Εργαστήριο Λογικού Προγραμματισμού

Μανόλης Μαρακάκης, Καθηγητής
mmarak@cs.hmu.gr

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Σχολή Μηχανικών
Ελληνικό Μεσογειακό Πανεπιστήμιο

Ενότητα 5: Μάθημα 9

Έλεγχος Ροής σε Προγράμματα Prolog

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog

5.1.1. Prolog και οι Κλασσικές Γλώσσες Υψηλού Επιπέδου

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog

- **Ορισμός 5.1:** ***Έλεγχος ροής (control flow)*** είναι η σειρά με την οποία οι εντολές ή οι κλήσεις των διαδικασιών ή οι κλήσεις των συναρτήσεων εκτελούνται ή υπολογίζονται σε μια δηλωτική ή προστακτική γλώσσα προγραμματισμού.
- Άτυπα η διαδικαστική σημασιολογία εκτέλεσης ενός προγράμματος Prolog μπορεί να εκφραστεί ως εξής:
- Για να εκτελεστεί ένας στόχος, το σύστημα ψάχνει την πρώτη πρόταση της οποίας η κεφαλή ενοποιείται με τον στόχο. Η διαδικασία ενοποίησης βρίσκει το ***πλέον γενικό ενοποιητή*** των δύο όρων, ο οποίος είναι μοναδικός εφόσον υπάρχει.

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog

- **Εάν** η ενοποίηση είναι **επιτυχής**,
  - η αντικατάσταση της ενοποίησης εφαρμόζεται στο σώμα της πρότασης και οι στοιχειώδεις τύποι του σώματος της πρότασης γίνονται νέοι στόχοι οι οποίοι εκτελούνται με σειρά από αριστερά προς τα δεξιά.
- **Εάν** το σύστημα **αποτύχει** να ενοποιήσει τον στόχο, **οπισθοδρομεί**. Δηλαδή,
  - **απορρίπτει τον πλέον πρόσφατα ενεργοποιημένο στόχο** ακυρώνοντας τις αντικαταστάσεις οι οποίες έγιναν με την κεφαλή της πρότασης που ταίριαξε.
  - Κατόπιν **επαναλαμβάνει τον αρχικό στόχο** ο οποίος ενεργοποίησε τον απορριφθέντα στόχο και προσπαθεί να βρει μια επόμενη πρόταση η οποία ενοποιείται με τον στόχο.

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog

- Η περιγραφή της ροής ελέγχου ενός προγράμματος Prolog με απλό τρόπο είναι αναγκαία για τους εξής δύο λόγους.
  - Πρώτον, κάποιος που μαθαίνει Prolog να **μπορεί ν' ακολουθήσει την ροή ελέγχου του προγράμματος του ενώ αυτό εκτελείται**.
  - Δεύτερον, τα **συστήματα ανίχνευσης λαθών χρειάζονται ένα απλό μοντέλο ροής ελέγχου** ώστε να βοηθούν αποτελεσματικά τους προγραμματιστές.
- Η **διαδικασία** είναι η βάση για τον μηχανισμό ανίχνευσης λαθών σε προγράμματα από τις γλώσσες Prolog.

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- ***Διαδικασία*** σε Prolog θεωρείται **ένα σύνολο προτάσεων με κεφαλή ίδιο κατηγόρημα** (**ίδιο όνομα κατηγορήματος** και **ίδια πληθηκότητα**).
- Έστω το **Πρόγραμμα 5.1** το οποίο ορίζεται από τις προτάσεις {**C1, C2, C3, C4**}. Οι προτάσεις {**C1, C2**} ορίζουν την διαδικασία για το κατηγόρημα **p/1**. Οι προτάσεις {**C3, C4**} ορίζουν την διαδικασία για το κατηγόρημα **q/2**.

C1: p(a).

C2: p(X) :- q(X,Y), p(Y).

C3: q(b,a).

C4: q(a,a).

Πρόγραμμα 5.1:Πρόγραμμα με διαδικασίες για τα κατηγορήματα **p/1** και **q/2.**

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- Η μοντελοποίηση της εκτέλεσης ενός Prolog προγράμματος γίνεται με **το κιβώτιο της διαδικασίας (*procedure box) ή*** με το **κιβώτιο *του Byrd (Byrd box).***
  - ***Ένα κιβώτιο παριστάνει τη κλήση μιας μόνο διαδικασίας η οποία κλήση συνδέετε με ένα στόχο.***
  - Μια διαδικασία εκτελείτε πολλές φορές και είναι αναγκαίο **να διακρίνουμε όλες τις διαφορετικές κλήσεις**.

