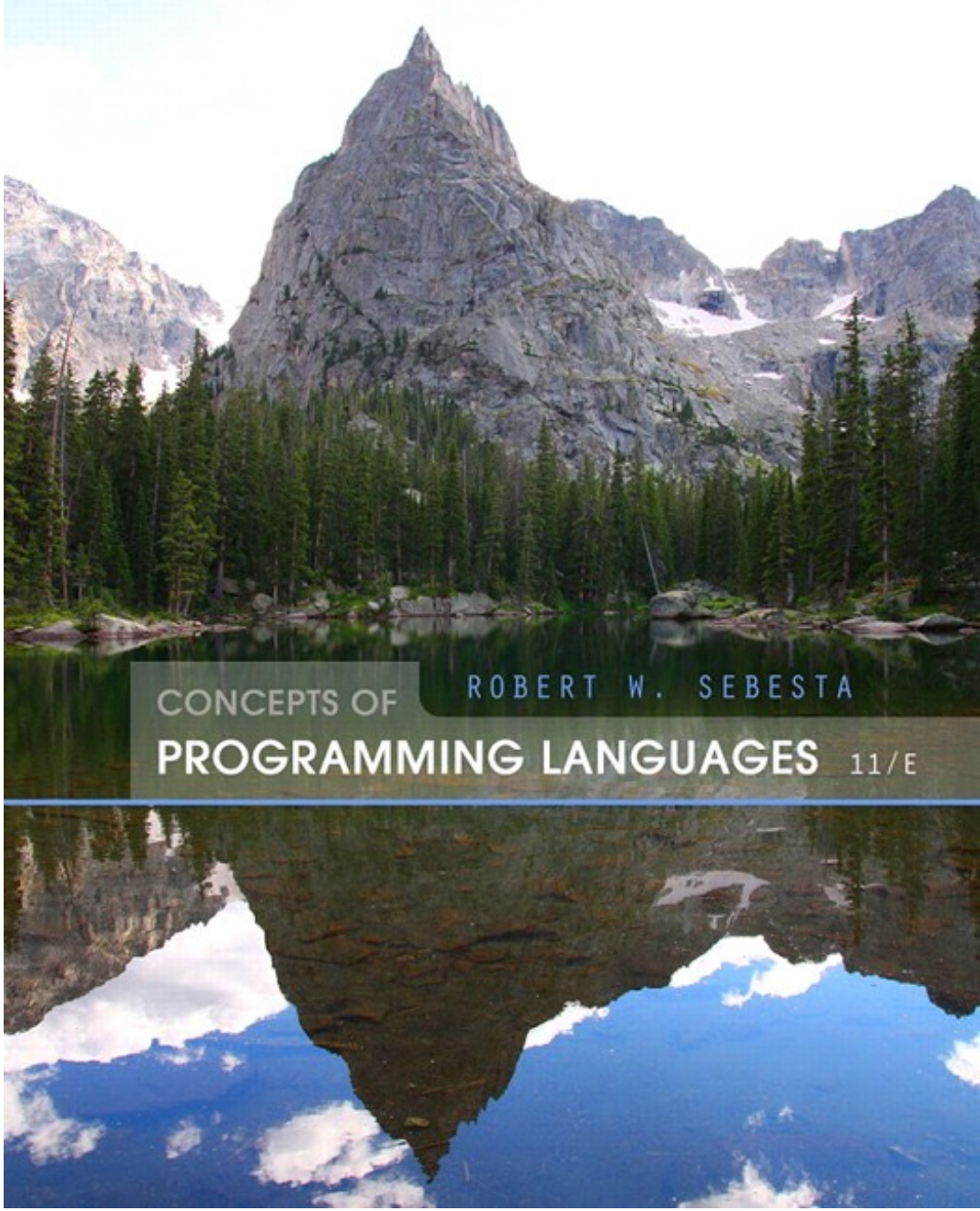
Κεφάλαιο 16

Γλώσσες Λογικού Προγραμματισμού

Γκόγκος Χρήστος
Τμήμα Πληροφορικής και Τηλεπικοινωνιών (Άρτα)
Πανεπιστήμιο Ιωαννίνων



