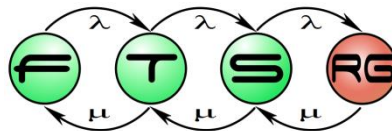


Szoftver-modellellenőrzés absztrakciós módszerekkel

Hajdu Ákos

hajdua@mit.bme.hu

2018.03.14.



BEVEZETŐ

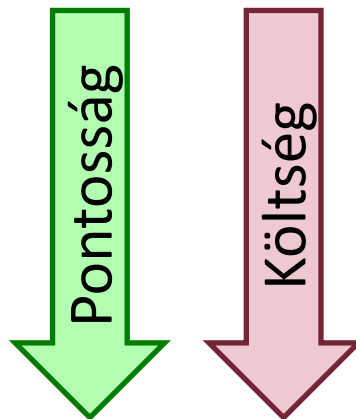
Bevezető

■ Motiváció

- Forráskód **közvetlen ellenőrzése**
- „Gombnyomásra” működjön
 - Komoly háttérismeretek nélkül

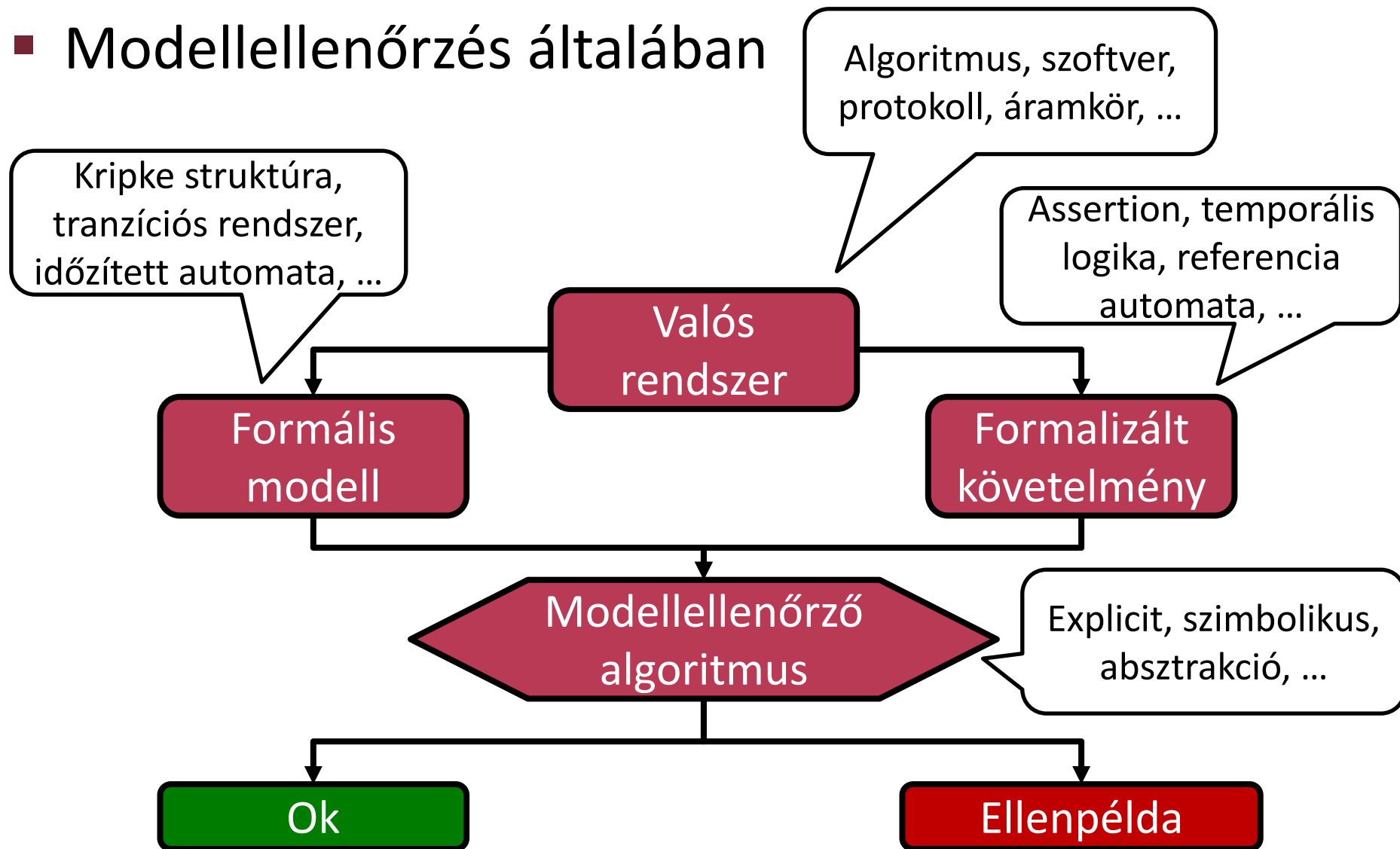
■ Jellegzetes szoftverellenőrzési technikák

- Statikus analízis
 - Hibaminta kereső
 - Absztrakt interpretáció
- **Modellellenőrzés**



