Absztrakció a szoftvertervezésben az Alloy specifikációs nyelv segítségével

Németh L. Zoltán

Számítástudomány Alapjai Tanszék
SZTE, Informatikai Tanszékcsoport

2009. szeptember 15.

Tartalom

- Röviden a formális módszerekről
- Az absztrakcióról és az Alloyról
- Alloy egy példán keresztül (e-mail címjegyzék)
- Alloy nyelv elemei (logika, nyelv, analízis)
- Az eszköz értékelése

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak.
Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

Röviden a formális módszerekről

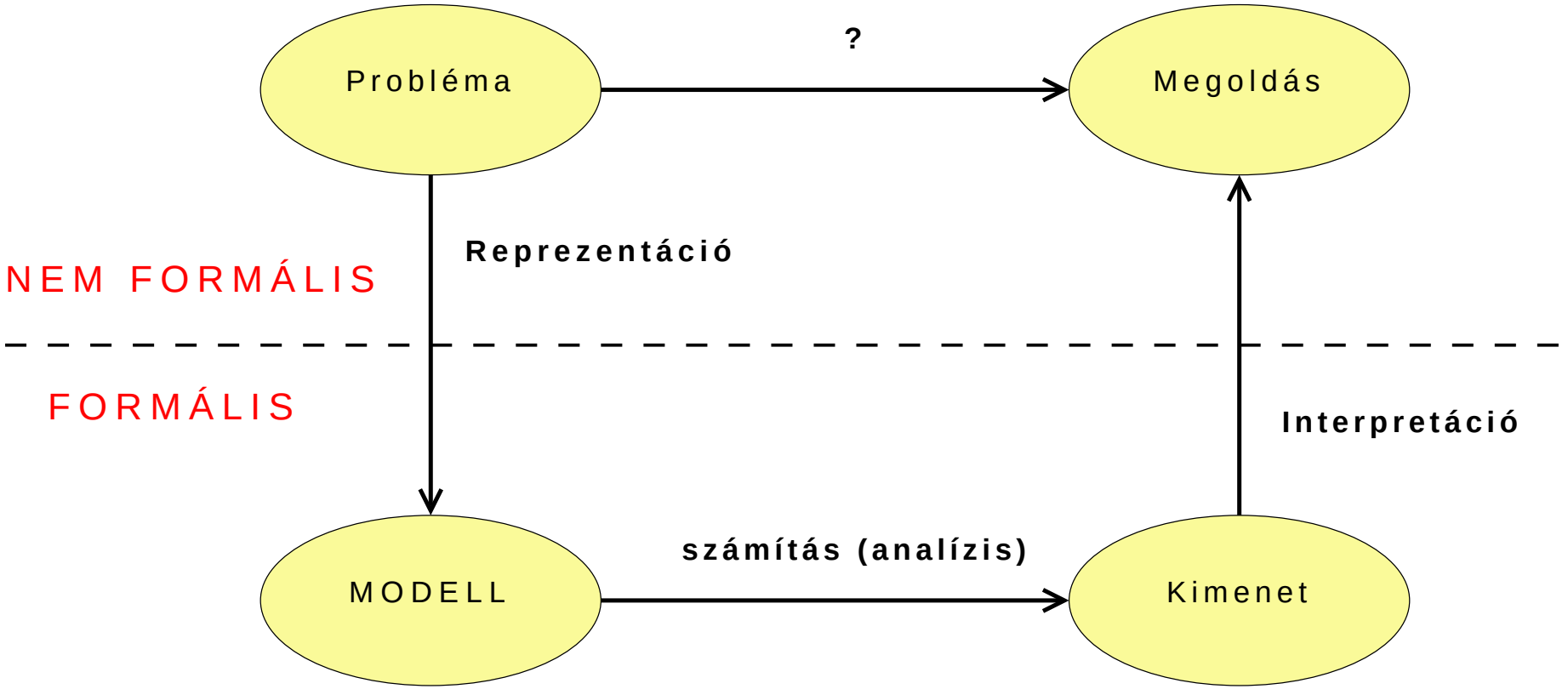
Meghatározás [3]

Formális módszerek alatt olyan matematikai elméleten alapuló technikákat értünk, melyeket hardver és szoftver rendszerek specifikációjára, fejlesztésére és ellenőrzésére használnak. Mára kiterjesztették mérnöki, biológiai, ipari, szociális, orvosi stb. rendszerekre is.

Mennyire formális?

- Természetes nyelvi megfogalmazás.
- Matematikai bizonyítás.
- Grafikus megjelenítési módszerek, diagrammok.
- Programozási nyelvek.
- Specifikációs nyelvek.

A problémamegoldás egy általános modellje



Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

Formális módszerek

Napjaink információs rendszerei

- egyre bonyolultabbak és
- egyre inkább függünk tőlük

az elmúlt pár évtizedben

- a számítási kapacitás jelentősen nőtt
- maguk a módszerek is jelentősen fejlődtek

$\Rightarrow$ **a formális módszerek térhódítása, számos sikeres alkalmazás**

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány alkalmazási területe

Igény szerint:

- biztonságkritikus alkalmazások
  - orvosi rendszerek
  - közlekedés irányítás
  - személy-, vagyon- és objektumvédelem
  - információ biztonsági rendszerek, banki rendszerek
  - katonai rendszerek
- missziókritikus alkalmazások
  - pl. űrkutatás, rendeléskezelés egy webáruháznak
- "költség kritikus" alkalmazások
  - hardver tervezés
  - termelésirányítás, stb.

Alkalmazhatóság szerint (ahol a hibák felderítése teszteléssel nehéz):

- osztott és konkurrens rendszerek
- protokoll tervezés
- beágyazott rendszerek

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

A formális módszerek néhány csoportja

- Automatikus tételbizonyítás
- Programhelyesség bizonyítás (Floyd–Hoare logika)
- Programozási nyelvek formális szemantikája
- Osztott és konkurrens rendszerek modelljei (Petri-hálók, processzus algebrák: CCS, CSP, Π-kalkulus, stb.)
- Modellellenőrzés
- Absztrakciós és finomítási technikák
- Specifikációs nyelvek (ASM, B, Z, VDM, Alloy, stb.)

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkalmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Könnyűsúlyú formális módszerek

Akkor miért nem vált általános gyakorlattá a szoftverfejlesztésben a formális módszerek alkakmazása?

- a specifikációk teljes formalizálása és ellenőrzése igen jelentős költséget jelent,
- jelentős **matematikai** és **rendszer specifikus** ismeretek kellenek a használatukhoz
  - az analízis nem mindig automatikus
  - még, ha az analízis automatikus is, a modellezés körülményes lehet
- a fejlesztőt segítő eszközök sem mindig a legkézenfekvőbbek
- „a szokás hatalma" a szoftveriparban: minél gyorsabb fejlesztés + utólagos karbantartás

$\Rightarrow$ **megjelentek a könnyűsúlyú (lightweight) formális módszerek, melyek a fenti korlátokat igyekeznek leküzdeni**

Az absztrakcióról

- alapvető a szoftverfejlesztésben
- a helyesen tervezett absztrakció esetén
  - a programozás a tervekből természetesen folyik
  - átlátható szerkezetet kapunk
  - kis és egyszerű interfészekkel
  - új funkcionalitások könnyen beilleszthetők
- helytelen absztrakció esetén
  - a kódolás kellemetlen meglepetéseket tartogat
  - bonyolult szerkezet eredményez
  - melyen kis változtatásokat is nehéz végbevinni
  - javítgatása helyett érdemesebb elölről kezdeni az egészet

A tervező könnyen a "wishful thinking" (vágyálom) csapdájába eshet, azaz azt gondolja, hogy az általa kitalált absztrakció egyszerű és robosztus és csak az implementáció során jön rá, hogy az valójában zavaros és inkonzisztens.

