Τεχνητή νοημοσύνη

Φροντιστήριο 6

Ασκήσεις μελέτης της 11ης και 12ης διάλεξης

Κανόνας απαλοιφής ∀

Αν η ΒΓ περιέχει:

$$\forall x\ ((\text{King}(x) \land \text{Greedy}(x)) \Rightarrow \text{Evil}(x))$$

Μπορούμε να προσθέσουμε στην ΒΓ:

$$((\text{King}(\text{John}) \land \text{Greedy}(\text{John})) \Rightarrow \text{Evil}(\text{John}))$$

Μπορούμε να αντικαταστήσουμε τη μεταβλητή (x) με κάποιον όρο (πχ John, FatherOf(John)).

Αρκεί ο όρος να μην περιέχει μεταβλητές.

Κανόνας απαλοιφής ∃

Αν η ΒΓ περιέχει:

$$\exists x ((Crown(x) \land OnHead(x, John))$$

Μπορούμε να τον αντικαταστήσουμε με:

$$\exists x ((Crown(C) \land OnHead(C, John))$$

Το C πρέπει να είναι σταθερά («σταθερά Skolem») που δε χρησιμοποιείται πουθενά αλλού στο σύνολο τύπων μας.

Κανόνας απαλοιφής ∃

Αν η ΒΓ περιέχει:

$$\exists x ((Crown(x) \land OnHead(x, John))$$

Μπορεί αυτό να αληθεύει

Μπορούμε να τον αντικαταστήσουμε με:

$$\exists x ((Crown(C) \land OnHead(C, John))$$

Σε κάποια μοντέλα μπορεί να μην αληθεύει

Ο νέος τύπος δεν είναι ταυτολογικά ισοδύναμος με τον αρχικό

Ο αρχικός τύπος είναι ικανοποιήσιμος ανν ο νέος είναι ικανοποιήσιμος

Μετατροπή τύπου ΠΚΛ σε CNF

$$\forall x\ (\forall y\ \text{Animal}(y) \Rightarrow \text{Loves}(x, y)) \Rightarrow \exists z\ \text{Loves}(z, x))$$

Βήμα 1: Απαλοιφή ⇒ και ⇔

$$\forall x\ (\neg\ \forall y\ (\neg\ \text{Animal}(y) \vee \text{Loves}(x, y))\ \vee \exists z\ \text{Loves}(z, x))$$

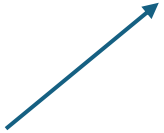
Βήμα 2: Μεταφορά ¬ στο εσωτερικό

$$\forall x\ (\exists y\ \neg\ (\neg\ \text{Animal}(y) \vee \text{Loves}(x, y))\ \vee \exists z\ \text{Loves}(z, x))$$

$$\forall x\ (\exists y\ (\text{Animal}(y) \wedge \neg\ \text{Loves}(x, y))\ \vee \exists z\ \text{Loves}(z, x))$$

Μετατροπή τύπου ΠΚΛ σε CNF

$$\forall x \, (\exists y \, (Animal(y) \land \neg Loves(x, y)) \lor \exists z \, Loves(z, x))$$

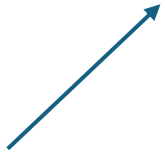


Εδώ μπορούμε να έχουμε διαφορετικό y για κάθε x.

Βήμα 3: Απαλοιφή των ∃

Αν βάζαμε σταθερές Skolem θα είχαμε:

$$\forall x \, ((Animal(A) \land \neg Loves(x, A)) \lor Loves(B, x))$$



Πρόβλημα: εδώ έχουμε το ίδιο Α για όλα τα x.

Μετατροπή τύπου ΠΚΛ σε CNF

$\forall x\ (\exists y\ (Animal(y) \wedge \neg Loves(x, y)) \vee \exists z\ Loves(z, x))$

Βήμα 3: Απαλοιφή των ∃

Χρησιμοποιούμε συναρτήσεις Skolem:

$\forall x\ ((Animal(F(x)) \wedge \neg Loves(x, F(x))) \vee Loves(G(x), x))$

Μετατροπή τύπου ΠΚΛ σε CNF

$$\forall x ((Animal(F(x)) \land \neg Loves(x, F(x))) \lor Loves(G(x), x))$$

Βήμα 4: Απαλοιφή ∀:

$$(Animal(F(x)) \land \neg Loves(x, F(x))) \lor Loves(G(x), x)$$

Βήμα 5: Επιμερισμός των Λ και V:

$$(Animal(F(x)) \lor Loves(G(x), x)) \land$$
$$(\neg Loves(x, F(x))) \lor Loves(G(x), x))$$

Άσκηση 11.1(α)

Μετατρέψτε σε κανονική συζευκτική μορφή (CNF) τους τύπους που γράψατε για τις προτάσεις (iv) και (vi) στην άσκηση 9.2. Δείξτε αναλυτικά τα βήματα της μετατροπής.

(iv) $\exists x\ (Course(x) \wedge Passed(John, x))$

(vi) $\forall x\ ((Student(x) \wedge \exists y\ (Course(y) \wedge Passed(x, y))) \Rightarrow Clever(x))$

Άσκηση 11.1(α)

Μετατρέψτε σε κανονική συζευκτική μορφή (CNF) τους τύπους που γράψατε για τις προτάσεις (iv) και (vi) στην άσκηση 9.2. Δείξτε αναλυτικά τα βήματα της μετατροπής.

(iv) $\exists x\,(\text{Course}(x) \wedge \text{Passed}(\text{John}, x))$

(iv) $(\text{Course}(C) \wedge \text{Passed}(\text{John}, C))$

Άσκηση 11.1(α)

Μετατρέψτε σε κανονική συζευκτική μορφή (CNF) τους τύπους που γράψατε για τις προτάσεις (iv) και (vi) στην άσκηση 9.2. Δείξτε αναλυτικά τα βήματα της μετατροπής.

(vi) $\forall x\ ((Student(x) \land \exists y\ (Course(y) \land Passed(x, y))) \Rightarrow Clever(x))$

$\forall x\ (\neg\ (Student(x) \land \exists y\ (Course(y) \land Passed(x, y))) \lor Clever(x))$

$\forall x\ (\neg\ Student(x) \lor \neg\ \exists y\ (Course(y) \land Passed(x, y))) \lor Clever(x))$

$\forall x\ (\neg\ Student(x) \lor \forall\ y\ \neg\ (Course(y) \land Passed(x, y))) \lor Clever(x))$

$\forall x\ (\neg\ Student(x) \lor \forall\ y\ (\neg\ Course(y) \lor \neg\ Passed(x, y))) \lor Clever(x))$

$(\neg\ Student(x) \lor \neg\ Course(y) \lor \neg\ Passed(x, y) \lor Clever(x))$

Ενοποίηση

UNIFY(π1, π2) = θ σημαίνει ότι υπάρχει ένα σύνολο αντικαταστάσεων θ με SUBST(θ, π1) = SUBST(θ, π2)

Πχ. UNIFY(Knows(John, x), Knows(y, z)) = ?

θ1 = {y/John, x/z}

Αντικαθιστούμε όπου y το John και όπου x το z

Γενικότερος ενοποιητής από τον θ2

θ2 = {y/John, x/John, z/John}

Αντικαθιστούμε όπου x, y, z το John

Κανόνας ανάλυσης για ΠΚΛ

- Αν UNIFY(l_i, ¬ m_j) = θ, τότε:

$(l_1 \vee \ldots \vee l_k)$, $(m_1 \vee \ldots \vee m_n)$ ├ SUBST(θ, $(l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \ldots m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n)$

Έχουμε αφαιρέσει τα l_i και m_j

Στην ΠΚΛ δεν χρειάζεται τα l_i και ¬ m_j να είναι ίδια, **αρκεί να ενοποιούνται**

Πχ [Animal(F(x)) ∨ Loves(G(x), x)], [¬Loves(u, v) ∨ ¬Kills(u, v)]
├ [Animal(F(x)) ∨ ¬Kills(G(x), x)]

με θ = {u/G(x), v/x}.

