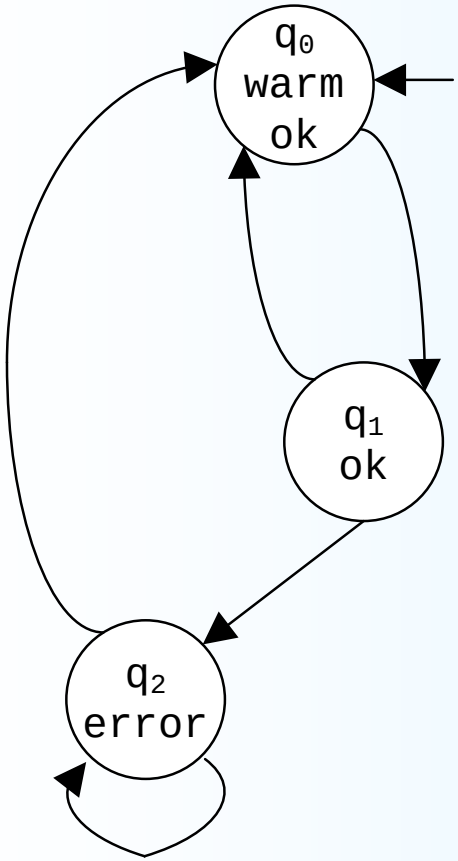
ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ Ι

- Η λογική CTL* (Computation Tree Logic) χρησιμοποιείται από εργαλεία ελέγχου μοντέλων για την τυπική περιγραφή ιδιοτήτων καταστάσεων που αναφέρονται στις εκτελέσεις ενός συστήματος.

- Χρησιμοποιεί ατομικές προτάσεις για τη διατύπωση στοιχειωδών ισχυρισμών, που σε δεδομένη κατάσταση έχουν συγκεκριμένη τιμή αλήθειας (true ή false).

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ ΙΙ



Οι καταστάσεις επιγράφονται με ατομικές προτάσεις που έστω ότι ανήκουν στο σύνολο Prop και παρακάτω έχουμε μερικές από τις πιθανές εκτελέσεις του αυτόματου.

σ_1: $(q_0$: warm, ok$) \rightarrow (q_1$: ok$) \rightarrow (q_0$: warm, ok$)$
$\rightarrow (q_1$: ok$) \rightarrow (q_0$: warm, ok$)$ . . . . .

σ_2: $(q_0$: warm, ok$) \rightarrow (q_1$: ok$) \rightarrow (q_2$: error$)$
$\rightarrow (q_0$: warm, ok$) \rightarrow (q_1$: ok$)$ . . . . .

σ_3: $(q_0$: warm, ok$) \rightarrow (q_1$: ok$) \rightarrow (q_2$: error$)$
$\rightarrow (q_2$: error$) \rightarrow (q_2$: error$)$ . . . . .

σ_4 : . . . . . .

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ ΙΙΙ

- Αναφερόμαστε σε ***προτασιακό τύπο*** όταν χρησιμοποιούμε προτάσεις και λογικούς συνδέσμους ($\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$)

  π.χ. ο προτασιακός τύπος $\texttt{error} \Rightarrow \neg\, \texttt{warm}$ είναι $\texttt{true}$ σε όλες τις καταστάσεις του αυτομάτου του παραδείγματος
- Οι ***χρονικοί σύνδεσμοι*** επιτρέπουν να διατυπώσουμε εκφράσεις για τη χρονική αλληλουχία των καταστάσεων στην πορεία μιας εκτέλεσης:

  X*P* υπονοεί ότι η επόμενη κατάσταση ικανοποιεί την P

  F*P* υπονοεί ότι η *P* ικανοποιείται σε μία μελλοντική κατάσταση

  G*P* υπονοεί ότι η *P* θα ικανοποιείται πάντα

  Παράδειγμα:

  σε όλες τις εκτελέσεις, κάθε εμφάνιση κατάστασης warm ακολουθείται από κατάσταση non-warm

  G ($\texttt{warm} \Rightarrow$ **X** $\neg\texttt{warm}$)

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ IV

- Το **G** είναι το αντίθετο του **F**: για οποιοδήποτε τύπο φ αν ο τύπος ικανοποιείται πάντα, τότε δεν είναι αληθές ότι κάποια στιγμή θα ικανοποιείται ο τύπος $\neg \varphi$. Αυτό το γράφουμε ως: $\mathbf{G}\varphi \equiv \neg \mathbf{F} \neg \varphi$
- Μπορούμε να εμφωλεύουμε τους διάφορους χρονικούς συνδέσμους τον ένα μέσα στον άλλο εκμεταλλευόμενοι κατ΄ αυτό τον τρόπο την εκφραστική ισχύ της γλώσσας, π.χ. $\mathbf{G}(\texttt{alert} \Rightarrow \mathbf{F}\,\texttt{halt})$. Ξεκινώντας από τους απλούστερους τύπους οι χρονικοί σύνδεσμοι δημιουργούν νέους τύπους που η σημασία τους καθορίζεται από τη σημασία των επιμέρους τμημάτων τους.

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ V

- Η τεχνική της εμφώλευσης του **F** μέσα στο **G** χρησιμοποιείται για να εκφράσουμε ιδιότητες επανάληψης.

  GFφ σημαίνει ότι πάντα θα υπάρχει μία κατάσταση στην οποία θα ισχύει η φ

  δηλαδή η ικανοποιείται στην πορεία της εκτέλεσης άπειρα πολλές φορές.

- Το αντίθετο **FG**φ σημαίνει ότι ικανοποιείται συνέχεια από κάποια κατάσταση μετά από πεπερασμένο αριθμό βημάτων.

  ΠΑΡΑΔΕΙΓΜΑ ΣΧΗΜΑΤΟΣ: **GF** `warm` $\vee$ **FG** `error`

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ VI

- Ο χρονικός σύνδεσμος $\cup$ «σημαίνει μέχρις ότου». ΠΑΡΑΔΕΙΓΜΑ: $\mathbf{G}\,(\texttt{alert} \Rightarrow (\texttt{alarm} \cup \texttt{halt}))$
- Ο σύνδεσμος **F** είναι ειδική περίπτωση του $\cup$ με την έννοια ότι ισχύει ότι το $\texttt{true} \cup \varphi$ είναι ισοδύναμο του $\mathbf{F}\varphi$.
- Υπάρχει και το «ασθενές μέχρις ότου» και συμβολίζεται με **W**. Ο τύπος $\varphi_1\ \mathbf{W}\ \varphi_2$ σημαίνει επίσης την ισχύ του τύπου φ_1 μέχρις ότου ισχύσει ο τύπος φ_2 χωρίς όμως να απαιτείται η δίχως άλλο ισχύς του φ_2 (αν ο δε γίνει ποτέ αληθής, τότε παραμένει αληθής ο φ_1).

