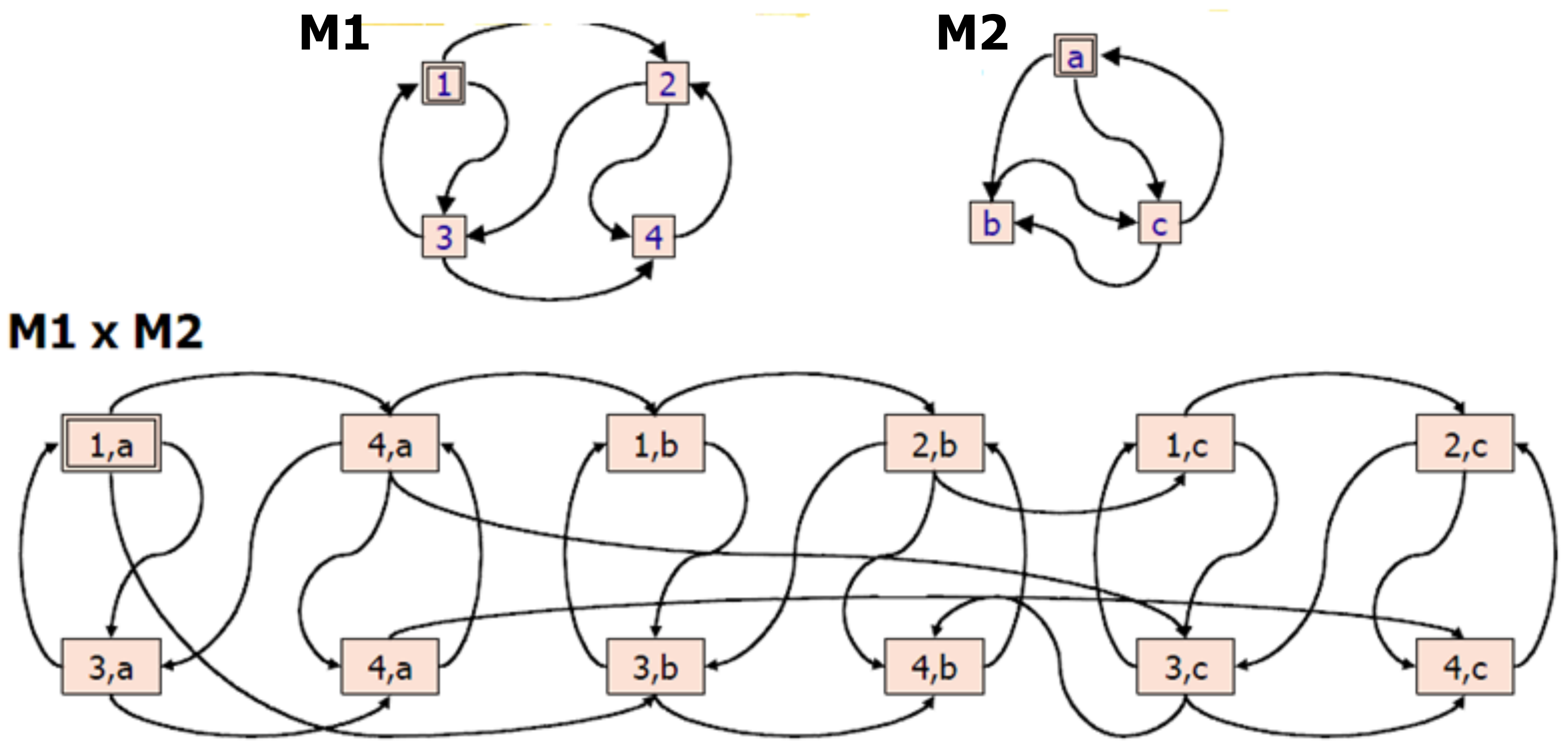
Лекция

Символьная проверка моделей

Экспоненциальный рост числа состояний параллельной системы



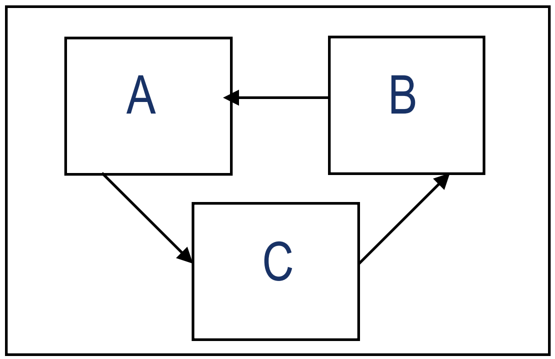
- Число состояний растет экспоненциально при увеличении числа параллельно работающих компонент

Верификация реактивных систем: State explosion problem

Классические алгоритмы верификации реактивных систем могут работать только с игрушечными системами – число состояний реальных систем (даже представленных моделями с конечным числом состояний) очень велико

Обычная техника Model checking работает с системами ~10^6 состояний – представление каждого состояния требует ~10^2 байт

Пример:
три процесса,
три канала связи



Простая система, а число состояний - миллионы миллиардов!!

Каждая подсистема – 4 состояния + 1 целая переменная (short, 1 байт) + каналы для передачи целых (байт) значений.
Пусть для простоты очередь каждом из в 3-х каналах не больше 1 сообщения

Состояние подсистемы – < *состояние*, *значение переменной*>

Всего $(4 \times 2^8)^3 \times (2^8)^3 = 2^{54} \cong 10^{16}$ глобальных состояний системы - **State Explosion Problem**. Система в своей жизни проходит ничтожную долю своих возможных состояний, но мы не знаем, какие именно!! Число частиц во Вселенной ~10^{78}

BDD: борьба с проблемой взрыва числа состояний

Model Checking – эффективная техника верификации, но есть недостатки. Число состояний при введении дополнительных параметров и/или компонент исследуемой системы растет экспоненциально: “State explosion problem”

Конечные математические структуры (множества, отношения, …) и операции над ними могут быть представлены логическими функциями и булевыми операциями над этими функциями. В свою очередь, БФ можно экономно представить в BDD.

BDD можно использовать в любых алгоритмах над конечными структурами и алгоритмы эти получаются очень эффективными

В 1992 г. все это было осознано в Carnegi Mellon University – были представлены методы задания структуры Крипке и алгоритмы (Model checking) - анализа темпоральных свойств структур Крипке с помощью BDD. Так был разработан метод “СИМВОЛЬНОЙ ВЕРИФИКАЦИИ”

J.Burch, E.Clarke, K.McMillan et.al. Symbolic model checking: 10^{20} states and beyond. Information and Computation, v.98, N2,1992

Первая же публикация: повышение эффективности Model checking в 10^{14} раз

Верификация систем с числом состояний 10^{20}

Явные алгоритмы model checking: до 10^6 состояний, ~ 100 состояний в секунду

- Пусть для запоминания одного состояния нужно только 10 байт
- Тогда запоминание 10^{20} состояний требует *тысячу миллионов терабайт*
- Пусть скорость перебора при выполнении алгоритмов верификации обычная ~ сотня состояний в секунду
- Тогда перебор 10^{20} состояний с помощью явных (обычных) алгоритмов *model checking* потребует сотен миллиардов лет
- ВЫВОД: без использования неявных (СИМВОЛЬНЫХ) методов хранения данных и манипулирования ими с помощью BDD верификация реальных систем была бы невозможна
- MODEL CHECKING позволяет оперировать множествами с числом элементов до ~10^{300} и даже более!!! (Baier, Katoen. Principles of Model checking, 2008)

Характеристические функции и их представление в BDD

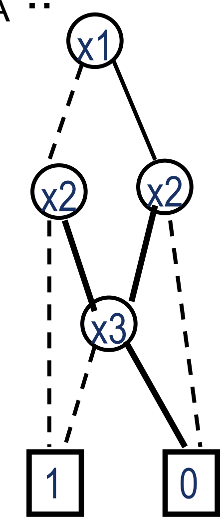
Пусть S={a0, ..., a7} и A={a2, a4, a5} – подмножество S. Закодируем элементы S двоичными кодами. Тогда A = { 010, 100, 101 }

Обозначим χ_A характеристическую функцию множества A. Она равна 1 на наборах, кодирующих элементы из A, т.е. на { 010, 100, 101 }

Пусть x_1, x_2, x_3 – разряды кодировки.

Тогда $\chi_A = \neg x_1 x_2 \neg x_3 \vee x_1 \neg x_2 \neg x_3 \vee x_1 \neg x_2 x_3$
задает A = { 010, 100, 101 }

χ_A :



Итак, подмножества конечного множества можно задавать логической формулой, представляющей характеристическую функцию

Будем писать A = { 010, 100, 101 } (x_1, x_2, x_3), чтобы показать переменные кодировки и их порядок

Можем также строить характеристическую функцию ОТНОШЕНИЙ на конечных множествах

Операции над множествами с помощью BDD

Нульарные операции (константы):

- полное множество: χ_S = True
- пустое множество: $\chi_\varnothing$ = False

Унарная операция:

- дополнение множества: $\chi_{S-Q} = \neg\chi_Q$

Бинарные операции:

- пересечение множеств: $\chi_{P\cap Q} = \chi_P \wedge \chi_Q$
- объединение множеств: $\chi_{P\cup Q} = \chi_P \vee \chi_Q$
- разность множеств: $\chi_{P-Q} = \chi_P \wedge \neg\chi_Q$

ВСЕ операции над множествами можно выразить через булевы операции над характеристическими булевыми функциями:

Приложение к Model Checking – темпоральные формулы

Синтаксис: CTL (грамматика):

$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid AX\ \varphi \mid EX\ \varphi \mid AG\ \varphi \mid$
$\quad EG\ \varphi \mid AF\ \varphi \mid EF\ \varphi \mid A\ [\varphi_1 U\ \varphi_2\ ] \mid E[\varphi_1 U \varphi_2\ ]$

State formulas: Каждая CTL формула – это формула состояний, которая в каждом состоянии либо истинна, либо ложна

Поэтому можно считать, что любая CTL формула φ описывает множество состояний, в которых она истинна:
$Q_\varphi = \{ s \in S \mid s \models \varphi \}$, т.е. Q_φ - множество таких состояний из S, на которых формула φ истинна

Для формулы φ логики CTL будем ОПРЕДЕЛЯТЬ характеристическую функцию множеством состояний, на котором φ истинно

Верификация структуры Крипке с 10^{20} состояний

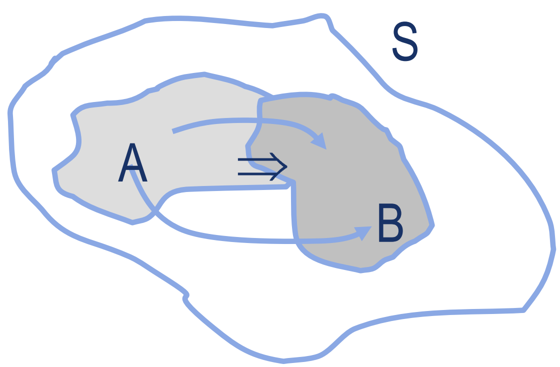
Любое подмножество множества из 10^{20} состояний может быть представлено булевой функцией от 70 переменных ($2^{10} \sim 10^3$, $10^{20} \sim 2^{70}$)

Операции над такими характеристическими булевыми функциями выполняются менее, чем за секунду

При верификации структуры Крипке с числом состояний 10^{20} нужно:
- множество начальных состояний (1 функция от 70 переменных)
- множество переходов (1 функция от 140 переменных)
- для каждого атомарного предиката – множество состояний, в которых этот предикат истинен (1 функция от 70 переменных)
- для каждой подформулы проверяемой темпоральной формулы – множество состояний, в которых эта формула истинна (1 функция от 70 переменных)

Прямой образ для бинарных отношений на множестве

Пусть $A \subseteq S$ – подмножество S,
R – бинарное отношение на S
В какие элементы S можно перейти из A?



Ограничение отношения R на тех начальных элементах R из S, которые принадлежат A:

$$\chi_{A(v)} \;\&\; \chi_{R(v,\, v')}$$

B= Forward Image (A,R)=FI(A,R) - Прямой Образ A относительно R:

$$\chi_{FI(A,R)\,(v)} = \exists v.[\underbrace{\chi_{A\,(v)} \& \;\chi_{R\,(v,\, v')}}_{\text{Переходы из элементов } \in A}]<v/v'>$$

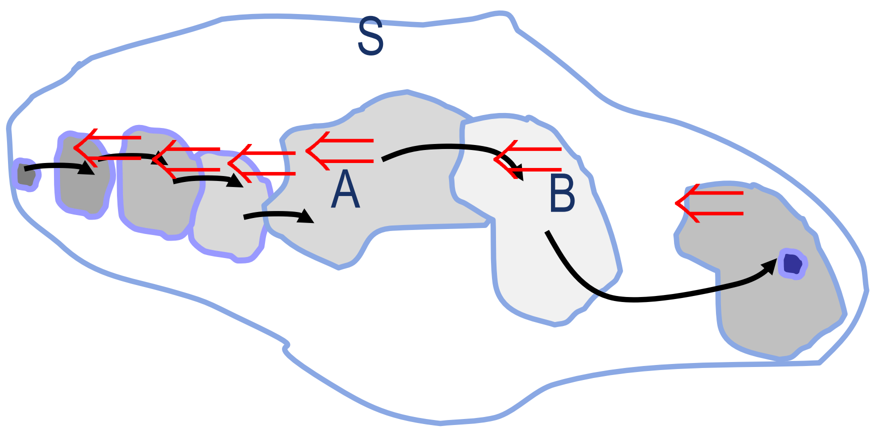
(< v / v' > - это замена переменными v штрихованных значений v')

Чтобы найти множество B всех тех элементов S, которые достижимы за один шаг отношения R из элементов множества A, нужно выполнить две операции с булевыми характеристическими функциями $\chi_{A\,(v)}$ и $\chi_{R\,(v,\, v')}$

Проверка свойств “безопасности” из начального состояния

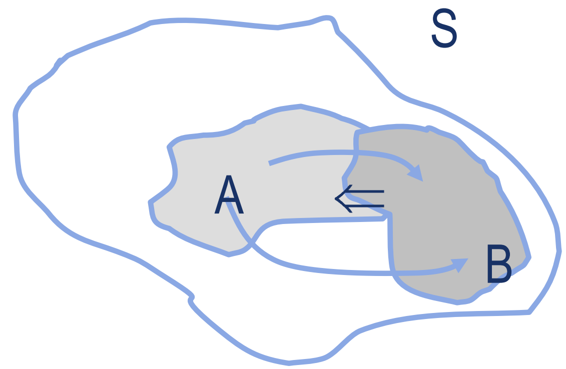
Для выполнения алгоритмов проверки свойств safety (безопасности) необходимо проверить достижимость состояний, в которых выполняется “нечто плохое”, например, блокировка, либо одновременный вход в критическую секцию.

- Операция Forward Image – это один шаг “forward-traversal calculating” транзитивного замыкания отношения перехода между состояниями. Поиск выполняется, начиная с начального состояния



Обратный образ для бинарных отношений на множестве

Пусть $B \subseteq S$ – подмножество S,
R – бинарное отношение на S
Из каких элементов S можно перейти в B?



