Temporalne własności systemów i model checking

Konrad Błachnio
K.Blachnio@students.mimuw.edu.pl

MIMUW
4 listopada 2009

1. Model checking
   - Idea
   - Przykłady
   - Model checking a temporalne własności systemów

2. Temporalne własności systemów
   - Program jako struktura Kripkego
   - LTL - Linear temporal logic
   - CTL - Computation tree logic
   - CTL i LTL porównanie

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Przykład - agitacja za

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Przykład - agitacja za

- System współbieżny

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Przykład - agitacja za

- System współbieżny
- Pożądane właściwości/zachowanie systemu

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Przykład - agitacja za

- System współbieżny
- Pożądane właściwości/zachowanie systemu
- Skomplikowana struktura systemu/ system wielowątkowy

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Przykład - agitacja za

- System współbieżny
- Pożądane właściwości/zachowanie systemu
- Skomplikowana struktura systemu/ system wielowątkowy
- Chcemy zweryfikować pewną własność automatycznie i powtarzalnie

Co to jest model checking?

Idea - weryfikacja modelowa

Automatyczna weryfikacja danego modelu pod kątem zadanych własności.

Przykład - agitacja za

- System współbieżny
- Pożądane właściwości/zachowanie systemu
- Skomplikowana struktura systemu/ system wielowątkowy
- Chcemy zweryfikować pewną własność automatycznie i powtarzalnie
- Np. dostęp do sekcji krytycznej czy zagłodzenie procesu

Model checking - przykład

Weryfikacja algorytmu

- Algorytm wejścia do sekcji krytycznej
- Chcemy sprawdzić, czy:

Model checking - przykład

Weryfikacja algorytmu

- Algorytm wejścia do sekcji krytycznej
- Chcemy sprawdzić, czy:
- Jeśli wejdę do sekcji krytycznej to będę tam sam

Model checking - przykład

Weryfikacja algorytmu

- Algorytm wejścia do sekcji krytycznej
- Chcemy sprawdzić, czy:
- Jeśli wejdę do sekcji krytycznej to będę tam sam
- Jeśli chcę to zawsze kiedyś wejdę do sekcji krytycznej

Model checking - przykład

Weryfikacja algorytmu

- Algorytm wejścia do sekcji krytycznej
- Chcemy sprawdzić, czy:
- Jeśli wejdę do sekcji krytycznej to będę tam sam
- Jeśli chcę to zawsze kiedyś wejdę do sekcji krytycznej
- Jeśli nieskończenie wiele razy chcę to zawsze kiedyś wejdę do sekcji krytycznej

Model checking - przykład

Weryfikacja algorytmu

- Algorytm wejścia do sekcji krytycznej
- Chcemy sprawdzić, czy:
- Jeśli wejdę do sekcji krytycznej to będę tam sam
- Jeśli chcę to zawsze kiedyś wejdę do sekcji krytycznej
- Jeśli nieskończenie wiele razy chcę to zawsze kiedyś wejdę do sekcji krytycznej
- Jeśli wejdę do sekcji krytycznej to zawsze kiedyś wyjdę

Model checking - przykład

Weryfikacja programu

- Nowy scheduler dla systemu operacyjnego
- Chcemy sprawdzić, czy:

Model checking - przykład

Weryfikacja programu

- Nowy scheduler dla systemu operacyjnego
- Chcemy sprawdzić, czy:
- Zawsze kiedyś dostanę procesor

Model checking - przykład

Weryfikacja programu

- Nowy scheduler dla systemu operacyjnego
- Chcemy sprawdzić, czy:
- Zawsze kiedyś dostanę procesor
- Zawsze nieskończenie wiele razy dostanę procesor

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ
- Prawie zawsze φ

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ
- Prawie zawsze φ
- Nieskończenie wiele razy φ

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ
- Prawie zawsze φ
- Nieskończenie wiele razy φ
- Jeśli kiedyś φ_1 to zawsze w końcu φ_2

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ
- Prawie zawsze φ
- Nieskończenie wiele razy φ
- Jeśli kiedyś φ_1 to zawsze w końcu φ_2
- Jeśli kiedyś φ_1 to potem zawsze φ_2

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ
- Prawie zawsze φ
- Nieskończenie wiele razy φ
- Jeśli kiedyś φ_1 to zawsze w końcu φ_2
- Jeśli kiedyś φ_1 to potem zawsze φ_2

Chcemy aby nasz napis był zrozumiały i jednoznaczny:

Model checking a temporalne własności systemów

Temporalne własności systemów chcemy sprawdzać, np:

- Kiedyś φ
- Prawie zawsze φ
- Nieskończenie wiele razy φ
- Jeśli kiedyś φ_1 to zawsze w końcu φ_2
- Jeśli kiedyś φ_1 to potem zawsze φ_2

Chcemy aby nasz napis był zrozumiały i jednoznaczny:

- LTL - Linear temporal logic
- CTL - Computation tree logic

Formalizacja programu

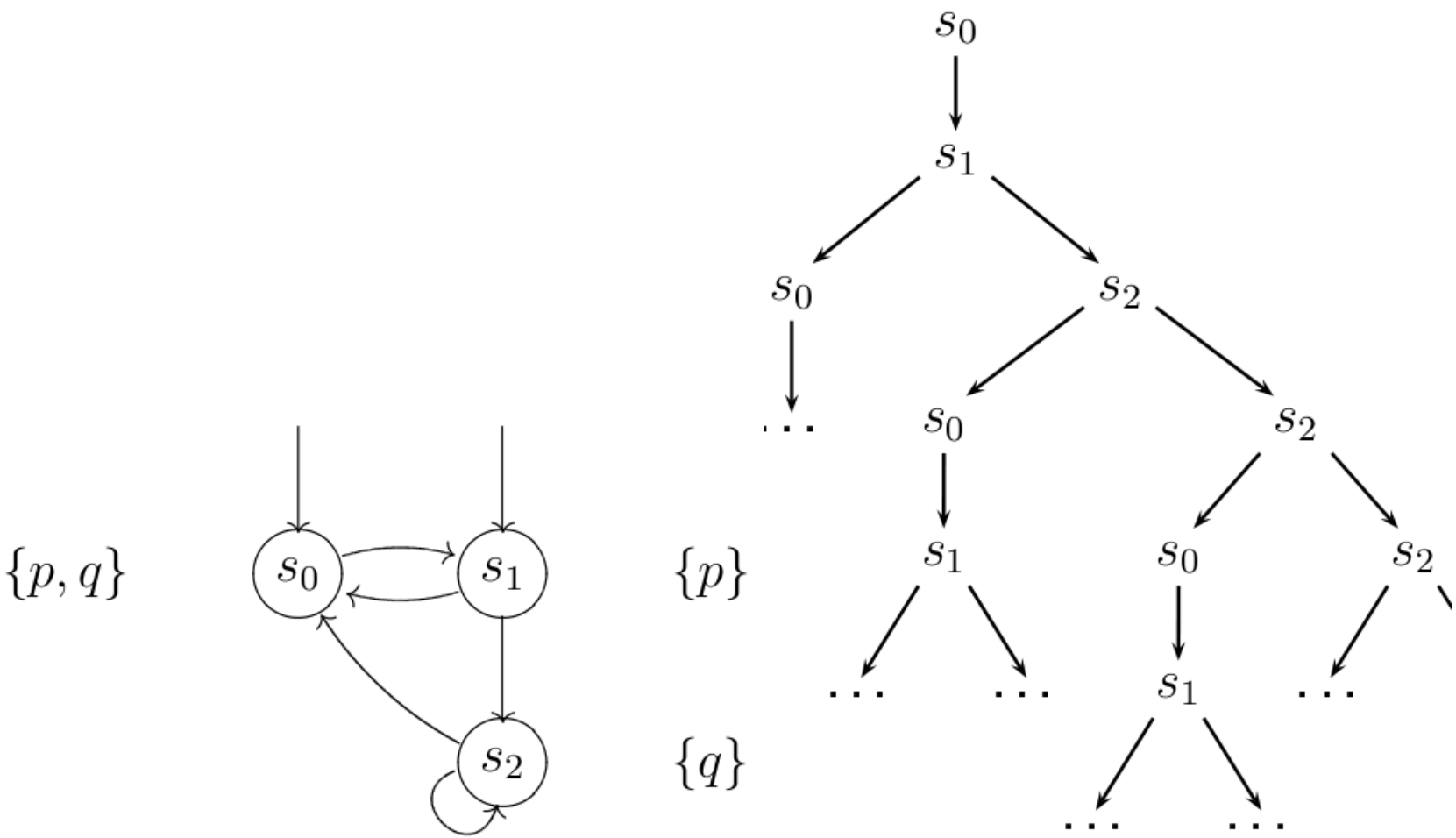
Formalizacja

- Chcemy udowodnić formalnie własność programu
- Potrzebujemy formalnego opisu programu: Struktura Kripkego

