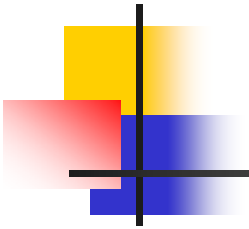
# Верификация параллельных программных и аппаратных систем

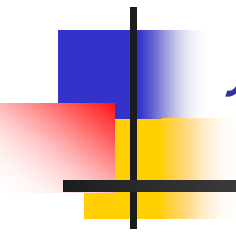
Курс лекций

Карпов Юрий Глебович
профессор, д.т.н., зав.кафедрой
“Распределенные вычисления и компьютерные сети”
Санкт-Петербургского политехнического университета
karpov@dcn.infos.ru



# План курса

1. Введение
2. Метод Флойда-Хоара доказательства корректности программ
3. Исчисление взаимодействующих систем (CCS) Р.Милнера
4. Темпоральные логики
5. Алгоритм model checking для проверки формул CTL
6. Автоматный подход к проверке выполнения формул LTL
7. Структура Крипке как модель реагирующих систем
8. Темпоральные свойства систем
9. Система верификации Spin и язык Promela. Примеры верификации
10. Применения метода верификации model checking
11. BDD и их применение
12. Символьная проверка моделей
13. Количественный анализ дискретных систем при их верификации
14. Верификация систем реального времени ( I )
15. Верификация систем реального времени ( II )
16. Консультации по курсовой работе



# *Лекция 6*

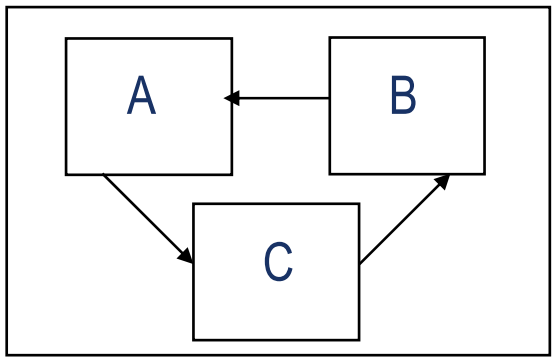
*Символьная проверка моделей*

# 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 ~$10^2$ – $10^3$ байт

Пример:




Простая система, а число состояний уже десять миллионов 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

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 ~$10^{100}$

# BDD: борьба с проблемой взрыва числа состояний

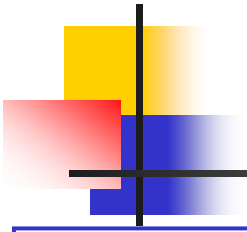
Model Checking – эффективная техника верификации, но есть недостатки. Число состояний при введении дополнительных параметров и/или компонент исследуемой системы растет экспоненциально: “State explosion problem”

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.

Поэтому 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



# 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 новых (СИМВОЛЬНЫХ) методов хранения данных и манипулирования ими с помощью BDD верификация реальных систем была бы невозможна
- Этот удивительный метод позволяет оперировать множествами с числом элементов ~$10^{100}$ и более

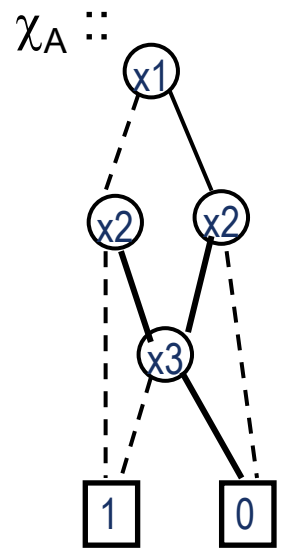
# Характеристические функции

Пусть S={a0, ..., a8} и A={a2, a4, a5} – подмножество S. Закодируем элементы S двоичными кодами. Тогда A = { 010, 100, 101 }

Обозначим $\chi_A$ 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 }




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: $\chi_S$ = True
- пустое множество: $\chi_\varnothing$ = False

Унарная операция:

