Literatur Transitions-Systeme Linear-Time Properties

Dekomposition von Linear-Time Properties

Stefan Walter

Universität Leipzig

30. November 2017

Literatur

Christel Baier und Joost-Pieter Katoen. *Principles of Model Checking (Representation and Mind Series)*. The MIT Press, 2008. ISBN: 026202649X, 9780262026499.

Motivation

$$\begin{tikzcd}
\boxed{\text{System}} \arrow[r, Leftrightarrow] & \boxed{\text{Eigenschaft}} \\
\boxed{\text{Transitions-System}} \arrow[u, "\text{modelliert}"] & \boxed{\text{Linear-Time-Property}} \arrow[u, "\text{modelliert}"]
\end{tikzcd}$$

Transitions-System

Definition (Transitions-System TS)

$TS = (S, Act, \rightarrow, I, AP, L)$ mit:

S	Menge von Zuständen (States)
Act	Menge von Aktionen (Actions)
$\rightarrow \subseteq S \times Act \times S$	Transitions- (Übergangs-) Relation
$I \subseteq S$	Menge von initialen Zuständen
AP	Menge von atomaren Aussagen (atomic propositions)
$L : S \rightarrow 2^{AP}$	Labeling-Funktion

Bemerkungen:

- TS endlich, gdw. S, Act und AP endlich
- Notation: $s \xrightarrow{\alpha} s'$ gdw. $(s, \alpha, s') \in \rightarrow$

Transitions-System

Definition (Transitions-System TS)

$TS = (S, Act, \to, I, AP, L)$ mit:

S	Menge von Zuständen (States)
Act	Menge von Aktionen (Actions)
$\to\ \subseteq\ S \times Act \times S$	Transitions- (Übergangs-) Relation
$I \subseteq S$	Menge von initialen Zuständen
AP	Menge von atomaren Aussagen (atomic propositions)
$L : S \to 2^{AP}$	Labeling-Funktion

Beispiele TS - Ampel

Beispiel (vereinfachte Ampel)

$TS_{Ampel} = (S, Act, \to, I, AP, L)$ mit:

- $Act = \{$r-g, g-r$\}$
- $AP = \{$fahren$\}$

Transitions-System

Definition (Transitions-System TS)

$TS = (S, Act, \rightarrow, I, AP, L)$ mit:

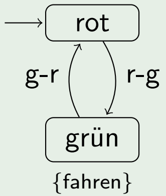
$$
\begin{array}{rl}
S & \text{Menge von Zuständen (States)} \\
Act & \text{Menge von Aktionen (Actions)} \\
\rightarrow \subseteq S \times Act \times S & \text{Transitions- (Übergangs-) Relation} \\
I \subseteq S & \text{Menge von initialen Zuständen} \\
AP & \text{Menge von atomaren Aussagen (atomic propositions)} \\
L : S \rightarrow 2^{AP} & \text{Labeling-Funktion}
\end{array}
$$

Beispiele TS - Ampel

Beispiel (vereinfachte Ampel)

$TS_{Ampel} = (S, Act, \rightarrow, I, AP, L)$ mit:

- $Act = \{$r-g, g-r$\}$
- $AP = \{$fahren$\}$



Transitions-System

Definition (Transitions-System TS)

$TS = (S, Act, \rightarrow, I, AP, L)$ mit:

S	Menge von Zuständen (States)
Act	Menge von Aktionen (Actions)
$\rightarrow \subseteq S \times Act \times S$	Transitions- (Übergangs-) Relation
$I \subseteq S$	Menge von initialen Zuständen
AP	Menge von atomaren Aussagen (atomic propositions)
$L : S \rightarrow 2^{AP}$	Labeling-Funktion

Nachfolger

Definition (Nachfolger)

$TS = (S, Act, \rightarrow, I, AP, L)$ und $s \in S$
Definiere:

$$\mathrm{Post}(s) = \left\{ s' \in S \mid \exists \alpha \in Act.\ s \xrightarrow{\alpha} s' \right\}$$

Transitions-System

Definition (Transitions-System TS)

$TS = (S, Act, \to, I, AP, L)$ mit:

S	Menge von Zuständen (States)
Act	Menge von Aktionen (Actions)
$\to \subseteq S \times Act \times S$	Transitions- (Übergangs-) Relation
$I \subseteq S$	Menge von initialen Zuständen
AP	Menge von atomaren Aussagen (atomic propositions)
$L : S \to 2^{AP}$	Labeling-Funktion

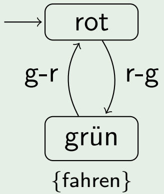
Nachfolger

Definition (Nachfolger)

$TS = (S, Act, \to, I, AP, L)$ und $s \in S$
Definiere:

$$\mathrm{Post}(s) = \left\{ s' \in S \mid \exists \alpha \in Act.\ s \xrightarrow{\alpha} s' \right\}$$

Beispiel



- $\mathrm{Post}(\text{rot}) = \{\text{grün}\}$
- $\mathrm{Post}(\text{grün}) = \{\text{rot}\}$

Transitions-System

Definition (Transitions-System TS)

$TS = (S, Act, \rightarrow, I, AP, L)$ mit:

S	Menge von Zuständen (States)
Act	Menge von Aktionen (Actions)
$\rightarrow \subseteq S \times Act \times S$	Transitions- (Übergangs-) Relation
$I \subseteq S$	Menge von initialen Zuständen
AP	Menge von atomaren Aussagen (atomic propositions)
$L : S \rightarrow 2^{AP}$	Labeling-Funktion

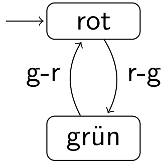
Terminal State

Definition (Terminal State - Endzustand)

$TS = (S, Act, \rightarrow, I, AP, L)$ und $s \in S$

Definiere: s heißt Terminal State gdw. $\mathrm{Post}(s) = \emptyset$

Pfade und Traces - Beispiel



- $\pi = \big(\text{rot grün}\big)^{\omega}$
- $\hat{\pi} = \text{grün rot grün}$

Pfade - Definition

Definition (Pfadfragment)

$TS = (S, Act, \rightarrow, I, AP, L)$

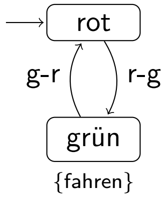
Unendliches Pfadfragment $\pi = s_0 s_1 \ldots \in S^{\omega}$:

$$\text{mit } \forall i > 0.\ s_i \in \mathrm{Post}(s_{i-1})$$

Endliches Pfadfragment $\hat{\pi} = s_0 s_1 \ldots s_n \in S^{*}, \quad n \geq 0$:

$$\text{mit } \forall 0 < i \leq n.\ s_i \in \mathrm{Post}(s_{i-1})$$

Pfade und Traces - Beispiel



- $\pi = \big(\text{rot grün}\big)^{\omega}$ maximal, $\pi \in \mathrm{Paths}(\text{rot})$
- $\hat{\pi} =$ grün rot grün

