Postacie normalne i prefiksowe

Jerzy Pogonowski

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

MDTiAR 3xi2015

Plan na dziś:

- Semantyczna równoważność formuł
- Postacie normalne: koniunkcyjna oraz alternatywna
- Postacie prefiksowe
- Skolemizacja

- Dodatek: funkcje prawdziwościowe, Twierdzenie Posta

Przypomnienie: spójnik, funktor, funkcja

- Podejście leniwe (np. Fitting): używanie tych samych symboli na oznaczenie funktorów (spójników) prawdziwościowych oraz odpowiadających im funkcji prawdziwościowych.
- Podejście skrupulatne (np. Batóg): 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 $Ng(0) = 1$, $Ng(1) = 0$.

Nieodróżnialni bliźniacy

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

Notacja

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

- $\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n$ zapisujemy jako $\langle \varphi_1, \varphi_2, \ldots, \varphi_n \rangle$.
- $\varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$ zapisujemy jako $[\varphi_1, \varphi_2, \ldots, \varphi_n]$.

Postacie normalne formuł

- *Koniunkcją elementarną* nazwiemy dowolną koniunkcję literałów.
- *Alternatywą elementarną* nazwiemy dowolną alternatywę literałów.
- *Alternatywną postacią normalną* (*apn*) nazwiemy dowolną alternatywę koniunkcji elementarnych.
- *Koniunkcyjną postacią normalną* (*kpn*) nazwiemy dowolną koniunkcję alternatyw elementarnych.
- 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.
- Każdą apn (odpowiednio: kpn, iapn, ikpn) semantycznie równoważną danej formule φ nazywamy *apn* (odpowiednio: *kpn, iapn, ikpn*) *formuły* φ.

Dlaczego postacie normalne są ważne

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:

- $(\varphi \equiv \psi) \equiv ((\varphi \to \psi) \wedge (\psi \to \varphi))$
- $(\varphi \to \psi) \equiv ((\neg\varphi) \vee \psi)$
- $\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi)$
- $\neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$
- $\neg\neg\varphi \equiv \varphi$
- $(\varphi \vee (\psi \wedge \chi)) \equiv ((\varphi \vee \psi) \wedge (\varphi \vee \chi))$
- $(\varphi \wedge (\psi \vee \chi)) \equiv ((\varphi \wedge \psi) \vee (\varphi \wedge \chi))$

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

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. $(\neg p \vee q \vee \neg q \vee p) \wedge (\neg q \vee p \vee \neg q \vee p) \wedge (p \vee \neg q \vee p) \wedge (\neg r \vee \neg q \vee p)$

Dlaczego koniunkcyjne postacie normalne są ważne

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, …, L_i^m występuje co najmniej jedna para literałów sprzężonych.

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

Dlaczego alternatywne postacie normalne są ważne

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, ..., L_i^m występuje co najmniej jedna para literałów sprzężonych.

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

Janie, proszę poddać redukcji panią hrabinę

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

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

$$\begin{array}{c} \beta \\ | \\ \beta_1 \\ | \\ \beta_2 \end{array} \qquad\qquad \begin{array}{c} \alpha \\ \diagup \quad \diagdown \\ \alpha_1 \qquad \alpha_2 \end{array}$$

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

Janie, proszę sprowadzić pana hrabiego do postaci normalnej

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

1. **begin**
   1. Niech S będzie $\langle [\varphi] \rangle$
   2. **while** *jakiś element S zawiera nie-literał* **do**
      1. wybierz z S element D zawierający nie-literał
      2. wybierz z D nie-literał N
      3. zastosuj odpowiednią regułę redukcyjną do N
      4. niech S oznacza nowoutworzoną formułę
   3. **end**
2. **end**

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

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.

