Examination Algorithms for Model Checking (2IW55)

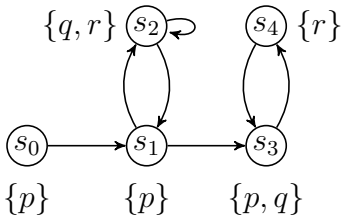
7 January, 2009, 14:00 – 17:00

Important notes:

- The exam consists of four questions.
- Weighting: 1: **20**, 2: **20**, 3: **30**, 4: **30**.
- Carefully read and answer the questions. The book, the course notes and other written material may be used during this examination.

---

1. Consider the following Kripke Structure, where $\{p, q, r\}$ is the set of atomic propositions. Assume as fairness constraint the set of states satisfying r.



Use the labelling algorithm for CTL to compute the set of states that fairly satisfy E G E $[p$ U $q]$. If E G E $[p$ U $q]$ holds fairly in state s_0 compute a witness from this state. Show the intermediate steps in all your computations.

Answer (sketch only).

The fairness constraint r translates to $\mathcal{F} = \{\{s_2, s_4\}\}$; so, any path hitting either s_2 or s_4 infinitely often is a fair path (it is not required to hit **both** *states infinitely often). With that in mind, it is easy to see that* E G E $[p$ U $q]$ *holds in s_0, s_1 and s_2. This can be substantiated by means of a computation. For this, the below steps, in the presented order, are required. Each step is worth 5 points; the witness (with a formal argument) yields another 5 points.*

1. *label all states in the KS that satisfy* E G *true fairly with the proposition fair; technically, this requires a decomposition into SCCs (trivial SCC $\{s_0\}$, non-trivial SCCs $\{s_1, s_2\}$ and $\{s_3, s_4\}$); results in labelling all states with fair;*
2. *label each state for which* E $[p$ U $(q \wedge fair)]$ *holds; perform a backwards reachability for states satisfying p, starting in states satisfying $q \wedge fair$. Results in labelling all states except for s_4;*

3. *label each state for which* $\mathsf{E}\ \mathsf{G}\ \mathsf{E}\ [p\ \mathsf{U}\ (q \wedge fair)]$ *holds; first restrict the KS to those states satisfying* $\mathsf{E}\ [p\ \mathsf{U}\ (q \wedge fair)]$*; compute the SCCs in the restricted graph (trivial SCC* $\{s_0\}$*, non-trivial SCC* $\{s_1, s_2\}$*); perform a backwards reachability for states satisfying* $\mathsf{E}\ [p\ \mathsf{U}\ (q \wedge fair)]$*. This results in labelling states* $\{s_0, s_1, s_2\}$*.*

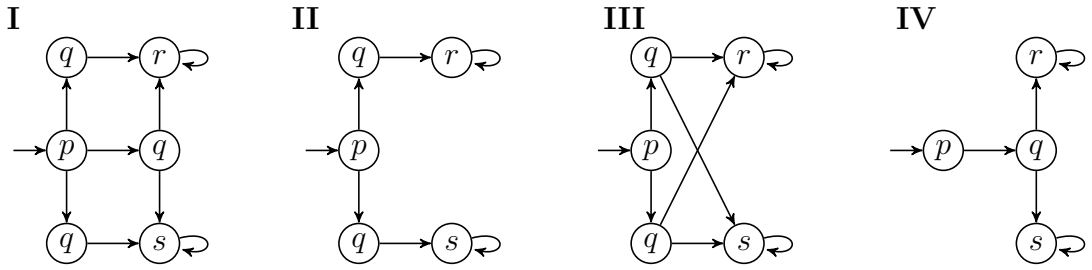
(a) Since the property holds in s_0*, one has to provide a witness. Two options:*

i. *use the witness calculation scheme based on fixed points (for that, the original formula must be rephrased to a fixed point formula).*

ii. *use the labels added to the states in computing* $\mathsf{E}\ \mathsf{G}\ \mathsf{E}\ [p\ \mathsf{U}\ q]$*. Construct an* **infinite** *path through states satisfying that have this label.*

Both routes should yield either $s_0\ s_1\ (s_2^\omega)$ *or* $s_0\ (s_1\ s_2)^\omega$*; both are fine.*

2. Consider the following four Kripke Structures (**I** through **IV**), where $\{p, q, r, s\}$ is the set of atomic propositions and $\rightarrow$ marks the initial state.