Ограничение отношения R на тех вторых элементах R из S, которые принадлежат B:

$$\chi_{B\ (v')} \ \& \ \chi_{R\ (v,v')}$$

A= Reverse Image (B,R) = RI(B,R)= обратный образ B относительно отношения R:

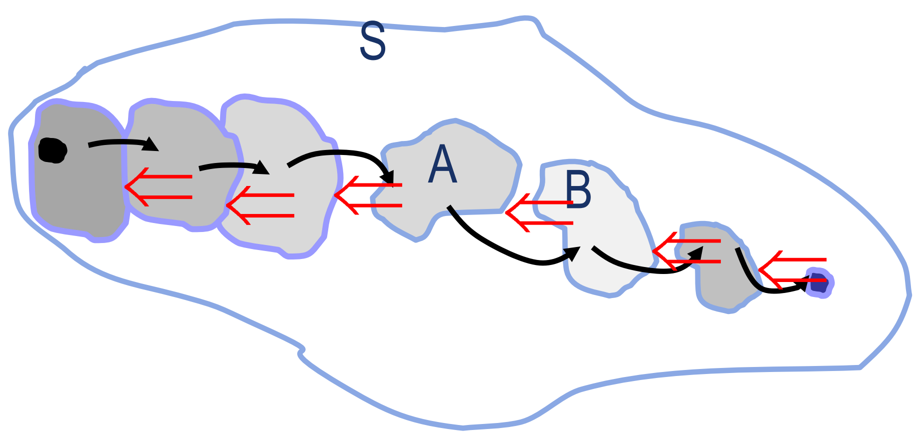
$\chi_{RI(B,R)\ (v)} = \exists v'.(\chi_{B\ (v')} \ \& \ \chi_{R\ (v,v')})$ – выбрасываем все вторые элементы ограничения отношения R на A

Чтобы найти множество A всех тех элементов S, из которых за один шаг отношения R достижимы элементы заданного множества B, нужно выполнить несколько две операции с булевыми характеристическими функциями $\chi_{B\ (v')}$ и $\chi_{R\ (v,v')}$

Проверка свойств “безопасности” из некорректного состояния

Для выполнения алгоритмов проверки свойств safety (безопасности) необходимо проверить достижимость некорректных состояний, в которых выполняется “нечто плохое”, например, блокировка, либо одновременный вход в критическую секцию.

- Операция Backward Image – это один шаг “backward-traversal calculating” транзитивного замыкания отношения перехода между состояниями. Поиск выполняется, начиная с некорректного состояния



BDD породили новый класс алгоритмов

- Основанные на BDD алгоритмы называют символьными (“symbolic”) или неявными (“implicit”). ПОЧЕМУ?
- Символьные
  - для манипулирования объектами и отношениями вводятся дополнительные булевы переменные, которые можно считать “дополнительными” параметрами, или “*символами*”
- Неявные:
  - классические алгоритмы обычно оперируют с явным представлением дискретных объектов, перебирая их последовательно, “один за другим” (explicit representation)
  - алгоритмы, основанные на BDD, работают с *неявным* представлением (implicit representation) конечных множеств и отношений объектов в виде Булевых Функций, представленных в форме BDD
  - Булева формула представляет целое множество объектов (например, состояний), и операция над БФ эквивалентна операции на МНОЖЕСТВЕ
  - размер представления МНОЖЕСТВ растет не линейно, а логарифмически, операции для всех объектов множества выполняются каждая за один шаг для всего множества, а не последовательно для каждого элемента множества

Символьная верификация для логики ветвящегося времени CTL

Задача верификации: Даны структура Крипке К и CTL-формула φ. Найти множество всех таких состояний s, для которых верно: К, s|= φ, т.е. те, для которых выполняется φ. Если начальное состояние принадлежит s, то К |= φ

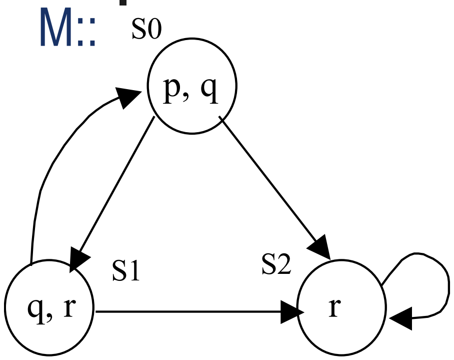
Алгоритм Model checking сводится к последовательному нахождению тех *подмножеств* состояний К, для которых выполняются *подформулы* формулы φ

Представляем все булевы функции в форме BDD для:
(а) множества начальных состояний К;
(б) множества переходов К;
(в) множеств тех состояний К, в которых истинен каждый атомарный предикат

Символьная верификация: Пусть задана CTL формула φ. Для каждой подформулы ϕ формулы φ будем строить в форме BDD характеристическую булеву функцию f_{ϕ}, задающую то множество состояний структуры Крипке К, на которых ϕ выполняется

Пример. Представление структуры Крипке в BDD

M::



$M=(S, s_0, R, L)$ Переменные: $v=\langle x_1, x_2\rangle$, $v'=\langle x_1', x_2'\rangle$

Кодирование состояний: $s_0 \Leftrightarrow 00$, $s_1 \Leftrightarrow 01$, $s_2 \Leftrightarrow 10$ (v)

Множество S = {00, 01, 10} (v) ; s_0= {00} – подмн-во S

Множество R = $\{\underbrace{00}_{v}\underbrace{01}_{v'}, 0010, 0100, 0110, 1010\}$ (v, v')

Функция пометок $L: S \to 2^{AP}$ Свойство p = {00} (v) - в состоянии s_0

Свойство q = {00, 01}(v) – в состояниях s_0, s_1

Свойство r = {01, 10}(v) – в состояниях s_1, s_2

Характеристические ф-ии :

$$\chi_{s0} = \neg x_1 \neg x_2$$

$$\chi_{R(v,v')} = \neg x_1 \neg x_2 (x_1' \oplus x_2') \vee \neg x_1 x_2 \neg x_2' \vee x_1 \neg x_2\, x_1' \neg x_2'$$

$$\chi_{p\ (v)} = \neg x_1 \neg x_2$$

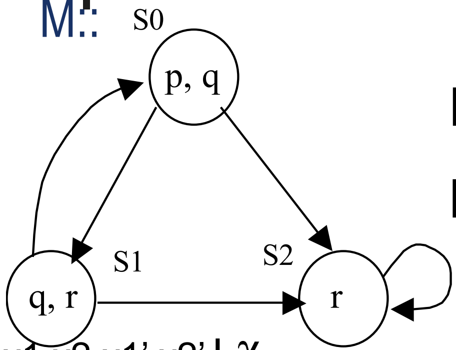
$$\chi_{q\ (v)} = \neg x_1$$

$$\chi_{r\ (v)} = x_1 \oplus x_2$$

Все эти функции представляем в BDD

Структура Крипке – характеристическая ф-ия R

M::

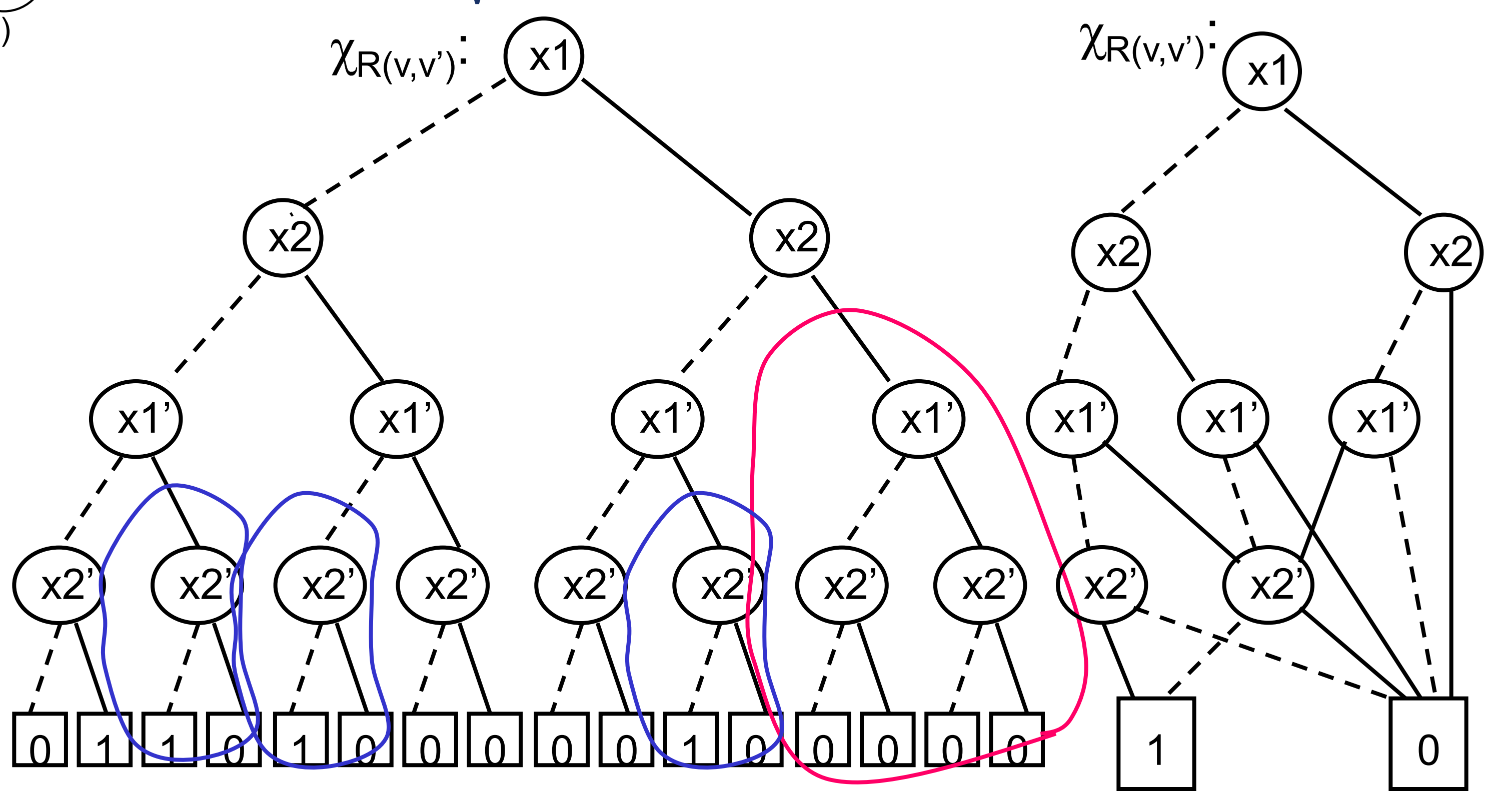


$v=\langle x_1, x_2\rangle$, $v'=\langle x_1', x_2'\rangle$

Кодирование состояний: $s_0 \Leftrightarrow 00$, $s_1 \Leftrightarrow 01$, $s_2 \Leftrightarrow 10$ (v)

Множество R = $\{\underbrace{00}_{v}\underbrace{01}_{v'}$, 0010, 0100, 0110, 1010$\}$ (v, v')

x1	x2	x1'	x2'	$\chi_{R(v,v')}$
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



Как построить структуру Крипке в BDD

- Структуры Крипке реальных систем содержат $> 10^{100}$ состояний
- Для представления подмножеств состояний нужны БФ от ~300 двоичных переменных, а для представления отношений нужны БФ от ~600 переменных, что при представлении БФ в форме BDD вполне выполнимо
- Но как их построить? Не перебирать же все состояния!

КАК представить структуру Крипке в форме BDD без предварительного явного построения состояний и переходов?

НУЖНО множество начальных состояний, переходы и функцию пометок для атомарных предикатов задавать только булевыми функциями в форме BDD!

Как задавать характеристические функции: Проблема фермер, волк, коза и капуста

Состояние – вектор значений переменных

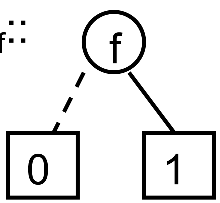
Булевы переменные – f, w, g, c.

Начальное состояние <0,0,0,0>; χ_{S0}= ~f~w~g~c

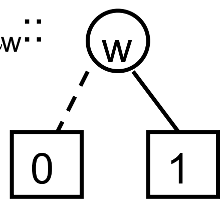
Атомарные предикаты:

χ_f = f; χ_w = w; χ_g = g; χ_c = c

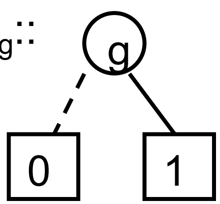
χ_f::



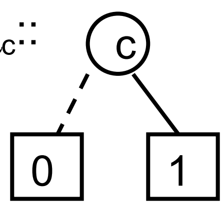
χ_w::



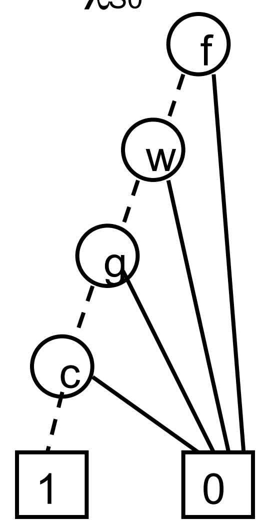
χ_g::

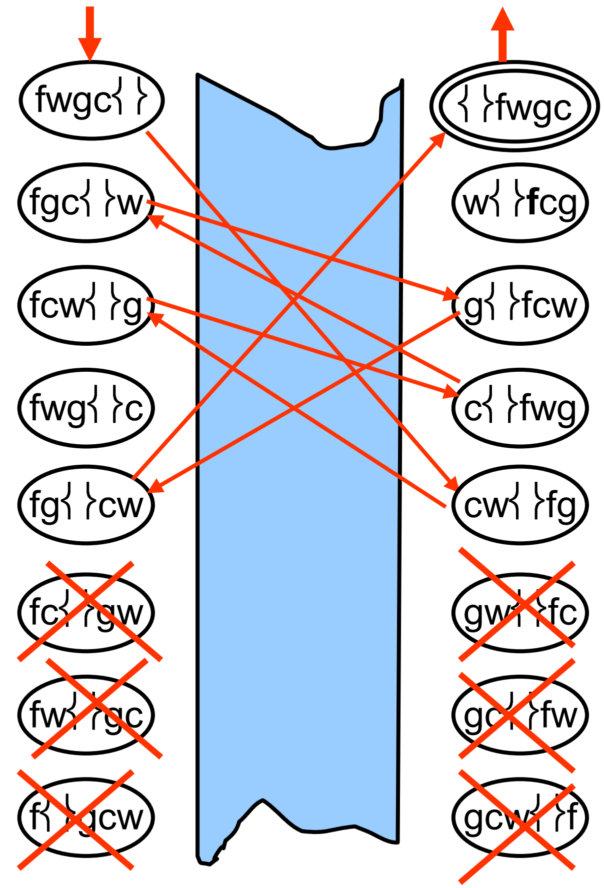


χ_c::



χ_{S0}::





Переходы:

а) сам фермер может поехать
б) может везти один предмет (волка, козу или капусту)

Проблема волк, коза и капуста: Представление переходов в BDD

Переходы:

- сам может переправиться или взять с собой не более одного объекта

R1 : сам едет $\chi_{R1}(v, v')$ =
$(f \equiv \sim f')(w \equiv w')(g \equiv g')(c \equiv c')$

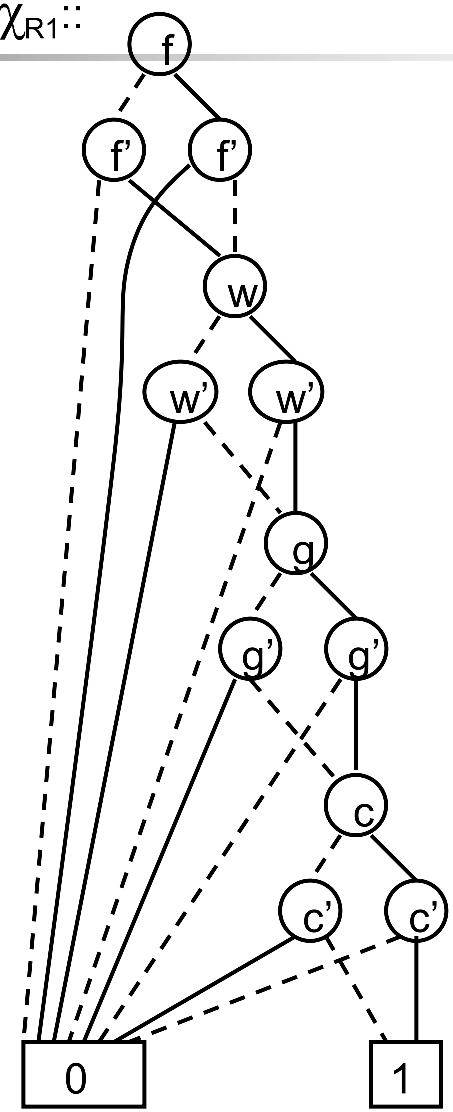
R2 : везет волка $\chi_{R2}(v, v')$ =
$(f \equiv w)(f \equiv \sim f')(w \equiv \sim w')(g \equiv g')(c \equiv c')$

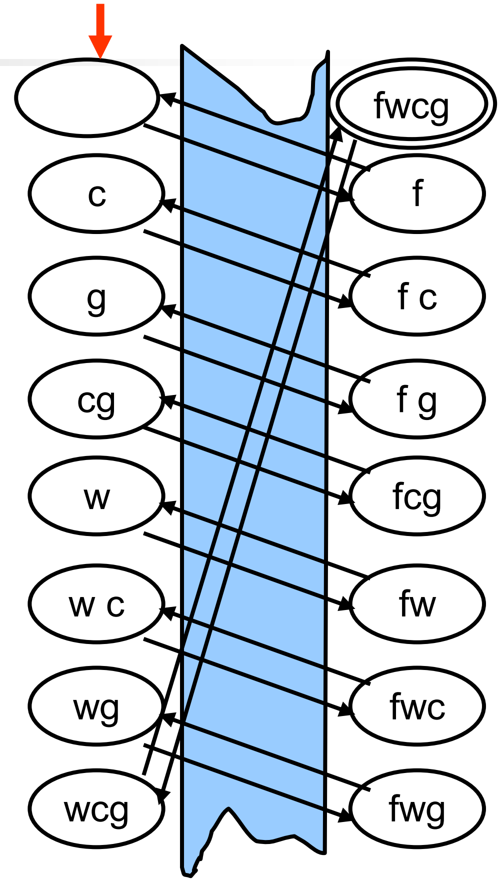
R3 : везет козу $\chi_{R3}(v, v')$ =
$(f \equiv g)(f \equiv \sim f')(w \equiv w')(g \equiv \sim g')(c \equiv c')$

R4 : везет капусту $\chi_{R4}(v, v')$ =
$(f \equiv c)(f \equiv \sim f')(w \equiv w')(g \equiv g')(c \equiv \sim c')$

$R = R1 \vee R2 \vee R3 \vee R4$

χ_{R1}::





Но явно проводить ребра НЕ НАДО!
Они все описываются характеристической функцией R

Все 16 переходов структуры Крипке, соответствующие R1, задаются простой булевой формулой χ_{R1}

Программа –> структура Крипке

Программа

```
begin
{x,y∈{0,1,2},x=y=1}
  x:=0;
  while x<2 do
    x:=x+1(mod3)
  od;
 y:= x-1(mod3)
end
```

Разметка

```
    begin
{x,y∈{0,1,2},x=y=1}
m0:  x:=0;
m1:   while x<2 do
m2:     x:=x+1(mod3)
      od;
m3:  y:=x-1(mod3)
m4:  end
```

состояния

M	x	y
…	…	…

R1=C(m0; [x:=0]; m1) =
$M{=}m0 \wedge M'{=}m1 \wedge x'{=}0 \wedge y{=}y'$

M	x	y
m0	1	1

$\xrightarrow{g}$

M	x	y
m1	0	1

$g \in R1$

R2= C(m1; [while x<2 do m2: x:=x+1 od]; m3) =
$M = m1 \wedge x<2 \wedge M' = m2 \wedge x'=x \wedge y'=y$
$\vee\, M = m1 \wedge \neg x<2 \wedge M' = m1 \wedge x'=0 \wedge y=y'$
$\vee\, C(m2, [x:=x+1], m1)$

R3 = C(m3; [y:=x-1]; m4) =
$M = m3 \wedge M'=m4 \wedge x'=x \wedge y'=x-1$

Программа –> структура Крипке

Кодировка выбирается автоматически

Программа

```
begin
{x,y∈{0,1,2},x=y=1}
  x:=0;
  while x<2 do
    x:=x+1(mod3)
  od;
 y:= x-1(mod3)
end
```

Разметка

```
   begin
{x,y∈{0,1,2},x=y=1}
m0: x:=0;
m1:  while x<2 do
m2:     x:=x+1(mod3)
        od;
m3: y:=x-1(mod3)
m4: end
```

состояния

M	x	y
…	…	…

M	x	y
$z_1z_2z_3$	t_1t_2	u_1u_2

$R1=C(m0;\ [x:=0];\ m1) =$
$M=m0 \wedge M'=m1 \wedge x'=0 \wedge y=y'$

$R3 = C(m3;\ [y:=x-1];\ m4) =$
$M = m3 \wedge M'=m4 \wedge x'=x \wedge y'=x-1$

M	x	y
$\neg z_1\neg z_2\neg z_3$	$\neg t_1t_2$	$\neg u_1u_2$

M	x	y
$\neg z_1\neg z_2z_3$	$\neg t_1\neg t_2$	$\neg u_1u_2$

$$\chi_{R1} = \neg z_1\neg z_2\neg z_3 \wedge \neg z'_1\neg z'_2z'_3 \wedge \neg t'_1\neg t'_2 \wedge (\neg u_1\neg u_2\neg u'_1\neg u'_2 \vee \neg u_1u_2\neg u'_1u'_2 \vee u_1\neg u_2u'_1\neg u'_2)$$

… …

$$\chi_{R3} = \neg z_1 z_2 z_3 \wedge z'_1\neg z'_2\neg z'_3 \wedge (\neg t_1\neg t_2\neg t'_1\neg t'_1\ u'_1 \neg u'_2 \vee \neg t_1\ t_2\neg t'_1\ t'_2 \neg u'_1\neg u'_2 \vee t_1 \neg t_2\ t'_1 \neg t'_2 \neg u'_1 u'_2)$$

Идея символьной верификации

Синтаксис: CTL (грамматика):

φ ::= p $| \neg\varphi$ | $\varphi_1 \vee \varphi_2$ | AXφ | EXφ | AGφ| EGφ | AFφ | EFφ | A[φ_1Uφ_2] | E[φ_1U φ_2]

Формулы состояний: Каждая CTL формула – это формула состояний, которая в каждом состоянии либо истинна, либо ложна

Для любой формулы φ логики CTL множество состояний Qφ, в которых эта формула истинна, будем задавать характеристической булевой функцией $\chi_{Q\varphi}$

Операции выполняются над характеристическими булевыми функциями этих множеств, представленными в BDD

Задача символьной верификации (например, конструкция EFφ):

ДАНО характеристическая функция множества состояний Q_φ структуры Крипке, на которых выполняется формула φ.

ТРЕБУЕТСЯ найти характеристическую функцию множества тех состояний $Q_{EF\varphi}$, на которых выполняется формула EFφ

Вычисление операторов: p, $\neg\varphi$, $\varphi 1 \vee \varphi 2$

Один из темпоральных базисов: ϕ ::= **p** | $\neg\varphi$ | $\varphi_1 \vee \varphi_2$ | EX φ | EG φ | E[φ_1U φ_2]

EX φ – из данного состояния есть ребро в состояние, помеченное φ

EG φ – из данного состояния есть путь, все состояния которого помечены φ

E [ϕ U φ] – из данного состояния существует путь, на котором есть состояние, помеченное φ, а все состояния до него помечены ϕ

Рассмотрим формулы базиса p, $\neg\varphi$, $\varphi_1 \vee \varphi_2$:

p – характеристическая функция χ_{Qp}, задающая для каждого атомарного предиката p множество состояний Q_p, в которых предикат p истинен, определяется в задании структуры Крипке

$\neg\varphi$ -- операция отрицания очевидным образом выполняется над характеристической функцией $\chi_{Q\varphi}$, задающей то множество состояний структуры Крипке, на котором выполняется формула φ

$\psi \vee \varphi$ -- операция дизъюнкции очевидным образом выполняется над соответствующими характеристическими функциями $\chi_{Q\psi}$ и $\chi_{Q\varphi}$

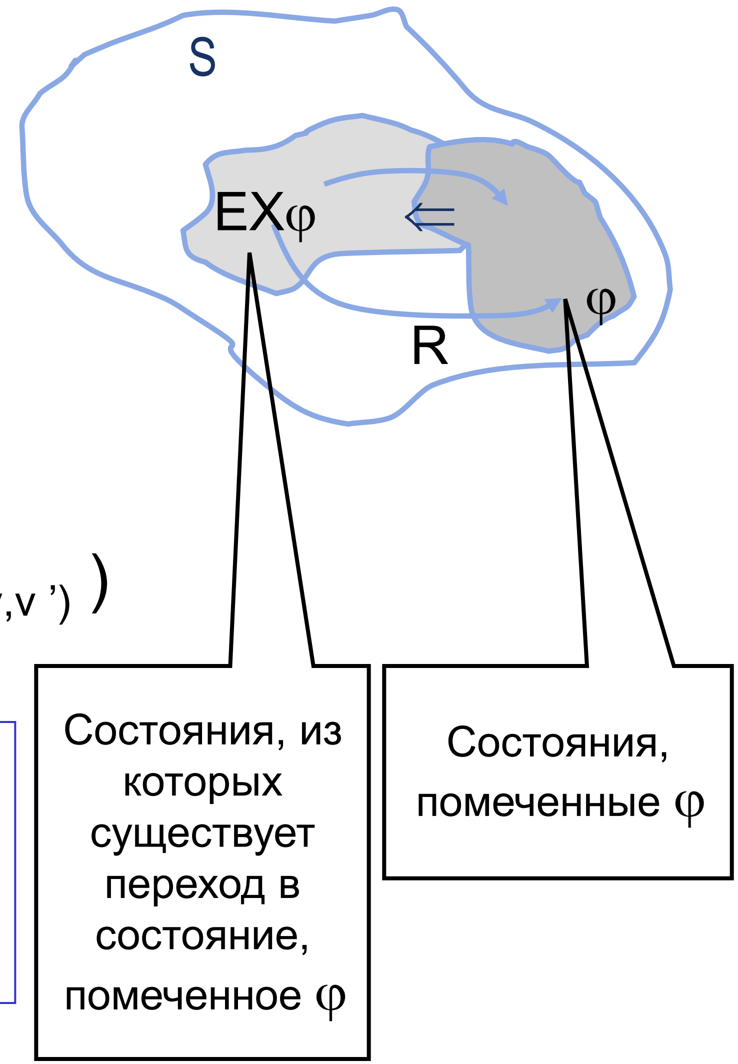
Вычисление множеств для операторов базиса: EXφ

Синтаксис: CTL (для одного из базисов):

$\varphi ::= \ p \mid \neg\varphi \mid \varphi\vee\varphi \mid EX\varphi \mid EF\varphi \mid E[\varphi\, U\, \varphi]$

Характеристическая БФ множества $Q_{EX\varphi}$ строится с помощью операции “Обратный Образ” (Reverse Image) над характеристическими БФ $\chi_{Q\varphi}$ и χ_R , представляющими Q_φ и R

$$\chi_{Rev\_Image(Q\varphi,R)(v)} = \exists v'.(\ \chi_{Q\varphi(v)}\langle v'/v\rangle \ \&\ \chi_{R(v,v')}\ )$$

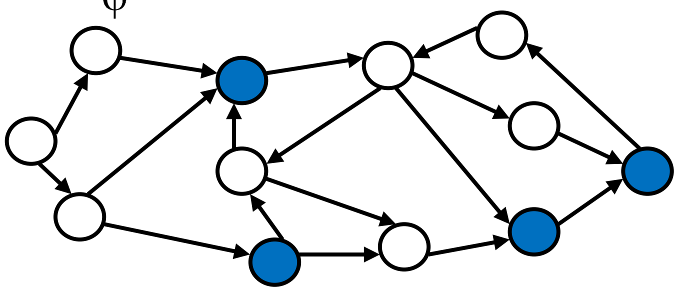


1. Строится ограничение отношения R на множестве Q_φ теми переходами, которые своими вторыми компонентами имеют вершины из Q_φ

2. Строится ‘обратный образ’ этого ограничения

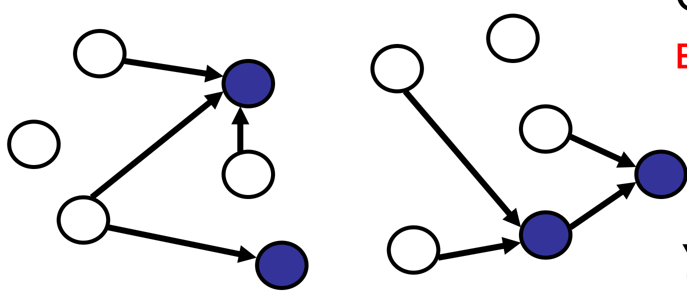
Пример вычисления оператора EXφ

Q_φ и R ::



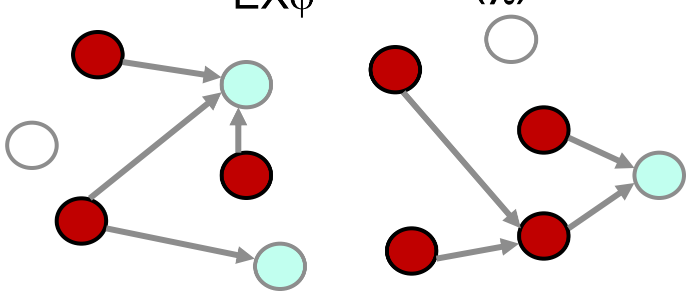
Состояния, удовлетворяющие ϕ, помечены (множество Q_φ определено). Отношение R задано