Na początku było tak: bociana dziobał szpak

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

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

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

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

A potem była zmiana: szpak dziobał bociana

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

1. **begin**
    1. Niech S będzie $[\langle\varphi\rangle]$
    2. **while** *jakiś element S zawiera nie-literał* **do**
        1. wybierz z S element D zawierający nie-literał
        2. wybierz z D nie-literał N
        3. zastosuj odpowiednią regułę redukcyjną do N
        4. niech S oznacza nowoutworzoną formułę
    3. **end**
2. **end**

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

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

- $[\langle \underline{(p \to q) \wedge (p \wedge \neg q)} \rangle]$
- $[\langle p \to q, \underline{p \wedge \neg q} \rangle]$
- $[\langle \underline{p \to q}, p, \neg q \rangle]$
- $[\langle \neg p, p, \neg q \rangle, \langle q, p, \neg q \rangle]$

- 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.

Definicja i twierdzenia

- 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:

- $\neg\forall x\varphi \equiv \exists x\neg\varphi$
- $\neg\exists x\varphi \equiv \forall x\neg\varphi$

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

- $\forall x\varphi \equiv \forall y\varphi[x/y]$
- $\exists x\varphi \equiv \exists y\varphi[x/y]$

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

- 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.

Przykłady

Formułę w prefiksowej postaci normalnej równoważną inferencyjnie z:

$$(1) \quad \forall x \exists y P(x,y) \vee \neg \exists x \forall y Q(x,y)$$

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. $\forall u \exists v \forall w \exists z(P(u,v) \vee \neg Q(w,z))$.

Przykłady

Formułę w prefiksowej postaci normalnej równoważną inferencyjnie z:

$$(1) \quad \forall x \exists y P(x,y) \vee \neg \exists x \forall y Q(x,y)$$

możemy znaleźć również 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. $\forall u (\exists y P(u,y) \vee \forall x \neg \forall y Q(x,y))$
4. $\forall u \forall w (\exists y P(u,y) \vee \forall x \neg \forall y Q(w,y))$
5. $\forall u \forall w \exists v (P(u,y) \vee \forall x \neg \forall y Q(w,y))$
6. $\forall u \forall w \exists v (P(u,y) \vee \exists y \neg Q(w,y))$
7. $\forall u \forall w \exists v \exists z (P(u,y) \vee \neg Q(w,z))$

Przykłady

Formułę w prefiksowej postaci normalnej równoważną inferencyjnie z:

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

możemy znaleźć np. w następujący sposób:

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

Skolemowa postać normalna

- 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.

Przykład

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.:

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

Czy to komukolwiek potrzebne?

- W rachunkach logicznych wykorzystujemy funktory jedno- oraz dwuargumentowe – funktory boolowskie o większej liczbie argumentów można przez nie zdefiniować.
- Dla przykładu: trójargumentowy funktor *większość* ma definicję: $m(p,q,r) =_{df} (p \wedge q) \vee (q \wedge r) \vee (p \wedge r)$.

- Pewne problemy matematyczne wymagają jednak rozważenia funkcji boolowskich o większej (od 2) liczbie argumentów.
- Problem: minimalizacja funkcji boolowskich.
- Pozostała część tej prezentacji przeznaczona jest dla ciekawskich. Omawiany materiał znaleźć można w podręcznikach matematyki dyskretnej, np.: Jabłoński, S.W. 1991. *Wstęp do matematyki dyskretnej*, Wydawnictwo Naukowe PWN, Warszawa.

Jeszcze o funkcjach prawdziwościowych

Pamiętamy, że (n-argumentową) funkcją prawdziwościową nazywamy dowolną funkcję $f : \{0,1\}^n \to \{0,1\}$, dla $n \geqslant 1$.

Wszystkich n-argumentowych funkcji prawdziwościowych jest 2^{2^n}. W szczególności, jest 16 dwuargumentowych funkcji prawdziwościowych oraz są 4 jednoargumentowe funkcje prawdziwościowe.

Do ważnych problemów (także praktycznych) należą:

- definiowanie jednych funkcji prawdziwościowych przez inne
- reprezentacje dowolnych funkcji prawdziwościowych przez stosowne postacie normalne
- znajdowanie zupełnych układów funkcji prawdziwościowych.

Kodowanie funkcji prawdziwościowych

Każda z 16 dwuargumentowych funkcji prawdziwościowych może zostać zakodowana czteroelementowym ciągiem 0 i 1 (zob. tabelę z pierwszego wykładu), a więc dwójkowym przedstawieniem jednej z liczb od 0 do 15.

