

Ulm, 27.05.2003

Approximationsbasierte Verifikationsmethoden

für Echtzeitsysteme

Promotionsvorhaben

Maria Sorea

SRI International

Computer Science Laboratory, USA

sorea@csl.sri.com

<http://www.csl.sri.com/users/sorea>

Betreuer: Prof. Dr. F.W. von Henke

27.05.2003

Reaktive Systeme und Echtzeitsysteme

Reaktive Systeme

- Ständige Interaktion mit der Umgebung.
- Fortlaufende Kommunikation, permanenter Datenaustausch.
- Reagieren auf Ereignisse aus der Umgebung und erzeugen Effekte in der Umgebung.

Echtzeitsysteme: Zeitinformation spielt eine wichtige Rolle für Korrektheit. Qualitative (logische) und quantitative (zeitliche) Abfolge der Ereignisse ist wichtig.

Z.B.: Airbag, der nach einer Kollision expandiert: nicht nur die Expansion ist lebenswichtig, sondern auch der Zeitpunkt.

Echtzeitsysteme – Anwendungen

- Kontrolle von Produktionsprozessen
- Kontrollsysteme im Flug- und Bahnbereich
- Telekommunikation
- Planung und Scheduling
- Asynchrone digitale Schaltkreise mit Verzögerungselementen
- Multimedia Anwendungen
- Biologische Systeme

Merkmale von Echtzeitsystemen

Komplex, viele separate Komponenten, Kommunikation

Eigenschaften

- **Sicherheit (safety):** Gewisse “schlechte” Situationen treten nie auf.
- **Lebendigkeit (liveness):** “Gute” Situationen kommen irgendwann vor.
- **Echtzeit Eigenschaften:**
 - Bounded response: eine Anforderung wird innerhalb einer gewissen Zeit gewährleistet.
 - Timeout: ein Ereignis muß innerhalb einer gegebenen Zeitspanne vorkommen.

Fragestellung

Erfüllt ein Echtzeitsystem eine gegebene Eigenschaft?

Modellüberprüfungsproblem (MÜP)

Modellüberprüfung (Model Checking)

$$M \stackrel{?}{\models} \varphi$$

M : Systembeschreibung

φ : Temporallogische Eigenschaft

- Automatische Verifikationsmethode
- Exploration des Zustandsraumes
- Problem: PSPACE-vollständig für Zeitsysteme

Gegenstand dieser Arbeit

*Entwicklung von Algorithmen zur Verifikation von Echtzeitsystemen,
die den Speicher- und Zeitverbrauch oftmals drastisch reduzieren.*

Deshalb: Verifikation größerer, realistischer Zeitsysteme.

Inhalt

1. Teil I

(a) Modellierung von Echtzeitsystemen

i. Zeitautomaten

ii. Regionengraph als Verifikationsgrundlage für Zeitautomaten

iii. Verifikationswerkzeuge - Tempo

(b) Spezifikation der Eigenschaften von Echtzeitsystemen

i. Zeitlogiken

ii. Event Recording Logik mit Fixpunkten (ERL)

2. Teil II – Abstraktionsbasierte Verifikationsmethoden

(a) Grundidee

(b) Verzögerte Abstraktion

(c) Verzögertes Theorembeweisen

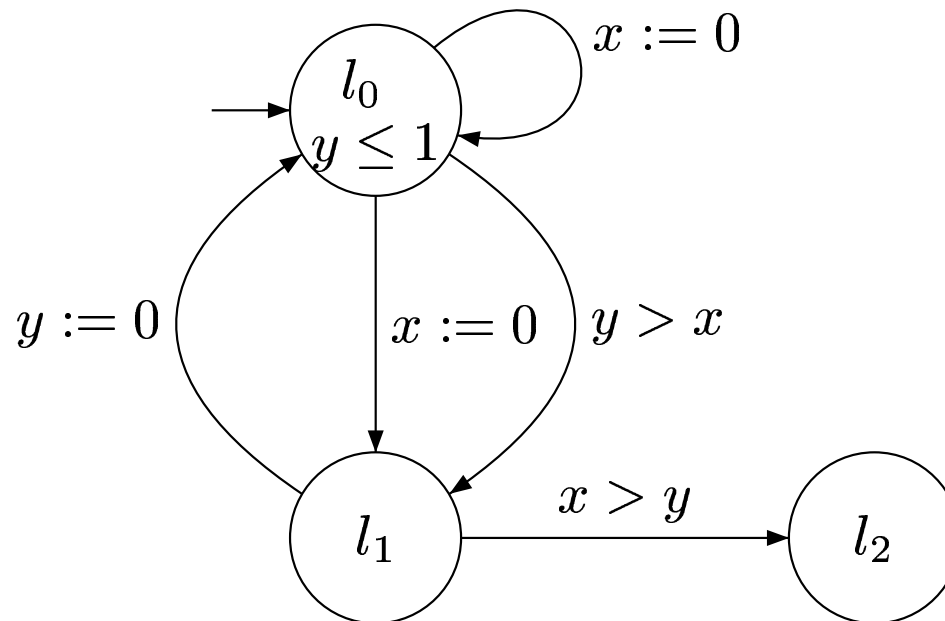
3. Teil III

(a) Fallbeispiele

(b) Zusammenfassung

Zeitautomaten – [Alur, Dill '94]

Zeitautomaten: $S = \langle L, l_0, Cl, E, Inv \rangle$



Uhreninterpretation: $\nu : Cl \rightarrow \mathbb{R}_0^+$

Zeitautomaten – Semantik

Zustandsübergangssystem (Kripkestruktur)

$$\mathcal{M} = \langle L \times \mathcal{V}_{Cl}, (l_0, v_0), \Sigma \times \mathbb{R}_0^+, \Rightarrow \rangle$$

Verhalten des Systems durch Pfade in der Kripkestruktur beschrieben.

- (l, v) Zustand von \mathcal{M} : $l \in L, v \in \mathcal{V}_{Cl} \rightsquigarrow$ unendliches Zustandsraum
- Zustandsübergangsrelation

- Zustandswechsel $(l, v) \longrightarrow (l', v[\lambda := 0])$

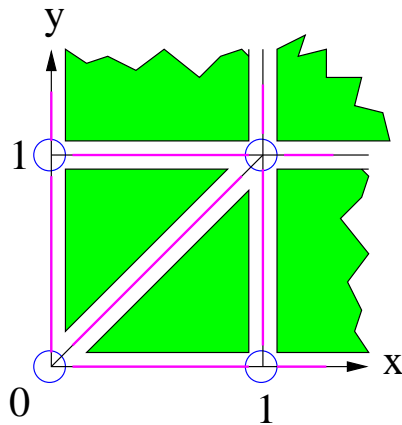
- Zeitverzögerungsschritt $(l, v) \xrightarrow{\delta} (l, v + \delta)$

- Z.B.

$$(l_0, x = 0, y = 0) \xrightarrow{0.5} (l_0, x = 0.5, y = 0.5) \longrightarrow (l_1, x = 0, y = 0.5)$$

Verifikation von Zeitautomaten

- Regionengraph [Alur, Courcoubetis, Dill '90]
- Gegeben: \mathcal{S}, Cl, c
- Endliche Partitionierung des unendlichen Zustandsraums
- Region: Menge $\mathcal{X} \subseteq \mathcal{V}_{Cl}$ von Uhrenvaluationen, s.d. für alle Zeitconstraints g und zwei Valuationen $v_1, v_2 \in \mathcal{X}$ es gilt $v_1 \models g$ iff $v_2 \models g$.
- $v_1 \equiv_{\mathcal{S}} v_2$.
- Beispiel: 2 Uhren x, y , größte Konstante $c = 1$