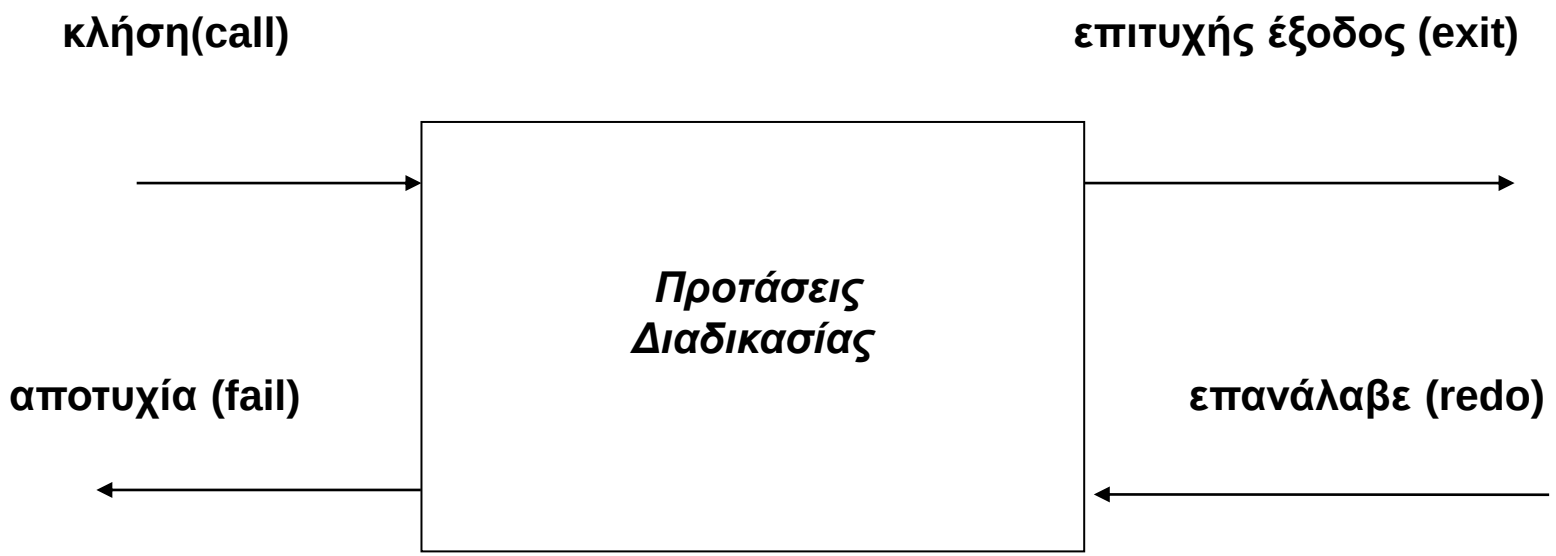
5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- Η μοντελοποίηση της εκτέλεσης φαίνεται σαν κινήσεις **προς** και **από** το κιβώτιο της διαδικασίας **μέσω θυρών (ports) οι οποίες υπάρχουν στο κιβώτιο.**
- Η μοντελοποίηση με το **κιβώτιο της διαδικασίας (*procedure box)*** δίνει ένα απλό τρόπο κατανόησης της ροής ελέγχου, ιδιαίτερα κατά την **οπισθοδρόμιση**
- Αυτό το μοντέλο είναι η **βάση για το μηχανισμό ανίχνευσης λαθών** σε προγράμματα Prolog.

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.



- **Σχήμα 5.1: Θύρες εισόδου και εξόδου σε υποπρόγραμμα.**

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- Οι **θύρες εισόδου** και **εξόδου** από το κιβώτιο μιας διαδικασίας είναι οι εξής:
- ***Κλήση (call):*** Αυτή η θύρα παριστάνει την **αρχική κλήση της διαδικασίας**.
- ***Επιτυχής έξοδος (exit):*** Αυτή η θύρα παριστάνει την **επιτυχή επιστροφή από την διαδικασία**.

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- ***Επανάλαβε (redo):*** Αυτή η θύρα δεικνύει ότι **ένας στόχος έχει αποτύχει** και ότι **το σύστημα οπισθοδρομεί σε μια προσπάθεια να βρει εναλλακτικές λύσεις για τον προηγούμενο του στόχο ώστε να προσπαθήσει ξανά τον στόχο που απέτυχε** και ούτω καθεξής.
- **Αρχικά** προσπαθεί να ικανοποιήσει ξανά ένα από τους προηγούμενους στόχους που τυχόν υπάρχουν στο σώμα της πρότασης που ικανοποιήθηκε τελευταία.
  - **Εάν** κάποιος **στόχος στην ίδια πρόταση επιτύχει**, **τότε** συνεχίζει για να ικανοποιήσει και τους υπόλοιπους στόχους της πρότασης.
  - **Εάν** αυτή η προσπάθεια **αποτύχει**, **τότε προσπαθεί να ενοποιήσει ξανά τον αρχικό (πατρικό) στόχο με μία άλλη πρόταση**.
    - **Εφόσον** ο αρχικός (πατρικός) στόχος **ενοποιηθεί με την κεφαλή μιας πρότασης**, **κατόπιν** προσπαθεί να ικανοποιήσει τους στόχους που τυχόν υπάρχουν στο σώμα της νέας πρότασης και ούτω καθεξής.
    - **Εάν** ο αρχικός (πατρικός) **δεν ενοποιηθεί με την κεφαλή άλλης πρότασης τότε** οπισθοδρομεί για την ικανοποίηση προηγούμενων στόχων και ούτω καθεξής.