(a) Determine whether the following properties hold for graphs **I**, **II** and **III**. If so, give the simulation relation that supports your answer. If not, give a formula in CTL^* that witnesses this fact.

i. **I** simulates **II** (i.e., **II** $\sqsubseteq$ **I**)

ii. **II** simulates **I** (i.e., **I** $\sqsubseteq$ **II**)

iii. **I** simulates **III** (i.e., **III** $\sqsubseteq$ **I**)

iv. **III** simulates **I** (i.e., **I** $\sqsubseteq$ **III**)

(b) Compute the bisimulation relation B^* between **III** and **IV**. Show the intermediate approximations B_i^* for B^*. Use explicit set notation instead of BDDs.

Answer (sketch only).

Each correctly motivated answer in (a) is worth 4 points; observations of (non-)simulation are worth only 1 point; (b) is worth another 4 points.

Question (a). We have: **II** $\sqsubseteq$ **I**, **III** $\sqsubseteq$ **I** *and* **I** $\sqsubseteq$ **III**, *which follows immediately from applying the definition of a simulation. We do not have* **I** $\sqsubseteq$ **II**. *A counterexample must be taken from the* CTL^* *subset* ACTL^**, or –by duality– the subset* ECTL^* *(analogous to* ACTL^* *but with* E *path quantifiers only). We have:*

- $\mathbf{I} \models \mathsf{E}\ \mathsf{X}\ (\mathsf{E}\ \mathsf{X}\ r \wedge \mathsf{E}\ \mathsf{X}\ s)$ *and* $\mathbf{II} \not\models \mathsf{E}\ \mathsf{X}\ (\mathsf{E}\ \mathsf{X}\ r \wedge \mathsf{E}\ \mathsf{X}\ s)$, *or*
- $\mathbf{I} \not\models \mathsf{A}\ \mathsf{X}\ (\mathsf{A}\ \mathsf{X}\ r \vee \mathsf{A}\ \mathsf{X}\ s)$ *and* $\mathbf{II} \models \mathsf{A}\ \mathsf{X}\ (\mathsf{A}\ \mathsf{X}\ r \vee \mathsf{A}\ \mathsf{X}\ s)$.

Question (b). Obviously, **III** *and* **IV** *are strong bisimilar. The computation stabilises immediately:* $B_0^* = B_1^*$*, and, hence,* $B^* = B_0^*$*. Give each state a unique name, e.g., label each state with their proposition and KS name, and in case of doubles (e.g., proposition q), introduce a tagging* q^1 *and* q^2*. We then have:*

$$\begin{aligned} B_0^* &= \{(p_{III}, p_{IV}), (q^1_{III}, q_{IV}), (q^2_{III}, q_{IV}), (r_{III}, r_{IV}), (s_{III}, s_{IV})\} \\ B_1^* &= \{(p_{III}, p_{IV}), (q^1_{III}, q_{IV}), (q^2_{III}, q_{IV}), (r_{III}, r_{IV}), (s_{III}, s_{IV})\} \end{aligned}$$

3. Consider the LPE description of a lossy channel system, where actions r, s and l represent *receiving*, *sending* and *losing*, respectively.

$$\begin{aligned} C(b{:}Bool, m{:}Nat) &= \sum_{k:Nat} b \longrightarrow r(k) \cdot C(\mathsf{false}, k) \\ &+ \neg b \longrightarrow s(m) \cdot C(\mathsf{true}, m) \\ &+ \neg b \longrightarrow l \cdot C(\mathsf{true}, m) \end{aligned}$$

Let ϕ be first-order modal μ-calculus formula $\nu X.\ \mu Y.(\langle l \rangle X \vee \langle \neg l \rangle Y)$.

