Foundations of System Development

Martin Wirsing

in cooperation with
Axel Rauschmayer

WS 05/06

Spezifikation von Transitionssystemen mit TLA

Temporale Logiken zur Beschreibung reaktiver Systeme und ihrer Eigenschaften wurden seit der 2. Hälfte der 1970er Jahre entwickelt (erste Veröffentlichungen 1977 von A. Pnueli und von F. Kröger).

Ab 1990 definierte L. Lamport die Temporal Logic of Actions (TLA), die sich durch folgende Charakteristika auszeichnet:

- natürliche Beschreibung von Transitionssystemen durch Formeln
- Charakterisierung von Begriffen wie (Parallel-)Komposition, Verfeinerung, Hiding durch einen möglichst kleinen Satz logischer Operatoren
- weitgehende Reduktion temporallogischer Verifikationsbedingungen auf prädikatenlogische Beweisverpflichtungen

Ziele

- TLA als Beschreibungssprache für reaktive Systeme
- unterschiedliche Spezifikationsstile, z.B. asynchrone vs. synchrone Kommunikation, Interleaving vs. echte Parallelität
- grundlegende Regeln zur Systemverifikation in TLA
- Komposition, Verfeinerung und Hiding
- Implementierungsbeweise durch Verfeinerungsabbildungen

Die Logik TLA

Example (Spezifikation einer Uhr mit Stunden- und Minutenanzeige):

$$
\begin{aligned}
ICl\!k &\equiv hr \in \{0,\ldots,23\} \wedge min \in \{0,\ldots,59\} \\
Min &\equiv min < 59 \wedge min' = min + 1 \wedge hr' = hr \\
Hr &\equiv min = 59 \wedge min' = 0 \wedge hr' = (hr + 1) \bmod 24 \\
Tick &\equiv Min \vee Hr \\
Clock &\equiv ICl\!k \wedge \Box[Tick]_{hr,min} \wedge \mathrm{WF}_{hr,min}(Tick)
\end{aligned}
$$

Üblicherweise haben Systemspezifikationen in TLA die Form

$$Init \wedge \Box[Next]_v \wedge L$$

Dabei ist

[$Init$:] eine prädikatenlogische Formel zur Beschreibung der Anfangszustände.

[$Next$:] eine Formel mit gestrichenen und ungestrichenen Variablen zur Beschreibung der erlaubten Zustandsübergänge. $Next$ hat meist die Form $A_1 \vee \ldots \vee A_n$, wobei die Formeln A_i die einzelnen Systemaktionen beschreiben.

[v :] ein Tupel aller Variablen, die vom System verändert werden können.

[L :] eine Konjunktion von Fairnessbedingungen $\mathrm{WF}_v(A_i)$ oder $\mathrm{SF}_v(A_i)$.

Man unterscheidet in TLA drei Klassen von Formeln:

- Zustandsformeln beschreiben Zustände,
- Aktionsformeln beschreiben Zustandsübergänge,
- temporallogische Formeln beschreiben Abläufe.

Zustandsformeln (state predicates)

Im folgenden sei eine (z.B. algebraisch beschriebene) Signatur Σ vorausgesetzt.

Prädikatenlogische (Σ-)Formeln heißen in TLA Zustandsformeln.

Beispiele: $n > 0$, $q =$ empty, $y = 0 \Leftrightarrow pc_1 =$ "g", $\exists\, k : n = m + k$

Formal wird die Menge X der Variablen unterteilt in disjunkte Teilmengen

- X_f von flexiblen (zustandsabhängigen) Variablen und
- X_r von rigiden (zustandsunabhängigen) Variablen.

Die rigiden Variablen entsprechen den üblichen Variablen der Prädikatenlogik.
Die flexiblen Variablen modellieren Zustandskomponenten (z.B. Programmvariablen).

Die Syntax von Zustandsformeln entspricht der Definition von Σ-Formeln in Abschnitt 2.2 von Teil II der Vorlesung.

Interpretation von Zustandsformeln

Vorausgesetzt sei eine gegebene Σ-Algebra A.

Ein Zustand s ist eine (sortenrichtige) Belegung der flexiblen Variablen mit Werten.

Zustandsformeln werden interpretiert relativ zu einem Zustand s und einer Belegung ξ der rigiden Variablen.

Terme:

$$[\![x]\!]_{s,\xi} = \xi(x) \qquad \text{für rigide Variablen } x \in X_r$$
$$[\![v]\!]_{s,\xi} = s(v) \qquad \text{für flexible Variablen } v \in X_f$$
$$[\![f(t_1,\ldots,t_n)]\!]_{s,\xi} = f^A([\![t_1]\!]_{s,\xi},\ldots,[\![t_n]\!]_{s,\xi})$$

Formeln:

$[\![p(t_1,\ldots,t_n)]\!]_{s,\xi} = \mathrm{T}$	gdw.	$([\![t_1]\!]_{s,\xi},\ldots,[\![t_n]\!]_{s,\xi}) \in p^A$
$[\![\neg\, P]\!]_{s,\xi} = \mathrm{T}$	gdw.	$[\![P]\!]_{s,\xi} = \mathrm{F}$
$[\![P \wedge Q]\!]_{s,\xi} = \mathrm{T}$	gdw.	$[\![P]\!]_{s,\xi} = \mathrm{T}$ und $[\![Q]\!]_{s,\xi} = \mathrm{T}$
$[\![\exists\, x : P]\!]_{s,\xi} = \mathrm{T}$	gdw.	$[\![P]\!]_{s,\eta} = \mathrm{T}$ für ein η mit $\eta(y) = \xi(y)$ für alle $y \in X_r \setminus \{x\}$
$[\![\exists\, v : P]\!]_{s,\xi} = \mathrm{T}$	gdw.	$[\![P]\!]_{t,\xi} = \mathrm{T}$ für ein t mit $t(w) = s(w)$ für alle $w \in X_f \setminus \{v\}$

Für eine gegebene Belegung ξ beschreiben Zustandsformeln also Mengen von Zuständen, z.B. die möglichen Anfangszustände eines Transitionssystems.

Aktionsformeln (actions, transition predicates)

Aktionsformeln sind prädikatenlogische Formeln, die gestrichene und ungestrichene Variablen enthalten dürfen.

Beispiele: $n' = n+1$, $\quad q' = \mathrm{cons}(i', q)$, $\quad \exists\, x : n = x + m'$

Formal sei $X_f' = \{v' \mid v \in X_f\}$ eine Kopie von X_f.

Aktionsformeln sind prädikatenlogische (Σ-)Formeln mit Variablen aus $X_r \cup X_f \cup X_f'$.

Semantik.

Aktionsformeln werden interpretiert relativ zu einem Paar (s,t) von Zuständen und einer Belegung ξ von X_r.