- **Αποτυχία (fail):** Αυτή η θύρα **παριστά την αποτυχία του αρχικού (πατρικού) στόχου**. Αποτυχία μπορεί να συμβεί για έναν από τους εξής λόγους:
  - **α) Δεν υπάρχουν προτάσεις** για να ικανοποιηθεί ο αρχικός (τρέχον) στόχος.
  - **β)** Ο αρχικός (τρέχον) στόχος **δεν ενοποιήθηκε με κάποια πρόταση**.
  - **γ)** Η **λύση που παράχθηκε** για τον αρχικό στόχο αυτής της διαδικασίας **απορρίπτεται από άλλη διαδικασία του προγράμματος**.
- Μετά από κάποια αποτυχία **το σύστημα αρχίζει να οπισθοδρομεί**.

5. Έλεγχος Ροής σε Προγράμματα Prolog

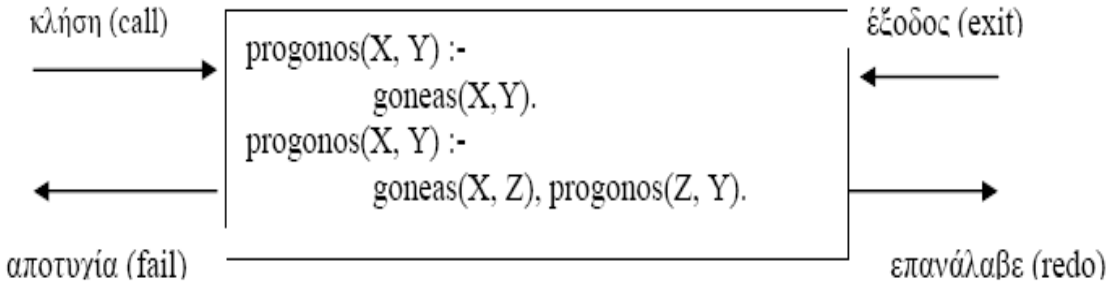
5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- ***Παράδειγμα 2***: Έστω το Πρόγραμμα 5.2 το οποίο αποτελείται από δύο διαδικασίες με θύρες εισόδου και εξόδου όπως φαίνονται στα σχήματα **5.2** και **5.3**.

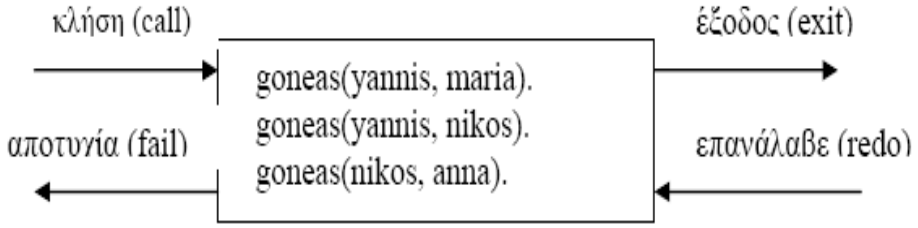
```
progonos(X, Y) :- goneas(X, Y).
progonos(X, Y) :-
    goneas(X, Z), progonos(Z, Y).

goneas(yannis, maria).
goneas(yannis, nikos).
goneas(nikos, anna).
```

Πρόγραμμα 5.2: ***Πρόγραμμα με δύο διαδικασίες***



Σχήμα 5.2:Θύρες εισόδου και εξόδου στο υποπρόγραμμα progonos/2



Σχήμα 5.3:Θύρες εισόδου και εξόδου στο υποπρόγραμμα goneas/2

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- ***Παράδειγμα 2 (συνέχεια)***: Η είσοδος και η έξοδος από τις διαδικασίες progonos/2 και :- goneas/2 για τους στόχους «?- progonos(yannis,nikos).» και «?- progonos(maria,anna).» φαίνονται παρακάτω.

```
progonos(X, Y) :- goneas(X, Y).
progonos(X, Y) :-
    goneas(X, Z), progonos(Z, Y).

goneas(yannis, maria).
goneas(yannis, nikos).
goneas(nikos, anna).
```

Πρόγραμμα 5.2: ***Πρόγραμμα με δύο διαδικασίες***

- ?- **progonos(yannis,nikos).**

```
1    1 Call: progonos(yannis,nikos) ?
2        2 Call: goneas(yannis,nikos) ?
2        2 Exit: goneas(yannis,nikos) ? ?
1    1 Exit: progonos(yannis,nikos) ?  yes
```

- **?- progonos(maria,anna).**

```
1    1 Call: progonos(maria,anna) ?
2        2 Call: goneas(maria,anna) ?
2        2 Fail: goneas(maria,anna) ?
3        2 Call: goneas(maria,_1144) ?
3        2 Fail: goneas(maria,_1144) ?
1    1 Fail: progonos(maria,anna) ? no
```