(a) Verify whether the PBES given below can be the result (up to logical equivalence) of the transformation $\mathsf{E}\ \phi$ applied to C. Clearly relate the (sub)expressions in the PBES to the (sub)expressions in ϕ, or mark the (sub)expression(s) of ϕ that demonstrate(s) an error in the transformation and correct it.

$$\begin{aligned} \Big(\nu X(b{:}Bool, m{:}Nat) &= Y(b, m)\Big) \\ \Big(\mu Y(b{:}Bool, m{:}Nat) &= (\neg b \wedge X(\mathsf{true}, m)) \\ &\vee (\neg b \wedge Y(\mathsf{true}, m) \\ &\vee (\exists k{:}Nat.\ b \wedge Y(\mathsf{false}, k))\Big) \end{aligned}$$

(b) If possible, compute and solve a Boolean Equation System from the above PBES that answers whether $X(\mathsf{true}, 0) = \mathsf{true}$, or clearly indicate why this cannot be done.

Answer (sketch only).

Building the carcas using E *will give you 5 points; doing the same for the* RHS *operator will earn you another 10 points. Showing your skills in instantiation, and observing that it will not terminate on the original PBES will give you 5 points; observing that parameter* m *is redundant in* X *and* Y *will give you 5 points; transforming the PBES to one without natural numbers, applying the right theorem and instantiating the resulting PBES is worth another 5 points.*

Question (a). Apply the following steps in the following order:

1. *Apply* E *to* ϕ*, which gives rise to the equation for* X *and* Y*.*
2. *Observe that* $\mathsf{RHS}(\mu Y.(\langle l\rangle X \vee \langle \neg l\rangle Y)) = Y(b, m)$*;*
3. *Observe that* $\mathsf{RHS}(\langle l\rangle X \vee \langle \neg l\rangle Y) = \mathsf{RHS}(\langle l\rangle X) \vee \mathsf{RHS}(\langle \neg l\rangle X)$*;*
4. *Observe that* $\mathsf{RHS}(\langle l\rangle X) = (\neg b \wedge X(\mathsf{true}, m))$*;*
5. *Observe that* $\mathsf{RHS}(\langle \neg l\rangle Y) = ((\neg b \wedge Y(\mathsf{true}, m)) \vee (\exists k{:}Nat.\ b \wedge Y(\mathsf{false}, k)))$*;*

Question (b). Observe that a straightforward instantiation of the given PBES will not terminate; the existential quantifier cannot be removed. However, we find that variable m *is redundant in both* X *and* Y*, so, we can find an equivalent PBES in which only Booleans occur. To prove that this is the case, proceed as follows:*

1. *Construct the marked influence graph, using the notions of* significant variables *and the* dependency set*:*
   - *Vertices:* $\{(X, b), (X, m), (Y, b), (Y, m)\}$*;*
   - *Edges:* $(X, b) \rightarrow (Y, b)$*,* $(X, m) \rightarrow (Y, m)$*,* $(Y, m) \rightarrow (X, m)$*,* $(Y, m) \rightarrow (Y, m)$*;*
   - *Markings:* (Y, b)
2. *Perform a reachability analysis, finding out from which vertices vertex* (Y, b) *is reachable (only* (X, b) *and* (Y, b)*);*
3. *This implies that variable* m *can be removed from* X *and* Y*.*

The resulting PBES after elimination of m *and logical simplification is:*

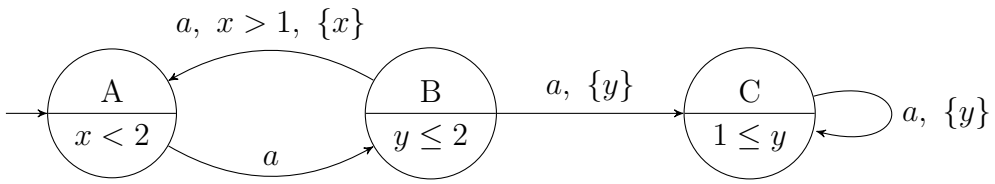
$$\begin{array}{lcl} \Big(\nu\tilde{X}(b{:}Bool) & = & \tilde{Y}(b)\Big) \\ \Big(\mu\tilde{Y}(b{:}Bool) & = & (\neg b \wedge \tilde{X}(\mathsf{true})) \vee (\neg b \wedge \tilde{Y}(\mathsf{true}) \vee (b \wedge \tilde{Y}(\mathsf{false}))\Big) \end{array}$$

