METODY DOWODZENIA TWIERDZEŃ I AUTOMATYZACJA ROZUMOWAŃ

WYKŁAD 4: POSTACIE NORMALNE I PREFIKSOWE. SKOLEMIZACJA.

III rok kognitywistyki UAM, 2016–2017

Plan na dziś:

1. Semantyczna równoważność formuł
2. Postacie normalne: koniunkcyjna oraz alternatywna
3. Postacie prefiksowe
4. Skolemizacja

Pewne dodatkowe informacje (układy funkcji prawdziwościowych, twierdzenie Posta) znajdą zainteresowani słuchacze w prezentacji (slajdy 25–42):
http://logic.amu.edu.pl/images/9/98/Mdt3xi2015.pdf

Przypomnijmy, że w sprawie notacji logicznej autorzy reprezentują różne stanowiska:

1. Podejście leniwe (np. Fitting 1990): używanie tych samych symboli na oznaczenie funktorów (spójników) prawdziwościowych oraz odpowiadających im funkcji prawdziwościowych.
2. Podejście skrupulatne (np. Batóg, T. 1999. *Podstawy logiki*. Wydawnictwo Naukowe UAM, Poznań): osobne symbole dla funktorów (w języku przedmiotowym) oraz funkcji prawdziwościowych (w metajęzyku). Dla przykładu, (dwuargumentowe) funkcje prawdziwościowe Kn, Al, Im, Rw, Ar:

$Kn(x,y)=1$	wtedy i tylko wtedy, gdy	$x=1$ oraz $y=1$
$Al(x,y)=0$	wtedy i tylko wtedy, gdy	$x=0$ oraz $y=0$
$Im(x,y)=0$	wtedy i tylko wtedy, gdy	$x=1$ oraz $y=0$
$Rw(x,y)=1$	wtedy i tylko wtedy, gdy	$x=y$
$Ar(x,y)=1$	wtedy i tylko wtedy, gdy	$x\neq y$

A także funkcja $Ng : \{0,1\} \to \{0,1\}$, gdzie $Ng(0)=1$, $Ng(1)=0$.

1 Semantyczna równoważność formuł

1. Formuły φ i ψ języka KRZ są *semantycznie równoważne*, gdy dla każdego wartościowania v: $v(\varphi) = v(\psi)$. Jeśli φ i ψ są semantycznie równoważne, to piszemy $\varphi \sim \psi$.

2. $\sim$ jest relacją równoważności.

3. Fakt: $\varphi \sim \psi$ wtedy i tylko wtedy, gdy $\varphi \equiv \psi$ jest tautologią KRZ.

Formuły φ i ψ języka KRZ są *inferencyjnie równoważne*, gdy tezą KRZ jest formuła $\varphi \equiv \psi$. [W tej definicji zakładamy, że odnosimy się do ustalonej relacji konsekwencji. Podobnie dla inferencyjnej równoważności w KRP.]

Formuły φ i ψ języka KRP są *równospełnialne*, gdy: φ jest spełnialna wtedy i tylko wtedy, gdy ψ jest spełnialna.

2 Postacie normalne

2.1 Notacja

1. *Literałami* nazywamy zmienne zdaniowe, negacje zmiennych zdaniowych oraz stałe $\top$ i $\bot$. Jeśli literał L ma postać p_n, to literałem *sprzężonym* z L jest $\neg p_n$. Jeśli literał L ma postać $\neg p_n$, to literałem *sprzężonym* z L jest p_n.

2. Wieloczłonową koniunkcję formuł φ_1, φ_2, ..., φ_n zapisywać możemy bez użycia nawiasów: $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$.

3. Podobnie, wieloczłonową alternatywę formuł φ_1, φ_2, ..., φ_n zapisywać możemy bez użycia nawiasów: $\varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$.

Notacja Fittinga:

1. $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$ zapisujemy jako $\langle \varphi_1, \varphi_2, \ldots, \varphi_n \rangle$.

2. $\varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$ zapisujemy jako $[\varphi_1, \varphi_2, \ldots, \varphi_n]$.

2.2 Terminologia

1. *Koniunkcją elementarną* nazwiemy dowolną koniunkcję literałów.

2. *Alternatywą elementarną* nazwiemy dowolną alternatywę literałów.

3. *Alternatywną postacią normalną* (*apn*) nazwiemy dowolną alternatywę koniunkcji elementarnych.

4. *Koniunkcyjną postacią normalną* (*kpn*) nazwiemy dowolną koniunkcję alternatyw elementarnych.

