Errata ”Principles of Model Checking” (2008)

Thanks to the Model Checking Reading Club at the Radboud University of Nijmegen, The Netherlands (in particular David N. Jansen and Frits W. Vaandrager), Holger Hermanns (Saarland University), Dave Parker (Oxford University, UK), René Thiemann (U. Innsbruck, Austria), Ahmed Khademzadeh (Azad University of Mashhad, Iran), Moti Ben-Ari (Weizmann Institute, Israel), Erika Abráh'am, Alexander Nyßen and Daniel Weber (RWTH Aachen University) and the students at the RWTH Aachen University attending the "(Advanced) Model Checking" lecture.

Comments are provided as:

$\langle$ page number $\rangle$ $\langle$ line number $\rangle$ $\langle$ short quote of the wrong word(s) $\rangle$ $\triangleright$ $\langle$ correction $\rangle$

Chapter 1: System Verification

pp. 1, l. -5, *Pentium II* $\triangleright$ Pentium

pp. 5, l. 9, *lines of code lines* $\triangleright$ lines of code

pp. 5, l. footnote, *much higher* $\triangleright$ as the number of lines of code in the "golden" version of Windows95 is about 15 million, the error rate is in fact lower than normal.

pp. 6, l. 4, *Pentium II* $\triangleright$ Pentium

Chapter 2: Modeling Concurrent Systems

pp. 25, l. 11, *heading Example 2.8* $\triangleright$ Execution fragments of the Beverage Vending Machine

pp. 27, l. -15, *function* λ_y $\triangleright$ The function λ_y has no impact on the transitions (as suggested), but only affects the state labeling.

pp. 31, l. Fig. 2.3, *beer, soda* $\triangleright$ *bget* and *sget*, respectively

pp. 31, l. Fig. 2.3, *state with 1 beer, 2 soda* $\triangleright$ the grey circle should be a white circle.

pp. 34, l. 2, $\langle \ell, v \rangle$ $\triangleright$ $\langle \ell, \eta \rangle$

pp. 42, l. -10, *interlock* $\triangleright$ interleave

pp. 46, l. Fig. 2.9, *locations in* PG_2 $\triangleright$ should be subscripted with 2 (rather than 1) pp. 48, l. -1, $H = Act_1 \cap Act_2$ ▷ $H = (Act_1 \cap Act_2) \setminus \{ \tau \}$

pp. 51, l. Fig. 2.12, $T_1 \parallel T_2$ ▷ $TS_1 \parallel TS_2$ (this occurs twice)

pp. 51, l. Fig. 2.12, ▷ All downgoing transitions should be labeled with *request*, and all upgoing ones with *release*

pp. 51, l. -7, *all trains* ▷ the train

pp. 52, l. 3, *(above)* ▷ (page 54)

pp. 53, l. -1, *finite set of channels* ▷ set of channels

pp. 54, l. Fig. 2.16, *the transition labeled approach emanating from state* $\langle far, 3, down \rangle$ ▷ should be removed, and all the states that thus become unreachable

pp. 54, l. Fig. 2.16, *the transition labeled exit emanating from state* $\langle in, 1, up \rangle$ ▷ should be removed, and all the states that thus become unreachable

pp. 55, l. -10, $(Cond(Var) \times$ ▷ $Cond(Var) \times$

pp. 62, l. -3, *gen_msg(1)* ▷ *snd_msg(1)*

pp. 64, l. 4, *ack* ▷ message

pp. 65, l. Fig. 2.21, *second* **do** ▷ **od**

pp. 66, l. 8, *Staements build* ▷ Statements built

pp. 71, l. 15, *label in conclusion of inference rule* $c!e$ ▷ it is meant that the value of expression e is transferred; cf. Exercise 2.8, pp. 85