Pfade - Definition

Definition (Pfadfragment)

$TS = (S, Act, \rightarrow, I, AP, L)$

Unendliches Pfadfragment $\pi = s_0 s_1 \ldots \ \in S^{\omega}$:

$$\text{mit } \forall i > 0.\ s_i \in \mathrm{Post}(s_{i-1})$$

Endliches Pfadfragment $\hat{\pi} = s_0 s_1 \ldots s_n \ \in S^{*}, \quad n \geq 0$:

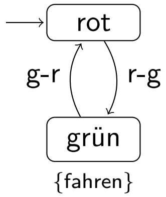
$$\text{mit } \forall 0 < i \leq n.\ s_i \in \mathrm{Post}(s_{i-1})$$

Maximales Pfadfragment π_{max}:

$\hat{\pi}$ mit Terminal State s_n; oder π

Für $s \in S$ definiere: $\mathrm{Paths}(s) = \{\pi_{max} \mid s_0 = s\}$

Pfade und Traces - Beispiel



- $\pi = \big(\text{rot grün}\big)^\omega$ maximal, $\pi \in \mathrm{Paths}(\text{rot})$
  $\mathrm{trace}(\pi) = \big(\{\}\{\text{fahren}\}\big)^\omega$
- $\hat{\pi} =$ grün rot grün
  $\mathrm{trace}(\hat{\pi}) = \{\text{fahren}\}\{\}\{\text{fahren}\}$

Traces - Definition

Definition (Trace und Trace-Fragment)

$TS = (S, Act, \to, I, AP, L)$ ohne Terminal States

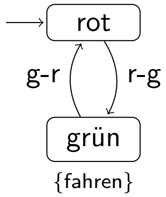
- unendliches Pfadfragment $\pi = s_0 s_1 \cdots \in S^\omega$

  definiere: $\mathrm{trace}(\pi) = L(s_0)L(s_1) \cdots \in (2^{AP})^\omega$

- endliches Pfadfragment $\hat{\pi} = s_0 s_1 \dots s_n \in S^*$

  definiere: $\mathrm{trace}(\hat{\pi}) = L(s_0)L(s_1) \dots L(s_n) \in (2^{AP})^*$

Pfade und Traces - Beispiel



- $\pi = \big(\text{rot grün}\big)^\omega$ maximal, $\pi \in \text{Paths}(\text{rot})$
  $\text{trace}(\pi) = \big(\{\}\{\text{fahren}\}\big)^\omega$
  $\in \text{Traces}(\text{rot})$, $\quad \in \text{Traces}(TS_{Ampel})$
- $\hat{\pi} =$ grün rot grün
  $\text{trace}(\hat{\pi}) = \{\text{fahren}\}\{\}\{\text{fahren}\}$

Traces - Definition

Definition (Trace und Trace-Fragment)

$TS = (S, Act, \to, I, AP, L)$ ohne Terminal States

- unendliches Pfadfragment $\pi = s_0 s_1 \cdots \in S^\omega$

  definiere: $\quad \text{trace}(\pi) = L(s_0)L(s_1)\cdots \in (2^{AP})^\omega$

- endliches Pfadfragment $\hat{\pi} = s_0 s_1 \ldots s_n \in S^*$

  definiere: $\quad \text{trace}(\hat{\pi}) = L(s_0)L(s_1)\ldots L(s_n) \in (2^{AP})^*$

Erweiterung auf:

- Mengen von Pfaden Π: $\quad \text{trace}(\Pi) = \bigcup_{\pi \in \Pi} \text{trace}(\pi)$
- $s \in S$: $\quad \text{Traces}(s) = \text{trace}\big(\text{Paths}(s)\big)$
- TS: $\quad \text{Traces}(TS) = \bigcup_{s \in I} \text{Traces}(s)$

Linear-Time Property

Motivation: Formalisierung Eigenschaften eines TS

Definition (Linear-Time Property P über AP)

$$P \subseteq (2^{AP})^\omega$$

Literatur Transitions-Systeme Linear-Time Properties
Einstieg

Linear-Time Property

Motivation: Formalisierung Eigenschaften eines TS

Definition (Linear-Time Property P über AP)

$$P \subseteq (2^{AP})^\omega$$

Definition (Erfüllbarkeit / Verifikation von LT-Properties)

$TS = (S, Act, \rightarrow, I, AP, L)$ ohne Terminal States; $s \in S$

TS bzw. s erfüllt (satisfies) P gdw.

$$\text{Traces}(TS) \subseteq P \quad \text{bzw.} \quad \text{Traces}(s) \subseteq P$$

Notation: $TS \models P$ bzw. $s \models P$

Invarianten

Modellierung von z. B. Mutual Exclusion, Deadlock-Freiheit

Definition (Invariante P_{inv})

LT-Property P_{inv} über AP ist Invariante, gdw.

$\exists$ aussagenlogische Formel Φ über AP.

$$P_{inv} = \left\{ A_0 A_1 A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0.\ A_i \models \Phi \right\}$$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1,2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{\sigma = A_0 A_1 ... \in (2^{AP})^\omega \mid \forall A_i.\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i\right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Beispiel Überprüfen von Invarianten - Tiefensuche

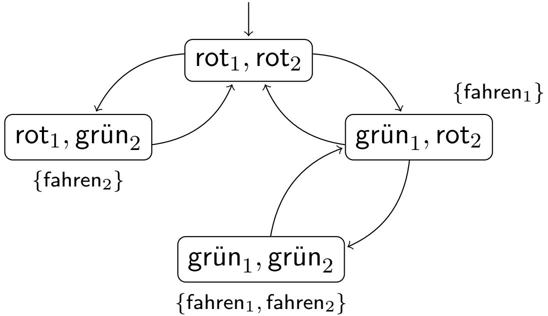


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$R = \{\}$

$U = \$$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1, 2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{ \sigma = A_0 A_1 ... \in (2^{AP})^\omega \mid \forall A_i.\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i \right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Beispiel Überprüfen von Invarianten - Tiefensuche

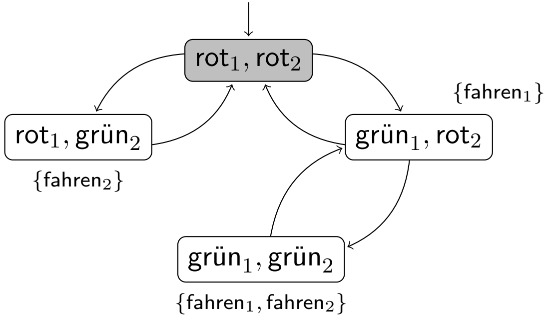


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$R = \{(\text{rot}_1, \text{rot}_2)\}$