ISBN 0-321-49362-1

Θέματα Κεφαλαίου 16

- Εισαγωγή
- Μια σύντομη εισαγωγή στον Κατηγορηματικό Λογισμό
- Κατηγορηματικός Λογισμός και Απόδειξη Θεωρημάτων
- Μια σύνοψη του Λογικού Προγραμματισμού
- Η προέλευση της Prolog
- Τα βασικά στοιχεία της Prolog
- Αδυναμίες της Prolog
- Εφαρμογές του Λογικού Προγραμματισμού

Εισαγωγή

- Τα προγράμματα στις λογικές γλώσσες εκφράζονται σε μια μορφή συμβολικής λογικής
- Χρησιμοποιείται μια διαδικασία λογικού συμπερασμού (logical inferencing) έτσι ώστε να παραχθούν αποτελέσματα
- Τα προγράμματα είναι *δηλωτικά (declarative)* αντί για *διαδικασιακά (procedural)*:
  - Καθορίζεται μόνο η προδιαγραφή των αποτελεσμάτων (και όχι λεπτομερείς *διαδικασίες* για την παραγωγή τους)

Πρόταση (proposition)

- Μια λογική εντολή που μπορεί να είναι αληθής ή να μην είναι αληθής
  - Αποτελείται από αντικείμενα και σχέσεις αντικειμένων μεταξύ τους

Συμβολική Λογική

- Η λογική μπορεί να χρησιμοποιηθεί για την ικανοποίηση των βασικών απαιτήσεων της τυπικής λογικής:
  - Διατύπωση προτάσεων
  - Διατύπωση σχέσεων μεταξύ προτάσεων
  - Περιγραφή του πως νέες προτάσεις μπορούν να προκύψουν ως συμπεράσματα από άλλες προτάσεις
- Η ιδιαίτερη μορφή της συμβολικής λογικής που χρησιμοποιείται στον λογικό προγραμματισμό ονομάζεται *κατηγορηματικός λογισμός (predicate calculus)*

Αναπαράσταση Αντικειμένων

- Τα αντικείμενα στις προτάσεις αναπαρίστανται με απλούς όρους: είτε με σταθερές είτε με μεταβλητές
- *Σταθερά*: ένα σύμβολο που αναπαριστά ένα αντικείμενο
- *Μεταβλητή*: ένα σύμβολο που μπορεί να αναπαριστά διαφορετικά αντικείμενα σε διαφορετικές χρονικές στιγμές
  - Οι μεταβλητές στις γλώσσες λογικού προγραμματισμού είναι διαφορετικές από τις μεταβλητές στις προστακτικές γλώσσες

Σύνθετοι Όροι

- *Οι ατομικές προτάσεις* αποτελούνται από σύνθετους όρους
- *Σύνθετος όρος*: Ένα στοιχείο μιας μαθηματικής σχέσης, που γράφεται όπως μια μαθηματική συνάρτηση
  - Οι μαθηματικές συναρτήσεις είναι αντιστοιχήσεις
  - Μπορεί να γραφεί ως ένας πίνακας

Τμήματα ενός Σύνθετου Όρου

- Ένας σύνθετος όρος αποτελείται από δύο μέρη
  - Έναν functor (συναρτητή): σύμβολο συνάρτησης που αποδίδει όνομα στη σχέση
  - Μια διατεταγμένη λίστα παραμέτρων (πλειάδα)
- Παραδείγματα:

```
student(jon)
like(seth, osx)
like(nick, windows)
like(jim, linux)
```

Μορφές Προτάσεων

- Οι προτάσεις μπορούν να διατυπωθούν σε δύο μορφές:
  - *Γεγονός*: πρόταση για την οποία γίνεται η υπόθεση ότι είναι αληθής
  - *Ερωτήματα*: Η αλήθεια της πρότασης πρόκειται να προσδιοριστεί
- Σύνθετες προτάσεις:
  - Αποτελούνται από δύο ή περισσότερες ατομικές προτάσεις
  - Οι προτάσεις συνδέονται με τελεστές