pp. 74, l. 1, $\xi[c := v_2 \ldots v_k]$ ▷ $\xi' = \xi[c := v_2 \ldots v_k]$

pp. 74, l. 1, $\xi[c := v_1 \ldots v_k v]$ ▷ $\xi' = \xi[c := v_1 \ldots v_k v]$

pp. 76, l. Figure 2.23 (top), x ▷ x'

pp. 79, l. -6,-8, $|dom(c)|^{cp(c)}$ ▷ $|dom(c)|^{cap(c)}$

pp. 82, l. Exercise 2.2, line 2, $P_i is$ ▷ P_i is

Chapter 3: Linear-Time Properties

pp. 89, l. 9, *parallel systems* ▷ reactive systems

pp. 90, l. 1, *Fault Designed Traffic Lights* ▷ Faulty Traffic Lights

pp. 91, l. 7, *a deadlock occurs when all philosophers* ▷ a deadlock may occur when all philosophers

pp. 92, l. Fig. 3.2, *request and release* ▷ req and rel pp. 92, l. 6, *request*$_4$ $\triangleright$ *req*$_{4,4}$; similar to the other request actions

pp. 93, l. -4,-5 and Fig. 3.3, Fig. 3.4, *state available*$_i$ $\triangleright$ *available*$_{i,i}$

pp. 93, l. -4,-5 and Fig. 3.3, Fig. 3.4, *state available*$_{i+1}$ $\triangleright$ *available*$_{i,i+1}$

pp. 93, l. 10, *The corresponding is* $\triangleright$ The corresponding condition is

pp. 94, l. Fig. 3.4, *falls* x_i $\triangleright$ x_i

pp. 96, l. 3, *finite paths* $\triangleright$ finite path fragments

pp. 96, l. 4, *infinite path* $\triangleright$ infinite path fragment

pp. 100, l. 9, *(over AP)* $\triangleright$ (over 2^{AP})

pp. 101, l. -3, *red*$_1$ *green*$_2$ $\triangleright$ *red*$_1$, *green*$_2$

pp. 103, l. 11, *lwait*$_i$ $\triangleright$ *wait*$_i$

pp. 103, l. 11, $\exists k \geqslant j.\, wait_i \in A_k$ $\triangleright$ $\exists k > j.\, crit_i \in A_k$

pp. 111, l. Theorem 3.21, $M = \sum_{s \in S} |Post(s)|$ $\triangleright$ $M = \sum_{s \in Reach(TS)} |Post(s)|$

pp. 111, l. 22, *The time needed to check* $s \models \Phi$ *is linear in the length of* Φ $\triangleright$ Add: This implicitly assumes that $a \in L(s)$ can be checked in $\mathcal{O}(1)$ time.

pp. 112, l. -2, $\triangleright$ A minimal bad prefix is one such that the first occurrence of Φ is the last symbol in the word.

pp. 115, l. Lemma 3.27, *Proof* $\triangleright$ add the following sentence to the beginning of the proof: First note that for $P = (2^{AP})^\omega$ the claim trivially holds, since $closure(P) = P$ and the fact that P is a safety property since $\overline{P}$ is empty. In the remainder of the proof we consider $P \neq (2^{AP})^\omega$.

pp. 118, l. 10,11, $\pi^{m_0}\pi^{m_1}\pi^{m_2}\ldots$ *of* $\pi^0\pi^1\pi^2\ldots$ *such that* $\triangleright$ $\pi^{m_0}, \pi^{m_1}, \pi^{m_2}, \ldots$ of $\pi^0, \pi^1, \pi^2, \ldots$ such that

pp. 124, l. -3, *By definition* $\triangleright$ By Lemma 3.27

pp. 130, l. 3, *without being taken beyond* $\triangleright$ without being taken infinitely often beyond

pp. 131, l. 17, *assignment* $x = -1$ $\triangleright$ assignment $x := -1$

pp. 132, l. 2, *an execution fragment ... but not strongly A-fair.* $\triangleright$ an execution fragment that visits infinitely many states in which no A-action is enabled is weakly A-fair (as the premise of weak A-fairness does not hold) but may not be strongly A-fair.