Άσκηση 11.1(β)

Σχεδιάστε δέντρο απόδειξης που να δείχνει με απαγωγή σε άτοπο χρησιμοποιώντας μόνο τον κανόνα της ανάλυσης (resolution) ότι από τις προτάσεις (i), (iv) και (vi) της άσκησης 9.2 μπορούμε να συμπεράνουμε πως ο Γιάννης είναι έξυπνος.

Student(John)

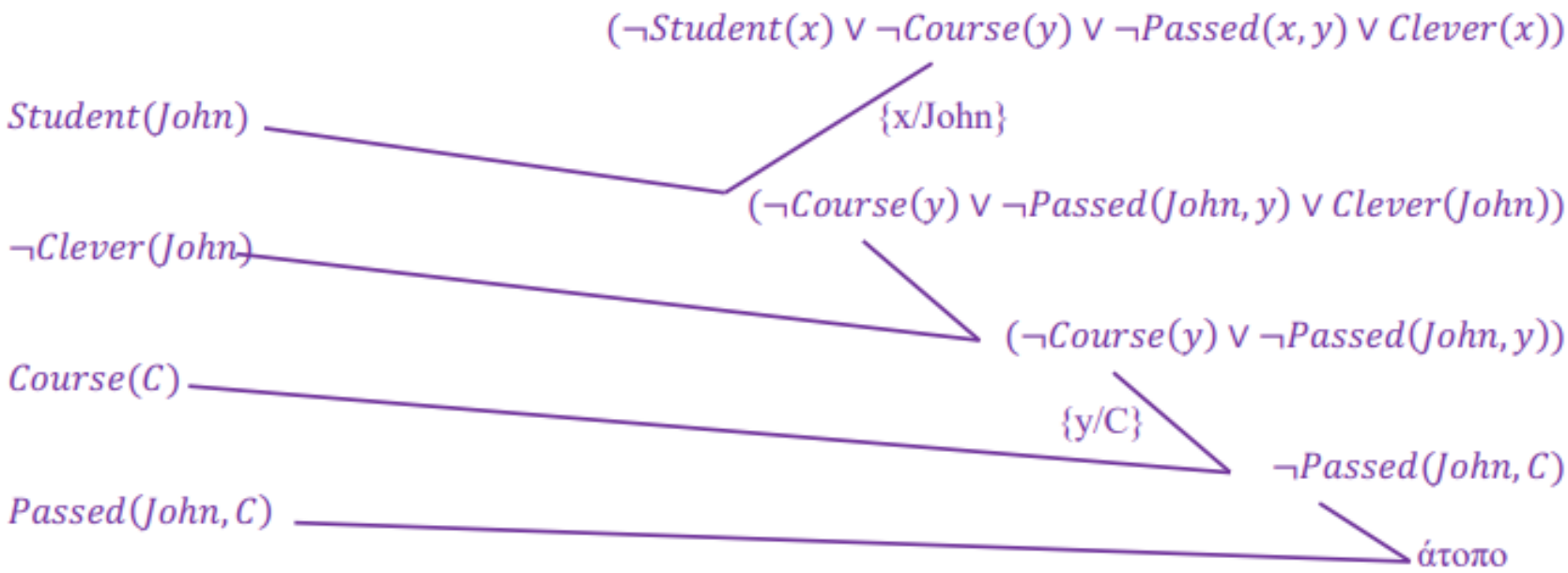
(Course(C) ∧ Passed(John, C))

(¬ Student(x) ∨ ¬ Course(y) ∨ ¬ Passed(x, y) ∨ Clever(x))

Κανόνας ανάλυσης: Αν UNIFY(l_i, ¬ m_j) = θ, τότε:

(l_1 ∨ … ∨ l_k) , (m_1 ∨ … ∨ m_n) ├ SUBST(θ, (l_1 ∨ … ∨ l_{i-1} ∨ l_{i+1} ∨ … ∨ l_k ∨ m_1 ∨ … … m_{j-1} ∨ m_{j+1} ∨ … ∨ m_n)

Άσκηση 11.1(β)


$(\neg Student(x) \vee \neg Course(y) \vee \neg Passed(x,y) \vee Clever(x))$
$Student(John)$
{x/John}
$(\neg Course(y) \vee \neg Passed(John,y) \vee Clever(John))$
$\neg Clever(John)$
$(\neg Course(y) \vee \neg Passed(John,y))$
$Course(C)$
{y/C}
$\neg Passed(John,C)$
$Passed(John,C)$
άτοπο

Άσκηση 11.2(α)

Μετατρέψτε τους τύπους (v), (vi) και (vii) της άσκησης 9.3 σε κανονική συζευκτική μορφή (CNF). Δείξτε αναλυτικά τα βήματα της κάθε μετατροπής.

$\exists x\, \exists y\, (Dog(x) \land Cat(y) \land Bite(x, y))$

$\forall y\, (Cat(y) \Rightarrow \neg\, Likes(y, Suzos)$

$\forall x\, ((Dog(x) \land \exists y\, (Cat(y) \land Bite(x, y))) \Rightarrow \forall z\, (Cat(z) \Rightarrow \neg\, Likes(z, x)))$

Άσκηση 11.2(α)

Μετατρέψτε τους τύπους (v), (vi) και (vii) της άσκησης 9.3 σε κανονική συζευκτική μορφή (CNF). Δείξτε αναλυτικά τα βήματα της κάθε μετατροπής.

$\exists x \exists y (\text{Dog}(x) \wedge \text{Cat}(y) \wedge \text{Bite}(x, y))$

$(\text{Dog}(C_1) \wedge \text{Cat}(C_2) \wedge \text{Bite}(C_1, C_2))$

Άσκηση 11.2(α)

Μετατρέψτε τους τύπους (v), (vi) και (vii) της άσκησης 9.3 σε κανονική συζευκτική μορφή (CNF). Δείξτε αναλυτικά τα βήματα της κάθε μετατροπής.

$\forall y\ (\text{Cat}(y) \Rightarrow \neg\ \text{Likes}(y, \text{Suzos})$

$\forall y\ (\neg\ \text{Cat}(y) \vee \neg\ \text{Likes}(y, \text{Suzos}))$

$(\neg\ \text{Cat}(x_1) \vee \neg\ \text{Likes}(x_1, \text{Suzos}))$

Άσκηση 11.2(α)

Μετατρέψτε τους τύπους (v), (vi) και (vii) της άσκησης 9.3 σε κανονική συζευκτική μορφή (CNF). Δείξτε αναλυτικά τα βήματα της κάθε μετατροπής.

$\forall x\ ((\text{Dog}(x) \wedge \exists y\ (\text{Cat}(y) \wedge \text{Bite}(x, y))) \Rightarrow \forall z\ (\text{Cat}(z) \Rightarrow \neg\ \text{Likes}(z, x)))$

$\forall x\ (\neg(\text{Dog}(x) \wedge \exists y\ (\text{Cat}(y) \wedge \text{Bite}(x, y))) \vee \forall z\ (\neg\ \text{Cat}(z) \vee \neg\ \text{Likes}(z, x)))$

$\forall x\ (\ (\neg\ \text{Dog}(x) \vee \neg\ \exists y\ (\text{Cat}(y) \wedge \text{Bite}(x, y))) \vee \forall z\ (\neg\ \text{Cat}(z) \vee \neg\ \text{Likes}(z, x)))$

$\forall x\ (\ (\neg\ \text{Dog}(x) \vee \forall\ y\ \neg\ (\text{Cat}(y) \wedge \text{Bite}(x, y))) \vee \forall z\ (\neg\ \text{Cat}(z) \vee \neg\ \text{Likes}(z, x)))$

$\forall x\ (\ (\neg\ \text{Dog}(x) \vee \forall\ y\ (\neg\ \text{Cat}(y) \vee \neg\ \text{Bite}(x, y))) \vee \forall z\ (\neg\ \text{Cat}(z) \vee \neg\ \text{Likes}(z, x)))$

$(\neg\ \text{Dog}(x_2) \vee \neg\ \text{Cat}(x_3) \vee \neg\ \text{Bite}(x_2, x_3) \vee \neg\ \text{Cat}(x_4) \vee \neg\ \text{Likes}(x_4, x_2))$

Άσκηση 11.2(β)

Χρησιμοποιώντας μόνο τον κανόνα της ανάλυσης (resolution), κατασκευάστε δέντρο απόδειξης που να αποδεικνύει με απαγωγή σε άτοπο πως από τους τύπους (i), (ii), (iii), (iv) και (vii) της άσκησης 9.3 προκύπτει ως συμπέρασμα ότι τη Ράνα τη δάγκωσε ο Σούζος.

