Συμπερασμός στη Λογική Πρώτης Τάξης

Συμπερασμός στη Λογική Πρώτης Τάξης

Σε οποιαδήποτε λογική, ο **συμπερασμός (inference)** ή η **συλλογιστική (reasoning)** είναι η διαδικασία της μηχανικής εξαγωγής συμπερασμάτων από δοσμένες υποθέσεις.

Οι υποθέσεις μπορεί να είναι η βάση γνώσεων που έχουμε δημιουργήσει για μια εφαρμογή και το συμπέρασμα ένας τύπος που θέλουμε να ελέγξουμε αν έπεται λογικά από τη γνώση που έχουμε.

Ο συμπερασμός είναι συνήθως μια διαδικασία με ένα ή περισσότερα βήματα. Κάθε βήμα αυτής της διαδικασίας περιλαμβάνει την εφαρμογή ενός **κανόνα συμπερασμού (inference rule)**.

Κανόνες Συμπερασμού

Ένας **κανόνας συμπερασμού (inference rule)** είναι μια έκφραση της μορφής

$$\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\beta}$$

όπου $\alpha_1, \alpha_2, \ldots, \alpha_n$ είναι τύποι της λογικής πρώτης τάξης οι οποίοι ονομάζονται **υποθέσεις** και β είναι ένας τύπος που ονομάζεται **συμπέρασμα**.

Ένας κανόνας συμπερασμού **εφαρμόζεται** ως εξής: οποτεδήποτε έχουμε ένα σύνολο τύπων που ταιριάζουν με τις υποθέσεις του κανόνα, μπορούμε να εξάγουμε το συμπέρασμα.

Επειδή ο συμπερασμός είναι μια μηχανική διαδικασία, μπορεί να κωδικοποιηθεί από ένα αλγόριθμο και να χρησιμοποιηθεί για την υλοποίηση ευφυών πρακτόρων.

Η Έννοια της Παραγωγής ή Απόδειξης

Έστω μια μέθοδος συμπερασμού της λογικής πρώτης τάξης και KB ένα σύνολο υποθέσεων.

Θα λέμε ότι ένα συμπέρασμα ϕ **παράγεται** από το σύνολο των υποθέσεων KB ανν

- ϕ είναι ένα στοιχείο του συνόλου KB
- ϕ είναι το αποτέλεσμα της εφαρμογής ενός κανόνα συμπερασμού σε ένα σύνολο προτάσεων που παράγονται από το σύνολο KB.

Η Έννοια της Παραγωγής ή Απόδειξης

Μια **παραγωγή (derivation)** ή **απόδειξη (proof)** ενός συμπεράσματος ϕ από ένα σύνολο υποθέσεων KB είναι μια ακολουθία τύπων στην οποία κάθε τύπος είναι στοιχείο της KB ή προκύπτει από την εφαρμογή ενός κανόνα συμπερασμού σε προηγούμενα στοιχεία της ακολουθίας.

Συμβολισμός: Όταν υπάρχει μια απόδειξη του τύπου ϕ από την KB, τότε γράφουμε $KB \vdash \phi$.

Όρθές και Πλήρεις Μέθοδοι Συμπερασμού

Θέλουμε να βρούμε μια μέθοδο συμπερασμού για τη λογική πρώτης τάξης που να είναι **ορθή και πλήρης**, δηλαδή να ισχύει

$$KB \models \alpha \;\; \text{ανν} \;\; KB \vdash \alpha$$

για κάθε σύνολο τύπων KB και κάθε τύπο α.

Κανόνες Συμπερασμού

Οι παρακάτω κανόνες συμπερασμού της **προτασιακής λογικής** ισχύουν και για την λογική πρώτης τάξης :

- Τρόπος του θέτειν (modus ponens): $\frac{\alpha,\ \alpha \Rightarrow \beta}{\beta}$
- Απαλοιφή σύζευξης: $\frac{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}{\alpha_i}$
- Εισαγωγή σύζευξης: $\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n}$
- Εισαγωγή διάζευξης: $\frac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n}$
- Απαλοιφή διπλής άρνησης: $\frac{\neg\neg\alpha}{\alpha}$
- Μοναδιαία ανάλυση : $\frac{\alpha \vee \beta,\ \neg\beta}{\alpha}$
- Ανάλυση: $\frac{\alpha \vee \beta,\ \neg\beta \vee \gamma}{\alpha \vee \gamma}$

Κανόνες Συμπερασμού (συνέχεια)

Υπάρχουν επίσης κανόνες συμπερασμού που είναι ειδικά για την λογική πρώτης τάξης.

Για να τους παρουσιάσουμε, θα χρειαστεί πρώτα να ορίσουμε μερικές νέες έννοιες.

Η Έννοια της Αντικατάστασης

Ορισμός. Μια **αντικατάσταση (substitution)** θ είναι ένα πεπερασμένο σύνολο της μορφής $\{v_1/t_1, \ldots, v_n/t_n\}$ όπου

- κάθε v_i είναι μια μεταβλητή και κάθε t_i είναι ένας όρος διαφορετικός από την v_i,
- οι μεταβλητές $v_1, \ldots, v_n$ είναι διαφορετικές μεταξύ τους, και
- καμιά μεταβλητή v_i δεν υπάρχει σε κάποιο από τους όρους t_i.

Κάθε στοιχείο t_i λέγεται **δέσμευση (binding)** για την v_i. Οι μεταβλητές που έχουν δεσμεύσεις λέγονται **δεσμευμένες (bound)**.

Η **κενή αντικατάσταση (empty substitution)** θα συμβολίζεται με $\{\}$.

Η Έννοια της Βασικής Αντικατάστασης

Ορισμός. Μια αντικατάσταση λέγεται **βασική (ground)** αν οι όροι t_i δεν περιέχουν μεταβλητές (δηλαδή, είναι **βασικοί** όροι - ground terms).

Παραδείγματα

Τα σύνολα

$$\{x/John, y/Mary\} \quad \text{και} \quad \{x/John, y/MotherOf(z)\}$$

είναι αντικαταστάσεις. Η πρώτη είναι βασική.

Τα σύνολα

$$\{x/F(x)\} \quad \text{και} \quad \{x/G(y), y/F(x)\}$$

δεν είναι αντικαταστάσεις.

Εφαρμογή μιας Αντικατάστασης σ΄ ένα Τύπο

Όρισμός. Έστω $\theta = \{v_1/t_1, \ldots, v_n/t_n\}$ μια αντικατάσταση και α μια έκφραση (όρος ή τύπος) της λογικής πρώτης τάξης χωρίς ποσοδείκτες. Θα συμβολίζουμε με

$$SUBST(\theta, \alpha)$$

την έκφραση που προκύπτει από την α όταν εφαρμόσουμε την αντικατάσταση θ, δηλαδή όταν αντικαταστήσουμε κάθε μεταβλητή v_i με τον όρο t_i $(i = 1, \ldots, n)$.

Παραδείγματα

$$SUBST(\{x/John, y/Mary\}, Loves(x, y)) = Loves(John, Mary)$$

$$SUBST(\{x/John, y/HouseOf(z)\}, Likes(x, y)) =$$

$$Likes(John, HouseOf(z))$$

Παρατήρηση: $SUBST(\{\}, \alpha) = \alpha$ για κάθε έκφραση της λογικής πρώτης τάξης α.

Δύο Νέοι Κανόνες Συμπερασμού

Θα δώσουμε δύο νέους κανόνες συμπερασμού για τη λογική πρώτης τάξης που αφορούν τους ποσοδείκτες:

- **Καθολικός Προσδιορισμός (universal instantiation):** Για κάθε τύπο α, μεταβλητή v και βασικό όρο t:

$$\frac{(\forall v)\alpha}{SUBST(\{v/t\}, \alpha)}$$

Παραδείγματα

- Από τον τύπο $(\forall x)Likes(x, IceCream)$ μπορούμε να συμπεράνουμε $Likes(Ben, IceCream)$ χρησιμοποιώντας την αντικατάσταση $\{x/Ben\}$.
- Από τον τύπο $(\forall x)(\exists z)Hates(x, z)$ μπορούμε να συμπεράνουμε $(\exists z)Hates(MotherOf(Ben), z)$ χρησιμοποιώντας την αντικατάσταση $\{x/MotherOf(Ben)\}$.

Προσοχή: Ο κανόνας του καθολικού προσδιορισμού ισχύει και για μη βασικούς όρους t, όμως πρέπει να είμαστε προσεκτικοί με τις μεταβλητές που εμφανίζονται στα α και t. Δείτε το βιβλίο λογικής του Enderton για τον ακριβή ορισμό του κανόνα σ΄ αυτή την περίπτωση.

Δύο Νέοι Κανόνες Συμπερασμού

- **Υπαρξιακός Προσδιορισμός (existential instantiation)**:
  Για κάθε τύπο α και μεταβλητή v:

$$\frac{(\exists v)\alpha}{SUBST(\{v/K\},\alpha)}$$

  όπου K είναι ένα **νέο** σύμβολο σταθεράς.

