Important notes:

- The exam consists of four questions.
- Weighting: 1: **20**, 2: **25**, 3: **35**, 4: **20**.
- Your grade is determined by dividing the points obtained by 10.
- Carefully read and answer the questions.
- The book, the course notes and other written material may be used during this examination. Laptops and other electronic equipment are not to be used.

---

1. Consider the following μ-calculus formulae interpreted over mixed Kripke structures with action alphabet $\{a, b\}$.:

   (A) $\nu X.\ (\ \langle b\rangle X \wedge [a]\mathsf{false} \wedge \mu Y.\ (\langle b\rangle Y \vee \langle a\rangle \mathsf{true})\ )$
   (B) $(\mu X.(\ [a]X \vee \langle b\rangle \mathsf{true}\ )) \wedge (\nu Y.\ (\ [b]\nu Z.\ \langle a\rangle (Z \wedge Y)\ ))$
   (C) $\nu X.\ \neg(\nu Y.\ (\ [a]\neg X \wedge [b]Y\ ))$

   (a) **(For 9pt).** Compute, for all three formulae A, B and C, their *nesting depth*, their *alternation depth* and their *dependent alternation depth*.

   (b) **(For 11pt).** Is there a mixed Kripke structure over action alphabet $\{a, b\}$ for which formula A holds for the initial state? If so, give such a mixed Kripke structure and prove using a transformation to Boolean equation systems and subsequent solving of the Boolean equation system using Gauß Elimination that this is the case. If not, prove that such a mixed Kripke structure cannot exist.

   Solution:

   (a) We first rewrite formula C to positive normal form, yielding:

   $$\nu X.\mu Y.(\langle a\rangle X \vee \langle b\rangle Y)$$

   Next, observe that the nesting depth of the formulae is as follows:

   $$ND(\mathsf{A}) = 2 \qquad ND(\mathsf{B}) = 2 \qquad ND(\mathsf{C}) = 2$$

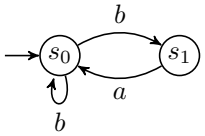
   For the alternation depth, we have the following:

   $$AD(\mathsf{A}) = 2 \qquad AD(\mathsf{B}) = 1 \qquad AD(\mathsf{C}) = 2$$

   For the dependent alternation depth, we have the following:

   $$dAD(\mathsf{A}) = 1 \qquad dAD(\mathsf{B}) = 1 \qquad dAD(\mathsf{C}) = 2$$

   (b) Informally, formula A states that there must be an infinite b-path through states that cannot perform an a-transition, but on which an a-transition is possible via a finite b-path. A mixed Kripke structure in which this holds in the initial state is as follows:



We translate the model checking problem to the following Boolean equation system:

$$(\nu X_{s_0} = (X_{s_0} \vee X_{s_1}) \wedge Y_{s_0})\ (\nu X_{s_1} = \mathit{false})\ (\mu Y_{s_0} = Y_{s_0} \vee Y_{s_1})\ (\mu Y_{s_1} = \mathit{true})$$

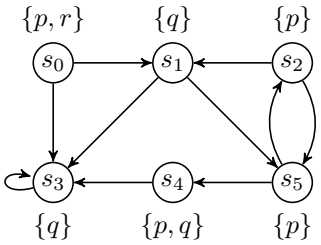
We use Gauß Elimination to sove this BES; the table below indicates each iteration in the algorithm, applying in each step (bottom up) first local solution, replacing every occurrence of a left-hand side variable X in its right-hand side with either true *or* false*, dependent on the fixpoint sign, and, next, substitutes the (simplified) right-hand side expressions upwards, replacing every occurrence of X with this simplified expression. The local solution and substitution steps are combined in one step in the table below.*

	1	*2*	*3*
$\nu X_{s_0} = (X_{s_0} \vee X_{s_1}) \wedge Y_{s_0}$	$(X_{s_0} \vee X_{s_1}) \wedge Y_{s_0}$	$X_{s_0} \vee X_{s_1}$	*true*
$\nu X_{s_1} = \mathit{false}$	*false*	*false*	-
$\mu Y_{s_0} = Y_{s_0} \vee Y_{s_1}$	*true*	-	-
$\mu Y_{s_1} = \mathit{true}$	-	-	-

