RAZŠIRITEV PEIRCEOVIH ALFA GRAFOV

JANEZ IGNACIJ JEREB

Fakulteta za matematiko in fiziko
Univerza v Ljubljani

V tem delu je predstavljen Peirceov alfa sistem skupaj z dokazoma njegovega zdravja in polnosti. Uvedena je razširitev, ki je uporabljena za sestavo in dokaz zdravja diagramatične racionalne Pavelkove logike ter diagramatične izjavne linearne logike. S pomočjo računalnika je bilo odkritih nekaj trovrednostnih logik, primernih za uporabo te razširitve.

AN EXTENSION OF PEIRCE'S ALPHA GRAPHS

In this work, Peirce's alpha graphs are examined along with their proofs of soundness and completeness. An extension is introduced, which is used to construct and prove the soundness of diagrammatic Rational Pavelka Logic and diagrammatic propositional linear logic. With the help of a computer, several three-valued logics were found to be useful in applying this generalization.

1. Uvod

Leta 1896 je Peirce razvil sistem, za katerega je verjel, da je njegovo največje odkritje [1]. Menil je tudi, da bi ta sistem moral postati logika prihodnosti. Sodobniki žal niso videli koristi v njegovi notaciji. Quine je celo rekel:

> One questions the efficacy of Peirce's diagrams, however, in their analytical capacity as well. Their basic machinery is too complex to allow one much satisfaction in analyzing propositional structure into terms of that machinery. While it is not inconceivable that advances in the diagrammatic method might open possibilities of analysis superior to those afforded by the algebraic method, yet an examination of Peirce's product tends rather, apagogically as it were, to confirm one's faith in the algebraic approach. [2]

K takemu pogledu je prispevalo več težav. Peirce, po lastnem priznanju, ni znal dobro razložiti diagramov. Diagrami so bili tudi težki za tiskanje. Prav tako nekatera pravila sklepanja niso bila najlažja.

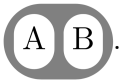
Peirce je razvil alfa, beta in gama sistem. Ti ustrezajo izjavnemu, predikatnemu računu in mešanici višje ter modalne logike. Najbližje praktični uporabi so se sistemi izkazali pri grafih konceptov (ang. conceptual graphs) [3] ter pedagogiki [4]. Vzbudil je zanimanje pri filozofih, saj potencialno predstavlja sistem sklepanja, ki je bližje našemu [5, 6].

V tem delu se omejimo le na alfa grafe. Bistvo tega sistema je, da logični operaciji *in* in *negacijo* predstavimo na grafičen način. *In* napišemo tako, da trditvi zapišemo drugo ob drugi, torej $A \wedge B$ postane $A\ \ B$. Negacijo pa tako, da narišemo elipso okoli trditve, torej $\neg A$ postane 

; prazen prostor predstavlja resnico $\top$.

Pri uporabi bomo večkrat govorili o *globini* podgrafa, kjer globina predstavlja „v koliko elipsah je vsebovan“ neki podgraf. Za lažje sklepanje s pravili bodo prostori v sodi globini beli in v lihi globini sivi.

Ostale veznike sestavimo na preprost način:

- Ali predstavimo z $A \vee B = \neg(\neg A \wedge \neg B)$ z 

.
- Implikacijo pa z $A \to B = \neg(A \wedge \neg B)$ z 

.

Še nekaj primerov za lažjo predstavo:

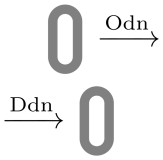
- $(A \wedge B) \to C$ z 

- $(A \vee B) \to (C \wedge D)$ z 

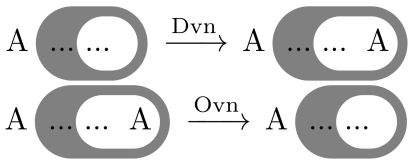
- $A \to (B \to C)$ z 

Alfa sistem uporablja za sklepanje tri pare pravil:

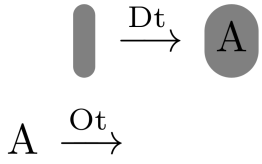
1. **Dodajanje in odstranjevanje dvojnih negacij (Ddn in Odn)**
   Vedno lahko enega ali več elementov dvakrat obkrožimo ali pa odstranimo dvojno obkrožitev.



2. **Nošenje trditve ven in noter (Ovn in Dvn)**
   Katerokoli trditev lahko „kopiramo“ v globlje gnezdene elipse. Lahko gremo tudi v obratno smer.



3. **Dodajanje in odstranjevanje trditev (Dt in Ot)**
   Na sodi globini lahko poljubno izjavo odstranimo, na lihi pa jo lahko dodamo.



Zanimivost pravil je, da jih lahko uporabimo kjerkoli v diagramu, podobno kot ekvivalenca v izjavnem računu. Prav tako sta prvi dve pravili ekvivalenci.

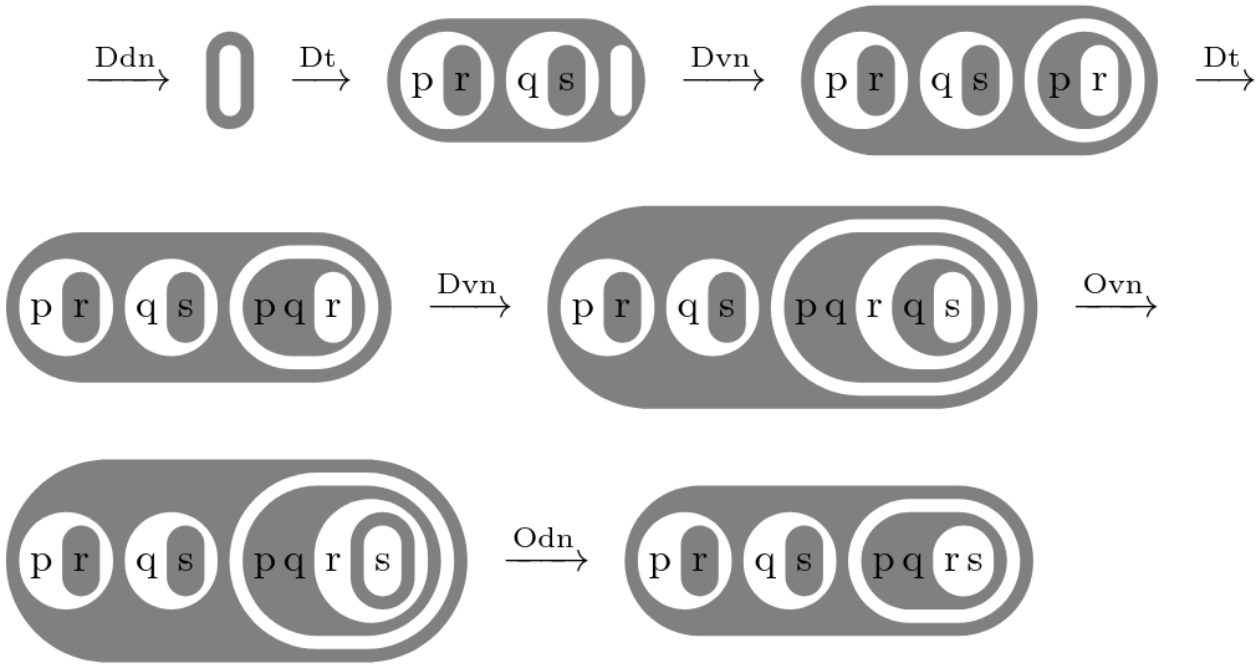
Z uporabo Odt lahko vidimo, da iz 

oz. $A \to (B \to C)$ sledi 

oz. $(A \wedge B) \to C$.