Ogólnie, każda n-argumentowa funkcja prawdziwościowa może zostać zakodowana 2^n-elementowym ciągiem 0 i 1, a więc dwójkowym przedstawieniem jednej z liczb od 0 do $2^n - 1$.

Wszystkie n-argumentowe funkcje prawdziwościowe można zatem łatwo wypisać w formie tabeli o 2^n wierszach oraz $n + 2^{2^n}$ kolumnach. Na pierwszych n miejscach w i-tym wierszu należy umieścić dwójkową reprezentację liczby $i - 1$. W kolumnach od $n + 1$ do 2^{2^n} umieszczamy kolejno (pionowo) reprezentacje dwójkowe liczb od 0 do $2^{2^n} - 1$.

Termy opisujące funkcje prawdziwościowe

Klasę wszystkich funkcji prawdziwościowych oznaczymy przez $\boldsymbol{C}$. Niech $G \subset \boldsymbol{C}$. Z każdą n-argumentową funkcją f z G stowarzyszymy symbol funkcyjny $\overline{f}$. Niech $Vbl = \{v_0, v_1, v_2, \ldots\}$ będzie przeliczalnym zbiorem symboli, zwanych zmiennymi (nazwowymi). Zdefiniujemy pojęcie *termu*:

- (a) każda zmienna v_i jest termem;
- (b) jeśli f jest n-argumentową funkcją z G, a $T_1, \ldots, T_n$ są termami, to $\overline{f}(T_1, \ldots, T_n)$ jest termem;
- (c) nie ma innych termów oprócz utworzonych na mocy warunków (a) i (b).

Uwaga. Termy to wyrażenia opisujące funkcje prawdziwościowe w pewnym (nowym!) języku formalnym.

Wartości termów

Wartościowaniem zmiennych nazwowych (wzn) nazwiemy każdą funkcję $w : Vbl \to \{0, 1\}$. Przyjmujemy oznaczenie: $w(v_i) = w_i$.
Oczywiście w każdym termie występuje jedynie skończona liczba zmiennych (nazwowych).

Określamy *wartość termu T* dla danych wartości zmiennych określonych przez wzn w:

- (a) jeśli T jest zmienną v_i, to wartością T dla wzn w jest w_i;
- (b) jeśli $T = \overline{f}(T_1, \ldots, T_n)$ i wartościami $T_1, \ldots, T_n$ są odpowiednio $\varepsilon_1, \ldots, \varepsilon_n$, to wartością T jest $f(\varepsilon_1, \ldots, \varepsilon_n)$.

Reprezentowalność

Mówimy, że n-argumentowa funkcja prawdziwościowa g jest *reprezentowana* przez term T, jeśli wszystkie zmienne T są wśród $v_1, \ldots, v_n$ i dla dowolnych wartości (przy każdym wzn) zmiennych $v_1, \ldots, v_n$ wartość termu T jest identyczna z wartością termu $\overline{g}(v_1, \ldots, v_n)$.

Mówimy, że funkcja g jest *złożeniem* funkcji $f_1, \ldots, f_n$, jeśli g jest reprezentowana przez term, którego wszystkie symbole funkcyjne znajdują się pośród $\overline{f}_1, \ldots, \overline{f}_n$.

Uwaga. W praktyce, fakt że jakaś funkcja jest złożeniem innych wyrażamy bezpośrednio w metajęzyku. Piszemy np.: $Im(x, y) = Al(Ng(x), y)$. Zwróć uwagę, że zachodzenie tej równości związane jest z faktem, że $(\varphi \rightarrow \psi) \equiv (\neg\varphi \vee \psi)$ jest tautologią KRZ.

Zbiory funkcji: zupełne, zamknięte, niezależne

Zbiór funkcji G nazywamy *zupełnym*, jeśli dowolna funkcja prawdziwościowa jest złożeniem pewnych funkcji z G. Zbiór funkcji G nazywamy *niezależnym*, jeśli żadna funkcja f z G nie daje się przedstawić jako złożenie funkcji z $G - \{f\}$.