$U = \$(\text{rot}_1, \text{rot}_2)$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1, 2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{\sigma = A_0 A_1 ... \in (2^{AP})^\omega \mid \forall A_i.\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i\right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Beispiel Überprüfen von Invarianten - Tiefensuche

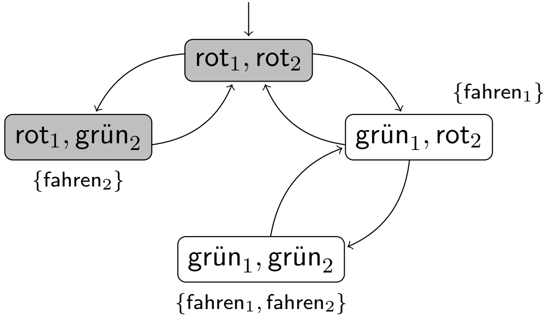


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$$R = \{(\text{rot}_1, \text{rot}_2), (\text{rot}_1, \text{grün}_2)\}$$

$$U = \$(\text{rot}_1, \text{rot}_2)(\text{rot}_1, \text{grün}_2)$$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1,2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{\sigma = A_0A_1... \in (2^{AP})^\omega \mid \forall A_i.\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i\right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Beispiel Überprüfen von Invarianten - Tiefensuche

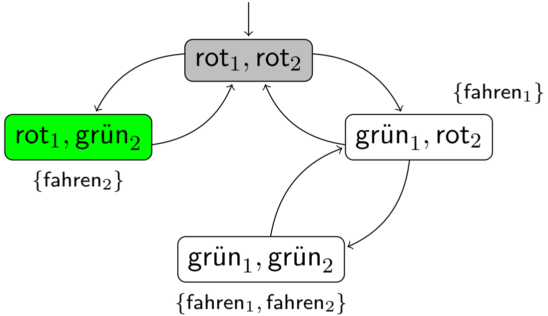


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$$R = \{(\text{rot}_1, \text{rot}_2), (\text{rot}_1, \text{grün}_2)\}$$

$$U = \$(\text{rot}_1, \text{rot}_2)$$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1,2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{\sigma = A_0A_1... \in (2^{AP})^\omega \mid \forall A_i.\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i\right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Literatur Transitions-Systeme Linear-Time Properties
Invarianten

Beispiel Überprüfen von Invarianten - Tiefensuche

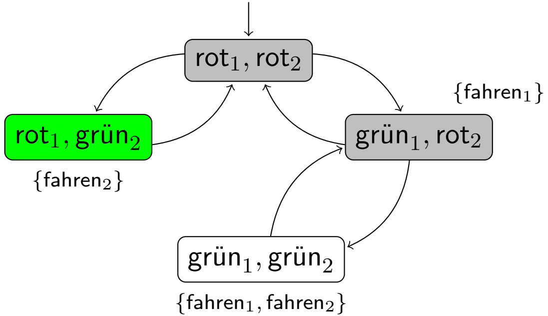


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$$R = \{(\text{rot}_1, \text{rot}_2), (\text{rot}_1, \text{grün}_2), (\text{grün}_1, \text{rot}_2)\}$$

$$U = \$(\text{rot}_1, \text{rot}_2)(\text{grün}_1, \text{rot}_2)$$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1, 2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{\sigma = A_0 A_1 ... \in (2^{AP})^\omega \mid \forall A_i.\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i\right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Beispiel Überprüfen von Invarianten - Tiefensuche

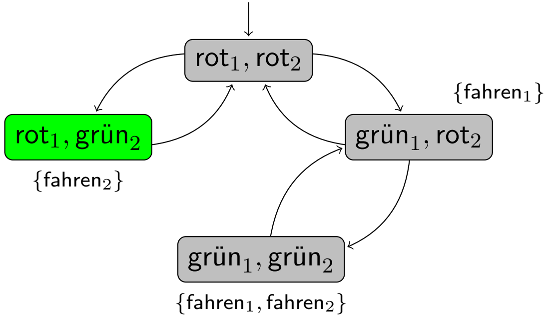


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$$R = \{(\text{rot}_1, \text{rot}_2), (\text{rot}_1, \text{grün}_2), (\text{grün}_1, \text{rot}_2)(\text{grün}_1, \text{grün}_2)\}$$

$$U = \$(\text{rot}_1, \text{rot}_2)(\text{grün}_1, \text{rot}_2)(\text{grün}_1, \text{grün}_2)$$

Beispiel Überprüfen von Invarianten - Tiefensuche

Bemerkung: Algorithmus gilt nur für endliche TS

Beispiel

Ampelkreuzung mit zwei Ampeln und $AP = \{\text{fahren}_i\}, \quad i \in \{1, 2\}$

In keinem Zustand dürfen beide Richtungen gleichzeitig fahren.

$$P_{inv} = \left\{\sigma = A_0 A_1 ... \in (2^{AP})^\omega \mid \forall A_i .\ \{\text{fahren}_1, \text{fahren}_2\} \not\subseteq A_i\right\}$$

R - Menge besuchter Zustände
U - Stack; aktuelle Position/Pfad-Fragment

eingefärbt $\in R$ (grün [rot]: Invariante gilt [nicht])

Beispiel Überprüfen von Invarianten - Tiefensuche

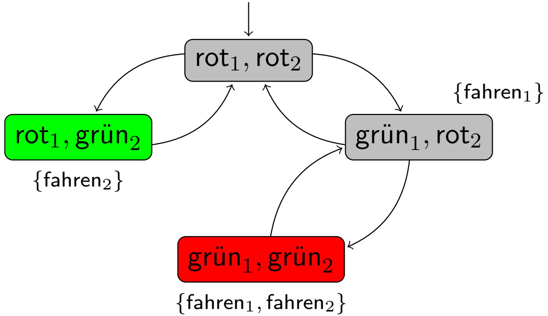


Abbildung: bewusst falsch modellierte Ampelkreuzung (ohne Act)

$$R = \{(\text{rot}_1, \text{rot}_2), (\text{rot}_1, \text{grün}_2), (\text{grün}_1, \text{rot}_2)(\text{grün}_1, \text{grün}_2)\}$$

$$U = \$(\text{rot}_1, \text{rot}_2)(\text{grün}_1, \text{rot}_2)(\text{grün}_1, \text{grün}_2)$$

Präfix

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\text{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\text{pref}(P) = \bigcup_{\sigma \in P} \text{pref}(\sigma)$

Präfix

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \cdots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \ldots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Beispiel (Präfix)

Sei $AP = \{\text{fahren}\}$.
Betrachte $\sigma = (\{rot\}\{grün\})^\omega$

$$\mathrm{pref}(\sigma) = \Big\{\epsilon, \{\}, \{\}\{\text{fahren}\}, \{\}\{\text{fahren}\}\{\}, \ldots\Big\}$$

