Γραμμική Χρονική Λογική (Linear Temporal Logic)
(HR Κεφάλαιο 3.1 και 3.2)

Στην ενότητα αυτή θα μελετηθούν τα εξής θέματα:

Επαλήθευση Συστημάτων και Μοντελοέλεγχος

Σύνταξη της PLTL

Δομές Kripke και Σημασιολογία

Ανάγκη για ανάλυση συστημάτων

- Η κοινωνία της πληροφορίας είναι γεγονός. Υπολογιστές και προγράμματα έχουν εξαπλωθεί σε πολλαπλούς τομείς της ζωής μας:
  - ενθυλακωμένα συστήματα (embedded systems)
  - e-banking και e-shopping
  - μεταφορικά μέσα
  - ιατρική
  - …
- Η αξιοπιστία υλικού και λογισμικού είναι κύριας σημασίας
- Λάθη μπορεί να αποβούν όχι μόνο δαπανηρά (FDIV στο Pentium-II – 475 εκατομμύρια USD) αλλά και μοιραία (Therac-25)

"It is fair to state, that in this digital era correct systems for information processing are more valuable than gold."

Η τύχη του Ariane-5

Το Ariane-5 εκτοξεύθηκε στις 4 Ιουνίου 1996, για να συντριφθεί 36 δευτερόλεπτα αργότερα λόγω ενός σφάλματος στο λογισμικό ελέγχου.

(Το οποίο εκ των υστέρων εντοπίστηκε χρησιμοποιώντας τυπικές μεθόδους.)

Επαλήθευση Συστημάτων

Επαλήθευση συστημάτων αφορά στον έλεγχο κατά πόσο ένα σύστημα ικανοποιεί τις απαιτήσεις που υπάρχουν από αυτό.

Διαδεδομένες τεχνικές επαλήθευσης λογισμικού

- Εξέταση κώδικα από τρίτο
  - στατική τεχνική
  - αναγνωρίζει από 31% μέχρι και το 90% των λαθών
  - "δύσκολα" λάθη, π.χ. αλγοριθμικά λάθη ή λάθη που προκύπτουν από παραλληλισμό στο πρόγραμμα, δεν εντοπίζονται εύκολα
- Testing
  - δυναμική τεχνική όπου ο κώδικας εκτελείται

30-50% του κόστους διεκπεραίωσης ενός λογισμικού προγράμματος αφιερώνεται στο testing.

Ο χρόνος και η προσπάθεια που ξοδεύεται στον έλεγχο ενός συστήματος είναι συχνά μεγαλύτερος από αυτόν της κατασκευής του.

Επιτρεπόμενη πυκνότητα λαθών: 1.5 λάθη ανά 1000 γραμμές κώδικα.

Τυπικές Μέθοδοι – Formal Methods

- Στόχος: η απόδειξη της ορθότητας ενός συστήματος με μαθηματική ακρίβεια
- Χρήση τυπικών μεθόδων σε συνδυασμό με εργαλεία
- Τι είναι οι τυπικές μέθοδοι;
  - *"εφαρμοσμένα μαθηματικά" για τη μοντελοποίηση και ανάλυση υπολογιστικών συστημάτων*
- Προσφέρουν
  - τη δυνατότητα της επαλήθευσης συστημάτων από τη φάση σχεδιασμού
  - αποδοτικές και αυστηρές μεθόδους ελέγχου συστημάτων (πιο μεγάλη κάλυψη και ψηλότερη εγγύηση ορθότητας)
  - μείωση του χρόνου/κόστους που απαιτείται για επαλήθευση

Τυπικές Μέθοδοι – Formal Methods

- Συστήνονται για δημιουργία λογισμικού κρίσιμης ασφάλειας από οργανισμούς όπως τους
  - ESA (European Space Agency
  - FAA (Federal Aviation Authority)
  - NASA

"Formal methods should be part of the education of every computer scientist and software engineer, just as the appropriate branch of applied maths is a necessary part of the education of all other engineers."

NASA

Τυπικές μέθοδοι ανάλυσης συστημάτων

- Στόχος: η απόδειξη της ορθότητας ενός συστήματος με μαθηματική ακρίβεια
- Χρήση τυπικών τεχνικών σε συνδυασμό με εργαλεία
- Διαφορετικές προσεγγίσεις:
  - *τεχνικές ελέγχου μοντέλου* (model-checking)
  - *παραγωγικές μέθοδοι* (deductive methods)
  - *τυπικός έλεγχος πειραμάτων* (formal testing)

Το σύστημα ικανοποιεί την απαίτηση ϕ;

- Παραγωγικές μέθοδοι
  - μέθοδος: δώσε μια λογική απόδειξη ότι η ιδιότητα ϕ ισχύει
  - εργαλεία: theorem prover ή proof checker
  - εφαρμόζεται αν: το σύστημα έχει τη μορφή μαθηματικής θεωρίας

- Προσομοιώσεις και πειράματα
  - μέθοδος: έλεγξε κατά πόσο ισχύει η ϕ σε κάθε εκτέλεση του συστήματος
  - εργαλεία: προσομοιωτής/tester
  - εφαρμόζεται αν: το σύστημα είναι εκτελέσιμο

