CTL - Λογική Δένδρου Υπολογισμού

Στην ενότητα αυτή θα μελετηθούν τα εξής θέματα:

Διακλαδωμένες Χρονικές λογικές

CTL – σύνταξη και ερμηνεία

Έλεγχος μοντέλου για τη CTL

Σύγκριση των PLTL και CTL

Δικαιοσύνη

Γραμμική και διακλαδωμένη χρονική λογική

- Γραμμική χρονική λογική:

  οι ιδιότητες αναφέρονται σε (και ελέγχουν) ***όλες*** *τις δυνατές εκτελέσεις* ενός μοντέλου

  π.χ. η δομή Kripke M ικανοποιεί την ιδιότητα `x<15 U x=100` αν σε *κάθε* εκτέλεση της δομής ισχύει ότι x <15 μέχρις ότου το x πάρει την τιμή 100.

- Διακλαδωμένη χρονική λογική:

  οι ιδιότητες αναφέρονται και ελέγχουν ***τη δενδρική δομή*** του μοντέλου,

  π.χ. η δομή Kripke M ικανοποιεί την ιδιότητα `A (x < 15 U x=100)` αν σε κάθε εκτέλεση της δομής ισχύει ότι x <15 μέχρις ότου το x πάρει την τιμή 100, και,

  η δομή Kripke M ικανοποιεί την ιδιότητα `E (x < 15 U x=100)` αν υπάρχει εκτέλεση της δομής όπου ισχύει ότι x <15 μέχρις ότου το x πάρει την τιμή 100.

Γιατί διακλαδωμένη χρονική λογική;

- *Εκφραστικότητα*: μας δίνει τα μέσα να εκφράσουμε διαφορετική κατηγορία ιδιοτήτων. Για παράδειγμα τη δυνατότητα (και όχι την αναγκαιότητα) να συμβεί κάτι.
  - Ισχύει όμως και το αντίθετο: υπάρχουν ιδιότητες γραμμικών λογικών που δεν μπορούν να εκφραστούν σε διακλαδωμένη λογική.

- *Πολυπλοκότητα μοντελοελέγχου*: Διαφορετικοί αλγόριθμοι για αυτοματοποιημένη επαλήθευση με διαφορετική πολυπλοκότητα χρόνου και χώρου.

Διακλαδωμένες χρονικές λογικές

- Διακλαδωμένες χρονικές λογικής που έχουν προταθεί περιλαμβάνουν
  - Λογική Hennessy – Milner (HML)
  - Λογική Δένδρου Υπολογισμού (Computation Tree Logic – CTL)
  - Επεκταμένη Λογική Δένδρου Υπολογισμού – CTL*

    (συνδυάζει σε ένα πρότυπο τις CTL και PLTL)
  - μ-calculus χωρίς εναλλαγή
  - μ-calculus

Προτασιακή γραμμική χρονική λογική

- Η Προτασιακή Γραμμική Χρονική Λογική ορίζεται ως το μικρότερο σύνολο ιδιοτήτων που παράγονται από τους πιο κάτω κανόνες
  - κάθε ατομική πρόταση p είναι ιδιότητα
  - Αν οι Φ και Ψ είναι ιδιότητες, τότε και οι $\neg\Phi$ και $\Phi \vee \Psi$ είναι ιδιότητες
  - Αν η Φ είναι μια ιδιότητα, τότε και η **X** Φ (next) είναι ιδιότητα
  - Αν οι Φ και Ψ είναι ιδιότητες, τότε και η Φ **U** Ψ (until) είναι ιδιότητα

Πως μπορούμε να εκφράσουμε ότι σε κάθε δυνατή εκτέλεση είναι πάντα δυνατή η επιστροφή στην αρχική κατάσταση;

G F `start;`

Προτασιακή διακλαδωμένη χρονική λογική

- Επεκτείνουμε την PLTL με ποσοτικούς τελεστές μονοπατιών
  - **A** , όπου **A** φ σημαίνει ότι η ιδιότητα φ ικανοποιείται σε όλες τις εκτελέσεις (μονοπάτια) του μοντέλου
  - **E** , όπου **E** φ σημαίνει ότι υπάρχει εκτέλεση του μοντέλου που ικανοποιεί την ιδιότητα φ

- Ιδιότητες του τύπου **A** φ και **E** φ ονομάζονται *ιδιότητες κατάστασης* (state formulae).

- PLTL ιδιότητες ονομάζονται *ιδιότητες εκτέλεσης* (path formulae).

Πως μπορούμε να εκφράσουμε ότι σε κάθε δυνατή εκτέλεση είναι πάντα δυνατή η επιστροφή στην αρχική κατάσταση; **AG EF** `start`

CTL (Computation Tree Logic)

Η CTL ορίζεται ως το μικρότερο σύνολο ιδιοτήτων που παράγονται ως εξής:

$$\Phi, \Psi ::= p \mid \neg\Phi \mid \Phi \vee \Psi \mid \mathbf{A}\ \varphi \mid \mathbf{E}\ \varphi$$

$$\varphi ::= \mathbf{X}\ \Phi \mid \Phi\ \mathbf{U}\ \Psi \mid \mathrm{F}\ \Phi \mid \mathrm{G}\ \Phi$$

1. Ιδιότητες κατάστασης – Φ
   - κάθε ατομική πρόταση p είναι ιδιότητα *κατάστασης*
   - Αν οι Φ και Ψ είναι ιδιότητες *κατάστασης*, τότε και οι $\neg\Phi$ και $\Phi \vee \Psi$ είναι ιδιότητες *κατάστασης*
   - Αν η φ είναι μια ιδιότητα *εκτέλεσης*, τότε οι $\mathbf{A}\ \varphi$ και η $\mathbf{E}\ \varphi$ είναι ιδιότητες *κατάστασης*
2. Ιδιότητες εκτέλεσης – φ
   - Αν οι Φ και Ψ είναι ιδιότητες *κατάστασης*, τότε οι $\mathbf{X}\ \Phi$ και $\Phi\ \mathbf{U}\ \Psi$ είναι ιδιότητες *εκτέλεσης*