  $\varphi_1\ \mathbf{W}\ \varphi_2 \equiv (\varphi_1 \cup \varphi_2) \vee \mathbf{G}\ \varphi_1$

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ VII

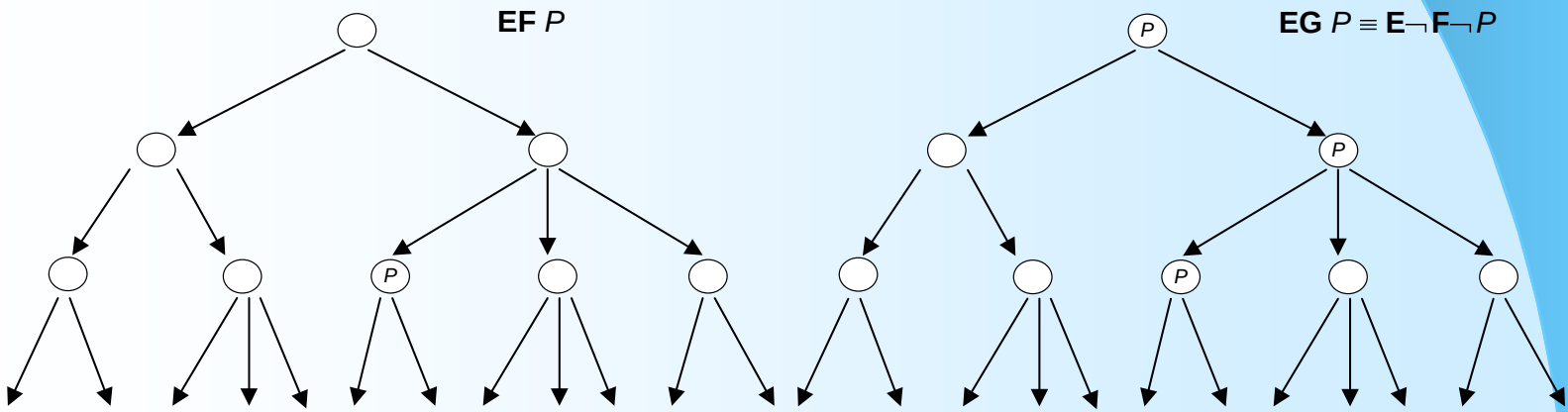
- Οι σύνδεσμοι που περιγράψαμε επαρκούν για να διατυπώσουμε ισχυρισμούς για μία μόνο εκτέλεση.
- Θέλουμε με βάση δεδομένη κατάσταση να μπορούμε να διατυπώσουμε ισχυρισμούς για όλες τις πιθανές εκτελέσεις από την κατάσταση αυτή, δηλ. για όλες τις διακλαδώσεις πάνω στη δενδροειδή αναπαράσταση των πιθανών εκτελέσεων με ρίζα τη συγκεκριμένη κατάσταση.
- Οι σύνδεσμοι **A** και **E** που χρησιμοποιούνται για το σκοπό αυτό ονομάζονται ***ποσοδείκτες διαδρομής*** (path quantifiers).
- Ο τύπος **A**φ αντιστοιχεί στον ισχυρισμό ότι ***όλες οι εκτελέσεις από την τρέχουσα κατάσταση*** ικανοποιούν τον τύπο φ.
- Ο τύπος **E**φ αντιστοιχεί στον ισχυρισμό ότι ***υπάρχει εκτέλεση από την τρέχουσα κατάσταση*** που ικανοποιεί τον τύπο φ.

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ VIII

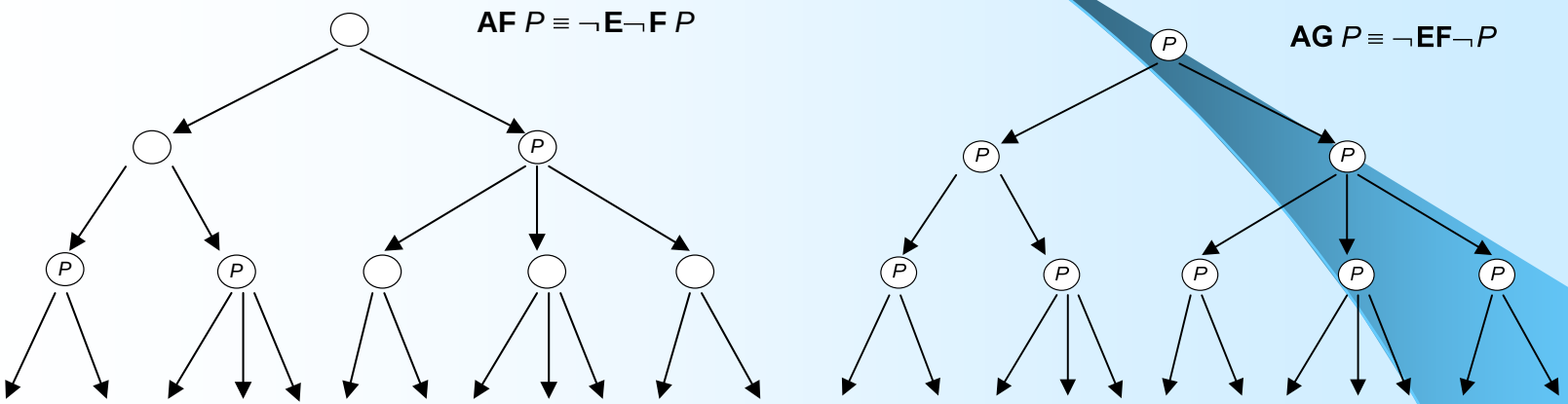
- Δεν πρέπει να συγχέεται ο τελεστής **A** με τον τελεστή **G**: ο τύπος **A**φ αντιστοιχεί στον ισχυρισμό ότι όλες οι δυνατές εκτελέσεις από την τρέχουσα κατάσταση ικανοποιούν τον τύπο φ. Αντίθετα, ο τύπος **G**φ αντιστοιχεί στον ισχυρισμό ότι ο τύπος φ ισχύει σε όλες τις καταστάσεις της υπό εξέταση εκτέλεσης.
- Οι σύνδεσμοι **A** και **E** χρησιμοποιούνται ως ***ποσοδείκτες διαδρομών πάνω σε ένα δέντρο με όλες τις πιθανές εκτελέσεις***, ενώ οι **F** και **G** χρησιμοποιούνται ***ως ποσοδείκτες για τις καταστάσεις κατά μήκος μιας δεδομένης διαδρομής***.