4 Eckpunkte (z.B. [x = 0 AND y = 0])

9 offene Segmente (z.B. [0 < x = y < 1])

5 offene Regionen (z.B. [0 < x < y < 1])

Symbolische Verifikationstechniken

- Anzahl der Regionen ist exponentiell in der Anzahl der Uhren und der größten Konstante.
- Beschreibung der Regionen als Boolesche Kombination von linearen (Un)gleichheiten über Uhren [Henzinger, Nicollin, Sifakis, Yovine '94].
 \rightsquigarrow Zonen
- Geeignete Datenstrukturen zur Darstellung der Regionen: DBM, CDD, DDD
- Verifikationswerkzeuge
 - Kronos [Daws, Olivero, Tripakis, Yovine '96]
 - Uppaal [Larsen, Pettersson, Yi '97]
 - Tempo [Sorea '01]

Echtzeitlogiken

- Erweiterung Temporallogiken (z.B. LTL, CTL, μ -Kalkül) mit diskreter vs. kontinuierlicher Zeit.
- Zeitgebundene Operatoren vs. explizite Uhren:
 - $\forall \square (p \rightarrow \forall \diamond_{\leq 5} q)$ “time-bounded response”
(TCTL, [Alur,Courcoubetis,Dill '91])
 - $\forall \square x. (p \rightarrow \forall \diamond (q \wedge x \leq 5))$
(TCTL, [Henzinger,Nicollin,Sifakis,Yovine '94])

Echtzeitlogiken (Forts.)

- **TCTL** [Alur,Courcoubetis,Dill '91]
- **RTCTL** [Emerson,Mok,Sistla,Srinivasan '91]
- **$T\mu$** [Henzinger,Nicollin,Sifakis,Yovine '94]
- **\mathcal{L}_v** [Laroussinie,Larsen,Weise '95]
- **EventClockTL** [Raskin,Schobbens '99]

Erfüllbarkeitsproblem

Gegeben φ . Existiert Modell M , so daß M erfüllt φ ?

Im allgemeinen unentscheidbar für Logiken mit kontinuierlichem Zeitmodell.

Event Recording Fixpunkt Logik (ERL)

Motivation: Bisher keine entscheidbare kontinuierliche Echtzeitlogik mit Fixpunkten.

Beitrag:

Echtzeitlogik basierend auf ERA Modell (determinisierbare Unterklasse der Zeitautomaten).

Erweiterung des μ -Kalküls mit Uhren.

Tableaubasierte Entscheidungsprozedur.

- Entscheidungsprozedur für μ -Kalkül [Emerson/Street '89, Niwiński/Walukiewicz '96]
- Uhren
- Erweiterter Widerspruchs-Begriff

ERL – Beispiele

“Wenn das letzte a Ereignis genau 2 Zeiteinheiten früher stattfand, muß ein b als nächstes kommen.” (time-out)

$$\langle S(a) = 2, b \rangle \text{tt} \wedge [S(a) = 2, \Sigma - b] \text{ff}$$

“Jede Anforderung a ist gefolgt von einer Antwort b innerhalb von 3 Zeiteinheiten und zwei Anforderungen sind durch mindestens 5 Zeiteinheiten getrennt.”

$$\nu X. \langle S(a) = \perp \vee S(a) > 5, a \rangle \langle S(a) < 3, b \rangle X$$

“Es gibt ein maximaler a -Pfad gerader Länge, so daß zwei konsekutive a Ereignisse durch genau 1 Zeiteinheit getrennt sind.”

$$\mu Z. [\text{tt}, a] \text{ff} \vee \langle S(a) = \perp \vee S(a) = 1, a \rangle \langle S(a) = 1, a \rangle Z$$

ERL – Ergebnisse

- **Theorem 1 [Terminierung]:** Jedes Tableau für $C; \varphi \vdash_{\Delta}$ ist endlich.
- **Theorem 2 [Soundness und Completeness]**
Formel φ ist erfüllbar g.d.w. es gibt ein Tableau für φ .
- **Theorem 3 [Komplexität]:** Das Erfüllbarkeitsproblem für ERL ist EXPTIME vollständig in der Größe der Formel ($|\varphi|$), der Größe des Eingabealphabets (m), und in der Größe der (binären) Kodierung der größten Konstante (k), die in den Constraints der Formeln vorkommt.
 $O(m! \cdot 2^m \cdot (2k + 2)^m \cdot 2^{|\varphi|})$
Keine Komplexitätserhöhung gegenüber (zeitfreiem) μ -Kalkül!
- **Theorem 4 [Modellsynthese]:**
Gegeben φ, \mathcal{T} . Es existiert ein ERA M , s.d. $M \models \varphi$
- [Sorea, CONCUR'02]

Weitere Eigenschaften von ERL

Modellüberprüfungsproblem (Model Checking):

- Gegeben ERA M , ERL Eigenschaft φ . Gefragt: $M \stackrel{?}{\models} \varphi$.
- EXPTIME-vollständig (polynomialzeit Reduktion des Akzeptanzproblems für LBATM auf Model Checking Problem).

Charakteristische Formeln für ERA

- Beschreibung des Verhaltens eines Systems bezüglich einer Äquivalenzrelation (z.B. Zeitbisimulation).
- Bisimulation für ERA kann auf MÜP zurückgeführt werden.
 $A \approx B$ g.d.w. $A \models \Phi_B^{\approx}$
- Bisimulation für ERA ist EXPTIME-vollständig.
- Zurückführung von Model Checking auf Erfüllbarkeitsproblem ($\Phi_M^{\approx} \Rightarrow \varphi$).

Technischer Bericht SRI-CSL-01-06 [Sorea, '01].

Ergebnisse – Modellüberprüfung

Vorhandene Techniken

- Regionenbasiert (Regionengraph RG)
- Symbolisch (zonenbasiert) (Zonengraph ZG)

Einschränkungen

- Zustandsraumexplosion
- Nicht kompositionell

Beobachtung

Oftmals ist nur ein Fragment des RG/ZG für die Verifikation der Eigenschaft nötig.

Frage: Welcher Fragment des RG/ZG ist relevant?

↪ Neue Techniken nötig, die eine Konstruktion des RG/ZG vermeiden.

Inhalt

1. Teil I

(a) ...

2. Teil II – Abstraktionsbasierte Verifikationsmethoden

(a) Grundidee

(b) Verzögerte Abstraktion

(c) Verzögertes Theorembeweisen

3. Teil III

(a) ...

Grundidee

- Zurückführung des MÜP für Zeitautomaten auf ein abstrahiertes, leichter zu lösendes MÜP für endliche Systeme.
- Zurücktransformation der gefundenen Lösung auf konkretes System.
- Überprüfung der konkreten Lösung.
- Gegebenenfalls Verfeinerung der Abstraktion.