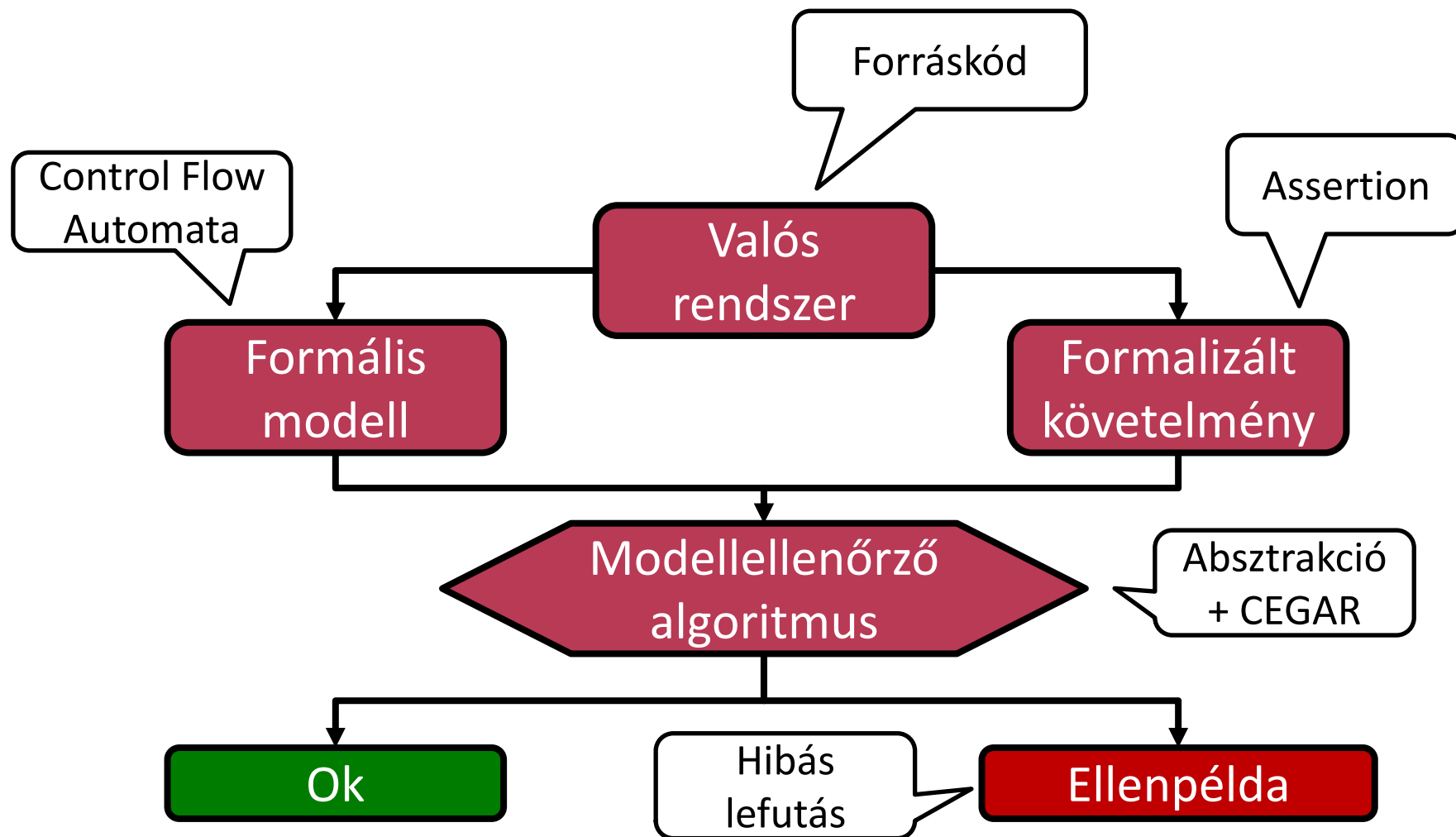
Bevezető – Modellellenőrzés

■ Modellellenőrzés általában



Bevezető – Modellellenőrzés

- Mai előadás: **szoftver** és **absztrakció**

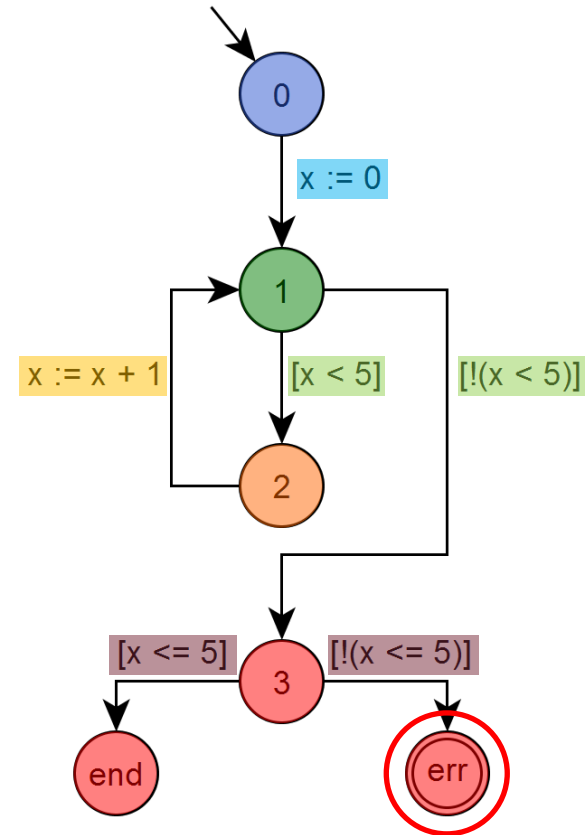
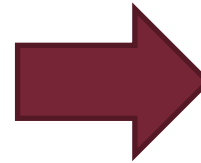


Bevezető – Modell és követelmény

■ Control-Flow Automata (CFA)

- Vezérlési **helyek**: L halmaz ($l_0, l_1, \dots$)
- **Élek** (G halmaz), **műveletekkel** a **változók** felett
 - Pl. őrfeltétel, értékadás, ...

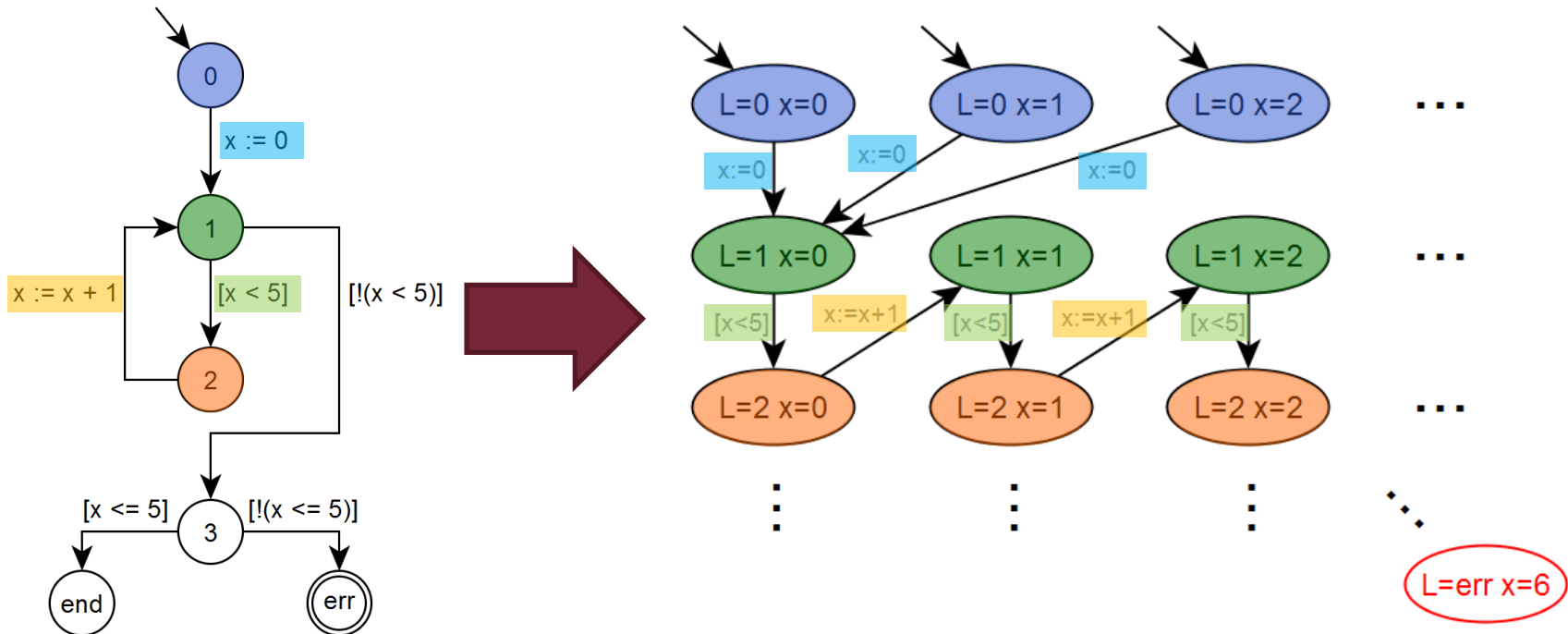
```
x : int
0: x = 0
1: while (x < 5) {
2:   x = x + 1
}
3: assert (x <= 5)
```



- Tipikus követelmény: „**error**” **hely** (jelölés: l_E) ne legyen elérhető

Bevezető – Állapotok és átmenetek

- **Állapot:** vezérlési hely + változók értékei ($L, x_1, x_2, \dots, x_n$)
- **Átmenet:** műveletek
- Probléma: **állapottér robbanás** az adatváltozók miatt
 - Pl.: 10 vezérlési hely, 2 db 32 bites int $\rightarrow 10 \cdot 2^{32} \cdot 2^{32}$ lehetséges állapot
- **Cél:** állapottér reprezentáció méretének **csökkentése absztrakcióval**



Bevezető – Matematikai logika

■ Propozicionális logika (nulladrendű)

- Boolean változók és operátorok
- SAT probléma: formula kielégíthető-e
 - Példa: korlátos modellellenőrzés
- Kifejezőerő nem mindig elégséges

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

■ Elsőrendű logika

- Függvények, predikátumok, kvantorok
- Általános esetben nem eldönthető

$$\forall x, y \exists z: p(f(x, y), g(z))$$

■ Satisfiability Modulo Theories (SMT)

