

Введение в моделирование и верификацию аппаратных и программных систем

Лекция 13: Свидетельства и контрпримеры. Абстракция.

Борис Юрьевич Конев

konev@liverpool.ac.uk

Liverpool University

Октябрь-Ноябрь 2007

Свидетельства и контрпримеры

Рассмотрим два подъязыка CTL:

- ACTL: формулы построенные только при помощи **AX**, **AF**, **AG**, **AU** (и отрицания — только “внутри” временных операторов)
- ECTL: формулы построенные только при помощи **EX**, **EF**, **EG**, **EU** (и отрицания — только “внутри” временных операторов)

Если φ — ACTL-формула, то $\neg\varphi$ эквивалентна ECTL-формуле, и наоборот

- Для ECTL формул можно предъявлять свидетельства,
- для ACTL — контрпримеры

Поиск свидетельства для $\mathbf{EG}\psi$

Т.к. $\mathbf{EG}\psi$ — наибольшая неподвижная точка функции

$$H(Z) = [\psi_1] \cap \mathbf{EX}Z$$

если для начальной вершины q_0 мы имеем

$$(S, q_0) \models \mathbf{EG}\psi$$

то найдется последовательность вершин q_0, q_1, \dots т.ч.

$$(S, q_i) \models \mathbf{EG}\psi \text{ и } (q_i, q_{i+1}) \in T.$$

Так как S конечна, найдутся $i < j : q_i = q_j$

Тогда $q_0, \dots, q_{i-1}, (q_i, \dots, q_j)^*$ — свидетельство для $\mathbf{EG}\psi$.

Поиск свидетельства для $\mathbf{EX}\psi$, $\mathbf{EF}\psi$, $\mathbf{E}\psi_1\mathbf{U}\psi_2$

- Для $\mathbf{EX}\psi$, $\mathbf{EF}\psi$ построить путь q_0, q_1, \dots, q_f , такой, что

$$(S, q_f) \models \psi$$

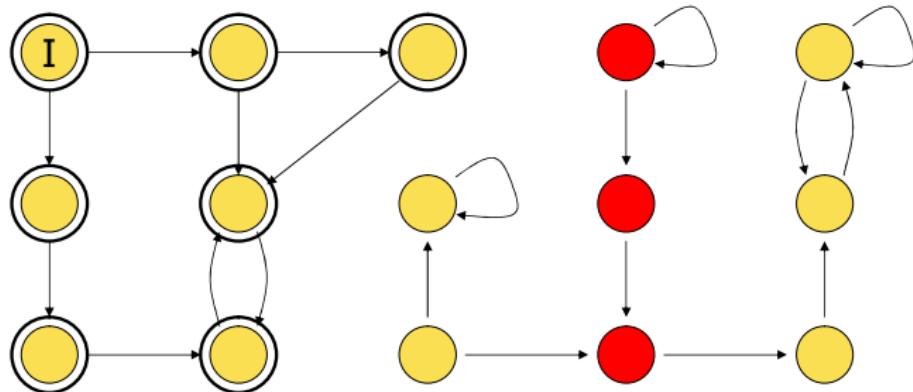
- Для $\mathbf{E}\psi_1\mathbf{U}\psi_2$ построить путь q_0, q_1, \dots, q_f , такой, что

$$(S, q_f) \models \psi_2 \text{ и } (S, q_i) \models \psi_1 \text{ для } 1 \leq i < f$$

Контрпримеры и свидетельства

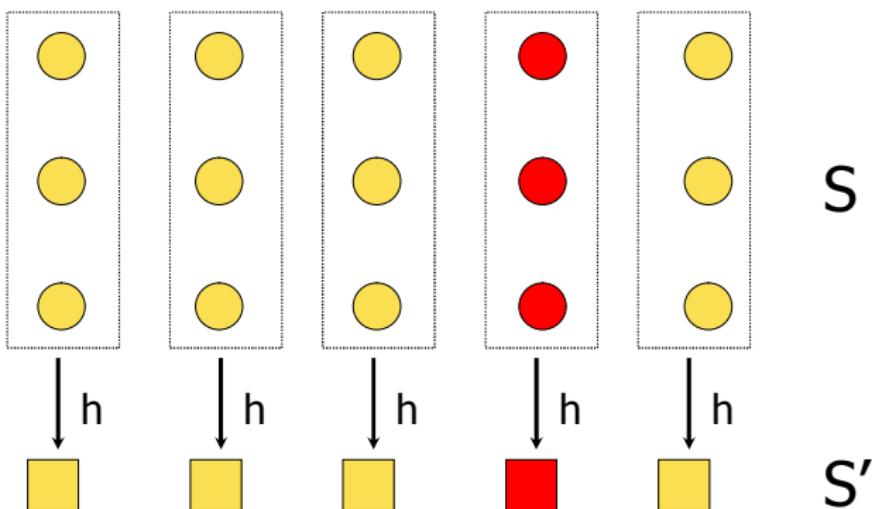
- Контрпример для ACTL формулы φ == свидетельство для ECTL формулы $\neg\varphi$