Dog(Milos) ∧ Dog(Suzos)

Cat(Psita) ∧ Cat(Rana)

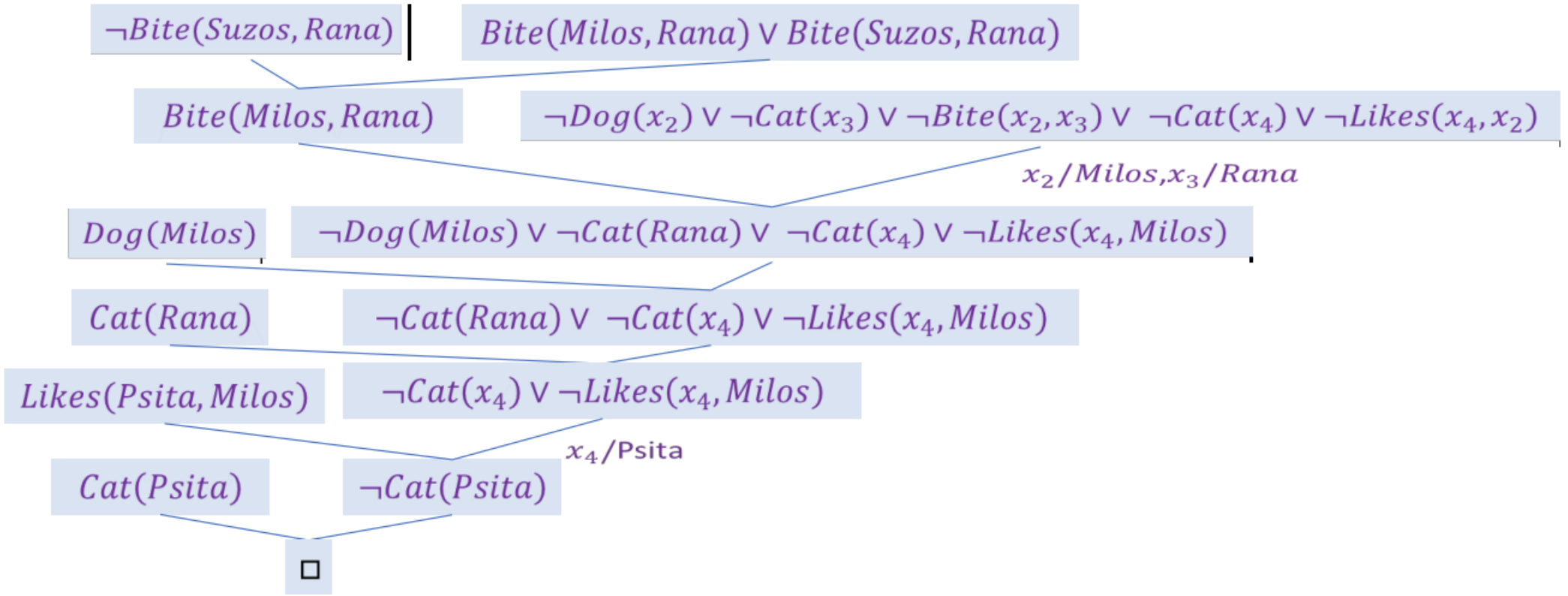
Likes(Psita, Milos)

Bite(Milos, Rana) ∨ Bite(Suzos, Rana)

(¬ Cat(x_1) ∨ ¬ Likes(x_1, Suzos))

(¬ Dog(x_2) ∨ ¬ Cat(x_3) ∨ ¬ Bite(x_2, x_3) ∨ ¬ Cat(x_4) ∨ ¬ Likes(x_4, x_2))

Άσκηση 11.2(β)


$\neg Bite(Suzos, Rana)$
$Bite(Milos, Rana) \vee Bite(Suzos, Rana)$
$Bite(Milos, Rana)$
$\neg Dog(x_2) \vee \neg Cat(x_3) \vee \neg Bite(x_2, x_3) \vee \neg Cat(x_4) \vee \neg Likes(x_4, x_2)$
$x_2/Milos, x_3/Rana$
$Dog(Milos)$
$\neg Dog(Milos) \vee \neg Cat(Rana) \vee \neg Cat(x_4) \vee \neg Likes(x_4, Milos)$
$Cat(Rana)$
$\neg Cat(Rana) \vee \neg Cat(x_4) \vee \neg Likes(x_4, Milos)$
$Likes(Psita, Milos)$
$\neg Cat(x_4) \vee \neg Likes(x_4, Milos)$
$x_4/Psita$
$Cat(Psita)$
$\neg Cat(Psita)$
$\square$

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Τι θα παρίστανε κάθε κατάσταση;

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Τι θα παρίστανε κάθε κατάσταση;

Κάθε κατάσταση θα περιείχε ένα (πιθανώς ημιτελές) δέντρο απόδειξης, τους αρχικούς τύπους της ΒΓ και την άρνηση του αποδεικτέου (σε μορφή CNF, για την ακρίβεια κάθε διάζευξη τύπου CNF θα παριστανόταν ως ξεχωριστός τύπος), καθώς και τους τύπους που έχουν προκύψει ως συμπεράσματα από τις εφαρμογές του κανόνα της ανάλυσης του δέντρου.

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Ποια θα ήταν η αρχική κατάσταση και ποιες θα ήταν οι τελικές καταστάσεις;

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Ποια θα ήταν η αρχική κατάσταση και ποιες θα ήταν οι τελικές καταστάσεις;

Η αρχική κατάσταση θα περιείχε ένα κενό δέντρο απόδειξης, τους αρχικούς τύπους της ΒΓ και την άρνηση του αποδεικτέου (σε μορφή CNF). Τελική θα ήταν μια κατάσταση της οποίας το δέντρο απόδειξης καταλήγει σε κενή διάζευξη.

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Ποιοι θα ήταν οι τελεστές μετάβασης;

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Ποιοι θα ήταν οι τελεστές μετάβασης;

Θα υπήρχε μόνο ένας τελεστής μετάβασης, ο οποίος θα επέκτεινε το δέντρο της τρέχουσας κατάστασης εφαρμόζοντας τον κανόνα της ανάλυσης σε ένα από τα φύλλα του δέντρου και έναν από τους διαθέσιμους τύπους της κατάστασης. Ο τελεστής θα αποθήκευε επίσης το συμπέρασμα που θα προέκυπτε από την εφαρμογή του κανόνα, ως πρόσθετο τύπο της νέας κατάστασης.