Primer 1. Dokažimo Leibnitzov *Praeclarum Theorema*:

$$((p \to r) \wedge (q \to s)) \to ((p \wedge q) \to (r \wedge s))$$



Slika 1. Dokaz Praeclorum Theorema

Alfa grafe lahko razširimo s predikati in eksistenčnimi kvantifikatorji, s katerimi dobimo beta grafe. To naredimo tako, da dodamo besede, ki predstavljajo predikate/relacije in črte, ki jih povezujejo. Slednje predstavljajo elemente, ki izpolnjujejo predikat oz. predstavljajo kompozicijo relacij. Primer: *moški–ljubiti–ženska* pomeni „obstaja moški, ki ljubi neko žensko“. Iz tukaj se vidi, zakaj se Peirceovim diagramom reče „eksistenčni grafi“ (ang. existential graphs). Beta grafi imajo manj intuitivna pravila sklepanja, a je Shin našla poenostavitev [7]. Kljub temu se beta grafom v tem članku izognemo.

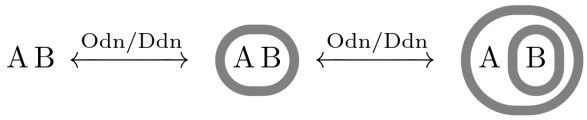
Ni težko videti, da z alfa grafi lahko predstavimo katerokoli izjavno formulo. Kljub temu pa ni takoj vidno, da lahko dokažemo katerikoli resničen sklep v izjavnem računu oziroma, ali je dokazovalni sistem *poln* (ang. complete proof system).

1.1 Dokaz polnosti Peirceovih alfa grafov

Polnosti sistema dokažemo tako, da izpeljemo osnovne aksiome in dokažemo možnost uporabe *modus ponens*. Za to vzamemo poln Churchov P sistem [1]. Ta uporablja modus ponens in aksiomatične sheme:

1. $(P \to (Q \to R)) \to ((P \to Q) \to (P \to R))$
2. $P \to (Q \to P)$
3. $(\neg P \to \neg Q) \to (Q \to P)$

Te sheme uporabljajo implikacijo in negacijo, medtem ko naš sistem uporablja konjunkcijo in negacijo. Kljub temu pravila omogočajo, da prosto pretvarjamo med sistemoma, oziroma da velja $A \wedge B = \neg(A \to \neg B)$:

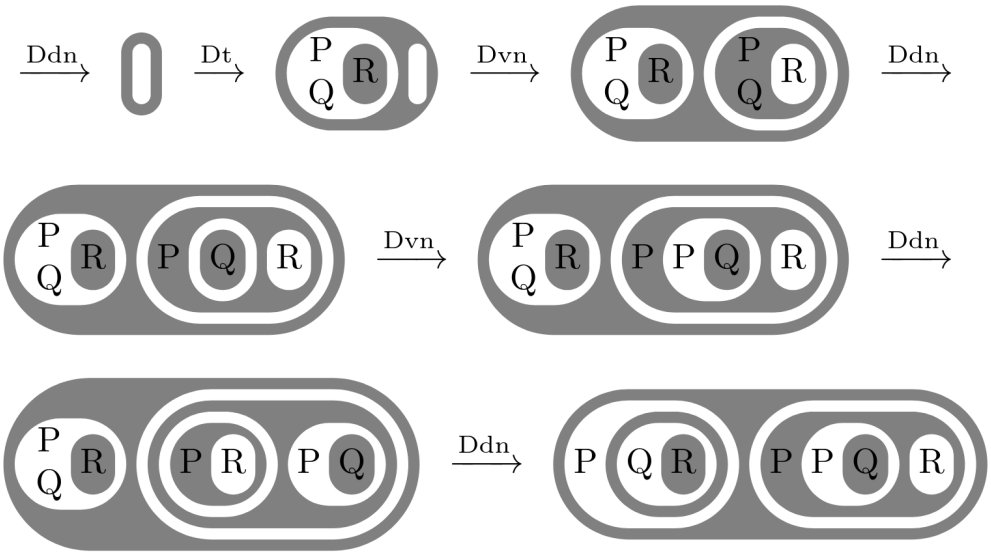


Slika 2. Dokaz pretvorbe implikacij

Možnost uporabe modus ponensa dokažemo z:

Slika 3. Dokaz modus ponensa

Dokažimo tudi, da lahko izpeljemo aksiomatske sheme.
Shemo $(P \to (Q \to R)) \to ((P \to Q) \to (P \to R))$ dokažemo z:

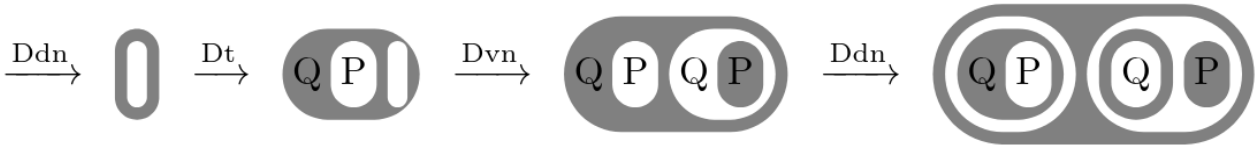


Slika 4. Dokaz prvega aksioma P

Shemo $P \to (Q \to P)$ dokažemo z:

Slika 5. Dokaz drugega aksioma P

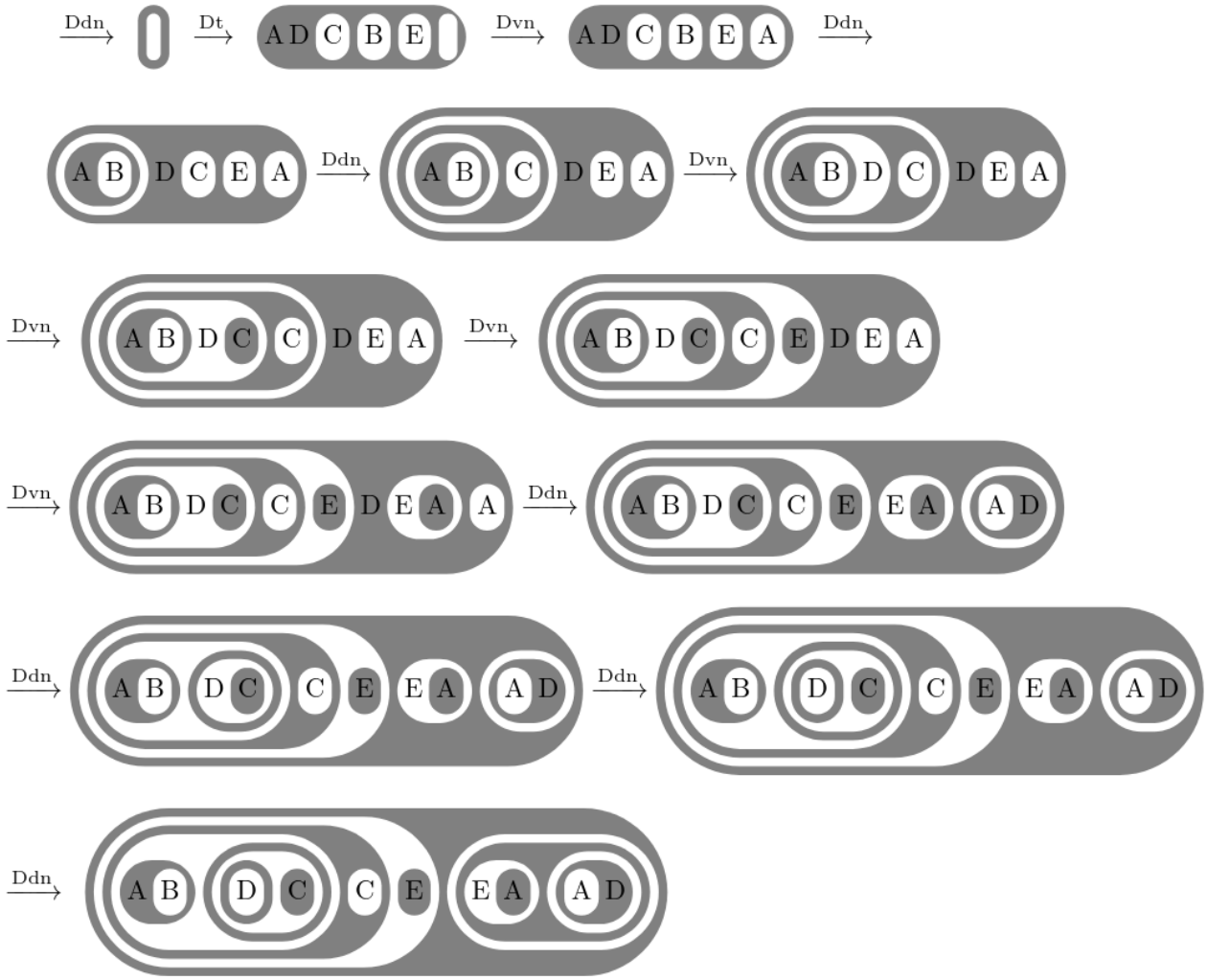
Shemo $(\neg P \to \neg Q) \to (Q \to P)$ dokažemo z:



Slika 6. Dokaz tretjega aksioma P

Kot dodaten izziv dokažimo polnost sistema še na drugačen način – z dokazom Meredithovega aksioma, ki je sam po sebi poln:

$$((((A \to B) \to (\neg C \to \neg D)) \to C) \to E) \to (E \to A) \to D \to A$$



Slika 7. Dokaz Meredithovega aksioma

Vse izvzemši prvih treh korakov dobimo tako, da začnemo od zadaj in nadaljujemo proti začetku z uporabo odstranitve po prvih dveh pravilih (Odn in Ovn). Ker sta pravili ekvivalenci, lahko gremo tudi v drugo smer (Ddn in Dvn).

2. Posplošitev alfa grafov

Za izgradnjo grafičnih sistemov za več logik bomo uporabili posplošitev alfa grafov. Glavne posplošitve so vpeljava več binarnih in unarnih operatorjev, vpeljava konstant in zamenjava implikacije za splošno urejenost.

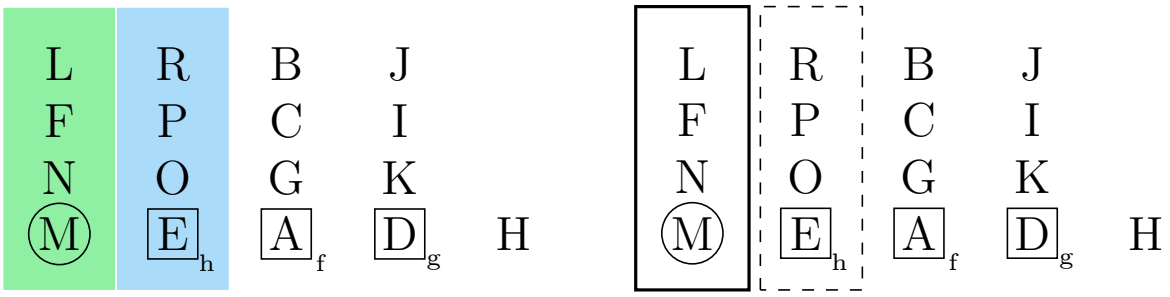
Zahtevamo, da področje pogovora sovpada z resničnostnimi vrednostmi. Označimo ga z D. Definirajmo relacijo izpeljave $\sqsubseteq$ na D, ki posploši implikacijo iz originalnih alfa grafov. Od nje zahtevamo, da je šibka urejenost. Razširimo jo z interpretacijo nad izrazi: $A \models B$ velja med izrazoma A in B, sestavljenima iz operatorjev in spremenljivk semantike, natanko tedaj, ko velja $A \sqsubseteq B$ za vsako prirejanje spremenljivk z vrednostmi iz D. Za binarne operacije $\{\odot_i\}_i^m$ zahtevamo, da so skladne s preurejanjem elementov in so veljavne v „praznem kontekstu“. Morajo biti asociativne, komutativne in imeti enoto, torej morajo tvoriti komutativni monoid nad D. Prav

tako zahtevamo, da so operacije monotone na urejenost izpeljave oz.da iz $A \sqsubseteq B$ in $Z \in D$ sledi $A \odot Z \sqsubseteq B \odot Z$. Unarne operacije označimo z $\{f_i\}_i^n$, konstante pa z $\{s_i\}_i^k$. Za vse skupaj dobimo peterico $(\sqsubseteq, \{\odot_i\}_i^m, \{f_i\}_i^n, \{s_i\}_i^k, D)$. Ta peterica predstavlja *semantiko sistema*.

Za grafičen prikaz logike lahko uporabimo več različnih načinov prikaza. Za unarne operatorje ohranimo obkroževanje. Za druge operatorje uporabimo različne sloge črt ali pa eksplicitno označimo uporabljen operator.

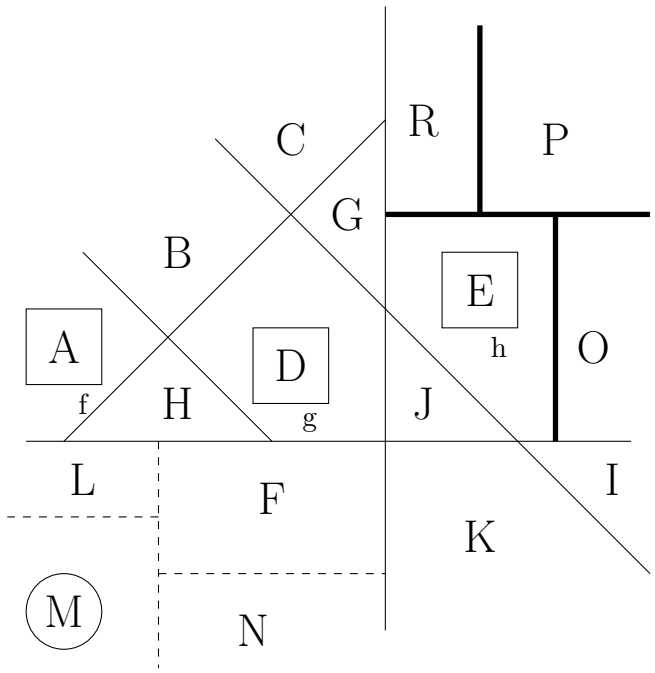
Binarne operatorje lahko označimo na več načinov. Slike simbolizirajo $f(A) \oplus B \oplus C \oplus g(D) \oplus G \oplus H \oplus I \oplus J \oplus K \oplus (L \otimes F \otimes N \otimes \neg M) \oplus (h(E) \odot O \odot R \odot P)$ za poljubno izbrane operatorje. Formalizacija teh zapisov je izpuščena. Nekaj možnih prikazov:

1. Izjave povezane z enim operatorjem označimo z drugače pobarvanim povezanim ozadjem (seveda moramo določiti tudi belo ozadje) (slika 8a).
2. Namesto barvanja uporabljamo obkroževanje (slika 8b).
3. Lahko uporabimo tudi mrežo (slika 8c).