Safety Property

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \text{pref}(\sigma). \quad \left(\hat{\sigma} \notin \text{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

$\neg P_{safe} \implies \exists$ endliches Trace-Fragment, das P_{safe} verletzt

Safety Property - Beispiel

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Safety Property - Beispiel

Beispiel

Computer mit $AP = \{\text{an, aus, startet, beendet}\}$

Direkt nach dem Herunterfahren soll der Computer aus sein:

$$P_{safe} = \left\{ A_0 A_1 \ldots \mid \forall A_i.\ \{\text{beendet}\} \subseteq A_i \Rightarrow \{\text{aus}\} \subseteq A_{i+1} \right\}$$

Beispiel bad prefix: $\{\text{an}\}\{\text{beendet}\}\{\text{an}\}$

Liveness Property

Modellierung von z. B. ... (mit $\{ap\} \in AP$)

- Eventualität: $P = \{A_0 A_1 \ldots \mid \exists i \geq 0.\ ap \in A_i\}$
- wiederholte Eventualität:
  $P = \{A_0 A_1 \ldots \mid \forall i \geq 0.\ \exists j \geq i.\ ap \in A_j\}$

Literatur Transitions-Systeme Linear-Time Properties
Liveness Properties

Liveness Property

Modellierung von z. B. ... (mit $\{ap\} \in AP$)

- Eventualität: $P = \{A_0 A_1 \ldots \mid \exists i \geq 0.\ ap \in A_i\}$
- wiederholte Eventualität:
  $P = \{A_0 A_1 \ldots \mid \forall i \geq 0.\ \exists j \geq i.\ ap \in A_j\}$

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

$\neg P_{live} \implies \exists$ unendliches Trace-Fragment, das P_{live} verletzt

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

$\Longrightarrow$ alternative Definition für Safety und Liveness: (ohne Beweis)

$$\mathrm{closure}(P_{safe}) = P_{safe}$$

$$\mathrm{closure}(P_{live}) = (2^{AP})^\omega$$

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe}) \right) \right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

$\Longrightarrow$ alternative Definition für Safety und Liveness: (ohne Beweis)

$$\mathrm{closure}(P_{safe}) = P_{safe}$$

$$\mathrm{closure}(P_{live}) = (2^{AP})^\omega$$

Lemma (Distributivität von Vereinigung und closure)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:*

$$\mathrm{closure}(P) \cup \mathrm{closure}(P') = \mathrm{closure}(P \cup P')$$

(ohne Beweis)

Dekompositions-Theorem

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Dekompositions-Theorem

Motivation: separate Verifikation einfacher

Satz (Dekomposition von Linear-Time Properties)

$\forall P \subseteq (2^{AP})^\omega.\ \exists P_{safe} \subseteq (2^{AP})^\omega$ *und* $P_{live} \subseteq (2^{AP})^\omega$

P_{safe} *ist Safety P. und* P_{live} *ist Liveness P., sodass*

$$P = P_{safe} \cap P_{live}$$

Dekompositions-Theorem Beweis

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\text{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\text{pref}(P) = \bigcup_{\sigma \in P} \text{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Dekompositions-Theorem Beweis

Beweis.

Sei $P \subseteq (2^{AP})^\omega$

Es gilt: $P \subseteq \text{closure}(P) \subseteq (2^{AP})^\omega$

Dekompositions-Theorem Beweis

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \cdots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \ldots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

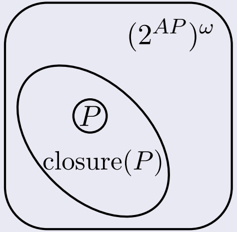
Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Dekompositions-Theorem Beweis

Beweis.

Sei $P \subseteq (2^{AP})^\omega$

Es gilt: $P \subseteq \mathrm{closure}(P) \subseteq (2^{AP})^\omega$



Dekompositions-Theorem Beweis

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

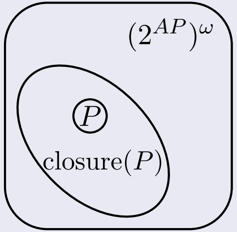
$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

Dekompositions-Theorem Beweis

Beweis.

Sei $P \subseteq (2^{AP})^\omega$

Es gilt: $P \subseteq \mathrm{closure}(P) \subseteq (2^{AP})^\omega$



$$P = \underbrace{\mathrm{closure}(P)}_{=P_{safe}} \cap \underbrace{\Big(P \cup \big((2^{AP})^\omega \setminus \mathrm{closure}(P)\big)\Big)}_{=P_{live}}$$

$\square$

Dekompositions-Theorem Beweis

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \left(\exists \hat{\sigma} \in \text{pref}(\sigma). \left(\hat{\sigma} \notin \text{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Dekompositions-Theorem Beweis

Beweis.

$$P = \underbrace{\text{closure}(P)}_{=P_{safe}} \cap \underbrace{\left(P \cup \left((2^{AP})^\omega \setminus \text{closure}(P)\right)\right)}_{=P_{live}}$$

1. zu zeigen: $P_{safe} = \text{closure}(P)$ ist Safety Property

$$\begin{aligned} \sigma \in (2^{AP})^\omega \setminus \text{closure}(P) &\iff \text{pref}(\sigma) \not\subseteq \text{pref}(P) \\ &\iff \exists \hat{\sigma} \in \text{pref}(\sigma). \ \hat{\sigma} \notin \text{pref}(P) \end{aligned}$$

□

Dekompositions-Theorem Beweis

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\text{pref}(P_{live}) = (2^{AP})^*$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Dekompositions-Theorem Beweis

Beweis.

2. z.z.: $P_{live} = P \cup \left((2^{AP})^\omega \setminus \text{closure}(P)\right)$ ist Liveness Property

   äquivalente Definition: $\text{closure}(P_{live}) = (2^{AP})^\omega$

   "$\subseteq$" $\text{closure}(P_{live}) \subseteq (2^{AP})^\omega$ gilt immer

□

Dekompositions-Theorem Beweis

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\text{pref}(P_{live}) = (2^{AP})^*$$

Lemma (Distributivität von Vereinigung und closure)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:*

$$\text{closure}(P) \cup \text{closure}(P') = \text{closure}(P \cup P')$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Dekompositions-Theorem Beweis

Beweis.