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- Ο **πρώτος αριθμός** είναι **ο μοναδικός αριθμός ταυτότητα κάθε κλήσης**. Αυξάνεται για κάθε νέα κλήση διαδικασίας.
- Ο **δεύτερος αριθμός είναι το τρέχον βάθος στο οποίο ένα κιβώτιο διαδικασίας έχει κατασκευαστεί από το πρόγραμμα εντοπισμού σφαλμάτων** (debugger). Δηλαδή, ο αριθμός των άμεσα προγόνων που έχει αυτός ο στόχος.
- Η **επόμενη λέξη** καθορίζει τη **θύρα** (**Call, Exit, Redo, Fail**) **στην οποία βρίσκεται η ανίχνευση του προγράμματος**.
- **Μετά εκτυπώνει το στόχο** έτσι ώστε να ελέγξετε/εξετάσετε τη κατάσταση της τρέχουσας δέσμευση του.

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

- ***Παράδειγμα 2 (συνέχεια)***: Η είσοδος και η έξοδος από τις διαδικασίες progonos/2 και :- goneas/2 για το στόχο «?- progonos(yannis,anna).» φαίνεται παρακάτω.

```
progonos(X, Y) :- goneas(X, Y).
progonos(X, Y) :-
    goneas(X, Z), progonos(Z, Y).

goneas(yannis, maria).
goneas(yannis, nikos).
goneas(nikos, anna).
```

Πρόγραμμα 5.2: ***Πρόγραμμα με δύο διαδικασίες***

- **?- progonos(yannis,anna).**

1 **1 Call:** progonos(yannis,anna) ?
2 2 Call: goneas(yannis,anna) ?
2 2 Fail: goneas(yannis,anna) ?
3 **2 Call:** goneas(yannis,_1181) ?
3 2 Exit: goneas(yannis,maria) ?
4 **2 Call**: progonos(maria,anna) ?
5 3 Call: goneas(maria,anna) ?
5 3 Fail: goneas(maria,anna) ?
6 3 Call: goneas(maria,_3112) ?
6 3 Fail: goneas(maria,_3112) ?
4 **2 Fail**: progonos(maria,anna) ?
3 **2 Redo**: goneas(yannis,maria) ? **backtracking**
3 2 Exit: goneas(yannis,nikos) ?
7 **2 Call**: progonos(nikos,anna) ?
8 3 Call: goneas(nikos,anna) ?
8 3 Exit: goneas(nikos,anna) ? ?
7 **2 Exit**: progonos(nikos,anna) ? ?
1 **1 Exit**: progonos(yannis,anna) ? yes

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

Παράδειγμα 3: Η ανίχνευση της εκτέλεσης του **Προγράμματος 5.3** μέσω των θυρών του μοντέλου «*κιβώτιο της διαδικασίας* ή *κιβώτιο του Byrd*» για το στόχο "?- p(a)".

```
p(X) :-   q(X),  r(X).
p(X) :-    s(X).
q(X) :-     s(X).
q(X) :-     t(X).
r(X) :-      u(X, Y),   v(Y).
s(b).
t(a).
u(a, b).
u(a, a).
v(a).
```

Πρόγραμμα 5.3: ***Πρόγραμμα με 7 διαδικασίες.***

```
| ?- p(a).
  1   1 Call: p(a) ?
  2         2  Call: q(a) ?
  3               3  Call: s(a) ?
  3               3  Fail: s(a) ?
  4               3 Call: t(a) ?
  4               3 Exit: t(a) ?
  2         2 Exit: q(a) ?
  5         2 Call: r(a) ?
  6               3 Call: u(a,_4262) ? ?
  6               3 Exit: u(a,b) ?
  7               3 Call: v(b) ?
  7               3 Fail: v(b) ? backtracking
  6               3 Redo: u(a,b) ?
  6               3 Exit: u(a,a) ?
  8               3 Call: v(a) ?
  8               3 Exit: v(a) ?
  5         2 Exit: r(a) ? ?
  1    1 Exit: p(a) ?
Yes
```

- Το **τελικό σύμβολο** '**?**' είναι η προτροπή που υποδεικνύει ότι θα πρέπει να πληκτρολογήσετε μια από τις επιτρεπτές εντολές του debugger (Κεφ. 5.5, σελ. 225 της Sicstus).
- Ουσιαστικά μόνο την εντολή '**h**' ακολουθούμενη από **RET** (**Return**) χρειάζεται να ξέρετε διότι σας δίνει τη λίστα των εντολών του debugger.
- Μερικές πολύ χρήσιμες εντολές είναι οι εξής:
  - **RET** % **creep**: συνέχισε την ανίχνευση, προχώρα ένα βήμα στην επόμενη θύρα
  - **c** % **creep**: ίδιο όπως το προηγούμενο
  - **s <i>** % **skip**: αυτή είναι έγκυρη εντολή μόνο σε θύρες **Call** και **Redo**. Υπερπήδησε % όλη την εκτέλεση του κατηγορήματος. Ο έλεγχος θα επιστρέψει με θύρα Exit ή Fail.
  - **r <i>** % **retry**: μπορεί να χρησιμοποιηθεί σε οποιαδήποτε θύρα. % Μεταφέρει τον έλεγχο πίσω στη θύρα **Call** στο κιβώτιο της διαδικασίας.
  - **a** % **abort:** προκαλεί διακοπή της τρέχουσας εκτέλεσης του προγράμματος.