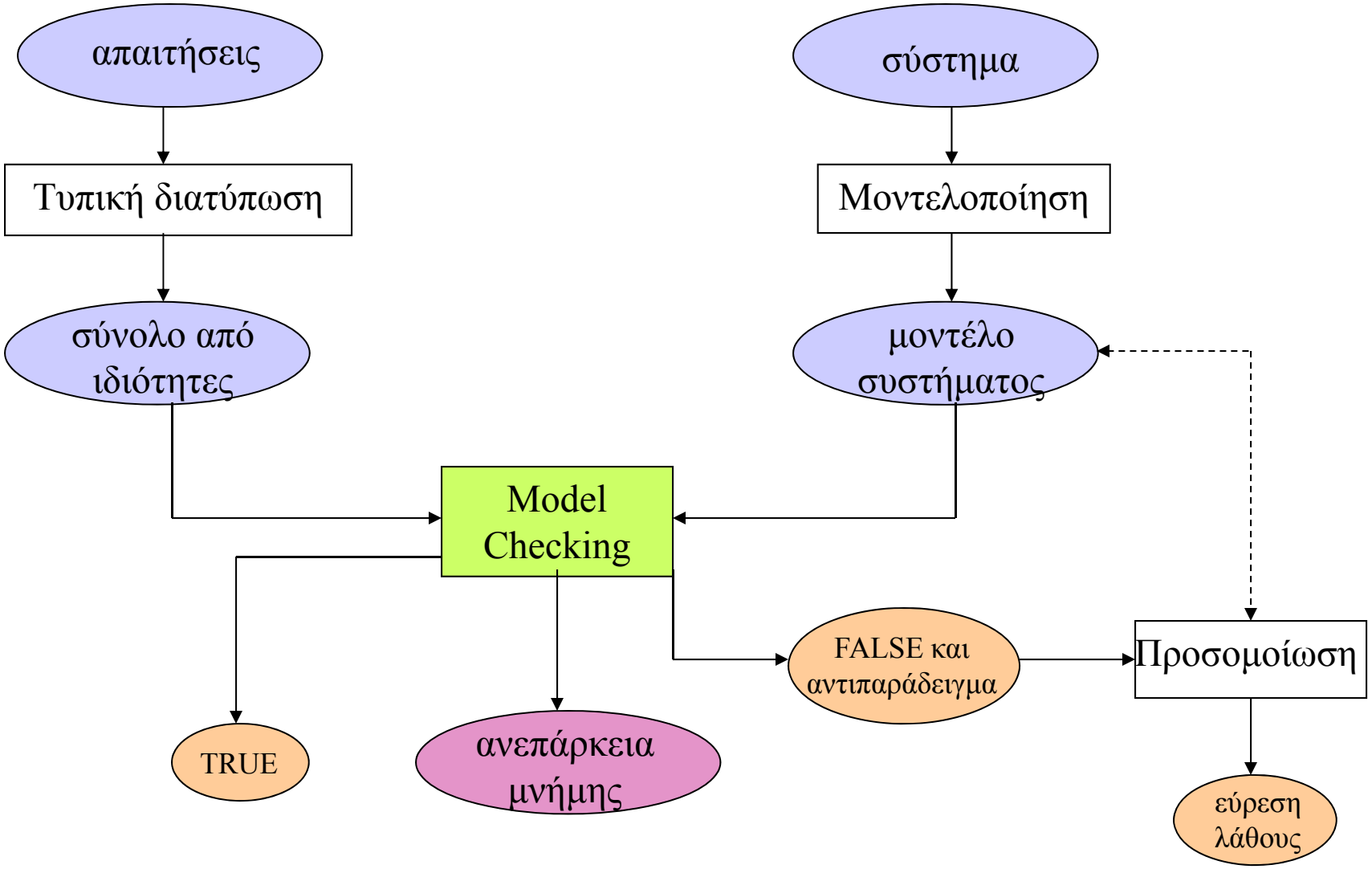
- Μοντέλο-έλεγχος
  - μέθοδος: συστηματικός έλεγχος της ιδιότητας ϕ στις καταστάσεις του μοντέλου.
  - εργαλεία: model-checker
  - εφαρμόζεται αν: το σύστημα παράγει ένα (πεπερασμένο) μοντέλο συμπεριφοράς

Model-checking

- Επαναστατική μέθοδος για την αυτοματοποιημένη επαλήθευση συστημάτων
- Ελέγχει κατά πόσο ένα μοντέλο ικανοποιεί μια τροπική ιδιότητα π.χ.
  - Το σύστημα δίνει το σωστό αποτέλεσμα;
  - Το σύστημα περιέχει αδιέξοδα (deadlock); Για παράδειγμα όταν δύο παράλληλα προγράμματα περιμένουν το ένα το άλλο, σταματώντας με αυτό τον τρόπο ολόκληρο το σύστημα.
  - Μπορεί να εμφανιστεί αδιέξοδο μέσα σε μια ώρα από την εκκίνηση της εκτέλεσης του συστήματος;
  - Η απάντηση δίνεται πάντα μέσα σε 8 λεπτά;
- Βασίζεται σε συστηματικό έλεγχο του συνόλου καταστάσεων ενός συστήματος.

Ο μοντέλο-έλεγχος απαιτεί μια ακριβή και ξεκάθαρη δήλωση των ιδιοτήτων που θα ελεγχθούν. Αυτό γίνεται συνήθως σε μια χρονική λογική (temporal logic).

Model-checking



Πλεονεκτήματα του μοντέλο-ελέγχου

- Εφαρμόζεται σε πολλά συστήματα (υλικό, λογισμικό, πρωτόκολλα,…)
- Ύπαρξη εργαλείων
- Σε περίπτωση μη ικανοποίησης μας δίνει αντιπαράδειγμα
- Μαθηματικά αποδεδειγμένη
- Κοιτάζει όλες τις δυνατές εκτελέσεις

Περιορισμοί του μοντέλο-ελέγχου

- Ασχολείται κυρίως με εφαρμογές με πολύπλοκη ροή και όχι επεξεργασία δεδομένων
- Τα αποτελέσματα βασίζονται στην ποιότητα του μοντέλου
- Η πολυπλοκότητα (state-space explosion problem) μπορεί να αποτελέσει εμπόδιο
- Η δημιουργία κατάλληλων μοντέλων απαιτεί πείρα.