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ IX

- Συχνά οι **A** και **E** συνδυάζονται με τους **G** και **F**.
- Ο τύπος **EF***P* π.χ. αντιστοιχεί στον ισχυρισμό ότι «είναι πιθανό (για κάποια εκτέλεση) να υπάρχει κατάσταση στην οποία ικανοποιείται η *P*».
- Ο τύπος **AF***P* από την άλλη αντιστοιχεί στον ισχυρισμό ότι «είναι σίγουρο (για όλες τις πιθανές εκτελέσεις) ότι υπάρχει κατάσταση στην οποία ικανοποιείται η *P*».



ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ Χ



- Στο παράδειγμα της διαφάνειας 2, κάθε εκτέλεση που αρχίζει από την q_0 ικανοποιεί τον ισχυρισμό
  F EX `error`. Εδώ είναι σημαντικό να μην ξεχάσουμε τον ποσοδείκτη **E** και ο λόγος είναι ότι υπάρχει μία εκτέλεση που δεν ικανοποιεί τον ισχυρισμό **F X** `error`

ΧΡΟΝΙΚΗ ΛΟΓΙΚΗ ΧΙ

- **AGF** *P*

  Για κάθε πιθανή εκτέλεση (**A**) και για όλες τους τις καταστάσεις (**G**) είναι σίγουρο ότι θα υπάρξει μία μελλοντική κατάσταση (**F**) που θα ικανοποιεί την *P*.

  Δηλαδή η *P* ικανοποιείται ***απείρως συχνά*** (infinitely often).

- **AG EF** *P*

  Για κάθε πιθανή εκτέλεση (**A**) και για όλες τους τις καταστάσεις (**G**) θα ήταν πιθανό να προσεγγιστεί κατάσταση που θα ικανοποιεί την *P*.

  Δηλαδή είναι ***πάντα δυνητικά προσεγγίσιμη*** (potentially reachable) μια κατάσταση που ικανοποιεί την *P*. Αυτό ισχύει ακόμη και όταν υπάρχει εκτέλεση στην οποία η *P* δεν ικανοποιείται ποτέ. Αρκεί να υπάρχει μία εκτέλεση (**E**) που να εκπληρώνει το παραπάνω αναφερόμενο γεγονός.

ΤΥΠΙΚΗ ΣΥΝΤΑΞΗ ΧΡΟΝΙΚΗΣ ΛΟΓΙΚΗΣ

φ, ψ	::=	$P_1 \mid P_2 \mid \ldots$	(ατομικές προτάσεις)
		$\mid \neg \varphi \mid \varphi \wedge \psi \mid \varphi \Rightarrow \psi \mid \ldots$	(λογικοί σύνδεσμοι)
		$\mid \mathbf{X}\, \varphi \mid \mathbf{F}\, \varphi \mid \mathbf{G}\, \varphi \mid \varphi \cup \psi \mid \ldots$	(χρονικοί σύνδεσμοι)
		$\mid \mathbf{E}\, \varphi \mid \mathbf{A}\, \varphi$	(ποσοδείκτες διαδρομών)

- Γενικά κάθε εργαλείο υιοθετεί μια συγκεκριμένη σύνταξη που μπορεί π.χ. να χρησιμοποιεί παρενθέσεις και συμβάσεις προτεραιότητας μεταξύ των συνδέσμων, όπως και ένα ιδιαίτερο σύνολο ατομικών προτάσεων και λογικών συνδέσμων.
- Επίσης, όλα τα εργαλεία ελέγχου μοντέλων μπορούν συνήθως να εκτελέσουν έλεγχο σε ένα υποσύνολο της CTL*, που τις περισσότερες φορές είναι η CTL ή η PLTL.

ΣΗΜΑΣΙΑ ΧΡΟΝΙΚΗΣ ΛΟΓΙΚΗΣ Ι

Έστω ότι δίνεται ένα σύνολο $Prop=\{P_1, \ldots\}$ στοιχειωδών προτάσεων. Ένα αυτόματο είναι μία πεντάδα $A=(Q, E, T, q_0, l)$ όπου

- Q είναι ένα πεπερασμένο σύνολο καταστάσεων
- E είναι ένα πεπερασμένο σύνολο επιγραφών μεταβάσεων
- $T \subseteq Q \times E \times Q$ είναι ένα σύνολο μεταβάσεων
- q_0 είναι η αρχική κατάσταση του αυτόματου
- l είναι μία απεικόνιση που αντιστοιχεί σε κάθε κατάσταση του Q το πεπερασμένο σύνολο των προτάσεων που είναι αληθείς στην κατάσταση αυτή

Θα γράφουμε $A, \sigma, i \models \varphi$ και θα το διαβάζουμε ως «τη στιγμή i της εκτέλεσης σ ο τύπος φ είναι αληθής». Συχνά το *περιβάλλον* A εννοείται και δεν το γράφουμε.

ΣΗΜΑΣΙΑ ΧΡΟΝΙΚΗΣ ΛΟΓΙΚΗΣ ΙΙ

- Θα γράφουμε $\sigma, i \not\models \varphi$ όταν θα θέλουμε να δείξουμε ότι τη στιγμή i της εκτέλεσης σ ο τύπος φ δεν είναι αληθής.
- Ο ορισμός της σχέσης $\sigma, i \models \varphi$ γίνεται με επαγωγή ως προς τη δομή του τύπου φ, δηλαδή η τιμή αληθείας του σύνθετου τύπου δίνεται ως συνάρτηση των τιμών αληθείας των «υποτύπων».