Άσκηση 11.2(γ)

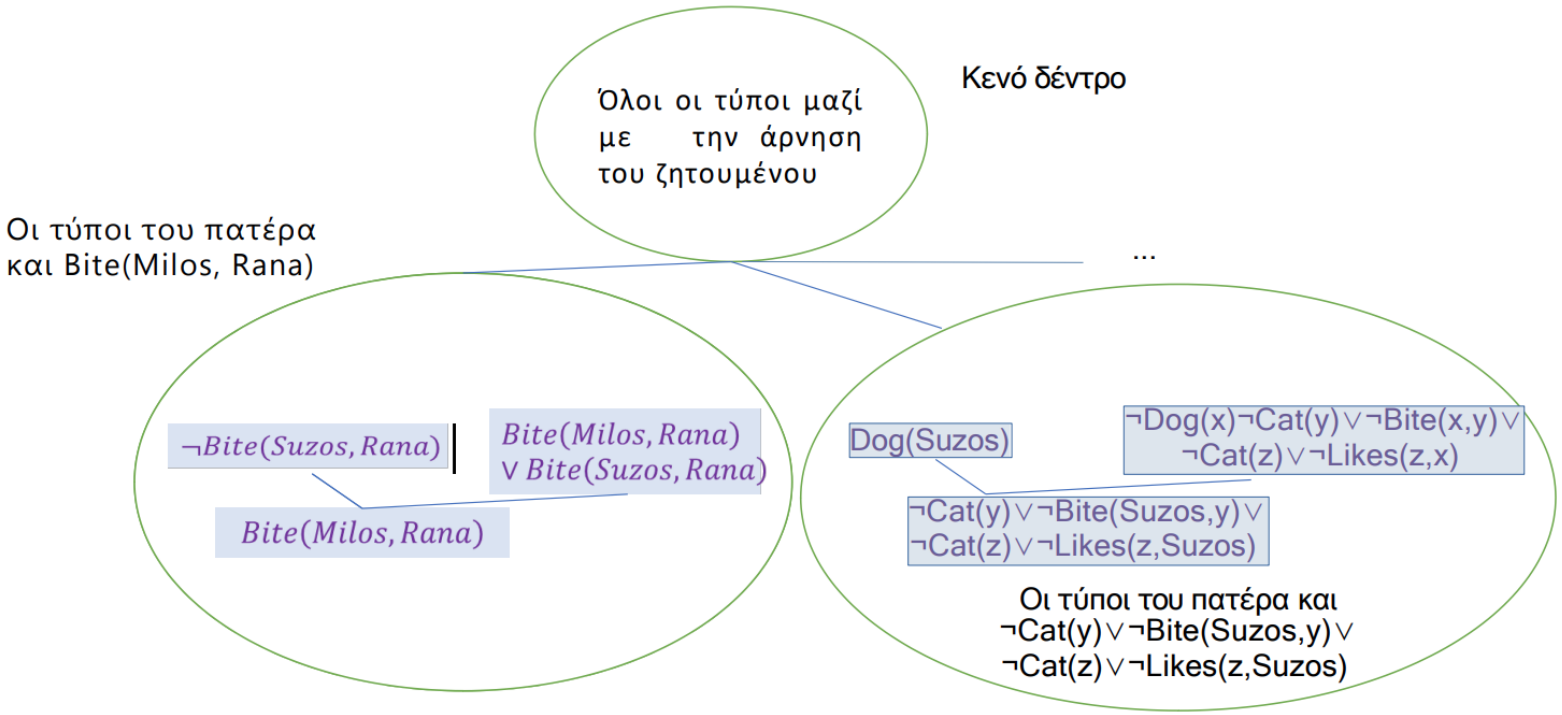
Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Σχεδιάστε το δέντρο αναζήτησης που θα κατασκεύαζε ο αλγόριθμος αναζήτησης πρώτα σε πλάτος (BFS) στην περίπτωση του σκέλους (γ). Αρκεί να σχεδιάσετε τη ρίζα και δύο παιδιά της.

Άσκηση 11.2(γ)

Εξηγήστε πώς θα μπορούσαμε να κατασκευάσουμε αυτόματα το δέντρο απόδειξης με αναζήτηση σε χώρο καταστάσεων.

Σχεδιάστε το δέντρο αναζήτησης που θα κατασκεύαζε ο αλγόριθμος αναζήτησης πρώτα σε πλάτος (BFS) στην περίπτωση του σκέλους (γ). Αρκεί να σχεδιάσετε τη ρίζα και δύο παιδιά της.



Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(i) Υπάρχει ένας σκύλος που γαβγίζει και φοβάται όλες τις γάτες

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(i) Υπάρχει ένας σκύλος που γαβγίζει και φοβάται όλες τις γάτες

$$\exists x\,(IsDog(x) \wedge Barks(x) \wedge \forall y\,(IsCat(y) \Rightarrow IsAfraidOf(x, y)))$$

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(ii) Ο Μίλος φοβάται τουλάχιστον μία γάτα που φοβάται τουλάχιστον ένα σκύλο

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(ii) Ο Μίλος φοβάται τουλάχιστον μία γάτα που φοβάται τουλάχιστον ένα σκύλο:

∃ x ∃ y (IsCat(x) ∧ IsDog(y) ∧ IsAfraidOf(Milos, x) ∧ IsAfraidOf(x, y))

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(iii) Κάθε σκύλος φοβάται κάθε γάτα που τον φοβάται

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(iii) Κάθε σκύλος φοβάται κάθε γάτα που τον φοβάται

$\forall x \forall y ((IsDog(x) \land IsCat(y) \land IsAfraidOf(y, x)) \Rightarrow IsAfraidOf(x, y))$

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(iv) Κάθε γάτα φοβάται τουλάχιστον ένα σκύλο (πιθανώς διαφορετικό για κάθε γάτα)

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(iv) Κάθε γάτα φοβάται τουλάχιστον ένα σκύλο (πιθανώς διαφορετικό για κάθε γάτα):

$\forall y\ (IsCat(y) \Rightarrow \exists x\ (IsDog(x) \wedge IsAfraidOf(y, x))$

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(ν) Κάθε γάτα φοβάται τουλάχιστον ένα σκύλο (πιθανώς διαφορετικό για κάθε γάτα) που τη φοβάται

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(ν) Κάθε γάτα φοβάται τουλάχιστον ένα σκύλο (πιθανώς διαφορετικό για κάθε γάτα) που τη φοβάται:

$\forall y\ (\text{IsCat}(y) \Rightarrow \exists x\ (\text{IsDog}(x) \wedge \text{IsAfraidOf}(y, x) \wedge \text{IsAfraidOf}(x, y))$

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(vi) Κάθε σκύλος φοβάται ακριβώς δύο γάτες (πιθανώς διαφορετικές για κάθε σκύλο)

Άσκηση 11.3(α)

Παραστήστε σε πρωτοβάθμια κατηγορηματική λογική τις σημασίες των παρακάτω ελληνικών προτάσεων. Συμβολίστε με z = x (ή z ≠ x) ότι οι μεταβλητές z και x παριστάνουν (ή όχι) την ίδια οντότητα.

(vi) Κάθε σκύλος φοβάται ακριβώς δύο γάτες (πιθανώς διαφορετικές για κάθε σκύλο):

$\forall$ x (IsDog(x) $\Rightarrow$ $\exists$ y_1 $\exists$ y_2 (IsCat(y_1) $\wedge$ IsCat(y_2) $\wedge$ $y_1 \neq y_2$ $\wedge$
IsAfraidOf(x, y_1) $\wedge$ IsAfraidOf(x, y_2) $\wedge$
$\forall$ z ((IsCat(z) $\wedge$ IsAfraidOf(x, z)) $\Rightarrow$ (z = y_1 $\vee$ z = y_2))))

Άσκηση 11.3(β)

Μετατρέψτε τις προτάσεις (iii), (iv) και την άρνηση της (v) του σκέλους (α) σε προτάσεις Horn πρωτοβάθμιας κατηγορηματικής λογικής.