  Παράδειγμα: Από $(\exists x)Likes(x, IceCream)$, μπορούμε να συμπεράνουμε $Likes(Somebody, IceCream)$ εφόσον το $Somebody$ είναι ένα νέο σύμβολο σταθεράς που δεν έχει χρησιμοποιηθεί ενωρίτερα.

Παράδειγμα Απόδειξης

Ας θεωρήσουμε το παρακάτω κείμενο:

> Ο νόμος λέει ότι είναι έγκλημα για έναν Αμερικανό να πουλάει όπλα σε εχθρικά κράτη. Το κράτος Nono, που είναι εχθρός της Αμερικής, έχει πυραύλους. Όλους αυτούς τους πυραύλους τους έχει πουλήσει στο Nono ο συνταγματάρχης West, ο οποίος είναι Αμερικανός.

Πως μπορούμε να 'μεταφράσουμε' αυτό το κείμενο σε λογική πρώτης τάξης, και να χρησιμοποιήσουμε τους κανόνες συμπερασμού που δώσαμε παραπάνω, για να αποδείξουμε ότι ο West είναι εγκληματίας;

Διατύπωση σε Λογική Πρώτης Τάξης

- ... είναι έγκλημα για έναν Αμερικανό να πουλάει όπλα σε εχθρικά κράτη.

$$(\forall x, y, z)\ (American(x) \wedge Weapon(y) \wedge Nation(z) \wedge$$

$$Hostile(z) \wedge Sells(x, z, y) \Rightarrow Criminal(x))$$

- Το κράτος Nono ...

$$Nation(Nono)$$

- ... που είναι εχθρός της Αμερικής ...

$$Enemy(Nono, America)$$

- ... έχει πυραύλους.

$$(\exists x)\ (Owns(Nono, x) \wedge Missile(x))$$

Διατύπωση σε Λογική Πρώτης Τάξης

- Όλους αυτούς τους πυραύλους τους έχει πουλήσει στο Nono ο συνταγματάρχης West.

$$(\forall x)\ (Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x))$$

Παρατήρηση: Δεν καταγράψαμε τον στρατιωτικό βαθμό του West επειδή δεν θα μας χρειαστεί για να βγάλουμε το συμπέρασμα μας.

- ... ο οποίος είναι Αμερικανός.

$$American(West)$$

Διατύπωση σε Λογική Πρώτης Τάξης

Θα χρειαστούμε επίσης τις παρακάτω προτάσεις που **δεν υπάρχουν στο κείμενο που μας δόθηκε**. Οι προτάσεις αυτές κωδικοποιούν γνώση που έχουμε ήδη σχετικά με το πεδίο εφαρμογής και μπορούμε να την χρησιμοποιήσουμε.

- Η Αμερική είναι κράτος.

$$Nation(America)$$

- Οι πύραυλοι είναι όπλα.

$$(\forall x)\ (Missile(x) \Rightarrow Weapon(x))$$

Διατύπωση σε Λογική Πρώτης Τάξης

- Στο παραπάνω κείμενο 'εχθρός της Αμερικής ' σημαίνει 'εχθρικό κράτος '.

$$(\forall x)\ (Enemy(x, America) \Rightarrow Hostile(x))$$

Παρατήρηση: Θα μπορούσαμε να έχουμε διπλή συνεπαγωγή. Η απλή συνεπαγωγή είναι αρκετή για να βγάλουμε το συμπέρασμα μας.

Απόδειξη

- Από τον τύπο $(\exists x)\ (Owns(Nono, x) \wedge Missile(x))$ και τον κανόνα του υπαρξιακού προσδιορισμού έχουμε:

$$Owns(Nono, M1) \wedge Missile(M1)$$

- Από τον τύπο $Owns(Nono, M1) \wedge Missile(M1)$ και τον κανόνα απαλοιφής της σύζευξης έχουμε:

$$Owns(Nono, M1),\ Missile(M1)$$

- Από τον τύπο $(\forall x)\ (Missile(x) \Rightarrow Weapon(x))$ και τον κανόνα του καθολικού προσδιορισμού έχουμε:

$$Missile(M1) \Rightarrow Weapon(M1)$$

Απόδειξη

- Από τούς τύπους $Missile(M1), Missile(M1) \Rightarrow Weapon(M1)$ και τον κανόνα Modus Ponens έχουμε:

$$Weapon(M1)$$

- Από τον τύπο

$$(\forall x)\ (Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x))$$

και τον κανόνα του καθολικού προσδιορισμού έχουμε:

$$Owns(Nono, M1) \wedge Missile(M1) \Rightarrow Sells(West, Nono, M1)$$

Απόδειξη

- Από τούς τύπους $Owns(Nono, M1) \wedge Missile(M1)$,

  $$Owns(Nono, M1) \wedge Missile(M1) \Rightarrow Sells(West, Nono, M1)$$

  και τον κανόνα Modus Ponens έχουμε:

  $$Sells(West, Nono, M1)$$

- Από τον τύπο
  $(\forall x, y, z)\ (American(x) \wedge Weapon(y) \wedge Nation(z) \wedge$
  $Hostile(z) \wedge Sells(x, z, y) \Rightarrow Criminal(x))$ και τον κανόνα του καθολικού προσδιορισμού (3 φορές!) έχουμε:

  $$American(West) \wedge Weapon(M1) \wedge Nation(Nono) \wedge$$

  $$Hostile(Nono) \wedge Sells(West, Nono, M1) \Rightarrow Criminal(West)$$

Απόδειξη

- Από τον τύπο $(\forall x)\ (Enemy(x, America) \Rightarrow Hostile(x))$ και τον κανόνα του καθολικού προσδιορισμού έχουμε:

$$Enemy(Nono, America) \Rightarrow Hostile(Nono)$$

- Από τους τύπους $Enemy(Nono, America)$,

$$Enemy(Nono, America) \Rightarrow Hostile(Nono)$$

και τον κανόνα Modus Ponens έχουμε:

$$Hostile(Nono)$$

Απόδειξη

- Από τους τύπους

$$American(West),\ Weapon(M1),\ Nation(Nono),$$

$$Hostile(Nono),\ Sells(West, Nono, M1)$$

και τον κανόνα εισαγωγής σύζευξης έχουμε:

$$American(West) \wedge Weapon(M1) \wedge Nation(Nono) \wedge$$

$$Hostile(Nono) \wedge Sells(West, Nono, M1)$$

Απόδειξη

- Από τους τύπους

$$American(West) \wedge Weapon(M1) \wedge Nation(Nono) \wedge$$

$$Hostile(Nono) \wedge Sells(West, Nono, M1),$$

$$American(West) \wedge Weapon(M1) \wedge Nation(Nono) \wedge$$

$$Hostile(Nono) \wedge Sells(West, Nono, M1) \Rightarrow Criminal(West)$$

και τον κανόνα Modus Ponens έχουμε το συμπέρασμα μας:

$$Criminal(West)$$

Η Εύρεση Απόδειξης ως Πρόβλημα Αναζήτησης

Η εύρεση απόδειξης μπορεί να διατυπωθεί σαν πρόβλημα αναζήτησης:

- **Αρχική κατάσταση**: η δοσμένη βάση γνώσης KB.
- **Ενέργειες**: οι εφαρμόσιμοι κανόνες συμπερασμού
- **Έλεγχος στόχου**: Η βάση γνώσης KB περιέχει την πρόταση που προσπαθούμε να αποδείξουμε.

Έτσι κάθε αλγόριθμος αναζήτησης που γνωρίζουμε θα μπορούσε να χρησιμοποιηθεί για να βρεθεί μια απόδειξη. Δυστυχώς όμως το πεδίο αναζήτησης είναι άπειρο!

Αναγωγή σε Προτασιακή Λογική

Παρατήρηση: Στο προηγούμενο παράδειγμα είδαμε ότι η χρήση του καθολικού προσδιορισμού και του υπαρξιακού προσδιορισμού, μας οδηγούσε σε **τύπους χωρίς μεταβλητές που είναι ουσιαστικά ισοδύναμοι με τύπους της προτασιακής λογικής.**

Ερώτηση: Μήπως θα μπορούσαμε να μετατρέψουμε οποιοδήποτε σύνολο τύπων της λογικής πρώτης τάξης KB σε ένα σύνολο τύπων της προτασιακής λογικής KB_p εφαρμόζοντας υπαρξιακό προσδιορισμό όπου αυτός είναι απαραίτητος και καθολικό προσδιορισμό για όλες τις δυνατές περιπτώσεις όρων;

Αναγωγή σε Προτασιακή Λογική

Αν η KB δεν περιέχει σύμβολα συναρτήσεων, τότε η KB_p είναι **πεπερασμένη** και μπορούμε **να εφαρμόσουμε ορθούς και πλήρεις κανόνες συμπερασμού της προτασιακής λογικής (ή πίνακες αληθείας)** για να αποφασίσουμε σχετικά με ικανοποιησιμότητα, λογική κάλυψη, λογική ισοδυναμία ή εγκυρότητα.