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

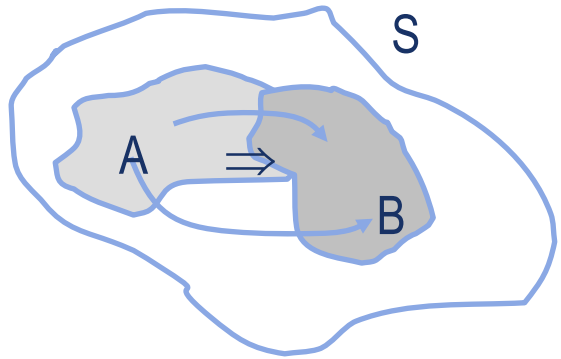
Бинарные операции:

- 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:

# Прямой образ для бинарных отношений

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:

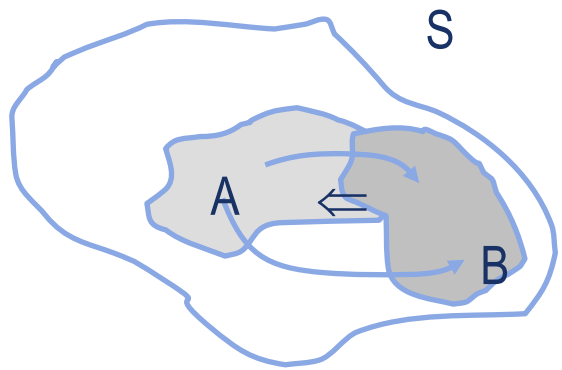
$$\chi_{FI(A,R)(v)} = \exists v.[\underbrace{\chi_{A(v)} \,\&\, \chi_{R(v,\,v')}}_{\text{Переходы из элементов } \in A}] < v/v' >$$

(< 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 несколько операций с булевыми характеристическими функциями $\chi_{A(v)}$ и $\chi_{R(v,\,v')}$

# Обратный образ для бинарных отношений

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



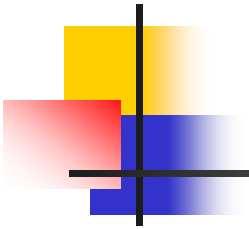

Ограничение отношения R на тех вторых элементах R из S, которые принадлежат B:

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

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

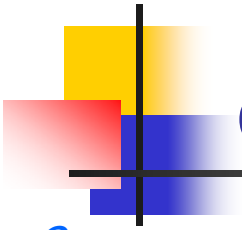
$\chi_{RI(B,R)\ (v)} = \exists v'.(\ \chi_{B\,(v')} \;\&\; \chi_{R\,(v,v')}\ )$ – выбрасываем все вторые элементы

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



# 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 с помощью БФ, представленных в форме BDD
  - 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

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

*Алгоритм Model checking* сводится к нахождению *подмножеств* состояний K, для которых выполняются *подформулы* формулы $\phi$

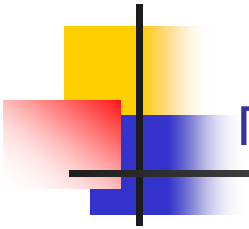
*Строим булевы функции* в форме BDD для:

(а) множества начальных состояний K;

(б) множества переходов K

(в) множеств состояний K, в которых истинен каждый атомарный предикат

*Символьная верификация:* Все подмножества состояний будем представлять булевыми функциями в форме BDD, т.е. для каждой CTL- формулы $\phi$ будем строить в форме BDD характеристическую булеву функцию $f_\phi$, задающую то множество состояний структуры Крипке K, на которых $\phi$ выполняется



# Приложение к Model Checking – темпоральные формулы

*Синтаксис*: CTL (грамматика):

$\phi ::= p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid AX\phi \mid EX\phi \mid AG\phi \mid$
$EG\phi \mid AF\phi \mid EF\phi \mid A[\phi_1 U \phi_2] \mid E[\phi_1 U \phi_2]$