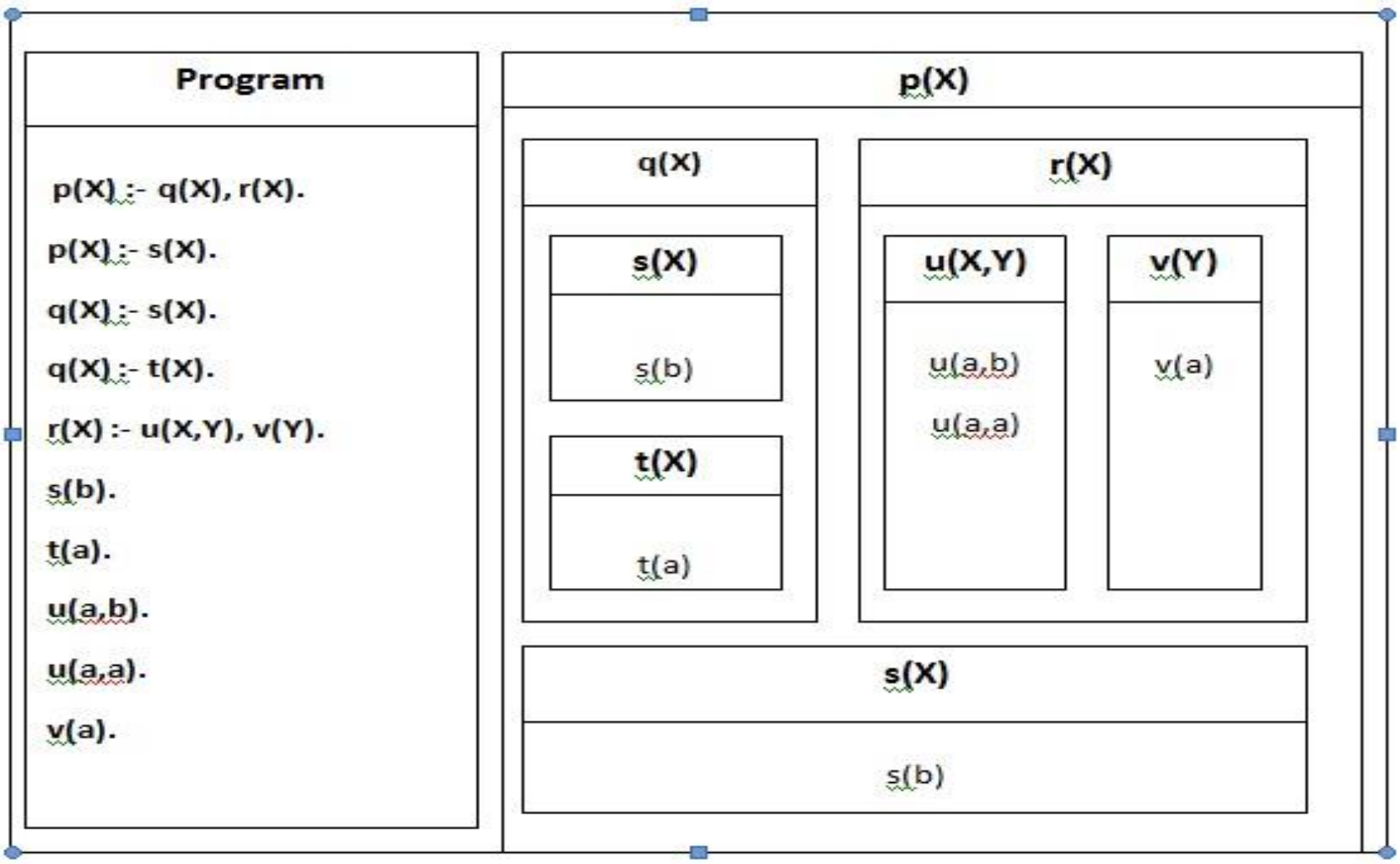
Παράδειγμα 3: Θα δείξουμε τις ανιχνεύσεις διαδοχικών εκτελέσεων του **Προγράμματος 5.3** μέσω των θυρών του μοντέλου «**κιβώτιο της διαδικασίας ή κιβώτιο του Byrd**» για τους στόχους **"?- p(a)", "?- p(b)" και "?- p(X)".**

```
p(X) :- q(X), r(X).
p(X) :-  s(X).
q(X) :-  s(X).
q(X) :-  t(X).
r(X) :-  u(X, Y),  v(Y).
s(b).
t(a).
u(a, b).
u(a, a).
v(a).
```

Πρόγραμμα 5.3: ***Πρόγραμμα με 7 διαδικασίες.***

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.