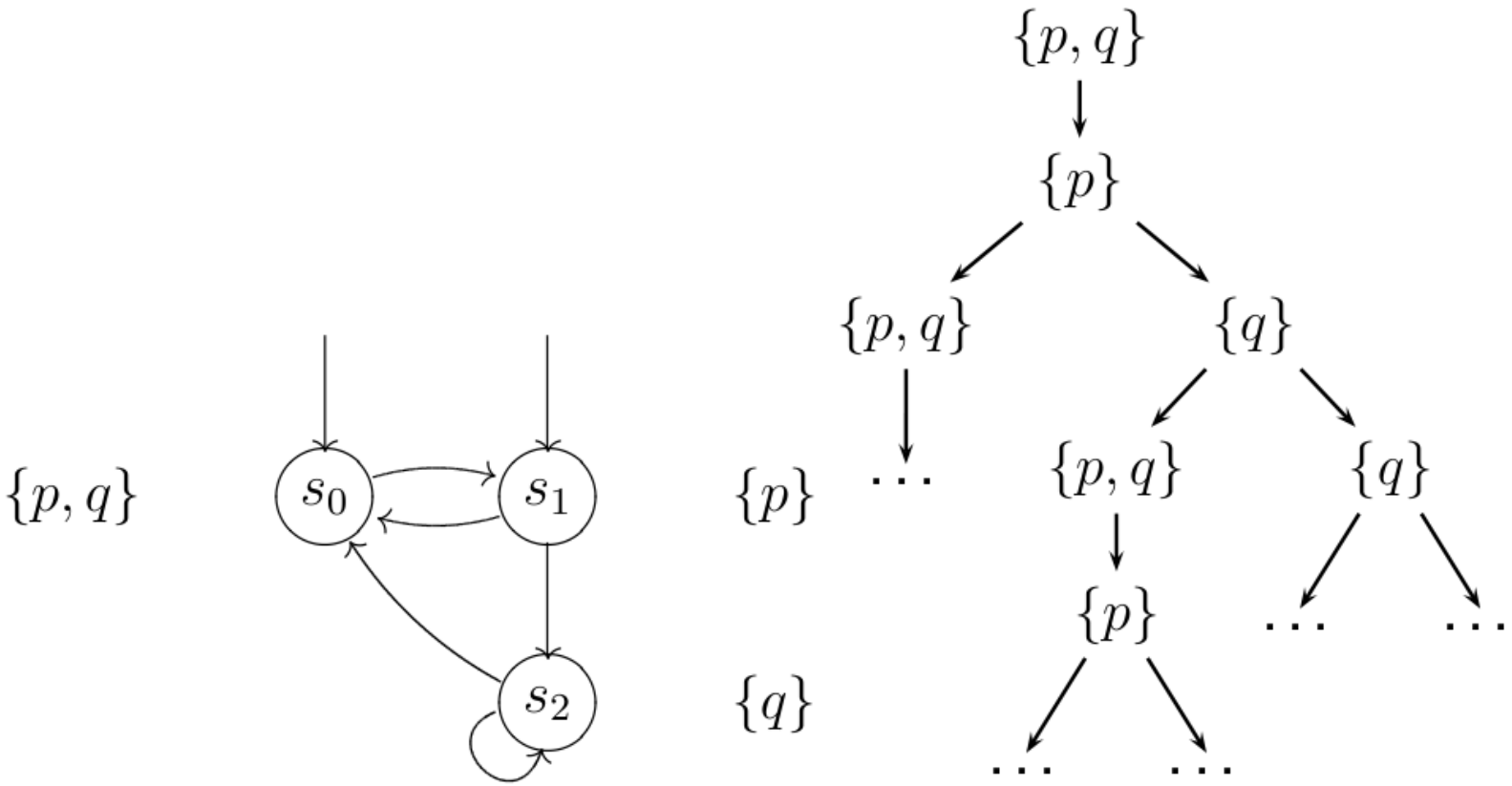
Struktura Kripkego

Def.: Struktura Kripkego $\kappa = <\Omega, \Omega_p, \rightarrow, \lambda>$
$\Omega_p \subseteq \Omega$ niepusty zbiór stanów początkowych
$\rightarrow \Omega \times \Omega$ relacja przejścia
$\lambda : \Omega \rightarrow P(\mathrm{A})$, gdzie A - zmienne zdaniowe (własności atomowe)