$\chi = (\chi_{Q\varphi\,(v)}\langle v'/v\rangle)\ \&\ \chi_{R\,(v,v')}$



Строим ограничение отношения R на множестве Q_φ (оставляем только такие переходы, у которых вторая компонента - из множества Q_φ)

$Q_{EX\varphi} = \exists v'.(\chi)$



У оставшихся переходов берем только первую компоненту с помощью операции квантификации по переменным второй компоненты. Так получаем множество $Q_{EX\varphi}$

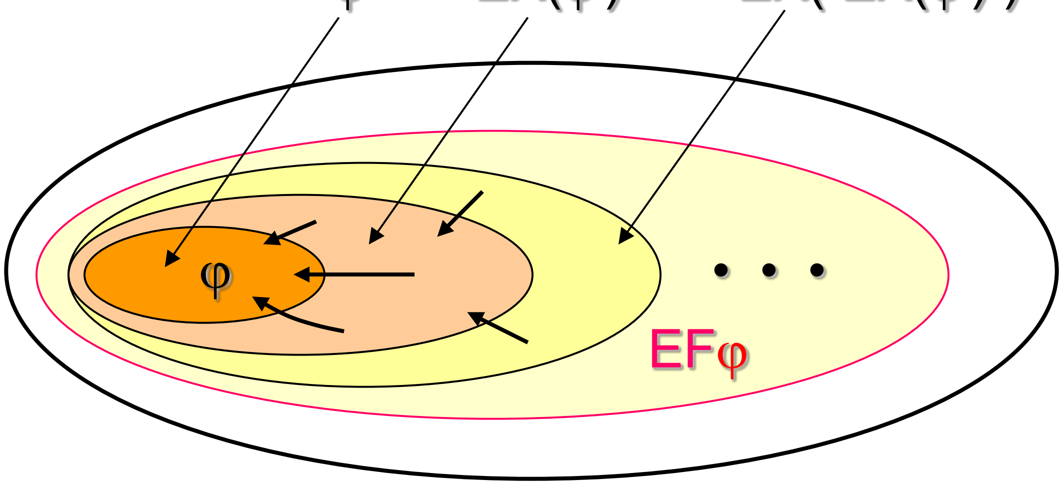
Всего несколько операций с BDD, представляющими Q_φ и R

Вычисление множества состояний, удовлетворяющих формуле EFφ (breadth-first алгоритм - поиск в ширину)

$EF\varphi = \varphi \vee EX\varphi \vee EX\ EX\varphi \vee EX\ EX\ EX\varphi \vee \ldots$

$EF\varphi \equiv$ состояния из которых можно достичь φ - состояния, помеченные:

$\varphi \ \cup \ EX(\varphi) \ \cup \ EX(EX(\varphi)) \ \cup \ldots$



$Q_0 = Q_\varphi$

$Q_1 = Q_0 \cup EX\ Q_0$

$Q_2 = Q_1 \cup EX\ Q_1$

…

$Q_{i+1} = \tau(Q_i)$

Вычисление искомых множеств состояний структуры Крипке состоит в повторяющихся преобразованиях этих множеств до тех пор, пока они не перестанут изменяться. Они называются НЕПОДВИЖНЫМИ точками

Вопросы:

всегда ли придем к пределу, или будет бесконечный перебор подмножеств?

если придем к пределу, будет ли это тем множеством, что мы хотели?

Операторы на множествах

Как разобраться в неподвижных точках??

Как построить операторы, которые после многократного применения к заданным множествам дадут нам требуемое множество состояний:

Для каждого типа подформул логики CTL нужно построить операторы, которые после многократного применения должны:

EF φ: по множеству, на котором выполняется φ, построить множество, на котором выполняется EF φ

AF φ: по множеству, на котором выполняется φ, построить множество, на котором выполняется AF φ

EG φ: по множеству, на котором выполняется φ, построить множество, на котором выполняется EG φ

… … …

Теория неподвижной точки преобразований множеств

- Вопросы:
  - всегда ли придем к пределу (неподвижной точке), или будет бесконечный перебор подмножеств?
  - если пришли к пределу, будет ли это предельное множество тем, что мы хотели?

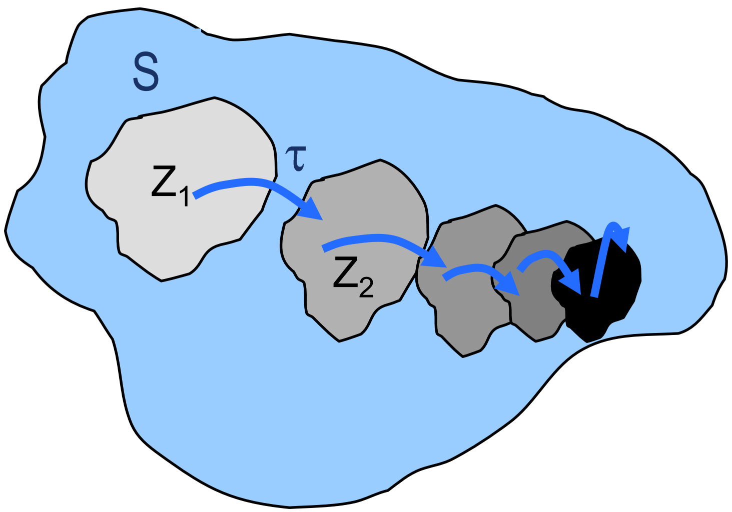
Ответ на эти вопросы дает теория неподвижной точки операторов на множествах

Теория фиксированной (неподвижной) точки нужна нам для обоснования применения операторов на множестве состояний структуры Крипке, которые дают множества таких состояний, на которых выполняются темпоральные формулы EFφ, EG φ, …, если уже построены множества состояний, на которых выполняется формула φ

Теория неподвижной точки (Fixpoint, FP) на конечном множестве

Пусть S – конечное множество, 2^S – совокупность его подмножеств

Оператор из S в S – это отображение τ: $2^S \to 2^S$, т.е. τ переводит подмножество в подмножество



τ: $2^S \to 2^S$ $\tau(Z_1) = Z_2$

Что будет, если оператор τ на 2^S применить несколько раз?

$$\tau^0(Z) = Z$$

$$\tau^k(Z) = \underbrace{\tau(\ \tau\ (\ldots \tau}_{k\ раз}(Z)\,)\,)$$

Поскольку S – конечно, результат будет либо сходиться к одному и тому же множеству, либо к циклическому повторению множеств

Неподвижной точкой отображения τ называется такое $Z \subseteq S$, что $\tau(Z)=Z$

Операторы на конечных множествах

$\tau_1(Q) = \{a, b\} \cup Q$

$S=\{a, b, c, d\}$

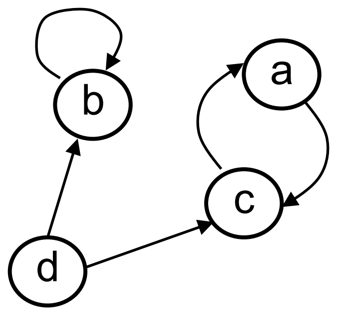
$\tau_1(\{a, d\}) = \{a, b, d\}$

$\tau_1(\{a, b, c\}) = \{a, b, c\}$ FP

$\tau_1(\{a, b\}) = \tau_1^2(\{a, b\}) = \ldots = \{a, b\}$ FP

$\tau_1(S) = S$ FP

$\tau_1(\varnothing) = \{a,b\}; \tau_1(\tau_1(\varnothing)) = \tau_1^k(\varnothing) = \{a,b\}$

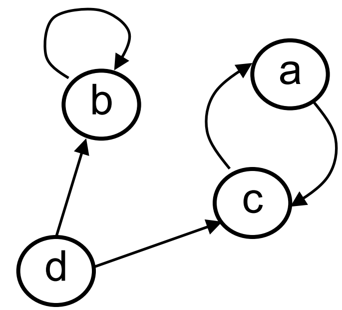


У τ_1 несколько неподвижных точек

Операторы на конечных множествах

$S=\{a, b, c, d\}$

$\tau_2\ (Q) = \{c\} \cup \{a\} \cap Q$



$\tau_2\ (\{a\}) = \{a, c\}.$

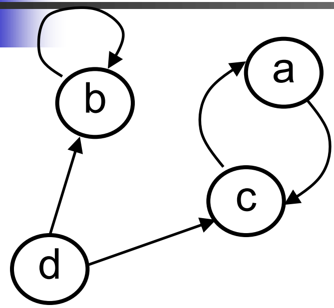
$\tau_2\ (\{a, c\}) = \{a, c\}$ FP

$\tau_2\ (S) = \{a, c\}$

$\tau_2\ (c) = \{c\}$ FP

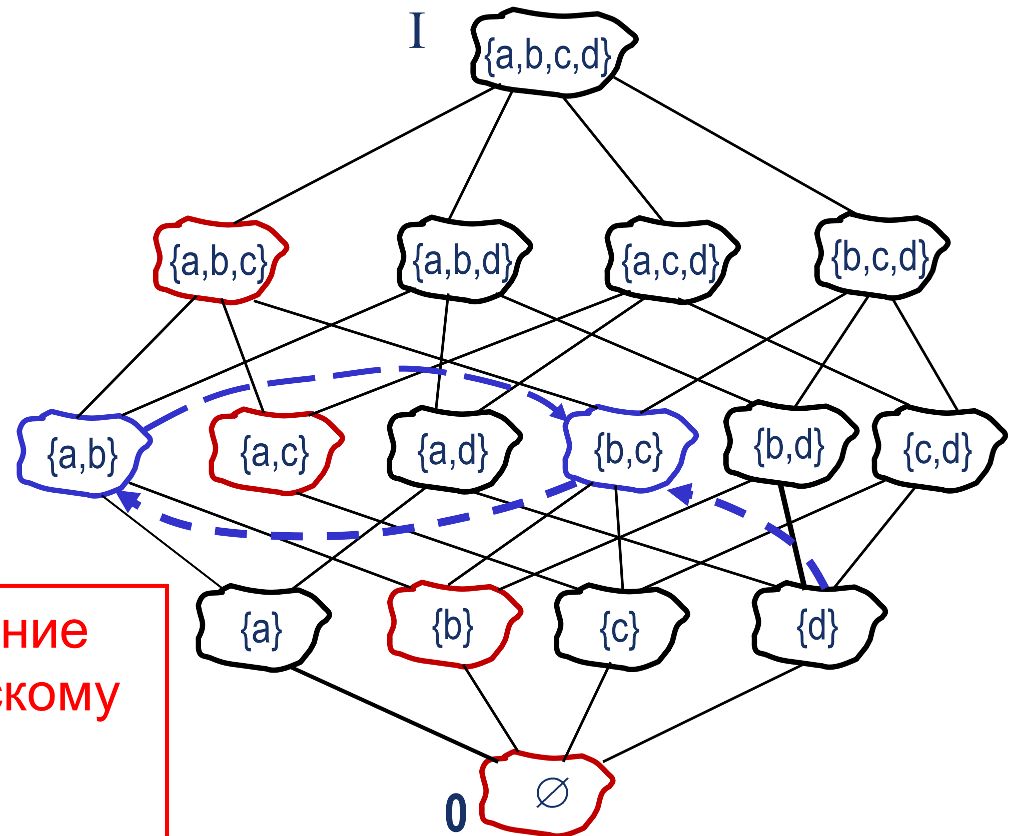
У τ_2 две неподвижные точки

Операторы на конечных множествах



$S=\{a, b, c, d\}$

Решетка всех подмножеств S



τ_3 (Q)=Q', где Q' – преемники Q

τ_3 ({b}) = {b} FP
τ_3 ({b, c}) = {a, b}
τ_3 ({a, b}) = {b, c}
τ_3 ({a, c}) = {a, c} FP

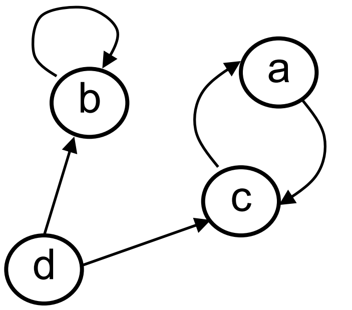
Существуют множества, применение τ_3 к которым приводит к циклическому повторению:

τ_3 ({d}) = {b,c};
τ_3 ({b,c}) = {a, b};
τ_3 ({a,b}) = {b, c};

Операторы на конечных множествах

$\tau_4 (Z) = Z - Z'$

$S=\{a, b, c, d\}$



Z′ – множество элементов, достижимых из Z за один шаг

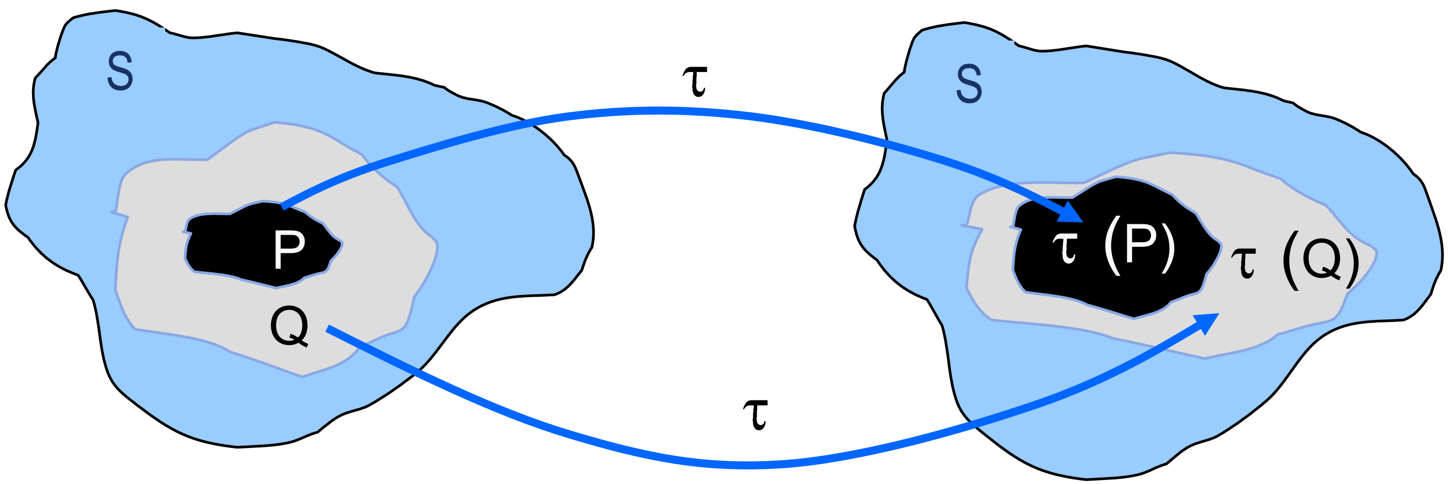
$\tau_4 (\{c,d\}) = \{d\}$
$\tau_4 (\{d\}) = \{d\}$ FP
$\tau_4 (\{a,c\}) = \varnothing$
$\tau_4 (\varnothing) = \varnothing$ FP
$\tau_4 (\{a\}) = \{a\}$

Как разобраться в неподвижных точках??