And the Turing award goes to…

- Milner, 1991: for the developments of logical formalisms in computer science (LCF, the mechanization of Scott's Logic of Computable Functions, probably the first theoretically based yet practical tool for machine assisted proof construction).

Robin Milner
(1934-2010)

- Pnueli, 1996: For seminal work introducing temporal logic into computing science and for outstanding contributions to program and systems verification.

Amir Pnueli
(1941-2009)

- Clarke, Emerson, Sifakis, 2007: For [their roles] in developing model checking into a highly effective verification technology, widely adopted in the hardware and software industries.

E. Allen Emerson (1945-)

Edmund M. Clarke, (1945-)

Ιωσήφ Σιφάκης (1946 -)

Παραδείγματα Εφαρμογών

- Ασφάλεια: Το πρωτόκολλο Needham Schroeder
  - Εύρεση λάθους 17 χρόνια μετά από τη δημιουργία και διαδεδομένη χρήση του
- Λογισμικό σε διαστημικούς πυραύλους
  - Το Mars Pathfinder της NASA
- Συστήματα συγκοινωνιών
  - Μοντέλο τραίνων με 10^{476} καταστάσεις
- Εργαλεία μοντελο-ελέγχου για τις C, Java, C++

Ιδιότητες πρωτοκόλλων αμοιβαίου αποκλεισμού

- Σε ένα πρωτόκολλο αμοιβαίου αποκλεισμού (mutual exclusion) συνήθως επιζητούμε ικανοποίηση ιδιοτήτων όπως:
  - *Απαγορεύεται να βρίσκονται περισσότερες από μία διεργασίες στην κρίσιμη τους περιοχή ταυτόχρονα*
  - *Αν σε κάποια χρονική στιγμή μια διεργασία επιθυμεί να μπει στην κρίσιμη της περιοχή, τότε, κάποτε θα της δοθεί η άδεια για να το πράξει*

- Πως μπορούμε να εκφράσουμε τέτοιες ιδιότητες με σαφήνεια και ακρίβεια;

Ιδιότητες φώτων της τροχαίας

- Ιδιότητες που αναμένουμε να ικανοποιούνται από τα φώτα της τροχαίας περιλαμβάνουν
  - *Από κόκκινο το φως δεν μπορεί να γίνει αμέσως πράσινο*
  - *Στο μέλλον το φως θα ξαναγίνει πράσινο*
  - *Από κόκκινο το φως θα γίνει πράσινο αφού περάσει από το κίτρινο για κάποιο χρονικό διάστημα*

- Πως μπορούμε να εκφράσουμε τέτοιες ιδιότητες με σαφήνεια και ακρίβεια;

- Χρησιμοποιώντας *χρονική λογική* (temporal logic).

Χρονική και τροπική λογική

- "Γλώσσα" για τη διατύπωση ιδιοτήτων συστημάτων που αφορούν *συμπεριφορά που μεταβάλλεται με τον χρόνο*.
- Τροπικές λογικές (modal logics) αναπτύχθηκαν από φιλόσοφους για τη μελέτη διαφορετικών τρόπων "αλήθειας" (δυνατότητα και αναγκαιότητα, π.χ. "απαραίτητα ισχύει η Φ", "δυνατόν να ισχύει η Φ").
- Χρονικές λογικές (temporal logics) αποτελούν μια ειδική κατηγορία τροπικών λογικών όπου οι λογικές τιμές των ισχυρισμών μεταβάλλονται με τον χρόνο.
- Τυπικοί κατασκευαστές (χρονικοί τελεστές) είναι
  - $\Diamond$ `p` ή `F p` : Το p θα συμβεί σε κάποια μελλοντική στιγμή
  - $\Box$ `p` ή `G p` : Το p θα συμβεί σε κάθε μελλοντική στιγμή
- Η χρονική λογική χρησιμοποιείται συχνά για την προδιαγραφή και επαλήθευση συστημάτων αλληλεπίδρασης.

Ατομικές προτάσεις

- Οι *ατομικές προτάσεις* είναι τα βασικά στοιχεία μιας χρονικής λογικής και αντιστοιχούν σε λογικές εκφράσεις p, q, r που χρησιμοποιούν
  - μεταβλητές δεδομένων (ακέραιοι, λίστες, σύνολα, κλπ) και μεταβλητές ελέγχου (π.χ. program counter)
  - σταθερές (ακέραιοι 0, 1, 2, ..., λίστες, σύνολα)
  - κατηγορηματικά σύμβολα ($\geq$, $\leq$, $\subseteq$, $\in$)

Σύνταξη γραμμικής χρονικής λογικής

- Η *Προτασιακή Γραμμική Χρονική Λογική* (PLTL) ορίζεται ως το μικρότερο σύνολο ιδιοτήτων που παράγονται ως εξής:

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

Δηλαδή:

- 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 $\mathbf{X}\ \Phi$ (next) είναι ιδιότητα
- Αν οι Φ και Ψ είναι ιδιότητες, τότε και η $\Phi\ \mathbf{U}\ \Psi$ (until) είναι ιδιότητα

Παραγόμενοι τελεστές

$$\Phi \wedge \Psi \equiv \neg (\neg \Phi \vee \neg \Psi)$$

$$\Phi \rightarrow \Psi \equiv \neg \Phi \vee \Psi$$

$$\Phi \leftrightarrow \Psi \equiv (\Phi \rightarrow \Psi) \wedge (\Psi \rightarrow \Phi)$$

$$\mathtt{true} \equiv \neg \Phi \vee \Phi$$

$$\mathtt{false} \equiv \neg \mathtt{true}$$

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

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

F (future) είναι η ιδιότητα που εκφράζει το “κάποτε”

G (globally) είναι η ιδιότητα που εκφράζει το “πάντα”