Λογικοί Τελεστές

Όνομα	Σύμβολο	Παράδειγμα	Νόημα
Άρνηση	$\neg$	$\neg$ a	όχι a
Σύζευξη	$\cap$	a $\cap$ b	a και b
Διάζευξη	$\cup$	a $\cup$ b	a ή b
Ισοδυναμία	$\equiv$	a $\equiv$ b	το a ισοδυναμεί με το b
Συνεπαγωγή	$\supset$ $\subset$	a $\supset$ b a $\subset$ b	Το a συνεπάγεται το b Το b συνεπάγεται το a

Ποσοδείκτες

Όνομα	Παράδειγμα	Νόημα
Καθολικός	$\forall X.P$	Για κάθε X, το P είναι αληθές
Υπαρξιακός	$\exists X.P$	Υπάρχει τιμή του X τέτοια ώστε το P να είναι αληθές

Προτασιακή Μορφή

- Υπάρχουν πολλοί τρόποι διατύπωσης του ίδιου πράγματος
- Χρήση μιας τυπικής μορφής για τις προτάσεις
- *Προτασιακή μορφή*:
  - $B_1 \cup B_2 \cup \ldots \cup B_n \subset A_1 \cap A_2 \cap \ldots \cap A_m$
  - Σημαίνει ότι αν όλα τα A είναι αληθή, τότε τουλάχιστον ένα από τα B θα είναι αληθές
- *Προηγούμενο (antecedent)*: δεξί μέρος
- *Επόμενο (consequent)*: αριστερό μέρος

Κατηγορηματικός Λογισμός και Απόδειξη Θεωρημάτων

- Μια χρήση των προτάσεων της λογικής είναι η ανακάλυψη νέων θεωρημάτων τα οποία μπορούν να εξαχθούν από γνωστά αξιώματα και θεωρήματα
- *Ανάλυση (resolution)*: Αρχή συμπερασμού που επιτρέπει να υπολογιστούν από δεδομένες προτάσεις, νέες προτάσεις που προκύπτουν λογικά

Ανάλυση

- *Ταυτοποίηση (unification)*: εύρεση τιμών για τις μεταβλητές προτάσεων που επιτρέπουν στη διεργασία αντιστοίχισης να είναι επιτυχής
- *Αρχικοποίηση (instantiation)*: ανάθεση προσωρινών τιμών σε μεταβλητές έτσι ώστε η ταυτοποίηση να επιτύχει
- Μετά την αρχικοποίηση μιας μεταβλητής σε μια τιμή, εάν η αντιστοίχιση αποτύχει, μπορεί να χρειαστεί οπισθοδρόμηση (*backtrack)* και αρχικοποίηση σε διαφορετική τιμή

Απόδειξη με Αντίφαση (εις άτοπο απαγωγή)

- *Υποθέσεις*: ένα σύνολο από σχετικές με την κατάσταση προτάσεις
- *Στόχος*: Άρνηση του θεωρήματος που διατυπώνεται ως μια πρόταση
- Το θεώρημα αποδεικνύεται εντοπίζοντας κάποια ασυνέπεια στην άρνησή του

Απόδειξη θεωρημάτων

- Βάση του λογικού προγραμματισμού
- Οι προτάσεις που χρησιμοποιούνται στην ανάλυση, πρέπει να είναι σε περιορισμένη μορφή
- *Προτάσεις Horn* – μπορούν να έχουν μόνο δύο μορφές
  - *Με κεφαλή*: απλή ατομική πρόταση στο αριστερό μέρος $B \subset A_1 \cap A_2 \cap \ldots \cap A_m$
  - *Χωρίς κεφαλή*: άδειο αριστερό μέρος (χρησιμοποιείται για να δηλώσει γεγονότα) $A_1 \cap A_2 \cap \ldots \cap A_m$
- Οι περισσότερες προτάσεις μπορούν να διατυπωθούν ως προτάσεις Horn