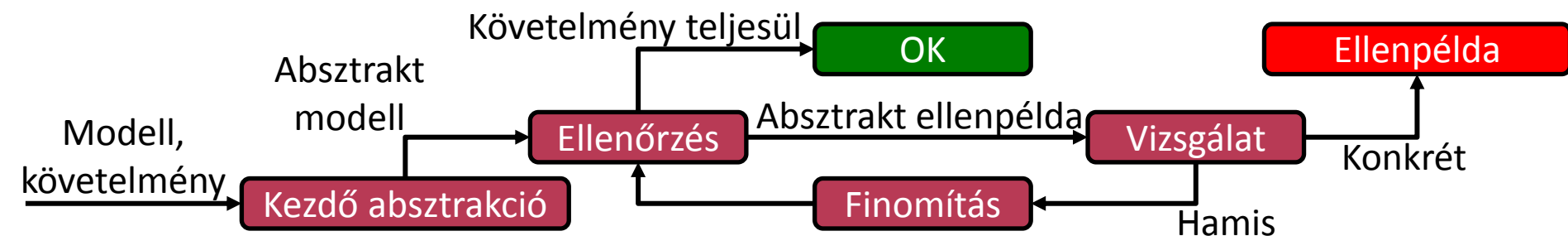
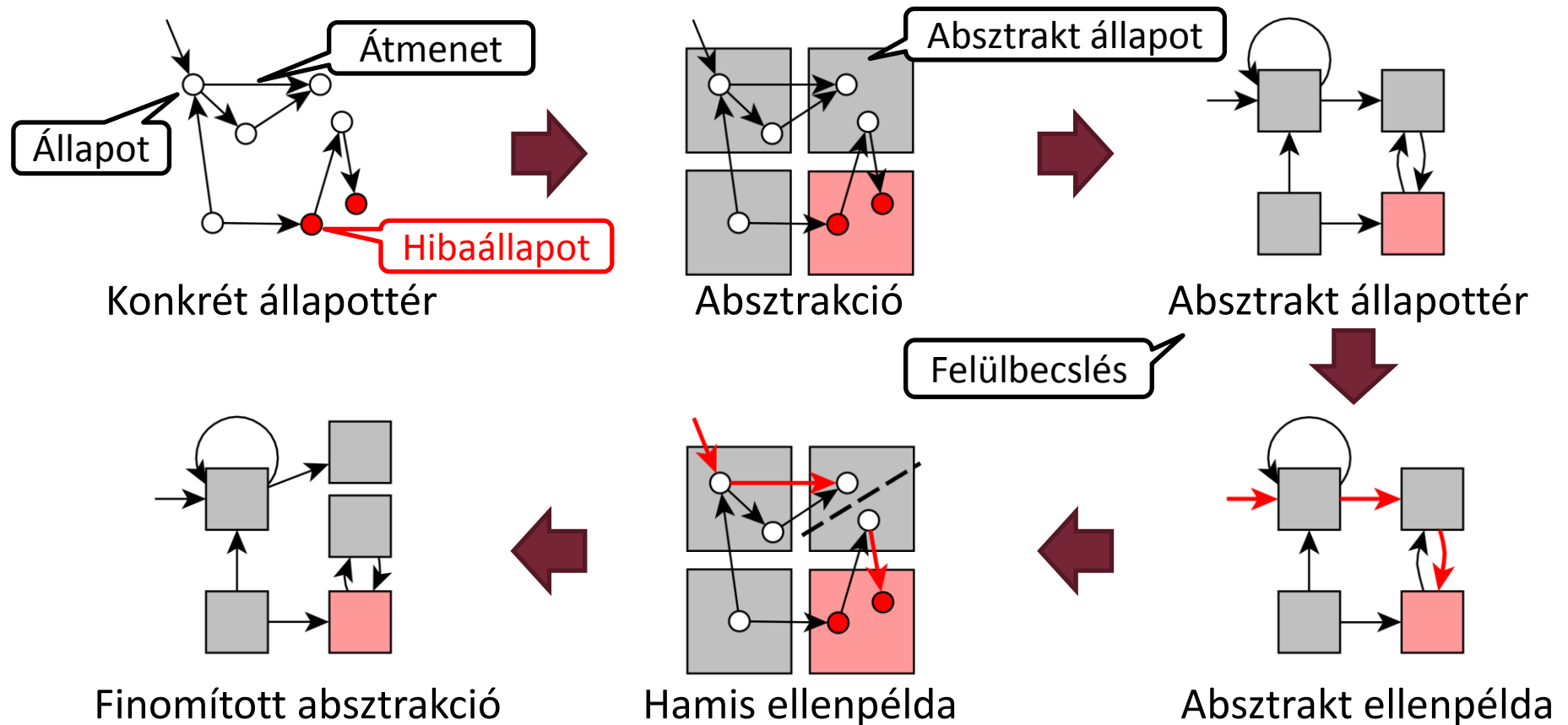
- Elsőrendű logikai formulák
- Interpretált szimbólumok
 - Pl. egész aritmetika

$$(x \leq y + 1) \wedge (y \geq 3)$$

COUNTEREXAMPLE-GUIDED ABSTRACTION REFINEMENT (CEGAR)

(Ellenpélda-alapú absztrakció finomítás)

CEGAR – Bevezető áttekintés

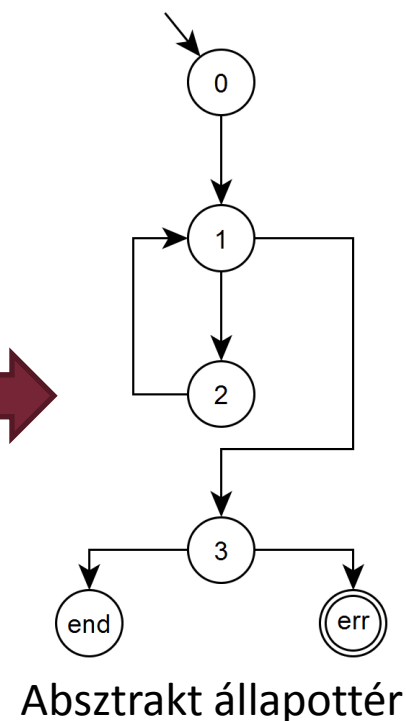
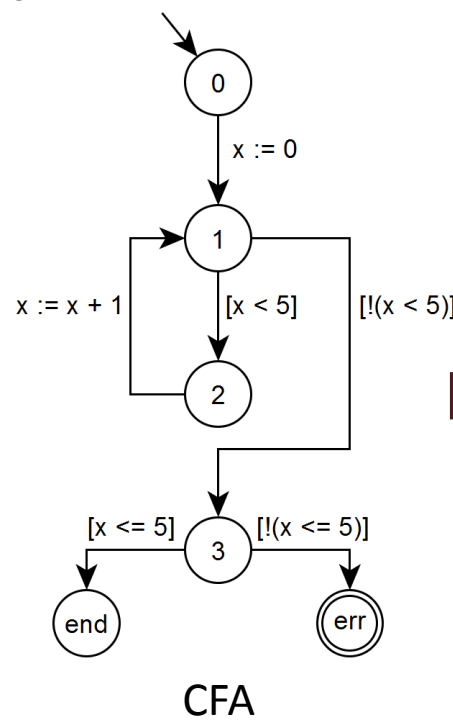


Absztrakció – Bevezető

- Absztrakció
 - Általános matematikai eszköz
 - Cél: Részletek elrejtése
 - Eredmény: Könnyebb probléma

- Absztrakció példa

- Vezérlési hely absztrakció
 - $(l, x_1, x_2, \dots, x_n) \rightarrow (l)$
- Önmagában általában nem célszerű
 - Triviálisan elérhető a hibaállapot
 - Kiegészítés
predikátumabsztrakcióval



Predikátumabsztrakció

■ Predikátumabsztrakció

- Változók **konkrét értékei helyett** a változókon értelmezett **predikátumok nyilvántartása**
- Absztrakt állapot: adott vezérlési helyhez tartozó konkrét állapotok, amelyekre **ugyanazok a predikátumok** teljesülnek

■ Absztrakció kiszámítása: Első lehetőség

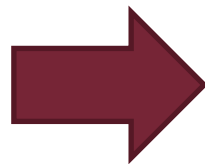
- Konkrét állapotok összegyűjtése és összevonása
 - A konkrét állapotok összegyűjtése: Állapottér robbanás ☹
- Példa: 3x3 konkrét állapot, 3 predikátum → 5 absztrakt állapot

