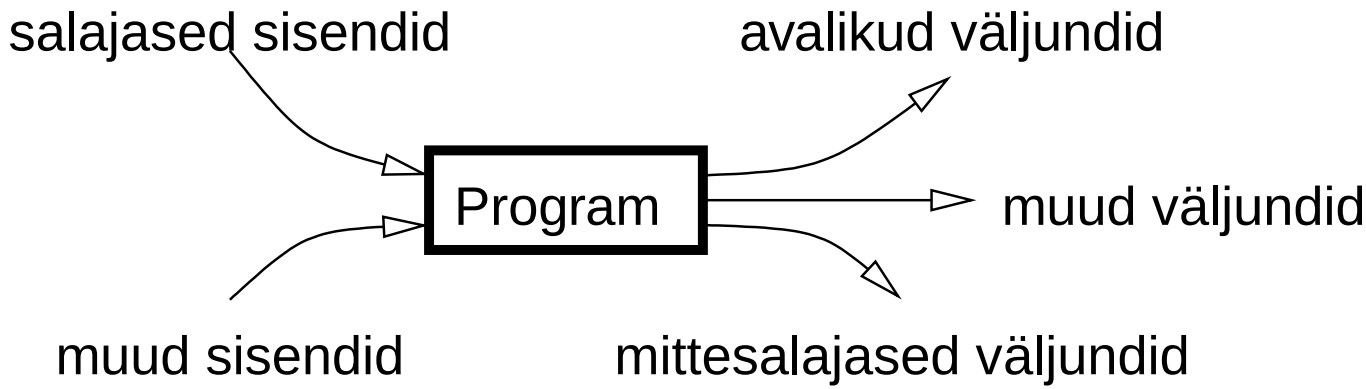
Suhteline salajasus

Peeter Laud

peeter_l@ut.ee

Tartu Ülikool

Probleemi olemus



Kas programm rahuldab järgnevat salajasustingimust?

- Avalikud väljundid tehakse kõigile teatavaks,. . .
- aga salajaste sisendite kohta ei tohi midagi avalikuks saada,. . .
- aga me oleme mingil viisil kindlaks teinud, et mittesalajased väljundid ei anna salajaste sisendite kohta sensitiivset infot.

Formaalne definitsioon? Programmianalüüs?

Ettekande ülesehitus

- Keele süntaks ja semantika.
- Turvalisuse definitsioon.
- Programmianalüüs.
  - Abstraktne domeen.
  - Üleminekufunktsioonid.
- $declassify$-lause.
  - Lihtne, semantiliselt ebakorrektne analüüs.
  - Programmi ümberkirjutamine.
  - Kahe analüüsi suhe.

Programmeerimiskeel

Lihtne imperatiivne keel (WHILE).

$$
\begin{array}{rcl}
\mathsf{P} & ::= & \mathtt{x} := o(\mathtt{x}_1, \ldots, \mathtt{x}_k) \\
& | & \mathit{skip} \\
& | & \mathsf{P}_1; \mathsf{P}_2 \\
& | & \mathit{if}\ \mathtt{b}\ \mathit{then}\ \mathsf{P}_1\ \mathit{else}\ \mathsf{P}_2 \\
& | & \mathit{while}\ \mathtt{b}\ \mathit{do}\ \mathsf{P}'
\end{array}
$$

Programmi olekute hulk $\mathbf{State}$ on $\mathbf{Var} \to \mathbf{Val}$.
$\mathtt{x}, \mathtt{x}_1, \ldots, \mathtt{x}_k, \mathtt{b} \in \mathbf{Var}$, $o \in \mathbf{Op}$.

- Salajased sisendid — muutujate $\mathbf{Var}_\mathrm{S} \subseteq \mathbf{Var}$ esialgsed väärtused.
- Avalikud/mittesalajased väljundid — muutujate $\mathbf{Var}_\mathrm{P} \subseteq \mathbf{Var}$ / $\mathbf{Var}_\mathrm{NS} \subseteq \mathbf{Var}$ lõppväärtused.

Deterministlik semantika

- Denotatsiooniline semantika kujutab programmi algoleku lõppolekuks.

$$[\![\mathrm{P}]\!] : \mathbf{State} \to \mathbf{State}_\perp$$

  - defineeritud induktiivselt üle süntaksi;
  - $\mathbf{State}_\perp = \mathbf{State} \dot\cup \{\perp\}$;
  - $\perp$ tähistab mittetermineerumist.
- Meie tulemused on hetkel termineerumistundetud.
  - Termineerumisega seotud küsimused on ilmselt teistest küsimustest sõltumatud.
  - Seetõttu loeme me $[\![\mathrm{P}]\!] : \mathbf{State} \to \mathbf{State}$.