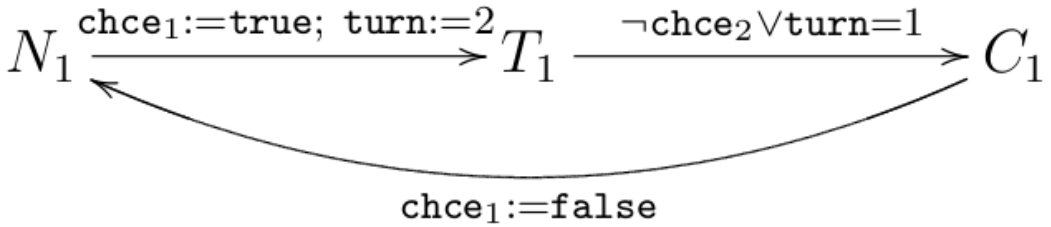
Struktura Kripkego - Przykład

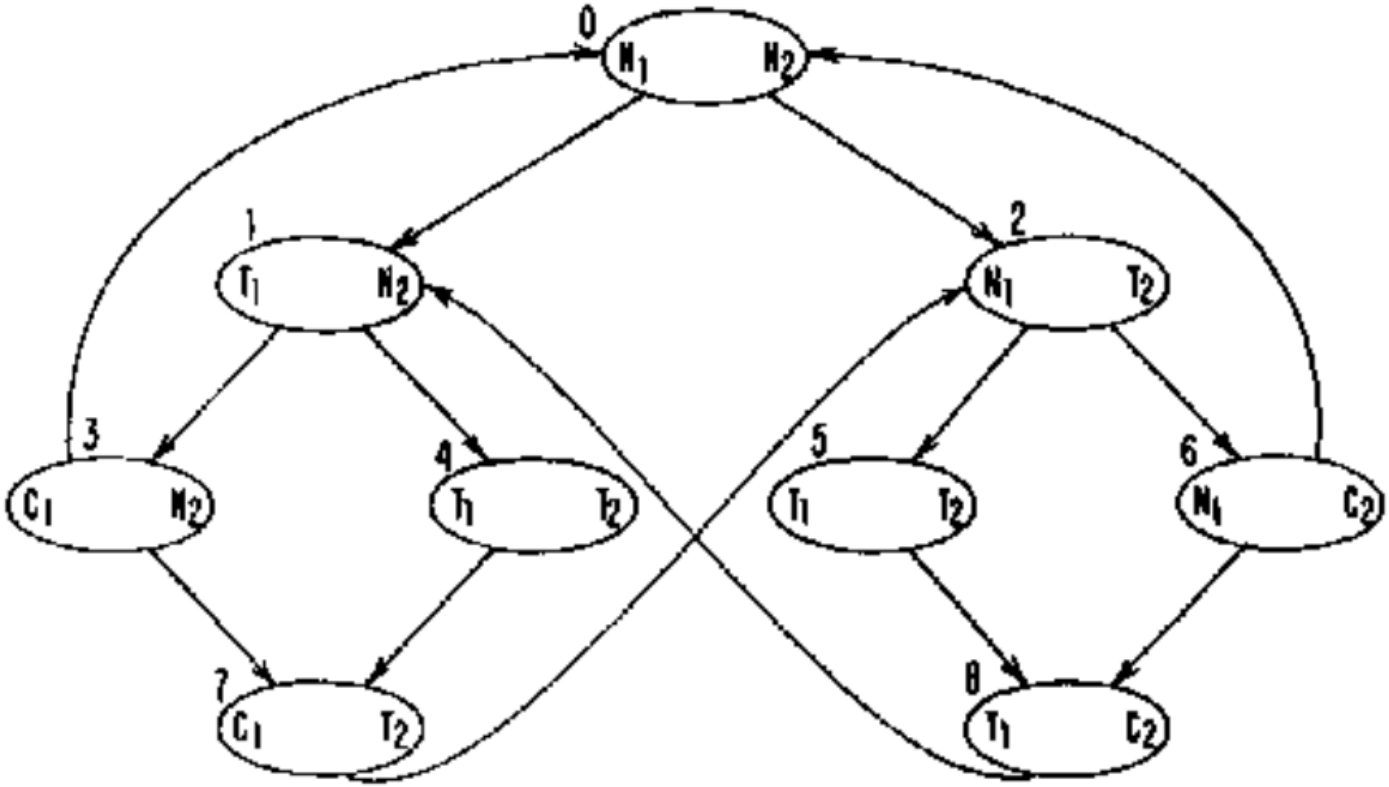


Struktura Kripkego - Przykład



Struktura Kripkego - Przykład





LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \longrightarrow$

Dodatkowo operatory temporalne (czasowe), czyli:

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$

Dodatkowo operatory temporalne (czasowe), czyli:

X (neXt), G (Globally), F (Finally), U (Until), R (Release)

$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$

Dodatkowo operatory temporalne (czasowe), czyli:

X (neXt), G (Globally), F (Finally), U (Until), R (Release)

$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$

Dodatkowo operatory temporalne (czasowe), czyli:

X (neXt), G (Globally), F (Finally), U (Until), R (Release)

$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

XG ϕ

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$

Dodatkowo operatory temporalne (czasowe), czyli:

X (neXt), G (Globally), F (Finally), U (Until), R (Release)

$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

XG ϕ - od jutra lubię krakersy, już na zawsze

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:
atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$
Dodatkowo operatory temporalne (czasowe), czyli:
X (neXt), G (Globally), F (Finally), U (Until), R (Release)
$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

XG ϕ - od jutra lubię krakersy, już na zawsze
F($\phi \rightarrow$ X $\neg\phi$)

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:
atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$
Dodatkowo operatory temporalne (czasowe), czyli:
X (neXt), G (Globally), F (Finally), U (Until), R (Release)
$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

XG ϕ - od jutra lubię krakersy, już na zawsze
F($\phi \rightarrow$ X $\neg\phi$) - kiedyś będzie tak, że jak jednego dnia lubię krakersy to następnego już nie

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:
atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$
Dodatkowo operatory temporalne (czasowe), czyli:
X (neXt), G (Globally), F (Finally), U (Until), R (Release)
$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

XG ϕ - od jutra lubię krakersy, już na zawsze
F($\phi \rightarrow$ X $\neg\phi$) - kiedyś będzie tak, że jak jednego dnia lubię krakersy to następnego już nie
$\mathrm{G}((\phi \rightarrow \mathrm{X}\ \neg\phi) \wedge (\neg\phi \rightarrow \mathrm{X}\ \phi))$

LTL - Linear temporal logic

Operatory

Jak dla zwykłej logiki, czyli:
atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$
Dodatkowo operatory temporalne (czasowe), czyli:
X (neXt), G (Globally), F (Finally), U (Until), R (Release)
$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{X}\ \phi \mid \mathrm{G}\ \phi \mid \mathrm{F}\ \phi \mid \phi_1 \mathrm{U} \phi_2 \mid \phi_1 \mathrm{R} \phi_2$

ϕ znaczy lubię krakersy

XG ϕ - od jutra lubię krakersy, już na zawsze
F($\phi \rightarrow$ X $\neg\phi$) - kiedyś będzie tak, że jak jednego dnia lubię krakersy to następnego już nie
G$((\phi \rightarrow \mathrm{X}\ \neg\phi) \wedge (\neg\phi \rightarrow \mathrm{X}\ \phi))$ - zawsze jak dziś lubię krakersy to jutro nie, a jak dziś nie lubię to jutro już tak

LTL - operatory temporalne (czasowe) unarne

(neXt - w następnym kroku) X ϕ

(Globally - w każdym kroku) G ϕ

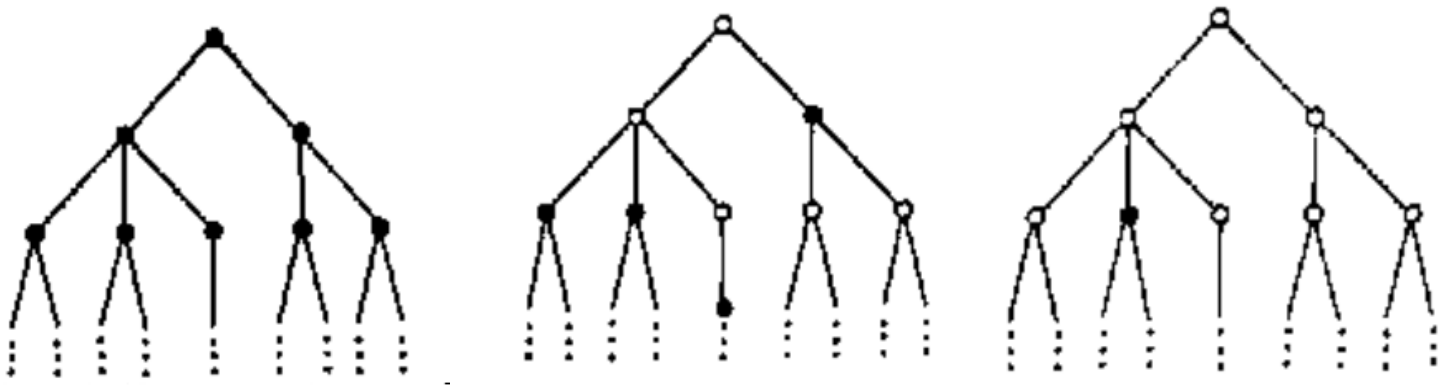
(Finally - w końcu kiedyś) F ϕ

LTL - operatory temporalne (czasowe) binarne

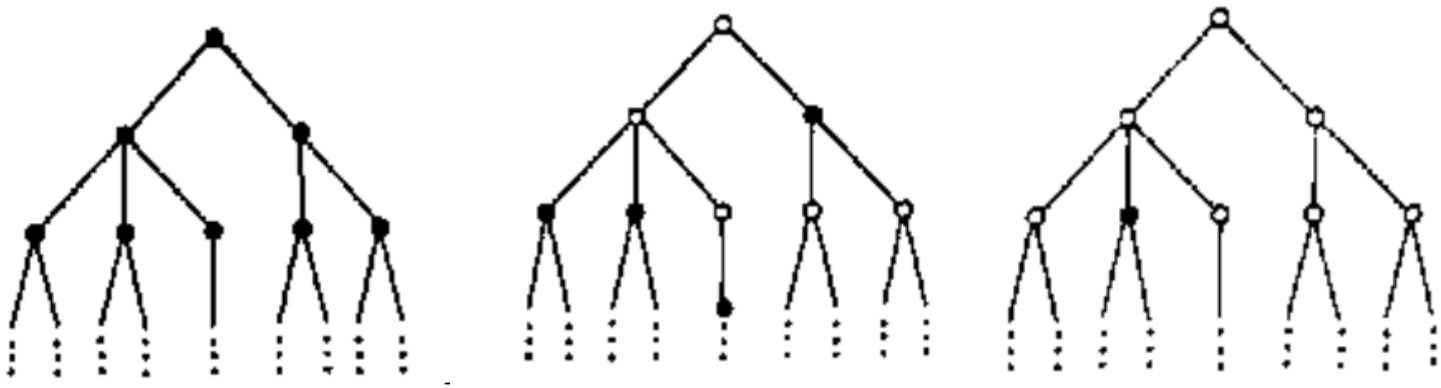
(Until - ψ jest prawdziwe co najmniej do kiedy w następnym kroku będzie ϕ) ψ U ϕ