Dabei ist

$$[\![v]\!]_{s,t,\xi} = s(v) \quad \text{für } v \in X_f$$
$$[\![v']\!]_{s,t,\xi} = t(v) \quad \text{für } v' \in X_f'$$
$$[\![x]\!]_{s,t,\xi} = \xi(x) \quad \text{für } x \in X_r$$

Wie bei Operationsschemata in Z werden gestrichene Variablen im Nachfolgerzustand interpretiert.

Für eine gegebene Belegung ξ von X_r beschreiben Aktionsformeln eine Menge von Zustandspaaren, z.B. die erlaubten Übergänge eines Transitionssystems.

Abkürzende Schreibweisen

- Für einen Term t bzw. eine Zustandsformel P bezeichnen t' und P' das Ergebnis der Ersetzung der freien Variablen $v \in X_f$ durch die entsprechenden Variablen $v' \in X'_f$. Dabei werden gebundene Variablen ggf. umbenannt, um Namenskonflikte zu vermeiden.

  Beispiele:
  $$\begin{aligned}(v+1)' &\equiv v'+1\\ (\exists\, x : n = x + m)' &\equiv \exists\, x : n' = x + m'\\ (\exists\, n' : n = n' + m)' &\equiv \exists\, n1 : n' = n1 + m'\end{aligned}$$

- Für eine Aktionsformel A und Terme $t_1, \ldots, t_n$ (ohne gestrichene Variablen):

$$\begin{aligned}[A]_{t_1,\ldots,t_n} &\equiv A \vee (t'_1 = t_1 \wedge \ldots \wedge t'_n = t_n)\\ \langle A\rangle_{t_1,\ldots,t_n} &\equiv A \wedge \neg\,(t'_1 = t_1 \wedge \ldots \wedge t'_n = t_n)\end{aligned}$$

  Ein Paar (s,t) von Zuständen erfüllt die Formel $[A]_{t_1,\ldots,t_n}$, wenn es A erfüllt oder wenn die Werte aller Terme $t_1, \ldots, t_n$ unverändert bleiben.
  Ein Paar (s,t) von Zuständen erfüllt die Formel $\langle A\rangle_{t_1,\ldots,t_n}$, wenn es A erfüllt und wenn sich der Wert mindestens eines Terms $t_1, \ldots, t_n$ ändert.

  Es gilt: $\langle A\rangle_{t_1,\ldots,t_n} \Leftrightarrow \neg\,[\neg\, A]_{t_1,\ldots,t_n}$ und $[A]_{t_1,\ldots,t_n} \Leftrightarrow \neg\,\langle \neg\, A\rangle_{t_1,\ldots,t_n}$

- Für eine Aktionsformel A definieren wir

$$\text{ENABLED}\, A \equiv \exists\, v'_1, \ldots, v'_n : A$$

  wobei $\{v'_1, \ldots, v'_n\}$ die Menge der freien gestrichenen Variablen ($\in X'_f$) ist.

  Die Zustandsformel ENABLED A gilt in einem Zustand s genau dann, wenn es einen Zustand t gibt, so dass A im Zustandspaar (s,t) erfüllt ist.

  Example:

$$\begin{aligned}&\text{ENABLED}(n' = n+1) \equiv \exists\, n' : n' = n+1\\ &\text{ENABLED}(q = \text{cons}(o', q')) \equiv \exists\, o', q' : q = \text{cons}(o', q')\\ &\text{ENABLED}(\exists\, z' : w = v' + z') \equiv \exists\, v', z' : w = v' + z'\\ &(\text{ENABLED}(q = \text{cons}(o', q')))' \equiv \exists\, o1, q1 : q' = \text{cons}(o1, q1)\end{aligned}$$

Temporallogische Formeln (temporal formulas) werden (zunächst) induktiv folgendermaßen definiert:

Definition:

- Jede Zustandsformel P ist eine temporale Formel.
- Ist A Aktionsformel und sind $t_1, \ldots, t_m$ Terme, so ist $\Box[A]_{t_1,\ldots,t_m}$ temporale Formel.
- Ist F temporale Formel, so ist $\Box F$ ("always F") temporale Formel.
- Aussagenlogische Kombinationen temporaler Formeln sind temporale Formeln.

Semantik temporaler Formeln definiert über Folgen $\sigma = s_0 s_1 \ldots$ von Zuständen

$$
\begin{aligned}
[\![P]\!]_{\sigma,\xi} = \mathrm{T} \quad &\text{gdw.} \quad [\![P]\!]_{s_0,\xi} = \mathrm{T} \qquad (P \text{ Zustandsformel}) \\
[\![\Box[A]_t]\!]_{\sigma,\xi} = \mathrm{T} \quad &\text{gdw.} \quad [\![[A]_t]\!]_{s_n,s_{n+1},\xi} = \mathrm{T} \text{ für alle } n \in \mathbb{N} \\
[\![\Box F]\!]_{\sigma,\xi} = \mathrm{T} \quad &\text{gdw.} \quad [\![F]\!]_{\sigma[n..],\xi} = \mathrm{T} \text{ für alle } n \in \mathbb{N} \\
[\![\neg F]\!]_{\sigma,\xi} = \mathrm{T} \quad &\text{gdw.} \quad [\![F]\!]_{\sigma,\xi} = \mathrm{F} \\
[\![F \wedge G]\!]_{\sigma,\xi} = \mathrm{T} \quad &\text{gdw.} \quad [\![F]\!]_{\sigma,\xi} = \mathrm{T} \text{ und } [\![G]\!]_{\sigma,\xi} = \mathrm{T}
\end{aligned}
$$

Dabei bezeichnet $\sigma[n..]$ den Suffix $s_n s_{n+1} \ldots$ von σ.

Abkürzende Schreibweisen für temporale Formeln

- Ist F temporale Formel, so steht $\Diamond F$ ("eventually F", "finally F") für die Formel

  $$\Diamond F \equiv \neg \Box \neg F$$

  Es ist $[\![\Diamond F]\!]_{\sigma,\xi} = \mathrm{T}$ gdw. $[\![F]\!]_{\sigma[n..],\xi} = \mathrm{T}$ für ein $n \in \mathbb{N}$.
- Analog schreiben wir

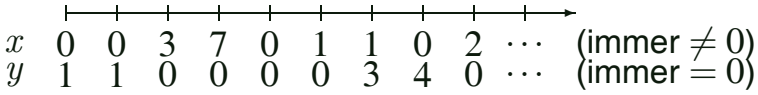
  $$\Diamond\langle A\rangle_{t_1,\ldots,t_m} \equiv \neg \Box[\neg A]_{t_1,\ldots,t_m}$$

  Die Zustandsfolge σ erfüllt $\Diamond\langle A\rangle_t$, wenn mindestens ein Paar aufeinanderfolgender Zustände die Formel A erfüllt und t verändert.
- Für temporale Formeln F, G definieren wir $F \rightsquigarrow G$ ("F leadsto G") durch

  $$F \rightsquigarrow G \equiv \Box(F \Rightarrow \Diamond G)$$

  Die Formel $F \rightsquigarrow G$ ist wahr in σ, wenn für jeden Suffix $\sigma[n..]$, der F erfüllt, ein Suffix $\sigma[m..]$ mit $m \geq n$ existiert, der G erfüllt.
- Klammerersparnis: $\Box$ und $\Diamond$ binden stärker als binäre aussagenlogische Operatoren. Z.B. steht $\Box F \wedge \Diamond G$ für $(\Box F) \wedge (\Diamond G)$.