❑ **Σχήμα 5.5: Τα κιβώτια των διαδικασιών για το πρόγραμμα 5.3.**

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

Παράδειγμα 3: Η ανίχνευση της εκτέλεσης του **Προγράμματος 5.3** μέσω των θυρών του μοντέλου «κιβώτιο της διαδικασίας ή κιβώτιο του Byrd» για το στόχο "**?- p(a)**".

```
p(X) :- q(X), r(X).
p(X) :- s(X).
q(X) :- s(X).
q(X) :- t(X).
r(X) :- u(X,Y), v(Y).
s(b).
t(a).
u(a,b).
u(a,a).
v(a).
```

Πρόγραμμα 5.3: ***Πρόγραμμα με 7 διαδικασίες.***

```
| ?- p(a).
  1   1 Call: p(a) ?
  2         2  Call: q(a) ?
  3               3  Call: s(a) ?
  3               3  Fail: s(a) ?
  4               3 Call: t(a) ?
  4               3 Exit: t(a) ?
  2         2 Exit: q(a) ?
  5         2 Call: r(a) ?
  6               3 Call: u(a,_4262) ? ?
  6               3 Exit: u(a,b) ?
  7               3 Call: v(b) ?
  7               3 Fail: v(b) ? backtracking
  6               3 Redo: u(a,b) ?
  6               3 Exit: u(a,a) ?
  8               3 Call: v(a) ?
  8               3 Exit: v(a) ?
  5         2 Exit: r(a) ? ?
  1    1 Exit: p(a) ?
Yes
```

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

Παράδειγμα 3: Η ανίχνευση της εκτέλεσης του **Προγράμματος 5.3** μέσω των θυρών του μοντέλου «κιβώτιο της διαδικασίας ή κιβώτιο του Byrd» για το στόχο **"?- p(b)".**

```
p(X) :- q(X), r(X).
p(X) :- s(X).
q(X) :- s(X).
q(X) :- t(X).
r(X) :- u(X,Y), v(Y).
s(b).
t(a).
u(a,b).
u(a,a).
v(a).
```

Πρόγραμμα 5.3: ***Πρόγραμμα με 7 διαδικασίες.***

```
| ?- p(b).
 1    1 Call: p(b) ?
 2         2 Call: q(b) ?
 3              3 Call: s(b) ?
 3              3 Exit: s(b) ? ?
 2         2 Exit: q(b) ?
 4         2 Call: r(b) ?
 5              3 Call: u(b,_4265) ?
 5              3 Fail: u(b,_4265) ? backtracking
 4         2 Fail: r(b) ? Continues backtracking
 2         2 Redo: q(b) ?
 6                   3 Call: t(b) ?
 6              3 Fail: t(b) ? backtracking
 2         2 Fail: q(b) ? Continues backtracking
 7         2 Call: s(b) ?
 7         2 Exit: s(b) ?
 1    1 Exit: p(b) ?
Yes
```

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

Παράδειγμα 3: Θα δείξουμε την εκτέλεση του **Προγράμματος 5.3** μέσω των θυρών του μοντέλου «κιβώτιο της διαδικασίας ή κιβώτιο του Byrd» για το στόχο **"?- p(X)".**

```
p(X) :- q(X), r(X).
p(X) :- s(X).
q(X) :- s(X).
q(X) :- t(X).
r(X) :- u(X,Y), v(Y).
s(b).
t(a).
u(a,b).
u(a,a).
v(a).
```

Πρόγραμμα 5.3: ***Πρόγραμμα με 7 διαδικασίες.***

```
| ?- p(X).
 1    1 Call: p(_514) ?
 2         2 Call: q(_514) ?
 3              3 Call: s(_514) ?
 3              3 Exit: s(b) ? ?
 2         2 Exit: q(b) ?
 4         2 Call: r(b) ?
 5              3 Call: u(b,_4288) ?
 5              3 Fail: u(b,_4288) ?  Start backtracking
 4         2 Fail: r(b) ?    Continues backtracking
 2         2 Redo: q(b) ?
 6              3 Call: t(_514) ?
 6              3 Exit: t(a) ?
 2          2 Exit: q(a) ?
 7         2 Call: r(a) ?
 8              3 Call: u(a,_4285) ? ?
 8              3 Exit: u(a,b) ?
 9              3 Call: v(b) ?
 9              3 Fail: v(b) ? Start backtracking
 8              3 Redo: u(a,b) ?
 8              3 Exit: u(a,a) ?
10              3 Call: v(a) ?
10              3 Exit: v(a) ?
 7         2 Exit: r(a) ? ?
 1    1 Exit: p(a) ?
   X = a ?    Yes
```