Επισκόπηση Λογικού Προγραμματισμού

- Δηλωτική σημασιολογία
  - Υπάρχει ένας απλός τρόπος για να καθοριστεί το νόημα της κάθε πρότασης
  - Είναι απλούστερη από τη σημασιολογία των προστακτικών γλωσσών
- Ο προγραμματισμός είναι μη-διαδικασιακός
  - Τα προγράμματα δεν προσδιορίζουν το πως κάτι υπολογίζεται, αλλά τη μορφή που θα πρέπει να έχει το αποτέλεσμα

Παράδειγμα: Ταξινόμηση Λίστας

- Περιγραφή των χαρακτηριστικών μιας ταξινομημένης λίστας, όχι της διαδικασίας επαναδιάταξης των στοιχείων της λίστας προκειμένου να ταξινομηθεί

sort(old_list, new_list) $\subset$ permute (old_list, new_list) $\cap$ sorted(new_list)

sorted (list) $\subset \forall_j$ έτσι ώστε $1 \leq j < n$, list(j) $\leq$ list (j+1)

Η προέλευση της Prolog

- Πανεπιστήμιο Aix-Marseille (Colmerauer & Roussel)
  - Επεξεργασία φυσικής γλώσσας
- Πανεπιστήμιο Edinburgh (Kowalski)
  - Αυτόματη απόδειξη θεωρημάτων

Όροι (terms)

- Στη συνέχεια χρησιμοποιείται η διάλεκτος του Εδιμβούργου για την Prolog
- *Όρος*: μια σταθερά, μεταβλητή, ή δομή
- *Σταθερά*: Ένα άτομο (atom) ή ένας ακέραιος
- *Άτομο*: μια συμβολική τιμή της Prolog
- Ένα άτομο είναι είτε:
  - Μια συμβολοσειρά γραμμάτων, ψηφίων, και κάτω παυλών που ξεκινούν με κάποιο πεζό γράμμα
  - Ένα λεκτικό με εκτυπώσιμους ASCII χαρακτήρες μέσα σε εισαγωγικά

Όροι: Μεταβλητές και δομές

- *Μεταβλητή*: οποιοδήποτε λεκτικό γραμμάτων, ψηφίων, και κάτω παυλών που ξεκινά με κάτω παύλα ή με πρώτο χαρακτήρα ένα γράμμα σε κεφαλαία
- *Αρχικοποίηση*: πρόσδεση μιας μεταβλητής σε μια τιμή
  - Διαρκεί όσο απαιτείται για την ικανοποίηση ενός πλήρους στόχου
- *Δομή*: αναπαριστά ατομικές προτάσεις της μορφής:

  functor (*parameter list*)

Προτάσεις Γεγονότων

- Χρησιμοποιούνται για τις υποθέσεις
- Πρόκειται για προτάσεις Horn χωρίς κεφαλές

```
female(shelley).
male(bill).
father(bill, jake).
```

Προτάσεις κανόνων

- Χρησιμοποιούνται για τις υποθέσεις
- Είναι προτάσεις Horn με κεφαλές
- Δεξί μέρος: *προηγούμενο* (*if* τμήμα)
  - Μπορεί να είναι ένας απλός όρος ή μια σύζευξη όρων
- Αριστερό μέρος: *επόμενο* (***then*** τμήμα)
  - Πρέπει να είναι ένας απλός όρος
- *Σύζευξη (conjunction)*: πολλαπλοί όροι που διαχωρίζονται με τελεστές AND (δεν χρησιμοποιούνται ειδικοί τελεστές, αλλά υπονοούνται)

Παραδείγματα Κανόνων

```
ancestor(mary, shelley):- mother(mary, shelley).
```

- Μπορούν να χρησιμοποιούνται μεταβλητές (*καθολικά αντικείμενα*) για να γενικεύσουν το νόημα:

```
parent(X,Y):- mother(X,Y).
parent(X,Y):- father(X,Y).
grandparent(X,Z):- parent(X,Y), parent(Y,Z).
```

Προτάσεις Στόχοι

- Για την απόδειξη θεωρημάτων, το θεώρημα είναι στη μορφή πρότασης που επιθυμούμε το σύστημα να αποδείξει ότι ισχύει ή να αποδείξει ότι δεν ισχύει – *πρόταση στόχος*
- Ίδια μορφή με τις προτάσεις Horn χωρίς κεφαλή

```
man(fred).
```

