Univerza v Mariboru

Fakulteta za elektrotehniko, računalništvo in informatiko

Diplomsko delo visokošolskega študija

Uporaba urejenih odločitvenih grafov pri računalniški obdelavi logičnih funkcij

Kandidat: Robert Meolic
Mentor: doc. dr. Zmago Brezočnik

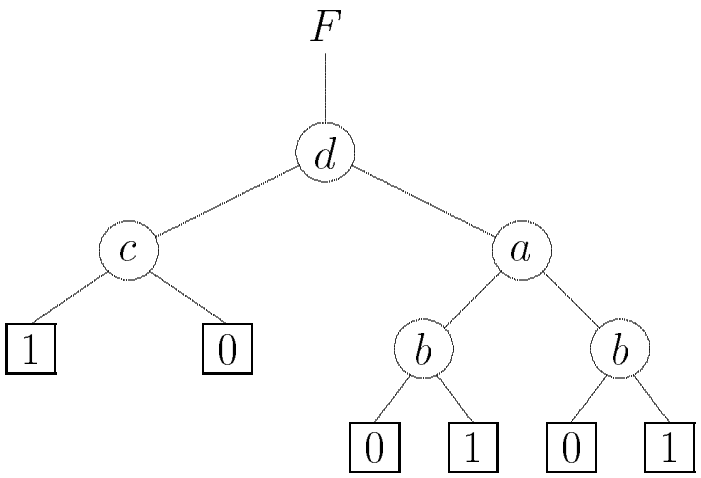
VSEBINA PREDSTAVITVE DIPLOMSKEGA DELA:

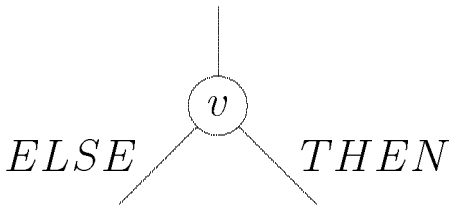
1. kratek uvod,
2. opis odločitvenih grafov,
3. predstavitev OBDD, OFDD in 0-sup-BDD,
4. prikaz opravljenih testov in rezultatov,
5. zaključek.

Uvod

- Odločitveni graf je podatkovna struktura za predstavitev logičnih funkcij.
- Prve učinkovite izvedbe odločitvenih grafov so se pojavile leta 1990.
- Odločitveni grafi so predmet številnih raziskav po vsem svetu.
- **DD** ('Decision Diagram') - odločitveni graf,
  OBDD ('Ordered Binary Decision Diagram') - urejeni binarni odločitveni graf,
  OFDD ('Ordered Functional Decision Diagram') - urejeni funkcijski odločitveni graf,
  0-sup-BDD ('Zero-Suppressed BDD') - binarni odločitveni graf s potlačenimi ničlami.