We obtain true *for X_{s_0}, which encodes our model checking problem for the initial state, indicating that the formula holds for state s_0.*

□

2. Consider the following Kripke Structure K over the set $AP = \{p, q, r\}$:



(For 25pt). Determine the set of states in K where the CTL formula $\mathsf{E}\,[\,(\mathsf{E}\ \mathsf{G}\ p)\ \mathsf{U}\ (\mathsf{E}\ \mathsf{G}\ q)\,]$ holds using the **symbolic model checking algorithm** (lecture 2) for CTL. Use set notation to represent states instead of BDDs and include the relevant intermediate steps in your answer.

Solution: The symbolic model checking algorithm check *runs inside-out so it will first compute the sets of states associated to the subformulae* $\mathsf{E}\ \mathsf{G}\ p$ *and* $\mathsf{E}\ \mathsf{G}\ q$ *using the following fixpoint formulae:*

$$\begin{aligned} \mathsf{E}\ \mathsf{G}\ p &= \nu Z.(\mathsf{E}\ \mathsf{X}\ Z \wedge p) \\ \mathsf{E}\ \mathsf{G}\ q &= \nu W.(\mathsf{E}\ \mathsf{X}\ W \wedge q) \end{aligned}$$

Determining in which states p and in which states q holds is elementary: p holds in $\{s_0, s_2, s_4, s_5\}$ whereas q holds in $\{s_1, s_3, s_4\}$. We start by evaluating $\mathsf{E}\ \mathsf{G}\ p$ *using the* Gfp *(sub)algorithm:*

$$\begin{aligned} Z^0 &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\ Z^1 &= \{s_0, s_2, s_4, s_5\} \\ Z^2 &= \{s_2, s_5\} \\ Z^3 &= \{s_2, s_5\} \end{aligned}$$

Next, we evaluate $\mathsf{E}\ \mathsf{G}\ q$*, again using the* Gfp *(sub)algorithm:*

$$\begin{aligned} W^0 &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\ W^1 &= \{s_1, s_3, s_4\} \\ W^2 &= \{s_1, s_3, s_4\} \end{aligned}$$

Finally, we transform the original formula $\mathsf{E}\,[\,(\mathsf{E}\;\mathsf{G}\;p)\;\mathsf{U}\;(\mathsf{E}\;\mathsf{G}\;q)\,]$ *to a fixpoint formula, substituting the answers returned by the* check *procedure for the subformulae that were already resolved:*

$$\mathsf{E}\,[\,(\mathsf{E}\;\mathsf{G}\;p)\;\mathsf{U}\;(\mathsf{E}\;\mathsf{G}\;q)\,] \;\; = \mu Y.((\nu Z.\{s_2, s_5\} \wedge \mathsf{E}\;\mathsf{X}\;Y) \vee \{s_1, s_3, s_4\})$$

Finally, we evaluate $\mathsf{E}\,[\,(\mathsf{E}\;\mathsf{G}\;p)\;\mathsf{U}\;(\mathsf{E}\;\mathsf{G}\;q)\,]$ *using the* Lfp *(sub)algorithm:*

$$\begin{aligned} Y^0 &= \emptyset \\ Y^1 &= \{s_1, s_3, s_4\} \\ Y^2 &= \{s_1, s_2, s_3, s_4, s_5\} \\ Y^3 &= \{s_1, s_2, s_3, s_4, s_5\} \end{aligned}$$

So, the property holds everywhere except for in state s_0. □

3. Consider the following four constraints for a parity game $G = (V, E, p, (V_\Diamond, V_\Box))$, consisting of 6 vertices:

   i. for every $1 \leq i \leq 6$, there is a vertex v_i such that $p(v_i) = i$; *i.e.*, all priorities are different and taken from the range $1 \ldots 6$,
   ii. G has at least 10 edges,
   iii. player □ **has** a strategy ρ that guarantees that priority 3 occurs infinitely often on **all** plays (i.e. plays starting in arbitrary vertices) consistent with ρ,
   iv. G has an ◊-dominion consisting of at least two vertices and an □-dominion consisting of at least two vertices.