(a) prikaz z obarvanimi ozadji

(b) prikaz z obkroževanjem



(c) prikaz z mrežo

Slika 8. nekaj možnih prikazov izrazov

Prva dva načina imata prednost, da sta premikanje operatorjev ter obravnava praznega prostora bolj intuitivna. Prvi način tudi dobro prikaže asociativnost, saj sta $(A \odot B) \odot C$ in $A \odot (B \odot C)$ grafično prikazana enako. Zadnji prikaz pa ima bolj poudarjeno ločitev med trditvami.

Sistem dokazovanja definiramo s seznamom *pravil sklepanja* $\{A_i \vdash B_i\}_i$. Pravila sklepanja uporabimo z „gnezdenim“ pravilom dedukcije, ki omogoča uporabo pravil sklepanja na katerikoli globini. Dokaz je sestavljen iz zaporedja izrazov, ki jih lahko enega za drugim izpeljemo s pomočjo: gnezdenega pravila dedukcije, seznama pravil sklepanja ali pa algebraičnih lastnosti binarnih operatorjev (asociativnost, komutativnost in enota). Algebraične lastnosti implicirajo, da lahko elemente v območju binarnega operatorja poljubno premikamo. Prav tako lahko katerikoli element obkrožimo z binarnim operatorjem (interpretira se kot $A \odot_i 1_i$) in lahko odstranimo ali dodamo enake gnezdene operatorje (asociativnost).

Z $A \vdash B$ označimo, da obstaja dokaz iz A v B. Zahtevamo tudi, da za vsako dodano pravilo $A_i \vdash B_i$ velja $A_i \models B_i$. Obojestransko izpeljavo $A_i \vdash B_i$, $B_i \vdash A_i$ bomo označili z $A_i \dashv\vdash B_i$.

Za definicijo pravila moramo definirati *B-standardno* in *B-primerno* obliko. B-standardna oblika za A je $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_1}(...A_n \odot_{i_n} B)\dots)$. To je vedno možno konstruirati: če je $A = X_1 \odot_{i_1} X_2 \odot_{i_1} X_3 \odot_{i_1} ... \odot_{i_1} X_l$ in je B vsebovan v X_l ali enak X_l, potem je $A_1 = X_1 \odot_{i_1} X_2 \odot_{i_1} X_3 \odot_{i_1} ... \odot_{i_1} X_{l-1}$. Če je $X_l = B$ smo končali, sicer je $B = f_{j_1}(Y_1)$ in ponovimo postopek na Y_1. To ponavljamo, dokler ne pridemo do B. Če izraz sestavljajo unarni operatorji brez binarnih, lahko uporabimo enoto neke binarne operacije. Torej lahko $f_{i_1}(f_{i_2}(X))$ enakovredno izrazimo z $1_{i_1} \odot_{i_1} f_{i_1}(1_{i_2} \odot_{i_2} f_{i_2}(X))$, kjer sta $1_{i_{1,2}}$ enoti. B-standardna oblika je B-primerna, če so vsi unarni operatorji $\{f_{j_k}\}_1^n$ monotoni (iz $A \sqsubseteq B$ sledi $f(A) \sqsubseteq f(B)$) ali antimonotoni (iz $A \sqsubseteq B$ sledi $f(B) \sqsubseteq f(A)$). Definirajmo še koncept globine.

Definicija 2 (Globina izraza)**.** Za A z B-standardno obliko $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_1}(\dots A_n \odot_{i_n} B)\dots)$ je *globina* izraza B število antimonotonih unarnih operatorjev.

Definicija 3 (Gnezdeno pravilo dedukcije)**.** Za izraz A z *B-primerno* standardno obliko $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_n \odot_{i_n} B)\dots)$ velja, da če je globina B soda ter velja $B \vdash C$ ali pa je globina B liha in velja $C \vdash B$, potem lahko sklepamo $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_n \odot_{i_n} B)\dots) \vdash A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_1}(\dots A_n \odot_{i_n} C)\dots)$.

Veljavnost pravila potrdi naslednja lema.

Lema 4 (Gnezdeno sklepanje)**.** Za izraz A z B-primerno obliko $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_n \odot_{i_n} B)\dots)$ velja, da če je globina B soda ter velja $B \models C$ ali pa je globina B liha in velja $C \models B$ potem lahko sklepamo $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_n \odot_{i_n} B)\dots) \models A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_1}(\dots A_n \odot_{i_n} C)\dots)$.

Dokaz. To dokažemo z indukcijo na globino B-primerne oblike.

Če je globina ničelna pomeni, da so vsi operatorji do B monotoni. V tem primeru dokažemo z indukcijo na število unarnih operatorjev. Če jih ni, je globina soda in velja $B \models C$. Sledi $A_1 \odot_{i_1} B \models A_1 \odot_{i_1} C$ iz monotonosti $\odot_{i_1}$. Indukcijski korak pa sledi iz monotonosti. Torej iz $B \models C$ sledi $f_{j_{n+1}}(A_{n+1} \odot_{i_{n+1}} B) \models f_{j_{n+1}}(A_{n+1} \odot_{i_{n+1}} C)$, saj je f_{j_1} monoton. Potem lahko z indukcijsko predpostavko sklepamo

$$A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_{n+1} \odot_{i_{n+1}} B)\dots) \models A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_{n+1} \odot_{i_{n+1}} C)\dots)$$

. Indukcijski korak za globino izraza naredimo za sodo globino. Za liho sledi po podobnem razmisleku. Za globino n začnemo dodajati monotone operatorje v izraz, tj. za k monotonih operatorjev

dobimo

$$A_{n-k+2} \odot_{i_{n-k+2}} f_{j_{n-k+2}}(A_{n-k+3} \odot_{i_{n-k+3}} f_{j_{n-k+3}}(\dots A_{n+1} \odot_{i_{n+1}} B)\dots)$$
$$\models$$
$$A_{n-k+2} \odot_{i_{n-k+2}} f_{j_{n-k+2}}(A_{n-k+3} \odot_{i_{n-k+3}} f_{j_{n-k+3}}(\dots A_{n+1} \odot_{i_{n+1}} C)\dots).$$

To lahko naredimo po prejšnjem razmisleku. Ko pridemo do antimonotonega operatorja, pa uporabimo antimonotonost in dobimo

$$A_{n-k+1} \odot_{i_{n-k+1}} f_{j_{n-k+1}}(A_{n-k+2} \odot_{i_{n-k+2}} f_{j_{n-k+2}}(\dots A_{n+1} \odot_{i_{n+1}} C)\dots)$$
$$\models$$
$$A_{n-k+1} \odot_{i_{n-k+1}} f_{j_{n-k+1}}(A_{n-k+2} \odot_{i_{n-k+2}} f_{j_{n-k+2}}(\dots A_{n+1} \odot_{i_{n+1}} B)\dots).$$

Potem uporabimo indukcijsko hipotezo, da dodamo še preostale operatorje, da dobimo $A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_{n+1} \odot_{i_{n+1}} B)\dots) \models A_1 \odot_{i_1} f_{j_1}(A_2 \odot_{i_2} f_{j_2}(\dots A_{n+1} \odot_{i_{n+1}} C)\dots)$. ∎

Lema 5 (Zdravje pravila dedukcije)**.** Če velja $A \vdash B$, potem velja $A \models B$.