Inhalt – Teil II

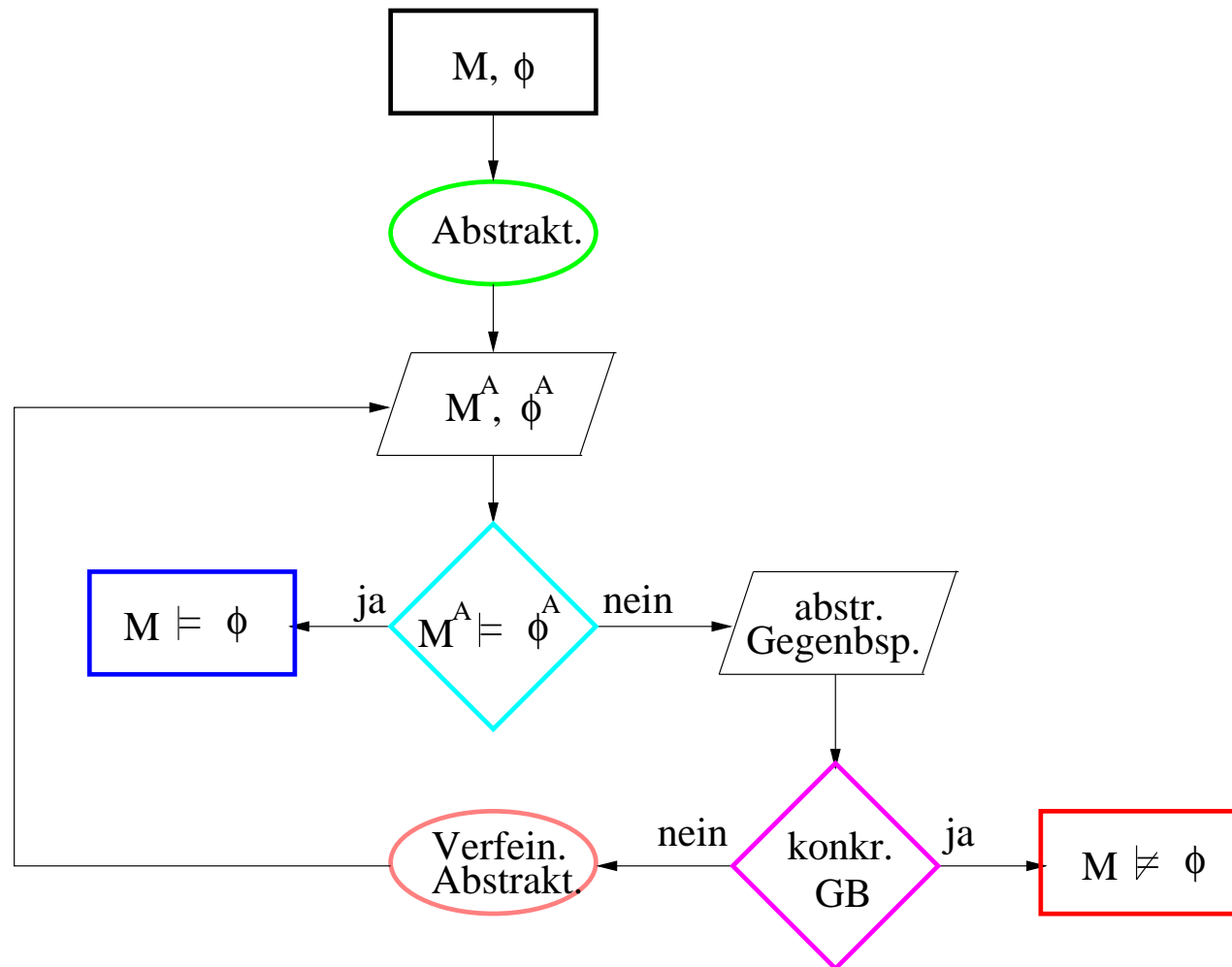
1. Explizite Realisierung: Verzögerte Abstraktion

- (a) Prädikatenabstraktion
- (b) Korrektheit und Vollständigkeit
- (c) Abstraktionsverfeinerungsalgorithmus
- (d) Generierung von Gegenbeispiele

2. Implizite Realisierung: Verzögertes Theorembeweisen

- (a) Beschränkte Modellüberprüfung
- (b) Erfüllbarkeitsproblem für Aussagenlogik mit linearen arithmetischen Constraints
- (c) Verzögerte Kombination von Erfüllbarkeitstest mit Theorembeweisen
- (d) Optimierungen

Verzögerte Abstraktion



Prädikatenabstraktion für Echtzeitsystemen

[O. Möller, H. Rueß, M. Sorea: TPTS '02, ENTCS 65(6), 2002]

Abstraktionprädikaten

- Bezüglich einer Menge Cl von Uhren.
- Formel mit freien Variablen in Cl (z.B. $x \geq 1, x - y < 2$).
- $\Psi = \{\psi_0, \dots, \psi_{n-1}\}$
- Partition des unendlichen Zustandsraums bezüglich deren Wahrheitswerte.

Abstraktionsfunktion

$$\alpha : L \times \mathcal{V}_C \rightarrow L \times B_n$$

$$\alpha(l, \nu)(i) := (l, \psi_i \nu)$$

Beispiel Prädikat $\psi \equiv x = 0$

$$\alpha(l_0, x = 0.5, y = 0.5) := (l_0, \neg\psi)$$

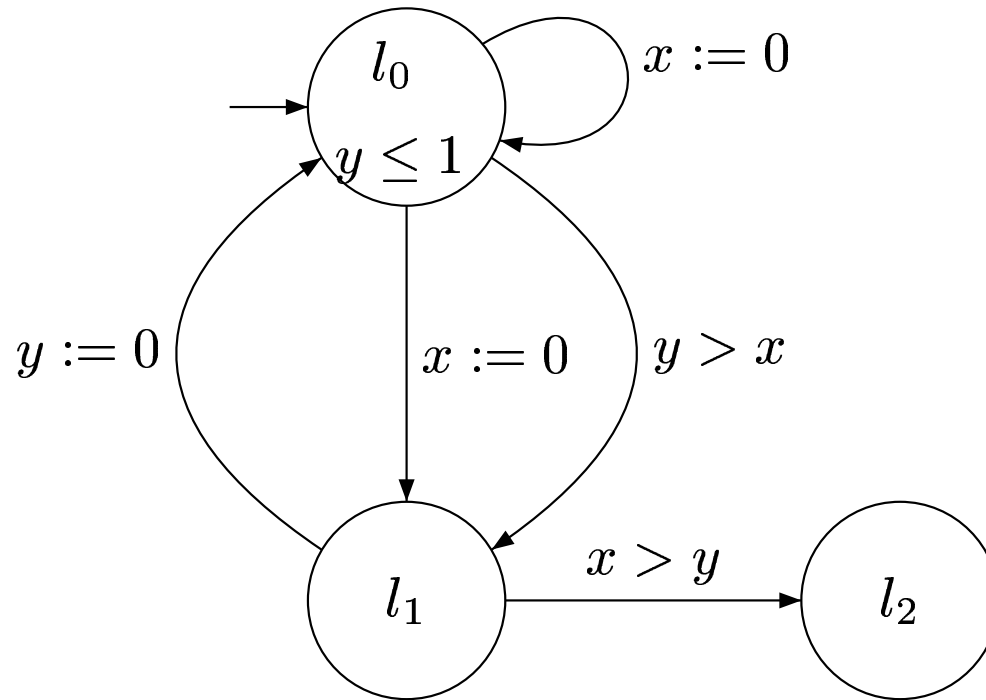
Konkretisierungsfunktion

$$\gamma : L \times B_n \rightarrow L \times \wp(\mathcal{V}_C)$$

$$\gamma(l, b) := \{(l, \nu) \in L \times \mathcal{V}_C \mid I(l) \wedge \bigwedge_{i=0}^{n-1} \psi_i \nu \equiv b(i)\}$$