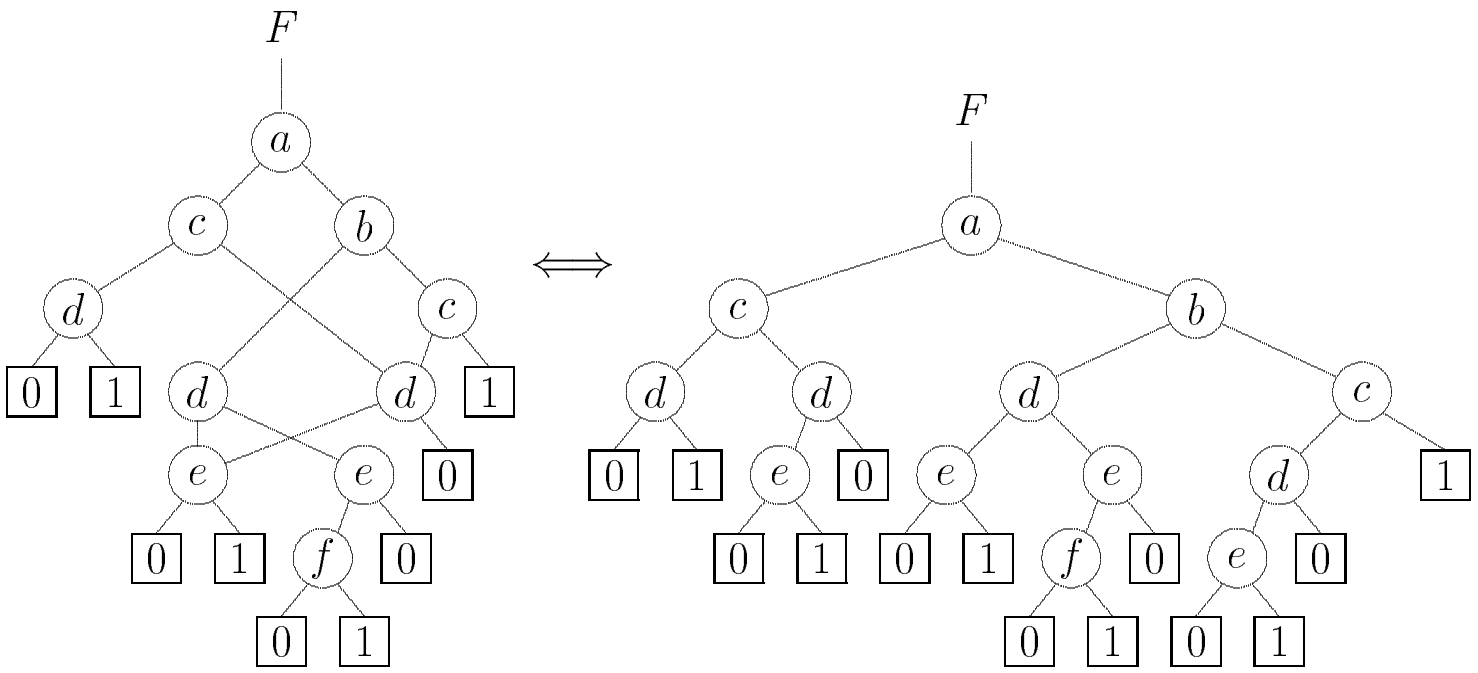
PRIMER ODLOČITVENEGA GRAFA:



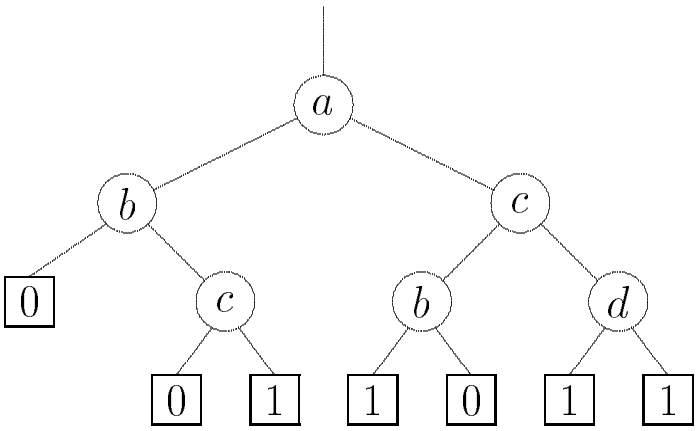


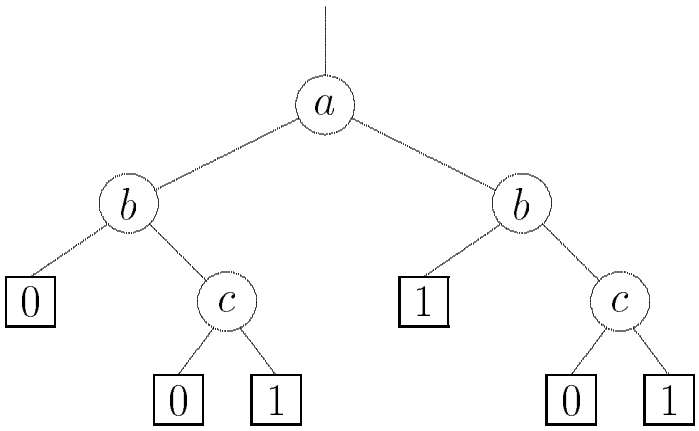
Odločitveni graf je aciklični, usmerjeni graf, sestavljen je iz vozlišč in povezav, ima natanko en koren, vozlišča v listih grafa se imenujejo končna vozlišča.

ZDRUŽEVANJE EKVIVALENTNIH PODGRAFOV



UREJENOST SPREMENLJIVK

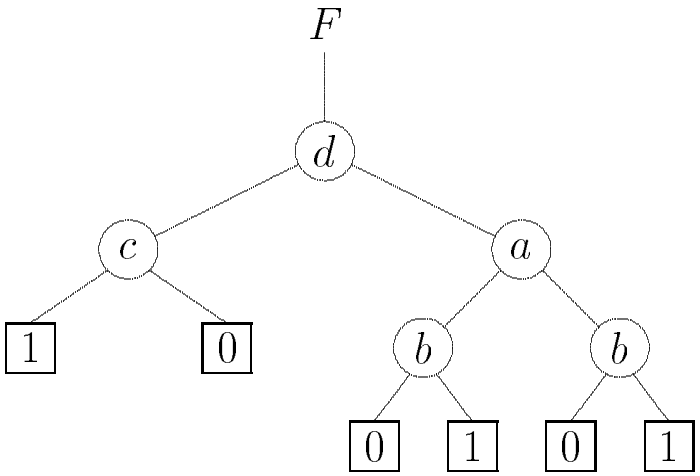




$a < b < c$

odločitveni graf **ni urejen**

odločitveni graf **je urejen**



pri ROBDD: $F = \overline{d} \cdot \overline{c} + d \cdot \overline{a} \cdot b + d \cdot a \cdot b$

pri 0-sup-BDD: $F = \overline{d} \cdot \overline{c} \cdot \overline{b} \cdot \overline{a} + d \cdot \overline{c} \cdot \overline{a} \cdot b + d \cdot \overline{c} \cdot a \cdot b$

pri ROFDD: $F = 1 \oplus d \cdot b \oplus d \cdot a \cdot b$

Shannonovo razčlenitveno pravilo (pri ROBDD):

$$F = \overline{v} \cdot E + v \cdot T$$

v - spremenljivka v korenu grafa
E - logična funkcija, ki jo predstavlja podgraf 'else'
T - logična funkcija, ki jo predstavlja podgraf 'then'
Za graf na sliki dobimo:

$$F = \overline{d} \cdot E_1 + d \cdot T_1$$

$$E_1 = \overline{c} \cdot 1 + c \cdot 0$$

$$T_1 = \overline{a} \cdot E_2 + a \cdot T_2$$

$$E_2 = \overline{b} \cdot 0 + b \cdot 1$$

$$T_2 = \overline{b} \cdot 0 + b \cdot 1$$

$$F = \overline{d} \cdot \overline{c} + d \cdot (\overline{a} \cdot b + a \cdot b) =$$

$$= \overline{d} \cdot \overline{c} + d \cdot \overline{a} \cdot b + d \cdot a \cdot b$$

OPIS TESTIRANJA

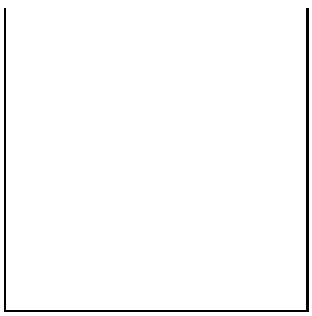
- logične funkcije
- množice kombinacij
- črnobele slike