Example (Semantik temporaler Formeln):



Welche der folgenden Formeln gelten in diesem Ablauf?

$\Box\neg\,(x = 0 \wedge y = 0)$
$\Box[x = 0 \Rightarrow y' = 0]_{x,y}$
$\Diamond(x = 7 \wedge y = 0)$
$\Diamond\langle y = 0 \wedge x' = 0\rangle_y$
$\Box\Diamond(y \neq 0)$
$\Diamond\Box(x = 0 \Rightarrow y \neq 0)$
$\Diamond\Box[\mathbf{false}]_y$

“Unendlich oft” und “irgendwann immer”

Es gilt: $[\![\Box\Diamond F]\!]_{\sigma,\xi} = \mathrm{T}$ gdw. für alle $m \in \mathbb{N}$ gibt es $n \geq m$ mit $[\![F]\!]_{\sigma[n..],\xi} = \mathrm{T}$.
Die Formel $\Box\Diamond F$ fordert also, dass F in der Zustandsfolge σ unendlich oft gilt.

Analog verlangt die Formel $\Box\Diamond\langle A\rangle_t$, dass die Aktion $\langle A\rangle_t$ unendlich oft ausgeführt wird.

Dagegen ist $[\![\Diamond\Box F]\!]_{\sigma,\xi} = \mathrm{T}$ gdw. ein $m \in \mathbb{N}$ existiert, so dass für alle $n \geq m$ gilt: $[\![F]\!]_{\sigma[n..],\xi} = \mathrm{T}$.
Die Formel $\Diamond\Box F$ fordert also, dass F in σ ab einem gewissen Zeitpunkt immer gilt.

Analog verlangt die Formel $\Diamond\Box[A]_t$, dass schließlich nur noch die Aktion $[A]_t$ stattfindet.

Für eine erfüllbare Zustandsformel P sind $\Box\Diamond P$ und $\Diamond\Box P$ Lebendigkeitseigenschaften.

Umformungsregeln: Es gelten

$\neg\,\Box\Diamond F \Leftrightarrow \Diamond\Box\neg\, F$ $\qquad$ $\Diamond\Box\Diamond F \Leftrightarrow \Box\Diamond F$
$\neg\,\Diamond\Box F \Leftrightarrow \Box\Diamond\neg\, F$ $\qquad$ $\Box\Diamond\Box F \Leftrightarrow \Diamond\Box F$

Fairness in TLA

Im Abschnitt über Fainess hatten wir definiert:

- Ein Ablauf ist schwach fair bezüglich A, falls gilt: Ist A ab einem gewissen Zeitpunkt immer ausführbar,
  so wird A unendlich oft ausgeführt.
- Ein Ablauf ist stark fair bezüglich A, falls gilt: Ist A unendlich oft ausführbar, so wird A unendlich oft ausgeführt.

Für Aktionen der Form $\langle A \rangle_t$ können wir dies durch TLA-Formeln ausdrücken:

$$\begin{aligned} \mathrm{WF}_t(A) &\equiv \Diamond\Box\,\mathsf{ENABLED}\langle A\rangle_t \Rightarrow \Box\Diamond\langle A\rangle_t \\ \mathrm{SF}_t(A) &\equiv \Box\Diamond\,\mathsf{ENABLED}\langle A\rangle_t \Rightarrow \Box\Diamond\langle A\rangle_t \end{aligned}$$

Äquivalente Formulierungen sind:

$$\mathrm{WF}_t(A) \equiv \Box\Diamond\neg\,\mathsf{ENABLED}\langle A\rangle_t \vee \Box\Diamond\langle A\rangle_t \qquad \mathrm{SF}_t(A) \equiv \Diamond\Box\neg\,\mathsf{ENABLED}\langle A\rangle_t \vee \Box\Diamond\langle A\rangle_t$$

$$\mathrm{WF}_t(A) \equiv \Box\Diamond(\mathsf{ENABLED}\langle A\rangle_t \Rightarrow \Diamond\langle A\rangle_t) \qquad \mathrm{SF}_t(A) \equiv \Diamond\Box(\mathsf{ENABLED}\langle A\rangle_t \Rightarrow \Diamond\langle A\rangle_t)$$

Proof 1 (für $\mathrm{WF}_t(A)$):

$$\begin{aligned} \mathrm{WF}_t(A) &\Leftrightarrow \neg\,\Diamond\Box\,\mathsf{ENABLED}\langle A\rangle_t \vee \Box\Diamond\langle A\rangle_t \\ &\Leftrightarrow \Box\Diamond\neg\,\mathsf{ENABLED}\langle A\rangle_t \vee \Box\Diamond\langle A\rangle_t \\ &\Leftrightarrow \Box\Diamond\neg\,\mathsf{ENABLED}\langle A\rangle_t \vee \Box\Diamond\Diamond\langle A\rangle_t \\ &\Leftrightarrow \Box\Diamond(\neg\,\mathsf{ENABLED}\langle A\rangle_t \vee \Diamond\langle A\rangle_t) \\ &\Leftrightarrow \Box\Diamond(\mathsf{ENABLED}\langle A\rangle_t \Rightarrow \Diamond\langle A\rangle_t) \end{aligned}$$

Spezifikationsstile in TLA

Ein reaktives System wird in TLA durch eine temporale Formel beschrieben, meist von der Form

$$Init \wedge \Box[Next]_v \wedge L$$

was der Modellierung durch ein Transitionssystem entspricht.

Zustandskomponenten (z.B. Programmvariablen, Kanäle) werden dabei durch flexible Variablen modelliert.

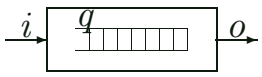
Das Zusammenspiel zwischen Zustandskomponenten (inkl. Synchronisierung bzw. Parallelität von Aktionen) muss explizit durch geeignete Beschreibung der Aktionen codiert werden.

Für unterschiedliche Arten von Systemen existieren jeweils typische Spezifikationsstile.

Die folgenden Beispiele zeigen exemplarisch verschiedene Modellierungen von FIFO-Kanälen.