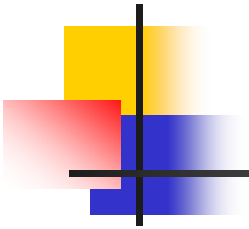
*State formulas:* Каждая CTL формула – это формула состояний, которая в каждом состоянии либо истинна, либо ложна

Поэтому можно считать, что любая CTL формула $\phi$ описывает множество состояний, в которых она истинна:

$Q_\phi = \{ s \in S \mid s \models \phi \}$,

т.е. $Q_\phi$ можно определить как множество таких состояний s из S, на которых эта формула истинна

Для CTL формулы $\phi$ будем ОПРЕДЕЛЯТЬ характеристическую функцию множества, на котором $\phi$ истинно



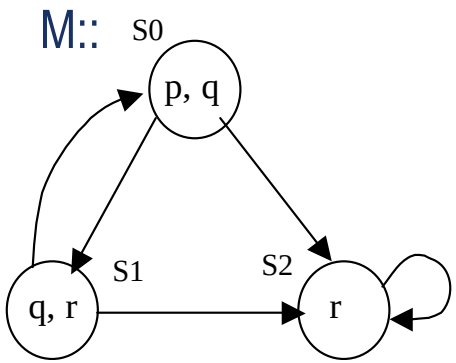
# Представление множества $10^{20}$ состояний

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

Структура Крипке для системы с числом состояний $10^{20}$ требует:
- множество начальных состояний (1 функция от 70 переменных)
- множество переходов (1 функция от 140 переменных)
- для каждого атомарного предиката и каждой подформулы темпоральной формулы – множество состояний, в которых эта формула истинна (1 функция от 70 переменных)

# BDD и структура Крипке




M=(S, $s_0$, R, L)  Переменные:v=<$x_1$, $x_2$> , v'=<$x_1$', $x_2$'>

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 = {0001, 0010, 0100, 0110,  1010 } (v, v')

v  v'

Функция пометок L: $S \rightarrow 2^{AP}$

Свойство p = {00} (v)

Свойство q = {00, 01}(v)

Свойство  r = {01, 10}(v)

Характеристические ф-ии для $s_0$, R и L:

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

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

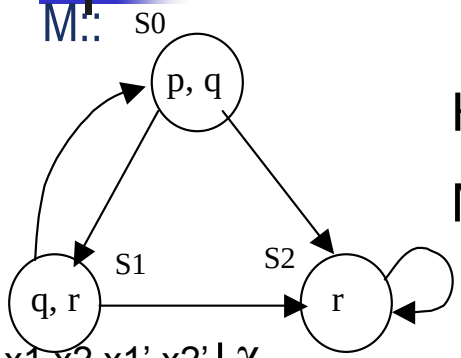
$\chi_{p\,(v)} = \neg x1 \neg x2$

$\chi_q\,(v) = \neg x1$

$\chi_r(v) = x_1 \oplus x_2$

Все эти функции представляем в BDD

# Структура Крипке – характеристические ф-ии

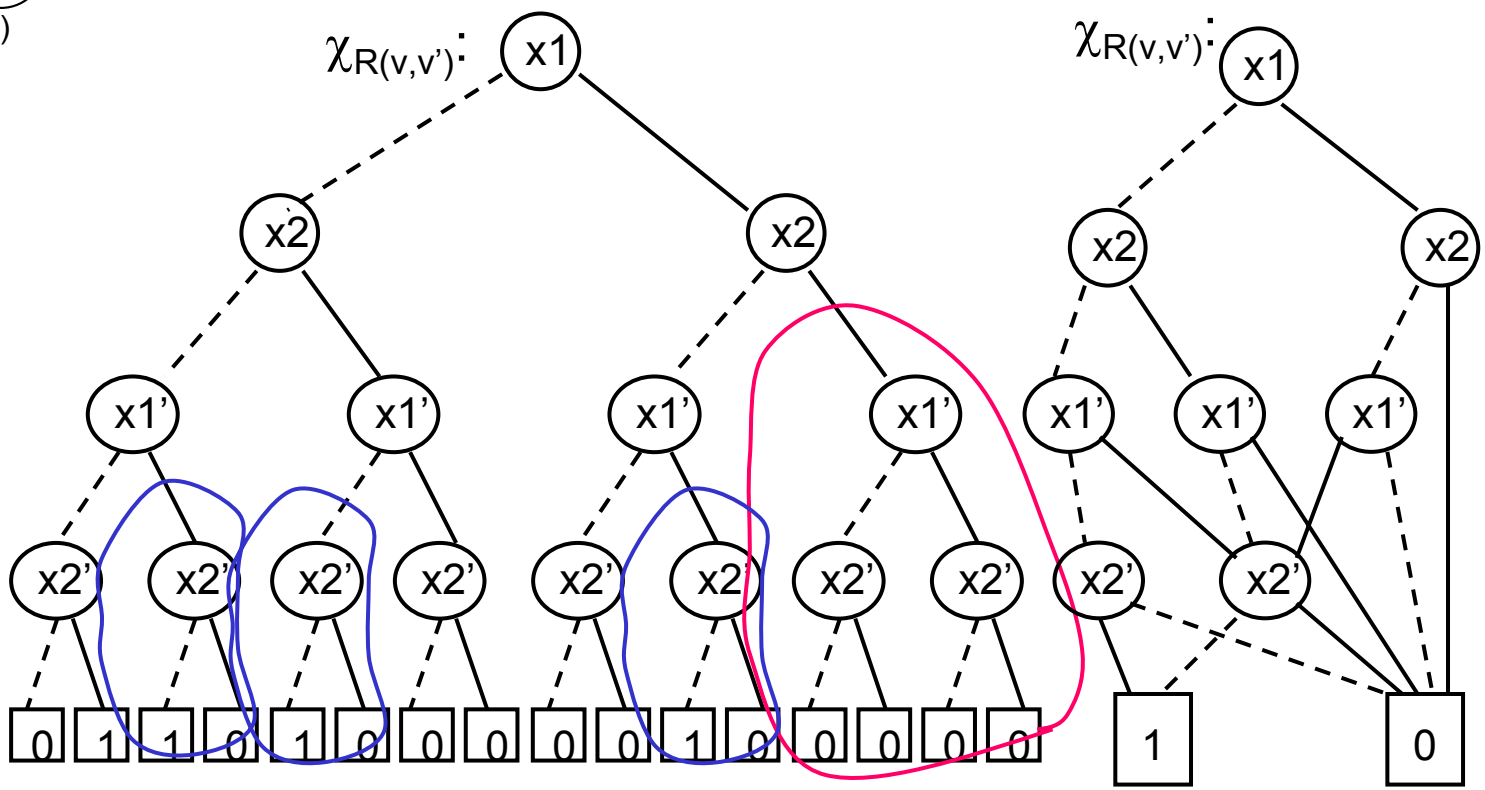


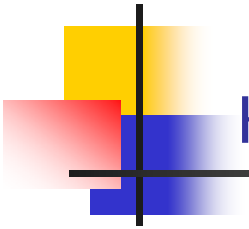

$v=\langle x_1, x_2\rangle$ , $v'=\langle x_1', x_2'\rangle$

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

Множество R = {0001, 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?

КАК представить структуру Крипке в форме BDD без построения самих состояний и переходов явно?

НУЖНО булевыми функциями задать только начальное состояние (множество начальных состояний) и переходы

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

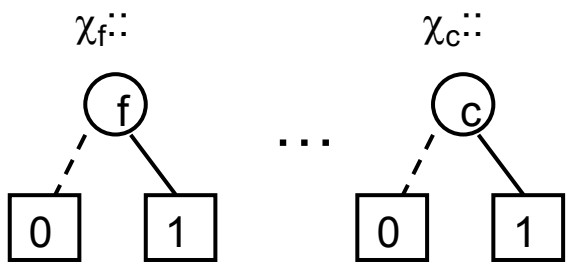
***Состояние*** – вектор значений переменных

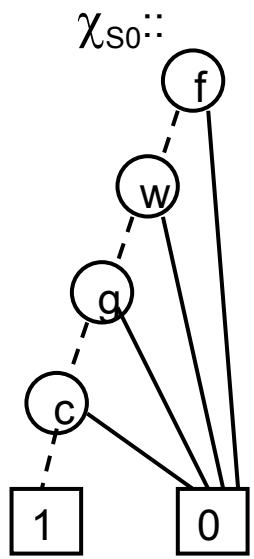
Булевы переменные – f, w, g, c.