pp. 134, l. 10, *any finite trace is fair by default* $\triangleright$ any finite trace is strongly or weakly fair by default

pp. 136, l. -5, *strong fairness property* $\triangleright$ fairness property

pp. 138, l. 4, *It forces synchronization actions to happen infinitely often.* $\triangleright$ It forces synchronization actions to happen infinitely often provided they are enabled infinitely often.

pp. 138, l. -14, *This requires that ... is enabled.* ▷ This requires that infinitely often a synchronization takes place when such synchronization is infinitely often enabled.

pp. 141, l. 5, *the set of properties that has* ▷ the property that has

pp. 145, l. Exercise 3.5(g), *between zero and two* ▷ between zero and non-zero

Chapter 4: Regular Properties

pp. 157, l. -11, $w = A_1 \dots A_n \in \Sigma$ ▷ $w = A_1 \dots A_n \in \Sigma^*$

pp. 157, l. -10, *starts in* Q_0 ▷ starts in state Q_0

pp. 157, l. -4, Q_0 ▷ $\{ Q_0 \}$

pp. 158, l. -14, *NFAs can be much more efficient.* ▷ NFAs can be much smaller.

pp. 161, l. -9, *(2) ... for all* $1 \leqslant i < n$ ▷ ... for all $0 \leqslant i < n$. (Note: the invariant false has minimal bad prefix ε.)

pp. 161, l. -8, $1 \leqslant i < n$ ▷ $0 \leqslant i < n$

pp. 163, l. Example 4.15, *Minimal bad prefixes for this safety property constitute the language* $\{ \mathit{pay}^n \mathit{drink}^{n+1} \mid n \geqslant 0 \}$ ▷ Bad prefixes for this safety property constitute the language $\{ \sigma \in \left(2^{\{pay,drink\}}\right)^\omega \mid w(\sigma, \mathit{drink}) > w(\sigma, \mathit{pay}) \}$ where $w(\sigma, a)$ denotes the number of occurrences of a in σ.

pp. 164, l. 5,6, *two NFAs intersect.* ▷ the languages of two NFAs intersect.

pp. 164, l. -8, *path fragment* π ▷ initial path fragment π

pp. 164, l. -6, $TS \otimes \mathcal{A}$ *which has an initial state* ▷ $TS \otimes \mathcal{A}$ such that there exists an initial state

pp. 167, l. 7, 11, -4, $P_{inv(A)}$ ▷ $P_{inv(\mathcal{A})}$

pp. 167, l. -2, $q_1, \dots, q_n \notin F$ ▷ Note: this condition is not necessary.

pp. 168, l. 1, $0 \leqslant i \leqslant n$ ▷ $0 < i \leqslant n$

pp. 171, l. 8, *single word* ▷ a set contaning a single word

pp. 177, l. -7, *Example 4.13 on page 161* ▷ Example 4.14 on page 162

pp. 183, l. -3, -1, $\mathcal{L}_{q_1 q_3} = \dots$ ▷ $\mathcal{L}_{q_1 q_3} = C^* AB(B + BC^* AB)^*$

pp. 196, l. Example 4.57, *page 193* ▷ page 194

pp. 200, l. -7, $\bigwedge_{q \in Q}$ ▷ $\bigwedge_{q \in F}$

pp. 202, l. Fig. 4.22, ▷ The two states should be labeled s_0 and s_1, respectively pp. 203, l. 4, $\overline{P}$ = *"eventually forever* ¬ *green* ▷ P = infinitely often *green*

pp. 206, l. Proof:, $TS = (S, Act, \rightarrow, I, AP)$ ▷ $TS = (S, Act, \rightarrow, I, AP, L)$

pp. 207, l. -4, *We now DFS-based cycle checks ... checking* ▷ We now present a DFS-based algorithm for persistence checking that searches backwards edges to check for cycles.