Παραγόμενοι τελεστές

$$\mathbf{F}\ \Phi \equiv \text{true}\ \mathbf{U}\ \Phi$$

$$\mathbf{G}\ \Phi \equiv \neg\, \mathbf{F} \neg\, \Phi$$

$$\mathbf{EF}\ \Phi \equiv \mathbf{E}(\text{true}\ \mathbf{U}\ \Phi) \qquad \text{δυνατόν } \Phi$$

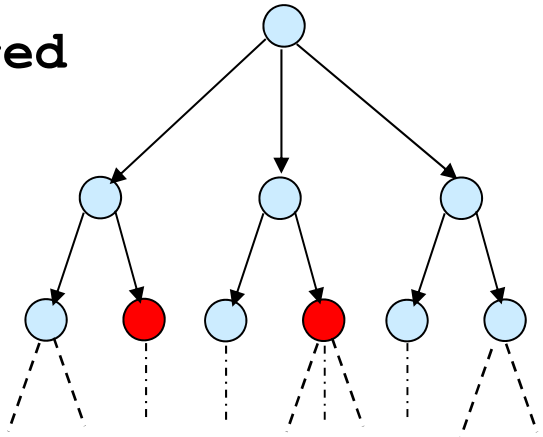
$$\mathbf{AG}\ \Phi \equiv \neg\, \mathbf{EF} \neg\, \Phi \qquad \text{σταθερά } \Phi$$

$$\mathbf{AF}\ \Phi \equiv \mathbf{A}(\text{true}\ \mathbf{U}\ \Phi) \qquad \text{αναπόφευκτα } \Phi$$

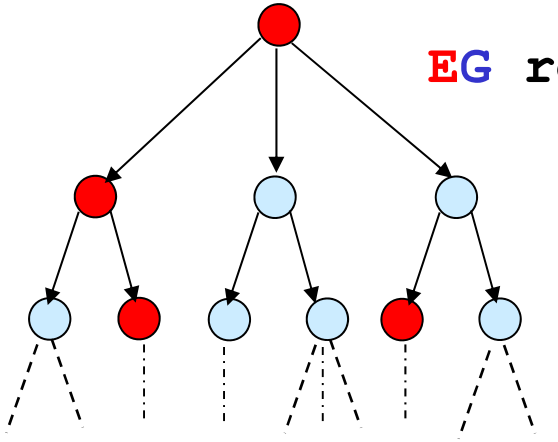
$$\mathbf{EG}\ \Phi \equiv \neg\, \mathbf{AF} \neg\, \Phi \qquad \text{δυνατόν πάντα } \Phi$$

Παράδειγμα

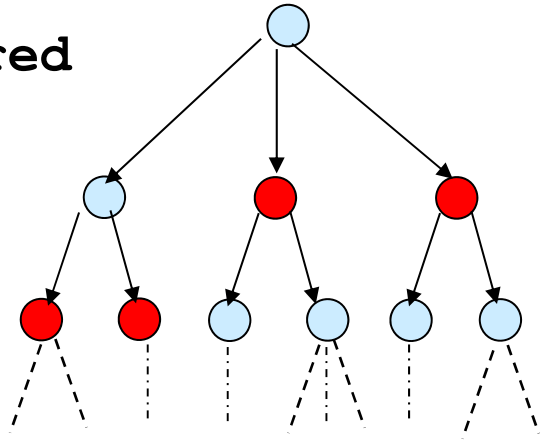
EF red



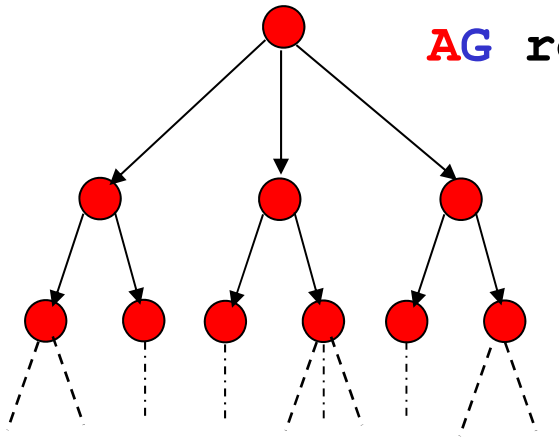
EG red



AF red



AG red



Παραδείγματα

- Έστω ΑΠ οι ατομικές προτάσεις που αφορούν τη μεταβλητή x, τους τελεστές <, ≥ , ≤ και =, και η συνάρτηση x+c

- Οι πιο κάτω ιδιότητες είναι νόμιμες CTL ιδιότητες
  - $\neg$ (x + 7 < 21) $\vee$ (x = 32)
  - **AF** (x + 12 < 30)
  - **EG** (x ≥ 0 $\wedge$ x < 20)
  - x = 10 $\rightarrow$ **AX E**(x ≥ 11 **U** x = 0)

- Οι πιο κάτω ιδιότητες δεν είναι νόμιμες
  - **E** (**F** x < 10 $\wedge$ **G** (x + 12 > y))
  - **E** (x < 21) $\vee$ **X** (x = 32)

Ερμηνεία της CTL

- Η τυπική ερμηνεία της CTL δίνεται (και πάλι) σε σχέση με τις δομές Kripke.
- Μια *δομή Kripke* ορίζεται ως μια πλειάδα M = (S, R, I, Label) όπου
  - S είναι ένα αριθμήσιμο σύνολο από καταστάσεις
  - $I \subseteq S$ είναι το σύνολο των αρχικών καταστάσεων
  - $R \subseteq S \times S$ είναι μία σχέση μεταβάσεων, όπου $(s, s') \in R$ αν υπάρχει μετάβαση από την κατάσταση s στην κατάσταση s'
  - Label : $S \rightarrow 2^{AP}$ είναι μια συνάρτηση η οποία συνδέει κάθε κατάσταση με τις ατομικές προτάσεις τις οποίες ικανοποιεί.
- Έστω μια κατάσταση $s \in S$. Τότε, Label(s) είναι το σύνολο των ατομικών προτάσεων που ισχύουν στην κατάσταση s.
- Ονομάζουμε μια ακολουθία από καταστάσεις $s_0\, s_1\, s_2 \ldots$ *μονοπάτι* αν s_0 είναι μια αρχική κατάσταση και $(s_i, s_{i+1}) \in R$ για κάθε $i \geq 0$.