   (a) **(For 5pt).** Give a parity game G that meets all of the above constraints.

   (b) **(For 5pt).** Define a strategy ρ for player □ and show that your game G of question (a) satisfies property (iii).

   (c) **(For 5pt).** Show your game G satisfies property (iv); i.e. state the two respective dominions and prove that these are dominions.

   (d) **(For 10pt).** Solve your game G using either the recursive algorithm or the small progress measures algorithm. In case you use the recursive algorithm, clearly indicate which subgames are solved in each recursive step of the algorithm. For the small progress measures algorithm clearly indicate the lifting strategy and the intermediate measures.

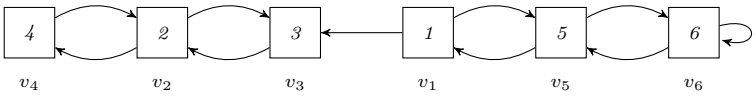
   (e) **(For 10pt).** Consider, in addition, the following constraint (v) on G:

      v. player ◊ **has** a strategy σ that guarantees that priority 4 occurs infinitely often on **all** plays (i.e. plays starting in arbitrary vertices) consistent with σ,

      Prove that player □ always wins the vertex with priority 4 in an arbitrary parity game that meets requirements (i)–(v), or give a counterexample by providing a parity game satisfying constraints (i)–(v) in which the vertex with priority 4 is won by player ◊.

Solution:

(a) There are multiple parity games that fulfil all properties. Most importantly, one must observe that player □ *must be able to force visiting vertex* v_3 *infinitely often from both dominions. That means that* v_3 *can only be part of the* ◊ *dominion: if if* v_3 *were part of the* □ *dominion, player* □ *could force play to* v_3 *from every vertex* x *in the* ◊ *dominion using strategy* ρ *of constraint (iii), and thus win vertex* x*, contradicting that* x *belongs to an* ◊*-dominion. Since* v_3 *belongs to an* ◊ *dominion and player* □ *can use* ρ *to infinitely often visit* v_3*, all such plays must be dominated by a lower even priority, so* v_2 *must belong to the* ◊*-dominion as well. The parity game depicted below is a possible solution to all constraints:*



Observe that G has exactly 10 edges, and all priorities in between 1 and 6 are present.

(b) *Let the strategy ρ for player $\Box$ be defined as: $\rho(v_1) = v_3, \rho(v_2) = v_3, \rho(v_3) = v_2, \rho(v_4) = v_2, \rho(v_5) = v_1$ and $\rho(v_6) = v_5$. Observe that any play that reaches vertex v_3 will subsequently reach vertex v_2. By definition of ρ, such a play will next again visit v_3. So, every play reaching v_3 will infinitely often visit v_3. Next observe that any play starting in an arbitrary vertex will reach v_3 in zero or more steps by following ρ. So all plays consistent with ρ will visit v_3 eventually, and, hence, infinitely often.*

(c) *The $\Diamond$-dominion consists of vertices v_2, v_3 and v_4: this is a closed subgame (player $\Box$ cannot escape from these vertices) and all plays within these vertices are $\Diamond$-dominated. The $\Box$-dominion consists of vertices v_1, v_5 and v_6. The strategy ρ given by $\rho(v_1) = v_5$ and $\rho(v_5) = v_1$ and $\rho(v_6) = v_5$ ensures that player $\Diamond$ cannot escape and the strategy ρ is winning for $\Box$.*