Dokaz. Pravila $A_i \vdash B_i$ veljajo po definiciji. Zdravje gnezdenega pravila sklepanja sledi iz leme 4. Zdravje celotnih dokazov sledi iz tranzitivnosti $\models$, ki sledi iz tranzitivnosti $\sqsubseteq$. ∎

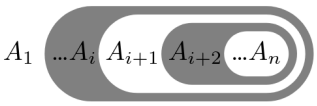
3. Primeri uporabe posplošitve

3.1 Peirceov alfa račun

Posplošena semantika Peircove logike je $(\rightarrow, \{\wedge\}, \{\neg\}, \emptyset, \{\top, \bot\})$. Operatorji ustrezajo potrebnim zahtevam, da velja pravilo gnezdene dedukcije. Pravila sklepanja (sheme) so $A \wedge B \vdash A$, $A \wedge \neg(B \wedge A) \vdash A \wedge \neg B$ in $A \dashv\vdash \neg\neg A$, kjer sta A in B poljubna izraza.

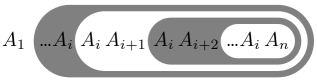
Pravili dodajanja in odstranjevanja dvojnih negacij sledita iz $\neg\neg A \dashv\vdash A$. Pravilo dodajanja in odstranjevanja trditev sledi iz $A \wedge B \vdash A$. Pravila za nošenje ven in noter sistem ne porodi sam od sebe, a ga lahko dokažemo z indukcijo in shemo $\neg(A \wedge \neg B) \vdash \neg(A \wedge \neg(B \wedge A))$. Na sodi globini to pravilo ni potrebno zaradi pravila odstranjevanja trditev, vendar je potrebno na lihi globini.

Dokažimo, da lahko noter nesemo trditev. Začnemo na poljubni globini s trditvijo, ki jo želimo nesti noter.



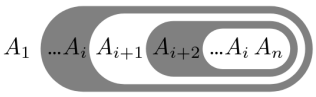
Slika 9. Začetno stanje

Uporabimo shemo in Dt, da trditev kopiramo do željene globine.



Slika 10. „Kopiramo“ trditev

Potem uporabimo isto shemo in Ot, da odstranimo vse vmesne trditve.



Slika 11. Odstranimo vmesne trditve

Zdravje Odt dokažemo na podoben način. S tem smo dokazali zdravje zadnjega pravila in zdravje Peirceovega alfa računa.

V prvem primeru uporabe razširitve bomo videli nenavaden primer logike, ki pokaže njeno fleksibilnost.

3.2 Urejenost aritmetičnih izrazov

Naredimo logiko za sklepanje o velikosti realnih aritmetičnih izrazov. Vsebuje naj seštevanje, odštevanje, množenje in deljenje, ter $+1$. Seštevanje in množenje dodajmo kot binarni operaciji. Deljenje in odštevanje pa predstavimo kot množenje (oz. seštevanje) z inverzom. Inverza skupaj z operacijo $+1$ dodamo kot unarne operatorje. S tem dobimo semantiko $(\leq, \{\cdot, +\}, \{x \mapsto \frac{1}{x}, x \mapsto -x, x \mapsto x+1\}, \mathbb{R})$.

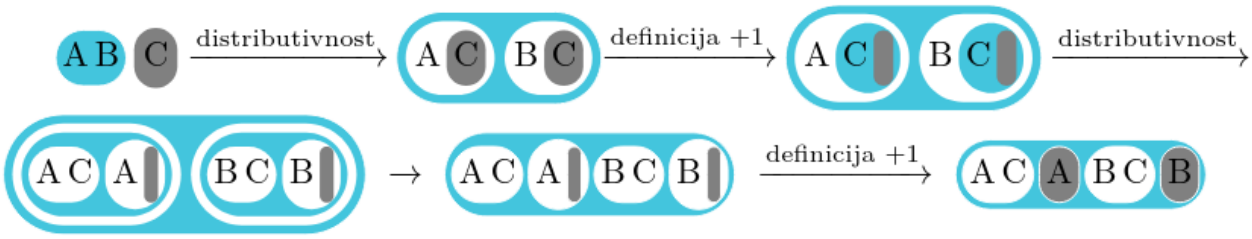
Operaciji za inverz sta antimonotoni. Operacija $+1$ je monotona. Sledi, da gnezdeno pravilo dedukcije deluje za vse unarne operacije.
Pravila sklepanja (sheme) so:

- naraščanje: $a \vdash a + 1$
- aditivni inverz: $a + (-a) \dashv\vdash 0$
- multiplikativni inverz: $a \cdot a^{-1} \dashv\vdash 1$
- distributivnosti: $a \cdot (b + \cdots + c) \dashv\vdash a \cdot b + \cdots + a \cdot c$
- definicija $+1$: $(+1)(a) \dashv\vdash a + ((+1)(0))$

za vsako število $a, b, c \in \mathbb{R}$.

Seštevanje bo prikazano z belim ozadjem, množenje s svetlomodrim in operacija $+1$ s sivim.

Primer 6. Dokažimo $(a + b) \cdot (c + 1) \vdash a \cdot c + b \cdot c + a + b$



Slika 12. Dokaz preproste neenakosti

Nadaljujmo s kakšnim bolj zanimivim primerom logike.

3.3 Trovrednostne logike

Ene prvih primerov neklasične logike so trovrednostne logike. Te logike imajo poleg resnice in neresnice še eno dodatno resničnostno vrednost. Pri različnih logikah ima ta vrednost različne interpretacije [8]: nedefinirano, delno resnično, nedokazano ter druge možne interpretacije.

V tem delu poskusimo najti logiko, ki omogoča uporabo gnezdenega pravila dedukcije. Za urejenost izpeljave uporabimo standarno relacijo teh logik. Najprej definiramo *dodeljene vrednosti* (ang. designated values). To so vrednosti, ki jih jemljemo kot „resnične“. Pri tem se osredotočimo na število dodeljenih vrednosti, saj lahko vrednosti preimenujemo. Za resničnostne vrednosti bomo

vzeli števila $-1, 0, 1$. Gledali bomo primere dodeljenih vrednosti $D = \{1\}$ ali $D = \{0, 1\}$. Relacija izpeljave $A \sqsubseteq B$ velja natanko tedaj, ko velja: če ima A dodeljeno vrednost, jo mora imeti tudi B.

Od negacije bomo zahtevali tri lastnosti: antimonotonost, slikanje dodeljenih vrednosti v nedodeljene ter obratno. Za operacijo *in* pa seveda zahtevamo vse potrebno, da bo uporabna za posplošitev.

Z uporabo variacij programa, priloženega k temu delu (program 1), bomo preiskali vse možne operatorje, ki upoštevajo prej omenjene zahteve. Program je bil napisan v modelirnem jeziku *Minizinc* [9], ki je namenjen diskretnemu modeliranju. Program opiše zahteve in prostor vseh možnih operatorjev. Uporabljen je bil privzeti reševalec *Gecode* [10], ki izčrpno preišče prostor vseh možnosti.