2. z.z.: $P_{live} = P \cup \left((2^{AP})^\omega \setminus \text{closure}(P)\right)$ ist Liveness Property

   äquivalente Definition: $\text{closure}(P_{live}) = (2^{AP})^\omega$

   "$\subseteq$" $\text{closure}(P_{live}) \subseteq (2^{AP})^\omega$ gilt immer

   "$\supseteq$" $\text{closure}(P_{live}) = \text{closure}\left(P \cup ((2^{AP})^\omega \setminus \text{closure}(P))\right)$

   $$\overset{Distr.}{=} \text{closure}(P) \cup \text{closure}\left((2^{AP})^\omega \setminus \text{closure}(P)\right)$$

   $$\underset{\text{Mon. } \cup \text{ bzgl. } \subseteq}{\overset{\text{closure}(P') \supseteq P'}{\supseteq}} \text{closure}(P) \cup \left((2^{AP})^\omega \setminus \text{closure}(P)\right) = (2^{AP})^\omega$$

$\square$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Backup Slides

Sharpest Decomposition

Sharpest Decomposition

Lemma (Sharpest Decomposition)

LT-Property $P \subseteq (2^{AP})^\omega$ *mit* $P = P_{safe} \cap P_{live}$.

P_{safe} *ist Safety P. und* P_{live} *ist Liveness P.*
Es gilt:

1. $\text{closure}(P) \subseteq P_{safe}$
2. $P_{live} \subseteq P \cup \left((2^{AP})^\omega \setminus \text{closure}(P)\right)$

Lemma (Sharpest Decomposition)

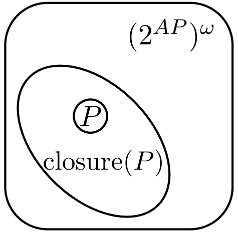
LT-Property $P \subseteq (2^{AP})^\omega$ *mit* $P = P_{safe} \cap P_{live}$.

P_{safe} *ist Safety P. und* P_{live} *ist Liveness P.*
Es gilt:

1. $\text{closure}(P) \subseteq P_{safe}$
2. $P_{live} \subseteq P \cup \left((2^{AP})^\omega \setminus \text{closure}(P)\right)$

Wir verwenden: (teilweise ohne Beweis)

- a. Monotonie von closure und $\subseteq$
- b. alternative Def. Safety
- c. Monotonie von $\setminus$ und $\subseteq$
- d. DeMorgan



Beweis.

1. $$\begin{aligned} \text{closure}(P) &= \text{closure}(P_{safe} \cap P_{live}) \\ &\overset{a}{\subseteq} \text{closure}(P_{safe}) \overset{b}{=} P_{safe} \end{aligned}$$

□

Lemma (Sharpest Decomposition)

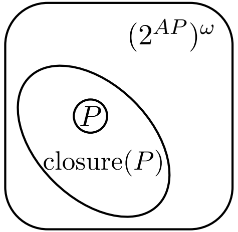
LT-Property $P \subseteq (2^{AP})^\omega$ *mit* $P = P_{safe} \cap P_{live}$.

P_{safe} *ist Safety P. und* P_{live} *ist Liveness P.*
Es gilt:

1. $\mathrm{closure}(P) \subseteq P_{safe}$
2. $P_{live} \subseteq P \cup \left((2^{AP})^\omega \setminus \mathrm{closure}(P)\right)$

Wir verwenden: (teilweise ohne Beweis)

- a) Monotonie von $\mathrm{closure}$ und $\subseteq$
- b) alternative Def. Safety
- c) Monotonie von $\setminus$ und $\subseteq$
- d) DeMorgan



Beweis.

1. $$\mathrm{closure}(P) = \mathrm{closure}(P_{safe} \cap P_{live}) \overset{a}{\subseteq} \mathrm{closure}(P_{safe}) \overset{b}{=} P_{safe}$$
2. Widerspruch: Sei $\sigma \notin P \cup \left((2^{AP})^\omega \setminus \mathrm{closure}(P)\right)$.
   zu zeigen: $\sigma \notin P_{live}$

$$\begin{aligned}(2^{AP})^\omega \setminus \left(P \cup \left((2^{AP})^\omega \setminus \mathrm{closure}(P)\right)\right) &= \mathrm{closure}(P) \setminus P \\ &\overset{1,c}{\subseteq} P_{safe} \setminus (P_{safe} \cap P_{live}) \\ &\overset{d}{=} \left(P_{safe} \setminus P_{safe}\right) \cup \left(P_{safe} \setminus P_{live}\right) \\ &= P_{safe} \setminus P_{live}\end{aligned}$$

Also $\sigma \notin P_{live}$. □

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\text{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\text{pref}(P) = \bigcup_{\sigma \in P} \text{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Lemma (Monotonie von $pref$ und $closure$ bzgl. $\subseteq$)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:* $P \subseteq P' \implies \dots$

1. $\dots \text{pref}(P) \subseteq \text{pref}(P')$
2. $\dots \text{closure}(P) \subseteq \text{closure}(P')$

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\text{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\text{pref}(P) = \bigcup_{\sigma \in P} \text{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Lemma (Monotonie von $pref$ und $closure$ bzgl. $\subseteq$)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:* $P \subseteq P' \implies \dots$

1. $\dots \text{pref}(P) \subseteq \text{pref}(P')$
2. $\dots \text{closure}(P) \subseteq \text{closure}(P')$

Beweis.

Annahme: $P \subseteq P'$

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \cdots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\text{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \ldots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\text{pref}(P) = \bigcup_{\sigma \in P} \text{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Lemma (Monotonie von $pref$ und $closure$ bzgl. $\subseteq$)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:* $\quad P \subseteq P' \implies \ldots$

1. $\ldots \text{pref}(P) \subseteq \text{pref}(P')$
2. $\ldots \text{closure}(P) \subseteq \text{closure}(P')$

Beweis.

Annahme: $P \subseteq P'$ also $\forall \sigma \in P.\ \sigma \in P'$

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

Lemma (Monotonie von $pref$ und $closure$ bzgl. $\subseteq$)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:* $\quad P \subseteq P' \implies \dots$

1. $\dots \mathrm{pref}(P) \subseteq \mathrm{pref}(P')$
2. $\dots \mathrm{closure}(P) \subseteq \mathrm{closure}(P')$

Beweis.

Annahme: $P \subseteq P'$ also $\forall \sigma \in P.\ \sigma \in P'$

1. trivial: Annahme $\overset{Def.}{\iff} \forall \sigma \in P.\ \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P')$
   $\overset{Def.}{\iff} \mathrm{pref}(P) \subseteq \mathrm{pref}(P')$

$\square$

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

Lemma (Monotonie von $pref$ und $closure$ bzgl. $\subseteq$)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:* $\quad P \subseteq P' \implies \dots$

1. $\dots \mathrm{pref}(P) \subseteq \mathrm{pref}(P')$
2. $\dots \mathrm{closure}(P) \subseteq \mathrm{closure}(P')$

Beweis.