Mittemõjutatavus

Harilik definitsioon:
Programmi avalike väljundite väärtused peavad olema määratud tema „teiste” sisendite väärtustega.

$$\exists f : (\mathbf{Var}_\mathrm{T} \to \mathbf{Val}) \to (\mathbf{Var}_\mathrm{P} \to \mathbf{Val})$$

nii et kõigi $S \in \mathbf{State}$ jaoks

$$[\![\mathrm{P}]\!](S)|_{\mathbf{Var}_\mathrm{P}} = f(S|_{\mathbf{Var}_\mathrm{T}}) \ .$$

Siin $\mathbf{Var}_\mathrm{T}$ tähistab programmi mittesalajasi sisendeid.

Mittemõjutatavus

Tähistagu $S_1 =_X S_2$ võrdust $S_1|_X = S_2|_X$, kus $X \subseteq \mathbf{Var}$.

Siis mittemõjutatavus tähendab, et kõigi $S_1, S_2 \in \mathbf{State}$ jaoks

$$S_1 =_{\mathbf{Var_T}} S_2 \Longrightarrow [\![P]\!](S_1) =_{\mathbf{Var_P}} [\![P]\!](S_2) \ .$$

Suhteline salajasus

Programmi avalike väljundite väärtused peavad olema määratud tema „teiste” sisendite ja mittesalajaste väljundite väärtustega.

$$S_1 =_{\mathbf{Var}_\mathrm{T}} S_2 \wedge [\![\mathsf{P}]\!](S_1) =_{\mathbf{Var}_\mathrm{NS}} [\![\mathsf{P}]\!](S_2) \Rightarrow [\![\mathsf{P}]\!](S_1) =_{\mathbf{Var}_\mathrm{P}} [\![\mathsf{P}]\!](S_2)$$

Kui eeldame (üldisust kitsendamata), et P ei omista muutujatele $\mathbf{Var}_\mathrm{T}$-s, siis

$$[\![\mathsf{P}]\!](S_1) =_{\mathbf{Var}_\mathrm{T}} [\![\mathsf{P}]\!](S_2) \wedge [\![\mathsf{P}]\!](S_1) =_{\mathbf{Var}_\mathrm{NS}} [\![\mathsf{P}]\!](S_2) \Rightarrow [\![\mathsf{P}]\!](S_1) =_{\mathbf{Var}_\mathrm{P}} [\![\mathsf{P}]\!](S_2)$$

Abstraktne domeen

Mingite (suvaliste) $\mathcal{S} \subseteq \mathbf{State}$ ja $X, Y, Z \subseteq \mathbf{Var}$ korral huvitab meid, kas

$$S_1 =_X S_2 \wedge S_1 =_Y S_2 \Rightarrow S_1 =_Z S_2$$

kehtib kõigi $S_1, S_2 \in \mathcal{S}$ jaoks.

Piisab, kui teame kõigi $X \subseteq \mathbf{Var}$ ja $\mathbf{z} \in \mathbf{Var}$ jaoks, kas

$$S_1 =_X S_2 \Rightarrow S_1(\mathbf{z}) = S_2(\mathbf{z})$$

kehtib kõigi $S_1, S_2 \in \mathcal{S}$ jaoks.

Me abstraheerime $\mathcal{P}(\mathbf{State})$-i domeeniga $\mathcal{P}(\mathcal{P}(\mathbf{Var}) \times \mathbf{Var})$. Olgu α abstraktsioonifunktsioon.

Analüüsi üldstruktuur

- Let $A_\circ$ võimalike algolekute hulga abstraktsioon.
- Rakenda P analüüsi $A_\circ$-le, see annab $A_\bullet$-i.
  - $A_\bullet$ abstraheerib võimalike lõppolekute hulka.
  - Ta on *konservatiivne* abstraktsioon — osad paarid $(X, \mathtt{z})$ võivad puududa.
- Kui $\left(\mathbf{Var}_\mathrm{T} \cup \mathbf{Var}_\mathrm{NS}, \mathtt{x}\right) \in A_\bullet$ kõigi $\mathtt{x} \in \mathbf{Var}_\mathrm{P}$ jaoks, siis võime programmi turvaliseks lugeda.

