Errata

Essentials of Constraint Programming

Computer Science – Monograph (English)

Thom Frühwirth and Slim Abdennadher
University of Ulm, Germany
Ludwig Maximilian University of Munich

Springer, ISBN 3-540-67623-6

Typically, page and line numbers are given to localize the error. A negative number indicates numbering from the bottom up.
Errors are classified according to four severity levels:

Level 1. A minor typographical error that should not affect your understanding.
Level 2. A minor technical or expository error.
Level 3. A more significant technical or expository error.
Level 4. A serious error.

Page 15, def. 4.1.4, line 2..3, level 3. *Change* ⌊A derivation is *infinite* if it does not have a final state.⌋ *to* ⌈An infinite sequence of states and transitions $S_1 \mapsto S_2 \mapsto S_3 \mapsto \ldots$ is called an *infinite derivation.*⌉.
Page 15, line –3..–2, level 4. *Change* ⌊A *fresh variant* of a clause is a renaming of this clause with variables that do not previously occur in P.⌋ *to* ⌈A *fresh variant* of a clause is a renaming of this clause with new variables that do not occur in P and $<G,\theta>$.⌉.
Page 20, par. after def. 4.2.3, line 1..2, level 3. *Change* ⌊the next three axioms⌋ *to* ⌈the next two axioms⌉.
Page 28, line –7..–6, level 4. *Change* ⌊The second clause yields the same answer.⌋ *to* ⌈The second clause yields a more general answer, $2\leq\mathtt{A}$.⌉.
Page 36, par. –2, line 1..2, level 4. *Change* ⌊A process `mults(S,N,SN)` delays (suspends) until the first argument `S` is a sequence with a known first element.⌋ *to* ⌈A process `mults(S,N,SN)` delays (suspends) until the first argument `S` is known to be a non-empty sequence.⌉.
Page 38, fig. 6.6, Unfold, line 1, level 3. *Change* ⌊$(B \leftarrow D_1 : D_2 \mid H)$ is a fresh variant of a clause in P⌋ *to* ⌈$(B \leftarrow D_1 : D_2 \mid H)$ is a fresh variant of a clause in P with variables $\bar{x}$⌉.
Page 39, par. 3, line 1, level 4. *Remove* ⌊The soundness results also applies to deadlocked states.⌋.
Page 39, theorem 6.3.2, line 3, level 3. *Remark:* ⌊consistent⌋ *is considered equal to* ⌈satisfiable⌉.
Page 43, line –3, level 3. *Change* ⌊the guard C⌋ *to* ⌈the guard D⌉.
Page 44, example 7.1.2, derivation, line 1..5, level 4.

$$\textit{Replace with}\ \lceil \begin{array}{ll} & <\underline{\mathtt{A}\leq\mathtt{B}} \wedge \underline{\mathtt{C}\leq\mathtt{A}} \wedge \mathtt{B}\leq\mathtt{C},\ \mathtt{true}> \\ \mapsto_{\textbf{Propagate}\ (r3)} & <\mathtt{A}\leq\mathtt{B} \wedge \mathtt{C}\leq\mathtt{A} \wedge \underline{\mathtt{B}\leq\mathtt{C}} \wedge \underline{\mathtt{C}\leq\mathtt{B}},\ \mathtt{true}> \\ \mapsto_{\textbf{Simplify}\ (r2)} & <\mathtt{A}\leq\mathtt{B} \wedge \mathtt{B}\leq\mathtt{A} \wedge \underline{\mathtt{B}\doteq\mathtt{C}},\ \mathtt{true}> \\ \mapsto_{\textbf{Solve}} & <\underline{\mathtt{A}\leq\mathtt{B}} \wedge \underline{\mathtt{B}\leq\mathtt{A}},\ \mathtt{B}\doteq\mathtt{C}> \\ \mapsto_{\textbf{Simplify}\ (r2)} & <\underline{\mathtt{A}\doteq\mathtt{B}},\ \mathtt{B}\doteq\mathtt{C}> \\ \mapsto_{\textbf{Solve}} & <\top,\mathtt{A}\doteq\mathtt{B} \wedge \mathtt{B}\doteq\mathtt{C}> \end{array} \rceil.$$