Pri preiskovanju je našel dve možnosti za eno dodeljeno vrednost in dve za dve dodeljeni vrednosti. V tabeli 1 so prikazane vse antimonotone operacije ter njihovo upoštevanje lastnosti. Nekonstantnih kandidatov za operacijo *in* je 20. Prikaz teh izpustimo.

Dodeljene vrednosti	-1	0	1	Slika dodeljene v nedodeljene	Slika nedodeljene v dodeljene	$A \sqsubseteq \neg\neg A$	$A \sqsubseteq \neg\neg A$
$\{1\}$	-1	-1	-1	✓	✗	✗	✓
$\{1\}$	0	-1	-1	✓	✗	✗	✓
$\{1\}$	-1	0	-1	✓	✗	✗	✓
$\{1\}$	0	0	-1	✓	✗	✗	✓
$\{1\}$	1	1	-1	✓	✓	✓	✓
$\{1\}$	-1	-1	0	✓	✗	✗	✓
$\{1\}$	0	-1	0	✓	✗	✗	✓
$\{1\}$	-1	0	0	✓	✗	✗	✓
$\{1\}$	0	0	0	✓	✗	✗	✓
$\{1\}$	1	1	0	✓	✓	✓	✓
$\{1\}$	1	1	1	✗	✓	✓	✗
$\{0, 1\}$	-1	-1	-1	✓	✗	✗	✓
$\{0, 1\}$	0	-1	-1	✓	✓	✓	✓
$\{0, 1\}$	1	-1	-1	✓	✓	✓	✓
$\{0, 1\}$	0	0	0	✗	✓	✓	✗
$\{0, 1\}$	1	0	0	✗	✓	✓	✗
$\{0, 1\}$	0	1	0	✗	✓	✓	✗
$\{0, 1\}$	1	1	0	✗	✓	✓	✗
$\{0, 1\}$	0	0	1	✗	✓	✓	✗
$\{0, 1\}$	1	0	1	✗	✓	✓	✗
$\{0, 1\}$	0	1	1	✗	✓	✓	✗
$\{0, 1\}$	1	1	1	✗	✓	✓	✗

Tabela 1. Antimonotoni unarni operatorji

Dodatno se lahko vprašamo, ali semantike izpolnjujejo lastnosti $\neg\neg A \sqsubseteq A$, $A \sqsubseteq \neg\neg A$ ter $A \wedge \neg(B \wedge A) \sqsubseteq A \wedge \neg B$. Semantikam, ki izpolnjujejo vse te lastnosti, jih lahko dodamo kot pravila sklepanja. Ta nam omogočajo uporabo vseh pravil Peirceovega alfa računa razen Dt in Ot. Izkaže se, da vsi antimonotoni unarni operatorji upoštevajo vsaj eno izmed zadnjih dveh lastnosti. Obe izpolnjujejo natanko tisti, ki upoštevajo dodeljenost pri slikanju. Vse kombinacije z *in* ter negacijami upoštevajo prvo lastnost. Lastnosti $A \wedge B \sqsubseteq A$, $A \sqsubseteq A \wedge B$ z antimonotono negacijo in neko *in*

operacijo ima 44 semantik. Pri 22 velja prva, pri drugih 22. Nobena nima obeh lastnosti.

Izkaže se, da lahko monotone unarne operatorje razdelimo v dve skupini: konstantne in tiste, ki ohranjajo (ne)resnično vrednost.

Zanimivo dejstvo pri teh logikah je, da nobena izmed njih ni standardna oz. ne ustreza nobeni izmed 13 logik v [8].

3.4 Pavelkova racionalna logika (PRL)

Pavelkova racionalna logika (ang. Rational Pavelka logic) je logika za mehko sklepanje (ang. fuzzy logic). Ta omogoča sklepanje z *delno* resnico, ki jo predstavimo z racionalnimi števili med 0 in 1. V tej logiki lahko izrazimo trditve kot „steklenica je pol polna“. To izrazimo, tako da resničnostni vrednosti za trditev „steklenica je polna“ priredimo 0.5. Čeprav imamo resničnostne vrednosti med 0 in 1, ne moremo te vrednosti interpretirati kot verjetnosti. Na primer: če je verjetnost dogodka A enaka 0.2 in verjetnost dogodka B enaka 0.3, ni nujno, da je verjetnost $A \wedge B$ enaka $0.2 \cdot 0.3 = 0.6$. Dogodka sta morda odvisna in imata posledično lahko mnogo različnih verjetnosti odvisnih od konteksta. Mehke logike ne morejo izraziti teh kontekstov.

Razširjena semantika za Pavelkovo racionalno logiko je $([0,1], \leq, \{*\}, \{\neg\}, [0,1])$. Negacija je definirana z $\neg a = 1 - a$. Operator $*$ je *Łukasiewiczeva t-norma*, ki je definirana z $a * b = \max(0, a + b - 1)$ in predstavlja operacijo *in*, kar se vidi iz seštevanja resničnostnih vrednosti.

Standardno [11] se vso logiko definira z implikacijo

$$a \Rightarrow b = \begin{cases} 1 & a \leq b \\ 1 - a + b & a > b \end{cases}.$$

Po preprostem izračunu vidimo $a \Rightarrow b = \neg(a * \neg b)$.

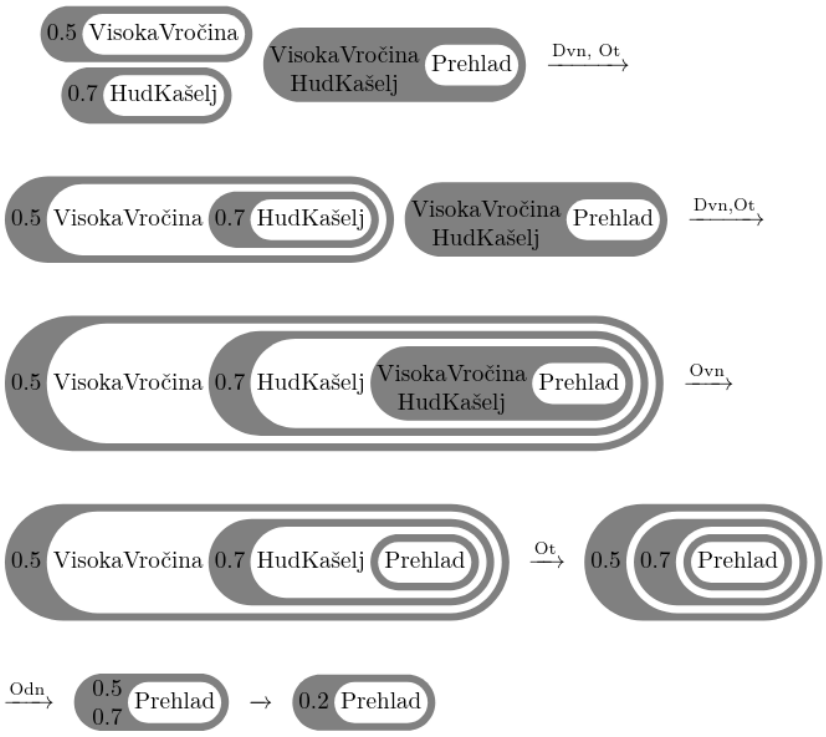
Łukasiewiczeva t-norma je monotona, asociativna, komutativna in ima enoto 1. Negacija je antimonotona kakor pri prejšnjem primeru. Dodamo še pravila sklepanja, ki smo jih dodali Peirceovim alfa grafom: $A * B \vdash A$, $A * \neg(B * A) \vdash A * \neg B$, $\neg\neg A \dashv\vdash A$ ter shemi $(\overline{r} \to \overline{s}) \dashv\vdash \overline{r \Rightarrow s}$ in $\overline{r} \dashv\vdash \overline{1 - r}$, ki omogočata „poenostavljanje“ konstantnih izrazov.
Pri dokazovanju zdravja in polnosti našega računa se bomo oprli na spodaj podan standarden poln in zdrav Hilbertov sistem za Pavelkovo racionalno logiko [11]:

$$\phi \to (\psi \to \phi) \tag{1}$$
$$(\phi \to \psi) \to ((\psi \to \chi) \to (\phi \to \chi)) \tag{2}$$
$$(\neg\phi \to \neg\psi) \to (\psi \to \phi) \tag{3}$$
$$((\phi \to \psi) \to \psi) \to ((\psi \to \phi) \to \phi) \tag{4}$$
$$(\overline{r} \to \overline{s}) \to \overline{r \Rightarrow s} \qquad \text{za racionalni } r, s \tag{5}$$
$$\neg\overline{r} \to \overline{1 - r} \qquad \text{za racionalen } r \tag{6}$$