5. Apn (odpowiednio: kpn) φ nazywamy *istotną* i oznaczamy *iafn* (odpowiednio: *ikpn*), jeśli każda zmienna zdaniowa formuły φ występuje w każdej elementarnej koniunkcji (odpowiednio: alternatywie) dokładnie raz, zaprzeczona bądź niezaprzeczona.

6. Każdą apn (odpowiednio: kpn, iapn, ikpn) semantycznie równoważną danej formule φ nazywamy *apn* (odpowiednio: *kpn, iapn, ikpn*) *formuły* φ.

2.3 Ideologia

Dla każdej formuły φ języka KRZ istnieje formuła ψ taka, że $\varphi \sim \psi$ i ψ jest kpn. Dla dowodu wystarczy zauważyć, że tautologiami KRZ są:

1. $(\varphi \equiv \psi) \equiv ((\varphi \to \psi) \wedge (\psi \to \varphi))$
2. $(\varphi \to \psi) \equiv ((\neg\varphi) \vee \psi)$
3. $\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi)$
4. $\neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$
5. $\neg\neg\varphi \equiv \varphi$
6. $(\varphi \vee (\psi \wedge \chi)) \equiv ((\varphi \vee \psi) \wedge (\varphi \vee \chi))$
7. $(\varphi \wedge (\psi \vee \chi)) \equiv ((\varphi \wedge \psi) \vee (\varphi \wedge \chi))$

Podobnie, dla każdej formuły φ języka KRZ istnieje formuła ψ taka, że $\varphi \sim \psi$ i ψ jest apn.

2.4 Przykład

Wykorzystamy powyższe prawa do znalezienia koniunkcyjnej postaci normalnej formuły $((p \equiv q) \to (p \to r)) \to (q \to p)$

1. $((p \equiv q) \to (p \to r)) \to (q \to p)$
2. $(((p \to q) \wedge (q \to p)) \to (p \to r)) \to (q \to p)$
3. $\neg(((p \to q) \wedge (q \to p)) \to (p \to r)) \vee (q \to p)$
4. $\neg(\neg((p \to q) \wedge (q \to p)) \vee (p \to r)) \vee (q \to p)$

5. $\neg(\neg((\neg p \vee q) \wedge (\neg q \vee p)) \vee (\neg p \vee r)) \vee (\neg q \vee p)$
6. $(\neg\neg((\neg p \vee q) \wedge (\neg q \vee p)) \wedge \neg(\neg p \vee r))) \vee (\neg q \vee p)$
7. $((\neg p \vee q) \wedge (\neg q \vee p) \wedge (\neg\neg p \wedge \neg r)) \vee (\neg q \vee p)$
8. $((\neg p \vee q) \wedge (\neg q \vee p) \wedge p \wedge \neg r) \vee (\neg q \vee p)$
9. $(\neg p \vee q \vee \neg q \vee p) \wedge (\neg q \vee p \vee \neg q \vee p) \wedge (p \vee \neg q \vee p) \wedge (\neg r \vee \neg q \vee p)$

2.5 Koniunkcyjne postacie normalne

Po pierwsze: jeśli φ jest tautologią KRZ oraz $\varphi \sim \psi$, to także ψ jest tautologią KRZ.

Po drugie: jeśli φ jest kpn, to jest postaci: $A_1 \wedge A_2 \wedge \ldots \wedge A_n$, gdzie każda formuła A_i jest alternatywą elementarną postaci: $L_i^1 \vee L_i^2 \vee \ldots \vee L_i^m$, gdzie z kolei każda formuła L_i^j jest literałem.

Koniunkcja $A_1 \wedge A_2 \wedge \ldots \wedge A_n$ jest tautologią KRZ wtedy i tylko wtedy, gdy wszystkie formuły A_i są tautologiami.

Formuła A_i (czyli formuła $L_i^1 \vee L_i^2 \vee \ldots \vee L_i^m$) jest tautologią KRZ wtedy i tylko wtedy, gdy wśród $L_i^1, L_i^2, \ldots, L_i^m$ występuje co najmniej jedna para literałów sprzężonych.

Tak więc: sprowadzanie formuł do kpn dostarcza algorytmu sprawdzającego tautologiczność.

2.6 Alternatywne postacie normalne

Po pierwsze: jeśli φ jest kontrtautologią KRZ oraz $\varphi \sim \psi$, to także ψ jest kontrtautologią KRZ.