Στην πράξη θα χρησιμοποιούσαμε κάποιο αποδοτικό αλγόριθμο για το πρόβλημα της ικανοποιησιμότητας.

Αναγωγή σε Προτασιακή Λογική

Αν όμως υπάρχουν σύμβολα συναρτήσεων στην KB, τότε η KB_p μπορεί να είναι **άπειρη**: μπορούμε να εφαρμόσουμε καθολικό προσδιορισμό για όλους τους (άπειρους) όρους που μπορούν να δημιουργηθούν με τα σύμβολα σταθερών και μεταβλητών της KB.

Σ′ αυτή την περίπτωση οι κανόνες συμπερασμού της προτασιακής λογικής δεν μπορούν να βοηθήσουν. Ευτυχώς για μας, υπάρχει το **θεώρημα του Herbrand** το οποίο μπορούμε να χρησιμοποιήσουμε σ′ αυτή την περίπτωση.

Θεώρημα του Herbrand

Το θεώρημα του Herbrand μας λέει το εξής (χωρίς να δίνουμε όλες τις λεπτομέρειες):

> Αν $KB \models \phi$ τότε υπάρχει ένα **πεπερασμένο** υποσύνολο S της KB_p ώστε $S \models \phi$.

Ερώτηση: Και τι γίνεται αν $KB \not\models \phi$; Μπορούμε να το ξέρουμε;

Ημιαποφασιμότητα της Λογικής Κάλυψης

Η απάντηση δυστυχώς είναι **όχι**. Το πρόβλημα της λογικής κάλυψης για την λογική πρώτης τάξης είναι **ημιαποφασίσιμο** (**semi-decidable**) όπως απέδειξαν οι Alan Turing και Alonzo Church (το ίδιο ισχύει και για το πρόβλημα της ικανοποιησιμότητας, της λογικής ισοδυναμίας και της εγκυρότητας).

Δηλαδή, για κάθε αλγόριθμο για το πρόβλημα της λογικής κάλυψης (π.χ., ανάλυση) ισχύει το εξής: Καθώς ο αλγόριθμος τρέχει, δεν μπορούμε να γνωρίζουμε ότι έχει πέσει σε ατέρμονα βρόγχο ή ότι η απόδειξη θα εμφανιστεί την επόμενη χρονική στιγμή.

Απόδειξη

Υπάρχουν διάφορες αποδείξεις στη βιβλιογραφία σχετικά με την ημιαποφασισιμότητα του προβλήματος της λογικής κάλυψης για τη λογική πρώτης τάξης.

Αυτό που αξίζει να παρατηρήσουμε είναι ότι χρειαζόμαστε ελάχιστη από την εκφραστική ικανότητα της λογικής πρώτης τάξης για να φτάσουμε σε ημιαποφασισιμότητα π.χ., **μόνο ένα διμελές κατηγόρημα και καθόλου σύμβολα συναρτήσεων.**

Το αποτέλεσμα αυτό είναι από ένα άρθρο των Kalmar and Suranyi στο Journal of Symbolic Logic, Vol. 15, 1950, pp. 161-173.

Αποδοτικές Μέθοδοι Συμπερασμού

Θα προχωρήσουμε τώρα την παρουσίαση μας και θα παρουσιάσουμε μεθόδους συμπερασμού που μας επιτρέπουν να κάνουμε αποδείξεις με **αποδοτικότερο** τρόπο απ΄ ότι κάναμε μέχρι τώρα.

Πριν προχωρήσουμε θα χρειαστούμε μερικές νέες έννοιες.

Σύνθεση Αντικαταστάσεων

Ορισμός. Έστω οι αντικαταστάσεις $\theta_1 = \{u_1/s_1, \ldots, u_m/s_m\}$ και $\theta_2 = \{v_1/t_1, \ldots, v_n/t_n\}$ που είναι τέτοιες ώστε καμιά μεταβλητή που είναι δεσμευμένη στη θ_1 να μην εμφανίζεται στην θ_2. Η **σύνθεση (composition)** της θ_1 με την θ_2 συμβολίζεται με $COMPOSE(\theta_1, \theta_2)$ και είναι η αντικατάσταση

$$\{u_1/SUBST(\theta_2, s_1), \ldots, u_m/SUBST(\theta_2, s_m), v_1/t_1, \ldots, v_n/t_n\}.$$

Διαισθητικά: Για να υπολογίσουμε τη σύνθεση $COMPOSE(\theta_1, \theta_2)$, εφαρμόζουμε τη θ_2 στους όρους της θ_1 και μετά προσθέτουμε τα ζεύγη της θ_2.

Συμβολισμός: Άλλοι συγγραφείς συμβολίζουν τη σύνθεση της θ_1 με την θ_2 ως εξής: $\theta_1\theta_2$

Παραδείγματα Σύνθεσης

- Έστω $\theta_1 = \{x/y, z/G(w)\}$ και $\theta_2 = \{y/A, w/D\}$. Τότε

$$COMPOSE(\theta_1, \theta_2) = \{x/A, z/G(D), y/A, w/D\}.$$

- Έστω $\theta_1 = \{x/y, z/G(w, y)\}$ και $\theta_2 = \{y/v, w/D\}$. Τότε

$$COMPOSE(\theta_1, \theta_2) = \{x/v, z/G(D, v), y/v, w/D\}.$$

- Έστω $\theta_1 = \{x/y, z/G(w)\}$ και $\theta_2 = \{y/x\}$. Η σύνθεση δεν ορίζεται σε αυτή την περίπτωση επειδή το x είναι δεσμευμένη μεταβλητή στην θ_1 και υπάρχει στην θ_2.

- Έστω $\theta_1 = \{x/F(y), z/y, w/D\}$ και $\theta_2 = \{y/A, v/E\}$. Τότε

$$COMPOSE(\theta_1, \theta_2) = \{x/F(A), z/A, w/D, y/A, v/E\}.$$

Ιδιότητες της Σύνθεσης

Θεώρημα. Έστω οι αντικαταστάσεις θ_1, θ_2 και θ_3 και η έκφραση της λογικής πρώτης τάξης ϕ. Τότε

1. $COMPOSE(\theta_1, \{\}) = COMPOSE(\{\}, \theta_1) = \theta_1$
2. $SUBST(\theta_2, SUBST(\theta_1, \phi)) = SUBST(COMPOSE(\theta_1, \theta_2), \phi)$
   οποτεδήποτε ορίζεται η σύνθεση.
3. $COMPOSE(COMPOSE(\theta_1, \theta_2), \theta_3) =$
   $COMPOSE(\theta_1, COMPOSE(\theta_2, \theta_3))$
   οποτεδήποτε ορίζεται η σύνθεση.

Η Έννοια της Ενοποίησης (Unification)

Ορισμός. Ένα σύνολο εκφράσεων $\{\phi_1, \ldots, \phi_n\}$ είναι **ενοποιήσιμο (unifiable)** αν και μόνο αν υπάρχει μια αντικατάσταση θ που όταν εφαρμοστεί στις εκφράσεις $\phi_1, \ldots, \phi_n$ τις κάνει ίδιες συντακτικά, δηλαδή

$$SUBST(\theta, \phi_1) = \cdots = SUBST(\theta, \phi_n).$$

Σε αυτή την περίπτωση, η θ λέγεται **ενοποιητής (unifier)** των εκφράσεων.

Παραδείγματα Ενοποίησης

- Το σύνολο εκφράσεων $\{P(A, y, z),\ P(x, B, z)\}$ είναι ενοποιήσιμο. Η αντικατάσταση $\{x/A, y/B, z/C\}$ είναι ένας ενοποιητής. Υπάρχουν άλλοι ενοποιητές όπως $\{x/A, y/B, z/F(w)\},\ \{x/A, y/B\}$ κτλ.

- Το σύνολο εκφράσεων $\{P(F(x), A),\ P(y, F(w))\}$ δεν είναι ενοποιήσιμο.

Πιο Γενικοί Ενοποιητές (Most General Unifiers)

Ορισμός. Ένας **πιο γενικός ενοποιητής (most general unifier ή mgu)** ενός συνόλου εκφράσεων S είναι ένας ενοποιητής γ με την ακόλουθη ιδιότητα. Για κάθε ενοποιητή σ του S, υπάρχει μια αντικατάσταση θ τέτοια ώστε $\sigma = COMPOSE(\gamma, \theta)$.

Παραδείγματα:

- Ένας mgu του συνόλου $\{P(A, y, z),\ P(x, B, z)\}$ είναι ο $\{x/A, y/B\}$.