$$\gamma(l_0, \neg\psi) := \{(l_0, \nu) \mid \nu(x) > 0\}$$

Verzögerte Abstraktion – Beispiel



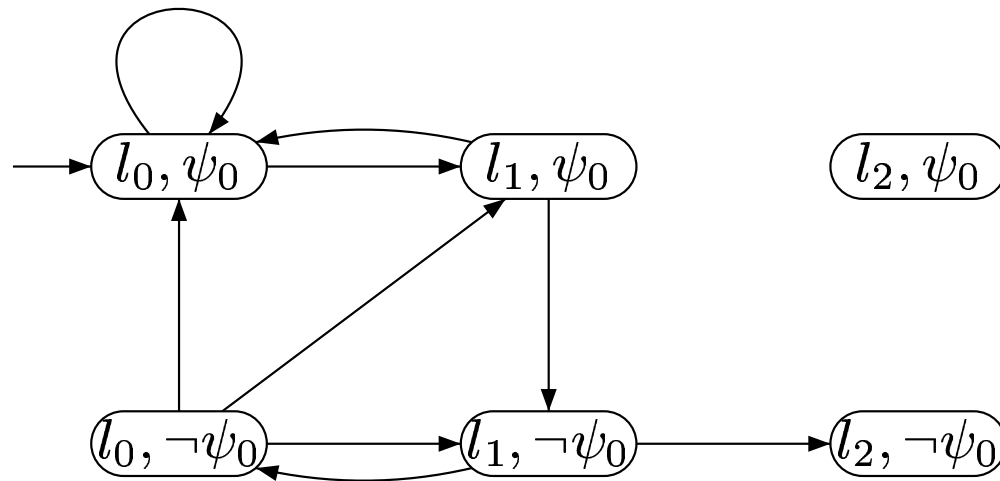
Eigenschaft: $\varphi = \mathbf{AG}(\neg at_l_2) = \neg \mathbf{EF}(at_l_2)$

Abstraktionsprädikate:

$\Psi := \{x = 0, y = 0, x \leq 1, x \geq 1, y \leq 1, y \geq 1, x > y, x < y\}$

Beispiel – Approx. und endl. MÜ

Erste Approximation, $\psi_0 \equiv x = 0$



$\mathcal{M}_{\{x=0\}}^a \stackrel{?}{\models} \neg \mathbf{EF}(at_l_2)$ **NEIN**

Gegenbeispiel: $\tau = \langle (l_0, \psi_0) \Rightarrow^a (l_1, \psi_0) \Rightarrow^a (l_1, \neg\psi_0) \Rightarrow^a (l_2, \neg\psi_0) \rangle$

Beispiel – Überprüfung des GB

$$\tau = \underbrace{((l_0, \psi_0))}_{s_0} \Rightarrow^a \underbrace{(l_1, \psi_0)}_{s_1} \Rightarrow^a \underbrace{(l_1, \neg\psi_0)}_{s_2} \Rightarrow^a \underbrace{(l_2, \neg\psi_0)}_{s_3}$$

Existiert ein zugehöriges GB auf dem konkreten System?

$$\exists \tau^c = (y_0 \Rightarrow^a y_1 \Rightarrow^a y_2 \Rightarrow^a y_3) \text{ s.d.}$$

$$y_0 \in \gamma(s_0), y_1 \in \gamma(s_1), y_2 \in \gamma(s_2), y_3 \in \gamma(s_3), y_0 = s_0^c$$

Ist die Formel F erfüllbar?

$$F := \exists y_0, y_1, y_2, y_3 \in S^C. y_0 \in \gamma(s_0) \wedge y_1 \in \gamma(s_1) \wedge y_2 \in \gamma(s_2) \wedge \\ y_3 \in \gamma(s_3) \wedge y_0 \Rightarrow^a y_1 \wedge y_1 \Rightarrow^a y_2 \wedge y_2 \Rightarrow^a y_3 \wedge y_0 = s_0^c$$

Erfüllbarkeitstest für Aussagenlogik mit linearen arithmetischen Constraints.

Beispiel – Approx.verfeinerung

In diesem Fall ist F unerfüllbar. Es gibt keinen Übergang zwischen y_2 und y_3 .

$$y_0 \in (l_0, x = y = 0) \in \gamma(s_0)$$

↓

$$y_1 \in (l_1, x = 0 \wedge 0 \leq y \leq 1) \in \gamma(s_1)$$

↓

$$y_2 \in (l_1, x > 0 \wedge x \leq y) \in \gamma(s_2)$$

↓

$$y_3 \in (l_2, x > 0 \wedge y \geq 0) \in \gamma(s_3)$$

Sei k s.d. es gibt ein Pfad von y_0 nach y_k aber nicht von y_k nach

$$y_{k+1}: \boxed{k = 2}$$

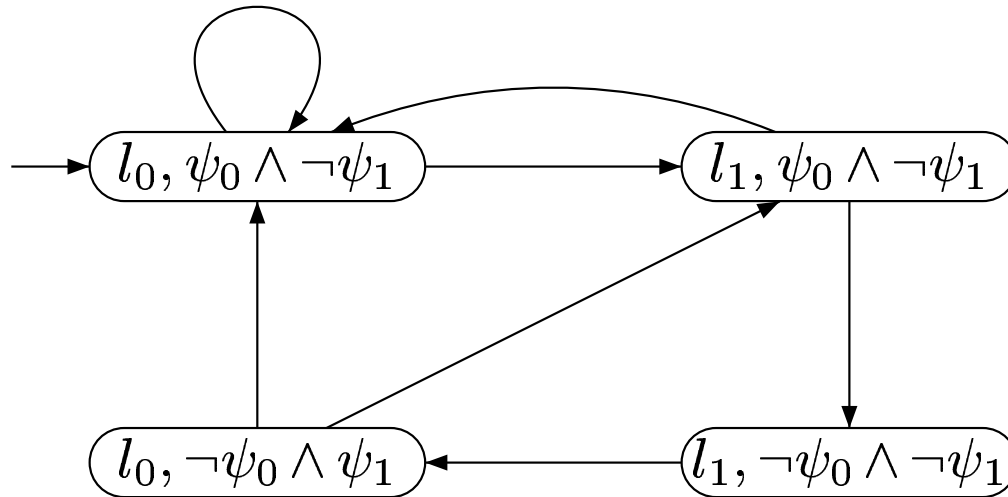
Wähle $\psi_1 \in \Psi$ s.d. $\exists y_i, y_j \in S^C. y_i \in \gamma(s_2) \wedge y_j \in \gamma(s_3) \wedge y_i \not\Rightarrow y_j$ gültig.

$$\boxed{\psi_1 \equiv x > y}$$

Beispiel (Forts. IV)

Neue Approximation $\mathcal{M}_{\{x=0, x>y\}}^a$

Erfüllt Formel $\varphi = \mathbf{AG}(\neg at_l_2)$



Algorithmus terminiert mit **true**

$\mathcal{M} \models \mathbf{AG}(\neg at_l_2)$

Korrektheit und Vollständigkeit

Basis

Für einen gegebenen Zeitautomaten, eine Basis ist eine Menge Ψ von Abstraktionsprädikaten, die ausdrucksstark genug ist, um zwischen Uhrenregionen zu unterscheiden.

Theorem 1 Falls Ψ Basis dann

$$\mathcal{M} \models \varphi \Leftrightarrow \mathcal{M}_{\Psi}^a \models \varphi_{\Psi}^a$$

Theorem 2 Falls $\Psi_1 \subset \Psi$ dann

$$\mathcal{M} \models \varphi \Leftarrow \mathcal{M}_{\Psi}^a \models \varphi_{\Psi}^a$$

Zusammenfassung

Was kann verifiziert werden

- Sicherheitseigenschaften
- Lebendigkeitseigenschaften

Beobachtungen

- Problem der Selbstschleifen: eingeschränkter Verzögerungsschritt.
- Logik beinhaltet keine Zeit (μ -Kalkül).
- Verallgemeinerung auf Echtzeitlogiken (z.B. ERL).
- Anpassung auf komplexere Modelle leicht möglich.
- Problem: Generierung von Gegenbeispielen für CTL, μ -Kalkül.