Annahme: $P \subseteq P'$ also $\forall \sigma \in P.\ \sigma \in P'$

1. trivial: Annahme $\overset{Def.}{\iff} \forall \sigma \in P.\ \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P')$
   $\overset{Def.}{\iff} \mathrm{pref}(P) \subseteq \mathrm{pref}(P')$
2. $\mathrm{closure}(P) \overset{Def.}{=} \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$
   $\underset{1}{\overset{Ann.}{\subseteq}} \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P')\}$
   $\overset{Def.}{=} \mathrm{closure}(P')$

□

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \dots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

Lemma (Distributivität von Vereinigung und pref)

$\forall P, P' \subseteq (2^{AP})^\omega$ gilt:

$$\mathrm{pref}(P) \cup \mathrm{pref}(P') = \mathrm{pref}(P \cup P')$$

Beweis.

$$\begin{aligned} \hat{\sigma} \in \mathrm{pref}(P) \cup \mathrm{pref}(P') &\iff \exists \sigma \in P \cup P'.\, \hat{\sigma} \in \mathrm{pref}(\sigma) \\ &\iff \hat{\sigma} \in \mathrm{pref}(P \cup P') \end{aligned}$$

□

Beweis Lemma Distributivität Vereinigung closure

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Beweis Lemma Distributivität Vereinigung closure

Lemma (Distributivität von Vereinigung und closure)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:*

$$\mathrm{closure}(P) \cup \mathrm{closure}(P') = \mathrm{closure}(P \cup P')$$

Lemma (Distributivität von Vereinigung und closure)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:*

$$\mathrm{closure}(P) \cup \mathrm{closure}(P') = \mathrm{closure}(P \cup P')$$

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \cdots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\mathrm{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \ldots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\mathrm{pref}(P) = \bigcup_{\sigma \in P} \mathrm{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\mathrm{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P)\}$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Beweis.

"$\subseteq$" $P \subseteq P \cup P' \overset{Mon.}{\Longrightarrow} \mathrm{closure}(P) \subseteq \mathrm{closure}(P \cup P')$, analog
$P' \subseteq P \cup P' \overset{Mon.}{\Longrightarrow} \mathrm{closure}(P') \subseteq \mathrm{closure}(P \cup P')$

also $\mathrm{closure}(P) \cup \mathrm{closure}(P') \subseteq \mathrm{closure}(P \cup P')$

"$\supseteq$" Sei $\sigma \in \mathrm{closure}(P \cup P')$

a. $\overset{Def.}{\Longleftrightarrow} \mathrm{pref}(\sigma) \subseteq \mathrm{pref}(P \cup P') \overset{Dist.}{=} \mathrm{pref}(P) \cup \mathrm{pref}(P')$
umschreiben ergibt:

$$\mathrm{pref}(\sigma) = \underbrace{\Big(\mathrm{pref}(\sigma) \cap \mathrm{pref}(P)\Big)}_{1} \cup \underbrace{\Big(\mathrm{pref}(\sigma) \cap \mathrm{pref}(P')\Big)}_{2}$$

b. $\sigma \in (2^{AP})^\omega \Rightarrow \mathrm{pref}(\sigma)$ unendlich

$\Rightarrow$ 3 Fälle: entweder 1 oder 2 oder beide unendlich $\square$

Lemma (Distributivität von Vereinigung und closure)

$\forall P, P' \subseteq (2^{AP})^\omega$ *gilt:*

$$\text{closure}(P) \cup \text{closure}(P') = \text{closure}(P \cup P')$$

Definition (Präfix)

Trace $\sigma = A_0 A_1 A_2 \cdots \in (2^{AP})^\omega$ und LT-Property $P \subseteq (2^{AP})^\omega$

Definiere:

- $\text{pref}(\sigma) = \{\hat{\sigma} = A_0 A_1 \ldots A_i \in (2^{AP})^* \mid i \geq 0\}$
- $\text{pref}(P) = \bigcup_{\sigma \in P} \text{pref}(\sigma)$

Definition (Closure/Hülle)

LT-Property $P \subseteq (2^{AP})^\omega$; definiere:

$$\text{closure}(P) = \{\sigma \in (2^{AP})^\omega \mid \text{pref}(\sigma) \subseteq \text{pref}(P)\}$$

Beweis.

"$\supseteq$" Gelte o.B.d.A. Fall 1: $\text{pref}(\sigma) \cap \text{pref}(P)$ unendlich

Dann gilt: $\text{pref}(\sigma) \subseteq \text{pref}(P)$
Widerspruchsbeweis: Sei $\hat{\sigma} \in \text{pref}(\sigma) \setminus \text{pref}(P)$ mit $k = |\hat{\sigma}|$

$$\begin{aligned} \text{Fall 2} &\implies \exists \widehat{\sigma'} \in \text{pref}(\sigma) \cap \text{pref}(P).\ |\widehat{\sigma'}| > k \\ &\implies \exists \sigma' \in P.\ \widehat{\sigma'} \in \text{pref}(\sigma') \\ &\implies \hat{\sigma} \in \text{pref}(\widehat{\sigma'}) \subseteq \text{pref}(P) \\ &\implies \hat{\sigma} \in \text{pref}(P) \qquad ⚡ \end{aligned}$$

Per Def.: $\sigma \in \text{closure}(P)$ und $\sigma \in \text{closure}(P) \cup \text{closure}(P')$ $\square$

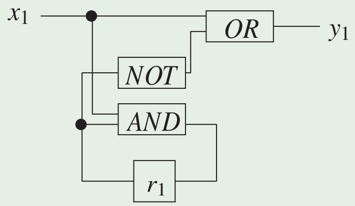
Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Beispiele TS - Schaltwerk

Beispiel (Exercise 2.1 a - TS von Schaltwerken (1))

$y_1 = x_1 \vee \neg r_1$
$\delta_{r_1} = x_1 \wedge r_1$

Annahme (b): initial $r_1 = 0$



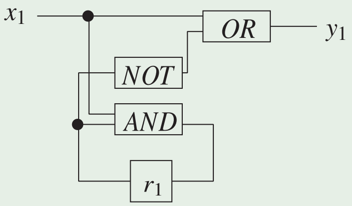
from [1, Exercise 2.1, p. 82]

Beispiele TS - Schaltwerk

Beispiel (Exercise 2.1 a - TS von Schaltwerken (1))

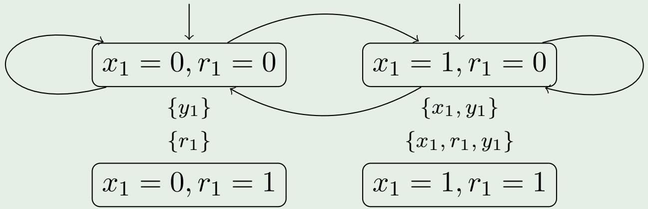
$$y_1 = x_1 \vee \neg r_1$$
$$\delta_{r_1} = x_1 \wedge r_1$$