  Άλλοι ενοποιητές προκύπτουν από τον mgu. Για παράδειγμα: $\{x/A, y/B, z/F(w)\} = COMPOSE(\{x/A, y/B\}, \{z/F(w)\})$.

- Ένας mgu του συνόλου $\{P(F(x), z),\ P(y, A)\}$ είναι $\{y/F(x), z/A\}$.

Πιο Γενικοί Ενοποιητές

Παράδειγμα: Ένας mgu των εκφράσεων $P(x)$ και $P(y)$ είναι ο $\{x/y\}$. Ένας άλλος mgu είναι ο $\{y/x\}$.

Ένας πιο γενικός ενοποιητής είναι **μοναδικός** αν δεν λάβουμε υπόψη μας τις πιθανές μετονομασίες μεταβλητών. Γι αυτό συνήθως μιλάμε για τον **τον** πιο γενικό ενοποιητή ενός συνόλου εκφράσεων.

Τώρα θα παρουσιάσουμε τον αλγόριθμο ενοποίησης UNIFY ο οποίος βρίσκει τον πιο γενικό ενοποιητή δύο εκφράσεων της λογικής πρώτης τάξης.

Ένας Αλγόριθμος Ενοποίησης

function UNIFY(x, y) **returns** the mgu of x and y

if $x = y$ **then return** {}

if VARIABLE(x) **then return** UNIFY-VAR(x, y)

if VARIABLE(y) **then return** UNIFY-VAR(y, x)

if CONSTANT(x) **or** CONSTANT(y) **then return** failure

if not(LENGTH(x)=LENGTH(y)) **then return** failure

$i \leftarrow 0$; $\gamma \leftarrow \{\}$

tag **if** i =LENGTH(x)+1 **then return** γ

$\sigma \leftarrow$ UNIFY(PART(x, i),PART(y, i))

if $\sigma =$ failure **then return** failure

$\gamma \leftarrow COMPOSE(\gamma, \sigma)$

$x \leftarrow SUBST(\gamma, x)$

$y \leftarrow SUBST(\gamma, y)$

$i \leftarrow i + 1$

goto tag

Ένας Αλγόριθμος Ενοποίησης

function UNIFY-VAR(x, y) **returns** a substitution

  if x occurs in y **then return** failure

  return { x/y }

Σχόλια για τον UNIFY

- Ο αλγόριθμος UNIFY παρουσιάστηκε πρώτη φορά από τον Robinson το 1965.
- Μπορείτε να χρησιμοποιείτε αυτό τον αλγόριθμο για να υπολογίζετε τον mgu όταν οι δοσμένες εκφράσεις είναι πολύπλοκες.
- Οι είσοδοι του αλγορίθμου UNIFY μπορεί να είναι **σταθερές, μεταβλητές, όροι ή ατομικές προτάσεις.**
- Το **μήκος (length)** ενός όρου ή μιας ατομικής πρότασης είναι ο αριθμός των ορισμάτων της.

Σχόλια για τον UNIFY

- Το i-οστό μέρος ενός όρου ή μιας ατομικής πρότασης x υπολογίζεται από την συνάρτηση PART(x, i).
- Για ένα όρο $F(t_1, \ldots, t_n)$, το **μηδενικό μέρος** του όρου είναι το σύμβολο συνάρτησης F και τα ορίσματα $t_1, \ldots, t_n$ είναι τα υπόλοιπα **μέρη** (πρώτο, δεύτερο, ... n-οστό).
- Για ένα ατομικό τύπο $P(t_1, \ldots, t_n)$, το **μηδενικό μέρος** του τύπου είναι το σύμβολο κατηγορήματος P και τα ορίσματα $t_1, \ldots, t_n$ είναι τα υπόλοιπα **μέρη** (πρώτο, δεύτερο, ... n-οστό).
- Η συνθήκη της εντολής if στη συνάρτηση UNIFY-VAR λέγεται **έλεγχος ύπαρξης (occurs-check)**. Αυτός ο έλεγχος επιβεβαιώνει ότι όροι όπως ο z και ο $F(z)$ δεν ενοποιούνται.

Πολυπλοκότητα του UNIFY

Η πολυπλοκότητα χειρίστης περίπτωσης του αλγόριθμου UNIFY είναι **εκθετική** ως προς το μήκος των εκφράσεων εισόδου.

Παράδειγμα: Αν ενοποιήσουμε τους παρακάτω δύο όρους

$$H(x_1, x_2, \ldots, x_n, F(y_0, y_0), \ldots, F(y_{n-1}, y_{n-1}), y_n)$$

$$H(F(x_0, x_0), F(x_1, x_1), \ldots, F(x_{n-1}, x_{n-1}), y_1, \ldots, y_n, x_n)$$

κάθε μεταβλητή x_i και y_i θα είναι δεσμευμένη σε έναν όρο με $2^{i+1} - 1$ σύμβολα.

Το πρόβλημα εδώ είναι ότι ο πιο γενικός ενοποιητής των παραπάνω όρων περιέχει τους ίδιους υποόρους πολλές φορές.

Πολυπλοκότητα του Unify

Υπάρχουν καλύτεροι αλγόριθμοι ενοποίησης στη βιβλιογραφία (πολυωνυμικού χρόνου).

Η βασική ιδέα σε αυτούς τους αλγορίθμους είναι η χρήση κατάλληλων δομών δεδομένων (π.χ., γράφων) για την αναπαράσταση των εκφράσεων της λογικής πρώτης τάξης και η εφαρμογή των αντικαταστάσεων με προσοχή ώστε να αποφεύγονται προβλήματα όπως αυτά του προηγούμενου παραδείγματος.

Γενικευμένος Τρόπος του Θέτειν

Ο κανόνας συμπερασμού **γενικευμένος τρόπος του θέτειν (Generalized Modus Ponens, GMP)** είναι ο εξής.

Αν οι τύποι $p_1, \ldots, p_n, p'_1, \ldots, p'_n, q$ είναι ατομικοί και υπάρχει μια αντικατάσταση θ τέτοια ώστε για κάθε $1 \leq i \leq n$,
$SUBST(\theta, p'_i) = SUBST(\theta, p_i)$ τότε:

$$\frac{p'_1,\ p'_2,\ \ldots,\ p'_n,\ (p_1 \wedge p_2 \wedge \cdots \wedge p_n \Rightarrow q)}{SUBST(\theta, q)}$$

Παράδειγμα

Από τους τύπους

$$Missile(M1),\ Owns(Nono, M1)$$

και

$$Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, Nono, x)$$

μπορούμε να συμπεράνουμε με GMP

$$Sells(West, Nono, M1).$$

Η αντικατάσταση θ σε αυτήν την περίπτωση είναι $\{x/M1\}$.

Γενικευμένος Τρόπος του Θέτειν

Σχόλια:

- Ο γενικευμένος τρόπος του θέτειν κάνει σε ένα βήμα ό,τι θα έκαναν σε δύο βήματα οι εξής κανόνες συμπερασμού: εισαγωγή σύζευξης και τρόπος του θέτειν.
- Ο γενικευμένος τρόπος του θέτειν είναι μια **ανυψωμένη (lifted)** μορφή του τρόπου του θέτειν: ανυψώνει τον κανόνα αυτό από την προτασιακή λογική στην λογική πρώτης τάξης.
- Ο γενικευμένος τρόπος του θέτειν εφαρμόζεται σε **φράσεις Horn (Horn clauses)**.

Οι φράσεις Horn αποτελούν ένα πολύ ενδιαφέρον υποσύνολο της λογικής πρώτης τάξης που θα ορίσουμε λεπτομερώς παρακάτω.

Ορθότητα του GMP

Ένας κανόνας συμπερασμού λέγεται **ορθός (sound)** αν οδηγεί μόνο σε προτάσεις που καλύπτονται λογικά. Δηλαδή, αν $KB \vdash \alpha$ τότε $KB \models \alpha$.

Θεώρημα. Ο GMP είναι ένας **ορθός** κανόνας συμπερασμού για βάσεις γνώσεων που αποτελούνται από φράσεις Horn.

Λεκτικά (Literals)

Ορισμός. Ένα **λεκτικό (literal)** είναι ένας ατομικός τύπος ή η άρνηση ενός ατομικού τύπου. Στην πρώτη περίπτωση έχουμε ένα **θετικό** λεκτικό και στη δεύτερη περίπτωση ένα **αρνητικό** λεκτικό.

Παραδείγματα:

$$Drives(John, BMW),\ \neg Drives(John, BMW),\ Drives(x, BMW),$$

$$Loves(Mary, FatherOf(Mary)),\ \neg P(x, F(G(x)))$$

Φράσεις Horn (Horn Clauses)

Ορισμός. Μια **φράση (clause)** είναι μια διάζευξη λεκτικών. Μια φράση **Horn** είναι μια διάζευξη λεκτικών από τα οποία το πολύ ένα είναι θετικό.