Zdravje dodanih pravil sklepanja in lastnosti operatorjev je prikazana v [11], kjer je tudi prikazano, da so vse tavtologije v izjavnem računu tudi tavtologije v PRL.

Polnost dokažemo z izpeljavo gornjih pravil. Implikacijo predstavimo na enak način kot pri alfa računu. Prva štiri pravila sklepanja se da izpeljati podobno kot pri Peirceovem računu, saj so tudi tavtologije v izjavni logiki. Zadnji dve pa sledita iz zadnjih dveh dodanih shem.

Primer 7. Imamo računalniški sistem za diagnozo prehladov. Pravila so spravljena v obliki PRL. Sistem ima shranjeno, da če ima nekdo visoko vročino in hud kašelj, je prehlajen. Sedaj ugotovimo, kaj bo sklepal sistem, če ima visoko vročino z močjo 0.5 in hud prehlad z močjo 0.7.



Slika 13. Dokaz preprostega PRL sklepa

Torej ima možnost prehlada moč 0.2.

3.5 Izjavna linearna logika

Za zadnji primer pa podajmo sistem dokazovanja za izjavno linearno logiko. Začetnik linearne logike je Jean-Yves Girard [12]. Je posplošitev navadne logike. Trditve v linearni logiki so viri z omejeno uporabo. Torej, če uporabimo neko trditev pri enem sklepu, je ne moremo uporabiti še enkrat. Prav tako se trditev ne moremo kar tako znebiti. Standarden primer: imamo kovanec in napravo, ki vrne čokolado za en kovanec.

$$\text{kovanec} \otimes (\text{kovanec} \multimap \text{čokolada})$$

iz tega lahko sklepamo

$$\text{čokolada}$$

vendar pa ne

$$\text{čokolada} \otimes \text{kovanec}$$

ali pa

$$\text{kovanec} \multimap \text{čokolada}.$$

Zanimivo je, da ta ločitev razbije *in* in *ali* v dve obliki. *In* se razbije v $\otimes$ in &, *ali* pa v $\oplus$ in $\parr$. Za povrnitev klasične logike uporabimo unarne operatorje, ki predstavljajo neskončno ustvarjanje in požiranje formul. Označimo ju s klicajem in vprašajem. Pomen nekaterih se ponavadi prikaže z naslednjim primerom [13]. V neki angleški restavraciji lahko za 5 funtov naročimo meni. Meni vsebuje ribo in krompirček. Lahko izberemo ali juho ali solato. Odvisno od dneva pa imamo na voljo ali sadje ali pa sir. Na voljo imamo kolikor hočemo kave. To predstavimo z naslednjo formulo:

$$(F \otimes F \otimes F \otimes F \otimes F) \multimap (\text{riba} \otimes \text{krompirček} \otimes (\text{juha} \,\&\, \text{solata}) \otimes (\text{sadje} \oplus \text{sir}) \otimes !\text{kava})$$

Pomen $A \parr B$ je žal manj razumljiv, vendar si lahko predstavljamo, da imamo A in B, a ju ne moremo uporabiti hkrati.

Semantiko bomo definirali s standardno [14] *fazno semantiko* za linearno logiko. V tej semantiki trditve predstavljajo množice dovoljenih kombinacij virov. Operatorji pa operacije nad njimi. Razširjeno semantiko podamo z $(\sqsubseteq, \{\otimes, \oplus\}, \{!, (-)^\perp\}, \mathcal{P}(M))$, kjer $(M, \cdot)$ tvorita komutativen monoid z enoto 1, $\mathcal{P}(M)$ je množica podmnožic M in imamo definirano množico $F \subseteq M$. Množica F predstavlja množico „prepovedanih“ kombinacij. Urejenost je definirana z $x \sqsubseteq y :\Leftrightarrow \ 1 \in x \multimap y$, kjer $A \multimap B := \{m \in M \mid \forall n \in A. \ \ m \cdot n \in B\}$. Operatorji so definirani z: $X^\perp := X \multimap F$, $A \otimes B := \{m \cdot n \mid m \in A, n \in B\}^{\perp\perp}$, $A \oplus B := (A \cup B)^{\perp\perp}$ in $!A := (A \cap I \cap \{1\}^{\perp\perp})$, kjer je I množica vseh idempotentnih elementov v M. Operatorja $\otimes$ in $\oplus$ sta asociativna, komutativna in imata enoto. Veljajo de Morganovi zakoni: $(A \otimes B)^\perp = A^\perp \parr B^\perp$, $(A \parr B)^\perp = A^\perp \otimes B^\perp$, $(A \oplus B)^\perp = A^\perp \mathbin{\&} B^\perp$, $(A \mathbin{\&} B)^\perp = A^\perp \oplus B^\perp$. Konstante se slikajo z $1^\perp = \bot$, $\bot^\perp = 1$, $0^\perp = \top$, $0^\top = \bot$. Za modalna operatorja pa velja: $(!A)^\perp = ?(A^\perp)$, $(?A)^\perp = !(A^\perp)$. Te lastnosti so pokazane v [14].

Primer 8 (Fazna semantika proračuna)**.** V tem primeru bomo sestavili semantiko za sklepanje o omejenem proračunu. Vzemimo komutativni monoid $(\mathbb{N}, +, 0)$ za fazno semantiko. Naj bo $F = \{a \in \mathbb{N} \mid a > p\}$, kjer p predstavljajo omejitev proračuna. Dodajmo interpretacije za vsa naravna števila. Vsako naravno število v formuli interpretiramo kot množico večjih ali enakih števil zapisanega števila. Torej 3 interpretiramo kot $\{x \mid x \geq 3\}$.

Sedaj lahko vidimo, da $3 \otimes 4 = \{m + n \mid m \geq 3, n \geq 4\} = \{k \mid k \geq 7\}$ in $1 \oplus 4 = (\{m \mid m \geq 1\} \cup \{m \mid m \geq 4\})^{\perp\perp} = \{m \mid m \geq 1\}$, kar ustreza intuiciji. Trditev je resnična, ko njeni interpretaciji pripada 0. Torej, ko lahko „plačamo“ za trditev.

Poskusimo videti ali lahko s tremi enotami plačamo pet enot. Trditev zapišemo z $3 \multimap 5$. Njena interpretacija je $\{b \in \mathbb{N} \mid \forall a \geq 3. \ \ a + b \geq 5\} = \{n \mid n \geq 2\}$. Število nič ne pripada tej množici, torej trditev ni resnična v tej semantiki.