$\sigma, i \models P$ ανν $P \in l(\sigma(i))$

$\sigma, i \models \neg\varphi$ ανν δεν είναι αληθές ότι $\sigma, i \models \varphi$

$\sigma, i \models \varphi \wedge \psi$ ανν και $\sigma, i \models \varphi$ και $\sigma, i \models \psi$

$\sigma, i \models \mathbf{X}\varphi$ ανν και $i < |\sigma|$ και $\sigma, i+1 \models \varphi$

$\sigma, i \models \mathbf{F}\varphi$ ανν υπάρχει j τέτοιο ώστε $i \le j \le |\sigma|$ και $\sigma, j \models \varphi$

$\sigma, i \models \mathbf{G}\varphi$ ανν για όλα τα j τέτοια ώστε $i \le j \le |\sigma|$ και $\sigma, j \models \varphi$

$\sigma, i \models \varphi \cup \psi$ ανν υπάρχει j τέτοιο ώστε $i \le j \le |\sigma|$ και $\sigma, j \models \psi$ και για όλα τα k τέτοια ώστε $i \le k < j$ ισχύει $\sigma, k \models \varphi$

$\sigma, i \models \mathbf{E}\varphi$ ανν υπάρχει μία εκτέλεση σ' τέτοια ώστε $\sigma(0) \ldots \sigma(i) = \sigma'(0) \ldots \sigma'(i)$ και $\sigma', i \models \varphi$

$\sigma, i \models \mathbf{A}\varphi$ ανν για όλες τις σ' τέτοιες ώστε $\sigma(0) \ldots \sigma(i) = \sigma'(0) \ldots \sigma'(i)$ έχουμε $\sigma', i \models \varphi$

ΣΗΜΑΣΙΑ ΧΡΟΝΙΚΗΣ ΛΟΓΙΚΗΣ ΙΙΙ

- Τι σημαίνει ο ισχυρισμός ότι το αυτόματο Α ικανοποιεί τον τύπο φ, που γράφεται $A \models \varphi$:

  $A \models \varphi$ ανν $\sigma, 0 \models \varphi$ για κάθε εκτέλεση σ του A

- ΠΡΟΣΟΧΗ: Η σχέση $A \not\models \varphi$ δεν σημαίνει απαραίτητα ότι $A \models \neg\varphi$ ενώ η $\sigma, i \not\models \varphi$ είναι ισοδύναμη με την $\sigma, i \models \neg\varphi$. Ο λόγος είναι ότι στη δεύτερη περίπτωση αναφερόμαστε σε μία μόνο εκτέλεση ενώ στην πρώτη περίπτωση εξετάζουμε ομαδικά την ορθότητα όλων των εκτελέσεων ενός μοντέλου από την αρχική κατάσταση q_0

ΠΡΟΤΑΣΙΑΚΗ ΓΡΑΜΜΙΚΗ ΧΡ. ΛΟΓΙΚΗ (PLTL) I

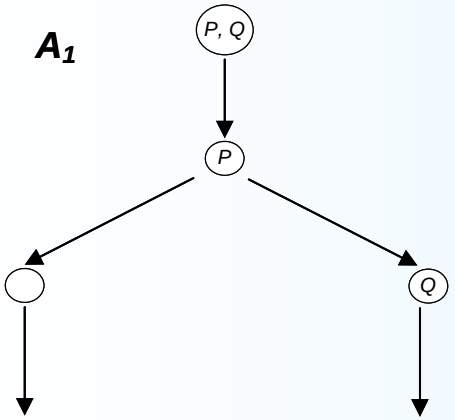
- Η προτασιακή γραμμική χρονική λογική (Propositional Linear Temporal Logic) αποτελεί μέρος της CTL* που προκύπτει από τη μη χρήση των συνδέσμων **A** και **E**.

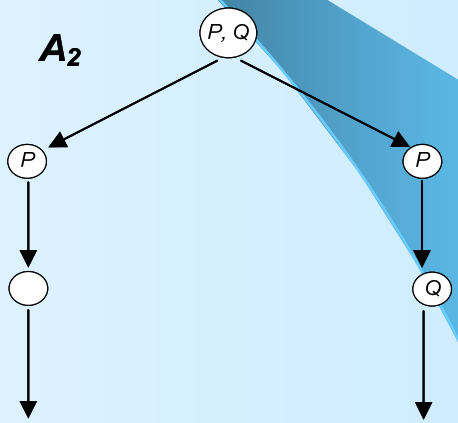
- Ένας τύπος φ στην PLTL για μία δεδομένη εκτέλεση δεν μπορεί να αναφέρεται σε εναλλακτικές εκτελέσεις που διαχωρίζονται από την τρέχουσα εκτέλεση σε καταστάσεις όπου είναι δυνατό να έχουμε μη ντεντερμινιστικές επιλογές. Η PLTL αναφέρεται αποκλειστικά σε ένα σύνολο εκτελέσεων και όχι στον τρόπο με τον οποίο αυτές είναι οργανωμένες σε ένα δέντρο εκτελέσεων. Αυτοί οι τύπο λέγονται ***τύποι διαδρομών*** (path formulas).

- Ένα παράδειγμα τύπου που δεν μπορεί να εκφραστεί στην PLTL είναι το ότι ο *P* είναι «πάντα δυνητικά προσεγγίσιμη», δηλαδή ο τύπος **AGEF***P*.

ΠΡΟΤΑΣΙΑΚΗ ΓΡΑΜΜΙΚΗ ΧΡ. ΛΟΓΙΚΗ (PLTL) ΙΙ

- Τα δύο αυτόματα του σχήματος για την PLTL αντιστοιχούν σε ένα και το αυτό σύνολο διαδρομών:





εκτέλεση 1: {P, Q} . {P} . { _ } . . .

εκτέλεση 2: {P, Q} . {P} . {Q} . . .

και άρα αν ένας οποιοσδήποτε τύπος φ ισχύει για το ένα αυτόματο τότε ισχύει και για το άλλο. Αντίθετα στη CTL αυτό δεν είναι απαραίτητο.