Po drugie: jeśli φ jest apn, to jest postaci: $A_1 \vee A_2 \vee \ldots \vee A_n$, gdzie każda formuła A_i jest koniunkcją elementarną postaci: $L_i^1 \wedge L_i^2 \wedge \ldots \wedge L_i^m$, gdzie z kolei każda formuła L_i^j jest literałem.

Alternatywa $A_1 \vee A_2 \vee \ldots \vee A_n$ jest kontrtautologią KRZ wtedy i tylko wtedy, gdy wszystkie formuły A_i są kontrtautologiami.

Formuła A_i (czyli formuła $L_i^1 \wedge L_i^2 \wedge \ldots \wedge L_i^m$) jest kontrtautologią KRZ wtedy i tylko wtedy, gdy wśród $L_i^1, L_i^2, \ldots, L_i^m$ występuje co najmniej jedna para literałów sprzężonych.

Tak więc: sprowadzanie formuł do apn dostarcza algorytmu sprawdzającego kontrtautologiczność.

2.7 Algorytmy

2.8 Algorytm dla kpn

Korzystamy z notacji Smullyana. Reguły redukcji (Fitting 1990, 26), wykorzystywane w tworzeniu kpn:

$$\frac{\neg\neg\psi}{\psi} \qquad \frac{\neg\top}{\bot} \qquad \frac{\neg\,\bot}{\top}$$

$$\begin{array}{c}\beta\\ |\\ \beta_1\\ |\\ \beta_2\end{array} \qquad\qquad \begin{array}{ccc}&\alpha&\\ \swarrow&&\searrow\\ \alpha_1&&\alpha_2\end{array}$$

Reguła dla β-formuł działa wewnątrz klauzuli, reguła dla α-formuł tworzy dwie klauzule.

Aby sprowadzić φ do kpn (Fitting 1990, 27) korzystamy z algorytmu:

1. **begin**
   (a) Niech S będzie $\langle[\varphi]\rangle$
   (b) **while** *jakiś element S zawiera nie-literał* **do**
      i. wybierz z S element D zawierający nie-literał
      ii. wybierz z D nie-literał N
      iii. zastosuj odpowiednią regułę redukcyjną do N
      iv. niech S oznacza nowoutworzoną formułę
   (c) **end**
2. **end**

Wykonanie algorytmu podaje kpn dla φ. Podobny algorytm działa w przypadku apn.

2.9 Kpn: przykład

Przykład: kpn dla $(p \to (q \to r)) \to ((p \to q) \to (p \to r))$

1. $\langle[\underline{(p \to (q \to r)) \to ((p \to q) \to (p \to r))}]\rangle$

2. $\langle[\neg(p \to (q \to r)), \underline{((p \to q) \to (p \to r))}]\rangle$
3. $\langle[\neg(p \to (q \to r)), \neg(p \to q), \underline{(p \to r)}]\rangle$
4. $\langle[\underline{\neg(p \to (q \to r))}, \neg(p \to q), \neg p, r]\rangle$
5. $\langle[p, \underline{\neg(p \to q)}, \neg p, r], [\neg(q \to r), \neg(p \to q), \neg p, r]\rangle$
6. $\langle[p, p, \neg p, r], [p, \neg q, \neg p, r], [\underline{\neg(q \to r)}, \neg(p \to q), \neg p, r]\rangle$
7. $\langle[p, p, \neg p, r], [p, \neg q, \neg p, r], [q, \underline{\neg(p \to q)}, \neg p, r], [\neg r, \neg(p \to q), \neg p, r]\rangle$
8. $\langle[p, p, \neg p, r], [p, \neg q, \neg p, r], [q, p, \neg p, r], [q, \neg q, \neg p, r],$
   $[\neg r, \underline{\neg(p \to q)}, \neg p, r]\rangle$
9. $\langle[p, p, \neg p, r], [p, \neg q, \neg p, r], [q, p, \neg p, r], [q, \neg q, \neg p, r],$
   $[\neg r, p, \neg p, r], [\neg r, \neg q, \neg p, r]\rangle$

Podkreślono formułę, do której stosowano regułę redukcji.

2.10 Algorytm dla apn

Korzystamy z notacji Smullyana. Reguły redukcji (Fitting 1990, 29), wykorzystywane w tworzeniu apn:

$$\frac{\neg\neg\psi}{\psi} \qquad \frac{\neg\top}{\bot} \qquad \frac{\neg\,\bot}{\top}$$

$$\begin{array}{c}\alpha\\ |\\ \alpha_1\\ |\\ \alpha_2\end{array} \qquad\qquad \begin{array}{ccc}&\beta&\\ \swarrow&&\searrow\\ \beta_1&&\beta_2\end{array}$$

Reguła dla α-formuł działa wewnątrz klauzuli, reguła dla β-formuł tworzy dwie klauzule.