(Release - ϕ jest prawdziwe co najmniej do czasu aż ψ i ϕ będą razem prawdziwe) ψ R ϕ

LTL - przykład



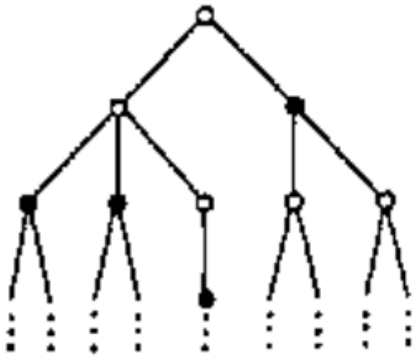
LTL - przykład



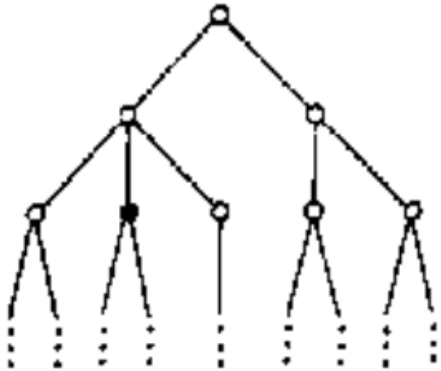
bezpieczeństwo

LTL - przykład

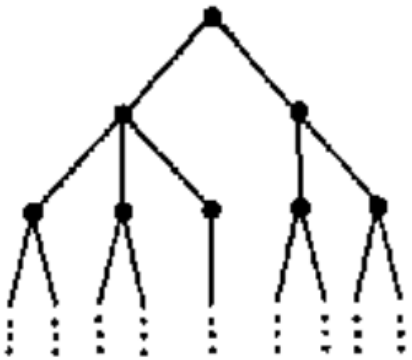
bezpieczeństwo



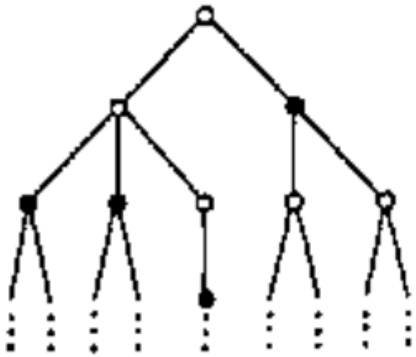
żywotność



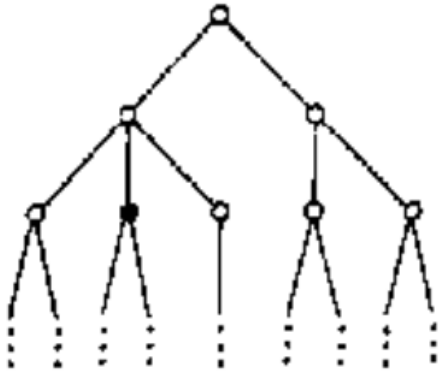
LTL - przykład



bezpieczeństwo

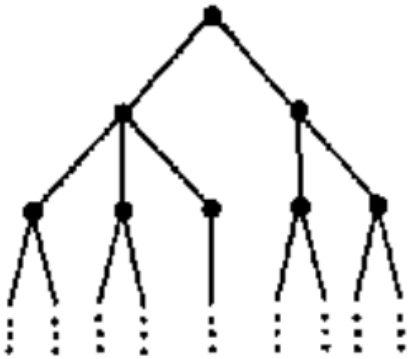


żywotność

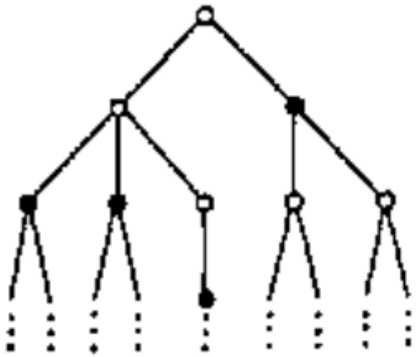


możliwość

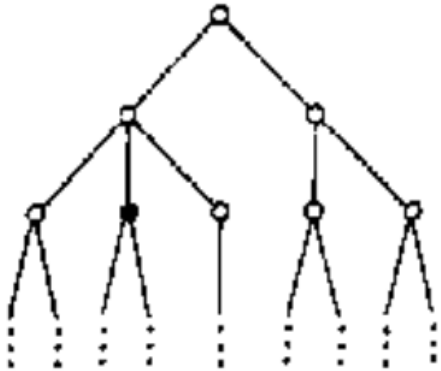
LTL - przykład



bezpieczeństwo
$G\phi$

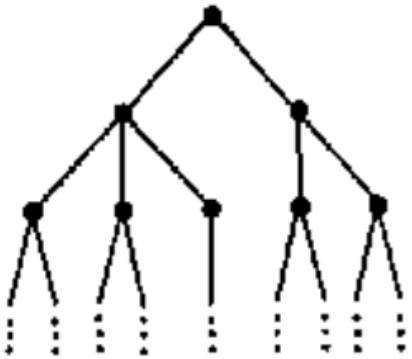


żywotność

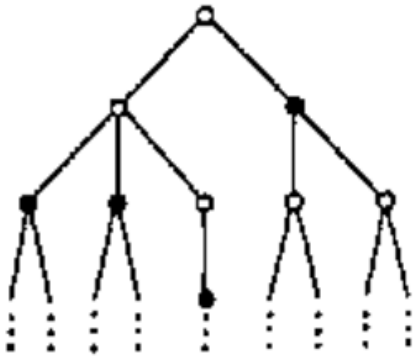


możliwość

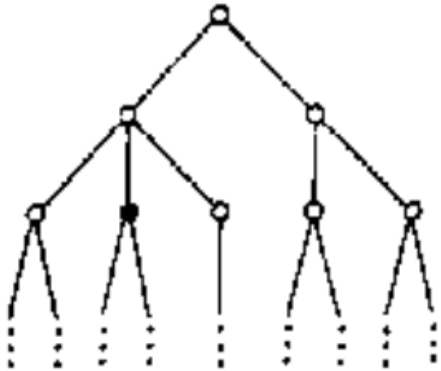
LTL - przykład



bezpieczeństwo
$G\phi$

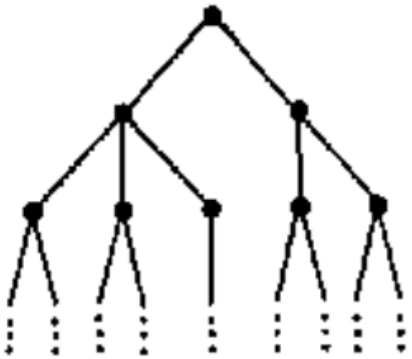


żywotność
$F\phi$

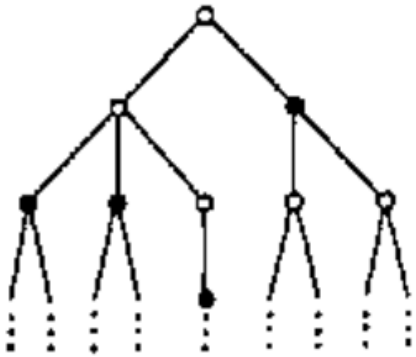


możliwość

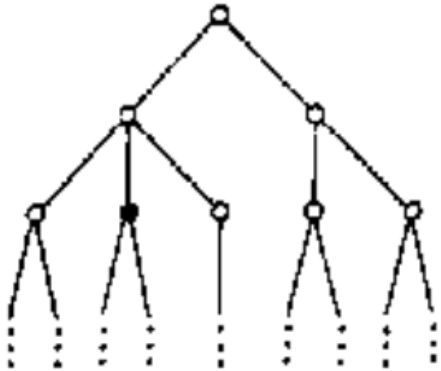
LTL - przykład



bezpieczeństwo
$G\phi$



żywotność
$F\phi$



możliwość
$\neg G \neg \phi$

LTL - definicja

Def.: Scieżka (przebieg) to maksymalny ciag

$$\pi = s_0 \longrightarrow s_1 \longrightarrow s_2 .$$

LTL wyraża własności ścieżek. Na strukturze Kripkego κ formułę $\phi \in$ LTL interpretujemy następujaco:
dla każdej ścieżki takiej, ze $s_0 \in \Omega_p$ (st. początkowe) , zachodzi ϕ

LTL - bezpieczeństwo (safety)

Sekcja krytyczna

$G\ \neg(critsec_1 \wedge critfsec_2)$

Przepełnienie stosu

$G\ \neg overflow$

W (Week until - słabe until)

$\phi_1\ W\ \phi_1 \qquad G\phi_1 \vee (\phi_1\ U\ \phi_2)$

LTL - żywotność (liveness)

Żądanie

$G\ (rec \rightarrow F\ sat)$

Bezpieczeństwo + żywotność

$\phi_1\ U\ \phi_2$

LTL - przykład

– nieskończenie wiele razy ϕ
– prawie zawsze ϕ
– jeśli ϕ (req) to w przyszłości ψ (granted)

LTL - przykład

– nieskończenie wiele razy ϕ $\quad$ $\mathsf{GF}\phi$
– prawie zawsze ϕ
– jeśli ϕ (req) to w przyszłości ψ (granted)

LTL - przykład

- nieskończenie wiele razy ϕ $\quad$ $GF\phi$
- prawie zawsze ϕ $\quad$ $FG\phi$
- jeśli ϕ (req) to w przyszłości ψ (granted)

LTL - przykład

- nieskończenie wiele razy ϕ $\quad$ $\mathsf{GF}\phi$
- prawie zawsze ϕ $\quad$ $\mathsf{FG}\phi$
- jeśli ϕ (req) to w przyszłości ψ (granted) $\quad$ $\mathsf{G}(\phi \rightarrow \mathsf{XF}\psi)$

LTL - przykład

- nieskończenie wiele razy ϕ $\quad$ $GF\phi$
- prawie zawsze ϕ $\quad$ $FG\phi$
- jeśli ϕ (req) to w przyszłości ψ (granted) $\quad$ $G(\phi \rightarrow XF\psi)$
- sprawiedliwość: jeśli "uparcie" ϕ (req) to ψ (granted)

LTL - przykład

– nieskończenie wiele razy ϕ $\quad$ $\mathsf{GF}\phi$
– prawie zawsze ϕ $\quad$ $\mathsf{FG}\phi$
– jeśli ϕ (req) to w przyszłości ψ (granted) $\quad$ $\mathsf{G}(\phi \rightarrow \mathsf{XF}\psi)$

– sprawiedliwość: jeśli "uparcie" ϕ (req) to ψ (granted)
„słaba”: uparcie = prawie zawsze