Az absztrakcióról

- alapvető a szoftverfejlesztésben
- a helyesen tervezett absztrakció esetén
  - a programozás a tervekből természetesen folyik
  - átlátható szerkezetet kapunk
  - kis és egyszerű interfészekkel
  - új funkcionalitások könnyen beilleszthetők
- helytelen absztrakció esetén
  - a kódolás kellemetlen meglepetéseket tartogat
  - bonyolult szerkezet eredményez
  - melyen kis változtatásokat is nehéz végbevinni
  - javítgatása helyett érdemesebb elölről kezdeni az egészet

A tervező könnyen a "wishful thinking" (vágyálom) csapdájába eshet, azaz azt gondolja, hogy az általa kitalált absztrakció egyszerű és robosztus és csak az implementáció során jön rá, hogy az valójában zavaros és inkonzisztens.

Az absztrakcióról

- alapvető a szoftverfejlesztésben
- a helyesen tervezett absztrakció esetén
  - a programozás a tervekből természetesen folyik
  - átlátható szerkezetet kapunk
  - kis és egyszerű interfészekkel
  - új funkcionalitások könnyen beilleszthetők
- helytelen absztrakció esetén
  - a kódolás kellemetlen meglepetéseket tartogat
  - bonyolult szerkezet eredményez
  - melyen kis változtatásokat is nehéz végbevinni
  - javítgatása helyett érdemesebb elölről kezdeni az egészet

A tervező könnyen a "wishful thinking" (vágyálom) csapdájába eshet, azaz azt gondolja, hogy az általa kitalált absztrakció egyszerű és robosztus és csak az implementáció során jön rá, hogy az valójában zavaros és inkonzisztens.

Az absztrakcióról

- alapvető a szoftverfejlesztésben
- a helyesen tervezett absztrakció esetén
  - a programozás a tervekből természetesen folyik
  - átlátható szerkezetet kapunk
  - kis és egyszerű interfészekkel
  - új funkcionalitások könnyen beilleszthetők
- helytelen absztrakció esetén
  - a kódolás kellemetlen meglepetéseket tartogat
  - bonyolult szerkezet eredményez
  - melyen kis változtatásokat is nehéz végbevinni
  - javítgatása helyett érdemesebb elölről kezdeni az egészet

A tervező könnyen a "wishful thinking" (vágyálom) csapdájába eshet, azaz azt gondolja, hogy az általa kitalált absztrakció egyszerű és robosztus és csak az implementáció során jön rá, hogy az valójában zavaros és inkonzisztens.

Az Alloyról

Az Alloy deklaratív specifikációs nyelv a szoftver rendszerek tervezésében a helyes absztrakció megtalálására.

Fejlesztői: MIT Szoftvertervezési Csoportja
Daniel Jackson vezetésével
a projekt 1994-től folyamatos
a jelenlegi verzió: Alloy Analyzer 4.1
A gyökerek:

- a Z specifikációs nyelv (Oxford)
  egyszerű, pontos jelölés (ZF halmazelmélet + relációs logika)
  de itt nem automatikus az analízis
- az SMV modellellenőrző (Pittsburg)
  gyors és automatikus analízis
  de ez nem deklaratív

Az Alloyról

Az Alloy deklaratív specifikációs nyelv a szoftver rendszerek tervezésében a helyes absztrakció megtalálására.
Fejlesztői: MIT Szoftvertervezési Csoportja
Daniel Jackson vezetésével
a projekt 1994-től folyamatos
a jelenlegi verzió: Alloy Analyzer 4.1

A gyökerek:

- a Z specifikációs nyelv (Oxford)
  egyszerű, pontos jelölés (ZF halmazelmélet + relációs logika)
  de itt nem automatikus az analízis
- az SMV modellellenőrző (Pittsburg)
  gyors és automatikus analízis
  de ez nem deklaratív

Az Alloyról

Az Alloy deklaratív specifikációs nyelv a szoftver rendszerek tervezésében a helyes absztrakció megtalálására.
Fejlesztői: MIT Szoftvertervezési Csoportja
Daniel Jackson vezetésével
a projekt 1994-től folyamatos
a jelenlegi verzió: Alloy Analyzer 4.1
A gyökerek:

- a Z specifikációs nyelv (Oxford)
  egyszerű, pontos jelölés (ZF halmazelmélet + relációs logika)
  de itt nem automatikus az analízis
- az SMV modellellenőrző (Pittsburg)
  gyors és automatikus analízis
  de ez nem deklaratív

Az Alloyról

Az Alloy deklaratív specifikációs nyelv a szoftver rendszerek tervezésében a helyes absztrakció megtalálására.
Fejlesztői: MIT Szoftvertervezési Csoportja
Daniel Jackson vezetésével
a projekt 1994-től folyamatos
a jelenlegi verzió: Alloy Analyzer 4.1
A gyökerek:

- a Z specifikációs nyelv (Oxford)
  egyszerű, pontos jelölés (ZF halmazelmélet + relációs logika)
  de itt nem automatikus az analízis
- az SMV modellellenőrző (Pittsburg)
  gyors és automatikus analízis
  de ez nem deklaratív

Az Alloy 4 alapötlete

1. Minden reláció.
2. Nem-specializált logika használata.
3. Ellenpéldák és terjedelem (scope)
   (a Small Scope Hypothesis alapján, [2])
4. analízis a SAT megoldók segítségével

Az Alloy 4 alapötlete

1. Minden reláció.
2. Nem-specializált logika használata.
3. Ellenpéldák és terjedelem (scope)
   (a Small Scope Hypothesis alapján, [2])
4. analízis a SAT megoldók segítségével

Az Alloy 4 alapötlete

1. Minden reláció.
2. Nem-specializált logika használata.
3. Ellenpéldák és terjedelem (scope)
   (a Small Scope Hypothesis alapján, [2])
4. analízis a SAT megoldók segítségével

Az Alloy 4 alapötlete

1. Minden reláció.
2. Nem-specializált logika használata.
3. Ellenpéldák és terjedelem (scope)
   (a Small Scope Hypothesis alapján, [2])
4. analízis a SAT megoldók segítségével

Az Alloy 4 alapötlete

1. Minden reláció.
2. Nem-specializált logika használata.
3. Ellenpéldák és terjedelem (scope)
   (a Small Scope Hypothesis alapján, [2])
4. analízis a SAT megoldók segítségével

Vágjunk vele: lássunk egy példát!