Εκφραστικότητα της PLTL

- Η Γραμμική Χρονική Λογική μπορεί να εκφράσει ιδιότητες
  - *ασφάλειας* (safety) : τίποτε "κακό" δεν θα συμβεί, **G** ¬ bad
  - *ζωτικότητας* (liveness): κάτι "καλό" τελικά θα συμβεί, **F** good
  - *ανταπόκρισης* : π.χ. κάθε αίτηση θα εξυπηρετηθεί

    G (req → **F** serviced)
  - *αντιδραστικότητας*: π.χ. (**G F** enabled) → (**G F** executed)

Παραδείγματα ιδιοτήτων

- Έστω ΑΠ οι ατομικές προτάσεις που αφορούν τη μεταβλητή x, τους τελεστές $<$, $\geq$, $\leq$ και $=$, και η συνάρτηση x+c

- Οι πιο κάτω ιδιότητες είναι νόμιμες PLTL ιδιότητες
  - $\neg$ (x + 7 < 21) $\vee$ (x = 32)
  - **F** (x + 12 > 30)
  - **G** (x $\geq$ 0 $\wedge$ x < 20)
  - x = 10 $\rightarrow$ **X** (x $\geq$ 11 **U** x = 0)

- Οι πιο κάτω ιδιότητες είναι μη-νόμιμες
  - **F** (x + 12 > y) $\wedge$ **U** x <10
  - **(X** $\vee$ **XX)**(x+4) $\vee$ (x = 32 **F** (x + 12 > 30))

Ιδιότητες φώτων της τροχαίας

- Από κόκκινο το φως δεν μπορεί να γίνεται αμέσως πράσινο

  $\mathbf{G}\ (\text{κόκκινο} \rightarrow \neg\ \mathbf{X}\ \text{πράσινο})$

- Στο μέλλον το φως θα γίνει πράσινο

  $\mathbf{F}\ \text{πράσινο}$

- Από κόκκινο το φως γίνεται πράσινο αφού περάσει από το κίτρινο για κάποιο χρονικό διάστημα

  $\mathbf{G}\ (\text{κόκκινο} \rightarrow (\text{κόκκινο}\ \mathbf{U}\ (\text{κίτρινο} \wedge (\text{κίτρινο}\ \mathbf{U}\ \text{πράσινο}))))$

Ερμηνεία της PLTL

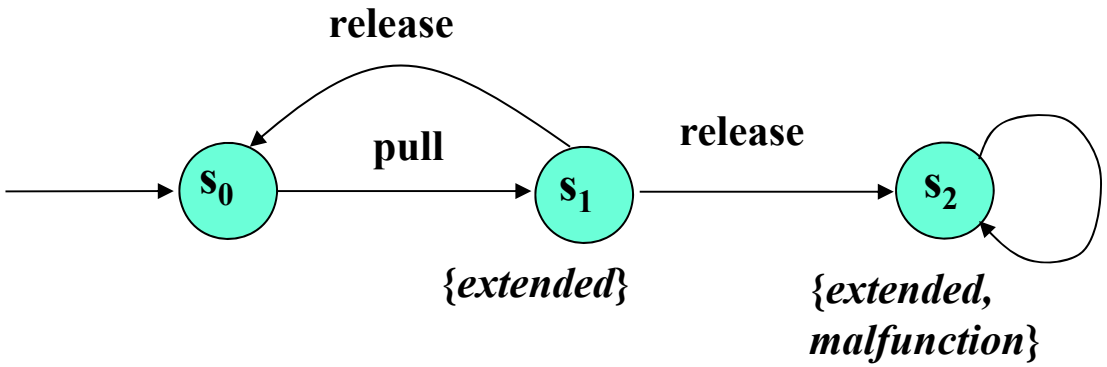
- Η σημασιολογική ερμηνεία της PLTL δίνεται σε σχέση με τις *δομές Kripke*.
- Μια *δομή Kripke* ορίζεται ως μια πλειάδα $M = (S, R, I, Label)$ όπου
  - S είναι ένα αριθμήσιμο σύνολο από καταστάσεις
  - $I \subseteq S$ είναι το σύνολο των αρχικών καταστάσεων
  - $R \subseteq S \times S$ είναι μία σχέση 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 $i \geq 0$.

Παράδειγμα

Η Δομή Kripke

($S = \{s_0, s_1, s_2\}$, $I = \{s_0\}$, $R = \{(s_0, s_1), (s_1, s_0), (s_1, s_2), (s_2, s_2)\}$,

$Label(s_1) = \{extended\}$, $Label(s_2) = \{extended, malfunction\}$)



Μονοπάτια

$p_1 = s_0\ s_1\ s_0\ s_1\ s_0\ s_1\ s_0\ \ldots$

$p_2 = s_0\ s_1\ s_2\ s_2\ s_2\ s_2\ s_2\ \ldots$

$p_3 = s_0\ s_1\ s_0\ s_1\ s_2\ s_2\ s_2\ \ldots$

...

Σημασιολογία της PLTL

- Έστω μονοπάτι $s = s_0\, s_1\, s_2 \ldots$, και ιδιότητα Φ. Ορίζουμε τη σχέση $\models$ όπου