„silna”: uparcie = nieskończenie wiele

LTL - przykład

– nieskończenie wiele razy ϕ $\quad$ $GF\phi$
– prawie zawsze ϕ $\quad$ $FG\phi$
– jeśli ϕ (req) to w przyszłości ψ (granted) $\quad$ $G(\phi \rightarrow XF\psi)$

– sprawiedliwość: jeśli "uparcie" ϕ (req) to ψ (granted)
„słaba”: uparcie = prawie zawsze $\quad$ $FG\phi \rightarrow F\ \psi$
„silna”: uparcie = nieskończenie wiele

LTL - przykład

– nieskończenie wiele razy ϕ $\quad$ $GF\phi$
– prawie zawsze ϕ $\quad$ $FG\phi$
– jeśli ϕ (req) to w przyszłości ψ (granted) $\quad$ $G(\phi \rightarrow XF\psi)$

– sprawiedliwość: jeśli "uparcie" ϕ (req) to ψ (granted)
„słaba”: uparcie = prawie zawsze $\quad$ $FG\phi \rightarrow F\ \psi$
„silna”: uparcie = nieskończenie wiele $\quad$ $GF\phi \rightarrow F\ \psi$

LTL - prawa de Morgane'a

Prawa de Morgane'a

$$\phi_1 \vee \phi_2 \equiv \neg(\neg\phi_1 \wedge \neg\phi_2)$$

$$\mathsf{X}\ \phi \equiv \neg\ \mathsf{X}\ \neg\phi$$

$$\mathsf{G}\ \phi \equiv \neg\ \mathsf{F}\ \neg\phi$$

$$\phi_1 R \phi_2 \equiv \neg(\neg\phi_1 U \neg\phi_2)$$

LTL - wyciąganie negacji

Wyciąganie negacji

$$\neg(\phi_1 \wedge \phi_2) \equiv \neg\phi_1 \vee \neg\phi_2$$

$$\neg G\phi \equiv F\neg\phi$$

$$\neg F\phi \equiv G\neg\phi$$

$$\neg X\phi \equiv X\neg\phi$$

$$\neg(\phi_1 U \phi_2) \equiv \neg\phi_1 R \neg\phi_2$$

LTL - granice ekspresji

Jak zdefiniować:

- na każdej parzystej pozycji jest ϕ

- na każdej parzystej pozycji jest ϕ na każdej nieparzystej jest $\neg\phi$

- z każdego stanu osiągalnego można wrócić do stanu początkowego

- na każdej ścieżce osiągniemy stan taki, że w każdym bezpośrednio następnym stanie zachodzi ϕ

LTL - granice ekspresji

Jak zdefiniować:

- na każdej parzystej pozycji jest ϕ ???

- na każdej parzystej pozycji jest ϕ na każdej nieparzystej jest $\neg\phi$

- z każdego stanu osiągalnego można wrócić do stanu początkowego

- na każdej ścieżce osiągniemy stan taki, że w każdym bezpośrednio następnym stanie zachodzi ϕ

LTL - granice ekspresji

Jak zdefiniować:

- na każdej parzystej pozycji jest ϕ ???

- na każdej parzystej pozycji jest ϕ na każdej nieparzystej jest $\neg\phi$
$G((\phi \rightarrow X\ \neg\phi) \wedge (\neg\phi \rightarrow X\ \phi))$

- z każdego stanu osiągalnego można wrócić do stanu początkowego

- na każdej ścieżce osiągniemy stan taki, że w każdym bezpośrednio następnym stanie zachodzi ϕ

LTL - granice ekspresji

Jak zdefiniować:

- na każdej parzystej pozycji jest ϕ ???

- na każdej parzystej pozycji jest ϕ na każdej nieparzystej jest $\neg\phi$ $G((\phi \rightarrow X\ \neg\phi) \wedge (\neg\phi \rightarrow X\ \phi))$

- z każdego stanu osiągalnego można wrócić do stanu początkowego ???

- na każdej ścieżce osiągniemy stan taki, że w każdym bezpośrednio następnym stanie zachodzi ϕ

LTL - granice ekspresji

Jak zdefiniować:

- na każdej parzystej pozycji jest ϕ ???