Computation Tree Logic (CTL) I

- Η CTL αποτελεί μέρος της CTL* που προκύπτει από τον περιορισμό κάθε χρήση χρονικού συνδέσμου (**X**, **F**, ∪, κ.α.) να βρίσκεται στην άμεση εμβέλεια ενός ποσοδείκτη **A** ή **E**. Οι συνδυασμοί λοιπόν που είναι διαθέσιμοι στη CTL για χρήση είναι οι **EX**, **AX**, **E_∪_**, **A_∪_** και οι παραγόμενοι όπως π.χ. ο **EF** κ.λ.π. Το κριτήριο του αν ένας τύπος είναι τύπος της CTL είναι η σύνταξη. Παραδείγματα:

  EP∪**A**(P_2∪P_3) είναι CTL

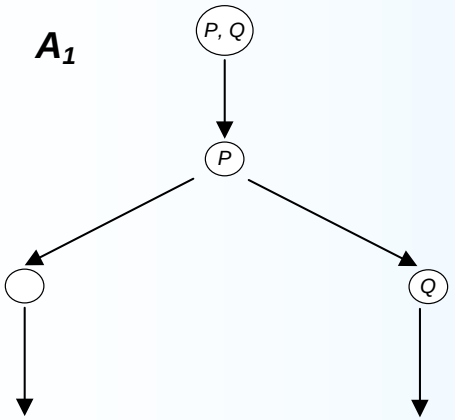
  EP∪**E**(P_2∪P_3) είναι CTL

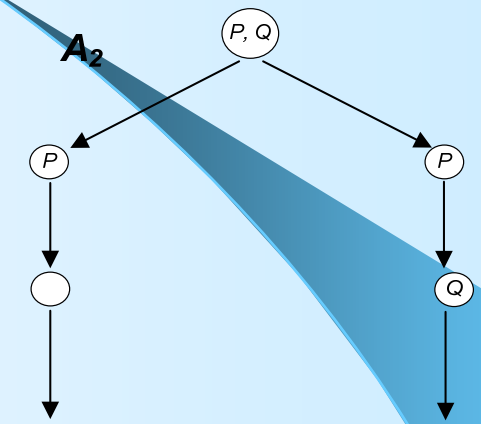
  EP∪(P_2∪P_3) δεν είναι CTL αλλά μπορεί ισοδύναμα να γραφεί ως

  EP∪**E**(P_2∪P_3) που είναι CTL

- Η απαίτηση να χρησιμοποιούμε πάντα ποσοδείκτη για όλα τις πιθανές μελλοντικές εκτελέσεις περιορίζει την εκφραστικότητα της CTL. Η βασική συνέπεια είναι ότι οι τύποι της CTL είναι ***τύποι καταστάσεων*** (state formulas), που σημαίνει ότι η αλήθεια τους εξαρτάται μόνο από την τρέχουσα κατάσταση και όχι από την τρέχουσα εκτέλεση.

Computation Tree Logic (CTL) II





- Είναι δυνατό να διακρίνουμε το A_1 από το A_2 γιατί ισχύει

  $A_1, q_0 \models \mathbf{AX}(\mathbf{EX}Q \wedge \mathbf{EX}\neg Q)$

  ενώ

  $A_2, q'_0 \not\models \mathbf{AX}(\mathbf{EX}Q \wedge \mathbf{EX}\neg Q)$

Computation Tree Logic (CTL) III

- Η CTL μας επιτρέπει να εκφράσουμε δυνητική προσεγγισιμότητα **AGEF***P* αλλά δεν μας επιτρέπει να εκφράσουμε πιο περίπλοκες ιδιότητες, όπως π.χ. η ύπαρξη διαδρομής που ικανοποιεί την *P* απείρως συχνά. Αυτό στην CTL* θα το γράφαμε ως **EGF***P*.

- Στη CTL ο πιο κοντινός σημασιολογικά ισχυρισμός είναι ο ¬ **AF**¬ **EF***P* που διαβάζεται ως «δεν είναι αλήθεια ότι θα προσεγγιστεί απαραιτήτως κατάσταση από δεν θα ξανασυμβεί σε καμία περίπτωση η *P* να είναι ξανά αληθής». Αυτή η τελευταία ιδιότητα είναι σαφώς ασθενέστερη από την **EGF***P* της CTL*. Για να γίνει αυτό κατανοητό φανταστείτε ένα σύστημα στο οποίο ένα self-loop επιτρέπει την παραμονή στην αρχική κατάσταση όσο θέλουμε πριν δυνητικά επιλέξουμε έναν υπολογισμό στον οποίο η *P* γίνεται αληθής μόνο μια φορά. Ένα τέτοιο σύστημα ικανοποιεί την ¬ **AF**¬ **EF***P*, αλλά δεν ικανοποιεί την **EGF***P*.

Computation Tree Logic (CTL) IV

- Γενικά τα μειονεκτήματα της PLTL σε ότι αφορά την εκφραστική ισχύ γίνονται πιο εύκολα ανεκτά από τα μειονεκτήματα της CTL παρόλο που ο έλεγχος μοντέλων με τη CTL είναι πιο γρήγορος.

- Η επιλογή είναι ένας συμβιβασμός που έχει να κάνει με διάφορους παράγοντες:
  - Αν θέλουμε απλά να ελέγξουμε κάποιες ιδιότητες προτιμάμε την PLTL
  - Αν θέλουμε να κάνουμε εξαντλητική επαλήθευση του συστήματος, τότε προτιμάμε την CTL.
  - Αν επιθυμούμε να κάνουμε on-the-fly επαλήθευση για τον εντοπισμό πιθανών λαθών αλλά δεν επιδιώκουμε την εξαντλητική επαλήθευση, τότε επιλέγουμε PLTL.

- Τέλος, η επιλογή μας βασικά εξαρτάται από το τι προσφέρει το εργαλείο που χρησιμοποιούμε.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL I

- Ο αλγόριθμος ελέγχου μοντέλων με CTL κατασκευάστηκε από τους Queille, Sifakis, Clarke, Emerson και Sistla το 1982 και βελτιώθηκε αργότερα το 1986 και το 1994.
- Ο αλγόριθμος είναι γραμμικής πολυπλοκότητας ως προς το μέγεθος του αυτόματου και του τύπου CTL που εξετάζεται.
- Η λειτουργία του αλγορίθμου στηρίζεται στο ότι η CTL εκφράζει αποκλειστικά τύπους καταστάσεων, δηλ. τύπους που η αλήθεια τους εξαρτάται μόνο από την τρέχουσα κατάσταση και όχι από την τρέχουσα εκτέλεση. Αυτό σημαίνει ότι αρκεί ο έλεγχος να περιορίζεται στο ποιες καταστάσεις ικανοποιούν τον ζητούμενο τύπο αγνοώντας την απάντηση στο ερώτημα για ποιες εκτελέσεις ο τύπος είναι αληθής.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL II

ΒΑΣΙΚΗ ΑΡΧΗ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ

Η θεμελιώδης διαδικασία για τη λειτουργία του αλγορίθμου είναι η **marking** που επιδρά πάνω σε ένα αυτόματο A για τον τύπο φ ***σημειώνοντας*** για κάθε κατάσταση q του αυτόματου και για κάθε υπο-τύπο ψ του φ το αν ο ψ ικανοποιείται στην κατάσταση q.