Контрпример для $\text{AG}\varphi$

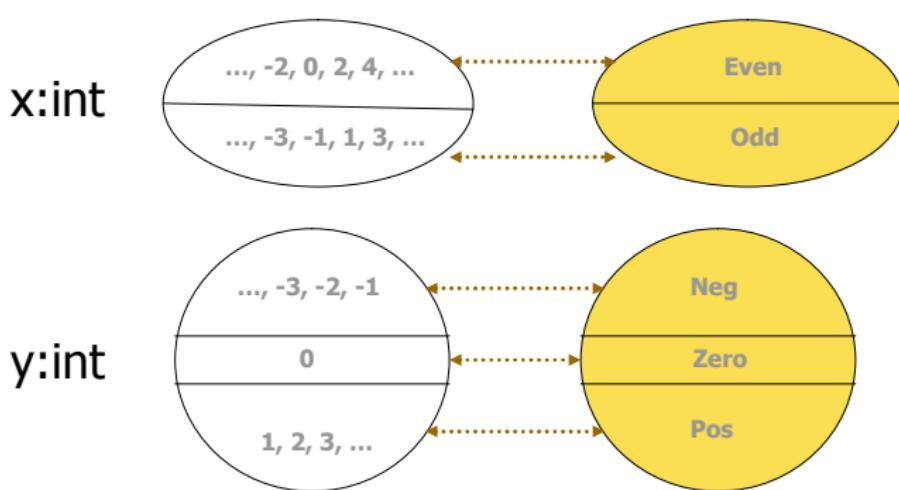


- Система переходов слишком велика, чтобы найти контрпример
- Абстракция
 - Сканируем некоторые состояния

Абстракция данных



Пример



Популярный подход

- Разобьем переменные на два множества
 - Видимые (V)
 - Невидимые (I)
- Сгруппируем состояния с идентичными видимыми переменными
- Например,

The diagram illustrates a transformation from a larger state space to a smaller one. On the left, there is a 4x4 grid of binary values (0 or 1) representing states. The columns are labeled x_1 , x_2 , x_3 , and x_4 . The rows show four distinct configurations: (0, 0, 0, 0), (0, 0, 0, 1), (0, 0, 1, 0), and (0, 0, 1, 1). An arrow points from this 4x4 grid to a smaller 2x2 grid on the right, which also has columns labeled x_1 and x_2 . This smaller grid contains only two rows, corresponding to the first two configurations of the original grid, effectively grouping states with identical visible variable values.

x_1	x_2	x_3	x_4
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1

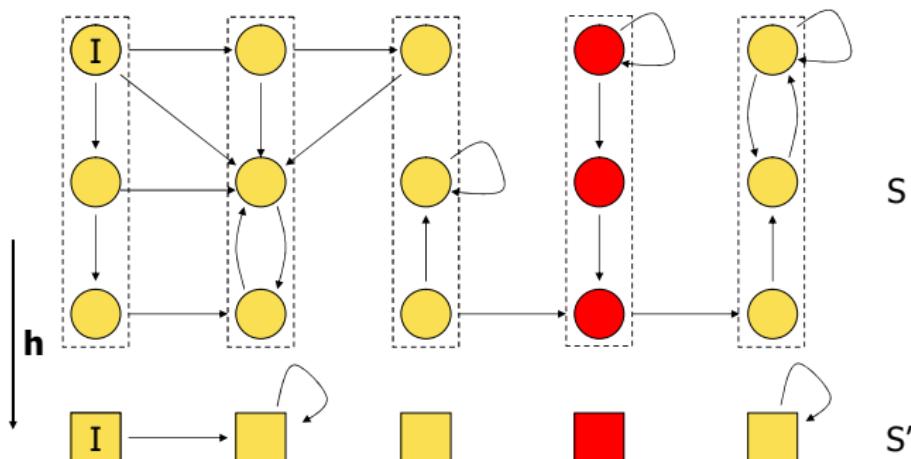
→

x_1	x_2
0	0

- Как определить отношение переходов?