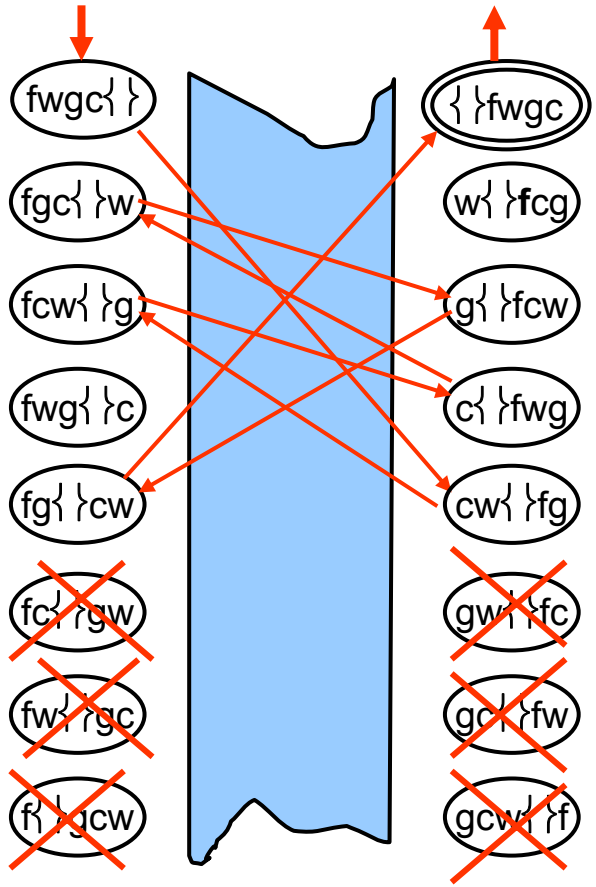
***Начальное состояние*** <0,0,0,0>; $\chi_{S0}$= ~f~w~g~c