Δηλαδή, μια φράση Horn απαντάται στις εξής μορφές:

$$q$$

$$\neg p_1 \vee \neg p_2 \vee \ldots \vee \neg p_n \vee q \quad (\text{ή} \quad p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)$$

$$\neg p_1 \vee \neg p_2 \vee \ldots \vee \neg p_n$$

όπου $p_1, \ldots, p_n, q$ είναι ατομικοί τύποι.

Συμβολισμός: Σε μια φράση Horn, θεωρούμε ότι οι (ελεύθερες) μεταβλητές των τύπων $p_1, \ldots, p_n, q$ εισάγονται από αντίστοιχους καθολικούς ποσοδείκτες που εμφανίζονται στην αρχή του τύπου.

Όρολογία

Η ελληνική μετάφραση του βιβλίου AIMA που χρησιμοποιούμε χρησιμοποιεί τον όρο **πρόταση Horn** μεταφράζοντας την Αγγλική λέξη clause με την Ελληνική λέξη 'πρόταση'.

Προτίμησα να χρησιμοποιήσω τον όρο **φράση Horn** επειδή χρησιμοποιώ τον όρο πρόταση για να αναφέρομαι στους τύπους της λογικής πρώτης τάξης που δεν έχουν ελεύθερες μεταβλητές.

Παραδείγματα

Οι τύποι

$$Drives(John, BMW)$$

$$Drives(x, BMW)$$

$$Person(x) \Rightarrow Animal(x)$$

$$Person(x) \wedge Knows(John, x) \Rightarrow Loves(John, x)$$

$$\neg Person(x) \vee \neg Knows(John, x)$$

είναι φράσεις Horn.

Ο τύπος

$$Drives(John, BMW) \vee Drives(John, Porsche)$$

δεν είναι φράση Horn.

Παραδείγματα

Οι προηγούμενες φράσεις Horn είναι ισοδύναμες με τους ακόλουθους τύπους της λογικής πρώτης τάξης:

$$Drives(John, BMW)$$

$$(\forall x) Drives(x, BMW)$$

$$(\forall x)(Person(x) \Rightarrow Animal(x))$$

$$(\forall x)(Person(x) \wedge Knows(John, x) \Rightarrow Loves(John, x))$$

$$(\forall x)(\neg Person(x) \vee \neg Knows(John, x))$$

Γεγονότα

Οι φράσεις Horn που αποτελούνται από ένα ατομικό τύπο λέγονται **γεγονότα (facts)**.

Παραδείγματα:

$$Drives(John, BMW)$$

$$Drives(x, BMW)$$

Κανόνες

Οι φράσεις Horn της μορφής

$$p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q$$

λέγονται **κανόνες (rules)**.

Στον παραπάνω κανόνα, ο ατομικός τύπος q λέγεται **κεφαλή (head)** του κανόνα και η σύζευξη $p_1 \wedge p_2 \wedge \ldots \wedge p_n$ λέγεται **σώμα (body)** του κανόνα.

Η ορολογία αυτή χρησιμοποιείται στη γλώσσα λογικού προγραμματισμού Prolog.

Παραδείγματα Κανόνων

$$Person(x) \Rightarrow Animal(x)$$

$$Person(x) \wedge Knows(John, x) \Rightarrow Loves(John, x)$$

Ερωτήσεις ή Περιορισμοί Ακεραιότητας

Οι φράσεις Horn της μορφής

$$\neg p_1 \vee \neg p_2 \vee \ldots \vee \neg p_n$$

προκύπτουν από **ερωτήσεις (queries)** ή **περιορισμούς ακεραιότητας (integrity constraints)** σε **συστήματα λογικού προγραμματισμού**.

Παράδειγμα: $\neg Person(x) \vee \neg Knows(John, x)$

Οριστικές Φράσεις Horn

Αν μια φράση Horn έχει ακριβώς ένα θετικό λεκτικό, τότε λέγεται **οριστική (definite)**. Δηλαδή, τα γεγονότα και οι κανόνες είναι οι οριστικές φράσεις Horn.

Παραδείγματα:

$$Drives(John, BMW)$$

$$Drives(x, BMW)$$

$$Person(x) \Rightarrow Animal(x)$$

$$Person(x) \wedge Knows(John, x) \Rightarrow Loves(John, x)$$

Datalog

Άν μια βάσης γνώσης αποτελείται από οριστικές φράσεις Horn και δεν περιέχει σύμβολα συναρτήσεων τότε θα λέγεται βάση **Datalog**.

Η Datalog ορίστηκε αρχικά σαν γλώσσα επερωτήσεων για σχεσιακές βάσεις δεδομένων εμπνευσμένη από το λογικό προγραμματισμό.

Ας δώσουμε τώρα μερικά παραδείγματα συνηθισμένων βάσεων γνώσεων με φράσεις Horn.

Παράδειγμα Datalog: Οικογενειακές Σχέσεις

$Parent(Pam, Bob)$ $Parent(Tom, Bob)$ $Parent(Tom, Liz)$

$Parent(Bob, Ann)$ $Parent(Bob, Pat)$ $Parent(Pat, Jim)$

$Male(Tom)$ $Male(Bob)$ $Male(Jim)$

$Female(Liz)$ $Female(Ann)$ $Female(Pam)$ $Female(Pat)$

Παράδειγμα Datalog

$$Parent(x, y) \Rightarrow Offspring(y, x)$$

$$Parent(x, y) \wedge Female(x) \Rightarrow Mother(x, y)$$

$$Parent(x, y) \wedge Parent(y, z) \Rightarrow Grandparent(x, z)$$

Παράδειγμα Datalog: Μονοπάτια σε ένα Γράφο

$$Arc(x, y) \Rightarrow Path(x, y)$$

$$Path(x, y) \wedge Arc(y, z) \Rightarrow Path(x, z)$$

$$Arc(A, B)\ Arc(B, C)\ Arc(C, D)\ Arc(B, E)$$

Παράδειγμα: Λίστες

Ας θεωρήσουμε το παρακάτω λεξιλόγιο:

- Το σύμβολο σταθεράς $Empty$ που συμβολίζει την κενή λίστα.
- Το σύμβολο δυαδικής συνάρτησης $Cons$ (από την αγγλική λέξη construct).

  Ο όρος $Cons(h, t)$ συμβολίζει τη λίστα που έχει πρώτο στοιχείο το h και τα υπόλοιπα στοιχεία της απαρτίζουν τη λίστα t.
- Το σύμβολο κατηγορήματος $Member$.

  Το γεγονός $Member(e, l)$ δηλώνει ότι το e είναι στοιχείο της λίστας l.

Παράδειγμα: Λίστες

Οι ακόλουθες φράσεις Horn εκφράζουν ιδιότητες του $Member$:

$$Member(e, Cons(e, t))$$

$$Member(e, t) \Rightarrow Member(e, Cons(h, t))$$

Οι παραπάνω κανόνες είναι οριστικοί αλλά όχι Datalog επειδή περιέχουν το σύμβολο συνάρτησης $Cons$.

Μια Εφαρμογή του GMP

Μπορούμε να λύσουμε το πρόβλημα με τον Συνταγματάρχη West χρησιμοποιώντας GMP; Ναι, αν είναι δυνατόν να γράψουμε την βάση γνώσης μας με φράσεις Horn. Ας το δοκιμάσουμε!

Στην επόμενη ενότητα του μαθήματος, θα δώσουμε έναν γενικό αλγόριθμο που μπορεί να χρησιμοποιηθεί για να μετατρέψει ένα τυχαίο τύπο της λογικής πρώτης τάξης σε ένα ισοδύναμο σύνολο φράσεων Horn (όταν υπάρχει τέτοιο ισοδύναμο σύνολο).

Παράδειγμα

- ... είναι έγκλημα για έναν Αμερικανό να πουλάει όπλα σε εχθρικά κράτη.

$$American(x) \wedge Weapon(y) \wedge Nation(z) \wedge$$

$$Hostile(z) \wedge Sells(x, z, y) \Rightarrow Criminal(x)$$

- Το κράτος Nono ...

$$Nation(Nono)$$

- ... που είναι εχθρός της Αμερικής ...

$$Enemy(Nono, America)$$

Παράδειγμα

- ... έχει πυραύλους.

$$Owns(Nono, M1), \quad Missile(M1)$$

Παρατήρηση: Επειδή οι φράσεις Horn δεν επιτρέπουν τη χρήση υπαρξιακού ποσοδείκτη, χρησιμοποιήσαμε την σταθερά $M1$ για τον ένα τουλάχιστον πύραυλο για τον οποίο μας λέει η παραπάνω πρόταση.