Változók:

$$x, y; D_x = D_y = \{0, 1, 2\}$$

Predikátumok:

$$(x = y), (x < y), (y = 2)$$



$y \backslash x$	0	1	2
0	$(x = y)$		
1	$(x < y)$	$(x = y)$	
2	$(x < y)$ $(y = 2)$	$(x < y)$ $(y = 2)$	$(x = y)$ $(y = 2)$

Predikátumabsztrakció

- Absztrakció kiszámítása: Másik módszer
 - Csak az absztrakt állapotok felsorolása: Mi lehetséges?
 - P predikátumhalmaz $\rightarrow |L| \cdot 2^{|P|}$ lehetséges absztrakt állapot

- Példa

- 3 predikátum $\rightarrow$ 8 lehetséges absztrakt állapot vezérlési helyenként
- Ténylegesen nem mind fordulhat elő
 - SMT megoldóval kiszűrhető
 - Pl. együtt nem fordulhat elő:
 $(x = y) \wedge (x < y) \wedge \neg(y = 2)$

	$x = y$	$x < y$	$y = 2$
1	X	X	X
2	X	X	✓
3	X	✓	X
4	X	✓	✓
5	✓	X	X
6	✓	X	✓
7	✓	✓	X
8	✓	✓	✓

Predikátumabsztrakció

■ Absztrakt állapotok Boole változókkal

- $(l, x_1, \dots, x_n) \rightarrow (l, b_1, \dots, b_m)$
- b_i Boole változó: i . predikátum teljesül vagy nem
- Jelölés: $p(b_i) = \begin{cases} p_i & \text{ha } b_i \text{ igaz} \\ \neg p_i & \text{egyébként} \end{cases}$

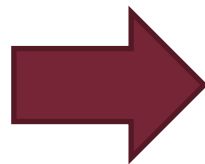
■ Példa

Változók:

$$x, y; D_x = D_y = \{0, 1, 2\}$$

Predikátumok:

$$(x = y), (x < y), (y = 2)$$



$$\begin{array}{ccc} l & x & y \\ \downarrow & \downarrow & \downarrow \\ (0, 0, 0) & \rightarrow & (0, T, F, F) \\ (6, 1, 2) & \rightarrow & (6, F, T, T) \end{array}$$

$l \quad (x = y) \quad (x < y) \quad (y = 2)$

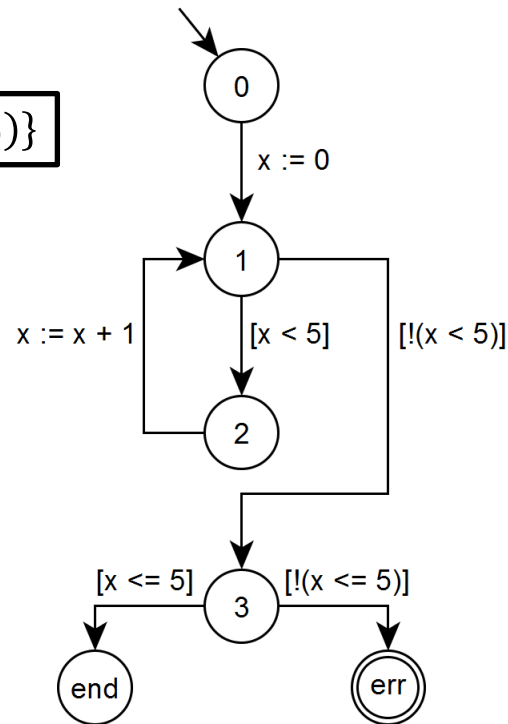
Predikátumabsztrakció

- Absztrakt kezdőállapot, hibaállapot
 - Absztrakt **kezdőállapot**: $(l, b_1, \dots, b_m)$, ahol $l = l_0$
 - Absztrakt **hibaállapot**: $(l, b_1, \dots, b_m)$, ahol $l = l_E$
- Absztrakt átmenetek
 - Absztrakt **átmenet**: Létezik, ha van **legalább egy konkrét** átmenet a tartalmazott konkrét állapotok között (azaz **egzisztenciális** tulajdonságú az absztrakció)
 - SMT megoldóval számítható (konkrét állapotok felsorolása nélkül)
 - $(l, b_1, \dots, b_m)$ és $(l', b'_1, \dots, b'_m)$ között létezik átmenet, ha:
 - $\exists op: (l, op, l') \in G$,
azaz van **op** művelettel él a két hely között a CFA-ban, és
 - $p(b_1) \wedge \dots \wedge p(b_m) \wedge op \wedge p(b'_1) \wedge \dots \wedge p(b'_m)$ kielégíthető

Predikátumabsztrakció

■ Példa

$P = \{(x \leq 5)\}$



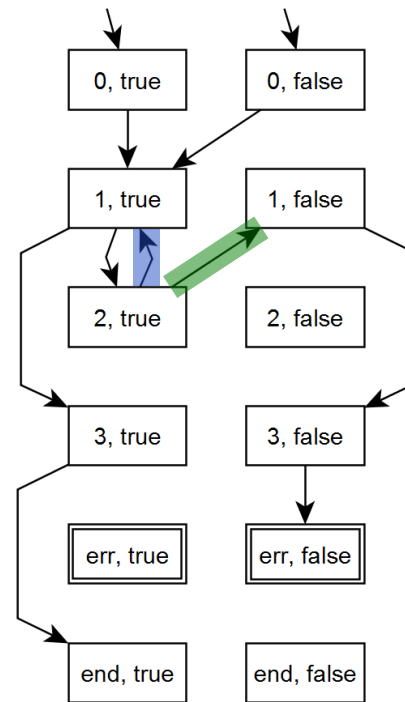
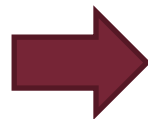
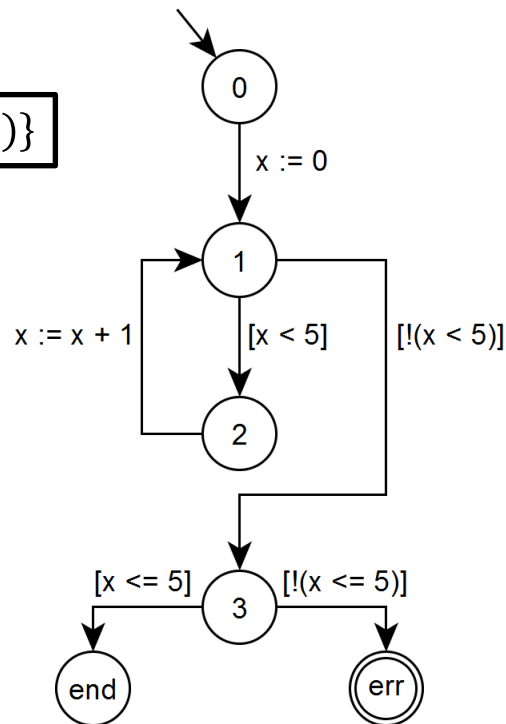
0, true	0, false
1, true	1, false
2, true	2, false
3, true	3, false
err, true	err, false
end, true	end, false

- 6 vezérlési hely, 1 predikátum $\rightarrow 6 \cdot 2^1 = 12$ absztrakt állapot