Σημασιολογία της CTL: *ιδιότητες κατάστασης*

- Ορίζουμε τη σχέση |= όπου

> $M, s \models \Phi$ αν και μόνο αν η ιδιότητα Φ ικανοποιείται στην κατάσταση s της δομής M ως εξής:

$M, s \models p$	αν και μόνο αν	$p \in$ Label(s)
$M, s \models \neg\Phi$	αν και μόνο αν	δεν ισχύει ότι $M,s \models \Phi$
$M, s \models \Phi \vee \Psi$	αν και μόνο αν	$(M, s \models \Phi)$ ή $(M, s \models \Psi)$
$M, s \models \mathbf{E}\ \varphi$	αν και μόνο αν	$M,w \models \varphi$ για κάποιο μονοπάτι w που ξεκινά από την s
$M, s \models \mathbf{A}\ \varphi$	αν και μόνο αν	$M,w \models \varphi$ για κάθε μονοπάτι w που ξεκινά από την s

Σημασιολογία της CTL: *ιδιότητες εκτέλεσης*

- Έστω μονοπάτι $w = s_0 s_1 s_2 \ldots$ της δομής Kripke M και ιδιότητα φ. Ορίζουμε τη σχέση $\models$ όπου

> $M, w \models \varphi$ αν και μόνο αν η ιδιότητα φ ικανοποιείται στο μονοπάτι w της δομής M

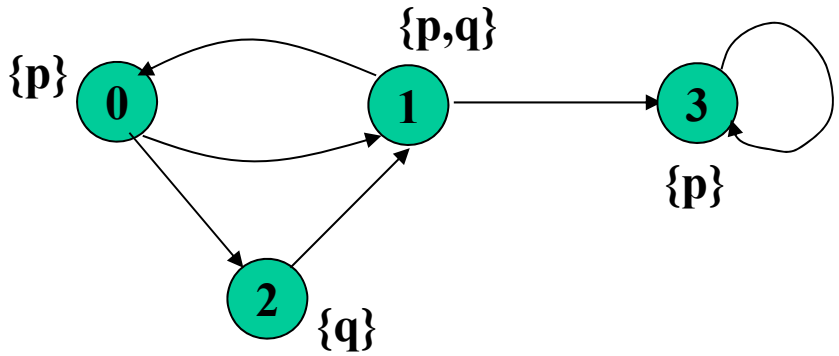
ως εξής:

$M, w \models \mathbf{X}\,\Phi$ αν και μόνο αν $M, w[1] \models \Phi$

$M, w \models \Phi\ \mathbf{U}\ \Psi$ αν και μόνο αν υπάρχει $j \geq 0$ τέτοιο ώστε $M, w[j] \models \Psi$ και για κάθε $0 \leq k < j$, $M, w[k] \models \Phi$

όπου αν $w = s_0 s_1 s_2 \ldots$, $w[k]$ είναι η κατάσταση s_k .

Παράδειγμα



Ποιες από τις πιο κάτω ιδιότητες ισχύουν;

M, 0 |= **E X** p √

M, 0 |= **A**(p **U** q) √

M, 0 |= **A X** p ×

M, 0 |= **A G** p ×

M, 0 |= **EF EG** p √

M, 0 |= **AX EG** p ×

Ισοδυναμίες της CTL

PLTL κανόνες ανάπτυξης

$$\Phi \; \mathbf{U} \; \Psi \equiv \Psi \vee (\Phi \wedge \mathbf{X}(\Phi \; \mathbf{U} \; \Psi))$$

$$\mathbf{F} \; \Phi \equiv \mathtt{true} \; \mathbf{U} \; \Phi$$

$$\mathbf{G} \; \Phi \equiv \mathtt{false} \; \mathbf{R} \; \Phi$$

CTL κανόνες ανάπτυξης

$$\mathbf{E}(\Phi \; \mathbf{U} \; \Psi) \equiv \Psi \vee (\Phi \wedge \mathbf{EX} \; \mathbf{E}(\Phi \mathbf{U} \Psi))$$

$$\mathbf{A}(\Phi \; \mathbf{U} \; \Psi) \equiv \Psi \vee (\Phi \wedge \mathbf{AX} \; \mathbf{A}(\Phi \mathbf{U} \Psi))$$