$\forall x \forall y ((\text{IsDog}(x) \land \text{IsCat}(y) \land \text{IsAfraidOf}(y, x)) \Rightarrow \text{IsAfraidOf}(x, y))$

$\forall y (\text{IsCat}(y) \Rightarrow \exists x (\text{IsDog}(x) \land \text{IsAfraidOf}(y, x))$

$\exists y (\text{IsCat}(y) \land \forall x (\neg \text{IsDog}(x) \lor \neg \text{IsAfraidOf}(y, x) \lor \neg \text{IsAfraidOf}(x, y))$

Προτάσεις Horn ΠΚΛ: $\neg L_{1,1} \lor \neg \text{Breeze} \lor B_{1,1}$

Ισοδύναμα μπορούμε να το μετατρέψουμε σε «κανόνα»:

$(L_{1,1} \land \text{Breeze}) \Rightarrow B_{1,1}$

Άσκηση 11.3(β)

Μετατρέψτε τις προτάσεις (iii), (iv) και την άρνηση της (v) του σκέλους (α) σε προτάσεις Horn πρωτοβάθμιας κατηγορηματικής λογικής.

$\forall x \forall y ((IsDog(x) \land IsCat(y) \land IsAfraidOf(y, x)) \Rightarrow IsAfraidOf(x, y))$

$\neg IsDog(x) \lor \neg IsCat(y) \lor \neg IsAfraidOf(y, x) \lor IsAfraidOf(x, y)$

Άσκηση 11.3(β)

Μετατρέψτε τις προτάσεις (iii), (iv) και την άρνηση της (v) του σκέλους (α) σε προτάσεις Horn πρωτοβάθμιας κατηγορηματικής λογικής.

$\forall y\ (\text{IsCat}(y) \Rightarrow \exists x\ (\text{IsDog}(x) \wedge \text{IsAfraidOf}(y, x))$

$\forall y\ (\neg\ \text{IsCat}(y) \vee (\text{IsDog}(F(y)) \wedge \text{IsAfraidOf}(y, F(y)))$

$(\neg\ \text{IsCat}(y) \vee (\text{IsDog}(F(y)) \wedge (\neg\ \text{IsCat}(y) \vee \text{IsAfraidOf}(y, F(y)))$

Άσκηση 11.3(β)

Μετατρέψτε τις προτάσεις (iii), (iv) και την άρνηση της (v) του σκέλους (α) σε προτάσεις Horn πρωτοβάθμιας κατηγορηματικής λογικής.

$\exists\, y\, (\mathrm{IsCat}(y) \wedge \forall\, x\, (\neg\, \mathrm{IsDog}(x) \vee \neg\, \mathrm{IsAfraidOf}(y, x) \vee \neg\, \mathrm{IsAfraidOf}(x, y))$

$(\mathrm{IsCat}(C) \wedge \forall\, x\, (\neg\, \mathrm{IsDog}(x) \vee \neg\, \mathrm{IsAfraidOf}(C, x) \vee \neg\, \mathrm{IsAfraidOf}(x, C))$

$(\mathrm{IsCat}(C) \wedge (\neg\, \mathrm{IsDog}(x) \vee \neg\, \mathrm{IsAfraidOf}(C, x) \vee \neg\, \mathrm{IsAfraidOf}(x, C))$

Άσκηση 11.3(δ)

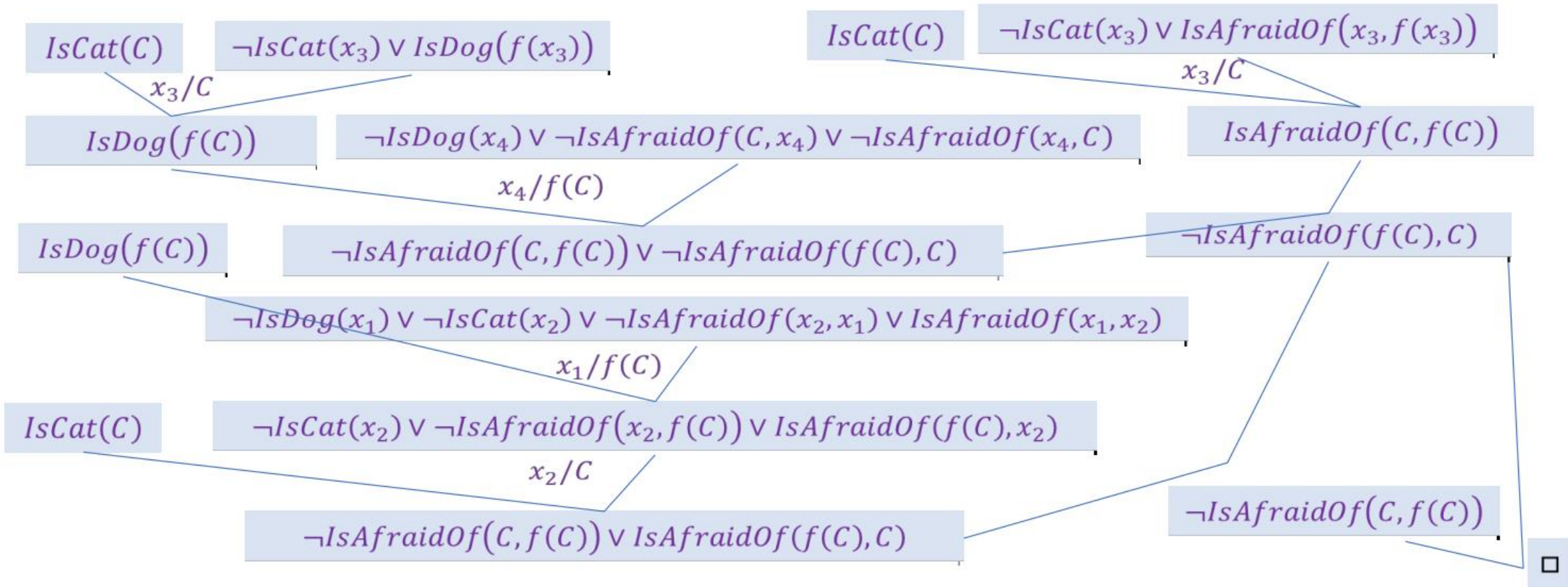
Σχεδιάστε δέντρο απόδειξης που να αποδεικνύει με απαγωγή σε άτοπο χρησιμοποιώντας μόνο τον κανόνα της ανάλυσης (resolution) ότι η πρόταση (v) του σκέλους (α) έπεται λογικά από τις προτάσεις (iii) και (iv).

¬ IsDog(x) ∨ ¬ IsCat(y) ∨ ¬ IsAfraidOf(y, x) ∨ IsAfraidOf(x, y)

(¬ IsCat(y) ∨ (IsDog(F(y)) ∧ (¬ IsCat(y) ∨ IsAfraidOf(y, F(y)))

(IsCat(C) ∧ (¬ IsDog(x) ∨ ¬ IsAfraidOf(C, x) ∨ ¬ IsAfraidOf(x, C))

Άσκηση 11.3(δ)


$IsCat(C)$
$\neg IsCat(x_3) \vee IsDog(f(x_3))$
x_3/C
$IsDog(f(C))$
$\neg IsDog(x_4) \vee \neg IsAfraidOf(C, x_4) \vee \neg IsAfraidOf(x_4, C)$
$x_4/f(C)$
$IsDog(f(C))$
$\neg IsAfraidOf(C, f(C)) \vee \neg IsAfraidOf(f(C), C)$
$\neg IsDog(x_1) \vee \neg IsCat(x_2) \vee \neg IsAfraidOf(x_2, x_1) \vee IsAfraidOf(x_1, x_2)$
$x_1/f(C)$
$IsCat(C)$
$\neg IsCat(x_2) \vee \neg IsAfraidOf(x_2, f(C)) \vee IsAfraidOf(f(C), x_2)$
x_2/C
$\neg IsAfraidOf(C, f(C)) \vee IsAfraidOf(f(C), C)$
$IsCat(C)$
$\neg IsCat(x_3) \vee IsAfraidOf(x_3, f(x_3))$
x_3/C
$IsAfraidOf(C, f(C))$
$\neg IsAfraidOf(f(C), C)$
$\neg IsAfraidOf(C, f(C))$
□