Example (FIFO mit möglichem Datenverlust):

$$\begin{aligned}
LQInit &\equiv q = \mathsf{empty} \wedge i = o \\
LQEnq &\equiv q' = \mathsf{append}(q, \langle i' \rangle) \wedge o' = o \\
LQDeq &\equiv q \neq \mathsf{empty} \wedge o' = \mathsf{first}(q) \wedge q' = \mathsf{rest}(q) \wedge i' = i \\
LQNext &\equiv LQEnq \vee LQDeq \\
LQLive &\equiv \mathrm{WF}_{q,o}(LQDeq) \\
LQSpec &\equiv LQInit \wedge \Box[LQNext]_{q,o} \wedge LQLive
\end{aligned}$$



- Variablen i und o repräsentieren Sender- und Empfängerseite des Kanals, q modelliert den (hier unbeschränkten) Puffer.
- Aktionen $LQEnq$ und $LQDeq$ modellieren Eingabe bzw. Ausgabe eines Datenwertes aus Sicht des Kanals. Sie schließen sich wechselseitig aus (“Interleaving-Modell”).
- Spezifikation muss nicht auf Änderungen von i reagieren. Daher können gesendete Daten verloren gehen oder auch dupliziert werden.
- Fairnessbedingung für $LQDeq$ garantiert, dass im Puffer gespeicherte Daten wieder ausgegeben werden.

Einfache Interleaving-Spezifikationen haben i.a. die Form

$$Init \wedge \Box[Next]_{v,o} \wedge L$$

Dabei sind:

i, o, v : die Eingabe-, Ausgabe- und internen Variablen der Spezifikation.

$Next$: ist äquivalent zu $Env \vee Sys$, wobei Sys reine Systemaktionen und Env Aktionen der Umgebung einschließlich ihrer Wirkung auf die Systemvariablen beschreibt. Nur o und v kommen im Index von $\Box[Next]_{v,o}$ vor, daher sind beliebige Änderungen an den Eingabevariablen erlaubt, die o und v unverändert lassen ("Umgebungsaktionen").

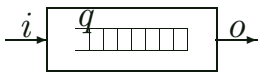
Umgekehrt sollte $Sys \Rightarrow i' = i$ gelten, d.h. die Umgebungsvariablen werden vom System nicht verändert.

L : eine Konjunktion von Fairnessbedingungen der Form $\mathrm{WF}_{v,o}(A)$ oder $\mathrm{SF}_{v,o}(A)$.

In der Regel ist $Next \equiv A_1 \vee \ldots \vee A_n$, und die Fairnessbedingungen betreffen einige der A_i.

Example (Interleaving-Modell, synchrone Kommunikation):

$$\begin{aligned}
SIQInit &\equiv q = \mathsf{empty} \wedge i = o \\
SIQEnq &\equiv i' \neq i \wedge q' = \mathsf{append}(q, \langle i' \rangle) \wedge o' = o \\
SIQDeq &\equiv q \neq \mathsf{empty} \wedge o' = \mathsf{first}(q) \wedge q' = \mathsf{rest}(q) \wedge i' = i \\
SIQNext &\equiv SIQEnq \vee SIQDeq \\
SIQLive &\equiv \mathrm{WF}_{i,q,o}(SIQDeq) \\
SIQSpec &\equiv SIQInit \wedge \Box[SIQNext]_{i,q,o} \wedge SIQLive
\end{aligned}$$



- Da i im Index der Übergangsrelation vorkommt, muss die Spezifikation auf Änderungen von i reagieren.
- Die Definitionen von $SIQEnq$ und $SIQDeq$ stellen sicher, dass der Kanal genau dann eine Eingabeaktion ausführt, wenn sich i ändert. In diesem Sinne modelliert $SIQSpec$ synchrone Kommunikation zwischen Umgebung und Kanal.
- $SIQEnq$ und $SIQDeq$ schließen sich wieder aus (Interleaving-Modellierung).
- Jeder Ablauf, der $SIQSpec$ erfüllt, erfüllt auch $LQSpec$.

Interleaving-Spezifikationen mit synchroner Kommunikation koppeln Umgebungs- und Systemaktionen. Sie sind typischerweise von der Form

$$Init \wedge \Box[Next]_{i,v,o} \wedge L$$

Dabei gilt:

$Next$: ist äquivalent zu $Env \vee Sys$, wobei Sys reine Systemaktionen und Env Aktionen der Umgebung, ggf. einschließlich ihrer Wirkungen auf die Systemvariablen beschreibt. Dabei sollte gelten

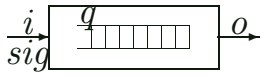
$$Env \Rightarrow o' = o \qquad \text{und} \qquad Sys \Rightarrow i' = i$$

d.h. keine Aktion verändert sowohl die Eingabe- als auch die Ausgabevariablen. Die Eingabevariablen i kommen im Index von $\Box[Next]_{i,v,o}$ vor, um sicherzustellen, dass die Spezifikation auf Änderungen der Eingaben reagiert.

L : beschreibt Fairnessbedingungen an Systemaktionen.

Example (Interleaving-Modell, asynchrone Kommunikation):

$$\begin{aligned}
AQInit &\equiv q = \text{empty} \wedge i = o \wedge sig = 0 \\
AQEnv &\equiv sig = 0 \wedge sig' = 1 \wedge q' = q \wedge o' = o \\
AQEnq &\equiv sig = 1 \wedge sig' = 0 \wedge q' = \text{append}(q, \langle i' \rangle) \wedge o' = o \wedge i' = i \\
AQDeq &\equiv q \neq \text{empty} \wedge o' = \text{first}(q) \wedge q' = \text{rest}(q) \wedge sig' = sig \wedge i' = i \\
AQNext &\equiv AQEnv \vee AQEnq \vee AQDeq \\
AQLive &\equiv \mathrm{WF}_{q,i,o,sig}(AQEnq) \wedge \mathrm{WF}_{q,i,o,sig}(AQDeq) \\
AQSpec &\equiv AQInit \wedge \Box[AQNext]_{q,i,o,sig} \wedge AQLive
\end{aligned}$$



- Zusätzliches Bit sig modelliert “Handshake-Protokoll” zwischen Sender und Kanal.
- “Umgebungsaktion” $AQEnv$ erlaubt Senden eines neuen Werts, falls $sig = 0$ gilt.
  “Systemaktion” $AQEnq$ übernimmt gesendeten Wert und übergibt Kontrolle wieder an Sender.
- Fairnessbedingung an $AQEnq$ garantiert, dass jeder gesendete Wert vom Kanal übernommen wird.
- Jeder Ablauf, der $AQSpec$ erfüllt, erfüllt auch $LQSpec$.

Asynchrone Kommunikation muss explizit modelliert werden.
Dazu werden “Interface-Variablen” wie sig eingeführt, die sowohl von Umgebungs- wie von Systemaktionen verändert werden.