Predikátumabsztrakció

■ Példa

$$P = \{(x \leq 5)\}$$



■ Átmenet példák

○ $(2, \text{true}) \rightarrow (1, \text{true})$

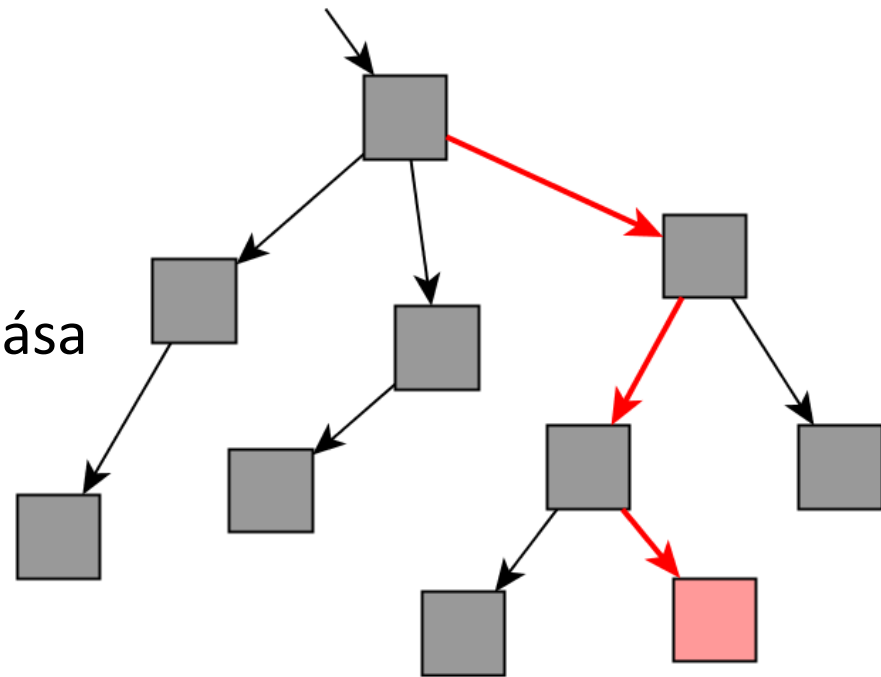
- $(2, x := x + 1, 1) \in G$ és $(x \leq 5) \wedge (x' = x + 1) \wedge (x' \leq 5)$ kielégíthető, pl. $x = 0, x' = 1$

○ $(2, \text{true}) \rightarrow (1, \text{false})$

- $(2, x := x + 1, 1) \in G$ és $(x \leq 5) \wedge (x' = x + 1) \wedge \neg(x' \leq 5)$ kielégíthető, pl. $x = 5, x' = 6$

Modellellenőrzés

- Absztrakt állapottér **bejárása**
 - Valamilyen **keresési stratégiával**, pl. DFS, BFS
 - Hibaállapot keresése
- Optimalizációk
 - „**On-the-fly**”
 - Absztrakt állapotok kiszámítása keresés közben
 - **Inkrementális**
 - Absztrakció finomítás után a változatlan részeket nem kell újra bejárni



Modellellenőrzés

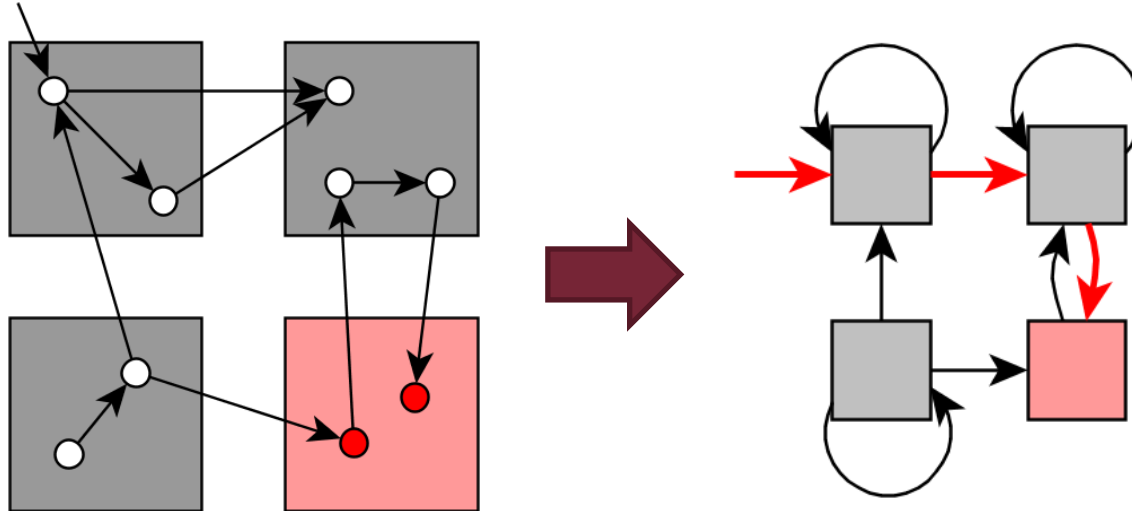
■ Egzisztenciális absztrakció tulajdonságai

○ Felülbecsli az eredeti modell viselkedéseit

- Minden konkrét útvonalhoz van megfelelő absztrakt útvonal
- Ha **nincs** absztrakt útvonal a hibaállapothoz: **nincs** konkrét útvonal sem
- Ha **van** absztrakt útvonal a hibaállapothoz: **nem biztos, hogy van** konkrét útvonal is

○ Tehát: Az **absztrakt ellenpéldát** ellenőrizni kell

- Van-e ennek megfelelő konkrét útvonal?

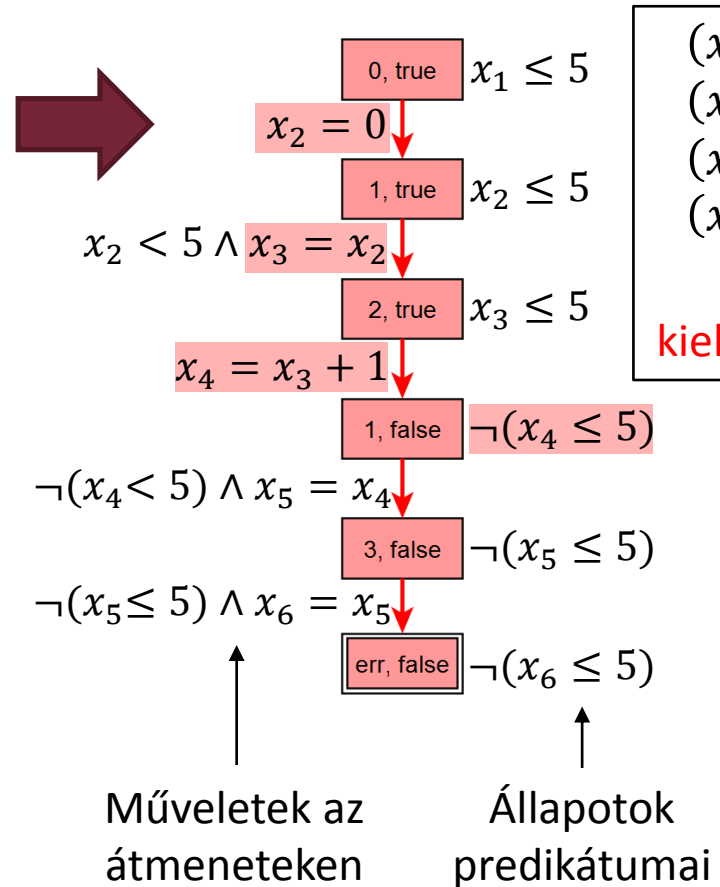
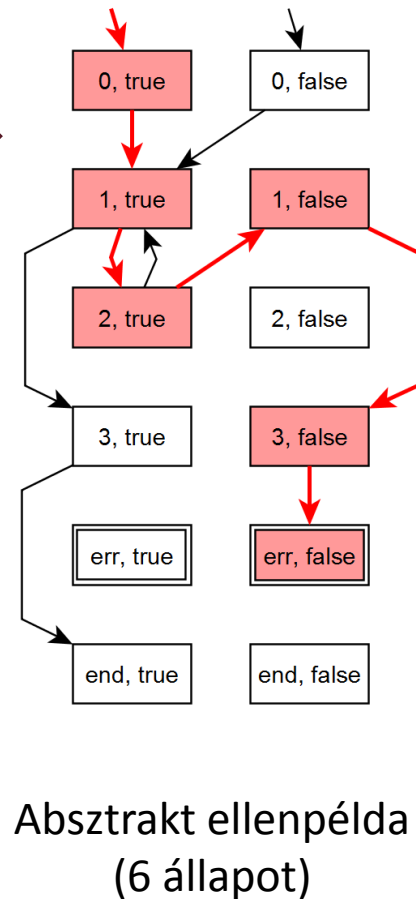
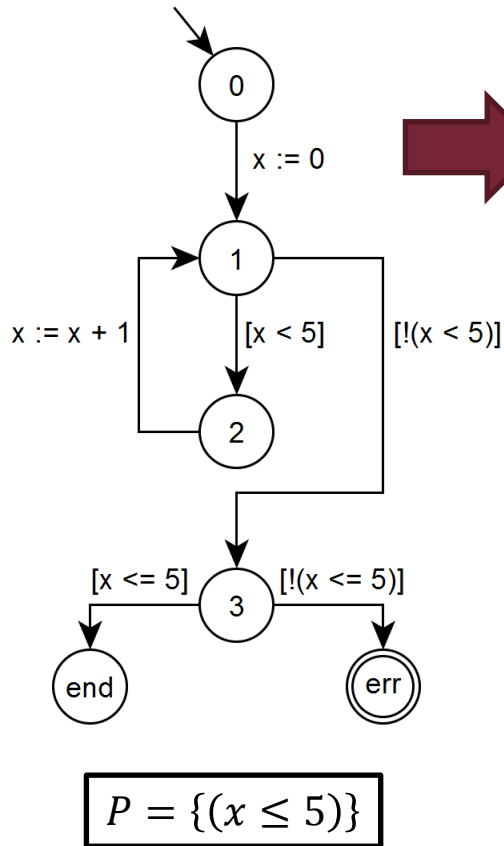