- Συζεύξεις προτάσεων καθώς και προτάσεις που περιέχουν μεταβλητές είναι επίσης έγκυροι στόχοι

```
father(X, mike).
```

Η διαδικασία συμπερασμού της Prolog

- Τα ερωτήματα ονομάζονται στόχοι
- Αν ένας στόχος είναι μια σύνθετη πρόταση, κάθε ένα από τα γεγονότα είναι ένας υποστόχος
- Για να αποδειχθεί ότι ένας στόχος είναι αληθής, θα πρέπει να βρεθεί μια αλυσίδα από κανόνες συμπερασμού και/ή γεγονότα.
- Για το στόχο Q:

  P_2 :- P_1

  P_3 :- P_2

  ...

  Q :- P_n

- Η διαδικασία απόδειξης ενός υποστόχου ονομάζεται ταίριασμα, ικανοποίηση, ή ανάλυση

Προσεγγίσεις

- *Αντιστοίχιση* (matching) είναι η διαδικασία απόδειξης μιας πρότασης
- Η απόδειξη ενός υποστόχου, ονομάζεται *ικανοποίηση* του υποστόχου
- *Ανάλυση από κάτω προς τα πάνω (bottom-up resolution), αλυσίδωση προς τα εμπρός (forward chaining)*
  - Ξεκινά με τα γεγονότα και τους κανόνες στη βάση γνώσης και επιχειρεί να εντοπίσει μια ακολουθία που οδηγεί στο στόχο
  - Λειτουργεί καλύτερα σε μεγάλα σύνολα πιθανών ορθών απαντήσεων
- *Ανάλυση από πάνω προς τα κάτω (top-down resolution), αλυσίδωση προς τα πίσω (backward chaining)*
  - Ξεκινά με το στόχο και επιχειρεί να εντοπίσει μια ακολουθία που οδηγεί σε ένα σύνολο από γεγονότα στη βάση γνώσης
  - Λειτουργεί καλύτερα σε μικρά σύνολα πιθανών ορθών απαντήσεων
- Οι υλοποιήσεις της Prolog χρησιμοποιούν backward chaining

Στρατηγικές Υποστόχων

- Όταν ένας στόχος έχει περισσότερους από έναν υποστόχους, μπορεί να χρησιμοποιηθεί είτε:
  - Αναζήτηση πρώτα κατά βάθος: εντοπισμός μιας πλήρους απόδειξης για τον πρώτο υποστόχο πριν ξεκινήσει η εργασία για τους άλλους
  - Αναζήτηση πρώτα κατά πλάτος: εργασία σε όλους τους υποστόχους παράλληλα
- Η Prolog χρησιμοποιεί αναζήτηση πρώτα κατά βάθος
  - Απαιτεί λιγότερους υπολογιστικούς πόρους

Οπισθοδρόμηση (backtracking)

- Όταν ένας στόχος έχει πολλαπλούς υποστόχους, αν αποτύχει η απόδειξη για έναν υποστόχο, πραγματοποιείται επανεξέταση του προηγούμενου υποστόχου έτσι ώστε να βρεθεί εναλλακτική λύση: *οπισθοδρόμηση*
- Εκκίνηση αναζήτησης στο σημείο που έμεινε η προηγούμενη αναζήτηση
- Μπορεί να απαιτήσει πολύ χρόνο και χώρο στη μνήμη, διότι μπορεί να εντοπίσει όλες τις πιθανές αποδείξεις για κάθε υποστόχο

Απλή αριθμητική

- Η Prolog υποστηρίζει ακέραιες μεταβλητές και αριθμητικές πράξεις σε ακεραίους
- τελεστής `is` : λαμβάνει μια αριθμητική έκφραση ως δεξί τελεστέο και μια μεταβλητή ως αριστερό τελεστέο

```
A is B / 17 + C
```

- Δεν είναι το ίδιο με την εντολή ανάθεσης!
  - Το ακόλουθο είναι μη έγκυρο:

```
Sum is Sum + Number.
```