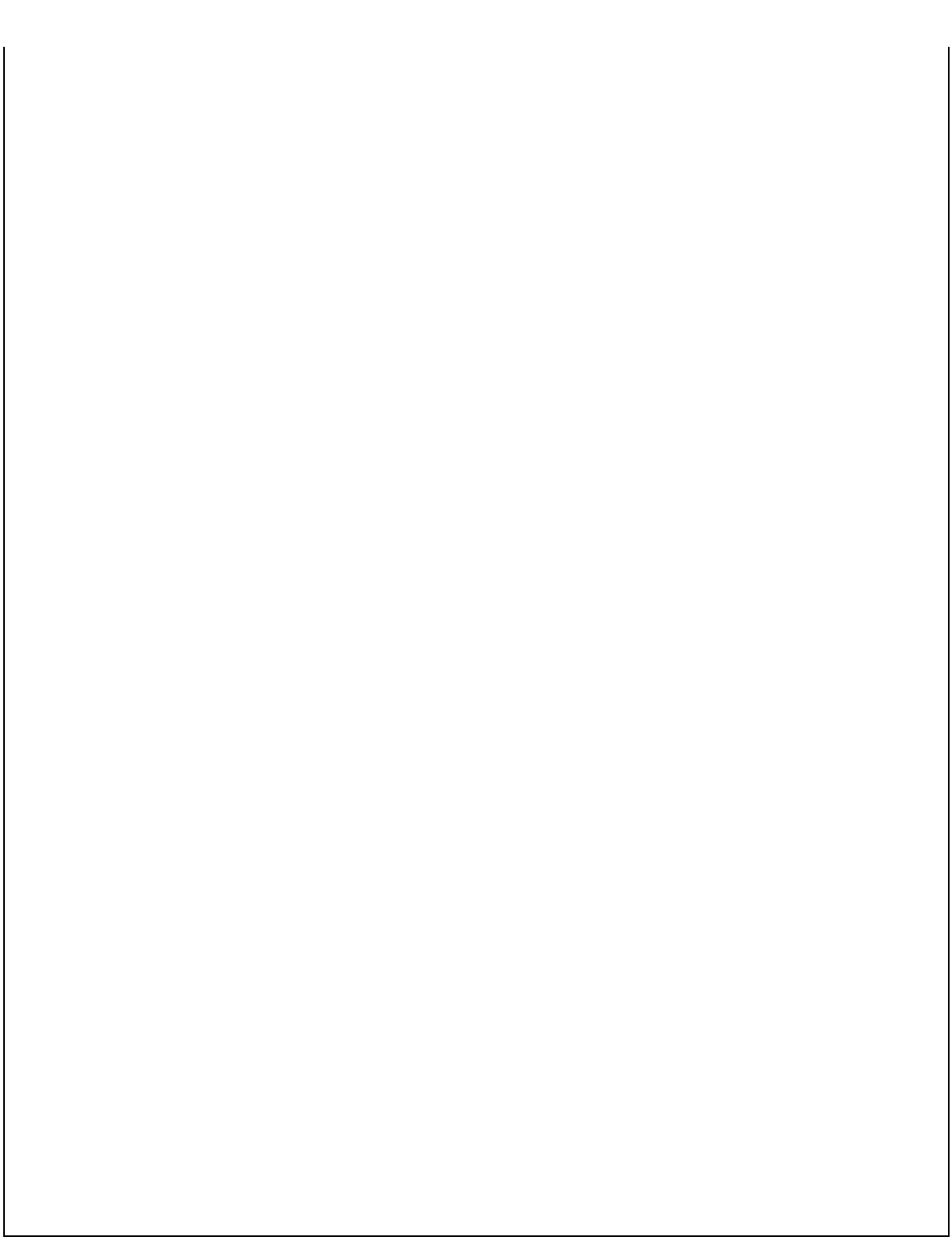
PREVERJANJE EKVIVALENCE LOGIČNIH VEZIJ

```
@BE1
@invar (A1 B1 A2 B2)
@sub
CAR1 = (AND A1 B1)
@out
SOM1 = (EXOR A1 B1)
SOM2 = (EXOR A2 B2 CAR1)
COUT = (OR (AND (OR A2 B2) CAR1) (AND A2 B2))
@end

@BE2
@invar (A1 B1 A2 B2)
@sub
COUT1 = (AND B1 A1)
@out
SOM1 = (NOT (OR (AND (NOT A1) (NOT B1)) (AND A1 B1)))
SOM2 = (NOT (OR (AND (OR (AND (NOT A2) (NOT B2)) (AND A2 B2)) (NOT COUT1)
       (AND COUT1 (NOT (OR (AND (NOT A2) (NOT B2)) (AND A2 B2))))))
COUT = (OR (AND A2 COUT1) (AND B2 COUT1) (AND A2 B2))
@end
```