Annahme (b): initial $r_1 = 0$



from [1, Exercise 2.1, p. 82]

$$TS_1 = (S_1, Act_1, \rightarrow_1, I_1, AP_1, L_1)$$

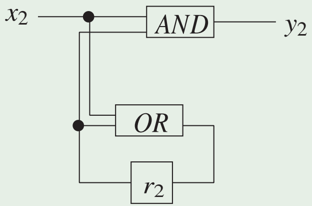


Beispiele TS - Schaltwerk

Beispiel (Exercise 2.1 a - TS von Schaltwerken (2))

$$y_2 = x_2 \wedge r_2$$
$$\delta_{r_2} = x_2 \vee r_2$$

Annahme (b): initial $r_2 = 1$

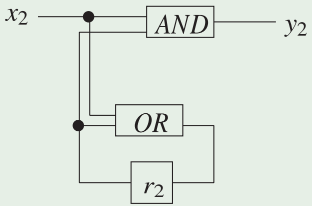


from [1, Exercise 2.1, p. 82]

Beispiele TS - Schaltwerk

Beispiel (Exercise 2.1 a - TS von Schaltwerken (2))

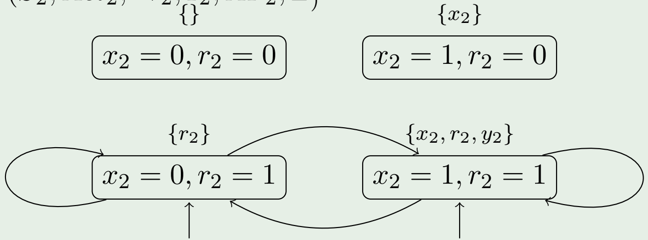
$$y_2 = x_2 \wedge r_2$$
$$\delta_{r_2} = x_2 \vee r_2$$



Annahme (b): initial $r_2 = 1$

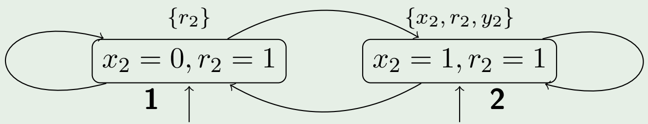
from [1, Exercise 2.1, p. 82]

$TS_2 = (S_2, Act_2, \rightarrow_2, I_2, AP_2, L)$



Paths und Traces - Beispiele

Beispiel (Schaltwerk 2 aus Exercise 2.1)



infinite Path Fragment:

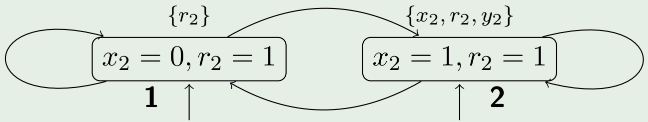
- $\pi_1 = 12111 \cdots = 121^\omega$ initial, maximal, also $\pi_1 \in \text{Paths}(TS)$
- $\pi_2 = 221212 \cdots = 22(12)^\omega$ initial, maximal, $\pi_2 \in \text{Paths}(TS)$

finite Path Fragment:

- $\hat{\pi}_1 = 111$ initial
- $\hat{\pi}_2 = 2211$ initial

Paths und Traces - Beispiele

Beispiel (Schaltwerk 2 aus Exercise 2.1)



infinite Path Fragment:

- $\pi_1 = 12111\cdots = 121^\omega$ initial, maximal, also $\pi_1 \in \mathrm{Paths}(TS)$
  $\mathrm{trace}(\pi_1) = \{r_2\}\{x_2, r_2, y_2\}\{r_2\}^\omega$
- $\pi_2 = 221212\cdots = 22(12)^\omega$ initial, maximal, $\pi_2 \in \mathrm{Paths}(TS)$
  $\mathrm{trace}(\pi_2) = \{x_2, r_2, y_2\}^2(\{r_2\}\{x_2, r_2, y_2\})^\omega$

finite Path Fragment:

- $\hat{\pi}_1 = 111$ initial
- $\hat{\pi}_2 = 2211$ initial

Beispielaufgaben

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

- a false $\quad P = \emptyset$

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

- a) false $\quad P = \emptyset$
- b) am Anfang gilt: $x = 0$
  $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

- a false $\quad P = \emptyset$
- b am Anfang gilt: $x = 0$
  $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
- c am Anfang gilt: $x \neq 0$
  $P = \{A\sigma \mid \{x > 1\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

a. false $\quad P = \emptyset$
b. am Anfang gilt: $x = 0$
   $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
c. am Anfang gilt: $x \neq 0$
   $P = \{A\sigma \mid \{x > 1\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
d. am Anfang ist $x = 0$, aber irgendwann $x > 1$
   $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
   $\cap \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \{x > 1\} \subseteq A_j\}$

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

a. false $\quad P = \emptyset$
b. am Anfang gilt: $x = 0$
   $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
c. am Anfang gilt: $x \neq 0$
   $P = \{A\sigma \mid \{x > 1\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
d. am Anfang ist $x = 0$, aber irgendwann $x > 1$
   $$P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\} \cap \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \{x > 1\} \subseteq A_j\}$$
e. $x > 1$ nur endlich oft
   $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \forall j \geq i.\ \{x > 1\} \not\subseteq A_j\}$

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

- a) false $\quad P = \emptyset$
- b) am Anfang gilt: $x = 0$
  $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
- c) am Anfang gilt: $x \neq 0$
  $P = \{A\sigma \mid \{x > 1\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
- d) am Anfang ist $x = 0$, aber irgendwann $x > 1$
  $$P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\} \cap \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \{x > 1\} \subseteq A_j\}$$
- e) $x > 1$ nur endlich oft
  $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \forall j \geq i.\ \{x > 1\} \not\subseteq A_j\}$
- f) $x > 1$ unendlich oft
  $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \forall i \geq 0.\ \exists j \geq i.\ \{x > 1\} \subseteq A_j\}$

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

a. false $\quad P = \emptyset$
b. am Anfang gilt: $x = 0$
   $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
c. am Anfang gilt: $x \neq 0$
   $P = \{A\sigma \mid \{x > 1\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
d. am Anfang ist $x = 0$, aber irgendwann $x > 1$
   $$P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\} \cap \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \{x > 1\} \subseteq A_j\}$$
e. $x > 1$ nur endlich oft
   $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \forall j \geq i.\ \{x > 1\} \not\subseteq A_j\}$
f. $x > 1$ unendlich oft
   $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \forall i \geq 0.\ \exists j \geq i.\ \{x > 1\} \subseteq A_j\}$
g. –

Exercise 3.5.