Generierung von Gegenbeispielen (GB)

N. Shankar, M. Sorea [abgelehnt CAV '03]

Gegeben: \mathcal{M}, I, φ

Gefragt: $\mathcal{M}, I \models \varphi$

Antwort = Nein \rightsquigarrow Gegenbeispiel als Pfad im \mathcal{M}

Schön, aber:

Pfadbasierte GB nur für kleines Fragment von CTL (z.B. SMV)

Antwort = Ja \rightsquigarrow Zeuge ?

Generierung von GB – Ansatz

Symbolischer Modellüberprüfungsalgorithmus.

Zeugen und GB für CTL: Bäume über Mengen von Zuständen.

Beschränkte Vorwärtsauffaltung des Zustandsraumes, gefolgt von einer lokalen Fixpunktberechnung.

Partition von I in I^+ und I^- , mit $\mathcal{M}, I^+ \models \varphi$ und $\mathcal{M}, I^- \not\models \varphi$.

Vorläufige Resultate weisen darauf hin, daß die Methode bedeutend effizienter ist als Standardmethoden zur CTL Modellüberprüfung.

Theoretische Ergebnisse

Sei $\mathbf{WMC}(\varphi, \mathcal{M}, I)$ MÜ Algorithmus für $\mathcal{M}, I \models \varphi$.

Ein Zeuge ist eine Begründung dafür daß $\mathcal{M}, G \models \varphi$: $w \vdash \mathcal{M}, G \models \varphi$.

Ein GB ist eine Begründung dafür, daß $\mathcal{M}, C \not\models \varphi$: $c \vdash \mathcal{M}, C \not\models \varphi$.

Theorem 1 [Gültigkeit der ZG/GB]

Falls $w \vdash \mathcal{M}, G \models \varphi$ dann $\mathcal{M}, G \models \varphi$, und falls $c \vdash \mathcal{M}, C \not\models \varphi$, dann $\mathcal{M}, C \not\models \varphi$.

Theorem 2 [Korrektheit]

Sei O die Ausgabe des Algorithmus. Dann existieren disjunkte Mengen von Zuständen G und C mit $I \subseteq G \cup C$, ein ZG $w = \mathit{witness}(\varphi, \mathcal{M}, O)$, und ein GB $c = \mathit{counter}(\varphi, \mathcal{M}, O)$, s.d. $w \vdash \mathcal{M}, G \models \varphi$ und $c \vdash \mathcal{M}, C \not\models \varphi$.

Experimentelle Ergebnisse – Synchrone Arbiter

“Mutual exclusion” Eigenschaft in fehlerhafter Version.

# Zellen	WMC			MC		
	BDD Größe	Msek	# Iterat.	BDD Größe	Msek	# Iterat.
5	40	0	3	381	10	3
10	92	130	3	1041	260	4
15	142	80	3	1317	440	3
20	199	229	3	2297	1210	5
25	251	290	3	4838	3510	4
30	301	470	3	5533	2500	3
35	347	530	3	5657	1710	3
40	407	660	3	9976	11360	4
45	455	880	3	6801	4804	3
47	101	150	2	3338	1330	3

Implizite Realisierung der Grundidee

1. Abstrakte Interpretation
2. Explizite Realisierung: Verzögerte Abstraktion
 - (a) ...
3. Implizite Realisierung: Verzögertes Theorembeweisen
 - (a) Beschränkte Modellüberprüfung
 - (b) Erfüllbarkeitsproblem für Aussagenlogik mit linearen arithmetischen Constraints
 - (c) Verzögerte Kombination von Erfüllbarkeitstest mit Theorembeweisen
 - (d) Optimierungen

Implizite Realisierung – Kurzfassung

MÜP \longrightarrow **BMC** \longrightarrow **Erüllbarkeitsprob. für Bool(\mathcal{A})** \longrightarrow **Verzögertes TB**

Verzögertes Theorembeweisen (VTB)

- Kombination von Erüllbarkeitstest (SAT solver) mit TB
- Kern: Verfeinerungsalgorithmus – aussagenlogische Abstraktionen von Bool(\mathcal{A}) Formeln werden inkrementell verfeinert.
- Verfeinerung basiert auf Analyse von scheinbaren GB, d.h. GB die vom SAT solver generiert, aber vom TB verworfen wurden durch Interpretation der aussagenlogischen Atome.
 - $p \equiv x > 2, q \equiv x = 1$
 - Erfüllbarkeitsbelegung: $p, \neg q$
 - TB stellt entsprechende Inkonsistenz fest, und generiert Lemma $\neg p \wedge q$ um die Suche weiter einzuschränken.

Beschränkte Modellüberprüfung (Bounded Model Checking)

Intuition: Gibt es ein Gegenbeispiel der Länge k für das MÜP $M \models \varphi$?

Zeitautomat: $\mathcal{S} = \langle L, l_0, Cl, E, Inv \rangle$, $Cl = \{x_1, \dots, x_n\}$, Zeitconstr. Φ

Theorie der linearen arithmetischen Constraints:

$$\mathcal{A} = \Phi \cup \{x' - x = y' - y \mid x, x', y, y' \in Cl\}$$

$\text{Bool}(\mathcal{A})$ aussagenlogische Kombination der Atome in \mathcal{A} .

BMC(\mathcal{A}) Probleme werden beschrieben durch:

- Kodierung von \mathcal{S} als Formel in $\text{Bool}(\mathcal{A})$
- LTL(\mathcal{A}) Formel φ , und
- Schranke k

Lösung des BMC Problems

Reduktion auf das Erfüllbarkeitsproblem für $\text{Bool}(\mathcal{A})$.

$$I(s_0) \wedge T(s_0, s_1) \wedge \dots \wedge T(s_{k-1}, s_k) \wedge (\neg\varphi(s_0) \vee \dots \vee \neg\varphi(s_k))$$

Erfüllbarkeitstests (SAT solvers): Extraktion von Gegenbeispielen für MÜP aus Modellen der aussagenlogischen Formel.

BMC Problem

1. $\llbracket M \rrbracket_k$ als Auffaltung des Programs M bis zu k Schritte beginnend im Initialzustand (k disjunkte Kopien von V nötig).

$$\llbracket M \rrbracket_k = I_0(M) \wedge T_0(M) \wedge \dots \wedge T_{k-1}(M)$$

2. Transformation von $\neg\varphi$ in zugehörigen Büchiautomaton $\mathcal{B}_{\neg\varphi}$. Die akzeptierte Sprache besteht aus den Pfaden, die $\neg\varphi$ erfüllen.

3. Kodierung des Zustandsübergangssystem von $\mathcal{B}_{\neg\varphi}$ und der Akzeptanzbedingung für Büchiautomaten als $\llbracket \mathcal{B} \rrbracket_k$.

$$\llbracket \mathcal{B} \rrbracket_k = I_0(\mathcal{B}) \wedge T_0(\mathcal{B}) \wedge \dots \wedge T_{k-1}(\mathcal{B}) \wedge acc(\mathcal{B})_k$$

4. $\llbracket M, \varphi \rrbracket_k := \llbracket \mathcal{B} \rrbracket_k \wedge \llbracket M \rrbracket_k$.

5. Eine erfüllende Variablenzuweisung für die Formel $\llbracket M, \varphi \rrbracket_k$ induziert Gegenbeispiele der Länge k für das MÜP $M \models \varphi$.