Παράδειγμα

```
speed(ford,100).
speed(chevy,105).
speed(dodge,95).
speed(volvo,80).
time(ford,20).
time(chevy,21).
time(dodge,24).
time(volvo,24).
distance(X,Y) :-   speed(X,Speed),
                   time(X,Time),
                   Y is Speed * Time.
```

```
?- distance(chevy, Chevy_Distance).
Chevy_Distance = 2205.
```

Ένα ερώτημα: `distance(chevy, Chevy_Distance).`

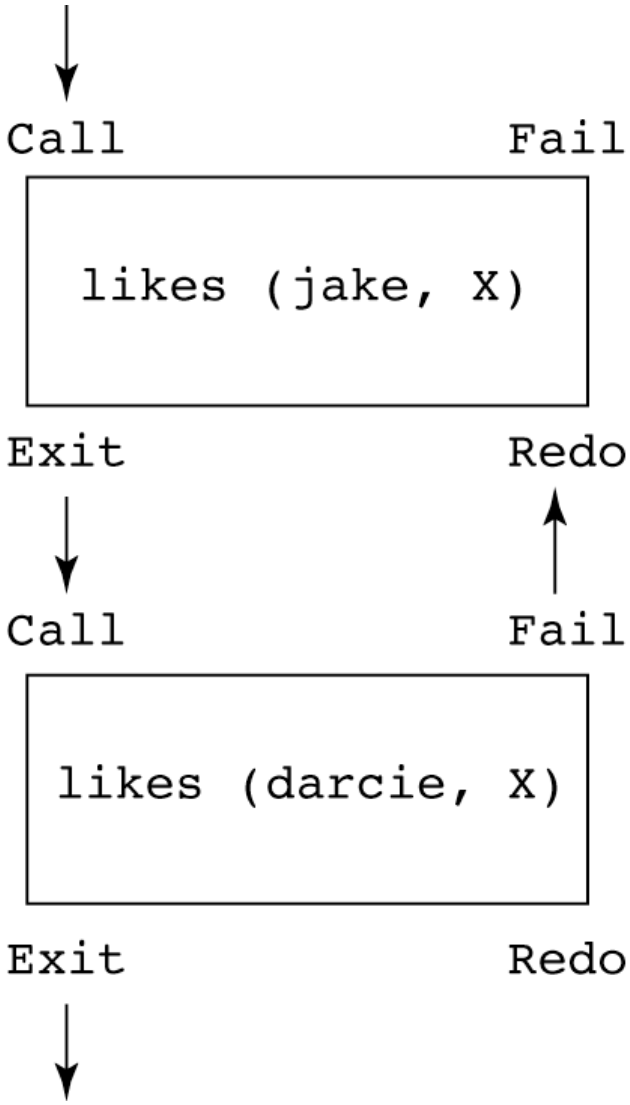
Ιχνηλάτηση (tracing)

- Ενσωματωμένη δομή που αναπαριστά αρχικοποιήσεις σε κάθε βήμα
- *Μοντέλο ιχνηλάτησης* εκτέλεσης–τέσσερα γεγονότα:
  - *Call* (εκκίνηση προσπάθειας ικανοποίησης στόχου)
  - *Exit* (όταν ο στόχος έχει ικανοποιηθεί)
  - *Redo* (όταν συμβαίνει οπισθοδρόμηση)
  - *Fail* (όταν ο στόχος αποτυγχάνει)

Παράδειγμα

```
likes(jake, chocolate).
likes(jake, apricots).
likes(darcie, licorice).
likes(darcie, apricots).

trace.
?- likes(jake, X), likes(darcie, X).
(1)  1 Call: likes(jake, _0)?
(1)  1 Exit: likes(jake, chocolate)
(2)  1 Call: likes(darcie, chocolate)?
(2)  1 Fail: likes(darcie, chocolate)
(1)  1 Redo: likes(jake, _0)?
(1)  1 Exit: likes(jake, apricots)
(3)  1 Call: likes(darcie, apricots)?
(3)  1 Exit: likes(darcie, apricots)
X = apricots
```