Umgebungsaktionen A (wie das Senden einer Nachricht) zerfallen in zwei Aktionen A_{env} und A_{sys}. Dabei modelliert A_{env} den eigentlichen Umgebungsschritt, es gilt daher

$$A_{env} \Rightarrow v' = v \wedge o' = o$$

Die Aktion A_{sys} modelliert die Reaktion des Systems auf den Umgebungsschritt, es gilt i.a.

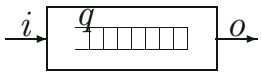
$$A_{sys} \Rightarrow i' = i \wedge o' = o$$

Die Interface-Variablen stellen sicher, dass sich A_{env} und A_{sys} abwechseln.

Fairnessbedingungen an A_{sys} garantieren, dass das System auf jeden Umgebungsschritt reagiert.

Example (Noninterleaving-Modell, synchrone Kommunikation):

$$
\begin{array}{lcl}
SNQInit & \equiv & q = \mathsf{empty} \wedge i = o \\
di & \equiv & \textbf{if } i' = i \textbf{ then } \mathsf{empty} \textbf{ else } \langle i' \rangle \\
do & \equiv & \textbf{if } o' = o \textbf{ then } \mathsf{empty} \textbf{ else } \langle o' \rangle \\
SNQEnq & \equiv & i' \neq i \wedge \mathsf{append}(q, di) = \mathsf{append}(do, q') \\
SNQDeq & \equiv & q \neq \mathsf{empty} \wedge o' = \mathsf{first}(q) \wedge \mathsf{append}(q, di) = \mathsf{append}(do, q') \\
SNQLive & \equiv & \mathrm{WF}_o(SNQDeq) \\
SNQSpec & \equiv & SNQInit \ \wedge \Box[SNQEnq]_i \wedge \Box[SNQEnq \vee SNQDeq]_q \\
 & & \qquad \wedge \Box[SNQDeq]_o \wedge SNQLive
\end{array}
$$



- Spezifikation fordert eine Übergangsrelation pro Variable (bzw. Menge zusammengehörender Variablen).
- Ein- und Ausgabe können im selben Schritt stattfinden (“Noninterleaving-Modell”), falls der Puffer nicht leer ist. Die dadurch bewirkten Änderungen werden über eine “Transitionsinvariante” zusammengefasst.
- Jeder Ablauf, der $SIQSpec$ erfüllt, erfüllt auch $SNQSpec$: gleichzeitige Ein- und Ausgabe wird nicht gefordert, aber zugelassen.

Noninterleaving-Spezifikationen erlauben gleichzeitige Schritte von System und Umgebung.
Sie können in der Form

$$Init \wedge \Box[Env]_i \wedge \Box[Mod]_v \wedge \Box[Out]_o \wedge L$$

geschrieben werden.

Env, Mod, Out : modellieren Eingabe-, interne und Ausgabeaktionen. Die Indizes i, v und o garantieren, dass alle Änderungen an den betreffenden Systemkomponenten diesen Aktionen entsprechen.

Diese Aktionen können miteinander über gemeinsame Variablen synchronisiert werden.

Bedingungen wie $Env \Rightarrow o' = o$ werden nicht verlangt, daher sind gleichzeitige Aktionen von System und Umgebung erlaubt.

L : enthält Fairnessbedingungen an Teilaktionen von Mod oder Out.

Zusammenfassung

- TLA erlaubt Beschreibung von Systemen auf unterschiedliche Weise. Modellierungskonzepte (z.B. Kommunikation über gemeinsame Variablen oder durch Nachrichtenaustausch) werden nicht fest vorgegeben, sondern müssen explizit codiert werden.
- Für übliche Klassen von Systemen existieren Spezifikationsstile, die Anhaltspunkte zum Schreiben eigener Spezifikationen bieten.
- Interleaving-Spezifikationen sind in der Regel einfacher zu schreiben. Noninterleaving-Spezifikationen spiegeln manchmal besser die Realität wider und haben bessere Kompositionalität (vgl. später).

Verifikationsregeln

In TLA können reaktive Systeme nicht nur spezifiziert, sondern auch deren Eigenschaften formuliert und bewiesen werden.

Ein durch die Formel $Spec$ beschriebenes (faires) Transitionssystem erfüllt die durch $Prop$ beschriebene Eigenschaft genau dann, wenn

$$[\![Spec \Rightarrow Prop]\!]_{\sigma,\xi} = \mathrm{T}$$

gilt für alle Zustandsfolgen σ und alle Belegungen ξ (über der zu Grunde liegenden Klasse von Algebren).

Im folgenden Kapitel werden Beweisregeln zum Nachweis von in TLA formulierten Sicherheits- und Lebendigkeitseigenschaften vorgestellt. Diese ermöglichen Beweise von Aussagen der Form

$$Spec \Rightarrow Prop$$

für Systemspezifikationen $Spec$ und typische Klassen von Eigenschaften $Prop$.

Beweise von Invarianten

Invarianten sind Formeln der Gestalt $\Box P$, wobei P eine Zustandsformel ist.

Ein Transitionssystem erfüllt eine Invariante $\Box P$ genau dann, wenn alle erreichbaren Zustände P erfüllen.

Invarianten sind die Basis für alle weiteren Verifikationsschritte und drücken die intuitive Korrektheitsidee des Algorithmus formal aus.

Die grundlegende Regel zum Beweis von Invarianten für TLA-Spezifikationen ist

$$\text{(INV1)} \quad \frac{P \wedge Next \Rightarrow P' \qquad P \wedge v' = v \Rightarrow P'}{P \wedge \Box[Next]_v \Rightarrow \Box P}$$

Eine nützliche Verallgemeinerung ist

$$\text{(INV1)}_m \quad \frac{P \wedge [N_1]_{v_1} \Rightarrow P' \qquad \cdots \qquad P \wedge [N_m]_{v_m} \Rightarrow P'}{P \wedge \Box[N_1]_{v_1} \wedge \ldots \wedge \Box[N_m]_{v_m} \Rightarrow \Box P}$$

Korrektheit von INV1: Seien $P \wedge Next \Rightarrow P'$ sowie $P \wedge v' = v \Rightarrow P'$ gültig.
D.h. für alle Zustände (der gegebenen Klasse von Algebren) s, t und alle Belegungen ξ gilt

$$[\![P \wedge Next \Rightarrow P']\!]_{s,t,\xi} = \mathrm{T} \qquad \text{und} \qquad [\![P \wedge v' = v \Rightarrow P']\!]_{s,t,\xi} = \mathrm{T}$$

Sei $\sigma = s_0 s_1 \ldots$ eine Zustandsfolge, ξ Belegung von X_r, zu zeigen ist

$$[\![P \wedge \Box[Next]_v \Rightarrow \Box P]\!]_{\sigma,\xi} = \mathrm{T}$$

Sei also $[\![P \wedge \Box[Next]_v]\!]_{\sigma,\xi} = \mathrm{T}$,
dann ist zu zeigen: $[\![P]\!]_{s_n,\xi} = \mathrm{T}$ für alle $n \in \mathbb{N}$.