Στο τέλος, για κάθε κατάσταση και κάθε υπο-τύπο, η **q.psi** έχει την τιμή true αν $q \models \psi$, διαφορετικά παίρνει την τιμή `false`.

Ο όρος marking τονίζει το ότι η τιμή **q.psi** πρώτα υπολογίζεται και στη συνέχεια καταγράφεται στη μνήμη. Αυτό είναι σημαντικό γιατί η καταγραφή **q.phi** χρησιμοποιεί τις τιμές **q΄.psi** για τους υπο-τύπους **psi** του **phi** και τις καταστάσεις q' που είναι προσεγγίσιμες από την q.

Όταν τελικά ολοκληρωθεί η καταγραφή για τον τύπο phi μπορούμε να βρούμε αν $A \models \varphi$ ελέγχοντας την τιμή **q0.phi** για την αρχική κατάσταση q_0 του A.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL III

ΑΛΓΟΡΙΘΜΟΣ ΕΛΕΓΧΟΥ ΜΟΝΤΕΛΩΝ ΜΕ CTL (α)

```
procedure marking(phi)

  case 1: phi = P
    for all q in Q, if P in l(q) then do q.phi := true,
                                    else do q.phi := false.

  case 2: phi = not psi
    do marking(psi);
    for all q in Q, do q.phi := not(q.psi).

  case 3: phi = psi1 /\ psi2
    do marking(psi1); marking(psi2);
    for all q in Q, do q.phi := and(q.psi1, q.psi2).

  case 4: phi = EX psi
    do marking(psi);
    for all q in Q, do q.phi := false;       /* initialisation */
    for all (q,q') in T, if q'.psi = true then do q.phi := true.
```

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL IV

- Στις τρεις πρώτες περιπτώσεις η καταγραφή για την φ απαιτεί ένα μόνο πέρασμα από το χώρο καταστάσεων Q που μπορεί να γίνει σε $O(|Q|)$ συν το χρόνο που απαιτείται για την καταγραφή των υπο-τύπων του φ.
- Όταν ο φ έχει τη μορφή **EX** ψ η καταγραφή απαιτεί ένα πέρασμα από το σύνολο T των μεταβάσεων του αυτόματου. Άρα στην περίπτωση αυτή το υπολογιστικό κόστος είναι $O(|T|)$ συν το κόστος αρχικοποίησης και το κόστος για την καταγραφή του ψ. Η περίπτωση **AX** ψ είναι ισοδύναμη με την $\neg$**EX** $\neg\psi$.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL V

ΑΛΓΟΡΙΘΜΟΣ ΕΛΕΓΧΟΥ ΜΟΝΤΕΛΩΝ ΜΕ CTL (β)

```
case 5: phi = E psi1 U psi2
  do marking(psi1); marking(psi2);
  for all q in Q,
    q.phi := false; q.seenbefore := false;/* initialisation */
  L := {};                          /* L: states to be processed */
  for all q in Q, if q.psi2 = true then do L := L + { q };
  while L nonempty {
    draw q from L;                                  /* must mark q */
    L := L - { q };
    q.phi := true;
    for all (q',q) in T {        /* q' is a predecessor of q */
      if q'.seenbefore = false then do {
        q'.seenbefore := true;
        if q'.psi1 = true then do L := L + { q' };
      }
    }
  }
```

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL VI

- Όταν ο τύπος φ έχει τη μορφή $\mathbf{E}\psi_1 \cup \psi_2$ τότε η καταγραφή για τον τύπο φ γίνεται με έναν τυπικό αλγόριθμο ελεγχόμενης προσεγγισιμότητας στο γράφο, ξεκινώντας από τις καταγραφές των ψ_1 και ψ_2. Το υπολογιστικό κόστος είναι $O(|Q|+|T|)$.

```
case 6: phi = A psi1 U psi2
  do marking(psi1); marking(psi2);
  L := {}                              /* L: states to be processed */
  for all q in Q,
    q.nb := degree(q); q.phi := false;        /* initialisation */
  for all q in Q, if q.psi2 = true then do L := L + { q };
  while L nonempty {
    draw q from L;                                /* must mark q */
    L := L - { q };
    q.phi := true;
    for all (q',q) in T {           /* q' is a predecessor of q */
      q'.nb := q'.nb - 1;                              /* decrement */
      if (q'.nb = 0) and (q'.psi1 = true) and (q'.phi = false)
        then do L := L + { q' };
    }
  }
```

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ CTL VII

- Για τον αλγόριθμο ελέγχου μοντέλων με CTL αποδεικνύεται ότι:
  - τερματίζει
  - επιτυγχάνει το επιθυμητό αποτέλεσμα δηλ. είναι ορθός
  - ο έλεγχος μοντέλων του «ικανοποιείται η $A, q_0 \models \varphi$;;» για έναν τύπο CTL έστω φ μπορεί να επιλυθεί σε χρόνο O $(|A| \times |\varphi|)$

  Το τελευταίο ισχύει γιατί πρέπει να γίνει μία καταγραφή για κάθε υπο-τύπο του τύπου φ.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL I

- Ο αλγόριθμος ελέγχου μοντέλων με PLTL κατασκευάστηκε από τους Lichtenstein, Pnueli, Vardi και Wolper το 1985, 1986.
- Στην περίπτωση της PLTL δεν έχουμε να κάνουμε με τύπους καταστάσεων και άρα δεν μπορούμε να βασίζουμε την τεχνική ελέγχου σε μία καταγραφή που θα αναφέρεται στις καταστάσεις του αυτομάτου. Οι τύποι της PLTL είναι γενικά τύποι διαδρομών και πρέπει να έχουμε υπόψη μας ότι ένα αυτόματο στη γενική περίπτωση αποδίδει άπειρα πολλές διαφορετικές εκτελέσεις που συχνά έχουν άπειρο μήκος. Η προσέγγιση που ταιριάζει σε ένα τέτοιο πρόβλημα είναι αυτή της Θεωρίας Γλωσσών.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL II