Bemutató: Címjegyzék kezelő program tervezése e-mail klienshez.

Daniel Jackson [6] könyvének 2. fejezete alapján.

Az Alloy elemei

Alloy = logika + nyelv + analízis

- logika
  elsőrendű logika + reláció kalkulus
- nyelv
  szintaxis a logikai specifikációk struktúrálására
- analízis
  kimerítő (de korlátozott) keresés ellenpéldákra, melyek sértik a specifikációt

Az Alloy elemei

Alloy = logika + nyelv + analízis

- logika
  elsőrendű logika + reláció kalkulus
- nyelv
  szintaxis a logikai specifikációk struktúrálására
- analízis
  kimerítő (de korlátozott) keresés ellenpéldákra, melyek sértik a specifikációt

Az Alloy elemei

Alloy = logika + nyelv + analízis

- logika
  elsőrendű logika + reláció kalkulus
- nyelv
  szintaxis a logikai specifikációk struktúrálására
- analízis
  kimerítő (de korlátozott) keresés ellenpéldákra, melyek sértik a specifikációt

Az Alloy elemei

Alloy = logika + nyelv + analízis

- logika
  elsőrendű logika + reláció kalkulus
- nyelv
  szintaxis a logikai specifikációk struktúrálására
- analízis
  kimerítő (de korlátozott) keresés ellenpéldákra, melyek sértik a specifikációt

LOGIKA: halmazműveletek (relációkra is)

`p + q`:	unió
`p - q`:	különbség
`p & q`:	metszet
`p in q`:	részhalmaz reláció
`p = q`:	egyenlőség reláció

Példa:

```
b'.addr = b.addr + n->a
```

A `b'` címjegyzék `b'.addr` relációja a `b.addr` reláció és az (n, a) pár (egyelemű reláció) uniója.

LOGIKA: halmazműveletek (relációkra is)

p + q:	unió
p - q:	különbség
p & q:	metszet
p in q:	részhalmaz reláció
p = q:	egyenlőség reláció

Példa:

```
b'.addr = b.addr + n->a
```

A **b'** címjegyzék **b'.addr** relációja a **b.addr** reláció és az (n, a) pár (egyelemű reláció) uniója.

LOGIKA: halmazműveletek (relációkra is)

p + q:	unió
p - q:	különbség
p & q:	metszet
p in q:	részhalmaz reláció
p = q:	egyenlőség reláció

Példa:

```
b'.addr = b.addr + n->a
```

A **b'** címjegyzék **b'.addr** relációja a **b.addr** reláció és az (n, a) pár (egyelemű reláció) uniója.

LOGIKA: halmazműveletek (relációkra is)

p + q:	unió
p - q:	különbség
p & q:	metszet
p in q:	részhalmaz reláció
p = q:	egyenlőség reláció

Példa:

```
b'.addr = b.addr + n->a
```

A **b'** címjegyzék **b'.addr** relációja a **b.addr** reláció és az (n, a) pár (egyelemű reláció) uniója.

LOGIKA: reláció műveletek: nyíl szorzat (arrow product)

p -> q:=

$\{ (p_1, \ldots, p_m, q_1, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \}$

Halmazok esetén ez a szokásos Descartes-szorzat.

Például

Name = { (N0), (N1) },

Addr = { (A0), (A1) },

Book = { (B0) } esetén

Name->Addr = { (N0,A0), (N0,A1), (N1,A0), (N1,A1) }

Book->Name->Addr = { (B0,N0,A0), (B0,N0,A1), (B0,N1,A0), (B0,N1,A1) }

LOGIKA: reláció műveletek: nyíl szorzat (arrow product)

p -> q:=

$\{ (p_1, \ldots, p_m, q_1, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \}$

Halmazok esetén ez a szokásos Descartes-szorzat.

Például

Name = { (N0), (N1) },

Addr = { (A0), (A1) },

Book = { (B0) } esetén

Name->Addr = { (N0,A0), (N0,A1), (N1,A0), (N1,A1) }

Book->Name->Addr = { (B0,N0,A0), (B0,N0,A1), (B0,N1,A0), (B0,N1,A1) }

LOGIKA: reláció műveletek: nyíl szorzat (arrow product)

p -> q:=

$\{ (p_1, \ldots, p_m, q_1, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \}$

Halmazok esetén ez a szokásos Descartes-szorzat.

Például

Name = { (N0), (N1) },

Addr = { (A0), (A1) },

Book = { (B0) } esetén

Name->Addr = { (N0,A0), (N0,A1), (N1,A0), (N1,A1) }

Book->Name->Addr = { (B0,N0,A0), (B0,N0,A1), (B0,N1,A0), (B0,N1,A1) }

LOGIKA: reláció műveletek: nyíl szorzat (arrow product)

p -> q:=

$\{ (p_1, \ldots, p_m, q_1, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \}$

Halmazok esetén ez a szokásos Descartes-szorzat.

Például

Name = { (N0), (N1) },

Addr = { (A0), (A1) },

Book = { (B0) } esetén

Name->Addr = { (N0,A0), (N0,A1), (N1,A0), (N1,A1) }

Book->Name->Addr = { (B0,N0,A0), (B0,N0,A1), (B0,N1,A0), (B0,N1,A1) }

LOGIKA: reláció műveletek: nyíl szorzat (arrow product)

p -> q:=

$\{ (p_1, \ldots, p_m, q_1, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \}$

Halmazok esetén ez a szokásos Descartes-szorzat.

Például

Name = { (N0), (N1) },

Addr = { (A0), (A1) },

Book = { (B0) } esetén

Name->Addr = { (N0,A0), (N0,A1), (N1,A0), (N1,A1) }

Book->Name->Addr = { (B0,N0,A0), (B0,N0,A1), (B0,N1,A0), (B0,N1,A1) }

LOGIKA: reláció műveletek: nyíl szorzat (arrow product)

p -> q:=

$\{ (p_1, \ldots, p_m, q_1, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \}$

Halmazok esetén ez a szokásos Descartes-szorzat.

Például

Name = { (N0), (N1) },

Addr = { (A0), (A1) },

Book = { (B0) } esetén

Name->Addr = { (N0,A0), (N0,A1), (N1,A0), (N1,A1) }

Book->Name->Addr = { (B0,N0,A0), (B0,N0,A1), (B0,N1,A0), (B0,N1,A1) }

LOGIKA: reláció műveletek: pont összefűzés (dot join)

$$\mathbf{p.q} := \{\, (p_1, \ldots, p_{m-1}, q_2, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \wedge p_m = q_1 \,\}$$

Binér relációk esetén ez a szokásos relációk kompozíciója.

(a, b) (a, c) (c, d)	·	(a, d, c) (b, c, c) (b, a, d) (c, c, c)	=	(a, c, c) (a, a, d)

Ha **f** binér reláció, **x** és **y** pedig halmazok, akkor
x.f = az **x** halmaz **f** általi képe,
f.y = az **y** halmaz **f** általi inverzképe.

LOGIKA: reláció műveletek: pont összefűzés (dot join)

$\mathbf{p.q} := \{ (p_1, \ldots, p_{m-1}, q_2, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \wedge p_m = q_1 \}$

Binér relációk esetén ez a szokásos relációk kompozíciója.