Absztrakt ellenpélda

- Absztrakt ellenpélda alakja
 - Vezérlési helyek és predikátumok sorozata
 - $(l_1, b_{1,1}, \dots, b_{1,m}), (l_2, b_{2,1}, \dots, b_{2,m}), \dots, (l_n, b_{n,1}, \dots, b_{n,m})$
- Konkrét útvonal keresése: konkrét állapottér egy részének bejárása
 - Absztrakt ellenpélda által vezérelve (mit kell bejárni)
 - SMT megoldó segítségével
 - Korlátos modellellenőrzéshez hasonlóan
 - Predikátumabsztrakciónál egy átmenetre bemutatott módszer általánosítása n lépésre
- Ha létezik konkrét útvonal $\rightarrow$ konkrét modell is hibás
- Ha nem létezik konkrét útvonal $\rightarrow$ hamis ellenpélda

Absztrakt ellenpélda

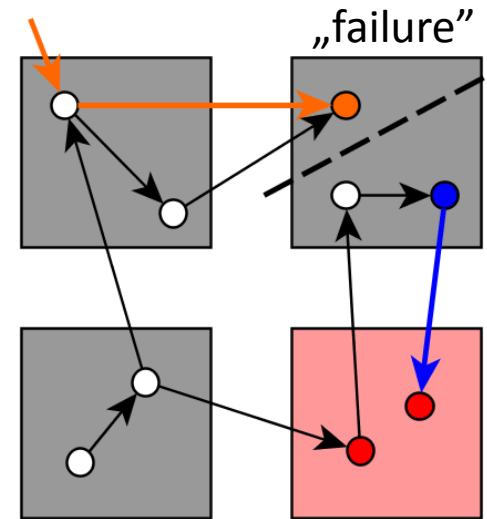
■ Példa



$(x_1 = 3)$
 $(x_2 = 0)$
 $(x_3 = 0)$
 $(x_4 = ?)$
Nem kielégíthető

Hamis ellenpélda

- „Failure” állapot: Eddig az állapotig van út és onnan tovább is, de ezek a konkrét modellben **elkülönülő utak**
- Konkrét állapotok csoportosítása a „failure” absztrakt állapotban
 - D = “Dead-end”: elérhető
 - B = “Bad”: következő állapotra lép
 - IR = “Irrelevant”: többi
- Hamis ellenpélda oka
 - Predikátumhalmaz nem különbözteti meg D-t és B-t



Absztrakciófinomítás

■ Cél: hamis ellenpélda kiküszöbölése

- Bővebb predikátumhalmaz (finomabb absztrakció)

- **D** és **B** szétválasztása

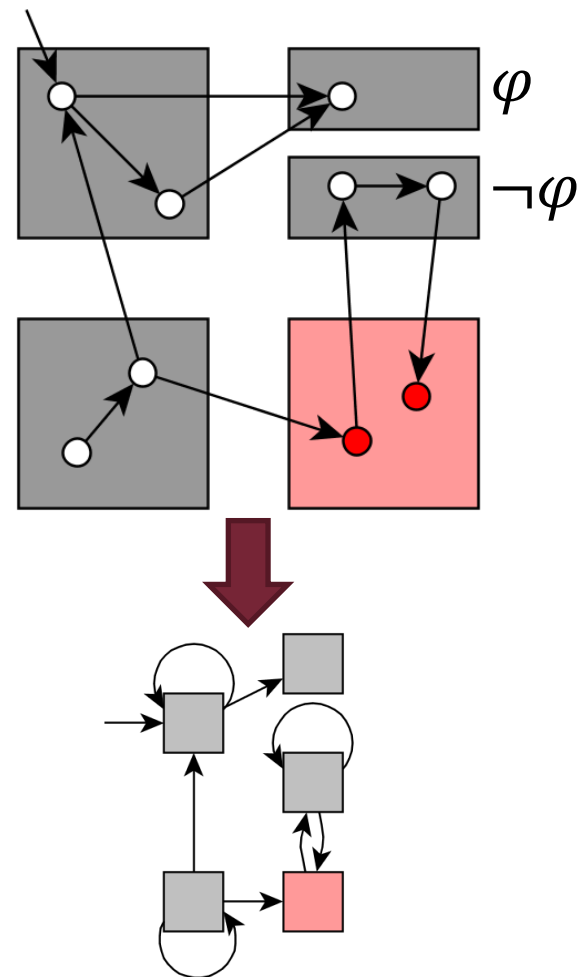
- Konkrét állapotok felsorolása nélkül kell
- **D** út és **B** leírása formulákkal
- SMT megoldó képes egy φ formulát generálni, ami szétválasztja ezeket (egyiken igaz, a másikon nem), ez az ún. *interpoláció*

- $P \cup \{\varphi\}$ predikátumhalmaz esetén ez a hamis ellenpélda **megszűnik**

- Sőt, φ predikátumot elég csak a „failure” állapotban alkalmazni („lusta” absztrakció)