Observe that there is no longer an existential quantification over natural numbers in this equivalent PBES, and all data types occurring in the PBES are of finite cardinality. This means that instantiation is guaranteed to terminate. Instantiation of the PBES starting from $\tilde{X}(\mathsf{true})$ *yields the following BES:*

$$\begin{array}{ll} \nu\tilde{X}^{\mathsf{true}} & = \tilde{Y}^{\mathsf{true}} \\ \mu\tilde{Y}^{\mathsf{true}} & = \tilde{Y}^{\mathsf{false}} \\ \mu\tilde{Y}^{\mathsf{false}} & = \tilde{X}^{\mathsf{true}} \vee \tilde{Y}^{\mathsf{true}} \end{array}$$

which leads to the answer true *for* $\tilde{X}^{\mathsf{true}}$*, and, hence, for any* $m \in Nat$*, we have* $X(\mathsf{true}, m) = \mathsf{true}$*; in particular, we have* $X(\mathsf{true}, 0) = \mathsf{true}$*.*

4. Consider the Timed Automaton with three locations A, B and C, two clocks x, y and one action a. Location A is the initial location.



(a) Is the Timed Automaton non-Zeno? If so, give a proof. If not, give a Zeno path.

(b) Can the Timed Automaton timelock? Substantiate your answer using a transformation to the region automaton of the Timed Automaton.

Answer (sketch only).

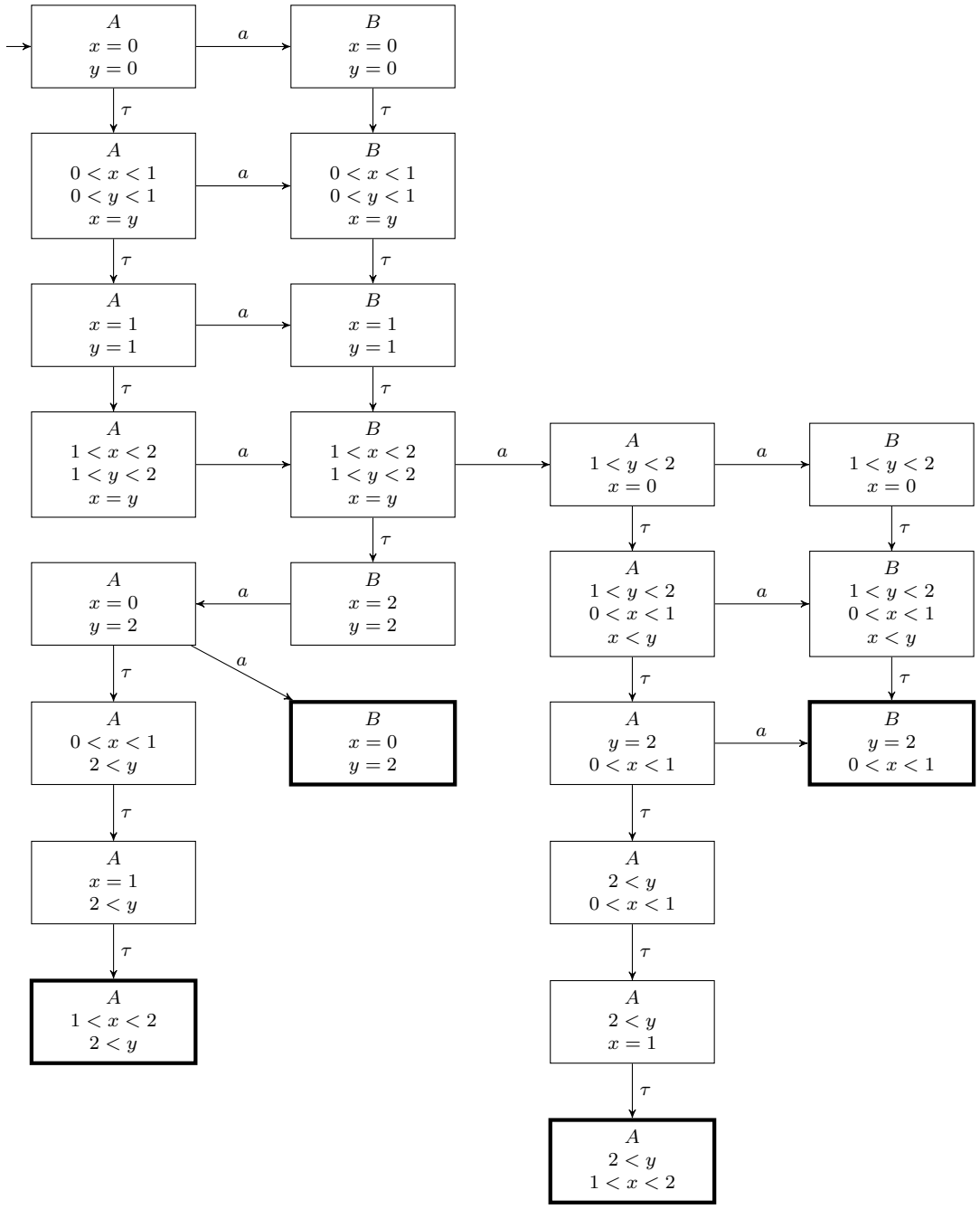
Observing that the TA is non-Zeno is worth 5 points; proving this leads to an additional 5 points. Observing that there is a time-lock in the TA is worth 5 points; constructing a Region Automaton from the TA, yields between 5 and 15 points, depending on the correctness and the appropriateness of the fragment of the RA which is displayed (15 points for the full RA, slightly less for partial RAs that still explain the time-lock).