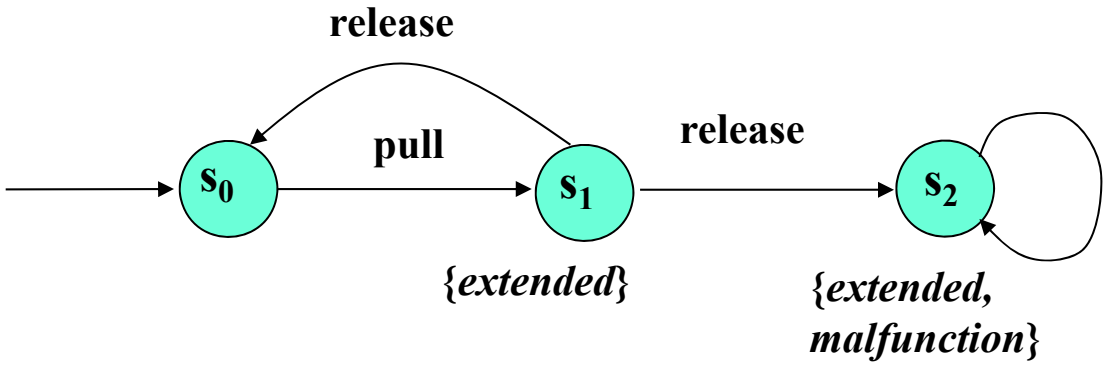
$s \models \Phi$ αν και μόνο αν η ιδιότητα Φ ικανοποιείται στο μονοπάτι s

ως εξής:

$s \models p$	αν και μόνο αν	$p \in \text{Label}(s_0)$
$s \models \neg\Phi$	αν και μόνο αν	δεν ισχύει ότι $s \models \Phi$
$s \models \Phi \vee \Psi$	αν και μόνο αν	$(s \models \Phi)$ ή $(s \models \Psi)$
$s \models \mathbf{X}\ \Phi$	αν και μόνο αν	$s^1 \models \Phi$
$s \models \Phi\ \mathbf{U}\ \Psi$	αν και μόνο αν	υπάρχει $j \geq 0$ τέτοιο ώστε $s^j \models \Psi$ και για κάθε $0 \leq k < j$ $s^k \models \Phi$

Ορισμός: Αν $s = s_0\, s_1\, s_2 \ldots$, s^i είναι το μονοπάτι $s_i\, s_{i+1}\, s_{i+2} \ldots$.

Παράδειγμα



Θεωρήστε το μονοπάτι $p_3 = s_0\ s_1\ s_0\ s_1\ s_2\ s_2\ s_2\ \ldots$

Ποιες από τις πιο κάτω ιδιότητες ικανοποιούνται;

p_3 |= extended ×

p_3 |= **X** extended √

p_3 |= **X X** extended ×

p_3 |= **F** extended √

p_3 |= **G** extended ×

p_3 |= **F G** extended √

p_3 |= ¬ **F G** extended ×

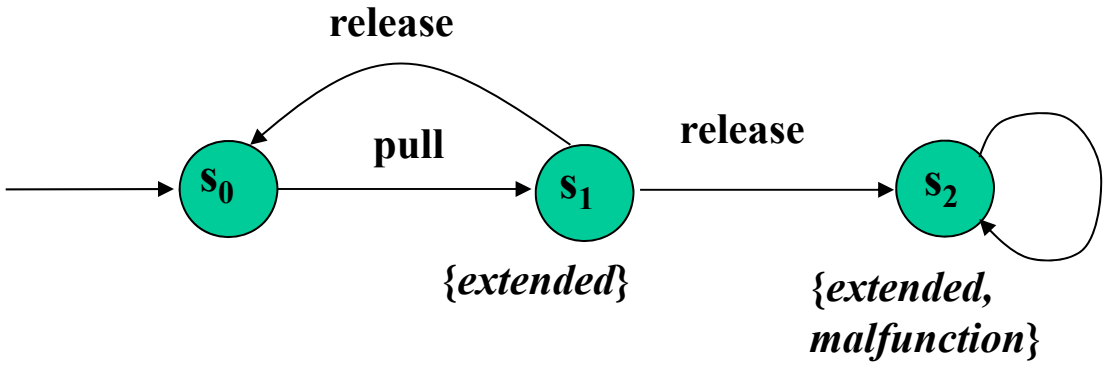
p_3 |= (¬extended) **U** malfunction ×

p_3 |= G(¬extended → **X** extended) √

Σημασιολογία της PLTL (συν.)

- M |= Φ αν και μόνο αν όλα τα μονοπάτια που ξεκινούν από κάθε αρχική κατάσταση της Kripke δομής *M* ικανοποιούν την ιδιότητα Φ.
- Κατά το μοντελο-έλεγχο δεδομένης κάποιας δομής Kripke και μίας ιδιότητας Φ ελέγχουμε κατά πόσο *M* |= Φ.

Παράδειγμα (συν.)



Ποιες από τις πιο κάτω ιδιότητες ικανοποιούνται από το σύστημα;

S |= extended ×

S |= **X** extended √

S |= **X X** extended ×

S |= **F** extended √

S |= **G** extended ×

S |= **F G** extended ×

S |= ¬ **F G** extended ×

S |= (¬extended) **U** malfunction ×

S |= G(¬extended → **X** extended) √

Ισοδυναμίες της PLTL

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

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

$$\neg\, \mathbf{X}\ \Phi \quad \equiv \quad \mathbf{X}\ \neg\ \Phi$$

$$\mathbf{G}\ \mathbf{G}\ \Phi \quad \equiv \quad \mathbf{G}\ \Phi$$

$$\mathbf{F}\ \mathbf{F}\ \Phi \quad \equiv \quad \mathbf{F}\ \Phi$$

$$\Phi\ \mathbf{U}\ (\Phi\ \mathbf{U}\ \Psi) \quad \equiv \quad \Phi\ \mathbf{U}\ \Psi$$

$$\mathbf{F}\ \mathbf{G}\ \mathbf{F}\ \Phi \quad \equiv \quad \mathbf{G}\ \mathbf{F}\ \Phi$$

$$\mathbf{G}\ \mathbf{F}\ \mathbf{G}\ \Phi \quad \equiv \quad \mathbf{F}\ \mathbf{G}\ \Phi$$