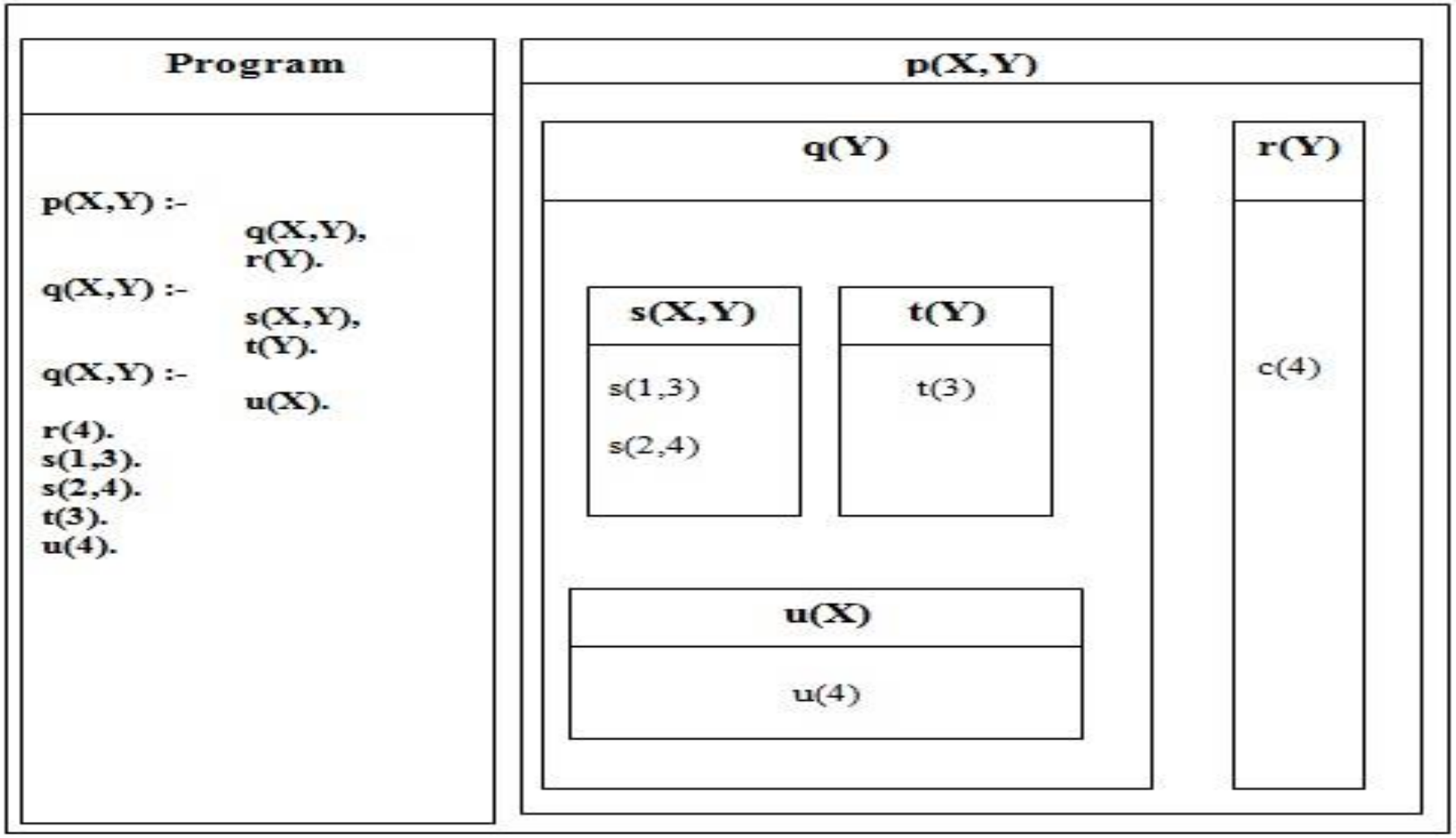
Παράδειγμα 3: Επίδειξη της εκτέλεση του **Προγράμματος 5.4** μέσω των θυρών του μοντέλου «κιβώτιο της διαδικασίας ή κιβώτιο του Byrd» για το στόχο "**?- p(X,Y)**".

```
p(X,Y) :- q(X,Y), r(Y).
q(X,Y) :- s(X,Y), t(Y).
q(X,Y) :- u(X).
r(4).
s(1,3).
s(2,4).
t(3).
u(4).
```

Πρόγραμμα 5.4: ***Πρόγραμμα με 6 διαδικασίες.***

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.



- **Σχήμα 5.6: Τα κιβώτια των διαδικασιών για το πρόγραμμα 5.4.**

5. Έλεγχος Ροής σε Προγράμματα Prolog

5.1. Μοντέλο Περιγραφής Ελέγχου Ροής σε Προγράμματα Prolog.