- Ας θεωρήσουμε για παράδειγμα τον τελεστή φ: **GF** P, δηλ. πάντα (σε μία εκτέλεση) θα υπάρχει μία κατάσταση στο μέλλον για την οποία θα ισχύει η P. Μία εκτέλεση $q_0, q_1, \ldots$ που ικανοποιεί την φ πρέπει να περιέχει άπειρα πολλές θέσεις $q_{n1}, q_{n2}, \ldots$ στις οποίες θα ισχύει ο τύπος P. Ενδιάμεσα από αυτές τις καταστάσεις θα υπάρχει πάντα ένας πεπερασμένος αριθμός καταστάσεων που θα ικανοποιεί την $\neg P$. Θα λέμε ότι η εκτέλεση είναι της μορφής

  $$((\neg P)^*.(P))^\omega$$

  Ο παραπάνω συμβολισμός είναι μία ω-κανονική έκφραση, όπου "+" αντιπροσωπεύει τη διάζευξη, "*" αντιπροσωπεύει έναν τυχαίο αλλά πεπερασμένο αριθμό επαναλήψεων, ενώ "ω" αντιπροσωπεύει έναν άπειρο αριθμό επαναλήψεων.

  Μία εκτέλεση που από ένα σημείο και πέρα δεν ικανοποιεί τον τύπο φ από το σημείο εκείνο και μετά θα ικανοποιεί τον τύπο $\neg P$

  $$(P + \neg P)^*.(\neg P)^\omega$$

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL III

ΒΑΣΙΚΗ ΑΡΧΗ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ

Ο έλεγχος μοντέλου με την PLTL βασίζεται στην αντιστοίχιση κάθε τύπου φ μία ω-κανονική έκφραση E_φ που περιγράφει τη μορφή που θα έχει μία εκτέλεση που ικανοποιεί τον τύπο φ. Το ερώτημα «ικανοποιείται η $A \models \varphi$;;» τελικά παίρνει τη μορφή «είναι όλες οι εκτελέσεις του A της μορφής που περιγράφεται από την E_φ;».

Ένα εργαλείο ελέγχου PLTL δοθείσης ενός τύπου φ κατασκευάζει το αυτόματο $B_{\neg\varphi}$ που θα αναγνωρίζει όλες τις εκτελέσεις που δεν ικανοποιούν τον τύπο φ. Στη χειρότερη περίπτωση ο το μέγεθος ενός τέτοιου αυτόματου θα είναι $O(2^{|\varphi|})$.

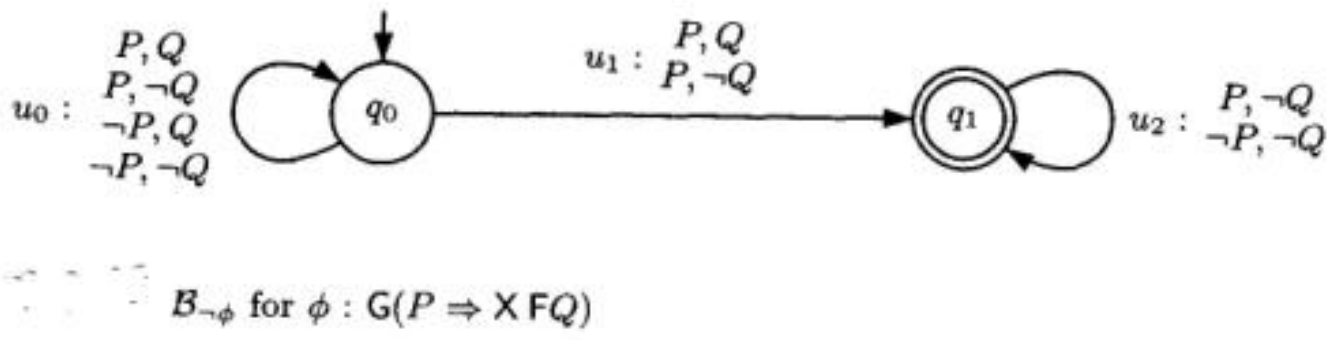
Θεωρούμε τον ισχυρό συγχρονισμό των αυτομάτων A και $B_{\neg\varphi}$ δηλ. το αυτόματο $A \otimes B_{\neg\phi}$ στο οποίο κάθε μετάβαση στο Α γίνεται ταυτόχρονα με μία μετάβαση στο $B_{\neg\varphi}$.

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL IV

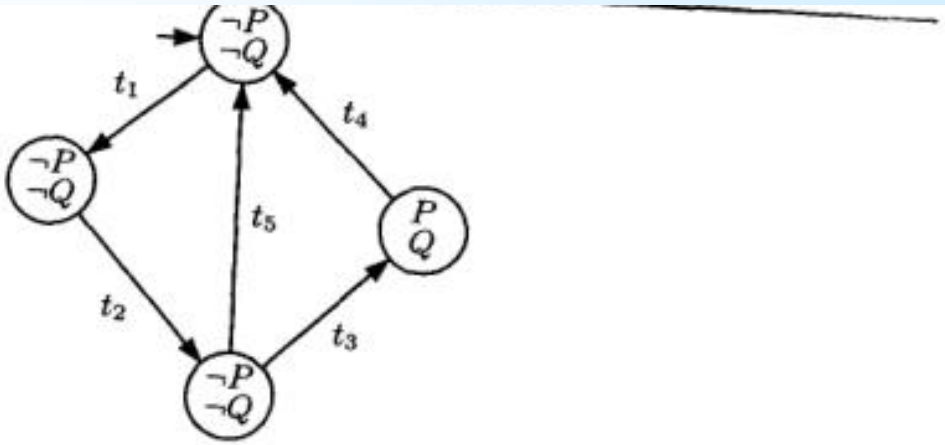
- Το συγκεκριμένο αυτόματο θα αναγνωρίζει τις συμπεριφορές του A που γίνονται δεκτές από το αυτόματο $B_{\neg\varphi}$ αυτές δηλαδή τις συμπεριφορές του Α που δεν ικανοποιούν τον τύπο φ.