Abstraktsiooni omadusi

- Olgu $A = \alpha(\mathcal{S})$ mingi $\mathcal{S} \in \mathbf{State}$ jaoks. Siis
  - $(\{\mathtt{z}\}, \mathtt{z}) \in A$,
  - $(X, \mathtt{z}) \in A \Rightarrow (X \cup Y, \mathtt{z}) \in A$,
  - $(X \cup \{\mathtt{y}\}, \mathtt{z}) \in A \wedge (X, \mathtt{y}) \in A \Rightarrow (X, \mathtt{z}) \in A$

  kehtivad kõigi $X, Y \subseteq \mathbf{Var}$ ja $\mathtt{y}, \mathtt{z} \in \mathbf{Var}$ jaoks.
- Ütleme, et $A \subseteq \mathcal{P}(\mathbf{Var}) \times \mathbf{Var}$ on kinnine, kui ta neid implikatsioone rahuldab.
- A sulund on vähim kinnine A-d sisaldav hulk.

Omistamiste analüüs

- Analüüs $\mathcal{A}(\mathtt{x} := o(\mathtt{x_1}, \ldots, \mathtt{x_k}))$, rakendatuna $A_\circ$-le, konstrueerib $A_\bullet$-i järgmisel viisil:
  - tapa $\mathtt{x}$, s.t. kustuta kõik $(X, \mathtt{z})$, kus $\mathtt{x} \in X$ või $\mathtt{x} = \mathtt{z}$;
  - lisa $(\{\mathtt{x_1}, \ldots, \mathtt{x_k}\}, \mathtt{x})$;
  - moodusta sulund.

  (eeldame, et $\mathtt{x} \notin \{\mathtt{x_1}, \ldots, \mathtt{x_k}\}$)
- Kui mõni $\mathtt{x_i}$ on peale tehet leitav mingist hulgast $X \subseteq \{\mathtt{x}, \mathtt{x_1}, \ldots, \mathtt{x_k}\}$ siis lisa teisel sammul ka $(X, \mathtt{x_i})$.
  - Näide: $\mathtt{y}$ on leitav hulgast $\{\mathtt{x}, \mathtt{z}\}$ peale tehet $\mathtt{x} := \mathtt{y} + \mathtt{z}$.

$skip$-i ja kompositsiooni analüüs

- $\mathcal{A}(skip)$ on identsusfunktsioon.
- $\mathcal{A}(\mathsf{P}_1;\mathsf{P}_2) = \mathcal{A}(\mathsf{P}_2) \circ \mathcal{A}(\mathsf{P}_1)$.

$if - then - else$-i analüüs

Vaatame programmi if $\mathtt{b}$ $then$ P_1 $else$ P_2.

- Olgu $\{\mathtt{x_1}, \ldots, \mathtt{x_k}\} = \mathbf{Var}_{\mathrm{asgn}} \subseteq \mathbf{Var}$ muutujad, millele P_1-s või P_2-s omistatakse.
- Olgu $\mathbf{Var}' = \mathbf{Var} \dot\cup \{N, \mathtt{x}_1^{\mathsf{true}}, \ldots, \mathtt{x}_k^{\mathsf{true}}, \mathtt{x}_1^{\mathsf{false}}, \ldots, \mathtt{x}_k^{\mathsf{false}}\}$
- Programmil $\rightarrow$ on sama funktsionaalsus.
- $\mathsf{P}_1^{\mathsf{true}}$ on P_1, kus kõik $\mathtt{x_i}$-d on asendatud $\mathtt{x}_\mathtt{i}^{\mathsf{true}}$-ga.
- $\mathsf{P}_2^{\mathsf{false}}$ on defineeritud sarnaselt.

$$
\begin{array}{l}
\mathtt{N} := \mathtt{b} \\
\mathtt{x}_1^{\mathsf{true}} := \mathtt{x}_1 \\
\mathtt{x}_1^{\mathsf{false}} := \mathtt{x}_1 \\
\ldots \\
\mathtt{x}_k^{\mathsf{true}} := \mathtt{x}_k \\
\mathtt{x}_k^{\mathsf{false}} := \mathtt{x}_k \\
\mathsf{P}_1^{\mathsf{true}} \\
\mathsf{P}_2^{\mathsf{false}} \\
\mathtt{x}_1 := \mathtt{N} \,?\, \mathtt{x}_1^{\mathsf{true}} : \mathtt{x}_1^{\mathsf{false}} \\
\ldots \\
\mathtt{x}_k := \mathtt{N} \,?\, \mathtt{x}_k^{\mathsf{true}} : \mathtt{x}_k^{\mathsf{false}}
\end{array}
$$

Analüüsi hoopis parempoolset programmi.