pp. 212, l. 6, *ignores* T ▷ does not revisit the states in T

pp. 218, l. 10, *Regula r* ▷ Regular

Chapter 5: Linear Temporal Logic

pp. 230, l. 5, *eventually in the future* ▷ now or eventually in the future

pp. 236, l. Figure 5.2, ▷ It is assumed that $\sigma = A_0 A_1 A_2 \ldots$

pp. 240, l. -10, $\delta_{r_2} = \neg r_1$ ▷ $\delta_{r_2} = \neg r_2$

pp. 241, l. Fig. 5.6, ▷ Note that the inputs of the r registers are on the right, and their outputs on the left.

pp. 267, l. 7, *as soon as* ▷ before

pp. 270, l. Fig. 5.15, ▷ The bottom cell should be white and not gray.

pp. 276, l. -11, $\psi \in B$*if and only if* ... ▷ $\psi \in B$ if and only if ...

pp. 281, l. 1-5, *For* $B_0 B_1 B_2 \ldots$ *a sequence ... we have for all* $\psi \in cl(\varphi)$*:* $\psi \in B_0 \Leftrightarrow A_0 A_1 A_2 \ldots \models \psi$ ▷ For all $\psi \in cl(\varphi)$ and $B_0 B_1 B_2 \ldots$ a sequence ... we have: $\psi \in B_0 \Leftrightarrow A_0 A_1 A_2 \ldots \models \psi$

pp. 283, l. 10, $\neq \bigcirc \psi \in B$ *if and* ... ▷ $\neg \bigcirc \psi \in B$ if and ...

pp. 283, l. 17, *and* $\varphi = \bigcirc a \in B_1, B_2$ ▷ and $\varphi = a \in B_1, B_2$

pp. 284, l. -14, $B_3\, B_3\, B_1 B_4^\omega$ ▷ $B_3\, B_3\, B_1 B_5^\omega$

pp. 287, l. -5, $|\neg(fair \rightarrow \varphi)| = |fair| + |\varphi|$ ▷ $|\neg(fair \rightarrow \varphi)| = |\neg(\neg fair \vee \varphi)| = |fair| + |\varphi| + 3$

pp. 289, l. 11, *a new vertex* b *to* G ▷ a new vertex b to TS

pp. 292, l. Figure 5.23, ▷ the self-loop at state $P(n)$ should be omitted

pp. 292, l. -1, $\bigcirc^{2i-1}(q, A, i) \rightarrow$ ▷ $begin \wedge \bigcirc^{2i-1}(q, A, i) \rightarrow$

pp. 294, l. -6, $\mathcal{G}_v arphi$ ▷ $\mathcal{G}_\varphi$

pp. 297, l. 7, *Membership to* ▷ Membership in

pp. 303, l. Exercise 5.7(b), W ▷ Y (to avoid confusion with unless)

Chapter 6: Computation Tree Logic

pp. 320, l. -4, *state formula* $\rhd$ State formula

pp. 327, l. -12, *since* $\exists(\varphi \mathsf{U} \psi \ \vee \ \Box \varphi)$ $\rhd$ since $\forall(\varphi \mathsf{U} \psi \ \vee \ \Box \varphi)$

pp. 333, l. 10, $\neg\exists\Diamond\neg\Phi \ = \ \neg\exists(\mathit{true} \, \mathsf{U} \, \Phi)$ $\rhd$ $\neg\exists\Diamond\neg\Phi \ \equiv \ \neg\exists(\text{true} \, \mathsf{U} \, \neg\Phi)$

pp. 338, l. -5 and -6, $\rhd$ transitions to s'_{n-1} are non-existing for n=0

pp. 342, l. Algorithm 13, and -8 and -4, *maximal genuine* $\rhd$ maximal proper

pp. 343, l. 4, *subformula of* Ψ $\rhd$ subformula of Ψ'

pp. 345, l. -2, $Sat(\exists(\Phi \, \mathsf{U} \, \Psi)$ $\rhd$ $Sat(\exists(\Phi \, \mathsf{U} \, \Psi))$

pp. 345, l. proof of (g)(ii), *Let* $\pi = s_0 s_1 s_2 \ldots$ *be a path starting in* s=s_0. $\rhd$ Delete.