- na każdej parzystej pozycji jest ϕ na każdej nieparzystej jest $\neg\phi$ $G((\phi \rightarrow X\ \neg\phi) \wedge (\neg\phi \rightarrow X\ \phi))$

- z każdego stanu osiągalnego można wrócić do stanu początkowego ???

- na każdej ścieżce osiągniemy stan taki, że w każdym bezpośrednio następnym stanie zachodzi ϕ ???

CTL - Computation tree logic

Operatory

Jak dla CTL, czyli:

CTL - Computation tree logic

Operatory

Jak dla CTL, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$, temporalne - X, G, F, U, R

CTL - Computation tree logic

Operatory

Jak dla CTL, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$, temporalne - X, G, F, U, R

Dodatkowo kwantyfikatory ścieżkowe czyli:

CTL - Computation tree logic

Operatory

Jak dla CTL, czyli:

atomy - $p_1, p_2, .., p_n$, operatory logiki - $\neg, \vee, \wedge, \rightarrow$, temporalne - X, G, F, U, R

Dodatkowo kwantyfikatory ścieżkowe czyli:

A (All paths), E (Exist path)

$\phi := \mathrm{p} \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \mathrm{AX}\ \phi \mid \mathrm{EX}\ \phi \mid \mathrm{AG}\ \phi \mid \mathrm{EG}\ \phi$
$\mid \mathrm{AF}\ \phi \mid \mathrm{EF}\ \phi \mid \mathrm{A}[\phi_1 \mathrm{U} \phi_2] \mid \mathrm{E}[\phi_1 \mathrm{U} \phi_2]$

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)

E (Exist path)

(Exist - na co najmniej jednej ścieżce)

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ

E (Exist path)

(Exist - na co najmniej jednej ścieżce)

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)

E (Exist path)

(Exist - na co najmniej jednej ścieżce)

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)
AF ϕ

E (Exist path)

(Exist - na co najmniej jednej ścieżce)

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)
AF ϕ (na każdej ścieżce w pewnym momencie spełnione jest ϕ)

E (Exist path)

(Exist - na co najmniej jednej ścieżce)

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)
AF ϕ (na każdej ścieżce w pewnym momencie spełnione jest ϕ)

E (Exist path)

(Exist - na co najmniej jednej ścieżce)
EG ϕ

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)
AF ϕ (na każdej ścieżce w pewnym momencie spełnione jest ϕ)

E (Exist path)

(Exist - na co najmniej jednej ścieżce)
EG ϕ (na co najmniej jednej ścieżce zawsze jest spełnione ϕ)

CTL - kwantyfikatory ścieżkowe

A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)
AF ϕ (na każdej ścieżce w pewnym momencie spełnione jest ϕ)

E (Exist path)

(Exist - na co najmniej jednej ścieżce)
EG ϕ (na co najmniej jednej ścieżce zawsze jest spełnione ϕ)
EF ϕ

CTL - kwantyfikatory ścieżkowe

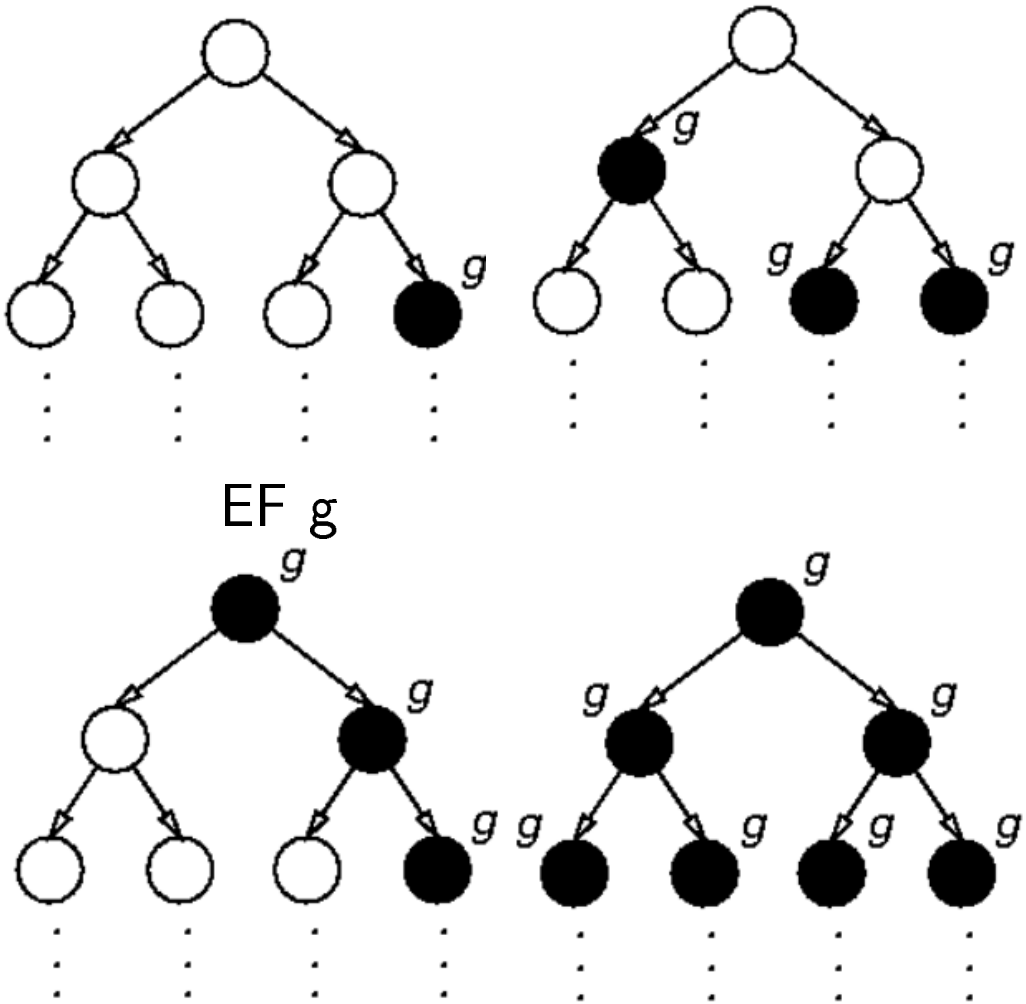
A (All paths)

(All - na każdej ścieżce)
AG ϕ (na każdej ścieżce zawsze spełnione jest ϕ)
AF ϕ (na każdej ścieżce w pewnym momencie spełnione jest ϕ)