(a, b) (a, c) (c, d)	·	(a, d, c) (b, c, c) (b, a, d) (c, c, c)	=	(a, c, c) (a, a, d)

Ha **f** binér reláció, **x** és **y** pedig halmazok, akkor
x.f = az **x** halmaz **f** általi képe,
f.y = az **y** halmaz **f** általi inverzképe.

LOGIKA: reláció műveletek: pont összefűzés (dot join)

$\mathbf{p.q} := \{ (p_1, \ldots, p_{m-1}, q_2, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \wedge p_m = q_1 \}$

Binér relációk esetén ez a szokásos relációk kompozíciója.

(a, b) (a, c) (c, d)	$\cdot$	(a, d, c) (b, c, c) (b, a, d) (c, c, c)	$=$	(a, c, c) (a, a, d)

Ha **f** binér reláció, **x** és **y** pedig halmazok, akkor
x.f = az **x** halmaz **f** általi képe,
f.y = az **y** halmaz **f** általi inverzképe.

LOGIKA: reláció műveletek: pont összefűzés (dot join)

$\mathbf{p.q} := \{ (p_1, \ldots, p_{m-1}, q_2, \ldots, q_n) \mid (p_1, \ldots, p_m) \in p \wedge (q_1, \ldots, q_n) \in q \wedge p_m = q_1 \}$

Binér relációk esetén ez a szokásos relációk kompozíciója.

(a, b) (a, c) (c, d)	·	(a, d, c) (b, c, c) (b, a, d) (c, c, c)	=	(a, c, c) (a, a, d)

Ha **f** binér reláció, **x** és **y** pedig halmazok, akkor
x.f = az **x** halmaz **f** általi képe,
f.y = az **y** halmaz **f** általi inverzképe.

LOGIKA: további reláció műveletek

Binér relációkra:

- megfordítás (transpose): `~p`
- tranzitív lezárt: `^p`
- reflexív tranzitív lezárt: `*p`

Általános relációkra:

- felülírás (override): `p++q`
- értelmezési tartomány megszorítás : `d<:p`
- értékkészlet megszorítás : `p:>r`

Például:

```
p ++ q = q + (p - (domq<:p))
```

LOGIKA: további reláció műveletek

Binér relációkra:

- megfordítás (transpose): **~p**
- tranzitív lezárt: **^p**
- reflexív tranzitív lezárt: ***p**

Általános relációkra:

- felülírás (override): **p++q**
- értelmezési tartomány megszorítás : **d<:p**
- értékkészlet megszorítás : **p:>r**

Például:

```
p ++ q = q + (p - (domq<:p))
```

LOGIKA: további reláció műveletek

Binér relációkra:

- megfordítás (transpose): **~p**
- tranzitív lezárt: **^p**
- reflexív tranzitív lezárt: ***p**

Általános relációkra:

- felülírás (override): **p++q**
- értelmezési tartomány megszorítás : **d<:p**
- értékkészlet megszorítás : **p:>r**

Például:

```
p ++ q = q + (p - (domq<:p))
```

LOGIKA: további reláció műveletek

Binér relációkra:

- megfordítás (transpose): **~p**
- tranzitív lezárt: **^p**
- reflexív tranzitív lezárt: ***p**

Általános relációkra:

- felülírás (override): **p++q**
- értelmezési tartomány megszorítás : **d<:p**
- értékkészlet megszorítás : **p:>r**

Például:

```
p ++ q = q + (p - (domq<:p))
```

LOGIKA: Boole-műveletek

!	**not**	tagadás
&&	**and**	konjunkció
||	**or**	diszjunkció
=>	**implies**	implikáció
,	**else**	alternetíva
<=>	**iff**	ekvivalencia

Így például

$$F \Rightarrow G, H \equiv$$

$$F \textbf{ implies } G \textbf{ else } H \equiv$$

$$(F \,\&\&\, G) \,||\, ((!F) \,\&\&\, H) \equiv$$

$$(F \textbf{ and } G) \textbf{ or } (\textbf{not } F) \textbf{ and } H)$$

LOGIKA: Boole-műveletek

!	**not**	tagadás
&&	**and**	konjunkció
||	**or**	diszjunkció
=>	**implies**	implikáció
,	**else**	alternetíva
<=>	**iff**	ekvivalencia

Így például

$$F \Rightarrow G, H \quad \equiv$$

$$F \textbf{ implies } G \textbf{ else } H \quad \equiv$$

$$(F \,\&\&\, G) \,||\, ((!F) \,\&\&\, H) \quad \equiv$$

$$(F \textbf{ and } G) \textbf{ or } (\textbf{not } F) \textbf{ and } H)$$

LOGIKA: kvantorok és számosságok

Kvantifikáció formulákra

`all x:e|F`	`F` minden `x`-re igaz `e`-ből
`some x:e|F`	`F` legalább egy `x`-re igaz `e`-ből
`no x:e|F`	`F` nem igaz egyetlen egy `x`-re sem `e`-ből
`lone x:e|F`	`F` legfeljebb egy `x`-re igaz `e`-ből
`one x:e|F`	pontosan egy `x`-re igaz `e`-ből

Kvantifikáció kifejezésekre

Definíció: `Q e <=> Q x:e | true`

`some e`	`e`-nek van legalább egy eleme
`no e`	`e`-nek nincs eleme
`lone e`	`e`-nek legfeljebb egy eleme van
`one e`	`e`-nek pontosan egy eleme van

LOGIKA: kvantorok és számosságok

Kvantifikáció formulákra

all x:e|F	**F** minden **x**-re igaz **e**-ből
some x:e|F	**F** legalább egy **x**-re igaz **e**-ből
no x:e|F	**F** nem igaz egyetlen egy **x**-re sem **e**-ből
lone x:e|F	**F** legfeljebb egy **x**-re igaz **e**-ből
one x:e|F	pontosan egy **x**-re igaz **e**-ből

Kvantifikáció kifejezésekre

Definíció: `Q e <=> Q x:e | true`

some e	**e**-nek van legalább egy eleme
no e	**e**-nek nincs eleme
lone e	**e**-nek legfeljebb egy eleme van
one e	**e**-nek pontosan egy eleme van

LOGIKA: kvantorok és számosságok

Kvantifikáció formulákra

all x:e|F	**F** minden **x**-re igaz **e**-ből
some x:e|F	**F** legalább egy **x**-re igaz **e**-ből
no x:e|F	**F** nem igaz egyetlen egy **x**-re sem **e**-ből
lone x:e|F	**F** legfeljebb egy **x**-re igaz **e**-ből
one x:e|F	pontosan egy **x**-re igaz **e**-ből

Kvantifikáció kifejezésekre

Definíció: `Q e <=> Q x:e | true`

some e	**e**-nek van legalább egy eleme
no e	**e**-nek nincs eleme
lone e	**e**-nek legfeljebb egy eleme van
one e	**e**-nek pontosan egy eleme van

Multiplicitások halmaz/reláció deklarációkban

Multiplikatív kulcsszavak

set	tetszőleges számú
one	pontosan egy
lone	nulla vagy egy
some	egy vagy több

- halmaz deklarációk:
  `s:m e = s in e and m s`,
  azaz `s` az `e`-nek egy `m` elemű részhalmaza
  `s:e <=> s:one e`, ez az alapértelmezés.
- reláció deklarációk:
  `r: e m -> n e'` `r in e -> e'`
  `all x:e | n x.r`
  `all y:e'| m r.y`
  `r: e -> e'` `r: e set -> set e'` (alapért.)