pp. 349, l. -9, $(a = c) \wedge (a \neq b)$ $\rhd$ $(a \leftrightarrow c) \wedge (a \not\leftrightarrow b)$

pp. 351, l. Algorithm 15, $\rhd$ comments in the first two lines of algorithm need to be swapped while replacing E by T and T by E

pp. 354, l. Example 6.28, *see the gray states* $\rhd$ Delete.

pp. 354, l. Example 6.28, *Figure 6.13(b), Figure 6.13(c)* $\rhd$ Figure 6.13(c), Figure 6.13(d)

pp. 358, l. 11, $\rhd$ Note that the length of $\Phi_n \in \mathcal{O}(n!)$

pp. 371, l. -6, *ifstatement* $\rhd$ if statement

pp. 372, l. Algorithm 19, line 4, $C \cap Sat(b_j) \neq \varnothing$ $\rhd$ $C \cap Sat(b_i) \neq \varnothing$

pp. 378, l. -6, *Eaxmple* $\rhd$ Example

pp. 383, l. 9 and 10, $\ldots z_m$ $\rhd$ $\ldots, z_m$

pp. 386, l. 6, $y_1 \ \vee \ y_2$ $\rhd$ $y_2 \ \vee \ y_1$

pp. 386, l. 6, $y_1 \wedge y_2$ $\rhd$ $y_2 \wedge y_1$

pp. 386, l. 13 and 15 (twice), $s\{\overline{y} \leftarrow \overline{z}\}$ $\rhd$ $s\{\overline{z} \leftarrow \overline{y}\}$

pp. 386, l. 15–17, $f\{\overline{z} \leftarrow \overline{y}\}$ $\rhd$ $f\{\overline{y} \leftarrow \overline{z}\}$

pp. 387, l. 18, $t\{\bar{x}/\bar{x}'\}$ $\rhd$ $t\{\bar{x}' \leftarrow \bar{x}\}$

pp. 388, l. 7, x' $\rhd$ x'_1

pp. 388, l. 7, $\bigwedge_{j<i\leqslant n}(x_j \leftrightarrow x'_j)$ $\rhd$ $(\neg x_1 \rightarrow x'_1) \ \wedge \ \bigwedge_{i<j\leqslant n}(x_j \leftrightarrow x'_j)$

pp. 388, l. 14–17, $\rhd$ x and x' should be swapped

pp. 388, l. Example 6.58 (four times), $\{x \leftarrow x'\}$ $\rhd$ $\{x' \leftarrow x\}$

pp. 390, l. 8, $\exists s' \in S s.t. s' \in Post(s)$ ▷ $\exists s' \in S.\, s' \in Post(s)$

pp. 390, l. Algorithm 20, line 4, $f_{j+1}(\bar{x}) := f_{j+1}(\bar{x}) \vee \ldots$ ▷ $f_{j+1}(\bar{x}) := f_j(\bar{x}) \vee \ldots$

pp. 391, l. Algorithm 21, line 4, $f_{j+1}(\bar{x}) := f_{j+1}(\bar{x}) \wedge \ldots$ ▷ $f_{j+1}(\bar{x}) := f_j(\bar{x}) \wedge \ldots$

pp. 393, l. Figure 6.21 (right), *solid line between* z_3 *and 0* ▷ dashed line between z_3 and 0