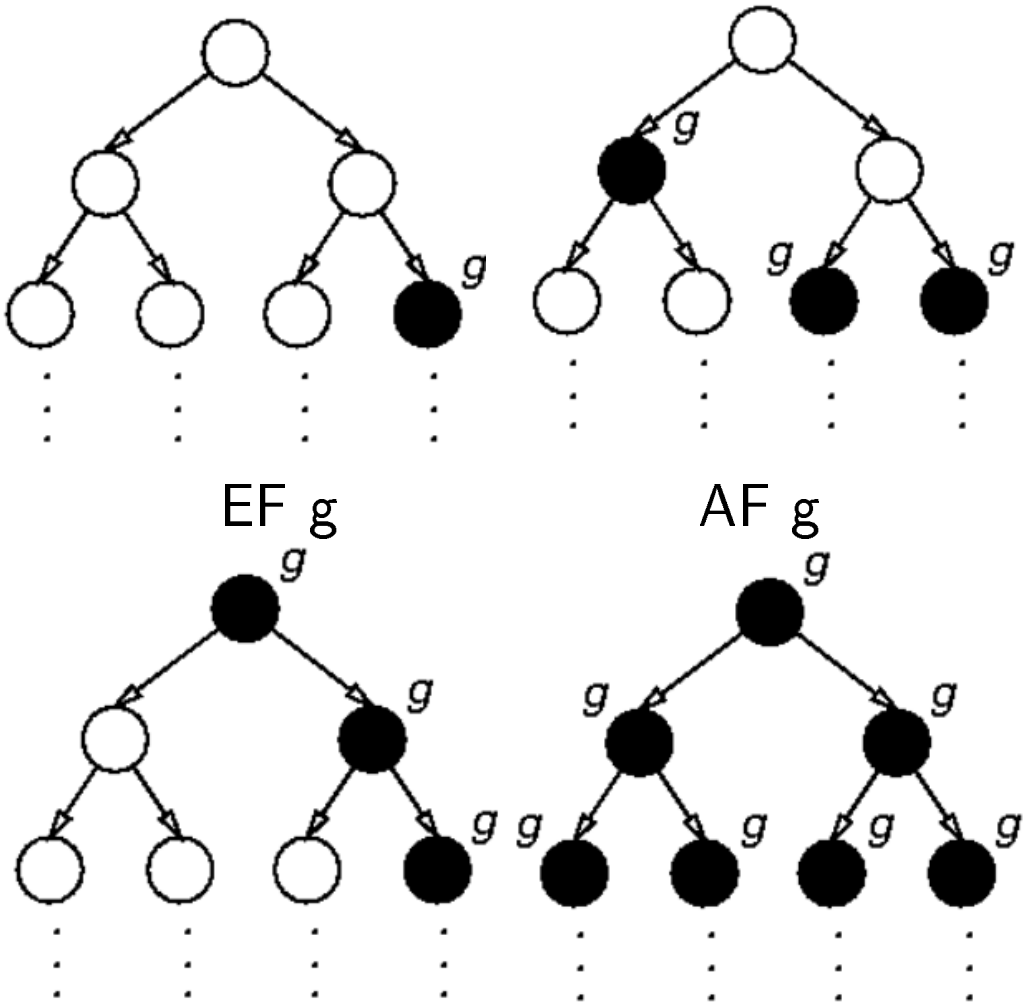
E (Exist path)

(Exist - na co najmniej jednej ścieżce)
EG ϕ (na co najmniej jednej ścieżce zawsze jest spełnione ϕ)
EF ϕ (na co najmniej jednej ścieżce w pewnym momencie spełnione jest ϕ)