Lisaks: $?$:-de jada analüüsimine

Vaatame programmi

$$\mathtt{x_1} := \mathtt{N} \,?\, \mathtt{x_1}^{\mathsf{true}} : \mathtt{x_1}^{\mathsf{false}}; \mathtt{x_2} := \mathtt{N} \,?\, \mathtt{x_2}^{\mathsf{true}} : \mathtt{x_2}^{\mathsf{false}}; \cdots ; \mathtt{x_k} := \mathtt{N} \,?\, \mathtt{x_k}^{\mathsf{true}} : \mathtt{x_k}^{\mathsf{false}}$$

Kui

$$X \subseteq \mathbf{Var} \backslash \{\mathtt{x_1}, \ldots, \mathtt{x_k}\} \qquad (X, \mathtt{N}) \in A_\circ$$

$$Y \subseteq \{\mathtt{x_1}, \ldots, \mathtt{x_k}\} \qquad (X \cup Y^{\mathsf{true}}, \mathtt{x_i}^{\mathsf{true}}) \in A_\circ$$

$$\mathtt{i} \in \{\mathtt{1}, \ldots, \mathtt{k}\} \qquad (X \cup Y^{\mathsf{false}}, \mathtt{x_i}^{\mathsf{false}}) \in A_\circ$$

siis võime võtta $(X \cup Y, \mathtt{x_i}) \in A_\bullet$.

Harude sõltumatu analüüs

Programmi $\mathit{if}\ \mathtt{b}\ \mathit{then}\ \mathsf{P}_1\ \mathit{else}\ \mathsf{P}_2$ võime analüüsida järgmiselt: Olgu $\mathtt{N}$ uus muutuja, olgu

$$A_1 = \mathcal{A}(\mathtt{N} := \mathtt{b}; \mathsf{P}_1)(A_\circ)$$
$$A_2 = \mathcal{A}(\mathtt{N} := \mathtt{b}; \mathsf{P}_2)(A_\circ) \quad .$$

Siis võtame $(X, \mathtt{z}) \in A_\bullet$, kui

- $(X, \mathtt{z}) \in A_1 \cap A_2$;
- kui $X \cup \{\mathtt{z}\} \not\perp \mathbf{Var}_{\mathrm{asgn}}$, siis $(X \backslash \mathbf{Var}_{\mathrm{asgn}}, \mathtt{N}) \in A_1$ (või $\in A_2$).

$while$-i analüüs

$\mathcal{A}(while\ \mathtt{b}\ do\ \mathsf{P})$, rakendatuna $A_\circ$-le, rakendab talle analüüsi $\mathcal{A}(if\ \mathtt{b}\ then\ \mathsf{P}\ else\ skip)$ seni kaua, kuni jõuab püsipunktini.

Korrektsus järeldub võrdusest

$$[\![while\ \mathtt{b}\ do\ \mathsf{P}]\!] = [\![while\ \mathtt{b}\ do\ \mathsf{P}]\!] \circ [\![if\ \mathtt{b}\ then\ \mathsf{P}\ else\ skip]\!] \ .$$

Alternatiiv

Programmi $while\ \mathtt{b}\ do\ \mathsf{P}$ analüüsiks:
Olgu A_1 suurim (hulkade sisalduvuse mõttes), mis rahuldab võrrandit

$$A_1 = A_\circ \cap \mathcal{A}(\mathtt{N} := \mathtt{b}; \mathsf{P})(A_1)$$

(leitakse iteratiivselt, algväärtuseks võime võtta $A_\circ$).
Siis võtame $(X, \mathtt{z}) \in A_\bullet$, kui

- $(X, \mathtt{z}) \in A_1$;
- kui $X \cup \{\mathtt{z}\} \not\perp \mathbf{Var}_{\mathrm{asgn}}$, siis $(X \backslash \mathbf{Var}_{\mathrm{asgn}}, \mathtt{N}) \in A_1$.