pp. 396, l. -15, *The semantics* ▷ The semantics of

pp. 398, l. 9, *left subtree* ▷ right subtree

pp. 393, l. Figure 6.21, right, *solid line* z_3 *between* 0 ▷ dashed line z_3 between 0

pp. 405, l. 2, $z_m = a_m, z_m = b_m, \ldots, z_i = a_i, z_i = b_i$ ▷ $z_m = a_m, y_m = b_m, \ldots, z_i = a_i, y_i = b_i$

pp. 405, l. 3, $z_m = a_m, z_m = b_m, \ldots, z_{i+1} = a_{i+1}, z_{i+1} = b_{i+1}, z_i = a_i$ ▷ $z_m = a_m, y_m = b_m, \ldots, z_{i+1} = a_{i+1}, y_{i+1} = a_{i+1}, z_i = a_i$

pp. 405, l. -4, *As f* $\overline{b}$, $\overline{\mathfrak{c}} \in \{0,1\}^m$ ▷ As $\overline{b}$, $\overline{\mathfrak{c}} \in \{0,1\}^m$

pp. 409, l. -12, $info(v) = \langle var(v), succ_0(v), succ_0(v)\rangle$ ▷ $info(v) = \langle var(v), succ_1(v), succ_0(v)\rangle$

pp. 412, l. 7, u ▷ v

pp. 413, l. 13, $f_2 z_1 = b_1, \ldots, z_i = b_i$ ▷ $f_2|_{z_1=b_1,\ldots,z_i=b_i}$

pp. 417, l. heading Algorithm 24, $(v, \{\overline{x} \leftarrow \overline{x}'\})$ ▷ $(v, \{\overline{x}' \leftarrow \overline{x}\})$

pp. 417, l. Algorithm 24, line 4, *ist* ▷ is a

pp. 417, l. Algorithm 24, ▷ replace z by x

pp. 418, l. -6, $f|_{x=\overline{b}}$ ▷ $f|_{x=b}$

pp. 469, l. Remark 7.19, line 10, $s_2 \models \varphi$, *but* $s_1 \not\models \varphi$ ▷ $s_2 \not\models \neg\varphi$, but $s_1 \models \neg\varphi$

Chapter 7: Equivalences and Abstraction

pp. 454, l. 3, *Sssume* ▷ Assume

pp. 466, l. 8, $H = Act_1 \cap Act_2$ ▷ $H = (Act_1 \cap Act_2) \setminus \{\tau\}$

pp. 498, l. Algorithm 32, line 6+7, ▷ these lines need to be swapped

pp. 513, l. 9, $\{a\}\varnothing \notin Traces(TS_1)$ ▷ $\{a\}\varnothing \notin Traces(TS_2)$

pp. 518, l. 8, $\forall\Phi \in \forall CTL^*$ ▷ $\forall\Phi \in \forall$CTL

pp. 519, l. -10, *fragment of CTL** ▷ fragment of CTL

pp. 528, l. -9, $s_1 \in Pre(s_2')$ ▷ $s_1 \in Pre(s_1')$

pp. 537, l. -5, $\langle c_2, n_1 \rangle$ ▷ $\langle n_1, c_2 \rangle$

pp. 539, l. 2, $\mathcal{R}$ *on* $(S_1 \times S_2) \cup (S_1 \times S_2)$ ▷ $\mathcal{R}$ on $TS_1 \oplus TS_2$

pp. 542, l. 5, $\langle c_2, n_1 \rangle$ ▷ $\langle n_1, c_2 \rangle$

pp. 546, l. 13, s_2 *is* $\approx_{TS}^{div}$*-divergent whereas* s_0 *and* s_1 *are not.* ▷ s_2 is not $\approx_{TS}^{div}$-divergent whereas s_0 and s_1 are.