Παράδειγμα

- Όλους αυτούς τους πυραύλους τους έχει πουλήσει στο Nono ο συνταγματάρχης West

$$Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x)$$

- ... ο οποίος είναι Αμερικανός.

$$American(West)$$

Παράδειγμα

- Η Αμερική είναι κράτος.

$$Nation(America)$$

- Οι πύραυλοι είναι όπλα.

$$Missile(x) \Rightarrow Weapon(x)$$

- 'Εχθρός της Αμερικής ' σημαίνει 'εχθρικό κράτος '.

$$Enemy(x, America) \Rightarrow Hostile(x)$$

Απόδειξη

- Από $Missile(M1)$ και $Missile(x) \Rightarrow Weapon(x)$, χρησιμοποιώντας GMP:

$$Weapon(M1)$$

- Από $Enemy(x, America) \Rightarrow Hostile(x)$ και $Enemy(Nono, America)$, χρησιμοποιώντας GMP:

$$Hostile(Nono)$$

- Από $Owns(Nono, M1),\ Missile(M1)$ και

$$Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x)$$

χρησιμοποιώντας GMP:

$$Sells(West, Nono, M1)$$

Απόδειξη

- Από
  $American(West), Weapon(M1), Nation(Nono), Hostile(Nono),$
  $Sells(West, Nono, M1)$ και

$$American(x) \wedge Weapon(y) \wedge Nation(z) \wedge$$

$$Hostile(z) \wedge Sells(x, z, y) \Rightarrow Criminal(x),$$

  χρησιμοποιώντας GMP:

$$Criminal(West)$$

Αλγόριθμοι Συμπερασμού με Χρήση του GMP

Η παραπάνω απόδειξη μας δείχνει ότι ο συμπερασμός με χρήση του GMP είναι φυσικός και εύκολος στο να τον ακολουθήσει κανείς.

Θα παρουσιάσουμε τώρα δύο σημαντικούς αλγόριθμους συμπερασμού για τη λογική πρώτης τάξης που βασίζονται σε GMP:

- **Προς τα εμπρός αλυσίδα εκτέλεσης (forward chaining).**
- **Προς τα πίσω αλυσίδα εκτέλεσης (backward chaining).**

Οι αλγόριθμοι αυτοί θα παρουσιαστούν μόνο για την περίπτωση βάσεων γνώσης που αποτελούνται από **οριστικές φράσεις Horn**.

Πριν προχωρήσουμε όμως χρειαζόμαστε ένα τεχνικό ορισμό ακόμα.

Παράδειγμα

Θεωρήστε την παρακάτω βάση γνώσης KB:

$$Likes(x, Mary)$$

$$Likes(John, x) \Rightarrow GreetsCheerfully(John, x)$$

Θα περιμέναμε να ακολουθεί λογικά από την KB ο τύπος:

$$GreetsCheerfully(John, Mary)$$

Όμως ο GMP δε μπορεί να χρησιμοποιηθεί για να συμπεράνουμε αυτό τον τύπο γιατί οι ατομικοί τύποι $Likes(x, Mary)$ και $Likes(John, x)$ **δεν ενοποιούνται!**

Προτυποποίηση Μεταβλητών

Το προηγούμενο πρόβλημα μπορεί να επιλυθεί με **προτυποποίηση των μεταβλητών:**

1. Καμία μεταβλητή δεν πρέπει να υπάρχει σε περισσότερους από μια φράσεις Horn στην αρχική βάση γνώσης.
2. Οποτεδήποτε έχουμε να εφαρμόσουμε τον GMP, πρώτα μετονομάζουμε τις μεταβλητές των φράσεων Horn που θα χρησιμοποιήσουμε. Τα νέα ονόματα μεταβλητών δεν πρέπει να έχουν χρησιμοποιηθεί ενωρίτερα.

Προς τα Εμπρός Αλυσίδα Εκτέλεσης

Ο αλγόριθμος **προς τα εμπρός αλυσίδα εκτέλεσης (forward chaining)** ξεκινάει με τις φράσεις στη βάση γνώσης και παράγει συμπεράσματα χρησιμοποιώντας τον GMP. Αυτά τα συμπεράσματα με τη σειρά τους οδηγούν σε περισσότερα συμπεράσματα που μπορούν να εξαχθούν από τη βάση γνώσης κ.ο.κ.

Όταν εφαρμόζουμε τον αλγόριθμο δεν πρέπει να ξεχνάμε την **προτυποποίηση** των μεταβλητών όπως συζητήσαμε προηγουμένως.

Forward Chaining

Ο αλγόριθμος forward chaining μπορεί να χρησιμοποιηθεί ως εξής:

- Ως μια **οδηγούμενη από τα δεδομένα (data-driven)** διαδικασία. Όταν ένα γεγονός προστίθεται στη βάση γνώσης, ο αλγόριθμος εκτελείται για να συμπεράνουμε νέα γεγονότα. Τα γεγονότα αυτά προστίθενται επίσης στη βάση γνώσης, και η διαδικασία συνεχίζεται.

Forward Chaining

- Ως μια διαδικασία **απάντησης ερωτημάτων (query answering)**. Όταν θέλουμε να μάθουμε αν ένας ατομικός τύπος α (πιθανά με ελεύθερες μεταβλητές $x_1, \ldots, x_n$) ακολουθεί λογικά από μια βάση γνώσης, μπορούμε να εφαρμόσουμε forward chaining για να παράγουμε νέα συμπεράσματα, μέχρι να παραχθεί ένας ατομικός τύπος που **ενοποιείται** με τον α.

  Έτσι δουλεύει η συνάρτηση FOL-FC-Ask που παρουσιάζουμε στην επόμενη διαφάνεια. Η αντικατάσταση σ μας δίνει την **πρώτη** απάντηση στο ερώτημα α: η απάντηση ορίζεται από το σύνολο των δεσμεύσεων των μεταβλητών $x_1, \ldots, x_n$ της α.

  Μπορούμε να συνεχίσουμε να εφαρμόζουμε forward chaining για να πάρουμε περισσότερες απαντήσεις.

Forward Chaining

function FOL-FC-ASK(KB, α) **returns** a substitution or false

inputs: KB, the knowledge base, a set of first-order definite clauses

α, the query, an atomic sentence

local variables: new, the new sentences inferred on each iteration

repeat until new is empty

$new \leftarrow \{\}$

for each clause r in KB **do**

Standardize the variables of r to transform it into $p_1 \wedge \ldots \wedge p_n \Rightarrow q$

for each θ such that $SUBST(\theta, p_1 \wedge \ldots \wedge p_n) = SUBST(\theta, p'_1 \wedge \ldots \wedge p'_n)$

for some $p'_1, \ldots, p'_n$ in KB **do**

$q' \leftarrow SUBST(\theta, q)$

if q' is not a renaming of some sentence already in KB or new **then do**

add q' to new

$\sigma \leftarrow$ UNIFY(q', α)

if σ is not $failure$ **then return** σ

add new to KB

return $false$

Παραδείγματα

Έστω η KB

$$Missile(M1)$$

$$Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x).$$

Αν η πρόταση $Owns(Nono, M1)$ εισαχθεί στην KB, τότε μπορούμε να εφαρμόσουμε τον αλγόριθμο forward chaining για να συμπεράνουμε τον τύπο

$$Sells(West, Nono, M1)$$

και να τον προσθέσουμε στην KB. Κανένας άλλος συμπερασμός δεν μπορεί να πραγματοποιηθεί στη συνέχεια, άρα ο αλγόριθμος σταματάει.

Παραδείγματα

Έστω η KB

$$Missile(M1),\ Owns(Nono, M1),$$

$$Missile(M2),\ Owns(Nono, M2),$$

$$Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x).$$

Έστω ότι θέλουμε να μάθουμε αν από την KB έπεται λογικά ότι υπάρχει κάποιος που έχει πουλήσει κάποιο όπλο στο κράτος Nono. Αυτό το **ερώτημα** μπορεί να αναπαρασταθεί από τον τύπο

$$Sells(z, Nono, w)$$

με ελεύθερες μεταβλητές z και w.

Παραδείγματα

Η συνάρτηση FOL-FC-Ask μπορεί να εφαρμοστεί για να δείξουμε ότι το γεγονός

$$Sells(West, Nono, M1)$$

ακολουθεί λογικά από την KB. Αυτό το νέο γεγονός ενοποιείται με τον τύπο

$$Sells(z, Nono, w)$$

με αντικατάσταση

$$\{z/West, w/M1\}$$

και δίνει μια **απάντηση** στο ερώτημα.

Παραδείγματα

Ο αλγόριθμος μπορεί να εφαρμοστεί ξανά για να δείξουμε ότι το γεγονός

$$Sells(West, Nono, M2)$$

ακολουθεί λογικά από την KB. Αυτό το νέο γεγονός ενοποιείται με τον τύπο

$$Sells(z, Nono, w)$$

με αντικατάσταση

$$\{z/West, w/M2\}$$

και δίνει μια νέα απάντηση στο ερώτημα.

Οι δυο αντικαταστάσεις που υπολογίστηκαν παραπάνω είναι το **σύνολο απαντήσεων** στο ερώτημα:

$$\{z/West, w/M1\}, \quad \{z/West, w/M2\}$$

Παραδείγματα

Έστω η KB

$$Q(x) \Rightarrow Q(F(x))$$

και το γεγονός $Q(A)$ εισάγεται στην KB.