Page 45, par. –3, line 1..3, level 3. *Move* ⌊In CLP, we had two answers for this goal, where one answer was a generalization of the other one.⌋ *to the end of paragraph 5.*
Page 46, par. 4, line 1..3, level 3. *Replace with* ⌈Given a CHR program P we call the conjunction of the logical readings of its rules $\mathcal{P}$. The logical reading of a CHR program P is $\mathcal{P}$ united with a *constraint theory CT* that defines the built-in constraint symbols.⌉.
Page 49, line –3..–1, level 3. *Change* ⌊Similar in spirit to the **UnfoldSplit** rule in CLP, we introduce the following additional transition **Split** for CHR$^\vee$, so we can deal with disjunction $\vee$ in Fig. 7.3.⌋ *to* ⌈Similar in spirit to the **UnfoldSplit** rule in LP, we define the following additional transition **Split** for CHR$^\vee$ (Fig. 7.3.), which introduces search.⌉.
Page 49, fig. 7.3, line 1, level 4.
Change ⌊$<H_1 \wedge G, C> \mid <H_2 \wedge G, C>$⌋ *to* ⌈$<H_1 \wedge G, C> \dot{\vee} <H_2 \wedge G, C>$⌉.

Page 56, line −4, level 4. *Change* ⌊into $X{\leq}4$, $2{*}X{=}6$ into $X{=}3$⌋ *to* ⌈into $X{\leq}2$ and $2{*}X{=}6$ into $X{=}3$⌉.

Page 57, Variable projection, line −3..−1, level 4. *Change* ⌊elimination of Y in $\exists Y (X_1{<}Y \wedge \ldots \wedge X_m{<}Y \wedge Y_1{<}Z \wedge \ldots \wedge Y_n{<}Z)$ yields $n{*}m$ constraints of the form $X_i{<}Y_j$⌋ *to* ⌈elimination of Y in $\exists Y (X_1{<}Y \wedge \ldots \wedge X_m{<}Y \wedge Y{<}Z_1 \wedge \ldots \wedge Y{<}Z_n)$ yields $n{*}m$ constraints of the form $X_i{<}Z_j$⌉.

Page 60, par. 6, line −1, level 4. *Change* ⌊$R{=}Z{+}F \ \wedge \ F{=}4$⌋ *to* ⌈$R{=}Z{+}Y \ \wedge \ Z{=}Y{+}Y$⌉.

Page 62, line 4, level 3. *Change* ⌊Rules are terminated with a period `'.'`.⌋ *to* ⌈Rules start with an optional `'Name @ '` and are terminated with a period `'.'`.⌉.

Page 62, par. −1, line 1..4, level 4. *Move paragraph below first paragraph on page 53.*

Page 64, par. 4, line 1..3, level 3. *Replace with* ⌈As the allowed atomic constraints $\neg X = Z$ and $X \odot Y = Z$ correspond to Boolean functions, we call the arguments X and Y *inputs* and Z *output.*⌉.

Page 71, par. −1, line 1, level 3. *Insert* ⌈Each rule has an optional name followd by the symbol `'@'`.⌉. *Cf. page 62.*

Page 82, par. 4, line 2, level 3. *Change* ⌊`S le 0, mortgage(100000,T,0.01,1025,0)`⌋ *to* ⌈`mortgage(100000,T,0.01,1025,0)`⌉.

Page 95, par. −2, line −2, level 4. *Change* ⌊`Y in 2.33`⌋ *to* ⌈`Y in 2.33..4`⌉.

Page 114, par. 2, line 3..6, level 3.
Replace with ⌈the constraint `sum(C0..C0+C1*X1+C2*X2+...+Cn*Xn+0=Y)`. The constant c_0 is replaced by the interval `C0..C0` and the summand `0` is introduced to end the summation. A constraint of the form `sum(Min..Max+Rest=Y)` means that the interval `Min..Max` plus the sum of the polynomial `Rest` gives an interval for⌉.