$n = 0$: Nach Annahme folgt $[\![P]\!]_{s_0,\xi} = \mathrm{T}$.

$n \to n+1$: Nach Induktionsvoraussetzung gilt $[\![P]\!]_{s_n,\xi} = \mathrm{T}$. Ferner gilt nach Annahme $[\![[Next]_v]\!]_{s_n,s_{n+1},\xi} = \mathrm{T}$. Gilt $[\![Next]\!]_{s_n,s_{n+1},\xi} = \mathrm{T}$, so folgt $[\![P]\!]_{s_{n+1},\xi} = \mathrm{T}$ aus der Annahme $[\![P \wedge Next \Rightarrow P']\!]_{s_n,s_{n+1},\xi} = \mathrm{T}$. Ansonsten gilt $[\![v' = v]\!]_{s_n,s_{n+1},\xi} = \mathrm{T}$, und die Behauptung folgt aus $[\![P \wedge v' = v \Rightarrow P']\!]_{s_n,s_{n+1},\xi} = \mathrm{T}$.

Example:

Im Puffer der synchronen Queue sind aufeinanderfolgende Elemente verschieden.

Sei $Diff \equiv \wedge \forall k : 1 \leq k < \mathrm{length}(q) \Rightarrow q[k] \neq q[k+1]$
$\wedge\ \mathrm{length}(q) > 0 \Rightarrow q[\mathrm{length}(q)] = i$

(Dabei bezeichnet $\mathrm{length}(q)$ die Länge der Folge q und $q[k]$ das Element an der k-ten Position.)

Dann gelten

$$Diff \wedge SIQEnq \Rightarrow Diff'$$
$$Diff \wedge SIQDeq \Rightarrow Diff'$$
$$Diff \wedge i' = i \wedge q' = q \wedge o' = o \Rightarrow Diff'$$

Also folgt mit (INV1) auch

$$Diff \wedge \Box[SIQNext]_{i,q,o} \Rightarrow \Box Diff$$

Wegen $SIQInit \Rightarrow Diff$
folgt daher $SIQSpec \Rightarrow \Box Diff$.

Die Regel (INV1) erlaubt den Beweis induktiver Invarianten.
In der Regel muss eine zu beweisende Invariante P zuvor geeignet verstärkt werden.

$$\text{(INV)} \quad \frac{Init \Rightarrow Q \qquad Q \wedge [Next]_v \Rightarrow Q' \qquad Q \Rightarrow P}{Init \wedge \Box[Next]_v \Rightarrow \Box P}$$

Das Finden induktiver Invarianten erfordert häufig Kreativität.
Der eigentliche Beweis erfolgt dagegen schematisch und erfordert kein temporallogisches Schließen.

Werden Systeme durch schrittweise Verfeinerung entwickelt, so wird die Invariante (für den jeweiligen Entwicklungsschritt) mit dokumentiert, vgl. Schema-Invarianten in Z.
Das nachträgliche Auffinden der Invariante wird so vermieden.

Exkurs: schwächste Vorbedingung
Zu einer Aktionsformel A und einer Zustandsformel P bezeichne $\mathrm{wp}(P, A)$ die Zustandsformel

$$\mathrm{wp}(P, A) \equiv \forall\, v_1', \ldots, v_n' : A \Rightarrow P'$$

wobei $v_1', \ldots, v_n'$ alle in A und P' frei vorkommenden gestrichenen Variablen seien.
$\mathrm{wp}(P, A)$ heißt schwächste Vorbedingung (weakest precondition) von P bezüglich A.
Es ist $[\![\mathrm{wp}(P, A)]\!]_{s,\xi} = \mathrm{T}$ genau dann, wenn für alle Zustände t mit $[\![A]\!]_{s,t,\xi} = \mathrm{T}$ folgt, dass $[\![P]\!]_{t,\xi} = \mathrm{T}$ ist. $\mathrm{wp}(P, A)$ charakterisiert also die Menge der Zustände, deren sämtliche A-Nachfolger die Formel P erfüllen.
Beispiele:

$$\begin{aligned}
\mathrm{wp}(x = 5, x' = x + 1) &\equiv \forall\, x' : x' = x + 1 \Rightarrow x' = 5 \\
&\Leftrightarrow x = 4 \\
\mathrm{wp}(y \in S, S' = S \cup T \wedge y' = y) &\equiv \forall\, y', S' : S' = S \cup T \wedge y' = y \Rightarrow y' \in S' \\
&\Leftrightarrow y \in S \vee y \in T \\
\mathrm{wp}(x > 0, x' = 0) &\equiv \forall\, x' : x' = 0 \Rightarrow x' > 0 \\
&\Leftrightarrow \mathbf{false}
\end{aligned}$$

Mit dieser Schreibweise kann (INV) umformuliert werden:

$$\frac{Init \Rightarrow Q \qquad Q \Rightarrow \mathrm{wp}(Q, [Next]_v) \qquad Q \Rightarrow P}{Init \wedge \Box[Next]_v \Rightarrow \Box P}$$

Eine nützliche Heuristik zum Finden induktiver Invarianten ist folgende:

1. Start mit der zu beweisenden Invariante: $Q_0 \equiv P$.
2. Versuche, Formeln $Q_i \wedge A \Rightarrow Q_i'$ für jede Teilaktion A von $[Next]_v$ zu beweisen.
   Schlägt der Beweis fehl, setze $Q_{i+1} \equiv Q_i \wedge \mathrm{wp}(Q_i, A)$.
3. Wiederhole Schritt 2, bis
   - entweder alle Teilbeweise gelingen und außerdem $Init \Rightarrow Q_i$ gilt; dann ist Q_i eine induktive Invariante, die P impliziert, oder
   - Q_i nicht aus $Init$ folgt; dann ist P keine Invariante des Systems.

Lebendigkeit 1: Fairness

Fairnessbedingungen garantieren, dass Aktionen irgendwann ausgeführt werden.
Sie sind die Basis für Lebendigkeitsbeweise.

Beispiel: Die schwache Fairnessbedingung an die Aktion $SIQDeq$ garantiert, dass das erste Element eines nichtleeren FIFO-Puffers irgendwann ausgegeben wird. Es gilt also

$$SIQSpec \Rightarrow ((q \neq \text{empty} \wedge \text{first}(q) = x) \leadsto o = x)$$

Dabei ist x eine rigide Variable. Die Aussage gilt für beliebige Belegungen von x (implizite Allquantifizierung).
Diese Überlegung (für Zustandsformeln P, Q) wird formalisiert durch folgende Regel:

$$\text{(WF1)} \quad \frac{\begin{array}{c} P \wedge [Next]_v \Rightarrow P' \vee Q' \\ P \wedge \langle Next \wedge A \rangle_v \Rightarrow Q' \\ P \Rightarrow \mathsf{ENABLED}\langle A \rangle_v \end{array}}{\Box[Next]_v \wedge \mathrm{WF}_v(A) \Rightarrow (P \leadsto Q)}$$

Die Prämissen von (WF1) sind wieder nicht-temporale Formeln.