TS mit $AP = \{x = 0, x > 1\}$. Formuliere als LT-Property P:

a. false $\quad P = \emptyset$
b. am Anfang gilt: $x = 0$
   $P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
c. am Anfang gilt: $x \neq 0$
   $P = \{A\sigma \mid \{x > 1\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\}$
d. am Anfang ist $x = 0$, aber irgendwann $x > 1$
   $$P = \{A\sigma \mid \{x = 0\} \in A \ \wedge \ \sigma \in (2^{AP})^\omega\} \cap \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \{x > 1\} \subseteq A_j\}$$
e. $x > 1$ nur endlich oft
   $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists i \geq 0.\ \forall j \geq i.\ \{x > 1\} \not\subseteq A_j\}$
f. $x > 1$ unendlich oft
   $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \forall i \geq 0.\ \exists j \geq i.\ \{x > 1\} \subseteq A_j\}$
g. –
h. true $\quad P = (2^{AP})^\omega$

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \text{pref}(\sigma). \quad \left(\hat{\sigma} \notin \text{pref}(P_{safe}) \right) \right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\text{pref}(P_{live}) = (2^{AP})^*$$

Exercise 3.6.

a. $P = \{A_0 A_1 A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \text{pref}(\sigma). \quad \left(\hat{\sigma} \notin \text{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\text{pref}(P_{live}) = (2^{AP})^*$$

Literatur
Transitions-Systeme
Linear-Time Properties
Dekompositions-Theorem

Exercise 3.6.

- a) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante
- b) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \exists!\, i \geq 0.\ A \in A_i\}$ in keiner bekannten Kategorie

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \text{pref}(\sigma). \quad \left(\hat{\sigma} \notin \text{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\text{pref}(P_{live}) = (2^{AP})^*$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Exercise 3.6.

- a) $P = \{A_0 A_1 A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante
- b) $P = \{A_0 A_1 A_2 \dots \in (2^{AP})^\omega \mid \exists!\, i \geq 0.\ A \in A_i\}$
  in keiner bekannten Kategorie
  - keine Liveness P.: $\text{pref}(P) \neq (2^{AP})^*$, da $\hat{\sigma} = \{A\}^2 \notin \text{pref}(P)$

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \text{pref}(\sigma). \quad (\hat{\sigma} \notin \text{pref}(P_{safe})) \right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\text{pref}(P_{live}) = (2^{AP})^*$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Exercise 3.6.

a. $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante
b. $P = \{A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \mid \exists!\, i \geq 0.\ A \in A_i\}$
   in keiner bekannten Kategorie
   - keine Liveness P.: $\text{pref}(P) \neq (2^{AP})^*$, da $\hat{\sigma} = \{A\}^2 \notin \text{pref}(P)$
   - keine Safety P.: Sei z. B. $\sigma = A_0 A_1 A_2 \cdots \in (2^{AP})^\omega \setminus P_{safe}$ ein Trace für den gilt: $\forall i \geq 0.\ A \notin A_i$
     Dann existiert kein Präfix von σ, welches sich zu einem Wort in P verlängern lässt.

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe}) \right) \right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

Exercise 3.6.

a. $P = \{A_0 A_1 A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante
b. $P = \{A_0 A_1 A_2 \dots \in (2^{AP})^\omega \mid \exists!\, i \geq 0.\ A \in A_i\}$
   in keiner bekannten Kategorie
   - keine Liveness P.: $\mathrm{pref}(P) \neq (2^{AP})^*$, da $\hat{\sigma} = \{A\}^2 \notin \mathrm{pref}(P)$
   - keine Safety P.: Sei z. B. $\sigma = A_0 A_1 A_2 \dots \in (2^{AP})^\omega \setminus P_{safe}$ ein Trace für den gilt: $\forall i \geq 0.\ A \notin A_i$
     Dann existiert kein Präfix von σ, welches sich zu einem Wort in P verlängern lässt.
   - keine Invariante.: trivial

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe})\right)\right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

Literatur Transitions-Systeme Linear-Time Properties
Dekompositions-Theorem

Exercise 3.6.

a) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante

b) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \exists!\, i \geq 0.\ A \in A_i\}$
in keiner bekannten Kategorie
- keine Liveness P.: $\mathrm{pref}(P) \neq (2^{AP})^*$, da $\hat{\sigma} = \{A\}^2 \notin \mathrm{pref}(P)$
- keine Safety P.: Sei z. B. $\sigma = A_0A_1A_2 \dots \in (2^{AP})^\omega \setminus P_{safe}$ ein Trace für den gilt: $\forall i \geq 0.\ A \notin A_i$
Dann existiert kein Präfix von σ, welches sich zu einem Wort in P verlängern lässt.
- keine Invariante.: trivial

c) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0.\ \exists j \geq i.\ A \in A_j \quad \wedge$
$\forall i \geq 0.\ \exists j \geq i.\ B \in A_j\}$
Liveness P.: Anhängen von $\left(\{A\}\{B\}\right)^\omega$

Exercise 3.6.

Definition (Safety Property P_{safe}, Bad Prefixes)

LT-Property P_{safe} über AP ist Safety Property, gdw.

$$\forall \sigma \in (2^{AP})^\omega \setminus P_{safe}. \quad \left(\exists \hat{\sigma} \in \mathrm{pref}(\sigma). \quad \left(\hat{\sigma} \notin \mathrm{pref}(P_{safe}) \right) \right)$$

$\hat{\sigma}$ heißt bad prefix für P_{safe}

Definition (Liveness Property P_{live})

LT-Property P_{live} über AP ist liveness Property, gdw.

$$\mathrm{pref}(P_{live}) = (2^{AP})^*$$

Exercise 3.6.

- a) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0 A \notin A_i\}$ Invariante
- b) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \exists!\, i \geq 0.\ A \in A_i\}$
  in keiner bekannten Kategorie
  - keine Liveness P.: $\mathrm{pref}(P) \neq (2^{AP})^*$, da $\hat{\sigma} = \{A\}^2 \notin \mathrm{pref}(P)$
  - keine Safety P.: Sei z. B. $\sigma = A_0A_1A_2 \dots \in (2^{AP})^\omega \setminus P_{safe}$ ein Trace für den gilt: $\forall i \geq 0.\ A \notin A_i$
    Dann existiert kein Präfix von σ, welches sich zu einem Wort in P verlängern lässt.
  - keine Invariante.: trivial
- c) $P = \{A_0A_1A_2 \dots \in (2^{AP})^\omega \mid \forall i \geq 0.\ \exists j \geq i.\ A \in A_j \quad \wedge$
  $\forall i \geq 0.\ \exists j \geq i.\ B \in A_j\}$
  Liveness P.: Anhängen von $\left(\{A\}\{B\}\right)^\omega$
- d) $P = \{A_0A_1A_2 \dots \mid \forall i \geq 0.\ (A \in A_i \Rightarrow (\exists j > i.\ B \in A_j))\}$
  Liveness P.: Anhängen von $\{B\}^\omega$