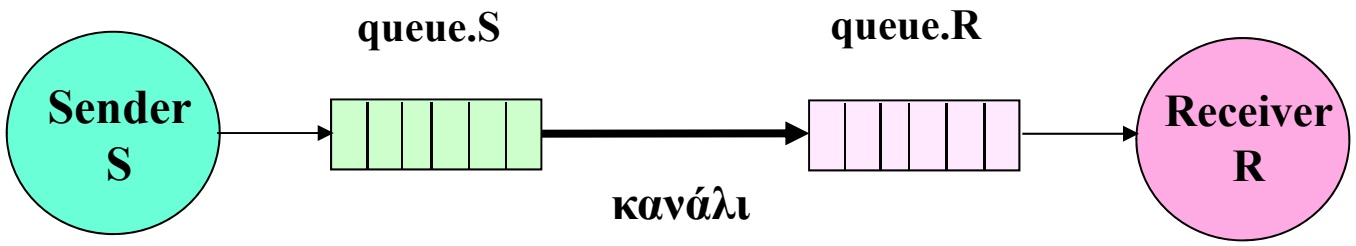
$$\mathbf{X}\ (\Phi\ \mathbf{U}\ \Psi) \quad \equiv \quad (\mathbf{X}\ \Phi)\ \mathbf{U}\ (\mathbf{X}\ \Psi)$$

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

$$\mathbf{F}\ \Phi \quad \equiv \quad \Phi \vee \mathbf{X}\ \mathbf{F}\ \Phi$$

$$\mathbf{G}\ \Phi \quad \equiv \quad \Phi \wedge \mathbf{X}\ \mathbf{G}\ \Phi$$

Προδιαγραφές στην PLTL



- Θα χρησιμοποιήσουμε ατομικές προτάσεις που αφορούν τις μεταβλητές m, m', queue.S, queue.R και τον τελεστή $\in$.

- Ιδιότητα 1: *Ένα μήνυμα δεν μπορεί να ανήκει και στις δύο ουρές*

$$\mathbf{G}\ \neg\ (m \in queue.S \wedge m \in queue.R)$$

- Ιδιότητα 2: *Το κανάλι δεν χάνει μηνύματα*

$$\mathbf{G}\ (m \in queue.S \rightarrow \mathbf{F}\ (m \in queue.R))$$

Προδιαγραφές στην PLTL

- Ιδιότητα 3: *Το κανάλι προωθεί μόνο μηνύματα που προέρχονται από τον S.*

$$\mathbf{G}\ ((\neg\ m \in queue.R)\ \mathbf{U}\ (m \in queue.S))$$

$$\mathbf{G}\ [(\neg(m \in queue.R) \wedge \mathbf{F}\ (m \in queue.R)) \rightarrow (\neg(m \in queue.R)\ \mathbf{U}\ (m \in queue.S))]$$

$$(\neg(m \in queue.R)) \wedge \mathbf{G}\ [(\neg(m \in queue.R) \wedge \mathbf{F}\ (m \in queue.R)) \rightarrow (\neg(m \in queue.R)\ \mathbf{U}\ (m \in queue.S))]$$

Προδιαγραφές στην PLTL

- Ιδιότητα 4: *Το κανάλι προωθεί μηνύματα με τη σειρά που στέλλονται*

$$\mathbf{G}\ [(m \in queue.S) \wedge \neg(m' \in queue.S) \wedge \mathbf{F}(m' \in queue.S))$$
$$\rightarrow (\mathbf{F}\ ((m \in queue.R) \wedge \neg(m' \in queue.R) \wedge \mathbf{F}(m' \in queue.R)))$$
$$]$$

Αμοιβαίος αποκλεισμός

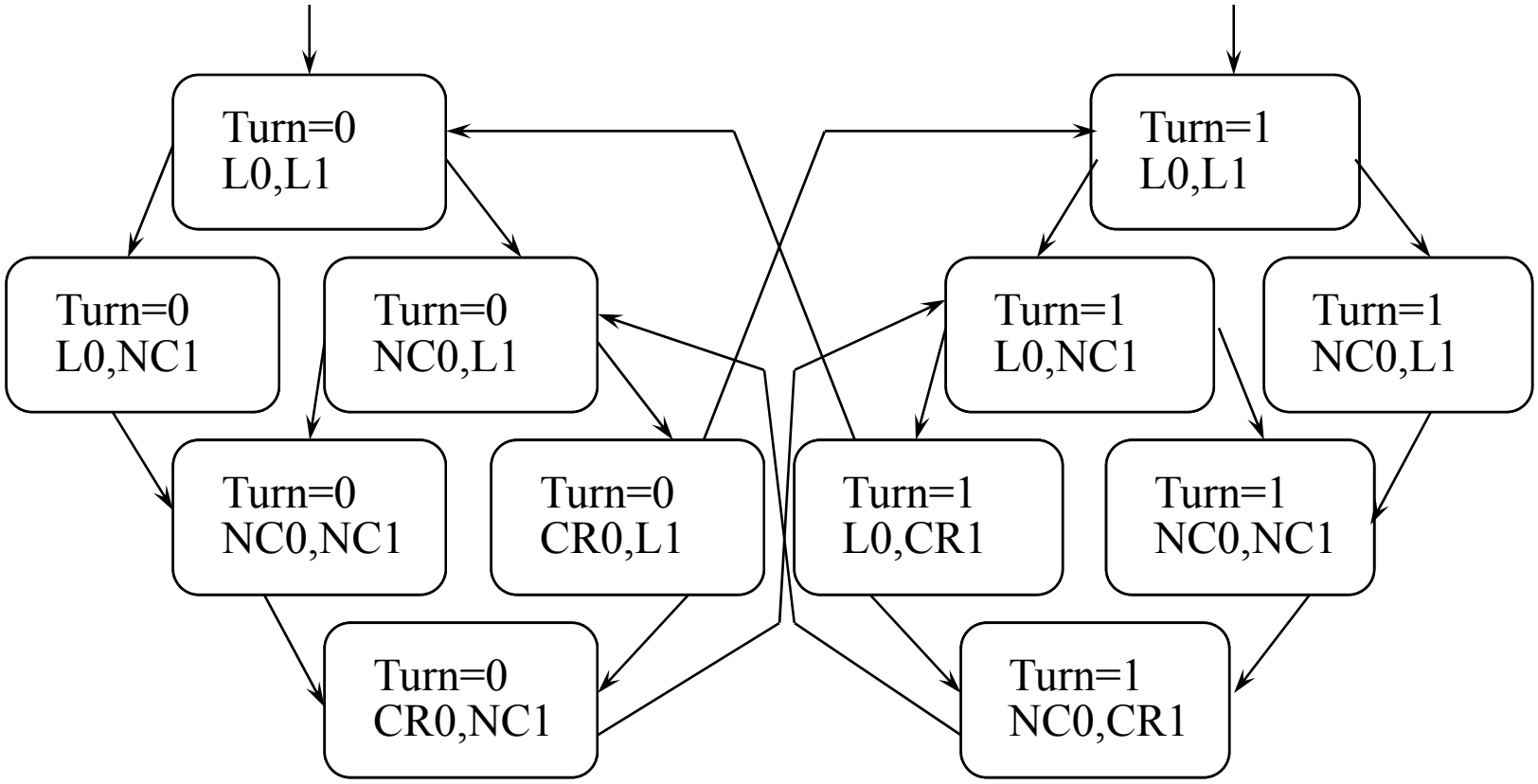
- Θεωρείστε το πιο κάτω πρωτόκολλο αμοιβαίου αποκλεισμού.

```
L0: while True do
    NC0: wait(Turn=0);
    //κρίσιμο τμήμα
    CR0: Turn=1
endwhile
||
L1: while True do
    NC1: wait(Turn=1);
    //κρίσιμο τμήμα
    CR1: Turn=0
endwhile
```