Klasę funkcji G nazywamy *zamkniętą*, jeśli zawiera ona wszystkie złożenia funkcji, które są jej elementami. Zamkniętą klasę G nazywamy *prapełną*, jeśli $G \neq \boldsymbol{C}$ i G nie jest zawarta w żadnej klasie zamkniętej, różnej od $\boldsymbol{C}$. Niezależny zbiór funkcji G nazywamy *bazą klasy zamkniętej* K, jeśli każda funkcja należąca do K jest złożeniem funkcji należących do G.

Do ważnych funkcji prawdziwościowych należą omówione poprzednio: Ng, Kn, Al, Im, Rw. Będziemy posługiwać się także funkcją n-argumentowej koniunkcji: $Kn(x_1, \ldots, x_n) = 1$ wtedy i tylko wtedy, gdy $x_1 = \ldots = x_n = 1$. Podobnie dla n-argumentowej alternatywy.

Klasy funkcji prawdziwościowych

- Przez $\boldsymbol{C}_1$ oznaczamy klasę funkcji spełniających warunek $f(1, 1, \ldots, 1) = 1$.
- Przez $\boldsymbol{C}_0$ oznaczamy klasę funkcji spełniających warunek $f(0, 0, \ldots, 0) = 0$.
- $\boldsymbol{L}$ jest klasą wszystkich funkcji *liniowych*, tj. funkcji postaci $x_1 + \ldots + x_n + \varepsilon$, gdzie $\varepsilon \in \{0, 1\}$.
- $\boldsymbol{D}$ jest klasą funkcji *samodualnych*, tj. funkcji spełniających warunek $f(x_1, \ldots, x_n) = Ng(f(Ng(x_1), \ldots, Ng(x_n)))$.
- przez $\boldsymbol{M}$ oznaczymy klasę wszystkich funkcji *monotonicznych*, tj. funkcji spełniających warunek: jeśli $x_1 \leqslant y_1, \ldots, x_n \leqslant y_n$, to $f(x_1, \ldots, x_n) \leqslant f(y_1, \ldots, y_n)$.

Uwaga. Symbolu $\leqslant$ używamy dla relacji niewiększości na zbiorze $\{0, 1\}$ traktowanym jako zbiór liczb. Symbol $+$ oznacza dodawanie modulo 2 w tym zbiorze.

Przedstawialność: zapis formalny

Mówimy, że funkcja f jest przedstawialna za pomocą funkcji f_1, ..., f_k, jeśli równość $\overline{f}(v_1, \ldots, v_n) = T$ zachodzi dla wszystkich wartości przyporządkowanych (przez każde wzn) zmiennym $v_1, \ldots, v_n$, gdzie T jest pewnym termem zbudowanym z symboli funkcyjnych $\overline{f}_1$, ..., $\overline{f}_k$ (niekoniecznie wszystkich) oraz zmiennych $v_1, \ldots, v_n$ (również niekoniecznie wszystkich).

Przykłady:

- Kn jest przedstawialna przez Ng oraz Al:

$$\overline{Kn}(v_1, v_2) = \overline{Ng}(\overline{Al}(\overline{Ng}(v_1), \overline{Ng}(v_2)))$$

- Al jest przedstawialna przez Ng oraz Kn:

$$\overline{Al}(v_1, v_2) = \overline{Ng}(\overline{Kn}(\overline{Ng}(v_1), \overline{Ng}(v_2)))$$

Przedstawialność: zapis uproszczony

- Im jest przedstawialna przez Ng oraz Al:
  $Im(x, y) = Al(Ng(x), y)$
- Im jest przedstawialna przez Ng oraz Kn:
  $Im(x, y) = Ng(Kn(x, Ng(y)))$
- Al jest przedstawialna przez Ng oraz Im:
  $Al(x, y) = Im(Ng(x), y)$
- Kn jest przedstawialna przez Ng oraz Im:
  $Kn(x, y) = Ng(Im(x, Ng(y)))$

Uwaga. Powyższe równości (z tego slajdu), zapisane w metajęzyku, dotyczą bezpośrednio funkcji prawdziwościowych. Milcząco wykorzystujemy tu pewne własności termów opisujących funkcje prawdziwościowe. Równości z poprzedniego slajdu zapisane były w języku termów.