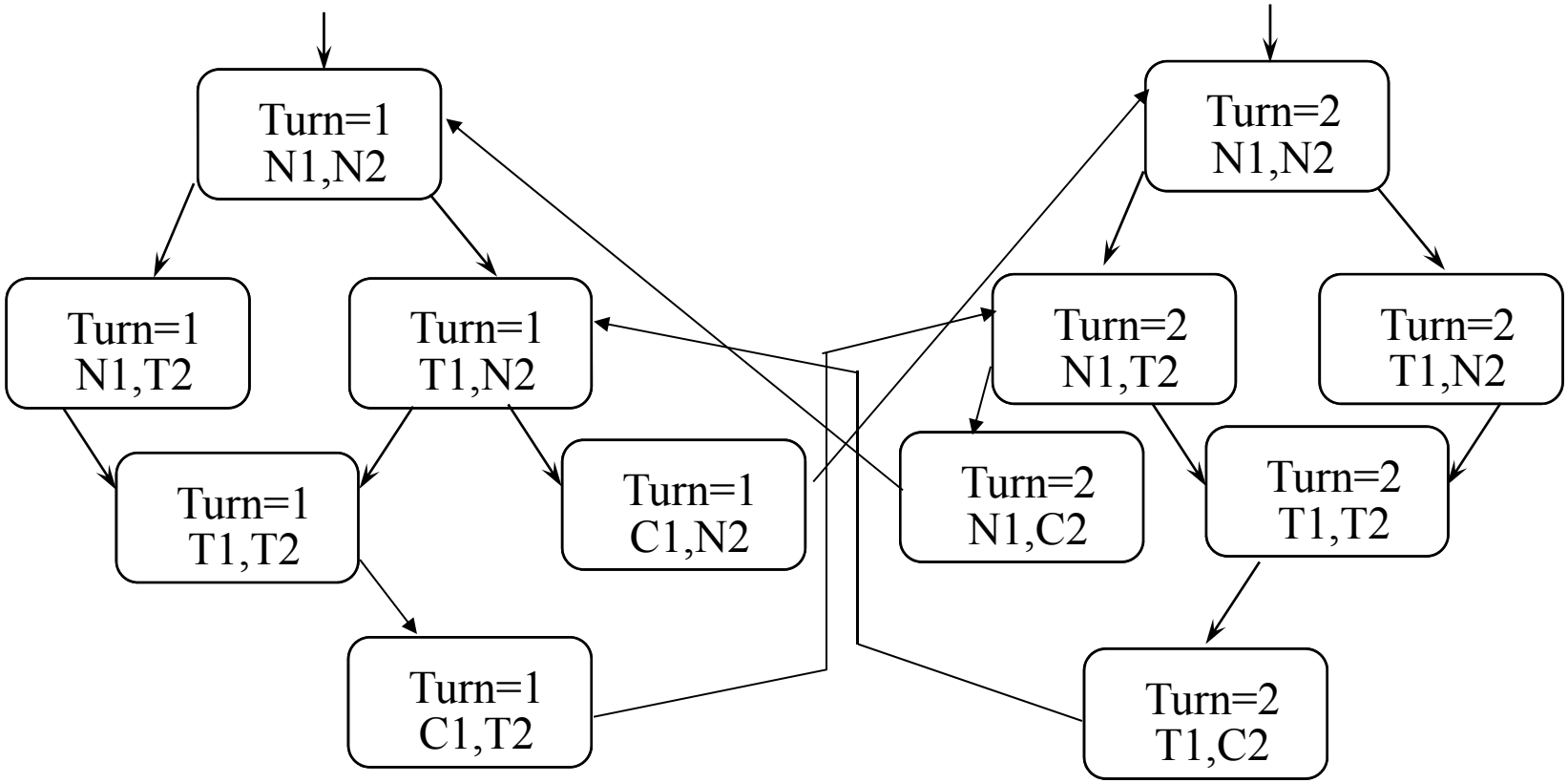
$$\mathbf{EF} \; \Phi \equiv \Phi \vee \mathbf{EX} \; \mathbf{EF} \; \Phi$$

$$\mathbf{AF} \; \Phi \equiv \Phi \vee \mathbf{AX} \; \mathbf{AF} \; \Phi$$

$$\mathbf{EG} \; \Phi \equiv \Phi \wedge \mathbf{EX} \; \mathbf{EG} \; \Phi$$

$$\mathbf{AG} \; \Phi \equiv \Phi \wedge \mathbf{AX} \; \mathbf{AG} \; \Phi$$

Αμοιβαίος αποκλεισμός



N_i : η διεργασία i είναι εκτός της κρίσιμής της περιοχής

T_i : η διεργασία i προσπαθεί να εισέλθει στην κρίσιμη περιοχή

C_i : η διεργασία i είναι εντός της κρίσιμής της περιοχής

Αμοιβαίος Αποκλεισμός

- Ασφάλεια: οι δύο διεργασίες δεν βρίσκονται ποτέ ταυτόχρονα στην κρίσιμη τους περιοχή

  AG(¬ (C1 ∧ C2))

- Ζωτικότητα: Κάθε φορά που η διεργασία 1 προσπαθεί να εισέλθει στην κρίσιμη της περιοχή θα το πράξει

  AG(T1 → **AF** C1)

- Έλλειψη εμποδίων: Κάθε φορά που η διεργασία 1 βρίσκεται στη μη κρίσιμη της περιοχή, είναι δυνατόν, στην επόμενη χρονική στιγμή να προσπαθήσει να εισέλθει στην κρίσιμη της περιοχή

  AG(N1 → **EX** T1)

- Έλλειψη αυστηρής διάταξης: Είναι δυνατόν, η διεργασία 1 να εισέλθει στην κρίσιμη της περιοχή δύο συνεχόμενες φορές χωρίς να παρεμβληθεί η διεργασία 2.

  EF(C1 ∧ **E**(C1 **U** (¬ C1 ∧ **E**(¬ C2 **U** C1))))

Ιδιότητες στη CTL

- Δυνατότητα προσέγγισης κατάστασης Φ
  - απλή — **EF** Φ
  - εξαρτώμενη — **E**(Ψ **U** Φ)
  - από κάθε κατάσταση — **AG** (**EF** Φ)

- Ασφάλεια (κάτι κακό δεν συμβαίνει ποτέ)
  - απλή — **AG** ¬bad
  - Εξαρτώμενη — **A**(¬bad **U** end)

- Ζωτικότητα (Liveness) — **AG** ($\Phi \rightarrow$ **AF** Ψ)

- Δικαιοσύνη — **AG** (**EF** Φ)

Έλεγχος Μοντελου της CTL

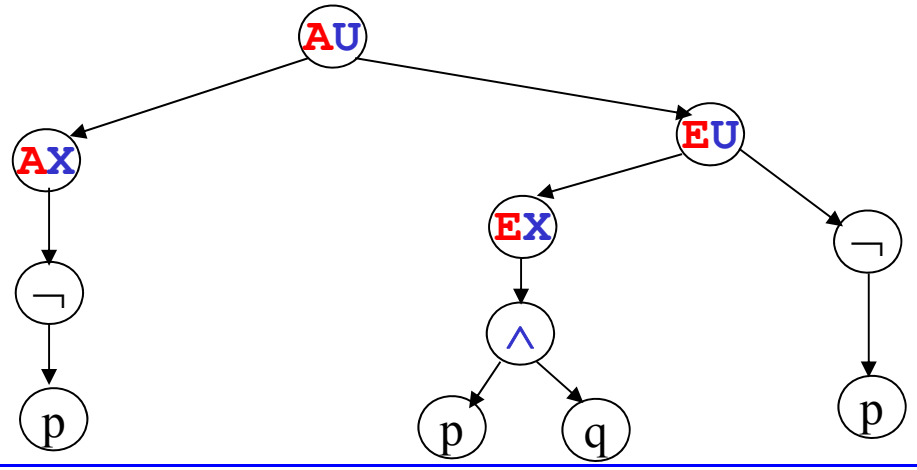
- Πως μπορούμε να ελέγξουμε κατά πόσο μια κατάσταση s ικανοποιεί μια CTL ιδιότητα Φ;
  - Υπολογίζουμε αναδρομικά το σύνολο *Sat*(Φ) των καταστάσεων που ικανοποιούν την Φ.
  - Ελέγχουμε αν η s ανήκει στο *Sat*(Φ).