Multiplicitások halmaz/reláció deklarációkban

Multiplikatív kulcsszavak

set	tetszőleges számú
one	pontosan egy
lone	nulla vagy egy
some	egy vagy több

- halmaz deklarációk:
  `s:m e = s in e and m s`,
  azaz `s` az `e`-nek egy `m` elemű részhalmaza
  `s:e` `<=> s:one e`, ez az alapértelmezés.
- reláció deklarációk:
  `r: e m -> n e'` `r in e -> e'`
  `all x:e | n x.r`
  `all y:e'| m r.y`
  `r: e -> e'` `r: e set -> set e'` (alapért.)

Multiplicitások halmaz/reláció deklarációkban

Multiplikatív kulcsszavak

set	tetszőleges számú
one	pontosan egy
lone	nulla vagy egy
some	egy vagy több

- halmaz deklarációk:
  `s:m e = s in e and m s`,
  azaz `s` az `e`-nek egy `m` elemű részhalmaza
  `s:e` `<=> s:one e`, ez az alapértelmezés.
- reláció deklarációk:
  `r: e m -> n e'` `r in e -> e'`
  `all x:e | n x.r`
  `all y:e'| m r.y`
  `r: e -> e'` `r: e set -> set e'` (alapért.)

Multiplicitások halmaz/reláció deklarációkban

Multiplikatív kulcsszavak

set	tetszőleges számú
one	pontosan egy
lone	nulla vagy egy
some	egy vagy több

- halmaz deklarációk:
  `s:m e = s in e and m s`,
  azaz `s` az `e`-nek egy `m` elemű részhalmaza
  `s:e` `<=> s:one e`, ez az alapértelmezés.
- reláció deklarációk:

```
r:  e m -> n e'   r in e -> e'
                  all x:e | n x.r
                  all y:e'| m r.y
r:  e -> e'       r:  e set -> set e'
```

(alapért.)

Példák

- `sig Book { names: set Name, addr: Name -> Addr}`
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges `Name`×`Addr` reláció tartozik.
- `sig Book { addr: Name -> lone Addr}`
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- `sig Book { addr: (Name -> lone Addr) one -> Time }`
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

Példák

- **`sig Book { names: set Name, addr: Name -> Addr}`**
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges `Name`×`Addr` reláció tartozik.
- **`sig Book { addr: Name -> lone Addr}`**
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- **`sig Book { addr: (Name -> lone Addr) one -> Time }`**
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

Példák

- **`sig Book { names: set Name, addr: Name -> Addr}`**
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges **`Name`**$\times$**`Addr`** reláció tartozik.
- **`sig Book { addr: Name -> lone Addr}`**
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- **`sig Book { addr: (Name -> lone Addr) one -> Time }`**
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

Példák

- `sig Book { names: set Name, addr: Name -> Addr}`
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges `Name`×`Addr` reláció tartozik.
- `sig Book { addr: Name -> lone Addr}`
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- `sig Book { addr: (Name -> lone Addr) one -> Time }`
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

Példák

- `sig Book { names: set Name, addr: Name -> Addr}`
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges `Name`×`Addr` reláció tartozik.
- `sig Book { addr: Name -> lone Addr}`
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- `sig Book { addr: (Name -> lone Addr) one -> Time }`
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

Példák

- **`sig Book { names: set Name, addr: Name -> Addr}`**
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges **`Name`**$\times$**`Addr`** reláció tartozik.
- **`sig Book { addr: Name -> lone Addr}`**
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- **`sig Book { addr: (Name -> lone Addr) one -> Time }`**
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

Példák

- **`sig Book { names: set Name, addr: Name -> Addr}`**
  Minden címjegyzékhez nevek egy halmaza és egy tetszőleges **`Name`**$\times$**`Addr`** reláció tartozik.
- **`sig Book { addr: Name -> lone Addr}`**
  Könyvenként minden névhez, legfeljebb egy cím tartozik.
- **`sig Book { addr: (Name -> lone Addr) one -> Time }`**
  Minden könyvben pontosan egy név-cím reláció van az időpontokhoz rendelve. Több időpontban is lehet azonos a reláció.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: az Alloy modellek elemei

Szignatúrák és mezők nevek

- a halmazok és relációk elemeinek megnevezésére
- klasszifikáció és altípusok megadására

A megszorítások megadására

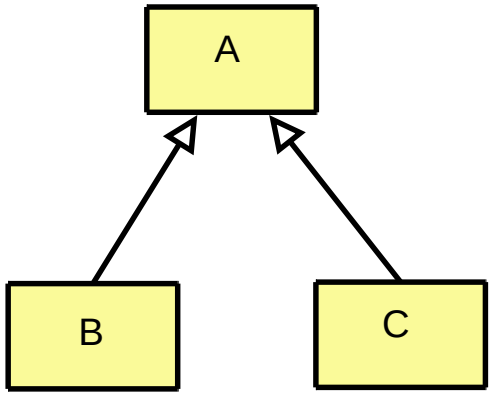
- fact: tény, csak olyan modelleket keresünk, melyek teljesítik.
- assertion: feltételezés, sejtés, olyan állítás, melynek helyességét ellenőrizni akarjuk.
- predicate: predikátum, többször használható állítás.
- function: függvény, többször használható kifejezés.

Parancsok

- run: egy állítást teljesítő példányokat generál.
- check: egy állítást sértő ellenpéldákat keres.

NYELV: Szignatúrák

A modell atomjainak (objektumainak) a típusát adjuk meg velük.




```
sig A {}
sig B extends A {}
sig C extends A {}
```

jelentése:

`B in A, C in A, no B&C`

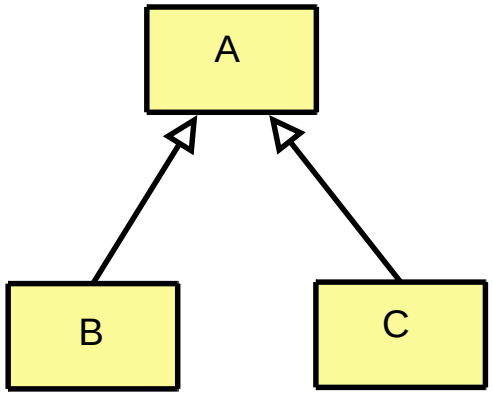
`abstract sig A {},`
`sig B extends A {},`
`sig C extends A {}`