pp. 562, l. 1, *and* $s_1 \exists \varphi$ ▷ and $s_1 \models \exists \varphi$

pp. 563, l. 4, $\Phi_B \mathsf{U} \Phi_C$ *is a* $CTL_{\setminus \bigcirc}$ *formula* ▷ $\exists(\Phi_B \mathsf{U} \Phi_C)$ is a $\mathrm{CTL}_{\setminus \bigcirc}$ formula

pp. 566, l. 16, $\ell_2 : \langle \textbf{if } (\mathit{free} > 0) \textbf{ then } i := 0; \mathit{free}{-}{-} \textbf{ fi} \rangle$ ▷ $\ell_2 : \langle \textbf{if } (\mathit{free} > 0) \textbf{ then } i := 0; \mathit{free}{-}{-} \textbf{ fi} \rangle \,;\, \textbf{goto } \ell_0$

pp. 566, l. -3, $\langle \ell_0, \ell_2', 2, 0, 0 \rangle \rightarrow \langle \ell_0, \ell_0', 2, 0, 0 \rangle$ ▷ $\langle \ell_1, \ell_2', 2, 0, 0 \rangle \rightarrow \langle \ell_1, \ell_0', 2, 0, 0 \rangle$

pp. 569, l. 7, *there are some states in* B *that cannot reach* C *by only visiting states in* B*. For such states, the only possibility is to reach* C *via some other block* $D \neq B, C$*.* ▷ C can only be reached via paths that entirely go through B.

pp. 569, l. -5, $B \cap \mathit{Pre}_{\Pi}^{*}(C)$ ▷ $B \cap \mathit{Pre}(C)$

pp. 572, l. 11, $t \in \mathit{Exit}(B)$ ▷ $t \in \mathit{Bottom}(B)$

pp. 578, l. item 3., *self-loops* $[s]_{div} \rightarrow [s]_{div}$ ▷ self-loops $[s] \rightarrow [s]$

Chapter 9: Timed Automata

pp. 699, l. -3, $\forall \Diamond^{>2} \neg on$ ▷ $\forall \Diamond^{\leqslant 2} \neg on$

Chapter 10: Probabilistic Systems

pp. 778, l. 4, $\mathbf{P}'(s,t) = \ldots$ ▷

$$\mathbf{P}'(s,t) = \begin{cases} 1 & \text{if } s = t \text{ and } s \in B \cup S \setminus (C \cup B) \\ 0 & \text{if } s \neq t \text{ and } s \in B \cup S \setminus (C \cup B) \\ \mathbf{P}(s,t) & \text{otherwise.} \end{cases}$$

pp. 857, l. 2, $\sum_{s \in S_? \setminus \{s\}} \mathbf{P}(s, \alpha, t) \cdot x_t$ ▷ $-\sum_{s \in S_? \setminus \{s\}} \mathbf{P}(s, \alpha, t) \cdot x_t$

pp. 870, l. Lemma 10.119, *any* $s \in S$ ▷ any $s \in T$

pp. 876, l. 11, $U_{\Box\Diamond P}$ ▷ $U_{\Box\Diamond B}$

pp. 903, l. Exercise 10.14, $\varphi = \Box \Diamond a$ ▷ $\varphi = \Diamond \Box a$

pp. 903/904, l. Exercise 10.17, *Markov chain* $\mathcal{M}$ ▷ Markov chain $\mathcal{M}$ where all states are equally labeled

pp. 905, l. Exercise 10.22, ▷ Compute also the values $y_s = Pr^{\max}(s \models C \cup B)$ with $C = S \setminus \{ s_3 \}$ and $B = \{ s_6 \}$

pp. 905, l. Exercise 10.23, *(a), 1. and (b)* ▷ (a), (b), (c)

Appendix

pp. 912, l. footnote, $\sigma = A_1 A_2 A_3 \ldots$ ▷ $\sigma = A_0 A_1 A_2 \ldots$

pp. 918, l. 8, *not to 1* ▷ not to n

pp. 925, l. 1, *they are composed of simple paths* ▷ they are composed of paths, at least one of which is simple.