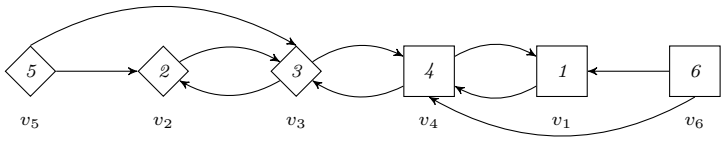
(d) *A straightforward application of the small progress measures algorithm stabilises after several liftings. We indicate the lifting scheme in the following table (we only indicate the initialisation and the update of a measure of a vertex, and we omit the 'even' positions in the tuples as these are constant 0):*

Vertex	*1*	*2*	*3*	*4*	*5*	*6*
v_1	$(0,0,0)$	$(1,0,0)$		$\top$		
v_2	$(0,0,0)$					
v_3	$(0,0,0)$		$(0,1,0)$			
v_4	$(0,0,0)$					
v_5	$(0,0,0)$				$\top$	
v_6	$(0,0,0)$					$\top$

All vertices lifted to $\top$ are won by player $\Box$. The remaining vertices are won by player $\Diamond$.

Applying the recursive algorithm will first compute the $\Box$-attractor into v_1, yielding the vertices $\{v_1, v_5, v_6\}$. Recursively solving the subgame restricted to vertices $\{v_2, v_3, v_4\}$, the algorithm concludes that this subgame is won entirely by player $\Diamond$. As a result, the algorithm will compute the $\Diamond$-attractor into the set $\{v_2, v_3, v_4\}$, yielding $\{v_2, v_3, v_4\}$ and solving the subgame restricted to $\{v_1, v_5, v_6\}$. For the latter it will conclude that player $\Box$ wins this entire subgame. Combining all information, the recursive algorithm will then return the partition $(\{v_2, v_3, v_4\}, \{v_1, v_5, v_6\})$ of vertices won by player $\Diamond$ and player $\Box$, respectively.

(e) *We prove that all games satisfying constraints (i)–(v) are such that vertex v_4 is won by player $\Box$. First, we illustrate that there are parity games that meet all constraints.*



Let σ be an arbitrary strategy for player $\Diamond$ such that vertex v_4 is visited infinitely often on all plays consistent with σ. Denote the non-empty $\Box$-dominion by $D_\Box$, and denote the closed, winning strategy for this dominion by $\sigma_\Box$.

We show that $v_4 \in D_\Box$. Pick a vertex $v \in D_\Box$; since $D_\Box$ is non-empty, such a vertex must exist. Consider the (unique) play π that emerges, starting in v, while playing

consistently with $\sigma_\Box$ and σ. Then v_4 occurs infinitely often on π, and, since $\sigma_\Box$ is closed on $D_\Box$ this can only be when $v_4 \in D_\Box$. But since $D_\Box$ is an $\Box$-dominion, this means that v_4 is won by player $\Box$.

$\Box$

4. Consider the following parameterised Boolean equation system:

$$\begin{array}{ll} (\nu X(n : Nat) & = (\mathsf{odd}(n) \wedge X(n+1)) \vee (\neg\mathsf{odd}(n) \wedge Y(n,n))) \\ (\mu Y(k : Nat, n : Nat) & = Y(k, n+1) \vee X(k+1)) \end{array}$$

Here, Nat represents the natural numbers, $\mathsf{odd}(n)$ yields true iff n is odd, and $+$ denotes addition.

(a) **(For 10pt).** Compute the solution to $X(0)$ in the above parameterised Boolean equation system using Gauß Elimination and Symbolic approximation. Include the important steps in the computations in your answer.

(b) **(For 10pt).** In case $X(0)$ has solution false, give a refutation graph, and, in case $X(0)$ has solution true, give a proof graph that explains this. In both cases, formally justify that this is a proof/refutation graph. That is, formally state the set of vertices, edges and labelling of the graph (or, if the graph is finite, draw the graph and clearly indicate the labelling of the graph) and show that it meets the properties of a refutation graph or a proof graph.