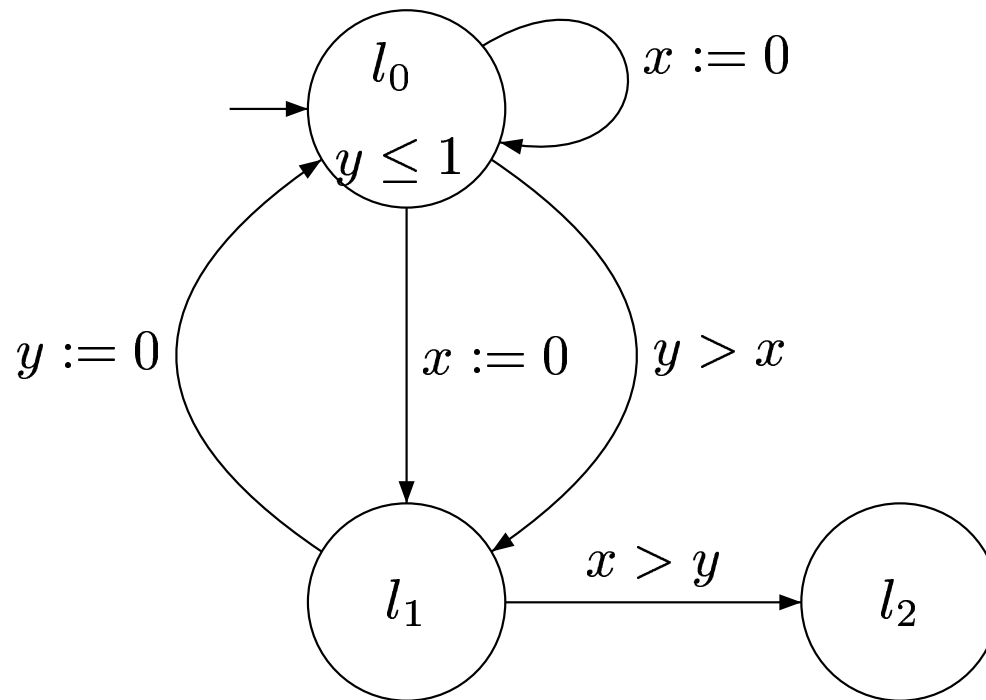
Zeitautomaten als Bool(\mathcal{A}) Formel

Definition von \mathcal{S} als \mathcal{A} -Program $M = \langle I, T \rangle$ über einer Menge V von Zustandsvariablen.

1. Definition des Initialzustands I
2. Definition der Zustandsübergangsrelation T
 - Definition eines Zustandsübergangsschrittes
 - Definition eines Zeitverzögerungsschrittes

Konstruktion des Regionengraphen nicht notwendig.

BMC – Beispiel



Zustände:

- Zustandsvariablen $V = \{at, x, y\}$ und $V' = \{at', x', y'\}$.
- at, at' interpretiert über $\{l_0, l_1, l_2\}$, und x, y, x', y' über \mathbb{R}_0^+ .
- Ein **Zustand** ist eine Zuweisung an diese Variablen.

Beispiel (Forts.)

Initialzustand:

$$I(at, x, y) := (at = l_0 \wedge x = 0 \wedge y = 0)$$

Übergangsrelation:

$$T(at, x, y, at', x', y') :=$$

$$(at = l_0 \wedge x' = 0 \wedge y' = y \wedge at' = l_0) \vee$$

$$(at = l_0 \wedge x' = 0 \wedge y' = y \wedge at' = l_1) \vee$$

$$(at = l_0 \wedge y > x \wedge x' = x \wedge y' = y \wedge at' = l_1) \vee$$

$$(at = l_1 \wedge y' = 0 \wedge x' = x \wedge at' = l_0) \vee$$

$$(at = l_1 \wedge x \geq y \wedge x' = x \wedge y' = y \wedge at' = l_2) \vee$$

$$D(at, x, y, at', x', y')$$

Beispiel (Forts.)

Zeitverzögerungsschritte:

$$D(at, x, y, at', x', y') := \\ \exists \delta \geq 0. ((at = l_0 \Rightarrow y' \leq 1) \wedge at' = at \wedge \\ x' = x + \delta \wedge y' = y + \delta).$$

Quantorenelimination:

$$D(at, x, y, at', x', y') := \\ (at = l_0 \Rightarrow y' \leq 1) \wedge at' = at \wedge \\ x' - x \geq 0 \wedge y' - y = x' - x.$$

Beispiel – Auffaltung des Programs

$$\llbracket simple \rrbracket_2 = I_0(\text{simple}) \wedge T_0(\text{simple}) \wedge T_1(\text{simple})$$

Initialzustand: $I_0(\text{simple}) = (at[0] = l_0 \wedge x[0] = 0 \wedge y[0] = 0)$

i . Auffaltung ($i = 0, 1$): $T_i(\text{simple})$

$$(at[i] = l_0 \wedge x[i+1] = 0 \wedge y[i+1] = y[i] \wedge at[i+1] = l_0) \vee$$

$$(at[i] = l_0 \wedge x[i+1] = 0 \wedge y[i+1] = y[i] \wedge at[i+1] = l_1) \vee$$

$$(at[i] = l_0 \wedge y[i] > x[i] \wedge x[i+1] = x[i] \wedge y[i+1] = y[i] \wedge at[i+1] = l_1) \vee$$

$$(at[i] = l_1 \wedge y[i+1] = 0 \wedge x[i+1] = x[i] \wedge at[i+1] = l_0) \vee$$

$$(at[i] = l_1 \wedge x[i] \geq y[i] \wedge x[i+1] = x[i] \wedge y[i+1] = y[i] \wedge at[i+1] = l_2) \vee$$

$$((at[i] = l_0 \Rightarrow y[i+1] \leq 1) \wedge at[i+1] = at[i] \wedge$$

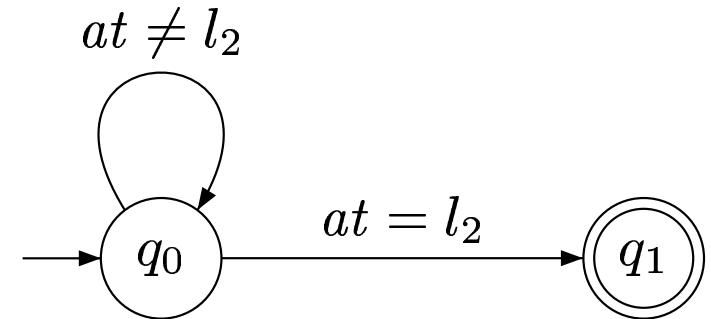
$$x[i+1] - x[i] \geq 0 \wedge y[i+1] - y[i] = x[i+1] - x[i])$$

Beispiel – Auffaltung des Büchi Automaten

Annahme: $\varphi = \mathbf{G} \neg(at = l_2)$

$\neg\varphi = \mathbf{F} (at = l_2)$

$[[\mathcal{B}]]_2 := I_0(\mathcal{B}) \wedge T_0(\mathcal{B}) \wedge T_1(\mathcal{B}) \wedge acc(\mathcal{B})_2$



$I_0(\mathcal{B}) := (pc[0] = q_0)$

$T_0(\mathcal{B}) := (pc[0] = q_0 \wedge \neg(at[0] = l_2) \wedge pc[1] = q_0) \vee$
 $(pc[0] = q_0 \wedge at[0] = l_2 \wedge pc[1] = q_1)$

$T_1(\mathcal{B}) := (pc[1] = q_0 \wedge \neg(at[1] = l_2) \wedge pc[2] = q_0) \vee$
 $(pc[1] = q_0 \wedge at[1] = l_2 \wedge pc[2] = q_1)$