Άσκηση 11.4(α)

Η προτασιακή λογική είναι:

α) αποκρίσιμη

β) ημι-αποκρίσιμη

γ) μη αποκρίσιμη

Αποκρισιμότητα: Υπάρχουν (πλήρεις) αλγόριθμοι που για κάθε ΒΓ και α με ΒΓ $\models$ α αποκρίνονται πάντα αληθές (ΒΓ $\vdash_i$ α) και υπάρχουν αλγόριθμοι που επιπλέον όταν ΒΓ $\not\models$ α να αποκρίνεται πάντα ψευδές (ΒΓ $\not\vdash_i$ α)

Άσκηση 11.4(α)

Η προτασιακή λογική είναι:

α) αποκρίσιμη

β) ημι-αποκρίσιμη

γ) μη αποκρίσιμη

Το πρόβλημα είναι αποκρίσιμο γιατί θα μπορούσαμε να απαντήσουμε ελέγχοντας εξαντλητικά όλα τα μοντέλα (βλ. TTEntails), γιατί υπάρχει πεπερασμένος αριθμός μοντέλων.

Άσκηση 11.4(β)

Η ΠΚΛ είναι:

α) αποκρίσιμη

β) ημι-αποκρίσιμη

γ) μη αποκρίσιμη

Αποκρισιμότητα: Υπάρχουν (πλήρεις) αλγόριθμοι που για κάθε ΒΓ και α με ΒΓ $\models$ α αποκρίνονται πάντα αληθές (ΒΓ $\vdash_i$ α) και υπάρχουν αλγόριθμοι που επιπλέον όταν ΒΓ $\not\models$ α να αποκρίνεται πάντα ψευδές (ΒΓ $\not\vdash_i$ α)

Άσκηση 11.4(β)

Η ΠΚΛ είναι:

α) αποκρίσιμη

β) ημι-αποκρίσιμη

γ) μη αποκρίσιμη

Υπάρχουν (πλήρεις) αλγόριθμοι που για κάθε ΒΓ και α με ΒΓ $\models$ α αποκρίνονται πάντα αληθές (ΒΓ $\vdash_i$ α). Δεν υπάρχει αλγόριθμος που επιπλέον όταν ΒΓ $\not\models$ α να αποκρίνεται πάντα ψευδές (ΒΓ $\not\vdash_i$ α).

Τα μοντέλα/ερμηνείες στην ΠΚΛ είναι πάρα πολλά (άπειρα αν π.χ. περιλάβουμε τους φυσικούς αριθμούς στο D) και δεν μπορούμε να τα ελέγξουμε εξαντλητικά.

Άσκηση 11.4(γ)

Κάθε τύπος προτασιακής λογικής μπορεί να μετατραπεί σε κανονική συζευκτική μορφή:

Α) συμφωνώ και μάλιστα ο νέος τύπος είναι ταυτολογικά ισοδύναμος με τον αρχικό

Β) συμφωνώ, αλλά ο νέος τύπος δεν είναι σίγουρα ταυτολογικά ισοδύναμος με τον αρχικό

Γ) διαφωνώ.

Άσκηση 11.4(γ)

Κάθε τύπος προτασιακής λογικής μπορεί να μετατραπεί σε κανονική συζευκτική μορφή:

Α) συμφωνώ και μάλιστα ο νέος τύπος είναι ταυτολογικά ισοδύναμος με τον αρχικό

Β) συμφωνώ, αλλά ο νέος τύπος δεν είναι σίγουρα ταυτολογικά ισοδύναμος με τον αρχικό

Γ) διαφωνώ.

Άσκηση 11.4(δ)

Κάθε τύπος ΠΚΛ μπορεί να μετατραπεί σε κανονική συζευκτική μορφή:

Α) συμφωνώ και μάλιστα ο νέος τύπος είναι ταυτολογικά ισοδύναμος με τον αρχικό

Β) συμφωνώ, αλλά ο νέος τύπος δεν είναι σίγουρα ταυτολογικά ισοδύναμος με τον αρχικό

Γ) διαφωνώ.

Άσκηση 11.4(δ)

Κάθε τύπος ΠΚΛ μπορεί να μετατραπεί σε κανονική συζευκτική μορφή:

Α) συμφωνώ και μάλιστα ο νέος τύπος είναι ταυτολογικά ισοδύναμος με τον αρχικό

Β) συμφωνώ, αλλά ο νέος τύπος δεν είναι σίγουρα ταυτολογικά ισοδύναμος με τον αρχικό

Γ) διαφωνώ.

Άσκηση 12.1

Δείξτε χρησιμοποιώντας τον αλγόριθμο εξαγωγής συμπερασμάτων προς τα εμπρός fol-fc-ask ότι ο τύπος Path(A, NextOf(Nextof(A))) προκύπτει ως συμπέρασμα από την ακόλουθη βάση γνώσεων προτάσεων Horn.

Link(x, NextOf(x))

Link(y, z) ⇒ Path(y, z)

((Path(u,v) ∧ Link(v, w)) ⇒ Path(u,w))

Άσκηση 12.1

Υπάρχει στη ΒΓ ο κανόνας Link(y, z) ⇒ Path(y, z), ο οποίος με νέες μεταβλητές γίνεται:

Link(y_1, z_1) ⇒ Path(y_1, z_1). 1

Επίσης, ο κανόνας Link(x, NextOf(x)), με νέες μεταβλητές γίνεται:

Link(x_2, NextOf(x_2)) 2

Για {y_1/ x_2, z_1/ NextOf(x_2)} ενοποιούμε τα 1 και 2 και προστίθεται στην ΒΓ το συμπέρασμα Path(x_2, NextOf(x_2)).

Άσκηση 12.1

Ο κανόνας ((Path(u,v) ∧ Link(v, w)) ⇒ Path(u,w)) με νέες μεταβλητές γράφεται ως:

$((\text{Path}(u_3, v_3) \wedge \text{Link}(v_3, w_3)) \Rightarrow \text{Path}(u_3, w_3))$ 3

Χρησιμοποιούμε νέες μεταβλητές στα ακόλουθα γεγονότα της ΒΓ:

$\text{Path}(x_5, \text{NextOf}(x_5))$
$\text{Link}(x_4, \text{NextOf}(x_4))$ 4

Για $\{u_3/\, x_5,\ v_3/\, \text{NextOf}(x_5),\ x_4/\, v_3,\ w_3/\, \text{NextOf}(v_3)\}$ ενοποιούμε τα 3 και 4 και προστίθεται στη ΒΓ το συμπέρασμα $\text{Path}(x_5, \text{NextOf}(\text{NextOf}(x_5))$, το οποίο για $\{x_5/\text{A}\}$ έχει ως αποτέλεσμα Path(A, NextOf(A)).

Λογικός προγραμματισμός

Πρόγραμμα συνένωσης λιστών.

append([], Y, Y).

Το να κάνουμε append μια κενή λίστα με μία λίστα Y έχει ως αποτέλεσμα την Y.

append([A|X], Y, [A|Z]):- append(X, Y, Z).

head tail

Αποτέλεσμα του να κάνουμε αναδρομικά append το tail X στη λίστα Y. Η αναδρομή τερματίζει όταν βρούμε κενή λίστα.

Λογικός προγραμματισμός

Πρόγραμμα συνένωσης λιστών.

append([], Y, Y).

append([A|X], Y, [A|Z]):- append(X, Y, Z).

?- append([a], [b, c], [a, b, c]).

yes

?- append([a], [b, c], Result).