Как построить операторы, которые дадут нам требуемое множество состояний – по множеству

Монотонные операторы на конечных множествах

Оператор τ на 2^S монотонный, если $P \subseteq Q \Rightarrow \tau(P) \subseteq \tau(Q)$



$P \subseteq Q$

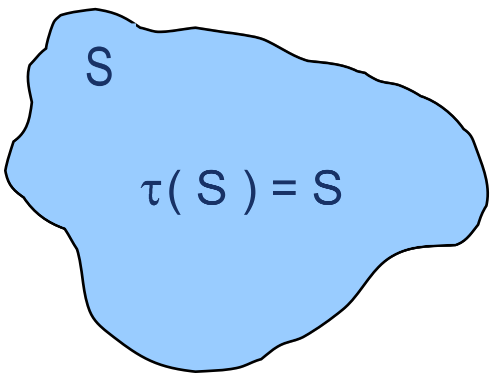
$\tau(P) \subseteq \tau(Q)$

Композиция монотонных операторов -- монотонный оператор

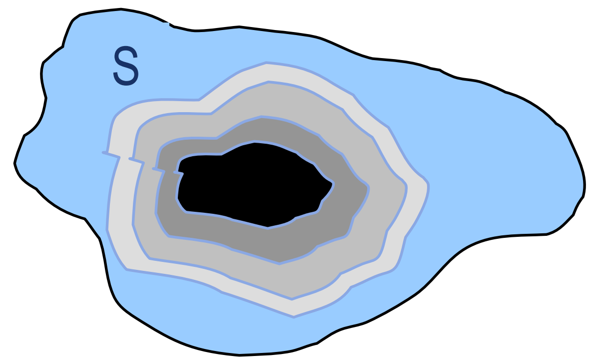
Теорема Тарского о неподвижной точке

Что, если **МОНОТОННЫЙ** оператор τ на 2^S применить к множеству S?

Очевидно, что $\tau(S) \subseteq S$, $\tau(\tau(S)) \subseteq \tau(S)$, и т.д.: $\tau^{k+1}(S) \subseteq \tau^{k}(S)$



Если $\tau(S) = S$,
то S – неподвижная точка τ,
$\tau^{k}(S) = S$



Если $\tau(S) \subset S$,
то $\tau^{k}(S) \subset S$, $\tau(S)$ – "сжимает" S:
$\tau^{k+1}(S) \subseteq \tau^{k}(S)$

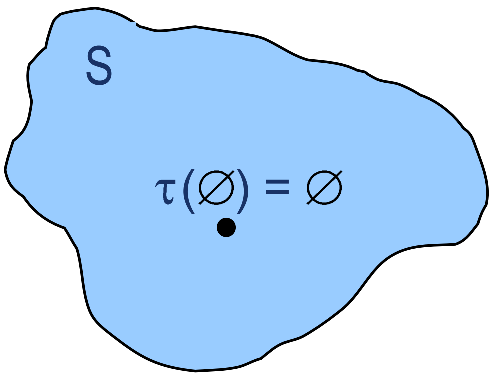
До каких пор?

Поскольку множество S конечное, то S, $\tau(S)$, $\tau^2(S)$ … $\tau^k(S)$ придет к FP

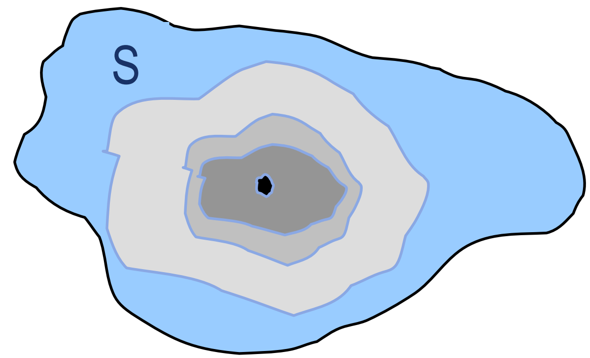
Теорема Тарского о неподвижной точке

Что, если монотонный оператор τ на 2^S применить к пустому множеству $\varnothing$?

Очевидно, что $\varnothing \subseteq \tau(\varnothing)$, $\tau(\varnothing) \subseteq \tau(\tau(\varnothing))$ и т.д.: $\tau^k(\varnothing) \subseteq \tau^{k+1}(\varnothing)$



Если $\varnothing = \tau(\varnothing)$,
то $\varnothing$ – неподвижная точка τ,
$\tau^k(\varnothing) = \varnothing$



Если $\varnothing \subset \tau(\varnothing)$,
то $\varnothing \subset \tau^k(\varnothing)$, $\tau(\varnothing)$ – “расширяющий” оператор: $\tau^k(\varnothing) \subseteq \tau^{k+1}(\varnothing)$

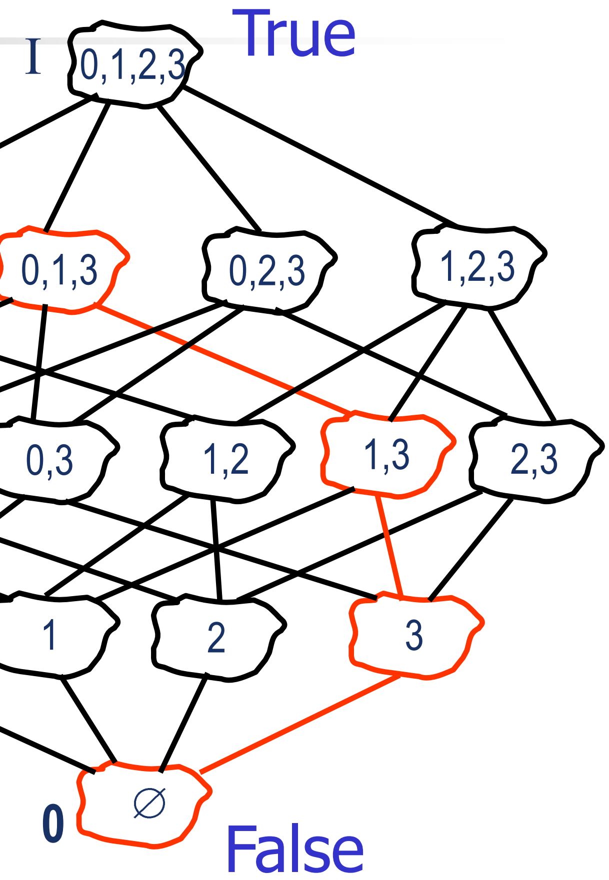
До каких пор?

Поскольку множество S конечное, то $\varnothing$, $\tau(\varnothing)$, $\tau^2(\varnothing)$ … $\tau^k(\varnothing)$ придет к FP

Решетка множества всех подмножеств

Пусть S={0,1,2,3}

2^S = {∅ {0}, {1}, {2}, {3}, {0,1}, … {0,1,2,3} }



Решетка – частичный порядок, в который для каждой пары элементов входит и наибольшая нижняя грань, и наименьшая верхняя грань.

Решетка имеет минимальный и максимальный элементы

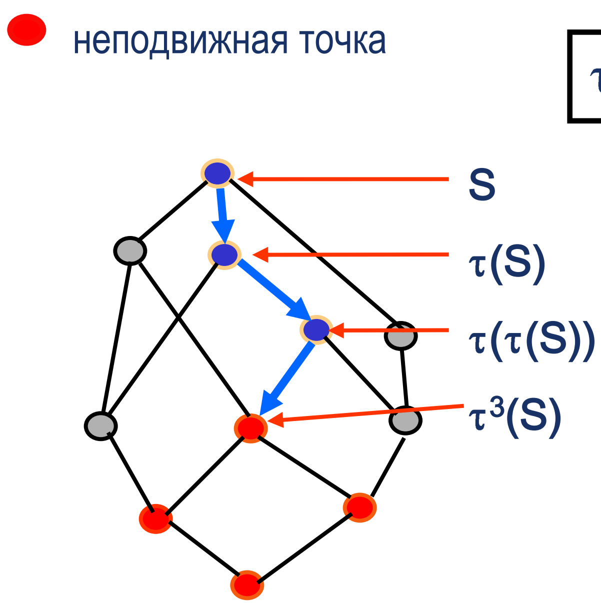
Все элементы 2^S образуют решетку по отношению частичной упорядоченности

A ≤ B iff A ⊆ B

Каждый элемент решетки – подмножество S, его можно считать предикатом, определенным на S. Конкретный предикат истинен в точности на тех состояниях, которые входят в подмножество. Таким образом, это – конечная решетка предикатов. Любой оператор на S – преобразователь предикатов

Теорема Тарского о неподвижной точке для конечных множеств

Теорема Тарского: Все неподвижные точки монотонного оператора на конечном множестве образуют решетку. Любой монотонный оператор имеет и наибольшую, и наименьшую FP, которые находятся простыми алгоритмами



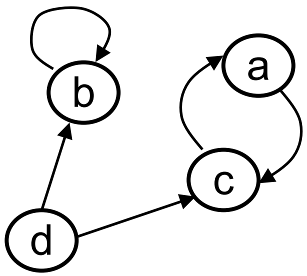
$\tau(S) \subseteq S$ всегда для монотонного оператора

$\tau(S) \subseteq S \Rightarrow \tau(\tau(S)) \subseteq \tau(S)$

Наибольшая неподвижная точка
$\nu S.\tau(S) = \tau^{\infty}(S)$

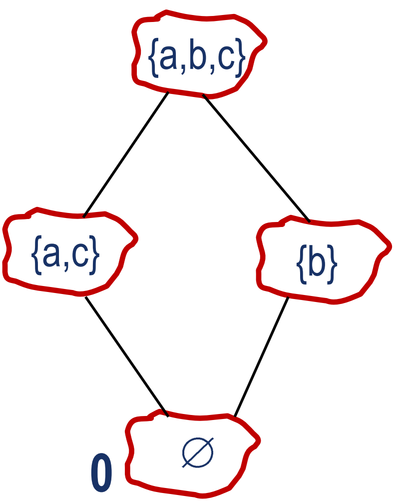
Во многих приложениях важны наибольшие FP

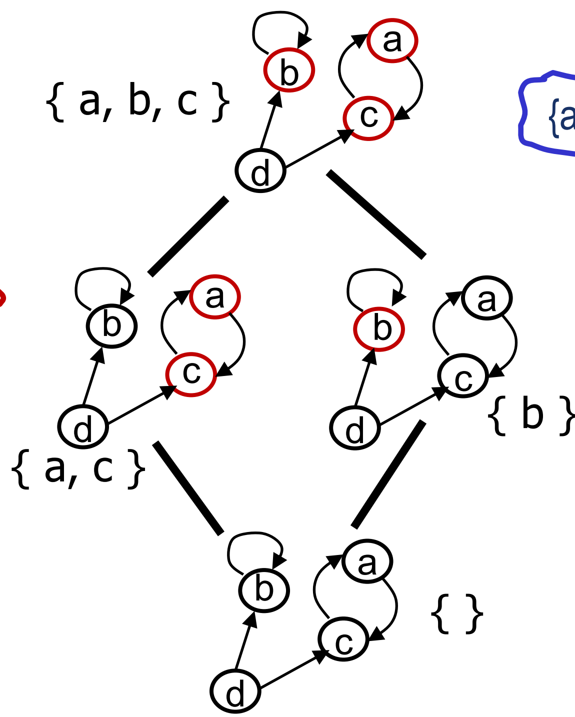
Решетка неподвижных точек оператора τ_3

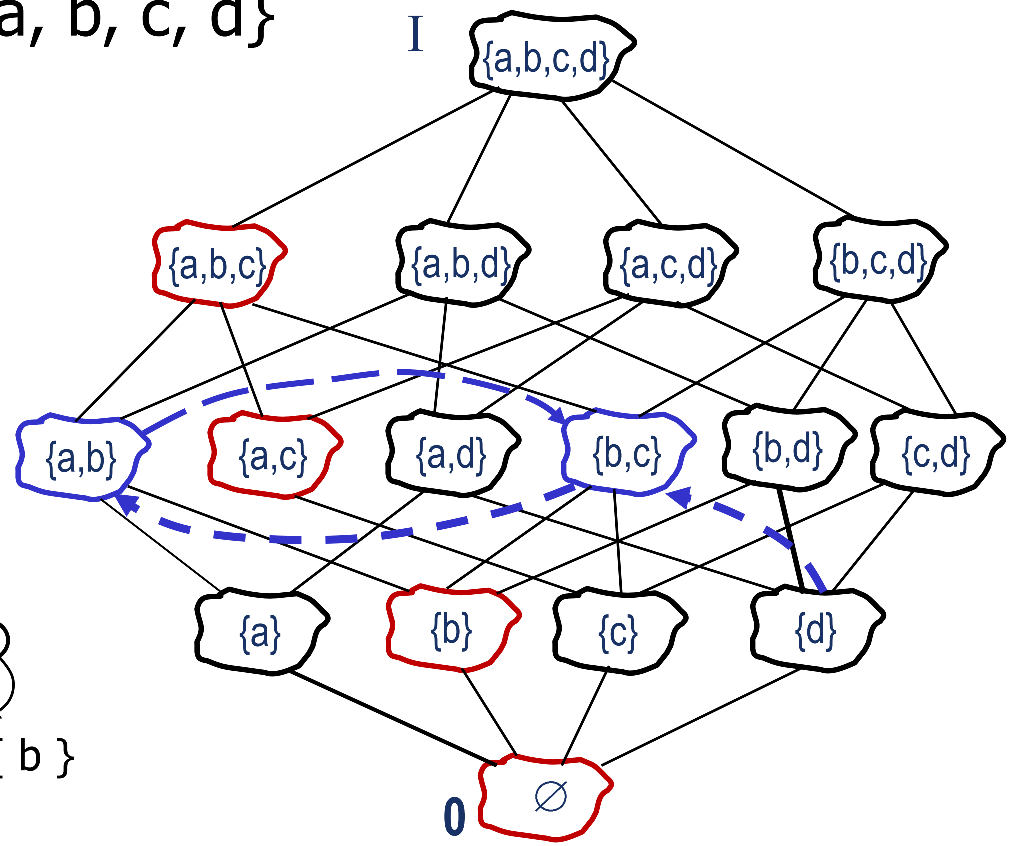


S={a, b, c, d}

τ_3 (Q)=Q', где Q' – преемники Q







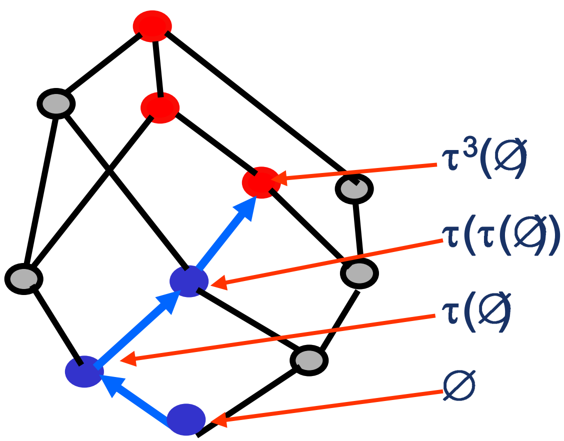
Все неподвижные точки этого оператора образуют решетку

Теорема Тарского о неподвижной точке для конечных множеств

Теорема Тарского: Все неподвижные точки монотонного оператора на конечном множестве образуют решетку. Любой монотонный оператор имеет и наибольшую, и наименьшую FP, которые находятся простыми алгоритмами