PREDSTAVITEV SLIK





EVROPA - 764×949
MEDVED - 560×880
SKAVT - 1232×1520
SRCE - 255×255

STROJNA IN PROGRAMSKA OPREMA:

- delovna postaja HP 712/80,
- 32 MB fizičnega pomnilnika, največ 75 MB navideznega pomnilnika,
- operacijski sistem HP-UX 9.05, prevajalnik gcc 2.6.3.

PREVERJANJE EKVIVALENCE LOGIČNIH VEZIJ

Ime	ROBDD		ROFDD		0-sup-BDD	
datoteke	vozlišč	čas	vozlišč	čas	vozlišč	čas
add3	665	0.47	272	1.26	1785	0.66
add4	933	0.77	281	3.36	3060	1.03
mul06	1156	0.91	1274	11.22	1302	1.36
mul07	3256	2.28	-	-	3708	3.78
alupla21	1497	0.57	1269	1.33	1733	0.61
alupla22	7589	1.56	2413	3.28	8173	1.63
x1dn	425	0.37	800	0.47	660	0.41
x6dn	341	0.47	405	0.77	1079	0.57
z9sym	25	0.42	27	0.99	31	0.42
in1	653	1.81	1525	4.44	628	1.68
pitch	237	0.70	227	0.72	338	0.65

PREDSTAVITEV SLIK

slika	ROBDD	ROFDD	0-sup-BDD	2-drevo
EVROPA	8502	-	8189	81203
MEDVED	10737	-	10556	131961
SKAVT	21296	-	20654	284377
SRCE	784	4856	770	5197

Značilnosti izvedbe:

- algoritmi so rekurzivni,
- uporablja označene povezave,
- ima vgrajeno avtomatsko čiščenje,
- uporablja dinamično urejanje spremenljivk.

Zaključek

- Obdelava logičnih funkcij z odločitvenimi grafi je v glavnem že nadomestila vse ostale metode.
- Realizacija odločitvenih grafov je dokaj kompleksen problem, vendar jih lahko potem, ko enkrat napišeš osnovne programske module, enostavno uporabljaš kot vsako drugo podatkovno strukturo.
- Uporaba: vsepovsod, kjer lahko problem opišemo z operacijami nad manjšimi, končnimi množicami elementov (testiranje ekvivalence kombinacijskih in sekvenčnih vezij, avtomatsko dokazovanje izrekov, avtomatično generiranje testnih vzorcev, analiza končnih avtomatov,...).

Cilji nadaljnjih raziskav:

- izboljšati učinkovitost izvedbe obstoječih odločitvenih grafov,
- poiskati nove vrste odločitvenih grafov, s katerimi bolje rešimo vse ali pa samo določene probleme,
- teoretično raziskati prostorsko in časovno zahtevnost ter druge lastnosti posameznih vrst odločitvenih grafov.