Deklassifitseerimislause

- Me lisame programmeerimiskeelde lause

$$declassify(\mathtt{x}),$$

  kus $\mathtt{x} \in \mathbf{Var}$.
- Tema semantika on sama, mis lausel $skip$.
- Intuitiivne tähendus: praegusel hetkel ei ole võimalik $\mathtt{x}$ väärtusest midagi salajaste sisendite kohta leida.
- Intuitiivne tähendus kajastub analüüsis.

Lihtne analüüs deklassifitseerimisega

- Vaatame analüüsi, mis kujutab programmi alguses mittesalajsed muutujad lõpus avalikeks muutujateks.
- $\mathcal{B}(\mathsf{P}) : \mathcal{P}(\mathbf{Var}) \to \mathcal{P}(\mathbf{Var})$, nii argument kui ka väärtus on avalike muutujate hulgad.

$$\mathcal{B}(\mathtt{x} := o(\mathtt{x_1}, \ldots, \mathtt{x_k}))(B_\circ) = \begin{cases} B_\circ \cup \{\mathtt{x}\}, & \text{if } \mathtt{x_1}, \ldots, \mathtt{x_k} \in B_\circ \\ B_\circ \backslash \{\mathtt{x}\}, & \text{otherwise.} \end{cases}$$

$$\mathcal{B}(\mathit{declassify}(\mathtt{x}))(B_\circ) = B_\circ \cup \{\mathtt{x}\}$$

skip ja kompositsioon — nagu $\mathcal{A}$.

Lihtne analüüs, $if - then - else$

$B_\circ = \mathcal{B}(if\ \mathtt{b}\ then\ \mathsf{P}_1\ else\ \mathsf{P}_2)(B_\circ)$ leitakse järgmiselt:

$$B_1 = \mathcal{B}(\mathtt{N} := \mathtt{b}; \mathsf{P}_1)(B_\circ)$$
$$B_2 = \mathcal{B}(\mathtt{N} := \mathtt{b}; \mathsf{P}_2)(B_\circ),$$

kus $\mathtt{N}$ on uus muutuja.
$\mathtt{z} \in B_\bullet$, kui

- $\mathtt{z} \in B_1 \cap B_2$;
- kui $\mathtt{z} \in \mathbf{Var}_{\mathrm{asgn}}$, siis $\mathtt{N} \in B_1$ (ehk $\mathtt{N} \in B_2$).

Lihtne analüüs, $while$

Programmi $while$ $\mathtt{b}$ do P analüüsiks:
Olgu B_1 suurim (hulkade sisalduvuse mõttes), mis rahuldab võrrandit

$$B_1 = B_\circ \cap \mathcal{B}(\mathtt{N} := \mathtt{b}; \mathsf{P})(B_1)$$

(leitakse iteratiivselt).
Siis $\mathtt{z} \in B_\bullet$ parajasti siis, kui

- $\mathtt{z} \in B_1$;
- kui $\mathtt{z} \in \mathbf{Var}_{\mathrm{asgn}}$, siis $\mathtt{N} \in B_1$.

Analüüside seostamine

- Olgu $B \subseteq \mathbf{Var}$. Etteantud $\mathbf{Var}_{\mathrm{T}}$ ja $\mathbf{Var}_{\mathrm{NS}}$ jaoks olgu

$$\xi(B) := \{ \left(\mathbf{Var}_{\mathrm{T}} \cup \mathbf{Var}_{\mathrm{NS}}, \mathtt{x}\right) : \mathtt{x} \in B \} \ .$$

  - ξ seob $\mathcal{B}$ ja $\mathcal{A}$ määramis-/muutumispiirkonnad.
  - B — analüüsi $\mathcal{B}$ leitud salajast infot mittesisaldavad muutujad.
- Me tahame defineerida programmiteisenduse $\overline{\cdot}$ ja hulga $\mathbf{Var}_{\mathrm{NS}}$ nii, et kõigi programmide P ja $B_\circ \subseteq \mathbf{Var}$ jaoks kehtiks

$$\xi\big(\mathcal{B}(\mathsf{P})(B_\circ)\big) \subseteq \mathcal{A}(\overline{\mathsf{P}})(\xi(B_\circ)) \ .$$

  - S.t. $\mathcal{A}$ oleks vähemalt sama täpne kui $\mathcal{B}$.
  - $\mathbf{Var}_{\mathrm{NS}}$-i kuuluvate muutujate väärtused iseloomustavad, kui palju $declassify$-d infot annavad.