неподвижная точка

$\emptyset \subseteq \tau(\emptyset)$ всегда для монотонного оператора



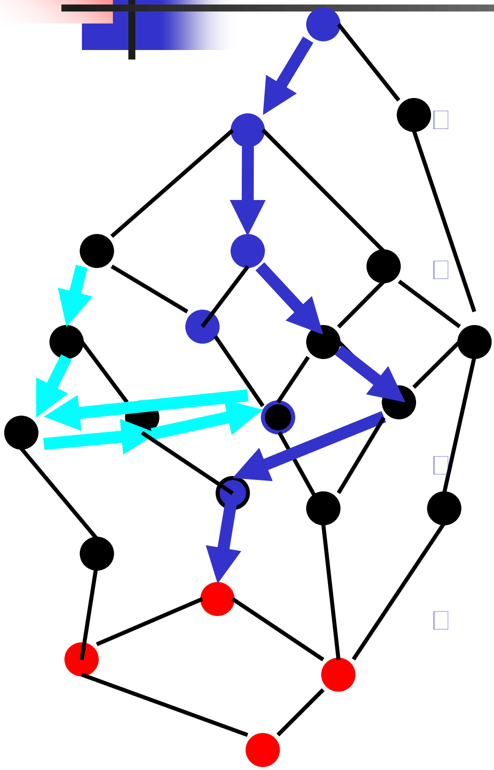
$\emptyset \subseteq \tau(\emptyset) \Rightarrow \tau(\emptyset) \subseteq \tau(\tau(\emptyset))$

Расширяющий оператор

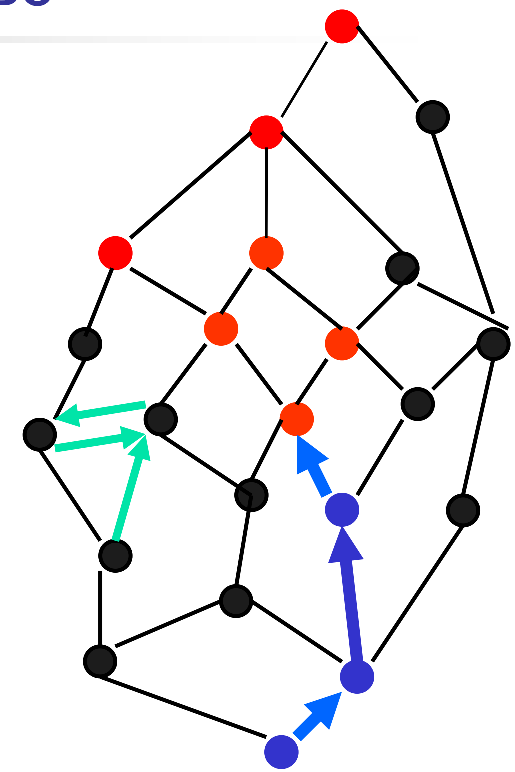
Наименьшая неподвижная точка
$\mu S.\tau(S) = \tau^{\infty}(\emptyset)$

Во многих приложениях важны наименьшие FP

Теорема Тарского о неподвижной точке на конечных множествах - доказательство



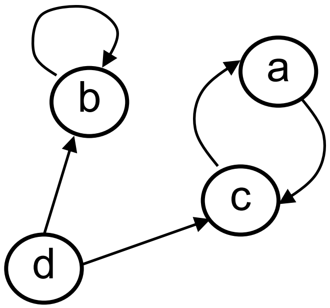
- Любой монотонный оператор на конечном множестве имеет неподвижные точки
- Все эти неподвижные точки образуют решетку, поэтому среди них есть минимальный и максимальный элемент
- Наибольшая неподвижная точка монотонного оператора может быть найдена как предел $\tau^k(S)$, k=0, 1, …
- Наименьшая неподвижная точка монотонного оператора может быть найдена как предел $\tau^k(\varnothing)$, k=0, 1, …



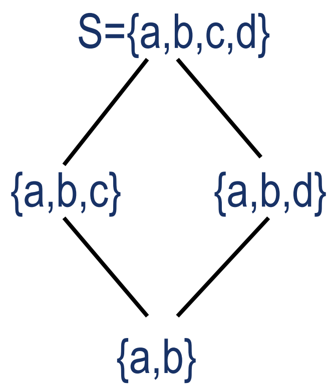
Доказательство. Для любого k, $\tau^k(\varnothing) \subseteq \tau^{k+1}(\varnothing)$ (база: $\varnothing \subseteq \tau(\varnothing)$, шаг индукции – по монотонности τ). Пусть $M = \tau^\infty(\varnothing)$. Поскольку S- конечное множество, то $\tau^k(\varnothing) = \tau^{k+1}(\varnothing)$ с какого-то k. Но тогда $\tau(M) = \tau^{k+1}(\varnothing) = M$ (след,M – FP)

Докажем, что $M \subseteq$ любой другой FP M*. Применим k раз к обеим частям очевидного неравенства $\varnothing \subseteq M^*$ оператор τ: $\tau^k(\varnothing) = M \subseteq \tau^k(M^*) = M^*$

Решетки неподвижных точек операторов

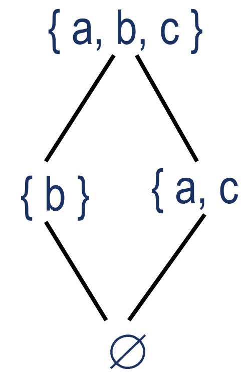


$\tau_1(X) = \{a, b\} \cup X$

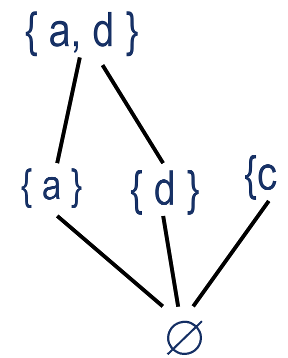


$\tau_2(Z) = \{c\} \cup \{a\} \cap Z$

$\tau_3(Z) = Z'$



$\tau_4(Z) = Z - Z'$



$\tau_1 - \tau_3$ монотонные операторы

Их неподвижные точки образуют решетку, а наибольшая и наименьшая неподвижные точки находятся простыми алгоритмами

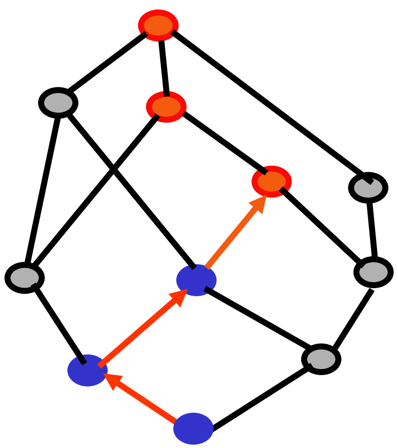
τ_4 – не монотонный оператор: $\tau_4(\{a\}) = \{a\}$, $\tau_4(\{a, c\}) = \varnothing$

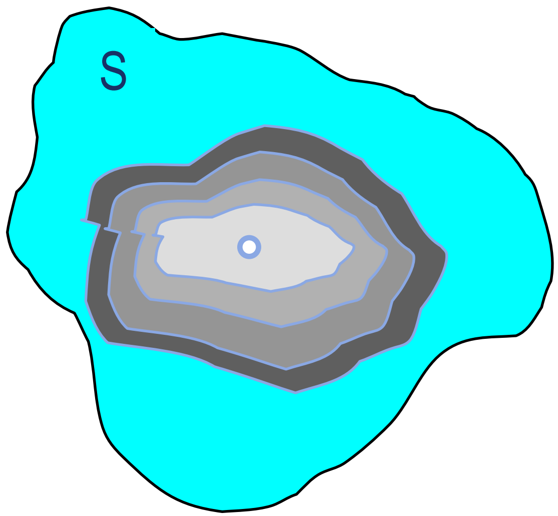
У этого оператора есть неподвижные точки, но они не образуют решетку. У него нет наибольшей неподвижной точки

Вычисление наименьшей неподвижной точки
Least Fixed Point (LFP)

$\mu Z.\ \tau(Z) = \bigcup_{k=0}^{\infty} \tau^k(\varnothing)$

$\tau^{k+1}(\varnothing) \supseteq \tau^k(\varnothing)$





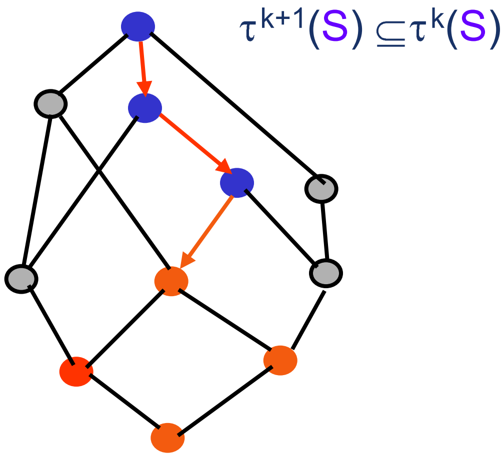

```
Вычисление LFP( Tau )
   A:= ∅;
  do
    B:=A;
    A:=Tau ( B );
   while (A ≠ B);
    return (A);
```

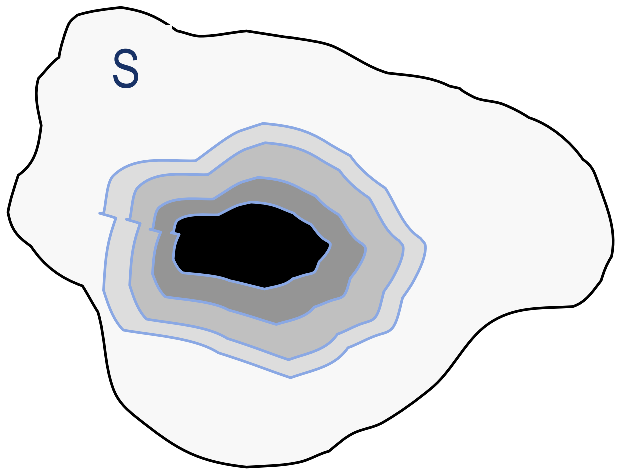
Наименьшая неподвижная точка монотонного оператора τ вычисляется как предел последовательности:
$\varnothing \subseteq \tau(\varnothing) \subseteq \tau(\tau(\varnothing)) \ \ldots$

Эта последовательность всегда имеет предел – Least Fixed Point

Вычисление наибольшей неподвижной точки
Greatest Fixed Point (GFP)

$\nu Z.\ \tau\ (Z) = \bigcap_{k=0}^{\infty} \tau^k\ (\ S\ )$






```
Вычисление GFP (Tau )
   A:= S;
  do
    B:=A;
    A:=Tau ( B );
   while (A ≠ B);
    return (A);
```

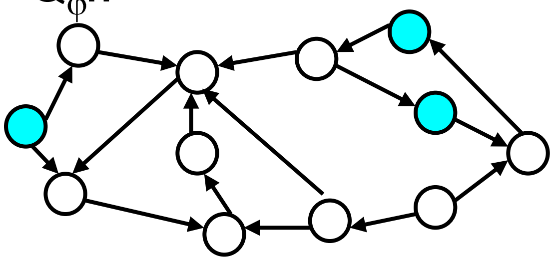
Наибольшая неподвижная точка монотонного оператора τ вычисляется как предел последовательности:
$S \supseteq \tau\ (S) \supseteq \tau\ (\ \tau\ (S)\ ) \supseteq \dots$

Эта последовательность всегда имеет предел – Greatest Fixed Point

Пример: Множество $Q_{EF\varphi}$ как минимальная неподвижная точка оператора

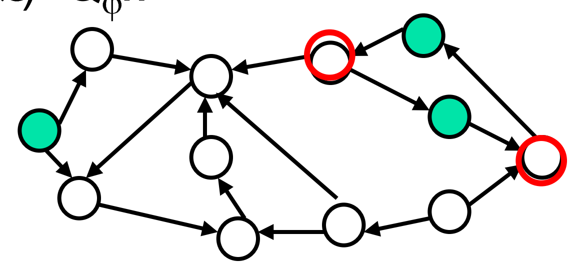
Состояния, которые помечены EFφ - это МИНИМАЛЬНОЕ множество таких состояний Q, которые включают Qφ, + все те, из которых за один шаг можно достичь состояния, помеченного Q (минимальное, потому что, например, все множество S тоже удовлетворяет этому определению!).

Q_φ::

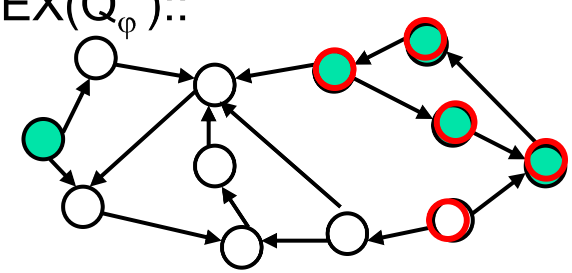


$$\tau(Q) = Q_\varphi \cup EX(Q)$$

$\tau^1(\varnothing)=Q_\varphi$::

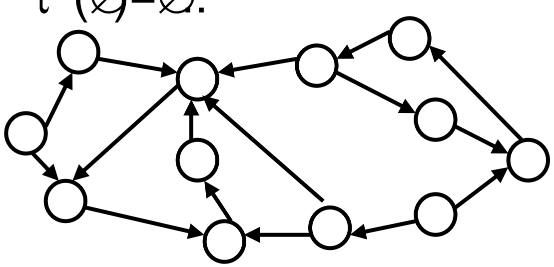


$\tau^2(\varnothing)= Q_\varphi \cup EX(Q_\varphi)$::

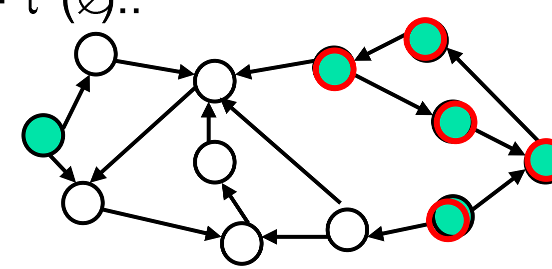


Наибольшая FP определяется как предел $\varnothing \subseteq \tau(\varnothing) \subseteq \tau^2(\varnothing) \subseteq \tau^3(\varnothing) \subseteq \ldots$

$\tau^0(\varnothing)=\varnothing$:



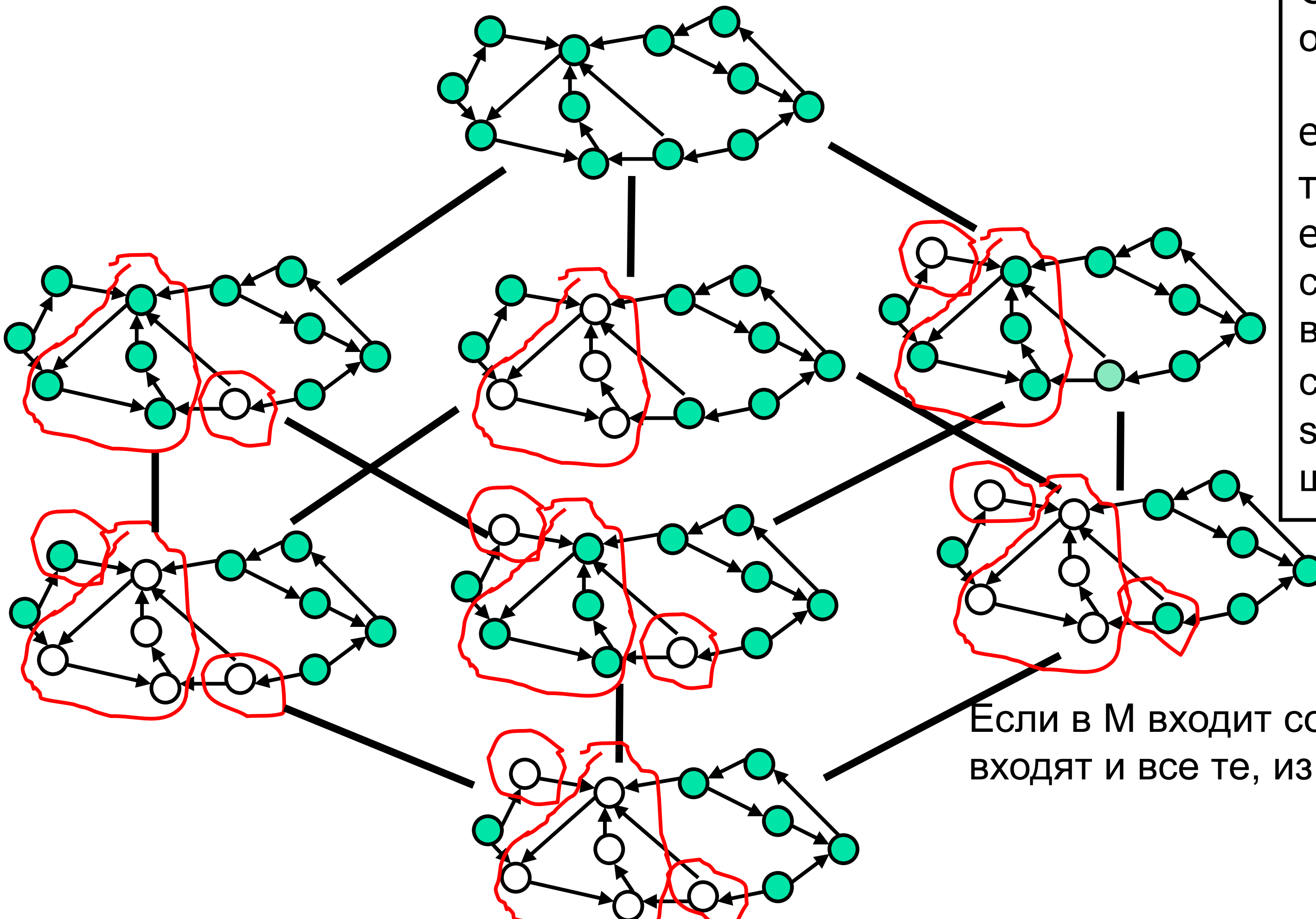
$\tau^3(\varnothing)= Q_\varphi \cup EX\tau^2(\varnothing)= \tau^\infty(\varnothing)$::



Красным обведены состояния, из которых за 1 шаг можно достичь помеченные, т.е.EX(помеченные)

Решетка неподвижных точек оператора $\tau(Q) = Q_\varphi \cup EX(Q)$ – нахождение множества $Q_{EF\varphi}$

Наибольшая неподвижная точка – все множество S

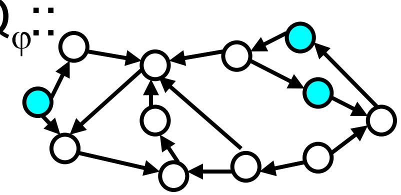


Если в M входит состояние s, то в M входят и все те, из которых s достижимо

Наименьшая неподвижная точка – искомое множество EFφ

Q – неподвижная точка оператора
$\tau(Q) = Q_\varphi \cup EX(Q)$,
если $M = Q_\varphi \cup EX(M)$,
т.е. M включает Q_φ и если в M входит состояние s, то в M входят и все те состояния, из которых s достижимо за один шаг

Q_φ:

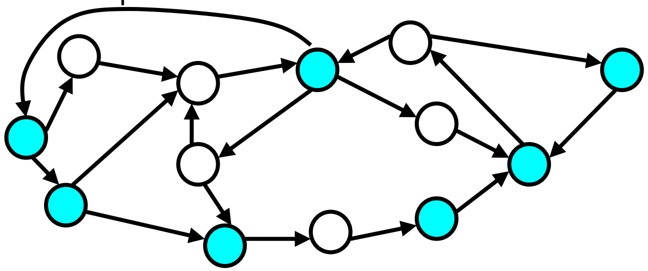


Пример: наибольшая неподвижная точка оператора

Множество состояний, помеченных φ, из которых существует путь, на котором φ выполняется на каждом расстоянии четной длины от этих состояний

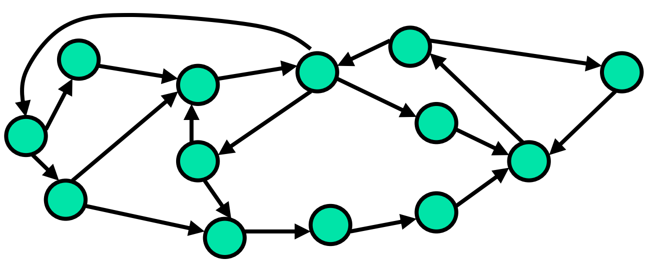
$\tau(Q) = Q_\varphi \cap EXEX(Q)$

φ ::

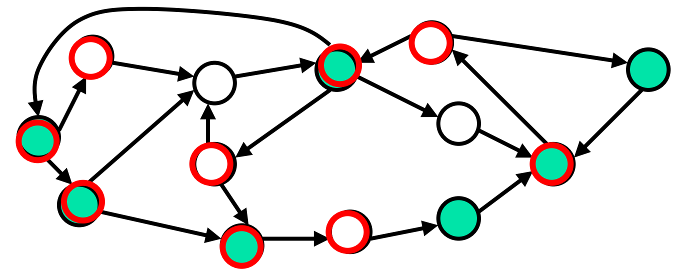


Наибольшая FP определяется как предел $S \supseteq \tau\ (S) \supseteq \tau^2(S) \supseteq \tau^3(S) \supseteq \ldots$

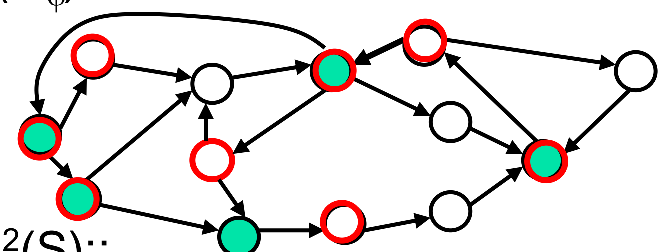
$\tau^0(S)=S$::



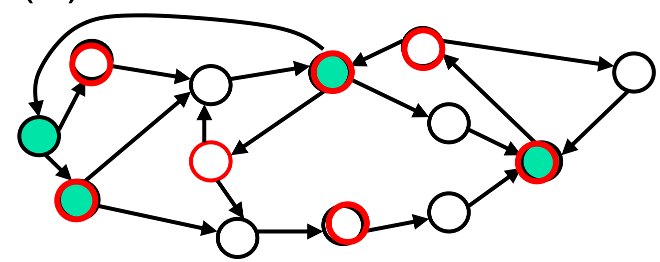
$\tau^1(S)= Q_\varphi$ ::



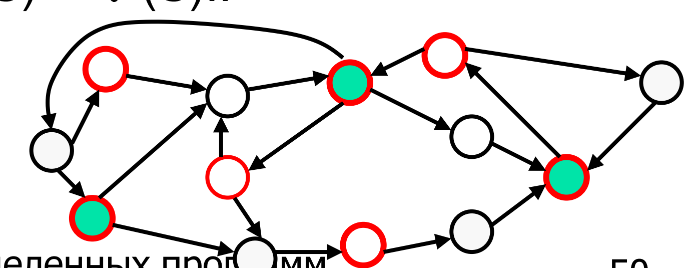
$\tau^2(S)= Q_\varphi \cap EXEX\ (Q_\varphi)$ ::



$\tau^3(S)= Q_\varphi \cap EXEX\tau^2(S)$::



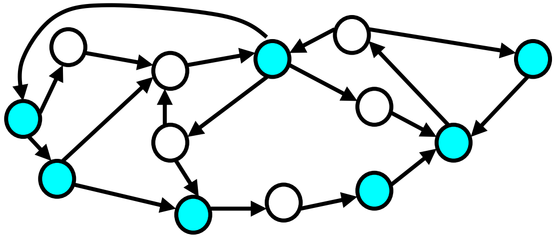
$\tau^4(S) = Q_\varphi \cap EXEX\tau^3(S)\ = \tau^\infty(S)$::



Красным обведены состояния EXEX (Q_φ), т.е. те, из которых можно достичь помеченные φ за 2 шага

Решетка неподвижных точек оператора $\tau(Q) = = Q_\varphi \cap EXEX\ Q$

Q_φ::



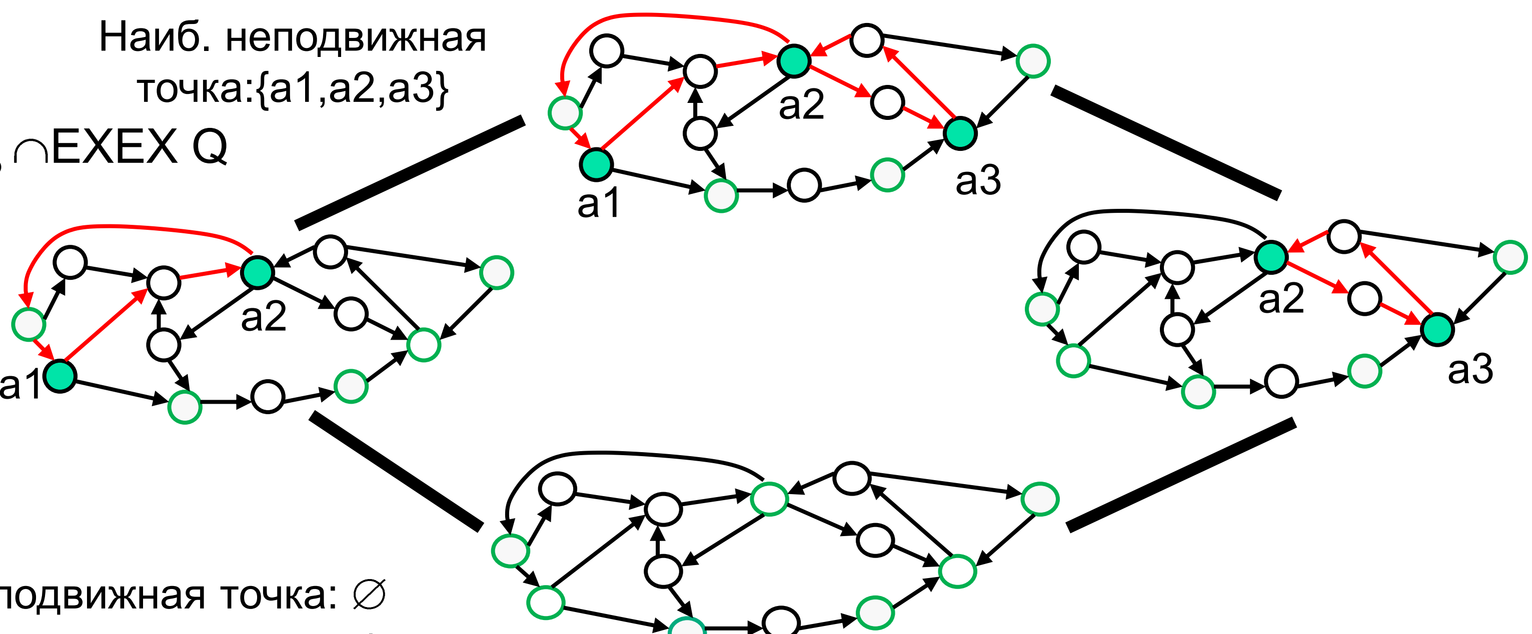
Четыре решения – неподвижные точки

Q – неподвижная точка оператора
$\tau(Q) = Q_\varphi \cap EXEX(Q)$,
если $M = Q_\varphi \cap EXEX(M)$, т.е. M включает только состояния из Q_φ, и если в M входит s, то в M входят и все те состояния, из которых s достижимо за два шага

Множества состояний из Q_φ, из которых существует путь, на котором φ выполняется на каждом расстоянии четной длины от них, образуют решетку

Наиб. неподвижная точка:{a1,a2,a3}

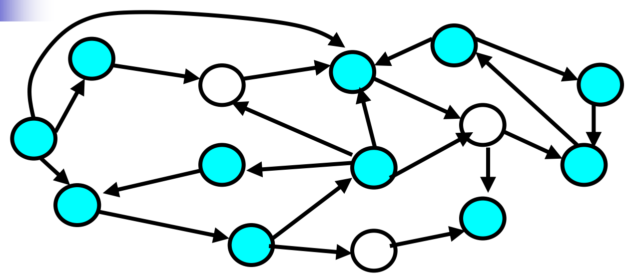
$\tau(Q) = Q_\varphi \cap EXEX\ Q$

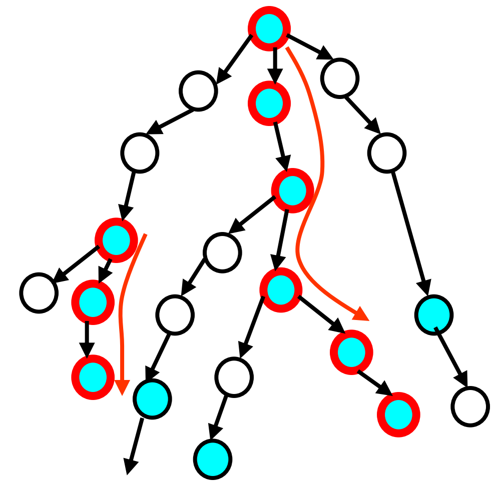


Наим. неподвижная точка: $\varnothing$

Пример: множество $Q_{EG\varphi}$ как наибольшая неподвижная точка оператора

Q_φ ::





На структуре Крипке задано множество состояний, в которых истинна формула φ. Найти множество *M* таких состояний, на которых истинна формула EG φ

Множество M можно определить так: Это подмножество таких состояний, помеченных φ, что в них формула EX φ истинна (из каждого из них существует переход в состояние из M). Очевидно, $M = Q_\varphi \cap EX\ M$

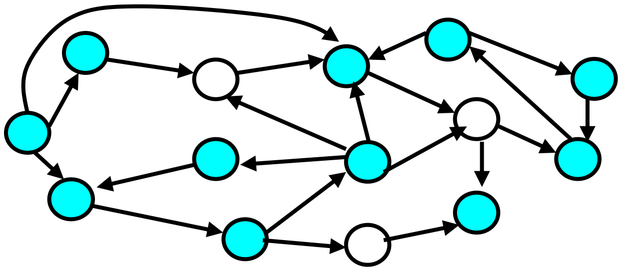
Это значит, что *M* является неподвижной точкой оператора $\tau(Q)=Q_\varphi \cap EXQ$