$acc(\mathcal{B})_2 := (pc[0] = q_1 \vee pc[1] = q_1 \vee pc[2] = q_1)$

Beispiel – BMC Problem

$$[[M, \varphi]]_2 = [[M]]_2 \wedge [[\mathcal{B}]]_2$$

Gegenbeispiel für $M \models \mathbf{G} \neg(at = l_2)$

$$(l_0, x = 0, y = 0) \rightarrow (l_1, x = 0, y = 0) \rightarrow (l_2, x = 0, y = 0)$$

BMC für Zeitautomaten – Ergebnisse

- **Theorem** [Korrektheit und Vollständigkeit]

$$M \not\models \varphi \text{ g.d.w. } \exists k \in \mathbb{N}. \llbracket M, \varphi \rrbracket_k \text{ ist erfüllbar}$$

- Untere Schranke für k : Länge des kürzesten Pfad vom Startzustand zu Endzustand im Büchiauxtomaten.
- Obere Schranke für k : $k \leq 2^{O(t \log(ct))} \cdot 2^{O(|\varphi|)}$
- $k = 2^{O(t \log(ct))} \cdot 2^{O(|\varphi|)}$ dann vollständige Verifikation:

$$M \models \varphi \text{ g.d.w. } \llbracket M, \varphi \rrbracket_j \text{ ist unerfüllbar für alle } j \leq k$$

- Kompositionell $A_1 \parallel \dots \parallel A_n$ wird kodiert als $\langle I_1, T_1 \rangle \wedge \dots \wedge \langle I_n, T_n \rangle$.
- [Sorea: (MTCS'02), ENTCS 68(5), 2002].

Erfüllbarkeitsproblem für $\text{Bool}(\mathcal{A})$

Gegeben

- Aussagenlogische Formel mit linearen arithmetischen Constraints (\mathcal{A}) als Literale.

Gesucht

- Effektive Lösung des Erfüllbarkeitsproblem für $\text{Bool}(\mathcal{A})$.

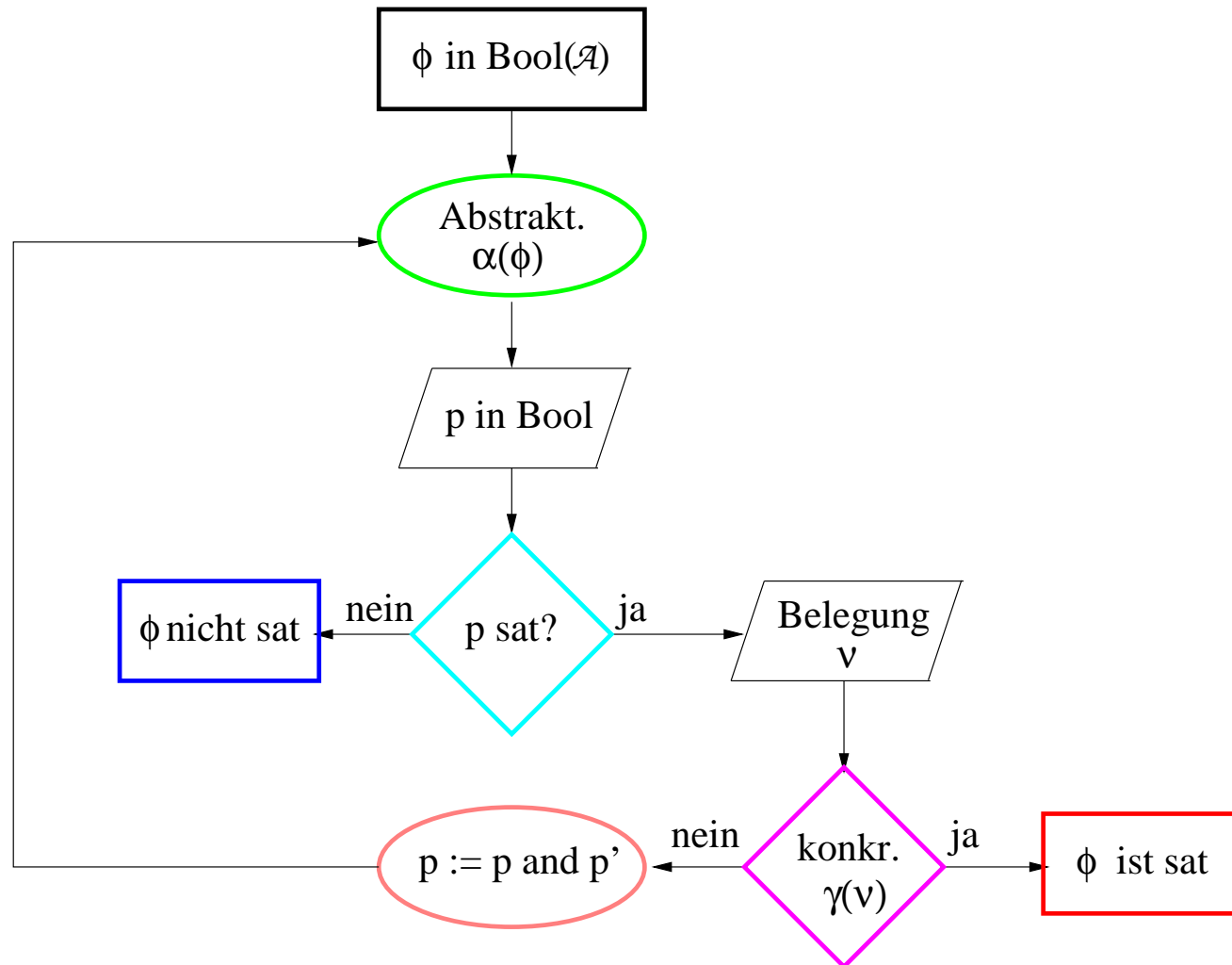
Ansatz

- Kombination von Erfüllbarkeitstests (SAT solvers) mit Theorembeweiser.
- Verfeinerungsalgorithmus basiert auf *verzögertem Theorembeweisen*.

Das Verfeinerungsprozess für aussagenlogische Formeln funktioniert ähnlich zur Verfeinerung von Abstraktionen basierend auf scheinbaren Gegenbeispiele.

[L. de Moura, H. Rueß, M. Sorea. CADE'02].

Verzögertes Theorembeweisen



Beispiel

$$\varphi \equiv (x = y \vee x > y) \wedge x > 2 \wedge y = 1$$

Abstraktion: $p = \alpha(\varphi) = (p_1 \vee p_2) \wedge p_3 \wedge p_4$

Belegung: $\nu = [p_1 \mapsto 1, p_2 \mapsto 0, p_3 \mapsto 1, p_4 \mapsto 1]$

Konkretisierung: $\gamma(\nu) = \{x = y, x \leq y, x > 2, y = 1\}$ **inkonsistent**

Verfeinerung: $p := (\neg p_1 \vee p_2 \vee \neg p_3 \vee \neg p_4) \wedge p$

Belegung: $\nu = [p_1 \mapsto 0, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 1]$

Konkretisierung: $\gamma(\nu) = \{x \neq y, x > y, x > 2, y = 1\}$ **konsistent**

φ ist erfüllbar: $[(x = y) \mapsto 0, (x > y) \mapsto 1, (x > 2) \mapsto 1, (y = 1) \mapsto 1]$

BMC – Experimentelle Ergebnisse

System Name	$k = 5$	$k = 10$	$k = 15$	$k = 20$
Bakery Protokoll	0.01s	0.18s	0.641s	1.6s
Simpson Protokoll	0.039s	0.271s	1.635s	18.05s
Train Gate Controller	0.059s	0.38s	2.77s	17.93s
Fischer Protokoll	0.033s	1.066s	1.17s	1.256s
Water Level Monitor	0.031s	0.047s	0.057s	0.084s
Evader-Pursuer	0.74s	0.78s	0.88s	0.99s
Leaking Gas Burner	0.035s	0.117s	0.35s	0.85s
Multi Rate Fischer	0.033s	1.98s	2.09s	2.20s

K-Induktion als Optimierung von BMC

Beobachtung

- Berechnung des Diameters für Zeitautomaten: PSPACE vollständig.
- Diameter existiert nicht für Hybride Systeme.