Nadaljujmo še z definicijo dokaznega sistema. Pri izbiri shem smo si pomagali z [15, 16]. Dokaz zdravja izpustimo zaradi dolžine. Je podan v [16]. Sheme za dokazni sistem:

- **Sheme za negacijo**
  Dvojno negacijo lahko vedno dodamo ali odstranimo:
  $$A \dashv\vdash (A^\perp)^\perp.$$
  Če imamo na sodi globini neko izjavo in v eni negaciji globlje enako izjavo, potem ju lahko odstranimo. Na lihi globini lahko dodamo katerokoli trditev, če jo dodamo tudi v eni negaciji globlje:
  $$A \otimes B \otimes (B \otimes C)^\perp \vdash A \otimes C^\perp.$$
  Izjavo lahko premikamo preko dveh negacij noter v sodi globini in ven v lihi globini:
  $$A \otimes B \otimes (C \otimes (D)^\perp)^\perp \vdash B \otimes (C \otimes (D \otimes A)^\perp)^\perp.$$
  Če je izjava sama lahko tudi odstranimo negacijo:
  $$A \otimes A^\perp \vdash 1_\otimes.$$
- **Sheme za ali ($\oplus$)**
  Ponovljene izbire lahko odstranimo:
  $$A \oplus A \vdash A.$$
  Izbire lahko dodajamo:
  $$A \vdash A \oplus B.$$

- **Sheme za sklepanje z modalnim operatorjem !**
  Klicaj je operator poljubnega števila izjav (tudi 0):

$$!A \vdash 1_{\otimes} \qquad !A \vdash A \qquad !A \vdash !A \otimes !A.$$

  Kombinacija poljubnega števila trditev je enakovredna poljubnemu številu takih kombinacij:

$$!A_1 \otimes \cdots \otimes !A_n \dashv\vdash !(\, !A_1 \otimes \cdots \otimes !A_n \,).$$

- **Odnos med $\oplus$ in $\otimes$**
  Velja distributivnost med $\otimes$ in $\oplus$:

$$(A \otimes B) \oplus (A \otimes C) \dashv\vdash A \otimes (B \oplus C).$$

  Vendar **ne** velja $(A \oplus B) \otimes (A \oplus C) \dashv\vdash A \oplus (B \otimes C)$.

Zaključimo z dokazom $(A \multimap B) \otimes (B \multimap C) \vdash A \multimap C$. Tukaj bomo uporabili le $\otimes$ in bomo zato barvali podobno kot pri izvornih alfa diagramih.

Slika 14. Dokaz hipotetičnega silogizma za linearno logiko

4. Zaključek

V tem delu smo si ogledali izvorni Peirceov alfa račun ter ga z razširitvijo aplicirali na Pavelkovi racionalni logiki, linearni izjavni logiki in trovrednostnih logikah. V prihodnosti bi lahko poskusili uporabiti razširitev tudi na drugih mehkih logikah. Prav tako bi si lahko bolj podrobno ogledali dobljene trovrednostne logike in jim dali primerno interpretacijo. Med delom se je izkazalo, da je ta razširitev neprimerna za določene logike. Verjetno bi bilo možno najti boljšo razširitev, morda po zgledu [16]. Delo bi bilo možno nadgraditi na Peirceove beta grafe.

5. Priloga

```
% resnicnostne vrednosti
set of int: TV = -1..1;

% dodeljene resnicnostne vrednosti
set of TV: DV = {0,1};

% deklaracije operatorjev
array[TV,TV] of var TV: and;
array[TV] of var TV: pnot;

% definicija relacije izpeljave
predicate ord(var TV: x, var TV: y) = x in DV -> y in DV;

% asociativnost, enota, komutativnost, monotonost ‘and’
constraint forall(u,v,w in TV)( and[and[u,v],w] = and[u,and[v,w]] );
```

```
constraint exists(v in TV)( forall(u in TV)( and[u,v] = v ) );
constraint forall(u,v in TV)( and[u,v] = and[v,u] );
constraint forall(x,y,z in TV)( ord(x,y) -> ord(and[x,z], and[y,z]) );

% antimonotonost, slikanje dodeljene v nedodeljene in obratno 'not'
constraint forall(u,v in TV)( ord(u,v) -> ord(pnot[v], pnot[u]) );
constraint forall(u,v in TV)( u in DV -> not (pnot[u] in DV) );
constraint forall(u,v in TV)( not (u in DV) -> pnot[u] in DV );

% dodatne lastnosti
constraint forall(a,b in TV)( ord(and[a,pnot[and[b,a]]], and[a,pnot[b]]) );
constraint forall(a in TV)( ord(a,pnot[pnot[a]]) );
constraint forall(a in TV)( ord(pnot[pnot[a]],a) );

solve satisfy;
```

Program 1. Program za iskanje ustreznih trovrednostnih logik

LITERATURA

[1] Don D. Roberts, *The existential graphs of Charles S. Peirce*, Number 27. Walter de Gruyter, 1973.

[2] Willard Van Orman Quine, *Review of the collected papers of Charles Sanders Peirce, volume 4: The simplest mathematics*, Isis **22** (1934), 551–553.

[3] John F. Sowa, *Conceptual graphs as a universal knowledge representation*, Computers & Mathematics with Applications **23** (1992), 75–93.

[4] John F. Sowa, *Peirce's tutorial on existential graphs* (2011).

[5] Ahti-Veikko Pietarinen, *Existential graphs: What a diagrammatic logic of cognition might look like*, History and Philosophy of Logic **32** (2011), 265–281.

[6] Ahti-Veikko Pietarinen. *Challenges and opportunities for existential graphs*, Ideas in action: Proceedings of the Applying Peirce conference, Helsinki: Nordic pragmatism network (2010), 288–303.

[7] Sun-Joo Shin, *Reconstituting beta graphs into an efficacious system*, Journal of Logic, Language and Information **8** (1999), 273–295.

[8] Vitor Greati, Giuseppe Greco, Sérgio Marcelino, Alessandra Palmigiano in Umberto Rivieccio, *Generating proof systems for three-valued propositional logics*, arXiv (2024).

[9] Nicholas Nethercote, Peter J Stuckey, Ralph Becket, Sebastian Brand, Gregory J Duck, and Guido Tack. *Minizinc: Towards a standard cp modelling language*, International Conference on Principles and Practice of Constraint Programming, Springer (2007), 529–543.

[10] Gecode Team, *Gecode: Generic constraint development environment*, http://www.gecode.org, Dostop: 25.5.2025.

[11] Petr Hájek, *Metamathematics of fuzzy logic*, Springer Science & Business Media (2001).

[12] Jean-Yves Girard, *Linear logic*, Theoretical computer science **50** (1987), 1–101.

[13] Richard Crouch in Josef van Genabith, *Linear logic for linguists*, https://www.asc.ohio-state.edu/pollard.4/681/crouch.pdf, Dostop: 5.6.2025.

[14] Jean-Yves Girard, *Linear logic: its syntax and semantics*, London Mathematical Society Lecture Note Series (1995), 1–42.

[15] Adele Lopez, *Visual linear logic*, http://adelelopez.com/visual-linear-logic, Dostopano: junij 2025.

[16] Geraldine Brady and Todd H. Trimble, *A categorical interpretation of C. S. Peirce's propositional logic alpha*, Journal of Pure and Applied Algebra **149** (2000), 213–239.