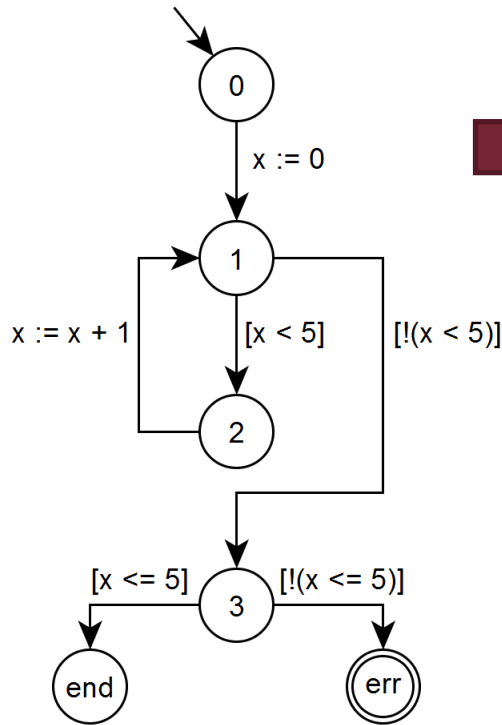
■ További hamis ellenpéldák

- További finomítások, újabb predikátumok



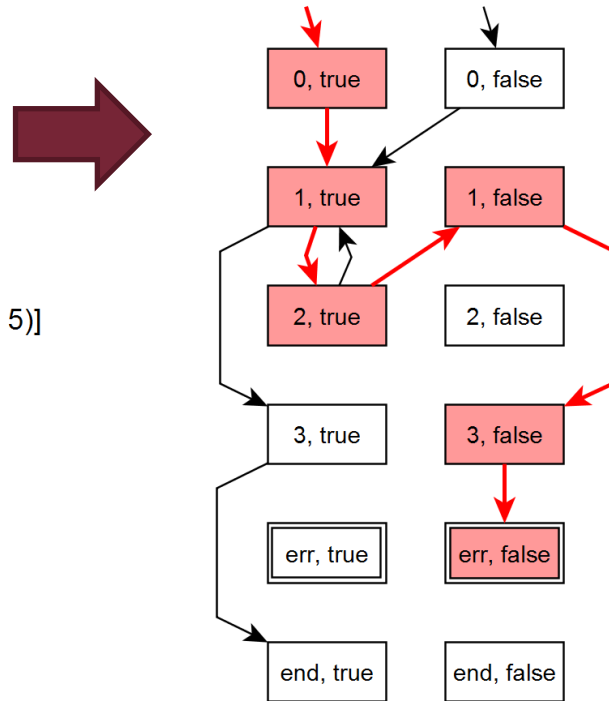
Absztrakciófinomítás

■ Példa

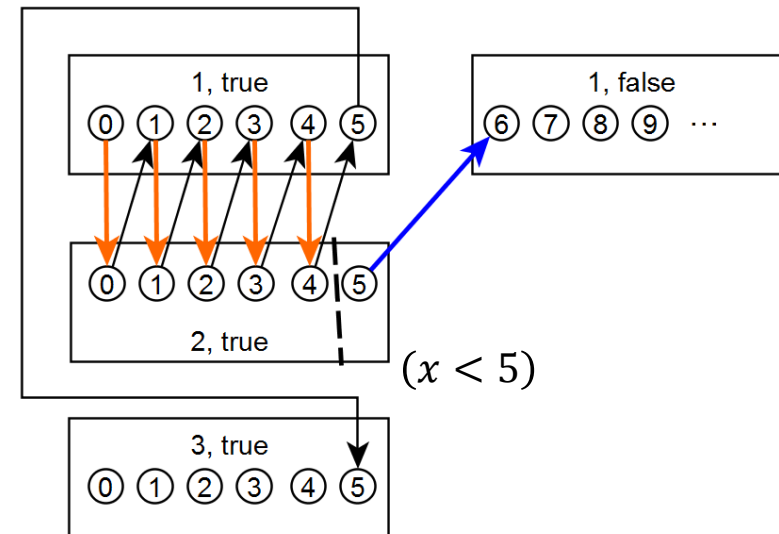


$P = \{(x \leq 5)\}$

Szétválasztás: $(x < 5)$ predikátum

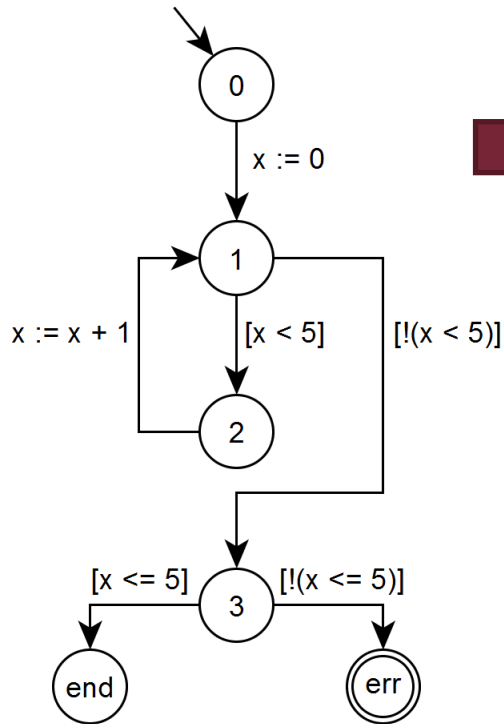


$(x_1 = 3)$
 $(x_2 = 0)$
 $(x_3 = 0)$
 $(x_4 = ?)$



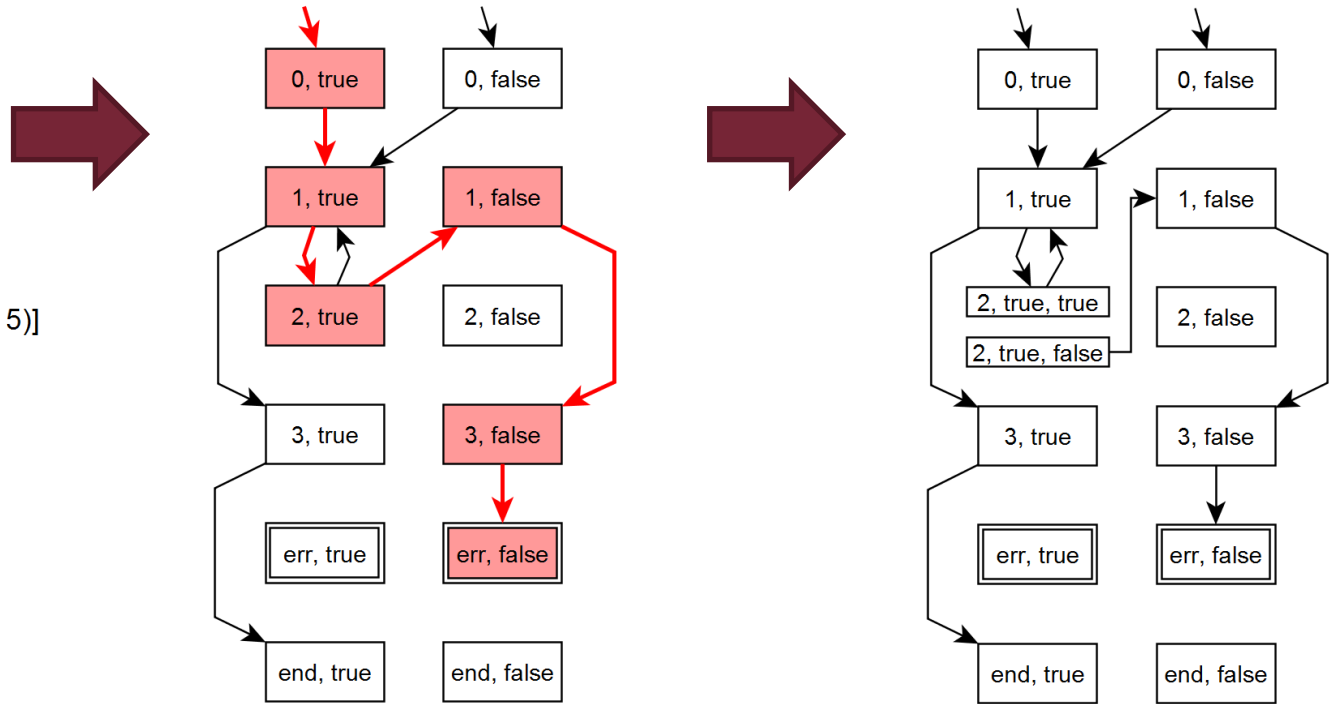
Absztrakciófinomítás

■ Példa



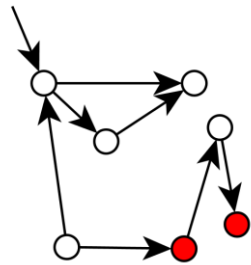
$P = \{(x \leq 5)\}$

Szétválasztás: $(x < 5)$ predikátum

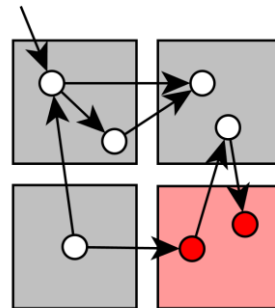


$P = \{(x \leq 5), (x < 5)\}$

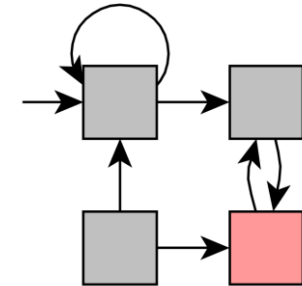
CEGAR – Összefoglaló



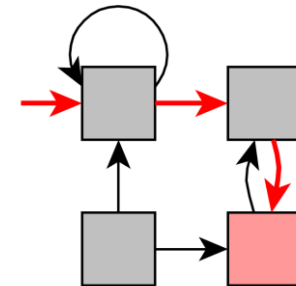
Konkrét állapottér



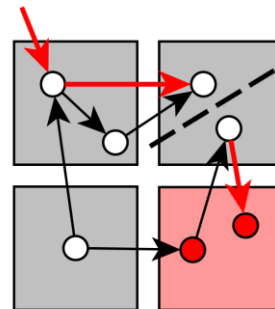
Absztrakció



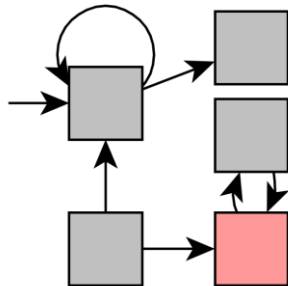
Absztrakt állapottér



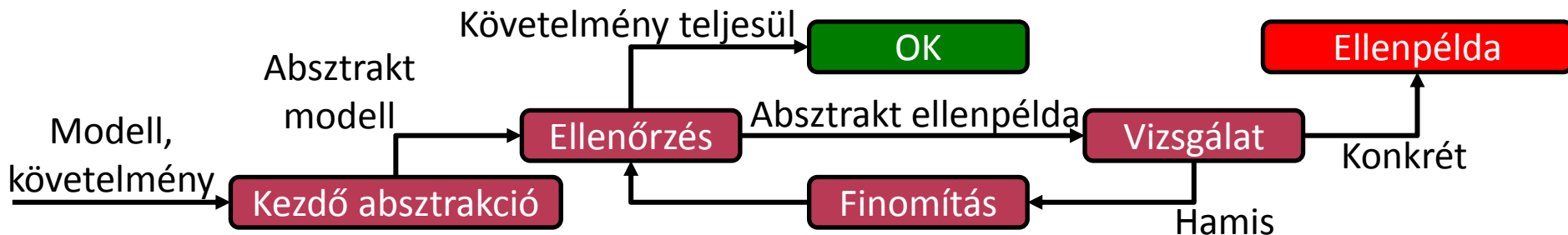
Absztrakt ellenpélda



Hamis ellenpélda

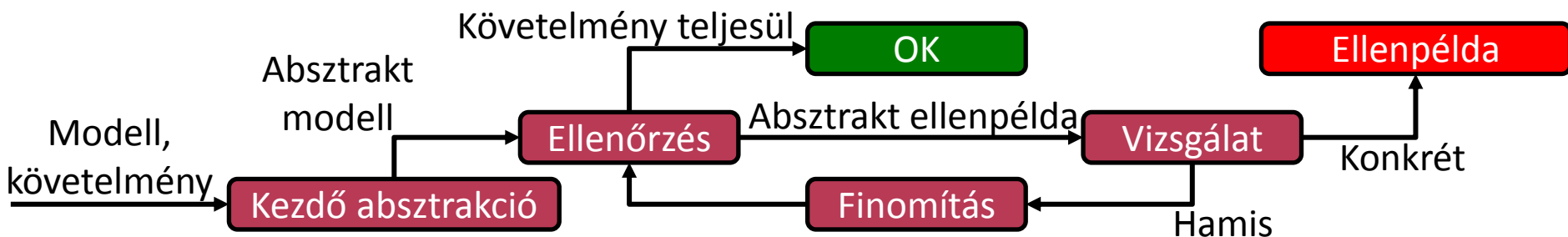


Finomított absztrakció



A CEGAR algoritmusról

- Counterexample-Guided Abstraction Refinement (CEGAR)
 - Automatikus módszer
 - Minden lépés automatikusan működik
 - Nem szükséges a formális módszerek részletes ismerete
 - Ki mondja meg a **kezdeti** predikátumhalmazt?
 - Lehet akár üres halmaz is (a CEGAR algoritmus fogja az absztrakt állapotokat szétválasztani)
 - Programban szereplő feltételes utasítások alapján
 - Egyéb heurisztikák alapján



ESZKÖZÖK

■ SLAM2

- Static Driver Verifier Research Platform (SDVRP) része
- Felépítése
 - Driver **C kód**: vizsgált komponens
 - Platform modell: környezet leírása
 - Ellenőrzés: **API használati szabályok** betartása
- Működése
 - Boole program előállítása **predikátumabsztrakcióval**
 - Szimbolikus modellellenőrzés: BEBOP eszköz
 - **CEGAR** ciklus
- research.microsoft.com/en-us/projects/slam/

Eszközök: BLAST

■ BLAST

- Berkeley Lazy Abstraction Software Verification Tool
- Bemenet: **C program** + követelmény (BLAST Query Language)
- **Predikátumabsztrakció**
 - Absztrakt elérhetőségi fa építése