Ugyanez és még: `A in (B+C)`

NYELV: Szignatúrák

A modell atomjainak (objektumainak) a típusát adjuk meg velük.

```
sig A {}
sig B extends A {}
sig C extends A {}
```



jelentése:

```
B in A, C in A, no B&C
```

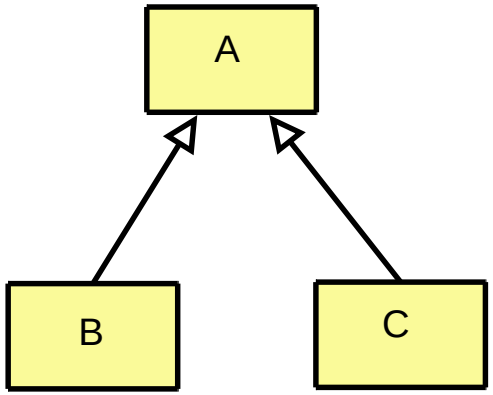
```
abstract sig A {},
sig B extends A {},
sig C extends A {}
```

Ugyanez és még: `A in (B+C)`

NYELV: Szignatúrák

A modell atomjainak (objektumainak) a típusát adjuk meg velük.

```
sig A {}
sig B extends A {}
sig C extends A {}
```



jelentése:

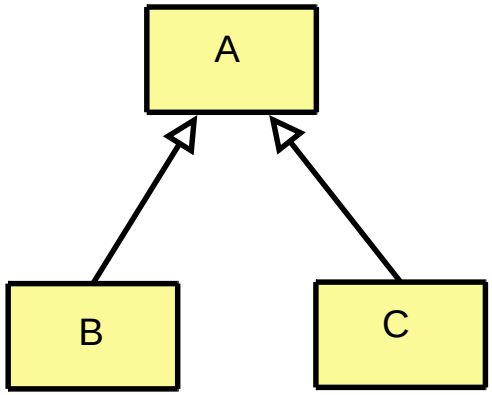
```
B in A, C in A, no B&C
```

```
abstract sig A {},
sig B extends A {},
sig C extends A {}
```

Ugyanez és még: `A in (B+C)`

NYELV: Szignatúrák

A modell atomjainak (objektumainak) a típusát adjuk meg velük.




```
sig A {}
sig B extends A {}
sig C extends A {}
```

jelentése:

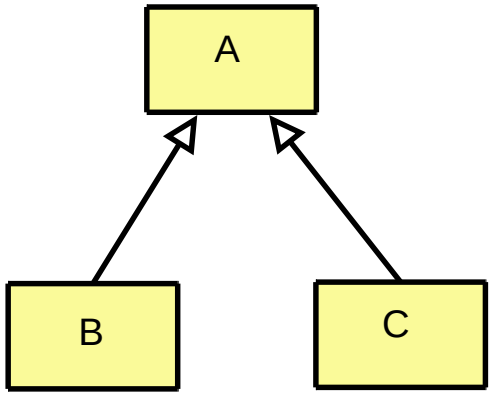
```
B in A, C in A, no B&C
```

```
abstract sig A {},
sig B extends A {},
sig C extends A {}
```

Ugyanez és még: `A in (B+C)`

NYELV: Szignatúrák

A modell atomjainak (objektumainak) a típusát adjuk meg velük.




```
sig A {}
sig B extends A {}
sig C extends A {}
```

jelentése:

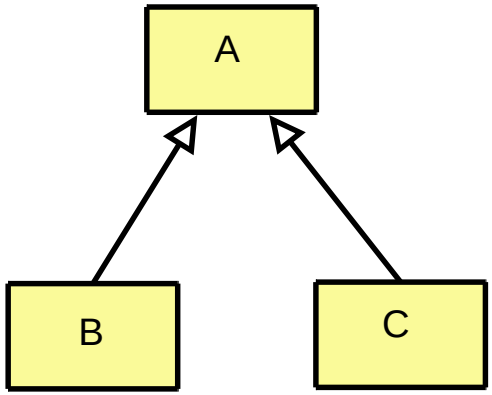
B in A, C in A, no B&C

abstract sig A {},
sig B extends A {},
sig C extends A {}

Ugyanez és még: `A in (B+C)`

NYELV: Szignatúrák

A modell atomjainak (objektumainak) a típusát adjuk meg velük.




```
sig A {}
sig B extends A {}
sig C extends A {}
```

jelentése:

B in A, **C in A**, **no B&C**

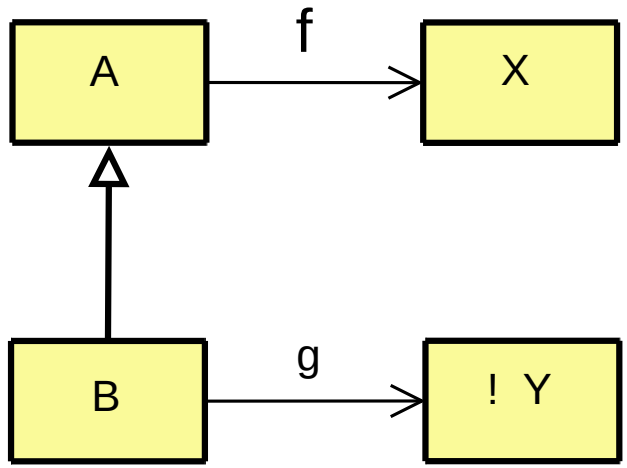
abstract sig A {},
sig B extends A {},
sig C extends A {}
Ugyanez és még: **A in (B+C)**

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

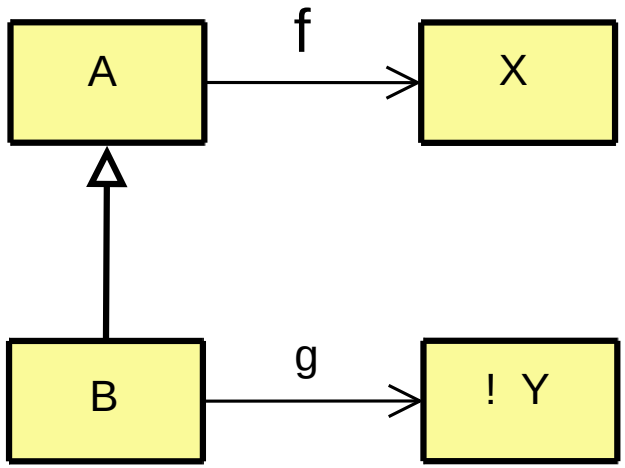
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