Nie pogub się!

Być może, jesteś wstrząśnięta (choć nie zmieszana) używanymi w tym wykładzie subtelnościami notacyjnymi. Tak trzeba, *trust me*. Zauważ, że:

- gdy piszemy np. równość $Im(x, y) = Al(Ng(x), y)$, to jest to zapis w metajęzyku, mówiący coś o funkcjach prawdziwościowych;
- gdy piszemy równość termów $\overline{Im}(v_1, v_2) = \overline{Al}(\overline{Ng}(v_1), v_2)$, to jest to zapis w języku formalnym opisującym funkcje prawdziwościowe;
- gdy piszemy równoważność $(\varphi \to \psi) \equiv (\neg\varphi \vee \psi)$, to jest to formuła języka KRZ.

Można ustanowić precyzyjną odpowiedniość między: spójnikami prawdziwościowymi $\neg$, $\wedge$, $\vee$, $\to$ oraz $\equiv$, a symbolami funkcyjnymi, odpowiednio: $\overline{Ng}$, $\overline{Kn}$, $\overline{Al}$, $\overline{Im}$ oraz $\overline{Rw}$.

Postacie normalne dla funkcji prawdziwościowych

W języku termów opisujących funkcje prawdziwościowe można zdefiniować postacie normalne termów:

- Każde wyrażenie postaci x lub $\overline{Ng}(x)$, gdzie x jest zmienną (nazwową), nazywamy literałem.
- Wyrażenia postaci $L_1 \wedge L_2 \wedge \ldots \wedge L_n$, gdzie każde L_i jest literałem, nazywamy koniunkcjami elementarnymi.
- Wyrażenia postaci $L_1 \vee L_2 \vee \ldots \vee L_n$, gdzie każde L_i jest literałem, nazywamy alternatywami elementarnymi.
- Wyrażenie w koniunkcyjnej postaci normalnej (kpn) jest to wyrażenie kształtu $A_1 \wedge A_2 \wedge \ldots \wedge A_n$, gdzie każde A_i jest alternatywą elementarną.
- Wyrażenie w alternatywnej postaci normalnej (apn) jest to wyrażenie kształtu $A_1 \vee A_2 \vee \ldots \vee A_n$, gdzie każde A_i jest koniunkcją elementarną.

Postacie normalne dla funkcji prawdziwościowych

Zachodzi następujące ważne twierdzenie o postaciach normalnych:

Twierdzenie. Każda funkcja prawdziwościowa jest przedstawialna zarówno w koniunkcyjnej, jak i w alternatywnej postaci normalnej.

Przykład:

- apn dla Rw: $\overline{Rw}(x, y) = \overline{Al}(\overline{Kn}(x, y), \overline{Kn}(\overline{Ng}(x), \overline{Ng}(y)))$
- kpn dla Rw: $\overline{Rw}(x, y) = \overline{Kn}(\overline{Al}(\overline{Ng}(x), y), \overline{Al}(x, \overline{Ng}(y)))$.

Zupełne układy funkcji prawdziwościowych

Z twierdzenia o postaciach normalnych wynika, że następujące układy funkcji są zupełne:

$\{Ng, Kn\} \quad \{Ng, Al\} \quad \{Ng, Im\}.$

Zupełny jest także układ funkcji $\{Ar, Kn, \mathbf{1}\}$, gdzie $\mathbf{1}$ jest funkcją stałą równą 1, a funkcja Ar (alternatywa rozłączna) odpowiada dodawaniu modulo 2. Zauważmy, że $Ng(x) = Ar(x, \mathbf{1}(x))$ oraz że Kn odpowiada (zwykłemu) mnożeniu w zbiorze $\{0, 1\}$.