Универсальная абстракция

- Перейти из абстрактного состояния, если для всех конкретных состояний, представленных данным абстрактным состоянием, можно сделать переход.

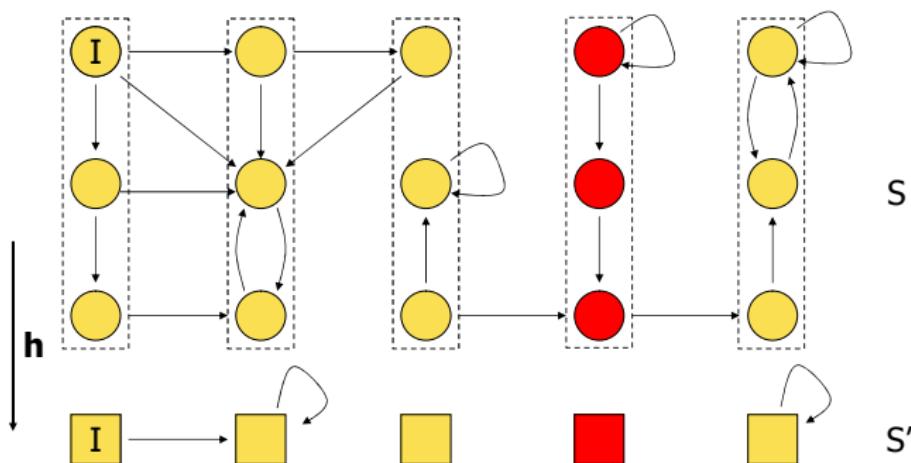


Универсальная абстракция и ECTL формулы

- S_A — универсальная абстракция системы S
- φ — ECTL формула

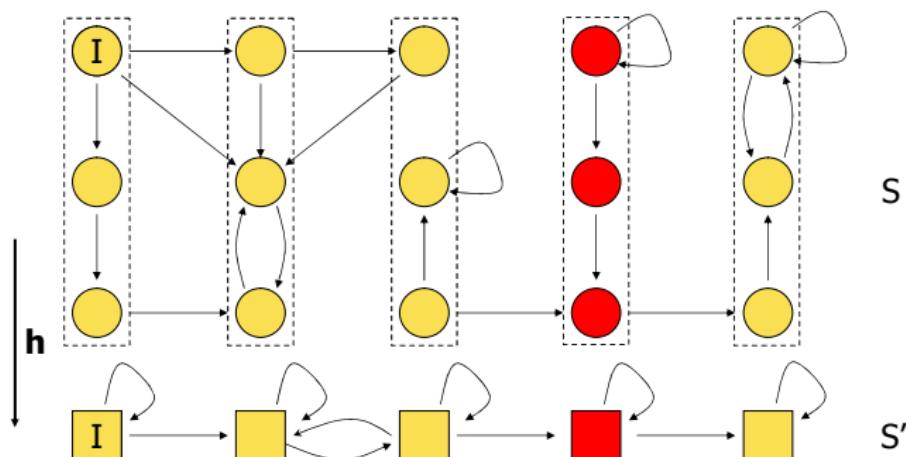
$$S \not\models \varphi \implies S_A \not\models \varphi$$

- Пойск ошибок



Экзистенциальная абстракция

- Перейти из абстрактного состояния, если можно перейти из **хотя бы одного** конкретного состояние, представленного данным абстрактным состоянием