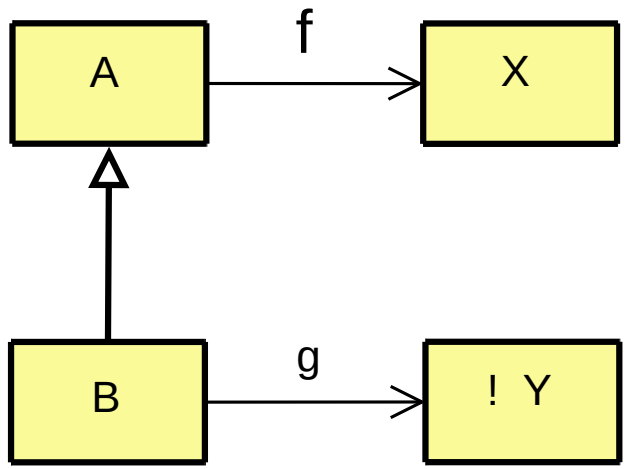
Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.




```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```

jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

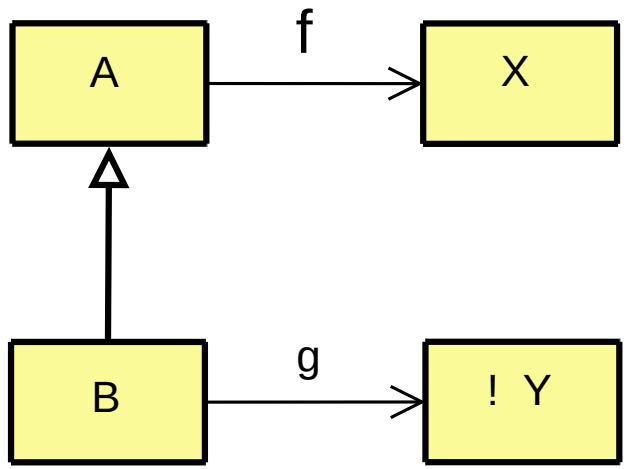
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

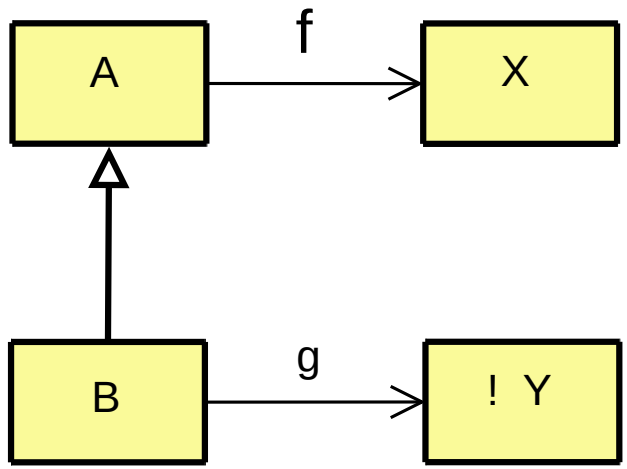
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

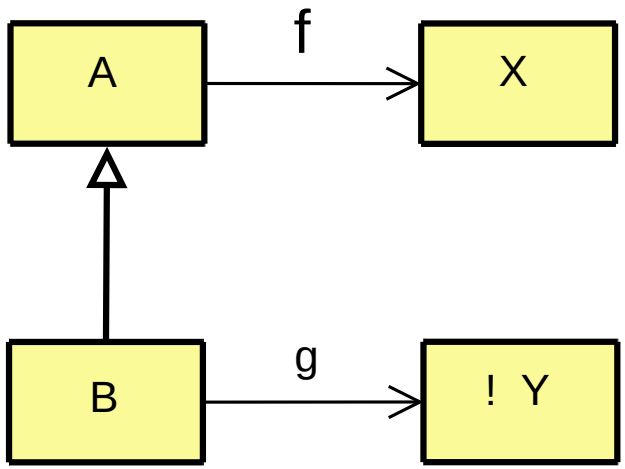
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

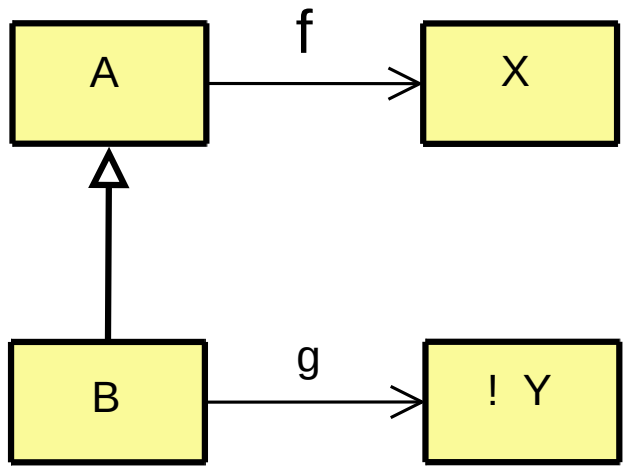
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

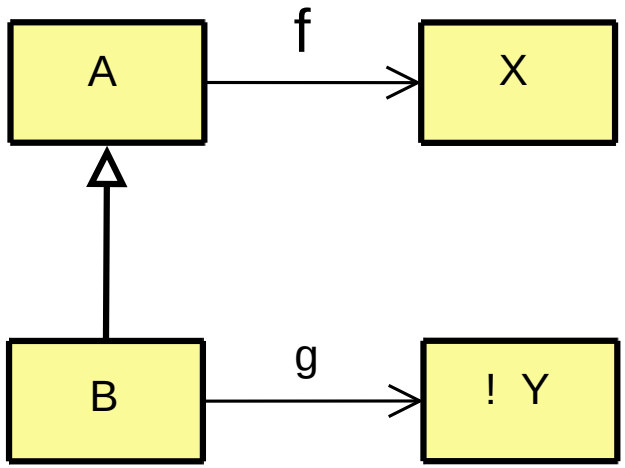
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert **`{g:  one Y}`** az alapértelmezés.

Ekkor jól definiált kifejezések minden **`a:A, b:B, x:X`** esetén:

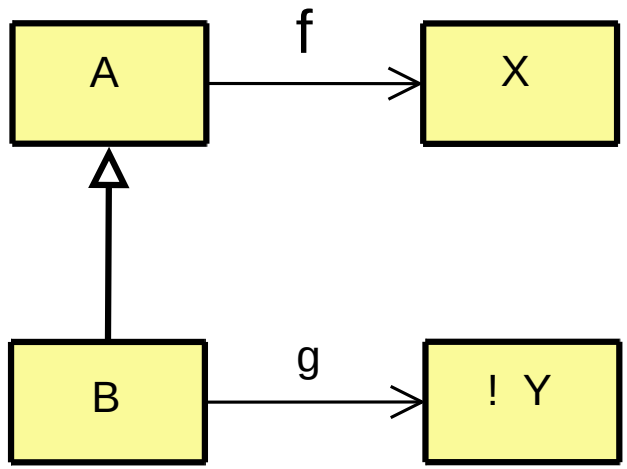
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

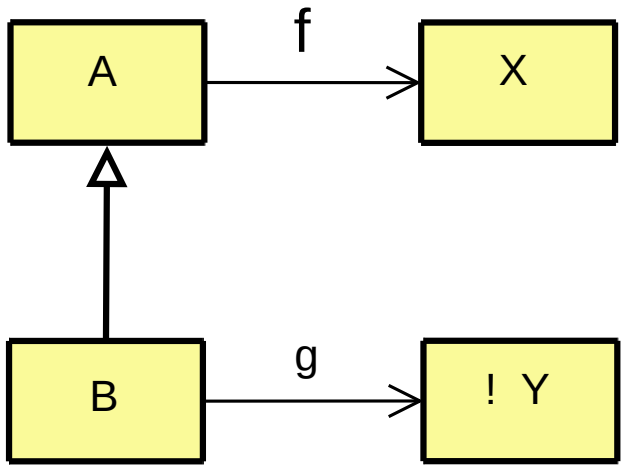
`a.f, f.x, b.g, b.f`

NYELV: Mezők

A mezők olyan relációk, melyek

- értelmezési tartománya azon szignatúra objektumai, melyben definiáltak;
- értékkészlete a mezőnév után megadott kifejezés értéke.