Τότε ο αλγόριθμος forward chaining μπορεί να εφαρμοστεί για να προστεθεί το γεγονός $Q(F(A))$ στην KB.

Παραδείγματα

Αλλά τώρα το $Q(F(A))$ ενοποιείται με την υπόθεση του κανόνα

$$Q(x) \Rightarrow Q(F(x))$$

και ο GMP μπορεί να εφαρμοστεί ξανά.

Σε αυτό το σημείο ο αλγόριθμος forward chaining πέφτει σε **ατέρμονα βρόγχο** προσθέτοντας στη βάση γνώσης τα ακόλουθα γεγονότα:

$$Q(F(F(A))),\ Q(F(F(F(A)))),\ \ldots$$

Ορθότητα και Πληρότητα του Forward Chaining

Η συνάρτηση FOL-FC-Ask είναι **ορθή** επειδή σε κάθε βήμα συμπερασμού απλά εφαρμόζει τον GMP και ο GMP είναι ορθός.

Η συνάρτηση FOL-FC-Ask είναι **πλήρης** για βάσεις γνώσεων που είναι σε μορφή Datalog, δηλαδή μπορεί να χρησιμοποιηθεί για να βρεθούν όλες οι απαντήσεις μιας ερώτησης.

Αν η βάση γνώσεων αποτελείται από οριστικές φράσεις Horn που έχουν σύμβολα συνάρτησης, τότε ο αλγόριθμος forward chaining μπορεί να πέσει σε ατέρμονα βρόγχο όπως είδαμε παραπάνω.

Το Πρόβλημα της Λογικής Κάλυψης

Έστω KB ένα σύνολο φράσεων Horn και ϕ ένας ατομικός τύπος.

Το πρόβλημα της λογικής κάλυψης (δηλ. η ερώτηση $KB \models \phi$;) για οριστικές φράσεις Horn είναι **ημιαποφασίσιμο** όπως και για την γενική περίπτωση της λογικής πρώτης τάξης.

Το πρόβλημα της λογικής κάλυψης (δηλ. η ερώτηση $KB \models \phi$;) όταν η KB είναι βάση Datalog μπορεί να λυθεί σε **πολυωνυμικό χρόνο** χρησιμοποιώντας τον αλγόριθμο forward chaining.

Θέματα Υλοποίησης και Απόδοσης

Για να υλοποιηθεί ο αλγόριθμος forward chaining, πρέπει να απαντήσουμε στις εξής ερωτήσεις:

- Ποιο **κανόνα** να χρησιμοποιήσουμε πρώτο στην περίπτωση που έχουμε περισσότερους του ενός κανόνες;
- Αφού διαλέξουμε κανόνα, με ποιά σειρά να εξετάσουμε τους **τύπους του σώματος του κανόνα** για να βρούμε αν το σώμα του κανόνα ενοποιείται με γεγονότα της βάσης;
- Ποιο **γεγονός** να χρησιμοποιήσουμε πρώτο σε περίπτωση που έχουμε περισσότερα του ενός κατάλληλα γεγονότα;

Η σωστή απάντηση των παραπάνω ερωτήσεων μπορεί να κάνει τη διαφορά στην **απόδοση** του αλγόριθμου.

Άλλα Θέματα Απόδοσης

Ο αλγόριθμος forward chaining μπορεί να εφαρμοστεί πολύ αποδοτικά αν έχουμε καλές τεχνικές για τα εξής:

- Αποφυγή της επανεξέτασης κάθε κανόνα σε κάθε επανάληψη (αφού μόνο μερικά γεγονότα προστίθενται στη βάση γνώσης σε κάθε επανάληψη).
- Αποφυγή δημιουργίας γεγονότων που είναι άσχετα με το ερώτημα (όταν η προς τα εμπρός αλυσίδα εκτέλεσης χρησιμοποιείται ως αλγόριθμος απάντησης ερωτημάτων).

Θέματα Υλοποίησης και Απόδοσης

Αυτά τα θέματα έχουν μελετηθεί διεξοδικά στις εξής περιοχές της Τεχνητής Νοημοσύνης και των Βάσεων Δεδομένων:

- **Συστήματα παραγωγών (production systems).**
- **Συστήματα ενεργών βάσεων δεδομένων (active databases).**
- **Συστήματα επαγωγικών βάσεων δεδομένων (deductive databases).**
- **Συστήματα κανόνων** σε υπάρχοντα σχεσιακά συστήματα βάσεων δεδομένων (π.χ., στην Oracle).

Προς τα Πίσω Αλυσίδα Εκτέλεσης

Ο αλγόριθμος της **προς τα πίσω αλυσίδας εκτέλεσης (backward chaining)** ξεκινάει με ένα γεγονός που θέλουμε να αποδείξουμε, βρίσκει ένα κανόνα που μας επιτρέπει να το συμπεράνουμε, και προσπαθεί να αποδείξει ότι ισχύει το σώμα αυτού του κανόνα.

Προς τα Πίσω Αλυσίδα Εκτέλεσης

function FOL-BC-Ask($KB, goals, \theta$) **returns** a set of substitutions

inputs: KB, a knowledge base

$goals$, a list of conjuncts forming a query (θ already applied)

θ the current substitution, initially the empty substitution

local variables: $answers$, a set of substitutions, initially empty

if $goals$ is empty **then return** $\{\theta\}$

$q' \leftarrow$ Subst(θ,First($goals$))

for each formula r in KB which after standardization of variables becomes $p_1 \wedge \ldots \wedge p_n \Rightarrow q$ and $\theta' \leftarrow$ Unify(q', q) succeeds **do**

$new\_goals \leftarrow [p_1, \ldots, p_n | \text{Rest}(goals)]$

$answers \leftarrow$FOL-BC-Ask(KB, new_goals,Compose(θ, θ')) $\cup answers$

return $answers$

Σχόλια

Ο αλγόριθμος FOL-BC-Ask μπορεί να χρησιμοποιηθεί για να βρούμε το σύνολο των αντικαταστάσεων σ που είναι τέτοιες ώστε $KB \models SUBST(\sigma, \phi)$ όπου ϕ είναι η σύζευξη που αντιστοχεί στην λίστα $goals$.

Με άλλα λόγια, ο αλγόριθμος FOL-BC-Ask μπορεί να χρησιμοποιηθεί για να βρούμε όλες τις **απαντήσεις** σε ένα **ερώτημα** ϕ που τίθεται στη βάση γνώσης KB.

Η **απάντηση** στο ερώτημα ϕ είναι ένα σύνολο αντικαταστάσεων που αποτελούνται από τις δεσμεύσεις για τις μεταβλητές του ερωτήματος ϕ.

Σχόλια

Ο αλγόριθμος της προς τα πίσω αλυσίδας εκτέλεσης απαιτεί **προτυποποίηση** των μεταβλητών όπως περιγράψαμε νωρίτερα για τον GMP.

Ο αλγόριθμος FOL-BC-Ask είναι ένας αλγόριθμος **αναζήτησης πρώτα κατά βάθος (depth-first search)** οπότε έχει όλα τα καλά και κακά χαρακτηριστικά αυτών των αλγορίθμων.

Για την **υλοποίηση** και την **αποδοτικότητα** του FOL-BC-Ask εγείρονται παρόμοια θέματα με αυτά που συζητήσαμε στην περίπτωση του αλγόριθμου forward chaining.

Λογικός Προγραμματισμός

Όλες οι γλώσσες **λογικού προγραμματισμού** (π.χ., Prolog) και τα **συστήματα παραγωγικών βάσεων δεδομένων (deductive databases)** βασίζονται σε κατάλληλες υλοποιήσεις του αλγόριθμου backward chaining.

Παραδείγματα

Έστω η βάση γνώσης KB

$$Q(A),\ Q(B)$$

και το ερώτημα $Q(x)$.

Η συνάρτηση FOL-BC-Ask$(KB, [Q(x)], \{\})$ επιστρέφει το σύνολο αντικαταστάσεων

$$\{\{x/A\}, \{x/B\}\}.$$

Παραδείγματα

Σ′ αυτό το παράδειγμα, ο αλγόριθμος FOL-BC-Ask λειτουργεί ως εξής.

Στο σώμα της εντολής **for each** γίνονται οι αναδρομικές κλήσεις FOL-BC-Ask$(KB, [], \{x/A\})$ και FOL-BC-Ask$(KB, [], \{x/B\})$ που επιστρέφουν τα σύνολα αντικαταστάσεων $\{\{x/A\}\}$ και $\{\{x/B\}\}$ αντίστοιχα. Οι κλήσεις αυτές αντιστοιχούν στους τύπους $Q(A)$ και $Q(B)$ που υπάρχουν στην KB.