Czasami zamiast $Kn(x, y)$ piszemy xy, zamiast $Ar(x, y)$ piszemy $x + y$, a zamiast $\mathbf{1}$ po prostu 1. Iloczyny zmiennych nazywamy jednomianami, sumy jednomianów wielomianami Żegałkina, a pusty iloczyn zmiennych utożsamiamy ze stałą 1.

Zupełne układy funkcji prawdziwościowych

Twierdzenie. Każda funkcja prawdziwościowa ma dokładnie jedno przedstawienie w postaci wielomianu Żegałkina (z dokładnością do kolejności czynników w jednomianach i składników w wielomianie).

Zauważmy, że:

- (a) funkcje liniowe są przedstawialne jako sumy skończenie wielu jednomianów prostych (tj. jednomianów bez mnożenia)
- (b) funkcje przedstawialne przez funkcje liniowe także są liniowe
- (c) z (a) oraz (b) wynika, że nie są zupełne np. układy: $\{+, 1\}$ oraz $\{Ng, Rw\}$.

Nie są zupełnymi także np. układy: $\{Rw, Ar\}$, $\{Al, Kn, Im\}$.

Binegacja i kreska Sheffera

- binegacja (NOR, strzałka Peirce'a): $\downarrow(x, y) = Ng(Al(x, y))$
- kreska Sheffera, NAND: $\uparrow(x, y) = Ng(Kn(x, y))$

$$Ng(x) = \downarrow(x, x) \quad Al(x, y) = \downarrow(\downarrow(x, y), \downarrow(x, y))$$
$$Ng(x) = \uparrow(x, x) \quad Kn(x, y) = \uparrow(\uparrow(x, y), \uparrow(x, y))$$

Binegacja odpowiada spójnikowi „ani ..., ani ...", a kreska Sheffera spójnikowi „co najwyżej jedno z dwojga ..., ...".

Twierdzenie. Jedyne zupełne jednoelementowe układy funkcji to: $\{\uparrow\}$ oraz $\{\downarrow\}$.

Przykłady zbiorów niezależnych i baz

Przykładami niezależnych układów funkcji są:
(a) $\{Ng, Rw\}$; (b) $\{Ng, Ar\}$; (c) $\{Rw, Ar\}$; (d) $\{Rw, Al\}$.

Zupełne i niezależne są np. następujące układy funkcji:
(a) $\{Im, /\}$, gdzie $/(x, y) = Ng(Im(y, x))$;
(b) $\{Rw, Al, \mathbf{0}\}$, gdzie $\mathbf{0}$ jest funkcją stałą równą 0.

- $\{Kn, Im\}$ jest bazą dla $\boldsymbol{C}_1$
- $\{Kn, Ar\}$ jest bazą dla $\boldsymbol{C}_0$
- $\{Al, Kn, \mathbf{0}, \mathbf{1}\}$ jest bazą dla $\boldsymbol{M}$
- $\{\mathbf{0}, Rw\}$ jest bazą dla $\boldsymbol{L}$
- $\{Ng, \curlywedge\}$ jest bazą dla $\boldsymbol{D}$, gdzie $\curlywedge(x, y, z) = xy + xz + yz$.

Klasy prapełne i twierdzenie Posta

- Klasy: $\boldsymbol{C}_1, \boldsymbol{C}_0$, $\boldsymbol{M}$, $\boldsymbol{D}$, $\boldsymbol{L}$ są wszystkie prapełne.
- Dowolna klasa zamknięta $\boldsymbol{K} \neq \boldsymbol{C}$ zawiera się w pewnej klasie prapełnej.
- Układ funkcji jest zupełny wtedy i tylko wtedy, gdy nie jest zawarty w żadnej klasie prapełnej.

Twierdzenie Posta. Nie istnieją klasy prapełne różne od $\boldsymbol{C}_1$, $\boldsymbol{C}_0$, $\boldsymbol{L}$, $\boldsymbol{D}$ oraz $\boldsymbol{M}$.

- Każdy zamknięty zbiór funkcji prawdziwościowych ma skończoną bazę.
- Rodzina wszystkich zamkniętych zbiorów funkcji prawdziwościowych jest przeliczalna.