Παράδειγμα 3: Επίδειξη της εκτέλεση του **Προγράμματος 5.4** μέσω των θυρών του μοντέλου «κιβώτιο διαδικασίας ή κιβώτιο Byrd» για το στόχο "?- p(X,Y)".

```
p(X,Y) :- q(X,Y), r(Y).
q(X,Y) :- s(X,Y), t(Y).
q(X,Y) :- u(X).
r(4).
s(1,3).
s(2,4).
t(3).
u(4).
```

Πρόγραμμα 5.4: ***Πρόγραμμα με 6 διαδικασίες.***

```
| ?- p(X,Y).
1     1 Call: p(_514,_534) ?
2           2 Call: q(_514,_534) ?
3                 3 Call: s(_514,_534) ? ?
3                 3 Exit: s(1,3) ?
4                 3 Call: t(3) ?
4                 3 Exit: t(3) ? ?
2           2 Exit: q(1,3) ?
5           2 Call: r(3) ?
5           2 Fail: r(3) ? Backtracking
2           2 Redo: q(1,3) ?
3                 3 Redo: s(1,3) ?
3                 3 Exit: s(2,4) ?
6                 3 Call: t(4) ?
6                 3 Fail: t(4) ? Backtracking
7                 3 Call: u(_514) ?
7                 3 Exit: u(4) ?
2           2 Exit: q(4,_534) ?
8           2 Call: r(_534) ?
8           2 Exit: r(4) ?
1     1 Exit: p(4,4) ?
X = 4,Y = 4 ?
```

5. Έλεγχος Ροής Προγράμματα Prolog.

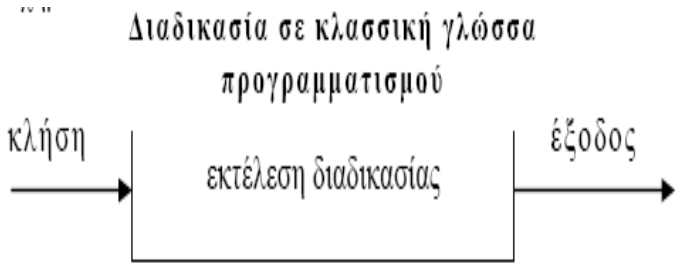
5.1.1. Prolog και οι Κλασσικές Γλώσσες Υψηλού Επιπέδου.

- Τρία χαρακτηριστικά διακρίνουν την Prolog από τις συμβατικές γλώσσες υψηλού επιπέδου.
  - Η **ενοποίηση.**
  - Οι **επαγωγικές δομές δεδομένων**.
  - Οι **οπισθοδρομήσεις κατά την εκτέλεση του προγράμματος.**
- Την **ενοποίηση** της Prolog θα μπορούσαμε να την αντιστοιχίσουμε με το πέρασμα παραμέτρων στις συμβατικές γλώσσες προγραμματισμού.
- Τις **οπισθοδρομήσεις** μπορούμε να τις δούμε σαν γενικευμένες κλήσεις διαδικασιών οι οποίες επιτρέπουν σε κάποιο κανόνα του οποίου κάποια κλήση έχει ολοκληρωθεί να ξανακληθεί.

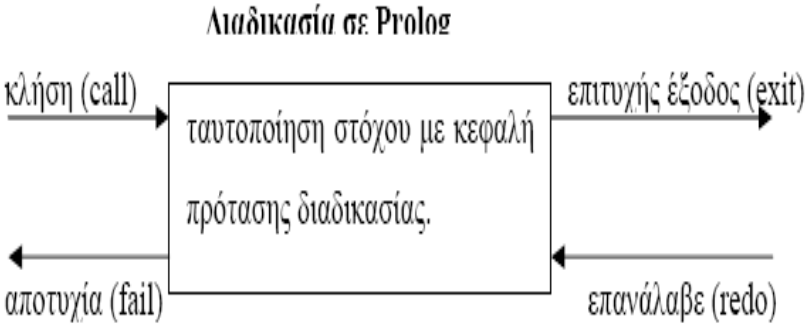
5. Έλεγχος Ροής Προγράμματα Prolog.

5.1.1. Prolog και οι Κλασσικές Γλώσσες Υψηλού Επιπέδου.

- Μια **διαδικασία σε γλώσσα υψηλού επιπέδου ενεργοποιείται με μια κλήση και σταματάει με την έξοδο** (π.χ. μια διαδικασία στην Basic σταματάει με το ‘‘RETURN’’, στην Pascal με το ‘‘end’’ του κυρίως μπλοκ begin…end; της διαδικασίας).
  - **Μετά που θα σταματήσει δεν υπάρχει άλλος τρόπος να ξαναενεργοποιηθεί παρά μόνο με κάποια άλλη κλήση**.
- Στην **Prolog** οι **προτάσεις οι οποίες ορίζουν κάποιο κατηγόρημα συμπεριφέρονται σαν μια διαδικασία**.
  - Μπορούν να κληθούν από ένα στόχο και να εξέλθουν επιτυχώς όπως οι διαδικασίες των συμβατικών γλωσσών προγραμματισμού.
  - **Οι οπισθοδρομήσεις όμως σε μια διαδικασία Prolog προσθέτουν κάποιο άλλο τρόπο εξόδου και εισόδου (κλήσης) σε αυτή**.



Σχήμα 5.5:*Μοντέλο διαδικασίας σε συμβατική γλώσσα*



Σχήμα 5.6:*Μοντέλο διαδικασίας σε Prolog*

Τέλος Διάλεξης

Ευχαριστώ!

Ερωτήσεις;