Aby sprowadzić φ do apn (Fitting 1990, 30) korzystamy z algorytmu:

1. **begin**
   (a) Niech S będzie $[\langle\varphi\rangle]$
   (b) **while** *jakiś element S zawiera nie-literał* **do**

i. wybierz z S element D zawierający nie-literał
ii. wybierz z D nie-literał N
iii. zastosuj odpowiednią regułę redukcyjną do N
iv. niech S oznacza nowoutworzoną formułę

(c) **end**

2. **end**

Wykonanie algorytmu podaje apn dla φ. Zauważ, że struktura algorytmu jest taka sama, jak poprzednio (inne są tylko reguły redukcji).

2.11 Apn: przykład

Przykład: apn dla $(p \to q) \wedge (p \wedge \neg q)$

1. $[\langle \underline{(p \to q) \wedge (p \wedge \neg q)} \rangle]$
2. $[\langle p \to q, \underline{p \wedge \neg q} \rangle]$
3. $[\langle \underline{p \to q}, p, \neg q \rangle]$
4. $[\langle \neg p, p, \neg q \rangle, \langle q, p, \neg q \rangle]$

Ten przykład był prosty – od razu widać, że $(p \to q) \wedge (p \wedge \neg q)$ jest semantycznie równoważna formule: $(\neg p \wedge p \wedge \neg q) \vee (q \wedge p \wedge \neg q)$. Widać też, że badana formuła jest kontrtautologią.

3 Postacie prefiksowe

Sprowadzenie formuł języka KRP do pewnych standardowych postaci (np. ze wszystkimi kwantyfikatorami na początku formuły) ułatwia tworzenie dowodów (w wielu metodach dowodowych).

Formuła φ języka KRP jest w *prefiksowej postaci normalnej*, gdy jest ona postaci $Q_1 x_1 \ldots Q_n x_n \ \psi$, gdzie ψ jest formułą bez kwantyfikatorów, a każdy symbol Q_i jest jednym z kwantyfikatorów: $\forall$ lub $\exists$. Jeśli w dodatku ψ jest w kpn, to φ jest w *koniunkcyjnej prefiksowej postaci normalnej*. Ciąg $Q_1 x_1 \ldots Q_n x_n$ nazywamy *prefiksem* formuły φ, a formułę ψ jej *matrycą*.

Przez *formułę uniwersalną* rozumiemy każdą formułę w prefiksowej postaci normalnej, w której prefiksie występują jedynie kwantyfikatory generalne. Formuła jest *otwarta*, jeśli zawiera zmienne wolne. W przeciwnym przypadku jest *zamknięta*.

Prawami KRP są (zmienna x nie jest wolna w φ):

1. $\exists x\psi \to \varphi \ \equiv \ \forall x(\psi \to \varphi)$
2. $\forall x\psi \to \varphi \ \equiv \ \exists x(\psi \to \varphi)$
3. $\varphi \to \exists x\psi \ \equiv \ \exists x(\varphi \to \psi)$
4. $\varphi \to \forall x\psi \ \equiv \ \forall x(\varphi \to \psi)$
5. $\exists x\psi \wedge \varphi \ \equiv \ \exists x(\psi \wedge \varphi)$
6. $\forall x\psi \wedge \varphi \ \equiv \ \forall x(\psi \wedge \varphi)$
7. $\varphi \wedge \exists x\psi \ \equiv \ \exists x(\varphi \wedge \psi)$
8. $\varphi \wedge \forall x\psi \ \equiv \ \forall x(\varphi \wedge \psi)$
9. $\exists x\psi \vee \varphi \ \equiv \ \exists x(\psi \vee \varphi)$
10. $\forall x\psi \vee \varphi \ \equiv \ \forall x(\psi \vee \varphi)$
11. $\varphi \vee \exists x\psi \ \equiv \ \exists x(\varphi \vee \psi)$
12. $\varphi \vee \forall x\psi \ \equiv \ \forall x(\varphi \vee \psi)$

Dalsze potrzebne prawa KRP:

1. $\neg\forall x\varphi \equiv \exists x\neg\varphi$
2. $\neg\exists x\varphi \equiv \forall x\neg\varphi$

Przy założeniu, że zmienna y nie jest wolna w φ oraz że y jest podstawialna za x w φ:

1. $\forall x\varphi \equiv \forall y\varphi[x/y]$
2. $\exists x\varphi \equiv \exists y\varphi[x/y]$

Ponadto: $\varphi \equiv \psi$ zastępujemy przez $(\varphi \to \psi) \wedge (\psi \to \varphi)$.