- Αναδρομικός υπολογισμός
  - προσδιόρισε όλες τις υποιδιότητες της Φ
  - υπολόγισε το σύνολο *Sat*(p) για όλες τις ατομικές προτάσεις p της Φ
  - συνέχισε με τις μικρότερες υποιδιότητες που περιέχουν ατομικές προτάσεις
  - έλεγξε τις υποιδιότητες που περιέχουν αυτές τις ιδιότητες…
  - και ούτω καθεξής μέχρι να φτάσεις στην Φ.

Αλγόριθμος Μοντελο-ελέγχου

- Αναδρομικός υπολογισμός από κάτω προς τα πάνω
  - θεώρησε το δένδρο που αντιστοιχεί στην ιδιότητα Φ
  - υπολόγισε το σύνολο *Sat*(p) για τις ιδιότητες που βρίσκονται στα φύλλα του δένδρου της Φ
  - συνέχισε με τις υποιδιότητες που βρίσκονται σε ύψος 1 στο δένδρο της Φ,
  - στις υποιδιότητες που βρίσκονται σε ύψος 2…
  - και ούτω καθεξής μέχρι να φτάσεις στη ρίζα του δένδρου, δηλαδή στην Φ.

- Το δένδρο που αντιστοιχεί στην ιδιότητα **A**[**AX** ¬p **U** **E**[**EX**(p ∧ q) **U** ¬p]] είναι το



Επαρκή Σύνολα Τελεστών

- Ο αλγόριθμος βασίζεται στο γεγονός ότι το σύνολο των τελεστών $\neg$, $\vee$, T (true), EX, EU, AF είναι επαρκές για τη CTL. Δηλαδή όλοι οι υπόλοιποι τελεστές μπορούν να διατυπωθούν βάσει αυτών:
  - $AX\ \Phi = \neg\ EX\ \neg\Phi$
  - $A(\Phi_1 U \Phi_2) = \neg\ (E[\neg\Phi_2 U(\neg\Phi_1 \wedge \neg\ \Phi_2)] \vee\ EG\ \neg\Phi_2)$
  - $EF\ \Phi = E(T\ U\ \Phi)$
  - $EG\ \Phi = \neg AF\ \neg\ \Phi$
  - $AG\ \Phi_1 = \neg EF\ \neg\Phi_1$

Αναδρομική Διαδικασία (1)

```
SAT(Φ){
Case
   Φ = T                 return S
   Φ = ⊥                 return ∅
   Φ = p                 return {s ∈ S | p ∈ Label(s)}
   Φ = ¬Φ1               return   S - SAT (Φ1)
   Φ = Φ1∨Φ2             return SAT(Φ1) ∪ SAT(Φ2)
   Φ = Φ1∧Φ2             return SAT(Φ1) ∩ SAT(Φ2)
   Φ = Φ1→Φ2             return SAT(¬Φ1 ∨ Φ2)
   Φ = AX Φ1             return SAT(¬EX ¬Φ1)
   …
```

Αναδρομική Διαδικασία (2)

…

$\Phi = \mathbf{EX}\ \Phi_1$ return $\mathbf{SAT_{EX}}(\Phi_1)$

$\Phi = \mathbf{A}(\Phi_1 \mathbf{U} \Phi_2)$ return $\mathrm{SAT}(\neg(\mathbf{E}[\neg\Phi_2 \mathbf{U} (\neg\Phi_1 \wedge \neg\ \Phi_2)] \vee\ \mathbf{EG}\neg\Phi_2))$

$\Phi = \mathbf{E}(\Phi_1\ \mathbf{U}\ \Phi_2)$ return $\mathbf{SAT_{EU}}(\Phi_1, \Phi_2)$

$\Phi = \mathbf{EF}\ \Phi_1$ return $\mathrm{SAT}(\mathbf{E}(\mathrm{T}\ \mathbf{U}\ \Phi_1))$

$\Phi = \mathbf{EG}\ \Phi_1$ return $\mathrm{SAT}(\neg\mathbf{AF}\ \neg\Phi_1)$

$\Phi = \mathbf{AF}\ \Phi_1$ return $\mathbf{SAT_{AF}}(\Phi_1)$

$\Phi = \mathbf{AG}\ \Phi_1$ return $\mathrm{SAT}(\neg\mathbf{EF}\ \neg\Phi_1)$

}

Η διαδικασία $SAT_{EX}(\Phi)$

- Υπολογίζει τις καταστάσεις που ικανοποιούν την Φ (SAT(Φ)) και μετά οπισθοδρομεί για να υπολογίσει το σύνολο των καταστάσεων που μπορούν να μεταβούν σ'αυτές.

```
SAT_EX(Φ){
   X = SAT(Φ);
   Y = {s ∈ S | υπάρχει s' τ.ώ. s → s', s'∈ X};
   return Y;
}
```

Η διαδικασία $SAT_{AF}(\Phi)$

- Υπολογίζει τις καταστάσεις X που ικανοποιούν τη Φ (SAT(Φ)) και μετά οπισθοδρομεί για να υπολογίσει το σύνολο των καταστάσεων των οποίων κάθε εκτέλεση φθάνει σε μία από τις καταστάσεις X σε ένα βήμα, δύο βήματα, κ.ο.κ.

```
SAT_AF(Φ){
  X = S;
  Y = SAT(Φ);
  while (X != Y)
      X = Y;
      Y = Y ∪{s ∈ S | αν, για κάθε s' τ.ω. s → s',
                                  τότε s'∈ X};
  return Y;
}
```