Оператор $\tau(Q)$ - монотонный, он имеет наибольшую и наименьшую FP

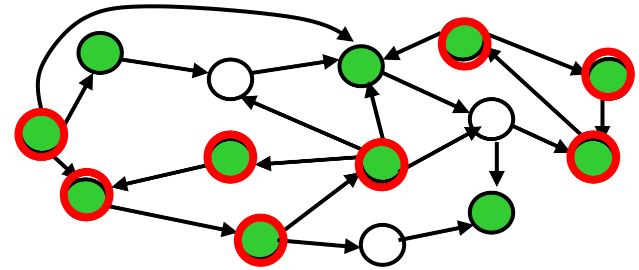
Определению удовлетворяет любая неподвижная точка. Например, наименьшая FP – пустое множество. Но нам, очевидно, нужна НАИБОЛЬШАЯ неподвижная точка оператора $\tau(Q) = Q_\varphi \cap EXQ$

Пример: Наибольшая неподвижная точка оператора $\tau(Q) = Q_\varphi \cap EX(Q)$

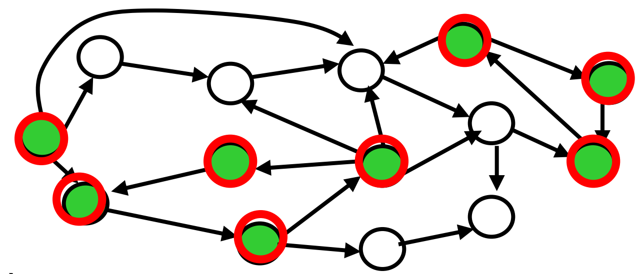
Q_φ ::



$\tau^1(S)=Q_\varphi \cap EX(S)$::



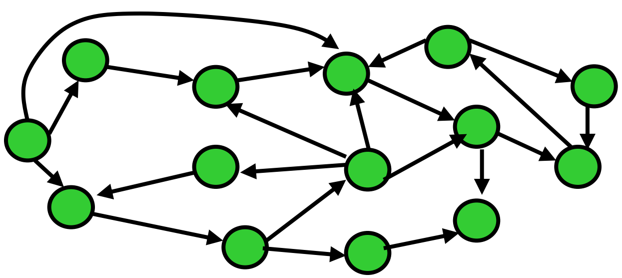
$\tau^2(S)= Q_\varphi \cap EXQ_\varphi$::



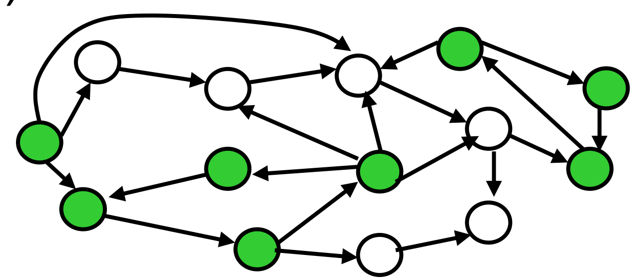
$\tau(Q) = Q_\varphi \cap EX(Q)$

Максимальная FP определяется как предел $S \supseteq \tau(S) \supseteq \tau^2(S) \supseteq \tau^3(S) \supseteq \ldots$

$\tau^0(S)=S$::



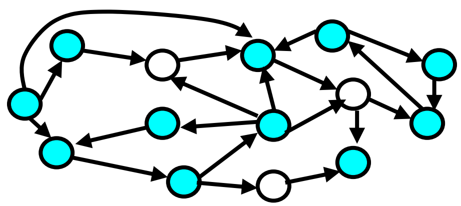
$\tau^3(S) =\varphi \cap EX\tau^2(S) = \tau^\infty(S)$::



Красным обведены состояния EXФ, (из которых можно достичь помеченные Ф за 1 шаг

Решетка неподвижных точек оператора $\tau(Q) = Q_\varphi \cap EX(Q)$

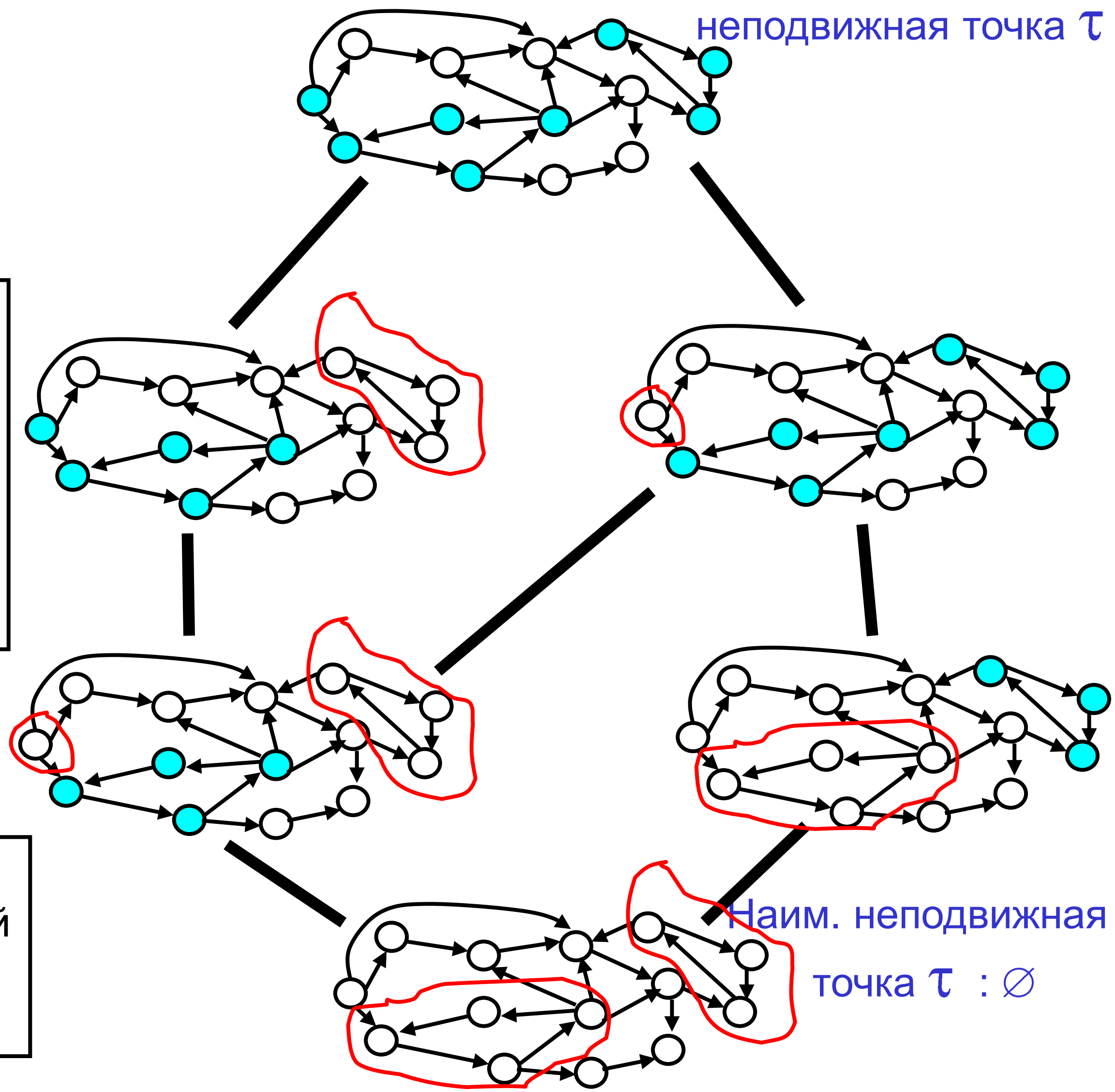
Q_φ ::



Q – неподвижная точка оператора $\tau(Q) = Q_\varphi \cap EX(Q)$, если $M = Q_\varphi \cap EX(M)$, т.е. M включает только состояния из Q_φ, и если в M входит s, то в M входят и все те состояния, из которых s достижимо за один шаг

Алгоритм построения FP – на множестве помеченных состояний находим ССК и те состояния, из которых ССК достижимы

$Q_{EG\varphi}$ - Наибольшая неподвижная точка τ



Наим. неподвижная точка τ : $\varnothing$

Рекурсивнре определение темпоральных операторов CTL

$EF\ f = f \vee EX\ EF\ f$

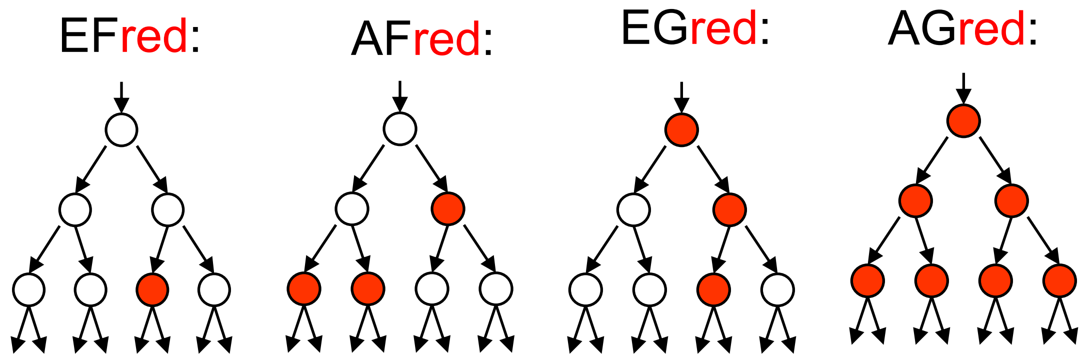
$AF\ f = f \vee AX\ AF\ f$

$EG\ f = f \wedge EX\ EF\ f$

$AG\ f = f \wedge AX\ AG\ f$

$E[\ f_1\ U\ f_2\ ] = f_2 \vee f_1 \wedge EX\ E[\ f_1\ U\ f_2\ ]$

$A[\ f_1\ U\ f_2\ ] = f_2 \vee f_1 \wedge AX\ A[\ f_1\ U\ f_2\ ]$



$EF\ f = \mu S\ .\ f \cup EX\ S$

$AF\ f = \mu S\ .\ f \cup AX\ S$

$EG\ f = \nu S\ .\ f \cap EX\ S$

$AG\ f = \nu S\ .\ f \cap AX\ S$

$E[\ f_1\ U\ f_2\ ] = \mu S\ .\ f_2 \cup (f_1 \cap EX\ S\ )$

$A[\ f_1\ U\ f_2\ ] = \mu S\ .\ f_2 \cup (f_1 \cap AX\ S\ )$

Определение темпоральных операторов CTL как неподвижных точек основывается на их рекурсивном определении

Символьная верификация: все операции

$EF\ f = \mu S\ .\ f \cup EX\ S$

$AF\ f = \mu S\ .\ f \cup AX\ S$

$EG\ f = \nu S\ .\ f \cap EX\ S$

$AG\ f = \nu S\ .\ f \cap AX\ S$

$E[\ f_1\ U\ f_2\ ] = \mu S\ .\ f_2 \cup (f_1 \cap EX\ S\ )$

$A[\ f_1\ U\ f_2\ ] = \mu S\ .\ f_2 \cup (f_1 \cap AX\ S\ )$

LFP $\mu S.\tau(S)$ - свойства eventuality

GFP $\nu S.\tau(S)$ - свойства forever

$\mu S.\tau(S)$ - наименьшая FP

$\nu S.\tau(S)$ - наибольшая FP

Операции $\cup$, $\cap$, EX и AX

Операции выполняются над характеристическими булевыми функциями для множеств состояний

- $\cup$, $\cap$ реализуются булевыми операциями $\vee$, $\wedge$ над ф-циями в BDD

- EX f реализуется с помощью Backward Image над булевыми ф-циями в BDD, представляющими подмножество Q_f и отношение R структуры Крипке

- $AX\ f = \neg EX\ \neg f$

Использование в верификаторах

- Символьные верификаторы для CTL (в, частности, SMV) представляют подмножества состояний и переходы структуры Крипке в форме BDD и используют характеризацию формул CTL в виде неподвижных точек (fixpoint, FP) для того, чтобы построить множество таких состояний структуры Крипке, на которых эти формулы выполняются.
- Например, множество состояний, на которых выполняется формула EF p, представляется булевой формулой, являющейся наименьшей неподвижной точкой оператора $\tau: 2^Q \to 2^Q$, имеющего следующий вид: $\tau(Z) = S_p \cup EX\ Z$, где S_p – множество состояний, на которых выполняется p
- Построение неподвижной точки $EF\ f = \mu Z\ .\ (f \cup EX\ Z)$ таких операторов требует стандартных операций с множествами (а, следовательно, стандартных булевых операций над характеристическими БФ этих множеств, операций квантификации переменных и операций подстановок переменных), и могут быть реализованы в форме BDD очень эффективно
- Least Fixpoint оператора τ (обозначается lfp(τ), $\mu(\tau)$) строится так:
  False, τ(False), $\tau(\tau$(False)), ... τ^n(False)

  Greatest Fixpoint оператора τ (обозначается gfp(τ), $\nu(\tau)$) строится так:
  True, τ(True), $\tau(\tau$(True)), ... τ^n(True)

  Для монотонных операторов эти последовательности сходятся
- $E[f\ U\ g] = \mu Z.\ g \vee (f \wedge EXZ)$; $EG\ f = \nu Z.\ f \wedge EX\ Z$.

Результаты

Все операции, нужные для проведения разметки на структуре Крипке (вычисления истинности) любых темпоральных формул CTL, могут быть выполнены над характеристическими булевыми функциями. Каждая операция над булевой функцией – это неявная операция над множеством состояний структуры Крипке

На практике число итераций до достижения FP невелико. Символьный метод позволяет увеличить размерность анализируемых систем от 10^6 до 10^{100}. Опубликованы работы, в которых описывался опыт верификации систем с 10^{130} состояний и даже 10^{300} состояний

Это комбинаторные числа: число атомов во вселенной менее 10^{80}

Разработано несколько инструментов выполняющих символьную верификацию. Самый известный из них – SMV (Беркли) – свободно распространяется. Эта система стала основой многих практических систем верификации

Заключение (1)

State explosion problem
при верификации дискретных систем с конечным числом состояний

Экономное каноническое представление булевых функций в виде BDD

Теорема Тарского о фиксированных точках монотонных операторов на решетках предикатов

Использование характеристических булевых функций для представления дискретных структур данных (подмн и отн)

Алгоритмы вычисления подмножеств, удовл формулам TL, как фиксированных точек операторов

Символьная верификация

Заключение (2)

The 1998 ACM Kanellakis Award for Theory and Practice

- was awarded to Randal E. Bryant, Edmund M. Clarke, Jr., E. Allen Emerson, and Kenneth L. McMillan for their invention of "**symbolic model checking**", a method of formally checking system designs that is widely used in the computer hardware industry and is beginning to show significant promise also in software verification and other areas

Спасибо за внимание