Result = [a, b, c]

Εδώ έχουμε:
A = a
X = [ ]
Y = [b, c]
Άρα κάνουμε append [a|X] με Y = [b,c]
Και το αποτέλεσμα είναι [a|Z]
Όπου Z είναι το αποτέλεσμα της αναδρομής του append του X στο Y, δηλαδή της κενής λίστας στο Y, άρα είμαστε στο base case, οπότε Z = Y = [b,c] άρα το Result θα είναι [a, b, c]

Λογικός προγραμματισμός

Πρόγραμμα συνένωσης λιστών.

```
append([], Y, Y).
append([A|X], Y, [A|Z]):- append(X, Y, Z).
```

```
?- append(A, B, [a, b, c]).
A = [], B = [a, b, c] ;
A = [a], B = [b, c] ;
A = [a, b], B = [c] ;
A = [a, b, c], B = [] ;
no
```

Ερώτηση στην prolog να αποσυνθέσει τη λίστα [a, b, c] σε 2 λίστες A και B έτσι ώστε όταν τις συνενώνουμε να μας βγάζει την αρχική λίστα.

Ως απάντηση έχουμε αυτές τις 4 εναλλακτικές και καμία άλλη.

Άσκηση 12.2

Γράψτε ένα πρόγραμμα Prolog το οποίο να αντιστρέφει λίστες, όπως στο παρακάτω παράδειγμα (με πλάγια γράμματα φαίνονται οι αποκρίσεις της Prolog, με έντονα ό,τι γράφει ο χρήστης).

Μπορείτε να χρησιμοποιήσετε το κατηγόρημα append των διαφανειών (προσοχή: τα πρώτα δύο ορίσματα του append είναι λίστες)

?- **myReverse([a, b, c, d], R).**

R = [d, c, b, a] ;

no

?- **myReverse([a, b, c, d], [d, c, b, a]).**

yes

Άσκηση 12.2

```
myReverse([], []).
```

Το αντίστροφο μιας κενής λίστας είναι ο εαυτός της

Λίστα προς αντιστροφή

Αποτέλεσμα (ανεστραμμένη λίστα)

```
myReverse([FirstIn | RestIn], ListOut) :-
myReverse(RestIn,RestOut), append(RestOut, [FirstIn], ListOut).
```

Ανεστραμμένη RestIn

Το ListOut προκύπτει κάνοντας append το FirstIn με το RestOut

Άσκηση 12.2 (επεξήγηση αναδρομής)

Βήμα 1: Decompose [a, b, c, d]
Είσοδος: [a, b, c, d]
Head: FirstIn = a
Tail: RestIn = [b, c, d]
Αναδρομή: myReverse([b, c, d], RestOut)

Βήμα 2: Decompose [b, c, d]
Είσοδος: [b, c, d]
Head: FirstIn = b
Tail: RestIn = [c, d]
Αναδρομή: myReverse([c, d], RestOut)

Άσκηση 12.2 (επεξήγηση αναδρομής)

Βήμα 3: Decompose [c, d]
Είσοδος: [c, d]
Head: FirstIn = c
Tail: RestIn = [d]
Αναδρομή: myReverse([d], RestOut)

Βήμα 4: Decompose [d]
Είσοδος: [d]
Head: FirstIn = d
Tail: RestIn = []
Αναδρομή: myReverse([], RestOut)

Άσκηση 12.2 (επεξήγηση αναδρομής)

Βήμα 5: Base Case

Είσοδος: []

Base case: myReverse([], [])

Result: RestOut = []

Βήμα 6: Ξαναφτίαχνουμε τη λίστα

Reverse [d]:

append([], [d], [d])

RestOut = [d]

Result: myReverse([d], [d])

Άσκηση 12.2 (επεξήγηση αναδρομής)

Reverse [c, d]:

append([d], [c], [d, c])

RestOut = [d, c]

Result: myReverse([c, d], [d, c])

Reverse [b, c, d]:

append([d, c], [b], [d, c, b])

RestOut = [d, c, b]

Result: myReverse([b, c, d], [d, c, b])

Άσκηση 12.2 (επεξήγηση αναδρομής)

Reverse [a, b, c, d]:

append([d, c, b], [a], [d, c, b, a])

RestOut = [d, c, b, a]

Result: myReverse([a, b, c, d], [d, c, b, a])

Άσκηση 12.3(α)

Παραστήστε τις προτάσεις (i), (ii), (iii) και (iv) σε κατάλληλη μορφή πρωτοβάθμιας κατηγορηματικής λογικής, ώστε να είναι δυνατόν να αποδειχθεί με τους αλγορίθμους fol-fcask και fol-bc-ask ότι τα (i), (ii) και (iii) συνεπάγονται το (iv).

(i) Η Μαρία συμπαθεί τον Γιάννη.

(ii) Τον Νίκο τον έκοψε ο Γιάννης ή ο Γιώργος.

(iii) Αν κάποιος συμπαθεί τον x, τότε ο x δεν έκοψε κανέναν.

(iv) Τον Νίκο τον έκοψε ο Γιώργος. Δείξτε σύντομα τα δέντρα απόδειξης (όπως στις διαφάνειες) που θα κατασκεύαζαν οι δύο αλγόριθμοι για να αποδείξουν το (iv).

Υπόδειξη: Χρησιμοποιήστε στην παράσταση του (ii) ένα κατηγόρημα της μορφής OrCut(x, y, z) και στην παράσταση του (iii) ένα κατηγόρημα της μορφής NotCut(x,y). Προσθέστε κανόνες που να συνδέουν τα OrCut(x,y,z), Cut(x,y) και NotCut(x,y).

Άσκηση 12.3(α)

(i) Likes(Mary, John)

(ii) OrCut(John, George, Nick)

(iii) Likes(y, x) => NotCut(x, z)

(iv) Cut(George, Nick) ← Θέλουμε να αποδείξουμε

Προσθέτουμε τους κανόνες:

(v) OrCut(x, y, z) ^ NotCut(y, z) => Cut(x, z)

(vi) OrCut(x, y, z) ^ NotCut(x, z) => Cut(y, z)

Άσκηση 12.3(α)

(i) Likes(Mary, John)

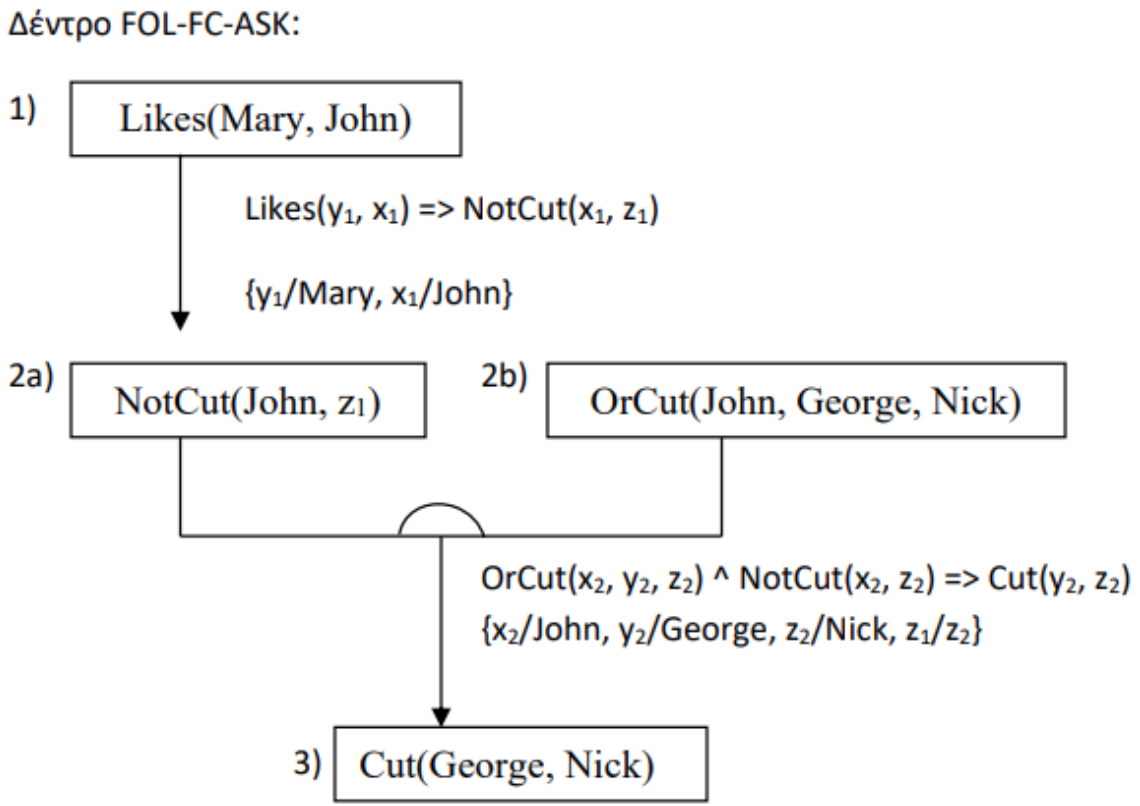
(ii) OrCut(John, George, Nick)