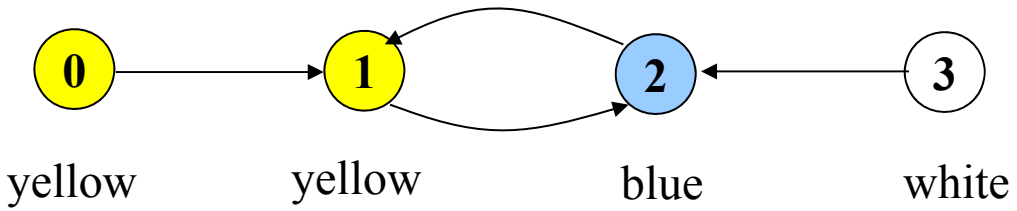
Η διαδικασία $SAT_{EU}(\Phi,\Psi)$

- Υπολογίζει τις καταστάσεις W και Y που ικανοποιούν τις Φ και Ψ αντίστοιχα και μετά οπισθοδρομεί από τις καταστάσεις Y προσθέτοντας στο σύνολο των αποδεκτών καταστάσεων εκείνες που ανήκουν στη W.

```
SAT_EU(Φ, Ψ){
   X = S;
   W = SAT(Φ);
   Y = SAT(Ψ);
   while (X != Y)
        X = Y;
        Y = Y ∪{s ∈ W | υπάρχει s' τ.ώ. s → s'
                                και s'∈ X};
   return Y;
}
```

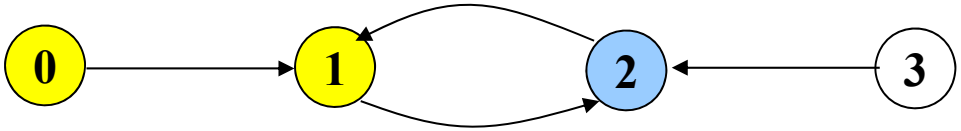
Έλεγχος της E(yellow U blue)

- Θα υπολογίσουμε το σύνολο *Sat*(E(yellow U blue))

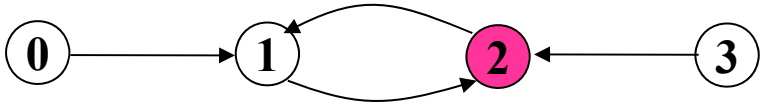


- *Sat*(yellow) = {0, 1}
- *Sat*(blue) = {2}

Έλεγχος της E(yellow U blue)

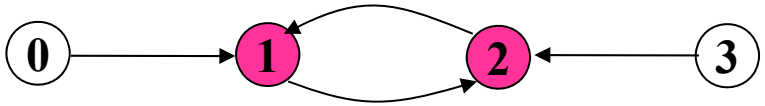


Επανάληψη 1



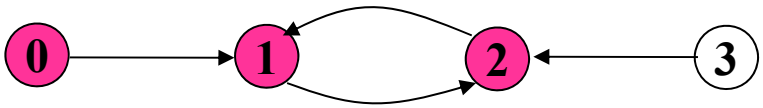
$Y^1 = \{2\}$

Επανάληψη 2



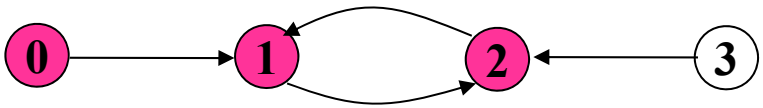
$Y^2 = \{1, 2\}$

Επανάληψη 3



$Y^3 = \{0, 1, 2\}$

Επανάληψη 4



$Y^4 = \{0, 1, 2\}$

Μοντελο-έλεγχος στη CTL

- Αλγόριθμος:
  - διάσχιση του δένδρου που αντιστοιχεί στην ιδιότητα από κάτω προς τα πάνω
  - εύρεση σταθερού σημείου για ιδιότητες AF και EU

- Χρόνος εκτέλεσης χείριστης περίπτωσης είναι της τάξης $O(|\Phi| \cdot N^2)$ όπου $|\Phi|$ είναι το μήκος της ιδιότητας Φ και N ο αριθμός καταστάσεων του μοντέλου του συστήματος.

- Υλοποιημένος σε εργαλεία όπως τα UPPAAL, SMV, Cadence, NuSMV…

CTL στην πράξη

- Τυπικές ιδιότητες μπορούν να διατυπωθούν ως ψηλού επιπέδου προδιαγραφές.

- Σε τέτοιες προδιαγραφές ο χρήστης
  - δεν χρειάζεται να γνωρίζει χρονική λογική
  - απλά τοποθετεί τις ατομικές προτάσεις που τον ενδιαφέρουν στις προδιαγραφές που θέλει να ελέγξει

- Τέτοιου είδους προδιαγραφές μπορούν να χωριστούν σε τρεις βασικές κατηγορίες
  - ολικές: αναφέρονται στο σύνολο της εκτέλεσης του συστήματος
  - μετά: αναφέρονται σε εκτελέσεις μετά από κάποια κατάσταση
  - ανάμεσα: αναφέρονται στους υπολογισμούς που λαμβάνουν χώρο ανάμεσα σε δύο καταστάσεις

Τυπικές προδιαγραφές ψηλού επιπέδου

- Μελέτη 555 προδιαγραφών έδειξε τις πιο κάτω συχνότητες για τις δημοφιλέστερες προδιαγραφές

προδιαγραφή	*τύπος*	*CTL ιδιότητα*	*συχνότητα*
ανταπόκριση	ολική	**AG** (p → **AF** q)	43.3%
καθολικότητα	ολική	**AG** p	19.8%
απουσία	ολική	**AG** ¬p	7.4%
προβάδισμα	ολική	**AG** ¬p ∨ **A**(¬ p **U** q)	4.5%
απουσία	ανάμεσα	**AG** ((q ∧ ¬r) → **A**((¬ p ∨ **AG** ¬r) **W** r)	3.2%
απουσία	μετά	**AG** (p → **AF** ¬ q)	2.1%
ύπαρξη	ολική	**AF** p	2.1%

CTL (Computation Tree Logic)