```
sig X,Y {}
sig A {f:  set X}
sig B extends A {g:  Y}
```



jelentése:

`B in A`, (altípus)

`f:A -> X` (többértékű) reláció

`g:B -> Y` függvény, mert `{g:  one Y}` az alapértelmezés.

Ekkor jól definiált kifejezések minden `a:A, b:B, x:X` esetén:

`a.f, f.x, b.g, b.f`

NYELV: fact, pred, run

```
fact F { ...}
pred P(param.)  { ...}
run P(param.)
```

jelentése:

1. Feltesszük, hogy az **F** megszorítás mindig igaz.
2. Definiáljuk a **P** megszorítást.
3. Olyan példányokat keresünk, melyek **F**-et és **P**-t egyaránt teljesítik.

NYELV: fact, pred, run

```
fact F { ...}
pred P(param.)  { ...}
run P(param.)
```

jelentése:

1. Feltesszük, hogy az **F** megszorítás mindig igaz.
2. Definiáljuk a **P** megszorítást.
3. Olyan példányokat keresünk, melyek **F**-et és **P**-t egyaránt teljesítik.

NYELV: assert, check

```
fact F { ...}
assert A { ...}
check A{ ...}
```

jelentése:

1. Feltesszük, hogy az **F** megszorítás mindig igaz.
2. Feltételezzük, hogy az **A** állítás következik **F**-ből.
3. Olyan példányokat keresünk, melyek **F**-et és nem **A**-t egyszerre teljesítik.

NYELV: assert, check

```
fact F { ...}
assert A { ...}
check A{ ...}
```

jelentése:

1. Feltesszük, hogy az **F** megszorítás mindig igaz.
2. Feltételezzük, hogy az **A** állítás következik **F**-ből.
3. Olyan példányokat keresünk, melyek **F**-et és nem **A**-t egyszerre teljesítik.

Egy példa

```
sig Node {
children:  set Node
}

one sig Root extends Node {}

fact { Node in Root.*children }

// invalid assertion:
assert HasParent {
all n:  Node | some children.n
}

// valid assertion:
assert HasParent2 {
all n:  Node - Root | some children.n
}
```

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy $\leftrightarrow$ Kodkod model finder $\leftrightarrow$ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy ↔ Kodkod model finder ↔ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy ↔ Kodkod model finder ↔ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy ↔ Kodkod model finder ↔ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy ↔ Kodkod model finder ↔ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy ↔ Kodkod model finder ↔ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy ↔ Kodkod model finder ↔ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy $\leftrightarrow$ Kodkod model finder $\leftrightarrow$ SAT solver (választható)

ANALÍZIS: Az Alloy Analyzer

- Szövegszerkesztő szintaxiskiemeléssel
- Példa/ellenpélda keresés futtatása
- Modellek megjelenítése
  - Szövegesen: fa szerkezetben vagy XML formátumban
  - Gráfként: testreszabható, témák + projekciók
  - Kiértékelő (Evaluator): kifejezések kiértékelésére a modellben
- Beállítások, optimalizálás

Az analízis menete:

Alloy $\leftrightarrow$ Kodkod model finder $\leftrightarrow$ SAT solver (választható)

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés

Pozitívumok

- **Egyszerűség**: könnyen tanulható, egyszerűen kezelhető
- **Vizualizáció**: jól testreszabható, azonnali kézzelfogható segítség
- **Általánosság**: erős kifejező erő, szabadság a modellezésben, nincsenek speciális szemantikai megkötések (pl. állapot gépekre, üzenetküldésre)
- **Fokozatosság**: mind a modellek, mind a specifikációk lépésről lépésre vezethetők be
- **Tömörség**: kisebb modellek szükségesek, mint a hasonló eszközökben.
- **Automatikus analízis**: az ellenpéldák kézzelfoghatók, további megszorításokkal méginkább testreszabhatók

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Rövid értékelés II

Negatívumok

- **a relációk sem jók mindenre**:
  - nagy aritás esetén nehézkes a kezelésük
  - igen korlátozott az egész számok kezelése
  - rekurzív függvényeket, iterációt nehéz velük kifejezni
  - gyorsan nő a modell mérete
- A visszavezetés SAT-ra kivitelezhető, de a
  - minden esetben való alkalmazhatósága és
  - hatékonysága

  számomra kérdéses.
- „Szakadék” a modell és az implementáció között.
- A keresés, a Kodkod és a SAT solver beállításai csak igen korlátozottan érhetők el az elemzőből.
- A pozitív példák minden igyekezet (például szimmetriát megtörő algoritmusok) ellenére sem mindig reprezentálják a lehetséges modellek összességét.

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Szubjektív értékelés

- Szerintem az ALLOY-ban a **szemléletváltás** a leglényegesebb: a könnyűsúlyúság megvalósítása
- Persze ez (mint minden gyakorlati verifikációs módszer) egyfajta „valamit-valamiért” (trade off)
- fel kellett áldoznunk érte
  - a specifikáció teljességét és
  - a bizonyítások matematikai szigorát (csak véges terjedelemre bizonyítunk)
- cserébe egy olyan eszközt kaptunk, mely
  - igen általános és
  - könnyen használható.
- jelenleg még inkább akadémiai és nem utolsósorban oktatási, mint ipari eszköz, melyben
  - számos fejlesztési lehetőség rejlik (optimalizálás)
  - és a könnyűsúlyú formális módszerek egyik vezérhajója (lehet).

Irodalomjegyzék I.

[1] Alloy Community: `http://alloy.mit.edu` (innen tölthető le az Alloy, tutoriálok, példák, fórum).
D. Jackson: Alloy in 90 minutes, tutorial, RE'05
G. Dennis, R. Seater Alloy Tutorial, FM 2006.

[2] Alexandr Andoni, Dumitru Daniliuc, Sarfraz Khurshid, Darko Marinov: Evaluating the „Small Scope Hypothesis", In: *POPL'02: Proceedings of the 29th ACM Symposium on the Principles of Programming Languages*, `http://mulsaw.lcs.mit.edu/papers/SSH.ps`

[3] R. W. Butler (2001-08-06). "What is Formal Methods?", `http://shemesh.larc.nasa.gov/fm/fm-what.html`, elérve 2009-09-02.

[4] J-F. Monin: *Understanding Formal Methods*, Springer, 2003.

Irodalomjegyzék II.

[5] Daniel Jackson: Alloy: a lightweight object modelling notation, *ACM Trans. Softw. Eng. Methodol.*, 11(2002), 256-290.

[6] Daniel Jackson: *Software Abstractions, Logic, Languages, and analysis*, MIT Press, 2006.

[7] Eunsuk Kang, Daniel Jackson: Formal Modeling and Analysis of a Flash Filesystem in Alloy, In: *Proc. ABZ 2008*, 294-308.