Το τελικό αποτέλεσμα δημιουργείται από την ένωση των παραπάνω συνόλων στην προτελευταία γραμμή του αλγόριθμου.

Παραδείγματα

Έστω η βάση γνώσης KB

$$Q(A),\ Q(B),\ R(A,C),\ R(B,D)$$

και το ερώτημα $Q(x) \wedge R(x,y)$.

Η συνάρτηση FOL-BC-Ask$(KB, [Q(x), R(x,y)], \{\})$ επιστρέφει

$$\{\{x/A, y/C\}, \{x/B, y/D\}\}.$$

Παραδείγματα

Σ′ αυτό το παράδειγμα, ο αλγόριθμος FOL-BC-Ask λειτουργεί ως εξής.

Αρχικά η θ είναι ίση με $\{\}$ και το q' γίνεται $Q(x)$. Μετά εκτελείται το σώμα της εντολής **for each** για $r = Q(A)$ και $\theta' = \{x/A\}$ και γίνεται η αναδρομική κλήση FOL-BC-Ask$(KB, [R(x, y)], \{x/A\})$ μια που

$$COMPOSE(\{\}, \{x/A\}) = \{x/A\}.$$

Ας δούμε τώρα τι συμβαίνει σ′ αυτή την αναδρομική κλήση. Η θ είναι ίση με $\{x/A\}$ και το q' γίνεται $R(A, y)$. Μετά εκτελείται το σώμα της εντολής **for each** για $r = R(A, C)$ και $\theta' = \{y/C\}$ και γίνεται η αναδρομική κλήση FOL-BC-Ask$(KB, [], \{y/C\})$ μια που

$$COMPOSE(\{x/A\}, \{y/C\}) = \{x/A, y/C\}.$$

Η κλήση αυτή επιστρέφει $\{\{x/A, y/C\}\}$.

Παραδείγματα

Άρα η κλήση FOL-BC-Ask$(KB, [R(x, y)], \{x/A\})$ επιστρέφει το ίδιο σύνολο αντικαταστάσεων και αυτό αποθηκεύεται στη μεταβλητή $answers$.

Μετά ακολουθεί η εκτέλεση της **for each** για $r = Q(B)$ και με ομοιο τρόπο δημιουργείται το σύνολο αντικαταστάσεων $\{\{x/A, y/D\}\}$ το οποίο προστίθεται στο σύνολο $answers$.

Έτσι έχουμε το τελικό αποτέλεσμα.

Παραδείγματα

Έστω η βάση γνώσης KB

$$Missile(M1),\ Owns(Nono, M1),$$

$$Owns(Nono, x) \wedge Missile(x) \Rightarrow Sells(West, Nono, x)$$

και το ερώτημα $Sells(West, Nono, y)$.

Η συνάρτηση FOL-BC-Ask$(KB, [Sells(West, Nono, y)], \{\})$ επιστρέφει

$$\{\{y/M1\}\}.$$

Μπορείτε να εξηγήσετε γιατί προσομοιώνοντας την εκτέλεση της συνάρτησης FOL-BC-Ask;

Παραδείγματα

Έστω η βάση γνώσης KB

$$P(3) \Rightarrow P(3)$$

και το ερώτημα $P(x)$ ή το ερώτημα $P(3)$.

Οι κλήσεις FOL-BC-Ask$(KB, [P(x)], \{\})$ και
FOL-BC-Ask$(KB, [P(3)], \{\})$ πέφτουν σε **ατέρμονα βρόγχο**.

Παραδείγματα

Προσέξτε ότι αυτή η βάση γνώσης δεν έχει συναρτησιακά σύμβολα!

Είμαστε στην περίπτωση της Datalog και μπορούμε να υπολογίσουμε **όλα** τα γεγονότα που ακολουθούν λογικά από μια βάση γνώσης (π.χ., χρησιμοποιώντας forward chaining). Οπότε η ανεπιθύμητη συμπεριφορά οφείλεται στη **δοσμένη βάση γνώσης**, και στο ότι η συνάρτηση FOL-BC-Ask δουλεύει με **πρώτα κατά βάθος** τρόπο.

Παραδείγματα

$$American(West)$$

$$Enemy(Nono, America)$$

$$Owns(Nono, M1),\ Missile(M1)$$

$$Enemy(x, America) \Rightarrow Hostile(x)$$

$$Missile(x) \Rightarrow Weapon(x)$$

$$Missile(x) \wedge Owns(Nono, x) \Rightarrow Sells(West, x, Nono)$$

$$American(x) \wedge Weapon(y) \wedge$$

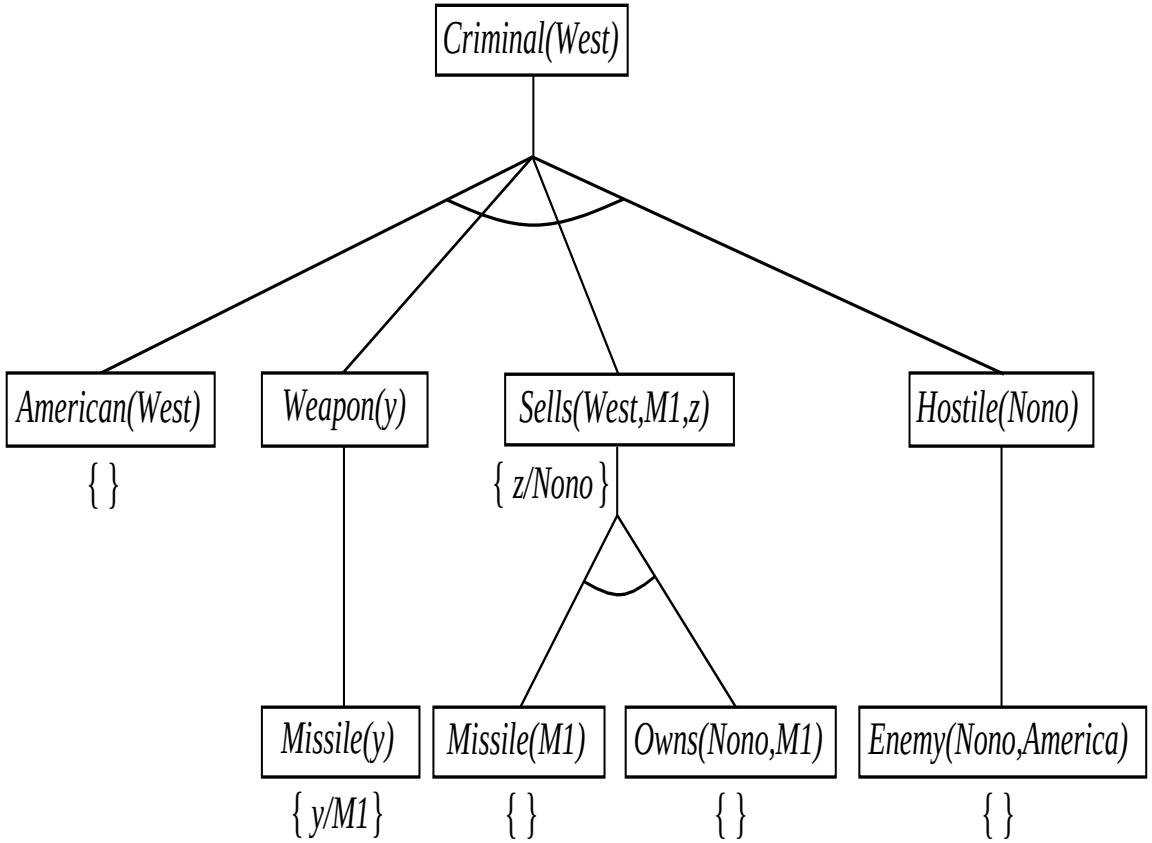
$$Sells(x, y, z) \wedge Hostile(z) \Rightarrow Criminal(x)$$

Παραδείγματα

Προσοχή: Στην προηγούμενη βάση γνώσης το κατηγόρημα $Sells(z, w, v)$ παριστάνει ότι ο z πουλάει το w στον v (δηλαδή, σε σχέση με τα προηγούμενα παραδείγματα που αφορούσαν τον West, έχουμε εναλλάξει τα δύο τελευταία ορίσματα του $Sells$).

Θα παρουσιάσουμε τώρα το δένδρο απόδειξης που αποδεικνύει το γεγονός $Criminal(West)$. Τα δένδρα απόδειξης είναι ένας δημοφιλής γραφικός τρόπος να εξηγήσουμε την λειτουργία του αλγόριθμου backward chaining.

Δένδρο Απόδειξης (Proof Tree)



Μελέτη

- ΑΙΜΑ, Κεφάλαιο 9.
- M. Genesereth, N. Nilsson. Logical Foundations of Artificial Intelligence. Morgan Kaufmann, 1987.

  Παράγραφος 4.2.

  Ο αλγόριθμος ενοποίησης που δώσαμε προέρχεται από αυτό το βιβλίο.