Question (a). The TA is non-Zeno. To substantiate this bold claim, it suffices to show that both control cycles in the TA increase time by at least 1 time-unit each time the loop is run. We have:

1. *Cycle $(A\ B)^\omega$: clock x is reset; there is a guard $x > 1$ in this cycle;*
2. *Cycle C^ω: clock y is reset; there is an invariant $y \geq 1$ immediately after the reset of y.*

The above analysis also reveals that once location C is reached, there is no way to ensure progress. A closer look at the TA indicates that location C is not reachable, so it suffices to further study the TA with locations A and B only.

Question (b). The TA can time-lock. The sequence of events that leads to this time-lock are for instance waiting for 1 sec in A, execute a, wait 1 sec in B, execute a, and immediately execute a again. This chain of events can be found by studying the RA of the TA. The construction of the RA follows the recipe given in the TA handout (also on the slides). The entire RA is depicted on the next page; the encoding is as follows: clocks are compared against integers, and, in case of inequalities of both x and y (other than against c_x or c_y, i.e., the largest values to which clocks are compared), an extra constraint tells whether the fraction of x is smaller, equal or greater than the fraction of y. The 4 different deadlocking states represent the different the 4 situations in which the TA can time-lock.


A
x = 0
y = 0
a
B
x = 0
y = 0
τ
τ
A
0 < x < 1
0 < y < 1
x = y
a
B
0 < x < 1
0 < y < 1
x = y
τ
τ
A
x = 1
y = 1
a
B
x = 1
y = 1
τ
τ
A
1 < x < 2
1 < y < 2
x = y
a
B
1 < x < 2
1 < y < 2
x = y
a
A
1 < y < 2
x = 0
a
B
1 < y < 2
x = 0
τ
τ
A
x = 0
y = 2
a
B
x = 2
y = 2
τ
A
1 < y < 2
0 < x < 1
x < y
a
B
1 < y < 2
0 < x < 1
x < y
τ
τ
a
A
0 < x < 1
2 < y
B
x = 0
y = 2
A
y = 2
0 < x < 1
a
B
y = 2
0 < x < 1
τ
τ
A
x = 1
2 < y
A
2 < y
0 < x < 1
τ
τ
A
1 < x < 2
2 < y
A
2 < y
x = 1
τ
A
2 < y
1 < x < 2

□