- Ερωτήματα:
  - Υπάρχει περίπτωση κάποια διεργασία να φθάσει σε αδιέξοδο περιμένοντας επ'άπειρω να της δοθεί άδεια πρόσβασης;
  - Υπάρχει περίπτωση να δοθεί και στις δύο διεργασίες άδεια ταυτόχρονα;

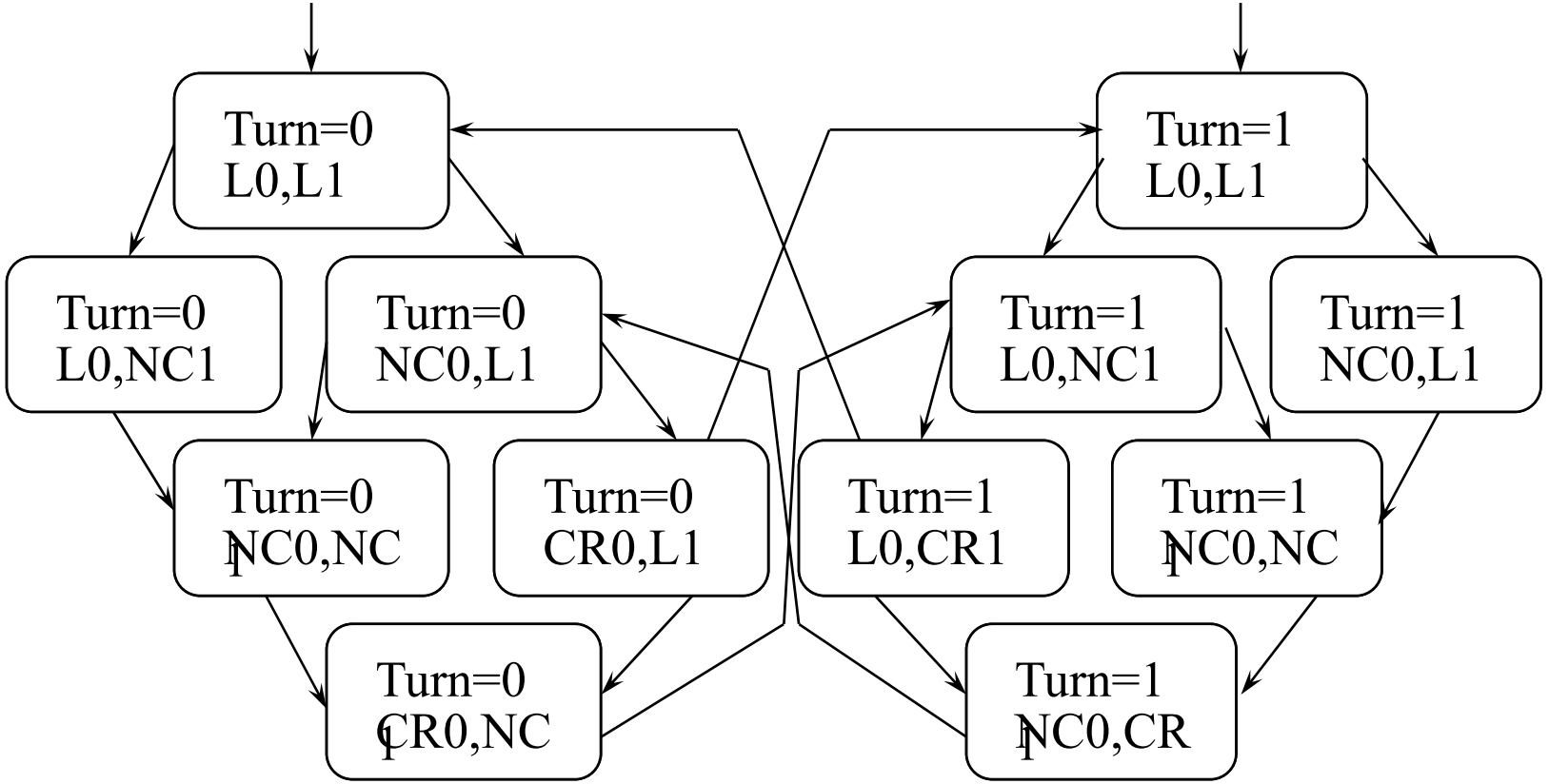
- Πως μπορούμε να διατυπώσουμε με σαφήνεια τις πιο πάνω απαιτήσεις και να ελέγξουμε την ικανοποίησή τους από το σύστημα;