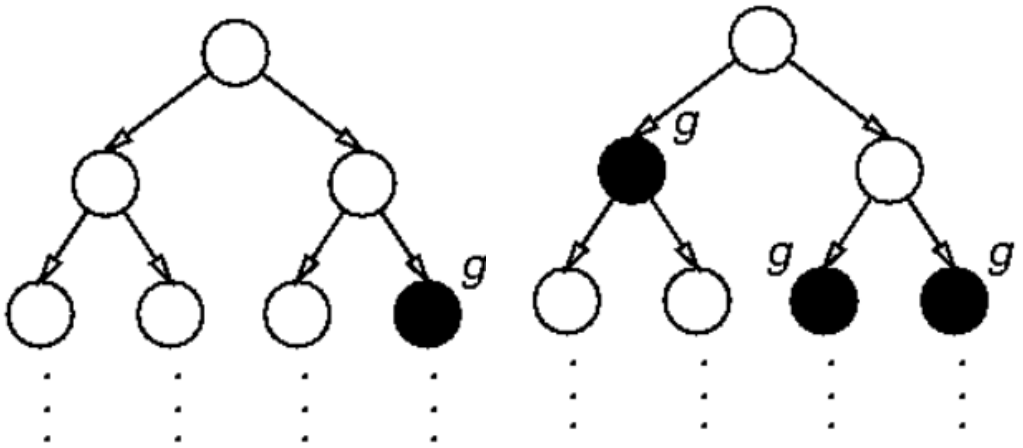
CTL - przykład



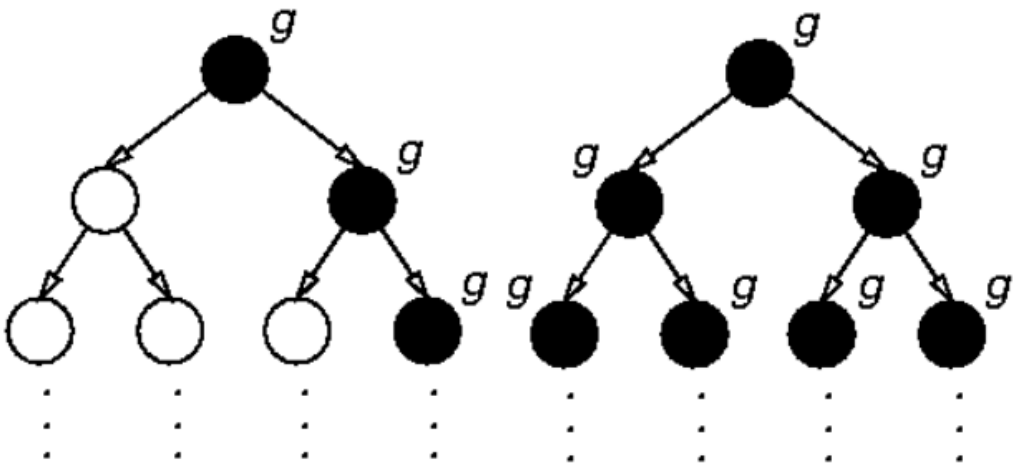
CTL - przykład



CTL - przykład

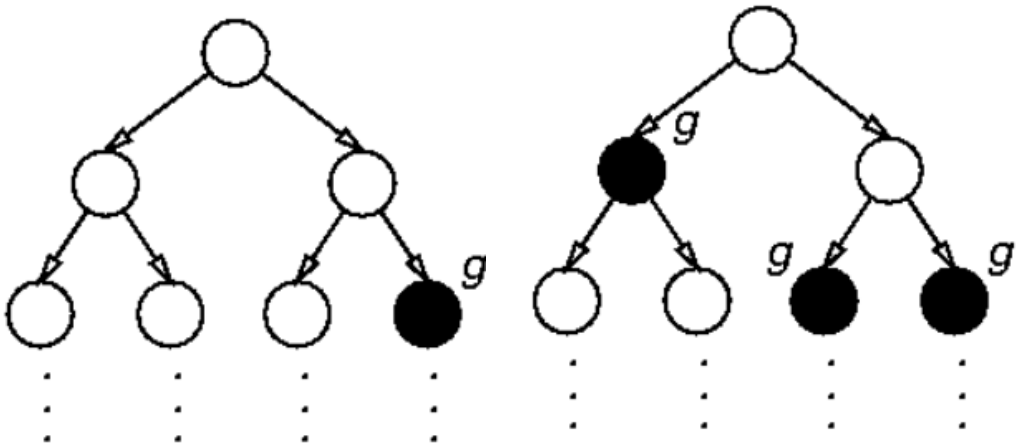


EF g AF g

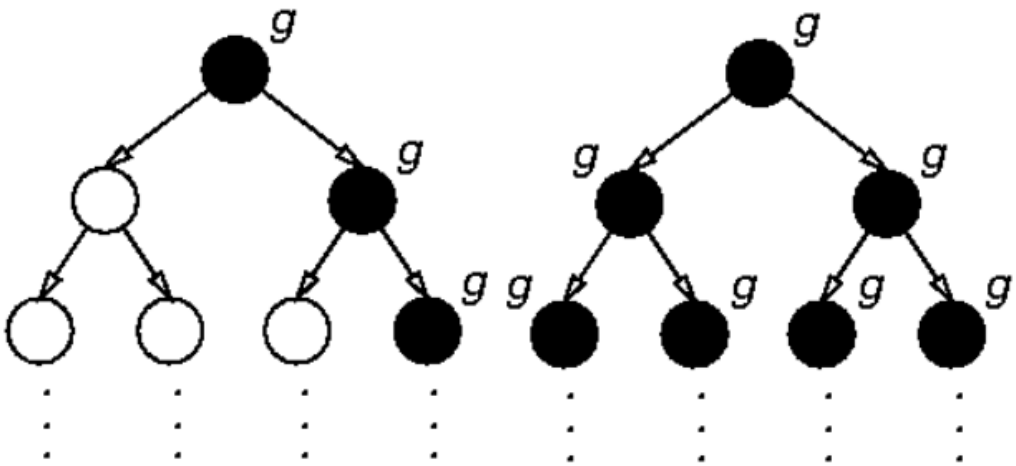


EG g

CTL - przykład



EF g AF g



EG g AG g

CTL - przykład

ϕ znaczy lubię krakersy

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co
AF ϕ

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

EF ϕ

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

EF ϕ - może będę lubił kiedyś krakersy przez co najmniej jeden dzień

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

EF ϕ - może będę lubił kiedyś krakersy przez co najmniej jeden dzień

AF EG ϕ

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

EF ϕ - może będę lubił kiedyś krakersy przez co najmniej jeden dzień

AF EG ϕ - zawsze jest możliwe (AF), że zacznę lubić krakersy już na stałe

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

EF ϕ - może będę lubił kiedyś krakersy przez co najmniej jeden dzień

AF EG ϕ - zawsze jest możliwe (AF), że zacznę lubić krakersy już na stałe

EG AF ϕ

CTL - przykład

ϕ znaczy lubię krakersy

AG ϕ - lubię krakersy od teraz choćby nie wiem co

AF ϕ - będę lubił krakersy w przyszłości przez co najmniej jeden dzień choćby nie wiem co

EF ϕ - może będę lubił kiedyś krakersy przez co najmniej jeden dzień

AF EG ϕ - zawsze jest możliwe (AF), że zacznę lubić krakersy już na stałe

EG AF ϕ - w zależności od tego co się teraz stanie (E), będzie już zawsze (G) tak, że będę kiedyś lubił krakersy przez co najmniej jeden dzień (AF), albo nic już nie będzie można powiedzieć, na pewno, na ten temat

CTL - bezpieczeństwo (safety)

Sekcja krytyczna

AG $\neg(critsec_1 \wedge critsec_2)$

Przepełnienie stosu

AG $\neg$overflow

CTL - żywotność (liveness)

Żądanie

AG (rec $\longrightarrow$ AF sat)

Powrót do stanu początkowego

AG EF init

CTL przykład

Osiągalność

Bezpieczeństwo

Żywotność

Brak blokady

CTL przykład

Osiągalność

EF ϕ

Bezpieczeństwo

Żywotność

Brak blokady

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

Żywotność

Brak blokady

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

AG ϕ

Żywotność

Brak blokady

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

AG ϕ (LTL - G ϕ)

Żywotność

Brak blokady

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

AG ϕ (LTL - G ϕ)

Żywotność

AG $(\phi \rightarrow \text{AF } \psi)$

Brak blokady

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

AG ϕ (LTL - G ϕ)

Żywotność

AG $(\phi \rightarrow$ AF $\psi)$ (LTL - G $(\phi \rightarrow$ F $\psi)$

Brak blokady

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

AG ϕ (LTL - G ϕ)

Żywotność

AG $(\phi \rightarrow$ AF $\psi)$ (LTL - G $(\phi \rightarrow$ F $\psi)$

Brak blokady

AF EX ϕ

CTL przykład

Osiągalność

EF ϕ (LTL - F ϕ "pewność" a nie osiągalność)

Bezpieczeństwo

AG ϕ (LTL - G ϕ)

Żywotność

AG ($\phi \rightarrow$ AF ψ) (LTL - G ($\phi \rightarrow$ F ψ)

Brak blokady

AF EX ϕ (LTL - F X ϕ)

Czas w CTL i LTL

W LTL czas był liniowy

W CTL czas jest rozgałęziony

CTL vs. LTL

LTL

G ϕ, F ϕ

G F ϕ

G ($\phi \rightarrow$ F ψ)

-

-

F G $\phi \rightarrow$ G F ψ

G F $\phi \rightarrow$ G F ψ

CTL

CTL vs. LTL

LTL

G ϕ, F ϕ

G F ϕ

G ($\phi \rightarrow$ F ψ)

-

-

F G $\phi \rightarrow$ G F ψ

G F $\phi \rightarrow$ G F ψ

CTL

AG ϕ, AF ϕ

CTL vs. LTL

LTL

G ϕ, F ϕ

G F ϕ

G ($\phi \rightarrow$ F ψ)

-

-

F G $\phi \rightarrow$ G F ψ

G F $\phi \rightarrow$ G F ψ

CTL

AG ϕ, AF ϕ

AG AFϕ

CTL vs. LTL

LTL

G ϕ, F ϕ

G F ϕ

G ($\phi \rightarrow$ F ψ)

-

-

F G $\phi \rightarrow$ G F ψ

G F $\phi \rightarrow$ G F ψ

CTL

AG ϕ, AF ϕ

AG AFϕ

AG ($\phi \rightarrow$ AF ψ)

CTL vs. LTL

LTL

G ϕ, F ϕ

G F ϕ

G ($\phi \rightarrow$ F ψ)

-

-

F G $\phi \rightarrow$ G F ψ

G F $\phi \rightarrow$ G F ψ

CTL

AG ϕ, AF ϕ

AG AFϕ

AG ($\phi \rightarrow$ AF ψ)

EF ϕ,EG ϕ

CTL vs. LTL

LTL	CTL
G ϕ, F ϕ	AG ϕ, AF ϕ
G F ϕ	AG AFϕ
G ($\phi \rightarrow$ F ψ)	AG ($\phi \rightarrow$ AF ψ)
-	EF ϕ,EG ϕ
-	AG EF ϕ
F G $\phi \rightarrow$ G F ψ	
G F $\phi \rightarrow$ G F ψ	

CTL vs. LTL

LTL	CTL
G ϕ, F ϕ	AG ϕ, AF ϕ
G F ϕ	AG AFϕ
G ($\phi \rightarrow$ F ψ)	AG ($\phi \rightarrow$ AF ψ)
-	EF ϕ,EG ϕ
-	AG EF ϕ
F G $\phi \rightarrow$ G F ψ	-
G F $\phi \rightarrow$ G F ψ	

CTL vs. LTL

LTL	CTL
G ϕ, F ϕ	AG ϕ, AF ϕ
G F ϕ	AG AFϕ
G ($\phi \rightarrow$ F ψ)	AG ($\phi \rightarrow$ AF ψ)
-	EF ϕ,EG ϕ
-	AG EF ϕ
F G $\phi \rightarrow$ G F ψ	-
G F $\phi \rightarrow$ G F ψ	-

CTL vs. LTL

Złożoność algorytmów sprawdzających

CTL - Linowa względem przestrzeni stanów, liniowa względem rozmiaru formuły
LTL - Linowa względem przestrzeni stanów, ekspotencjalna względem rozmiaru formuły

CTL vs. LTL morał

LTL

Ograniczenie:
Logika tylko ścieżkowa, czyli nie ma E ani A

CTL

Ograniczenie:
Kwantyfikatory ścieżkowe i operatory temporalne parami

Formuła

Formuła w LTL może być wykładniczo bardziej zwięzła niż w CTL

Nagroda Turinga

2007 rok

Edmund M. Clarke otrzymał nagrodę Turinga w 2007 roku za: "For his role in developing Model-Checking into a highly effective verification technology, widely adopted in the hardware and software industries."

Dziękuję

Dziękuję

Materiały

http://zls.mimuw.edu.pl/ sl/teaching/PMW/SLAJDY/wyklad02.pdf
http://zls.mimuw.edu.pl/ sl/teaching/PMW/SLAJDY/wyklad06.pdf
http://students.mimuw.edu.pl/ rw209225/petri/LogikiTemporalne.ppt
http://en.wikipedia.org/wiki/Computation_tree_logic
http://en.wikipedia.org/wiki/Linear_temporal_logic
Książka:
Systems and Software Veryfication Model-Checking Techniques and Tools
Autorzy:
B.Berard, M.Bidoit, A.Finkel i inni.