- Άρα το πρόβλημα του ελέγχου «ικανοποιείται η $A \models \varphi$;;» μετασχηματίζεται στο «είναι η γλώσσα του $A \otimes B_{\neg\phi}$ κενή;».

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL V

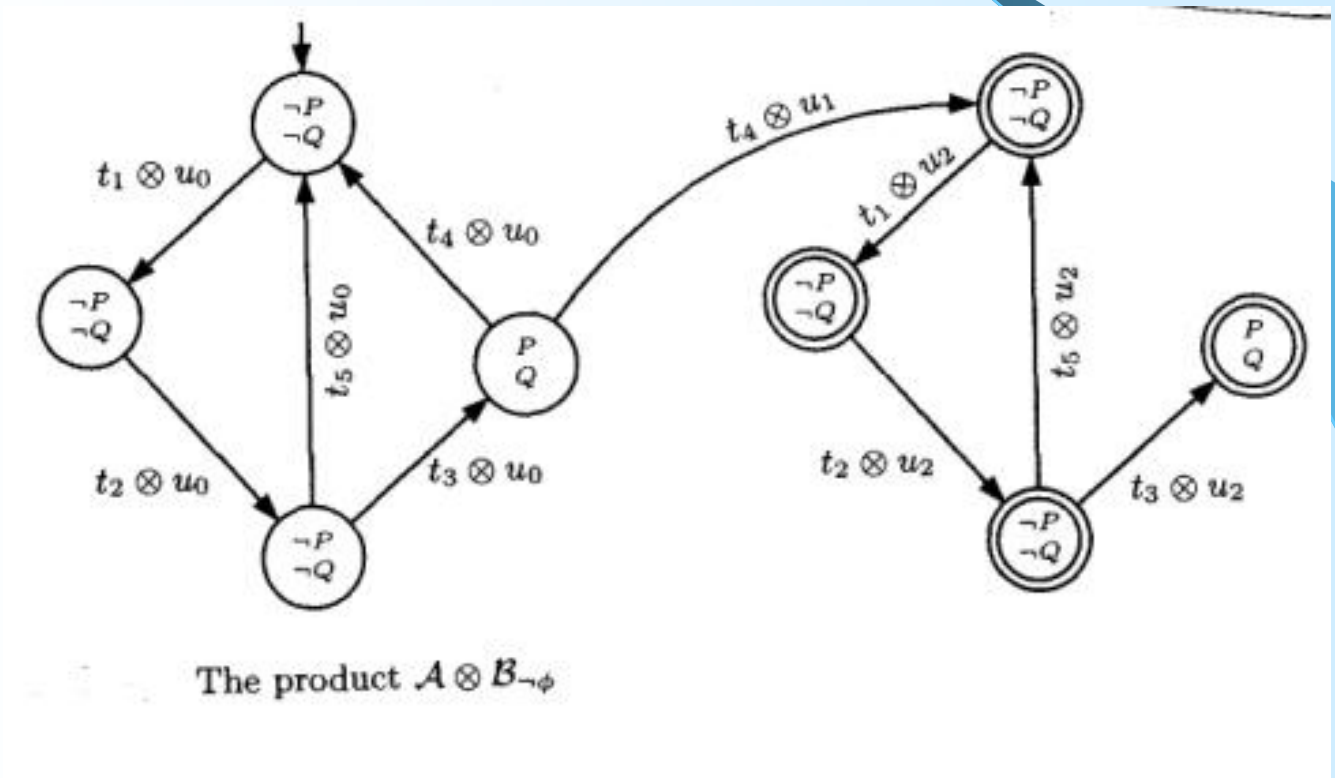


$\mathcal{B}_{\neg\phi}$ for $\phi : \mathsf{G}(P \Rightarrow \mathsf{X}\,\mathsf{F}Q)$



Does the automaton $\mathcal{A}$ verify $\phi : \mathsf{G}(P \Rightarrow \mathsf{X}\,\mathsf{F}Q)$?

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL VI



The product $\mathcal{A} \otimes \mathcal{B}_{\neg\phi}$

ΕΛΕΓΧΟΣ ΜΟΝΤΕΛΩΝ ΜΕ PLTL VII

- Το αυτόματο $B_{\neg\varphi}$ έχει στη χειρότερη περίπτωση μέγεθος $O(2^{|\varphi|})$
- Το αυτόματο $A \otimes B_{\neg\phi}$ έχει μέγεθος $O(|A| \times |B_{\neg\varphi}|)$
- Αν το αυτόματο $A \otimes B_{\neg\phi}$ χωράει στη μνήμη μπορούμε να καθορίσουμε αν αυτό δέχεται μία μη κενή γλώσσα σε χρόνο $O(|A \times B_{\neg\varphi}|)$

- Ο έλεγχος μοντέλου «ικανοποιείται η $A, q_0 \models \varphi$;;» για έναν τύπο φ της PLTL μπορεί να γίνει σε χρόνο $O(|A| \times 2^{|\varphi|})$

ΕΚΡΗΞΗ ΧΩΡΟΥ ΚΑΤΑΣΤΑΣΕΩΝ

- Στην πράξη, το αυτόματο A εύκολα γίνεται ιδιαίτερα μεγάλο (έκρηξη χώρου καταστάσεων).
- Η έκρηξη χώρου καταστάσεων είναι ακόμη πιο πιθανή όταν π.χ. χρησιμοποιούμε μεταβλητές καταστάσεων.
- Στην περίπτωση που οι καθολικές καταστάσεις σχετίζονται με τιμές που δεν είναι εκτων προτέρων φραγμένες (π.χ. ακέραιες μεταβλητές, ουρές αναμονής κ.λ.π.) προκύπτει αυτόματο με άπειρο αριθμό καταστάσεων και τότε δεν εφαρμόζεται καμία από τις τεχνικές που παρουσιάστηκαν.

ΠΡΟΣΕΧΩΣ

- Έλεγχος Μοντέλων με Χρονική Λογική
- Έκρηξη χώρου καταστάσεων
- Συμβολικός Έλεγχος Μοντέλων
- Ιδιότητες Προσεγγισιμότητας (Reachability)
- Ιδιότητες Ασφάλειας (Safety Properties)
- Ιδιότητες Βιωσιμότητας (Liveness Properties)
- Απουσία Αδιεξόδου (Deadlock-freeness)
- Ιδιότητες Δικαιότητας (Fairness Properties)
- Μέθοδοι Αφαίρεσης