Η δομή Kripke



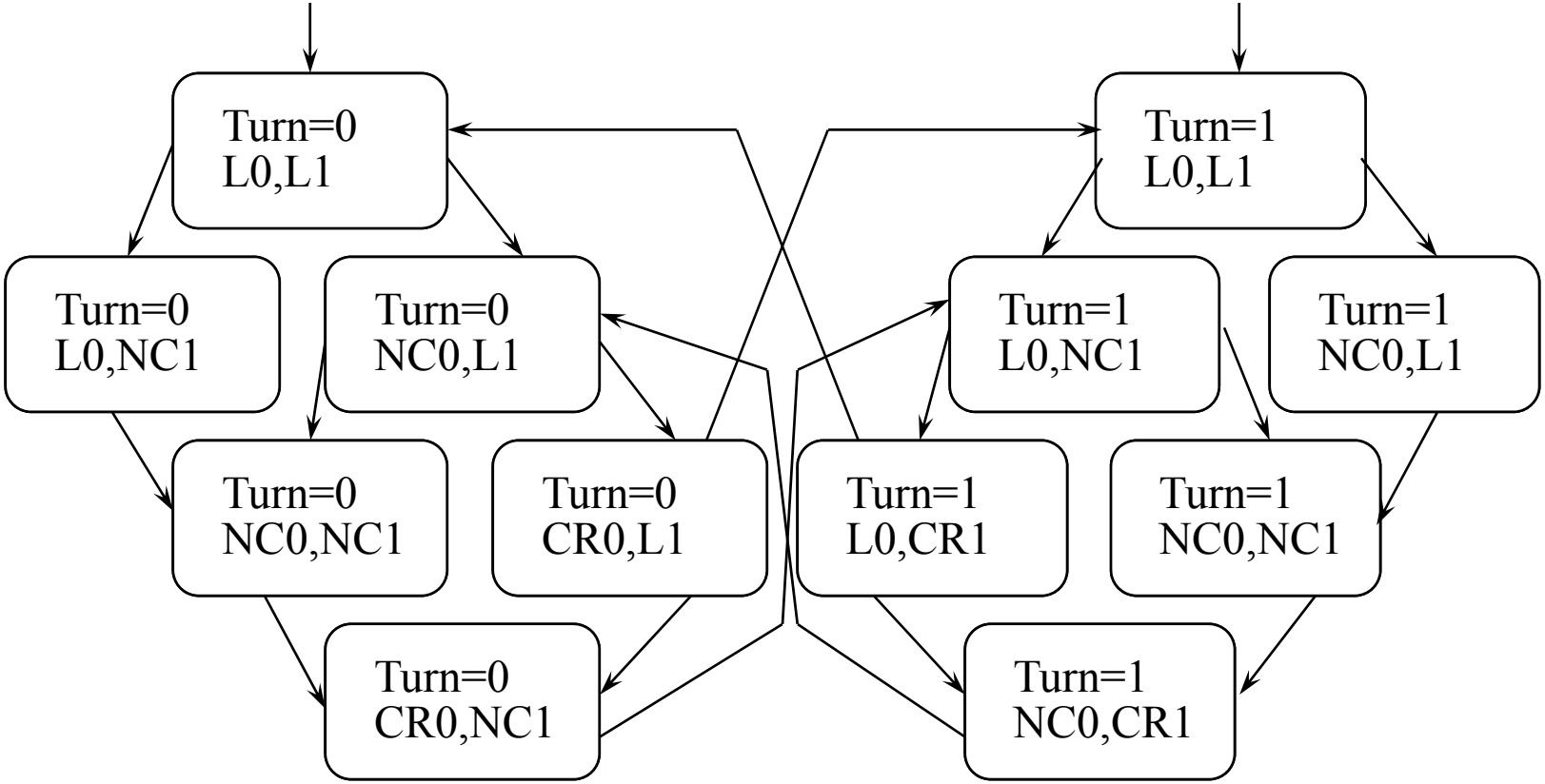
$G\ \neg(PC0=CR0\ \wedge\ PC1=CR1)$

Ποτέ δεν ισχύει ότι και οι δύο διεργασίες έχουν πάρει ταυτόχρονα άδεια



G (Turn=0 → F Turn=1)

Πάντα αν Turn=0 στο μέλλον Turn = 1



Παραλλαγές της PLTL

- Παραλλαγές της PLTL περιλαμβάνουν
  - γενικεύσεις που επιτρέπουν και μονοπάτια πεπερασμένου μήκους
  - προσθήκες τελεστών που επιτρέπουν συλλογισμούς για το παρελθόν

    π.χ. **X** Φ εκφράζει ότι η Φ ικανοποιείται στην προηγούμενη κατάσταση

    G Φ εκφράζει ότι η Φ ικανοποιείται σε κάθε προηγούμενη κατάσταση
  - προσθήκες τελεστών που επιτρέπουν συλλογισμούς για χρονικούς περιορισμούς

    π.χ. $\mathbf{F}^{<t}$ Φ εκφράζει ότι η Φ ικανοποιείται στο μέλλον σε λιγότερες από t μονάδες χρόνου
  - πρωτοβάθμιους κατασκευαστές, κλπ.