Programmiteisendus (1/3)

Olgu $\mathtt{d}$ uus muutuja. Siis

$$\overline{\mathsf{P}} := \Big[\mathtt{d} := \mathsf{Nil}; \mathcal{T}(\mathsf{P})\Big]$$

ja $\mathbf{Var}_{\mathrm{NS}} = \{\mathtt{d}\}$. $\mathcal{T}$ on defineeritud järgmiselt:

- $\mathcal{T}(\mathsf{P}) = \mathsf{P}$, kui P-s pole deklassifitseerimislauseid;
- $\mathcal{T}(\mathit{declassify}(\mathtt{x})) := \big[\mathtt{tmp} := \mathtt{d}; \mathtt{d} := (\mathtt{x}, \mathtt{tmp})\big]$, kus $\mathtt{tmp}$ on uus muutuja;
  - Analüüsi $\mathcal{A}$ juures: nii $\mathtt{y}$ kui ka $\mathtt{z}$ on $\mathtt{x}$-st peale tehet $\mathtt{x} := (\mathtt{y}, \mathtt{z})$ leitavad.
- $\mathcal{T}(\mathsf{P}_1; \mathsf{P}_2) = \mathcal{T}(\mathsf{P}_1); \mathcal{T}(\mathsf{P}_2)$;

Programmiteisendus (2/3)

Olgu if- ja $while$-laused märgendatud. Samas programmis ei esine sama märgend kaks korda.

$$\mathcal{T}(if^l\ \mathtt{b}\ then\ \mathsf{P}_1\ else\ \mathsf{P}_2) :=$$
$$if\ \mathtt{b}\ then\ \mathtt{tmp} := \mathtt{d}; \mathtt{d} := (\star_l, \mathtt{tmp}); \mathcal{T}(\mathsf{P}_1)$$
$$else\ \mathtt{tmp} := \mathtt{d}; \mathtt{d} := (\star_l, \mathtt{tmp}); \mathcal{T}(\mathsf{P}_2),$$

kus

- l on antud if-lause märgend.
- $\mathtt{tmp}$ on uus muutuja;
- $\star_l \in \mathbf{Val}$ on väärtus, mis kuskil mujal ei esine.

Programmiteisendus (3/3)

$$\mathcal{T}(\mathit{while}^{l}\ \mathtt{b}\ \mathit{do}\ \mathsf{P}) :=$$

$$\mathtt{tmp} := \mathtt{d}; \mathtt{d} := (\star_l^{\mathrm{a}}, \mathtt{tmp}); \mathit{while}\ \mathtt{b}\ \mathit{do}\ \left[\mathcal{T}(\mathsf{P}); \mathtt{tmp} := \mathtt{d}; \mathtt{d} := (\star_l^{\mathrm{i}}, \mathtt{tmp})\right],$$

kus

- l on antud while-lause märgend.
- $\mathtt{tmp}$ on uus muutuja;
- $\star_l^{\mathrm{a}}, \star_l^{\mathrm{i}} \in \mathbf{Val}$ on väärtused, mis kuskil mujal ei esine.
  - $\star_l^{\mathrm{a}}$ — „algus”;
  - $\star_l^{\mathrm{i}}$ — „iteratsioon”.

Analüüs $\mathcal{A}$ kohendamine

if- ja $while$-lausete analüüse on võimalik optimistlikumaks muuta, sest muutuja $\mathtt{d}$ lõppväärtusest on tänu $\star$-le võimalik leida tema algväärtus ja väärtus peale iga iteratsiooni.

Võime lugeda, et $d \notin \mathbf{Var}_{\mathrm{asgn}}$, kuigi talle if-lause harudes või tsüklis omistatakse.
S.t. tingimuse

$$(X, \mathtt{z}) \not\perp \mathbf{Var}_{\mathrm{asgn}} \Longrightarrow (X \backslash \mathbf{Var}_{\mathrm{asgn}}, \mathtt{N}) \in A_1$$

võib asendada tingimusega

$$(X, \mathtt{z}) \not\perp \mathbf{Var}_{\mathrm{asgn}} \backslash \{d\} \Longrightarrow (X \backslash (\mathbf{Var}_{\mathrm{asgn}} \backslash \{\mathtt{d}\}), \mathtt{N}) \in A_1 \quad .$$

Seda läheb $\mathcal{A}$ ja $\mathcal{B}$ seostamisel vaja. . .