(iii) Likes(y, x) => NotCut(x, z)

(iv) Cut(George, Nick)

(v) OrCut(x, y, z) ^ NotCut(y, z) => Cut(x, z)

(vi) OrCut(x, y, z) ^ NotCut(x, z) => Cut(y, z)



Άσκηση 12.3(α)

(i) Likes(Mary, John)

(ii) OrCut(John, George, Nick)

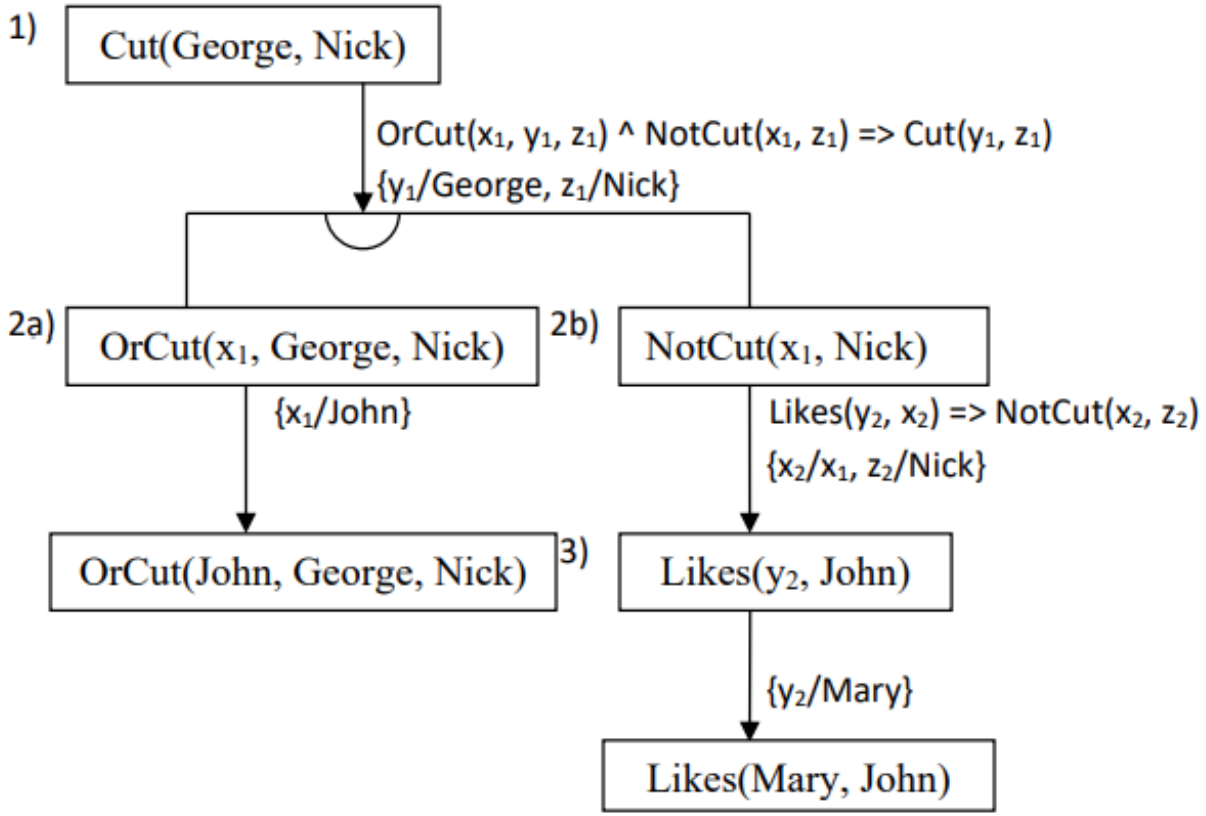
(iii) Likes(y, x) => NotCut(x, z)

(iv) Cut(George, Nick)

(v) OrCut(x, y, z) ^ NotCut(y, z) => Cut(x, z)

(vi) OrCut(x, y, z) ^ NotCut(x, z) => Cut(y, z)

Δέντρο FOL-BC-ASK:



Άσκηση 12.3(β)

Γράψτε ως πρόγραμμα Prolog τους τύπους του σκέλους (α) για τα (i), (ii), (iii), (iv) και τους επιπλέον κανόνες για τις σχέσεις μεταξύ των OrCut(x,y,z), Cut(x,y) και NotCut(x,y).

Χρησιμοποιήστε το πρόγραμμά σας και την SWI-Prolog για να αποδείξετε ότι τον Νίκο τον έκοψε ο Γιώργος, όπως φαίνεται παρακάτω:

?- cut(george, nick).

yes

?- cut(X, nick).

X = george ;

no

Άσκηση 12.3(β)

Το αντίστοιχο πρόγραμμα σε Prolog είναι το εξής :

```
likes(mary, john).
orCut(john, george, nick).
notCut(X, _) :- likes(_, X).
cut(X, Z) :- orCut(X, Y, Z), notCut(Y, Z).
cut(Y, Z) :- orCut(X, Y, Z), notCut(X, Z).
```

Άσκηση 12.4

Συμπληρώστε τα υπογραμμισμένα κενά στο παρακάτω πρόγραμμα Prolog, ώστε να συνδυάζει δύο λίστες ως εξής:

?- combine([a, b, c], [d, e, f], L).

L = [a, d, b, e, c, f]

?- combine([a, b, c], [d, e], L).

L = [a, d, b, e, c]

?- combine([a, b], [d, e, f, g], L).

L = [a, d, b, e, f, g]

?- combine([a, b], [], L).

L = [a, b]

?- combine([], [d, e, f], L).

L = [d, e, f].

?- combine([], [], []).

Yes

?- combine([a, b, c], [d, e], [a, d, b, e, c]).

Yes

```
combine(_____, _____, _____).
combine(_____, _____, _____).
combine(_________, _________, _________) :-
combine(_____, _____, _____).
```

Άσκηση 12.4

Συμπληρώστε τα υπογραμμισμένα κενά στο παρακάτω πρόγραμμα Prolog, ώστε να συνδυάζει δύο λίστες ως εξής:

?- combine([a, b, c], [d, e, f], L).

L = [a, d, b, e, c, f]

?- combine([a, b, c], [d, e], L).

L = [a, d, b, e, c]

?- combine([a, b], [d, e, f, g], L).

L = [a, d, b, e, f, g]

?- combine([a, b], [], L).

L = [a, b]

?- combine([], [d, e, f], L).

L = [d, e, f].

?- combine([], [], []).

Yes

?- combine([a, b, c], [d, e], [a, d, b, e, c]).

Yes

```
combine([], B, B).
combine(A, [], A).
combine([F1|R1], [F2|R2], [F1, F2|R]):- combine(R1, R2, R).
```