Ansatz

- Kombination von BMC mit k-Induktion.

k-Induktion

- Verifizierung von Sicherheitseigenschaften mittels Induktion nur für induktive Eigenschaften. Problem: Finden induktiver Eigenschaften.
- φ ist *k-induktiv*
 - φ gilt in den ersten k Schritte startend im Initialzustand.
 - Falls φ in jedem Zustand jeder Systemausführung der Länge k gilt, dann gilt φ auch in jedem Nachfolgezustand.

K-Induktion – Schema

$$\mathbf{IND}(k)(\varphi)$$

Basisfall $I(s_0) \wedge \pi(s_0, \dots, s_{k-1}) \rightarrow \varphi(s_0) \wedge \dots \wedge \varphi(s_{k-1})$

Schritt $\forall n . \varphi(s_n) \wedge \dots \wedge \varphi(s_{n+k}) \wedge \pi(s_k, \dots, s_{n+k}) \rightarrow \varphi(s_{n+k+1})$

Theorem: k-Induktion ist eine vollständige Beweismethode für
Zeitautomaten.

$$M \models \Box\varphi \text{ iff } \exists k. \mathbf{IND}(k)(\varphi)$$

Obere Schranke für k : Anzahl der Uhrenregionen.

K-Induktion – Experimentelle Ergebnisse

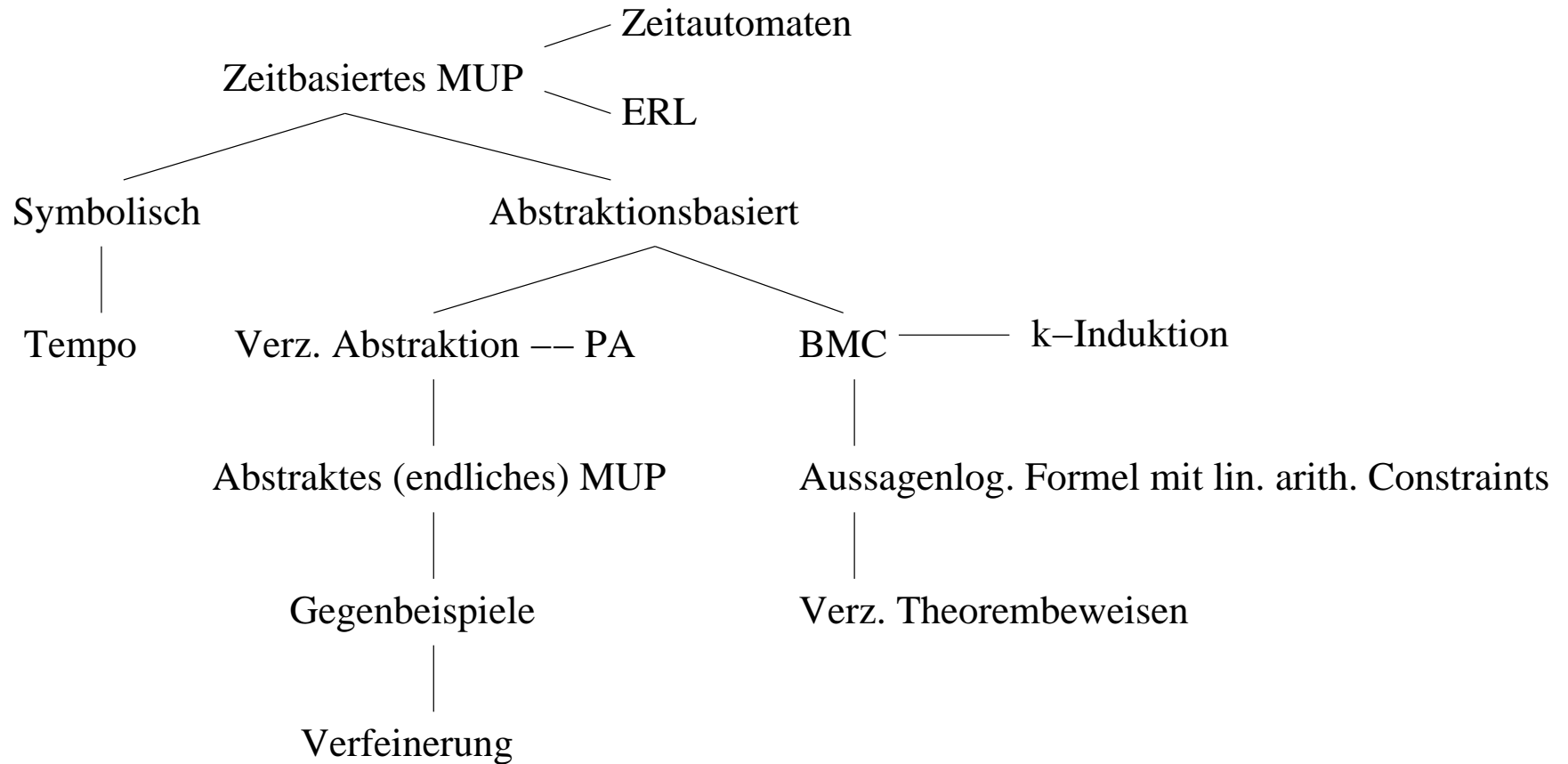
System Name	Bewiesen mit k -Induktion	Zeit (Sek.)
Train Gate Controller	5	0.52
Fischer Protokoll	4	0.71
Water Level Monitor	1	0.08
Leaking Gas Burner	6	1.13
Multi Rate Fischer	4	0.84

Geplante Fallbeispiele

Startup Protokolle für TTA.

Modellierung eines biologischen Systems.

Zusammenfassung



Beiträge

- M. Sorea. Tempo: A model-checker for event-recording automata. In *Proceedings of RT-TOOLS'01*, Aalborg, Denmark, August 2001.
- M.O. Möller, H. Rueß, M. Sorea. Predicate Abstraction for Dense Real-Time Systems. In *Proceedings of Theory and Practice of Timed Systems (TPTS'02)*, volume 65 (6) of *Electronic Notes in Theoretical Computer Science*, 2002.
- L. de Moura, H. Rueß, M. Sorea. Lazy Theorem Proving for Bounded Model Checking over Infinite Domains. In *Proceedings of 18th Conference on Automated Deduction (CADE)*, 2002.
- M. Sorea. A Decidable Fixpoint Logic for Time-Outs. In L. Brim, P. Jancar, and M. Kretinsky, eds., *Proceedings of the 13th International Conference on Concurrency (CONCUR)*, LNCS 2421, pp. 255–272, 2002.

- M. Sorea. Bounded Model Checking for Timed Automata. In *Proceedings of MTCS'02*, volume 68 (5) of *Electronic Notes in Theoretical Computer Science*, 2002.
- L. de Moura, H. Rueß, M. Sorea. Bounded Model Checking and Induction: From Refutation to Verification. Accepted to CAV'2003.
- N. Shankar, M. Sorea. Counterexample-Driven Model Checking. Rejected from CAV'03.