Korrektheit von (WF1)

Seien die Prämissen von (WF1) gültig, sei $\sigma = s_0 s_1 \ldots$ eine Zustandsfolge und gelte

$$[\![\Box[Next]_v \wedge \mathrm{WF}_v(A)]\!]_{\sigma,\xi} = \mathrm{T}$$

Zu zeigen ist

$$[\![P \rightsquigarrow Q]\!]_{\sigma,\xi} = \mathrm{T}$$

Sei also $n \in \mathbb{N}$ beliebig und gelte $[\![P]\!]_{s_n,\xi} = \mathrm{T}$. Angenommen, $[\![Q]\!]_{s_m,\xi} = \mathrm{F}$ für alle $m \geq n$.

1. Für alle $m \geq n$ ist $[\![P]\!]_{s_m,\xi} = \mathrm{T}$.
   Beweis durch Induktion nach $m \geq n$:
   - Für $m = n$ gilt die Aussage nach Voraussetzung.
   - Gelte $[\![P]\!]_{s_m,\xi} = \mathrm{T}$. Nach Voraussetzung ist $[\![[Next]_v]\!]_{s_m,s_{m+1},\xi} = \mathrm{T}$, also folgt mit Annahme

     $$P \wedge [Next]_v \Rightarrow P' \vee Q'$$

     dass $[\![P]\!]_{s_{m+1},\xi} = \mathrm{T}$ oder $[\![Q]\!]_{s_{m+1},\xi} = \mathrm{T}$ gilt. Mit der Widerspruchsannahme folgt $[\![P]\!]_{s_{m+1},\xi} = \mathrm{T}$.

2. Für alle $m \geq n$ ist $[\![\mathsf{ENABLED}\langle A\rangle_v]\!]_{s_m,\xi} = \mathrm{T}$.
   Dies folgt aus Aussage (1) und Prämisse $P \Rightarrow \mathsf{ENABLED}\langle A\rangle_v$.

3. Für ein $m \geq n$ ist $[\![\langle A\rangle_v]\!]_{s_m,s_{m+1},\xi} = \mathrm{T}$.
   Gilt wegen Aussage (2) und Fairnessannahme $[\![\mathrm{WF}_v(A)]\!]_{\sigma,\xi} = \mathrm{T}$.

4. Für jedes $m \geq n$ mit $[\![\langle A\rangle_v]\!]_{s_m,s_{m+1},\xi} = \mathrm{T}$ folgt $[\![Q]\!]_{s_{m+1},\xi} = \mathrm{T}$.
   Denn es gilt $[\![P \wedge [Next]_v \wedge \langle A\rangle_v]\!]_{s_m,s_{m+1},\xi} = \mathrm{T}$, und mit der Prämisse

   $$P \wedge \langle Next \wedge A\rangle_v \Rightarrow Q'$$

   folgt die Behauptung.

5. Widerspruch, also folgt $[\![Q]\!]_{s_m,\xi} = \mathrm{T}$ für ein $m \geq n$.

Example (zur Anwendung von (WF1)):

Für die Spezifikation $SIQSpec$ aus Beispiel 4 zeigen wir

$$SIQSpec \Rightarrow (\underbrace{(q \neq \text{empty} \wedge \text{first}(q) = x)}_{P} \rightsquigarrow \underbrace{o = x}_{Q})$$

(1a) $P \wedge SIQEnq \Rightarrow P'$
(1b) $P \wedge SIQDeq \Rightarrow Q'$
(1c) $P \wedge i' = i \wedge q' = q \wedge o' = o \Rightarrow P'$
(2) $P \wedge \langle SIQNext \wedge SIQDeq \rangle_{i,q,o} \Rightarrow Q'$
(3) $P \Rightarrow \text{ENABLED} \langle SIQDeq \rangle_{i,q,o}$

Mit (WF1) folgt

$$\Box[SIQNext]_{i,q,o} \wedge \mathrm{WF}_{i,q,o}(SIQDeq) \Rightarrow (P \rightsquigarrow Q)$$

und damit die Aussage.

Temporallogische Gesetze

Logische Gesetze dienen zur Umformulierung und Vereinfachung von Beweisverpflichtungen.
Sie bereiten die Anwendung von Verifikationsregeln wie (INV1), (WF1) und (WFO) vor.

Temporallogische Basisregeln machen Aussagen über $\Box$ und $\Diamond$ ("simple temporal logic").

(STL1) alle Instanzen gültiger Zustandsformeln

(STL2) $\Box F \Rightarrow F$

(STL3) $\Box F \Rightarrow \Box\Box F$

(STL4) $\Box(F \Rightarrow G) \Rightarrow (\Box F \Rightarrow \Box G)$

(STL5) $\Box(F \wedge G) \Leftrightarrow \Box F \wedge \Box G$

(STL6) $\Diamond\Box F \wedge \Diamond\Box G \Leftrightarrow \Diamond\Box(F \wedge G)$

(GEN) $\dfrac{F}{\Box F}$

(MP) $\dfrac{F \qquad F \Rightarrow G}{G}$

Bemerkungen:

- (STL1) erlaubt die Verwendung gültiger “Daten”-Formeln und von Tautologien wie $\Box F \wedge \Box G \Rightarrow \Box F$.
- Aus (STL1) und (MP) folgt, dass beliebige aussagenlogische Schlüsse in TLA-Herleitungen zulässig sind.
- (GEN) muss als Regel formuliert werden: die Formel $F \Rightarrow \Box F$ ist nicht gültig.

Aus diesen Basisregeln sind viele weitere Gesetze beweisbar.