- Finomítás: új predikátum(ok) interpolációval
 - „**Lusta**” **absztrakció**: új predikátum alkalmazása lokálisan
- Korlátok: szorzás, bitműveletek, túlcsordulás
- mtc.epfl.ch/software-tools/blast/index-epfl.php

Eszközök: CPAchecker

■ CPAchecker

- The Configurable Software-Verification Platform
- Bemenet: **C program** + specifikáció
 - Ellenőrzések: Assertion, error címke, deadlock, null dereference, ...
- Nagymértékben **konfigurálható**
 - Többféle absztrakció (nem csak predikátumok)
 - Absztrakt ellenpélda több prefixét tekinti
 - Többféle lehetséges finomítás közül választ (finomítási stratégia)
- cpachecker.sosy-lab.org/

■ Theta

- Általános, moduláris, konfigurálható verifikációs keretrendszer
- BME MIT saját fejlesztés
- **Általános**: különböző formális modellek támogatása
 - Tranzíciós rendszer, control flow automata, időzített automata
- **Moduláris**: újrafelhasználható, kombinálható modulok
 - Absztrakciók, finomítások, interpoláció, ...
- **Konfigurálható**: különböző algoritmusok és stratégiák
- github.com/FTSRG/theta

Eszközök: SV-COMP verseny

- **Competition** on Software Verification 2017 (SV-COMP)
 - sv-comp.sosy-lab.org/2017/
 - **32 eszköz, 8908 verifikációs feladat** (program + követelmény)
 - Program kategóriák
 - Arrays (ArraysReach, ArraysMemSafety)
 - Bit Vectors (BitVectorsReach, Overflows)
 - Heap Data Structures (HeapReach, HeapMemSafety)
 - Floats
 - Integers and Control Flow (ControlFlow, Simple, ECA, Loops, Recursive, ProductLines, Sequentialized)
 - Termination
 - Concurrency
 - Software Systems (DeviceDriversLinux64, BusyBox)
 - Adott kategóriához megtalálható a hatékony eszköz

ÖSSZEFOGLALÁS

Összefoglalás

■ Szoftver-modellellenőrzés

○ „Gombnyomásos” módszer

- Kezelendő probléma: állapottér robbanás (a változók miatt)

○ Megoldás: absztrakció

- Vezérlési hely + predikátumok a változókon
- Egzisztenciális absztrakció (hamis ellenpélda lehet)

○ CEGAR: megfelelő absztrakció automatikus előállítása

1. Kezdő absztrakt modell
2. Modellellenőrzés
3. Ellenpélda vizsgálata
4. Absztrakció finomítása

○ Eszközök