Dla dowolnej formuły φ języka KRP istnieje równoważna jej formuła φ' w prefiksowej postaci normalnej, o tych samych zmiennych wolnych co φ. Każdą taką formułę φ' nazywamy *prefiksową postacią normalną* formuły φ.

Formułę w prefiksowej postaci normalnej równoważną inferencyjnie z:

$$(1) \quad \forall x\exists yP(x,y) \vee \neg\exists x\forall yQ(x,y)$$

możemy znaleźć np. w następujący sposób:

1. $\forall x \exists y P(x,y) \vee \neg \exists x \forall y Q(x,y)$
2. $\forall u (\exists y P(u,y) \vee \neg \exists x \forall y Q(x,y))$
3. $\forall u \exists v (P(u,v) \vee \neg \exists x \forall y Q(x,y))$
4. $\forall u \exists v (P(u,v) \vee \forall x \neg \forall y Q(x,y))$
5. $\forall u \exists v (P(u,v) \vee \forall x \exists y \neg Q(x,y))$
6. $\forall u \exists v \forall w (P(u,v) \vee \exists y \neg Q(w,y))$
7. $\forall u \exists v \forall w \exists z (P(u,v) \vee \neg Q(w,z))$.

Formułę w prefiksowej postaci normalnej równoważną inferencyjnie z:

$$(2) \quad \forall x \forall y (\exists z (P(x,z) \wedge P(y,z)) \rightarrow \exists u Q(x,y,u))$$

możemy znaleźć np. w następujący sposób:

1. $\forall x \forall y (\exists z (P(x,z) \wedge P(y,z)) \rightarrow \exists u Q(x,y,u))$
2. $\forall x \forall y \forall w ((P(x,w) \wedge P(y,w)) \rightarrow \exists u Q(x,y,u))$
3. $\forall x \forall y \forall w \exists z ((P(x,w) \wedge P(y,w)) \rightarrow Q(x,y,z))$.

4 Skolemizacja

Można wyeliminować kwantyfikatory egzystencjalne kosztem wprowadzenia nowych symboli funkcyjnych.

Niech $\mathcal{L}$ będzie językiem KRP ustalonej sygnatury Σ.

Dla dowolnego zdania φ o postaci $\forall x_1 \ldots \forall x_n \exists y \psi$ języka $\mathcal{L}$ zdanie φ' o postaci $\forall x_1 \ldots \forall x_n \psi(f(x_1, \ldots, x_n))$, gdzie f jest nowym symbolem funkcyjnym spoza Σ, jest równospełnialne z φ.

Dla dowolnego zdania φ języka $\mathcal{L}$ istnieje formuła uniwersalna φ' w języku $\mathcal{L}'$ o sygnaturze rozszerzonej o nowe symbole funkcyjne taka, że φ oraz φ' są równospełnialne.

Każdą formułę φ' spełniającą tezę powyższego twierdzenia nazywamy *skolemową postacią normalną* formuły φ.

Przypomnijmy, jak wyglądały formuły (1) oraz (2):

1. $\forall x \exists y P(x,y) \vee \neg \exists x \forall y Q(x,y)$
2. $\forall x \forall y (\exists z (P(x,z) \wedge P(y,z)) \rightarrow \exists u Q(x,y,u))$

Przypomnijmy też postacie prefiksowe formuł (1) oraz (2):

1. $\forall u \exists v \forall w \exists z (P(u, v) \vee \neg Q(w, z))$
2. $\forall x \forall y \forall w \exists z ((P(x, w) \wedge P(y, w)) \rightarrow Q(x, y, z))$.

Możliwymi postaciami skolemowymi wspomnianych wyżej formuł (1) oraz (2) są np.:

1. $(1)'$ $\quad \forall u \forall w (P(u, f(u)) \vee \neg Q(w, g(u, w)))$
2. $(2)'$ $\quad \forall x \forall y \forall w ((P(x, w) \wedge P(y, w)) \rightarrow Q(x, y, f(x, y, w)))$.

JERZY POGONOWSKI
Zakład Logiki i Kognitywistyki UAM
www.kognitywistyka.amu.edu.pl
http://logic.amu.edu.pl/index.php/Dydaktyka
pogon@amu.edu.pl