Δομές Λίστας

- Μια άλλη βασική δομή δεδομένων (επιπλέον των ατομικών προτάσεων που έχουμε ήδη δει): λίστα
- *Μια λίστα είναι μια ακολουθία οποιουδήποτε πλήθους στοιχείων*
- Τα στοιχεία μπορούν να είναι άτομα, ατομικές προτάσεις, ή άλλοι όροι (συμπεριλαμβανομένων άλλων λιστών)

```
[apple, prune, grape, kumquat]
[]          (άδεια λίστα)
[X | Y]     (κεφαλή X και ουρά Y)
```

Παράδειγμα append

```
append([], List, List).
append([Head | List_1], List_2, [Head | List_3]) :-
            append (List_1, List_2, List_3).
```

Η πρώτη πρόταση (βασική περίπτωση της αναδρομής) σημαίνει ότι αν η πρώτη παράμετρος του κατηγορήματος append/3 είναι η κενή λίστα τότε η τρίτη παράμετρος θα είναι ίση με τη δεύτερη παράμετρο

Η δεύτερη πρόταση (αναδρομική περίπτωση) σημαίνει ότι αν η πρώτη παράμετρος είναι μια λίστα με πρώτο στοιχείο το Head, τότε η τρίτη παράμετρος θα είναι μια λίστα με πρώτο στοιχείο το Head που θα ακολουθείται από μια λίστα (την List_3) που θα προκύπτει ως προσάρτηση της List_2 στη List_1

Περισσότερα παραδείγματα

```
reverse([], []).
reverse([Head | Tail], List) :-
  reverse(Tail, Result),
  append (Result, [Head], List).


member(Element, [Element | _]).
member(Element, [_ | List]) :-
                        member(Element, List).
```

Ο χαρακτήρας της κάτω παύλας _ σημαίνει ανώνυμη μεταβλητή — δηλαδή ότι δεν ενδιαφερόμαστε για το ποια αρχικοποίηση θα συμβεί στη συγκεκριμένη μεταβλητή κατά την ενοποίηση

Μειονεκτήματα της Prolog

- Έλεγχος της σειράς με την οποία πραγματοποιείται η ανάλυση
  - Σε ένα περιβάλλον καθαρού λογικού προγραμματισμού, η σειρά των επιχειρούμενων αντιστοιχήσεων είναι μη-ντετερμινιστική και όλες οι αντιστοιχήσεις επιχειρούνται ταυτόχρονα
- Η υπόθεση του κλειστού-κόσμου
  - Η μόνη γνώση είναι αυτή που περιέχεται στη βάση γνώσης
- Το πρόβλημα της άρνησης
  - Για οτιδήποτε δεν υπάρχει στη βάση γνώσης γίνεται η υπόθεση ότι είναι ψευδές
- Εσωτερικοί (intrinsic) περιορισμοί
  - Για παράδειγμα, είναι εύκολο να διατυπωθεί η διαδικασία ταξινόμησης στη λογική, αλλά δύσκολο να επιτευχθεί στην πράξη — δεν υπάρχει κάποια έννοια λογικής που να σχετίζεται πρωτογενώς με την ταξινόμηση

Εφαρμογές Λογικού Προγραμματισμού

- Σχεσιακά συστήματα διαχείρισης βάσεων δεδομένων
- Έμπειρα συστήματα
- Επεξεργασία φυσικής γλώσσας

Σύνοψη

- Η συμβολική λογική αποτελεί τη βάση του λογικού προγραμματισμού
- Τα προγράμματα σε λογικό προγραμματισμό θα πρέπει να είναι μη-διαδικαστικά
- Οι εντολές Prolog είναι γεγονότα, κανόνες, ή στόχοι
- Η ανάλυση (resolution) είναι η κύρια δραστηριότητα του διερμηνευτή της Prolog
- Αν και υπάρχουν μειονεκτήματα στην τρέχουσα κατάσταση του λογικού προγραμματισμού, ο λογικός προγραμματισμός χρησιμοποιείται σε διάφορα πεδία