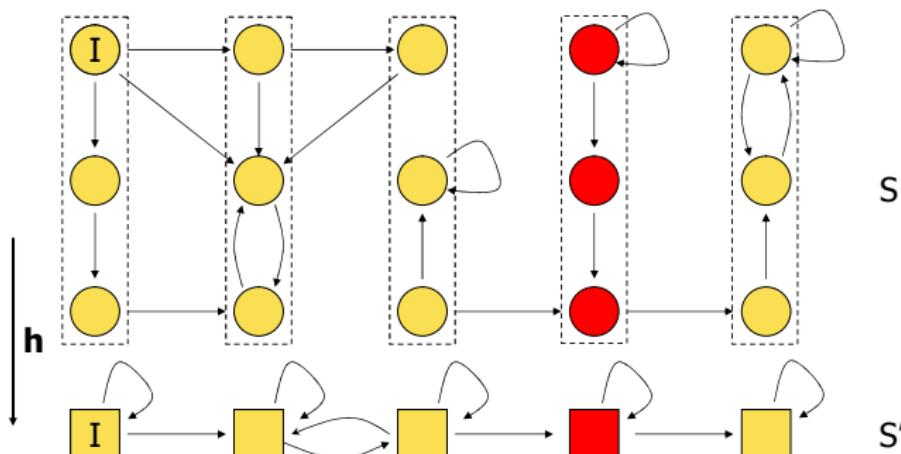


Экзистенциальная абстракция и ACTL формулы

- S_A — экзистенциальная абстракция системы S
- φ — ACTL формула

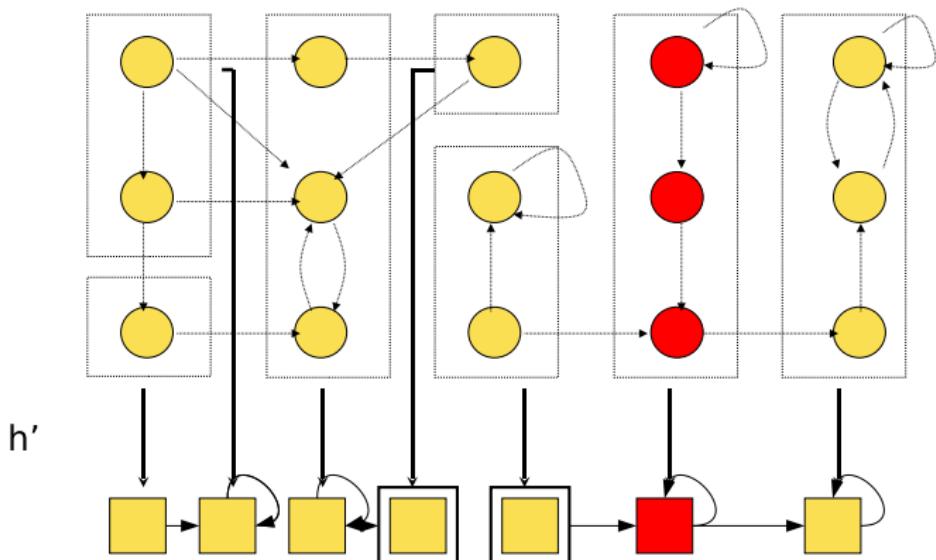
$$S_A \models \varphi \implies S \models \varphi$$

• Верификация



- Проблема ложных контрпримеров

Уточнение абстракции



Автоматическое уточнение абстракции

Counter Example Guided Abstraction Refinement (CEGAR)

- ➊ Построить абстрактную систему S_A
- ➋ Проверить ACTL свойство φ
 - Если $S_A \models \varphi$, конкретная система обладает свойством φ .
 - Иначе, найти контрпример (путь π_A в S_A , ведущий в $\neg\varphi$)
 - Если существует путь π в S , $H(\pi) = \pi_A$, то π — контрпример в конкретной системе
 - Иначе, уточнить абстракцию и goto 2

Видимые и невидимые переменные

- Проверим, ложен ли контрпример

The diagram illustrates a reduction process. On the left is a 4x4 matrix with columns labeled x_1, x_2, x_3, x_4 . The rows contain the following values:

x_1	x_2	x_3	x_4
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1

An arrow points from this matrix to a smaller 2x2 matrix on the right, also labeled x_1, x_2 , with the following values:

x_1	x_2
0	0

- Если ложен, сделаем какую-то из невидимых переменных видимой

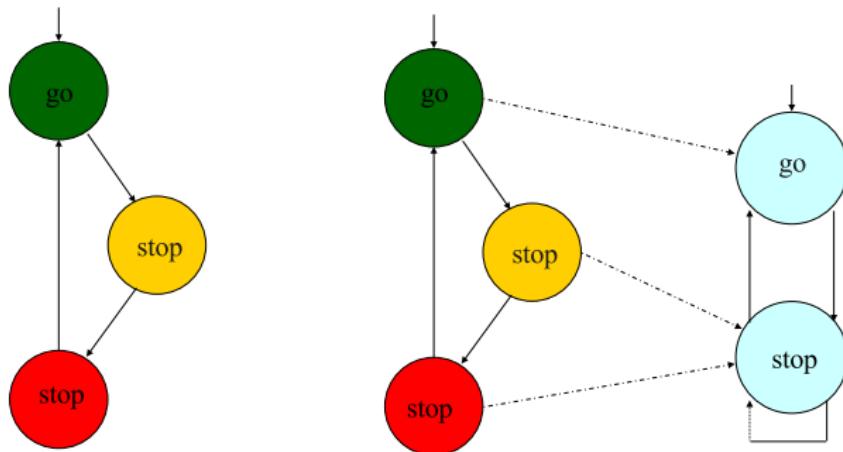
The diagram illustrates a reduction process. On the left is a 4x4 matrix with columns labeled x_1, x_2, x_3, x_4 . The rows contain the following values:

x_1	x_2	x_3	x_4
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1

An arrow points from this matrix to a smaller 3x3 matrix on the right, labeled x_1, x_2, x_3 , with the following values:

x_1	x_2	x_3
0	0	0
0	0	1

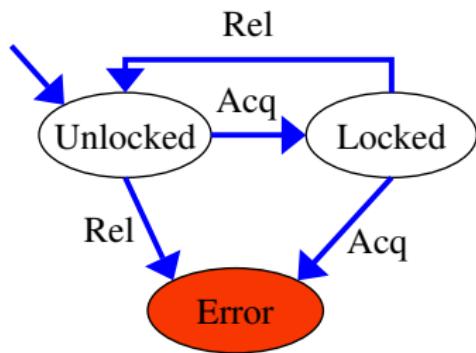
Пример



- **AG**($go \rightarrow AX stop$)
- **AGAF** go
 - $go, stop, stop, stop, stop, stop \dots$

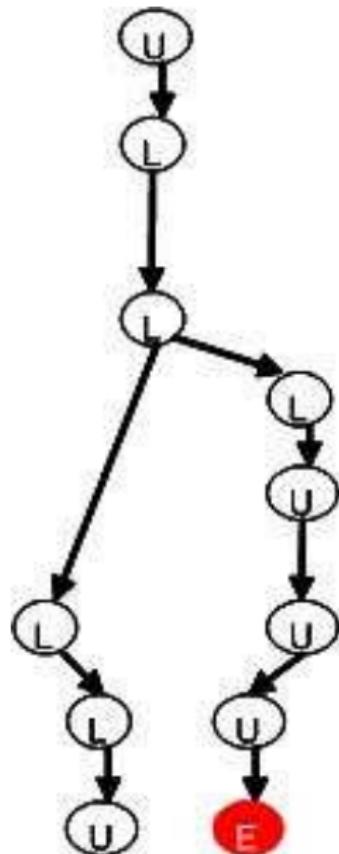
Абстракция предикатов

Помните?



```
do {  
    KeAcquireSpinLock();  
  
    nPacketsOld = nPackets;  
  
    if(request){  
        request = request->Next;  
        KeReleaseSpinLock();  
        nPackets++;  
    }  
} while (nPackets != nPacketsOld);  
KeReleaseSpinLock();
```

Булева программа



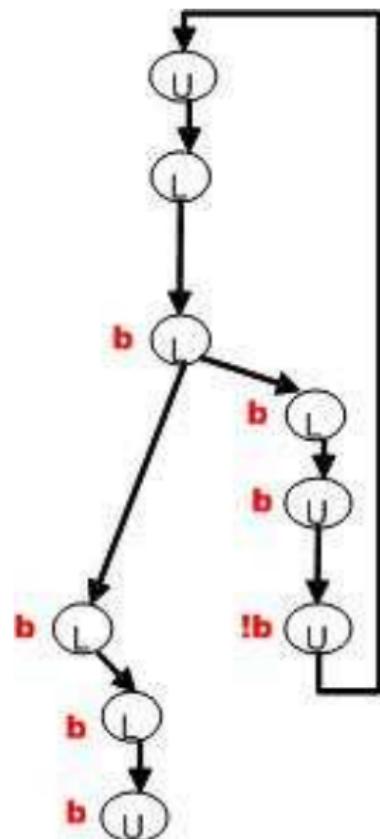
```
do {  
    KeAcquireSpinLock();  
  
    if (*) {  
        KeReleaseSpinLock();  
    }  
} while (*);  
KeReleaseSpinLock();
```

Ложный контрпример

Контрпример ложен из-за того, что условие выхода из цикла `nPackets == nPacketsOld`, но `KeReleaseSpinLock` может исполнится только вместе с `nPackets++`

- Введем булеву переменную `b`
`b = (nPackets == nPacketsOld)`
- проследим, как операторы меняют `b`

Уточненная булева программа



```
do {
    KeAcquireSpinLock();

    b := true;

    if(*) {

        KeReleaseSpinLock();
        b := b? false : *;
    }
} while ( !b );

KeReleaseSpinLock();
```