Solution:

(a) We first solve the equation for Y using symbolic approximation. The solution to Y is then inserted in the equation for X. Symbolic approximation of X yields the following:

$$\begin{array}{ll} Y^0(k,n) & = \mathit{false} \\ Y^1(k,n) & = \mathit{false} \vee X(k+1) \\ & = X(k+1) \\ Y^2(k,n) & = X(k+1) \vee X(k+1) \\ & = X(k+1) \end{array}$$

Since we have $Y^1 \leftrightarrow Y^2$, Y^2 is the stable solution for Y. We next substitute solution Y^2 for Y in the right-hand side of the equation for X, yielding:

$$(\nu X(n : Nat) = (\mathit{odd}(n) \wedge X(n+1)) \vee (\neg\mathit{odd}(n) \wedge X(n+1))$$

Again using a symbolic approximation, we obtain:

$$\begin{array}{ll} X^0(n) & = \mathit{true} \\ X^1(n) & = (\mathit{odd}(n) \wedge \mathit{true}) \vee (\neg\mathit{odd}(n) \wedge \mathit{true}) \\ & = \mathit{true} \end{array}$$

Clearly, $X^0 \leftrightarrow X^1$, so we find that $X(v) = \mathit{true}$ for all $v \in \mathbb{N}$. In particular, $X(0) = \mathit{true}$.

(b) Given that the solution to $X(0)$ is true, we should construct a proof graph. Despite the fact that we could simply prove that $X(0)$ has solution true, the proof graph underlying this fact is infinite. Consider the graph $G = (V, R, L)$, where:

- *Vertices $V = \{(X,i), (Y,(2i,2i)) \mid i \in \mathbb{N}\}$;*
- *Edges $R \subseteq V \times V$ are given by:*

$$R = \{\langle (X,2i), (Y,2i,2i)\rangle, \langle (X,2i+1), (X,2i+2)\rangle, \langle (Y,2i,2i), (X,2i+1)\rangle\}$$

- *Labelling $L((X,i)) = 0, L((Y,(2i,2i))) = 1$ for all i.*

Observe that there is exactly one infinite path through the proof graph starting in vertex $(X, 0)$*, and this path is even-dominated as we infinitely often pass through a vertex of the form* (X, i)*, which has the lowest rank. To see that the graph is a* true*-dependency graph, note that:*

- *For vertex* $(X, 2i)$ *we have:*
  $[\![(\mathsf{odd}(n) \wedge X(n+1)) \vee (\neg\mathsf{odd}(n) \wedge Y(n,n))]\!](X,2i)^\bullet{}_{\mathsf{true}}\varepsilon[2i/n]$
  $=$ $\{2i$ *is not odd*$\}$
  $(2i, 2i) \in (X, 2i)^\bullet_{\mathsf{true}}(Y)$
  $=$
  true since $(Y, 2i, 2i) \in (X, 2i)^\bullet$
- *For vertex* $(X, 2i+1)$ *we have:*
  $[\![(\mathsf{odd}(n) \wedge X(n+1)) \vee (\neg\mathsf{odd}(n) \wedge Y(n,n))]\!](X,2i)^\bullet{}_{\mathsf{true}}\varepsilon[2i+1/n]$
  $=$ $\{2i+1$ *is odd*$\}$
  $(2i+1)+1 \in (X, 2i+1)^\bullet_{\mathsf{true}}(Y)$
  $=$
  true since $(X, 2i+2) \in (X, 2i+1)^\bullet$
- *For vertex* $(Y, 2i, 2i)$ *we have:*
  $[\![Y(k, n+1) \vee X(k+1)]\!](Y,2i,2i)^\bullet{}_{\mathsf{true}}\varepsilon[2i/k, 2i/n]$
  $=$
  $(2i, 2i+1) \in (Y, 2i, 2i)^\bullet_{\mathsf{true}}(Y))$ *or* $2i+1 \in (Y, 2i, 2i)^\bullet_{\mathsf{true}}(X)$
  $=$
  true since $(X, 2i+1) \in (Y, 2i, 2i)^\bullet$

□