Beispiel: Herleitung von $\Box F \wedge \Diamond G \Rightarrow \Diamond(F \wedge G)$

(1)	$F \wedge \neg (F \wedge G) \Rightarrow \neg G$	(STL1)
(2)	$\Box(F \wedge \neg (F \wedge G) \Rightarrow \neg G)$	(GEN)(1)
(3)	$\Box(F \wedge \neg (F \wedge G) \Rightarrow \neg G) \Rightarrow (\Box(F \wedge \neg (F \wedge G)) \Rightarrow \Box\neg G)$	(STL4)
(4)	$\Box(F \wedge \neg (F \wedge G)) \Rightarrow \Box\neg G$	(MP)(2)(3)
(5)	$\Box F \wedge \Box\neg (F \wedge G) \Rightarrow \Box(F \wedge \neg (F \wedge G))$	(STL5)
(6)	$\Box F \wedge \Box\neg (F \wedge G) \Rightarrow \Box\neg G$	(prop)(4)(5)
(7)	$\Box F \wedge \Diamond G \Rightarrow \Diamond(F \wedge G)$	(prop)(6)

Abgeleitete temporallogische Gesetze (Auswahl)

(T1) $\neg \Box F \Leftrightarrow \Diamond \neg F$

(T2) $\neg \Diamond F \Leftrightarrow \Box \neg F$

(T3) $F \Rightarrow \Diamond F$

(T4) $\Diamond\Diamond F \Rightarrow \Diamond F$

(T5) $$\frac{F \Rightarrow G}{\Box F \Rightarrow \Box G}$$

(T6) $$\frac{F \Rightarrow G}{\Diamond F \Rightarrow \Diamond G}$$

(T7) $\Diamond(F \vee G) \Leftrightarrow \Diamond F \vee \Diamond G$

(T8) $\Box\Diamond F \vee \Box\Diamond G \Leftrightarrow \Box\Diamond(F \vee G)$

(T9) $\Box F \wedge \Diamond G \Rightarrow \Diamond(\Box F \wedge G)$

(T10) $\Box F \wedge \Diamond G \Rightarrow \Diamond(F \wedge G)$

(T11) $\Diamond\Box\Diamond F \Leftrightarrow \Box\Diamond F$

(T12) $\Box\Diamond\Box F \Leftrightarrow \Diamond\Box F$

(T13) $\Diamond\Box F \Rightarrow \Box\Diamond F$

(T14) $\Box\Diamond F \wedge \Diamond\Box G \Rightarrow \Box\Diamond(F \wedge G)$

(T15) $\Box \mathrm{WF}_v(A) \Leftrightarrow \mathrm{WF}_v(A)$

(T16) $\Box \mathrm{SF}_v(A) \Leftrightarrow \mathrm{SF}_v(A)$

(T17) $$\frac{F \Rightarrow G}{F \rightsquigarrow G}$$

(T18) $$\frac{F \rightsquigarrow G \qquad G \rightsquigarrow H}{F \rightsquigarrow H}$$

(T19) $((F \vee G) \rightsquigarrow H) \Leftrightarrow (F \rightsquigarrow H) \wedge (G \rightsquigarrow H)$

(T20) $((F \wedge \Box\neg G) \rightsquigarrow G) \Leftrightarrow (F \rightsquigarrow G)$

(WF1-Inv) $$\frac{\begin{array}{c} I \wedge P \wedge [Next]_v \Rightarrow P' \vee Q' \\ I \wedge P \wedge \langle Next \wedge A\rangle_v \Rightarrow Q' \\ I \wedge P \Rightarrow \mathsf{ENABLED}\langle A\rangle_v \end{array}}{\Box I \wedge \Box[Next]_v \wedge \mathrm{WF}_v(A) \Rightarrow (P \rightsquigarrow Q)}$$

Bemerkung: In der Praxis müssen temporallogische Gesetze nicht aus den Regeln hergeleitet werden, da die Gültigkeit von Formeln der temporalen Aussagenlogik entscheidbar ist.

Implementierung: `ptl`

(http://www.ics.ele.tue.nl/es/research/fv/research/research_ptl.html)

Beispiel:

```
> ptl
PTL V4.2 Copyright (C) 1994-97 G. Janssen, Eindhoven University.

([](F & []!G -> <>G)) <-> [](F -> <>G);
True

[](F -> <>G) & [](F -> <>H) -> [](F -> <>(G & H));
False

<>[]F & []<>G -> []<>(F & G);
True
```

Warnung: `ptl` basiert nicht auf TLA-Syntax und kennt keine Formeln wie $\Box[A]_v$.

TLA-spezifische Gesetze kombinieren Aktionen und temporale Formeln

(TLA1) $\Box P \Leftrightarrow P \wedge \Box[P \Rightarrow P']_{vars(P)}$

(TLA2) $\dfrac{P \wedge [A]_v \Rightarrow [B]_w}{\Box P \wedge \Box[A]_v \Rightarrow \Box[B]_w}$

(INV2) $\Box P \wedge \Box[A]_v \Rightarrow \Box[A \wedge P \wedge P']_v$

(TLA3) $\Box[A]_v \wedge \Box[B]_w \Leftrightarrow \Box\big[[A]_v \wedge [B]_w\big]_{v,w}$

(P Zustandsformel, A, B Aktionsformeln, v, w Tupel von Termen, $vars(P)$ freie Variablen in P)

Diese Gesetze ermöglichen “Simulationsbeweise”, z.B. gilt $SIQSpec \Rightarrow LQSpec$.

Beweis:

(1)	$SIQInit \Rightarrow LQInit$	(prop)
(2)	$SIQEnq \Rightarrow LQEnq$	(prop)
(3)	$SIQDeq \Rightarrow LQDeq$	(prop)
(4)	$[SIQNext]_{i,q,o} \Rightarrow [LQNext]_{q,o}$	(prop)(2)(3)
(5)	$\Box[SIQNext]_{i,q,o} \Rightarrow \Box[LQNext]_{q,o}$	(TLA2)(4)
(6)	$\langle SIQDeq\rangle_{i,q,o} \Leftrightarrow \langle LQDeq\rangle_{q,o}$	(data)
(7)	$\mathrm{WF}_{i,q,o}(SIQDeq) \Leftrightarrow \mathrm{WF}_{q,o}(LQDeq)$	(6)(TLA2)(STL)
(8)	$SIQSpec \Rightarrow LQSpec$	(prop)(1)(5)(7)

Summary

- Die Logik TLA erweitert klassische Prädikatenlogik um gestrichene Variablen und temporale Operatoren.
  Man unterscheidet Zustands-, Aktions- und temporallogische Formeln, die über Zuständen, Zustandspaaren und unendlichen Zustandsfolgen interpretiert werden.
- Temporale Formeln enthalten Aktionsformeln nur in Teilformeln $\Box[A]_v$ bzw. $\Diamond\langle A\rangle_v$.
- Spezifikationen und Eigenschaften reaktiver Systeme werden durch TLA-Formeln ausgedrückt. Systemspezifikationen sind i.a. von der Form

  $$Init \wedge \Box[Next]_v \wedge L$$

  wobei L eine Konjunktion von Fairnessbedingungen ist. Flexible Variablen repräsentieren Zustandskomponenten des Systems.
- Ein System $Spec$ erfüllt eine Eigenschaft $Prop$, wenn die Formel

  $$Spec \Rightarrow Prop$$

  gültig ist.

- Zum Beweis von Eigenschaften dienen logische Beweisregeln. Typische Verifikationsregeln reduzieren temporallogische Aussagen auf nicht-temporale Beweisverpflichtungen. Temporallogische Gesetze dienen zur Vereinfachung und Umformung von Aussagen.