Η CTL ορίζεται ως το μικρότερο σύνολο ιδιοτήτων που παράγονται ως εξής:

$$\Phi, \Psi ::= p \mid \neg\Phi \mid \Phi \vee \Psi \mid \mathbf{A}\ \varphi \mid \mathbf{E}\ \varphi$$

$$\varphi ::= \mathbf{X}\ \Phi \mid \Phi\ \mathbf{U}\ \Psi$$

1. Ιδιότητες κατάστασης – Φ
   - κάθε ατομική πρόταση p είναι ιδιότητα *κατάστασης*
   - Αν οι Φ και Ψ είναι ιδιότητες *κατάστασης*, τότε και οι $\neg\Phi$ και $\Phi \vee \Psi$ είναι ιδιότητες *κατάστασης*
   - Αν η φ είναι μια ιδιότητα *εκτέλεσης*, τότε οι $\mathbf{A}\ \varphi$ και η $\mathbf{E}\ \varphi$ είναι ιδιότητες *κατάστασης*
2. Ιδιότητες εκτέλεσης – φ
   - Αν οι Φ και Ψ είναι ιδιότητες *κατάστασης*, τότε οι $\mathbf{X}\ \Phi$ και $\Phi\ \mathbf{U}\ \Psi$ είναι ιδιότητες *εκτέλεσης*

PLTL

Η PLTL μπορεί παρόμοια να οριστεί με βάση την πιο κάτω γραμματική

$$\Phi \quad ::= \quad \mathbf{A}\ \varphi$$

$$\varphi \quad ::= \quad p \quad | \quad \neg\varphi \quad | \quad \varphi \vee \psi \quad | \quad \mathbf{X}\ \varphi \quad | \quad \varphi\ \mathbf{U}\ \psi$$

1. Ιδιότητες κατάστασης – Φ
   - Αν η φ είναι μια ιδιότητα *εκτέλεσης* τότε η $\mathbf{A}\ \varphi$ είναι ιδιότητα *κατάστασης*
2. Ιδιότητες εκτέλεσης – φ
   - κάθε ατομική πρόταση p είναι ιδιότητα *εκτέλεσης*
   - Αν οι φ και ψ είναι ιδιότητες *εκτέλεσης*, τότε και οι $\neg\varphi$ και $\varphi \vee \psi$ είναι ιδιότητες *εκτέλεσης*
   - Αν οι φ και ψ είναι ιδιότητες *εκτέλεσης*, τότε οι $\mathbf{X}\ \varphi$ και $\varphi\ \mathbf{U}\ \psi$ είναι ιδιότητες *εκτέλεσης*

PLTL και CTL

- Οι δύο τύποι λογικής έχουν διαφορετική εκφραστικότητα:
  - υπάρχουν ιδιότητες της CTL που δεν μπορούν να εκφραστούν στην PLTL, π.χ. AG EF p
  - υπάρχουν ιδιότητες που δεν μπορούν να εκφραστούν στην CTL, π.χ. F (p $\wedge$ X p)

    Η ιδιότητα αυτή εκφράζει ότι σε κάθε εκτέλεση η p θα ικανοποιηθεί για δύο συνεχόμενες χρονικές στιγμές. Οι ιδιότητες AF (p $\wedge$ AX p) και AF (p $\wedge$ EX p) εκφράζουν διαφορετικές προτάσεις.

- Η πολυπλοκότητα χρόνου του μοντελο-ελέγχου για τους δύο τύπους λογικής είναι
  - CTL : $O(|\text{Formula}|\cdot|\text{System}|^2)$
  - PLTL : $O(2^{|\text{Formula}|}.|\text{System}|^2)$

  Συχνά όμως ιδιότητες της CTL είναι μακρύτερες από ιδιότητες της PLTL.

CTL*

- Διακλαδωμένη χρονική λογική με μεγαλύτερη εκφραστικότητα.
- Η σύνταξη της δίνεται ως εξής:

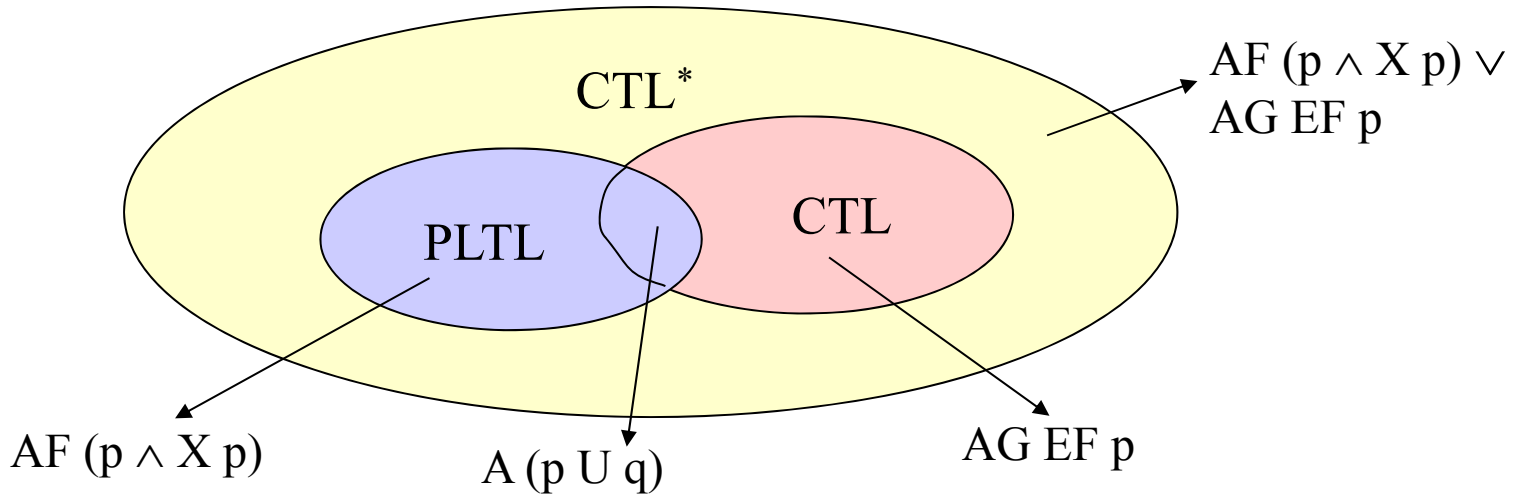
Φ :: = p | $\neg\Phi$ | $\Phi \vee \Psi$ | **A** φ | **E** φ

φ :: = Φ | $\neg\varphi$ | $\varphi \vee \psi$ | **X** ψ | φ **U** ψ

- Έτσι, για παράδειγμα οι ιδιότητες **AXX** p, **EGF** p είναι νόμιμες CTL* ιδιότητες.
- Προσοχή: οι ιδιότητες **AF AF** p (CTL) και **AGF** p (CTL*) αν και συντακτικά διαφορετικές, εκφράζουν την ίδια προδιαγραφή.
- Η πολυπλοκότητα του μοντελο-ελέγχου για τη CTL* είναι PSPACE και $O(2^{|System|}.|Formula|)$. Μέχρι στιγμής δεν έχει διαδοθεί η χρήση εργαλείου για μοντελο-έλεγχο της CTL*.

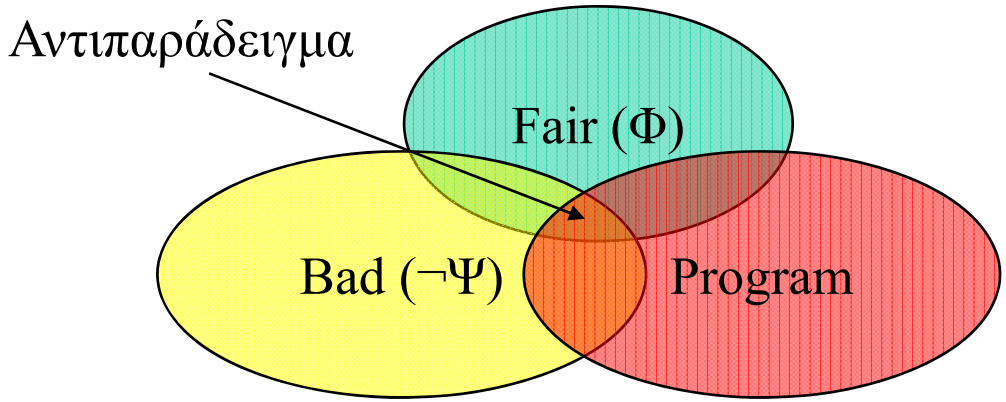
Εκφραστικότητα χρονικών λογικών

- Δύο ιδιότητες είναι *ισοδύναμες* αν και μόνο αν ικανοποιούνται από ακριβώς τις ίδιες καταστάσεις όλων των δομών Kripke.
- Μία χρονική λογική Λ είναι *τουλάχιστον τόσο εκφραστική* όσο και μια λογική Λ' αν και μόνο αν για κάθε ιδιότητά της Λ' υπάρχει ισοδύναμη ιδιότητα της Λ.
- Η εκφραστικότητα των λογικών PLTL, CTL και CTL* φαίνεται στο πιο κάτω διάγραμμα.



Μοντελο-έλεγχος με δικαιοσύνη

- Εκφράζουμε τη ζητούμενη ιδιότητα δικαιοσύνης σαν μια ιδιότητα, έστω Φ και για να αποδείξουμε κάποια ιδιότητα Ψ ελέγχουμε τη $\Phi \rightarrow \Psi$.



- Εναλλακτικά, τροποποιούμε τον αλγόριθμο επαλήθευσης. Για παράδειγμα, για ασθενή δικαιοσύνη διεργασιών: ψάξε για ισχυρά συνδεδεμένους υπογράφους, όπου για κάθε διεργασία P είτε
  - ο υπογράφος περιέχει κάποια μετάβαση της P, ή
  - περιέχει κατάσταση όπου η P δεν μπορεί να προχωρήσει.

Ιδιότητες δικαιοσύνης

- Μη εξαρτώμενη δικαιοσύνη: η ιδιότητα running ικανοποιείται απείρως συχνά

$$\mathbf{G}\ \mathbf{F}\ \texttt{running}$$

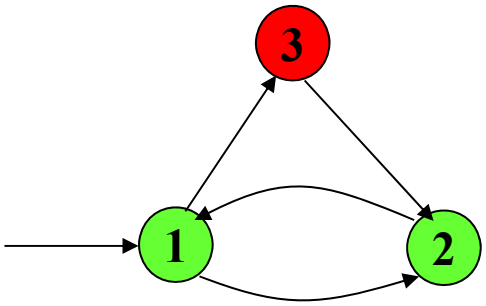
- Ελαφρά δικαιοσύνη: αν κάποια μετάβαση είναι συνεχώς έτοιμη για εκτέλεση θα εκτελεστεί άπειρες φορές

$$\mathbf{F}\ \mathbf{G}\ \texttt{enabled} \rightarrow \mathbf{G}\ \mathbf{F}\ \texttt{running}$$

- Ισχυρή δικαιοσύνη: αν κάποια μετάβαση είναι έτοιμη για εκτέλεση απείρως συχνά τότε θα τρέξει άπειρες φορές

$$\mathbf{G}\ \ \mathbf{F}\ \texttt{enabled} \rightarrow \mathbf{G}\ \mathbf{F}\ \texttt{running}$$

Παράδειγμα



Ισχύει η ιδιότητα

$\Phi = \mathbf{G}(\texttt{green} \rightarrow \mathbf{F}\ \texttt{red});$

- Όχι, αφού υπάρχει ένα "πράσινο" μονοπάτι.
- Το μονοπάτι αυτό όμως δεν είναι δίκαιο διότι η "κόκκινη" μετάβαση είναι έτοιμη για εκτέλεση άπειρες φορές στο μονοπάτι (από την κατάσταση 1) χωρίς ποτέ να εκτελείται.
- Για να αποκλείσουμε αυτό το μονοπάτι μπορούμε να προσθέσουμε μια υπόθεση δικαιοσύνης, όπως την

$$\Phi_\delta = \mathbf{GF}\ \texttt{redenabled} \rightarrow \mathbf{GF}\ \texttt{red}$$

- Τότε ελέγχουμε την ιδιότητα $\Phi_\delta \rightarrow \Phi$, η οποία ισχύει.