***Атомарные предикаты***:

$\chi_f$ = f;
$\chi_w$ = w;
$\chi_g$ = g;
$\chi_c$ = c



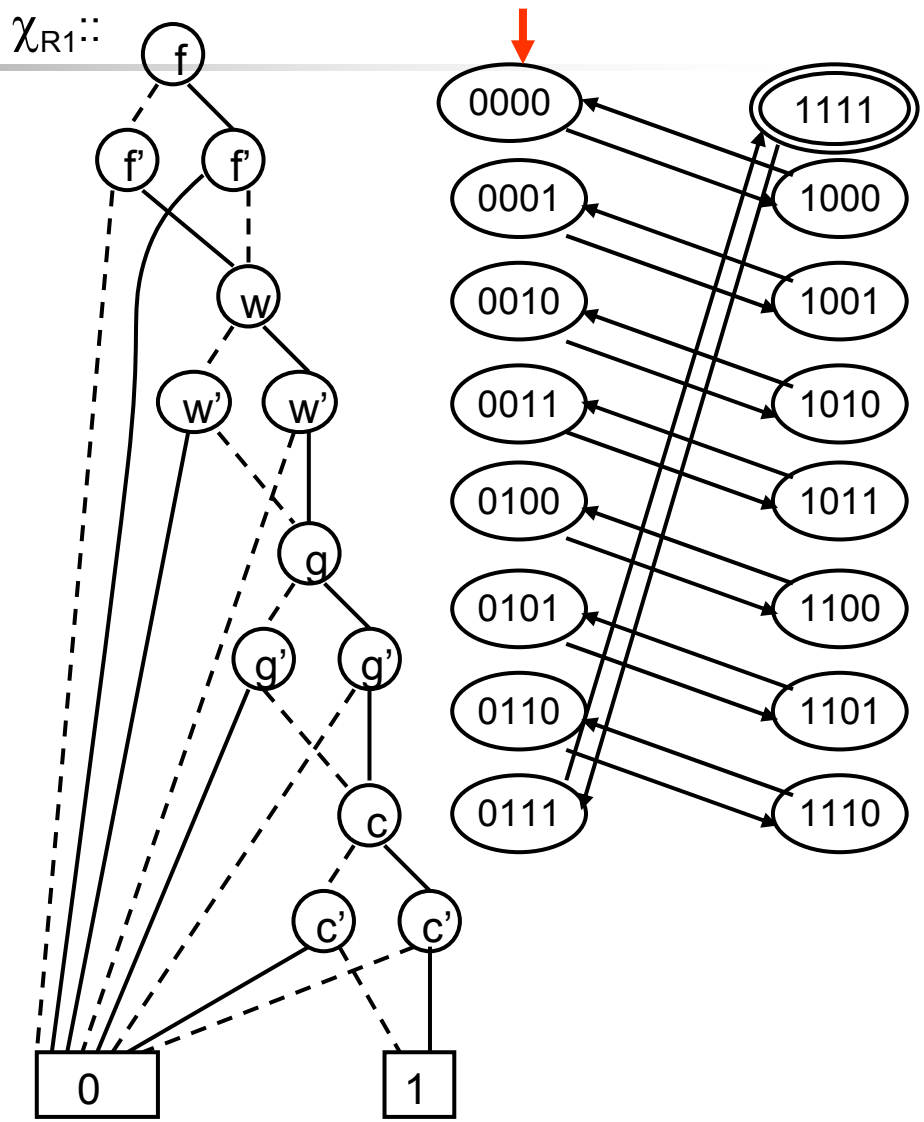




***Переходы:***

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 $\chi_{R4\,(v,\,v')} =$

$(f \equiv c)(f \equiv \sim f')(w \equiv w')(g \equiv g')(c \equiv \sim c')$

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

Но явно проводить ребра НЕ НАДО!

Они все описываются характеристической функцией

# Идея символьной верификации

*Синтаксис*: CTL (грамматика):
$\phi ::= p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid AX\phi \mid EX\phi \mid AG\phi \mid EG\phi \mid AF\phi \mid EF\phi \mid A[\phi_1 U \phi_2] \mid E[\phi_1 U \phi_2]$

*Формулы состояний:* Каждая CTL формула – это формула состояний, которая в каждом состоянии либо истинна, либо ложна
Для любой формулы CTL $\phi$ множество состояний $Q_\phi$, в которых эта формула истинна, будем задавать характеристической булевой функцией $\chi_{Q\phi}$
Все характеристические булевы функции будем представлять в BDD

Операции выполняются над характеристическими булевыми функциями

Задача символьной верификации состоит в следующем (возьмем конструкцию $EF\phi$):

ДАНО характеристическая функция множества состояний $Q_\phi$ структуры Крипке, на которых выполняется формула $\phi$.

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

# Вычисление операторов базиса: p, $\neg\phi$, $\phi 1 \vee \phi 2$

*Один из темпоральных базисов:*

EX $\phi$ – из данного состояния есть ребро в состояние, помеченное $\phi$

EG $\phi$ – из данного состояния есть путь, все состояния которого помечены $\phi$

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

*Синтаксис* CTL:

$\phi$ ::= p $|\neg\phi$ | $\phi_1 \vee \phi_2$ | EX$\phi$ | EG$\phi$ | E[ $\phi_1$U $\phi_2$ ]

*Рассмотрим формулы базиса* p, $\neg\phi$, $\phi 1 \vee \phi 2$ :

p – для каждого атомарного предиката p функция $\chi_{Q\,p}$, задающая множество состояний, в которых p истинен, определяется при постановке задачи

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

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

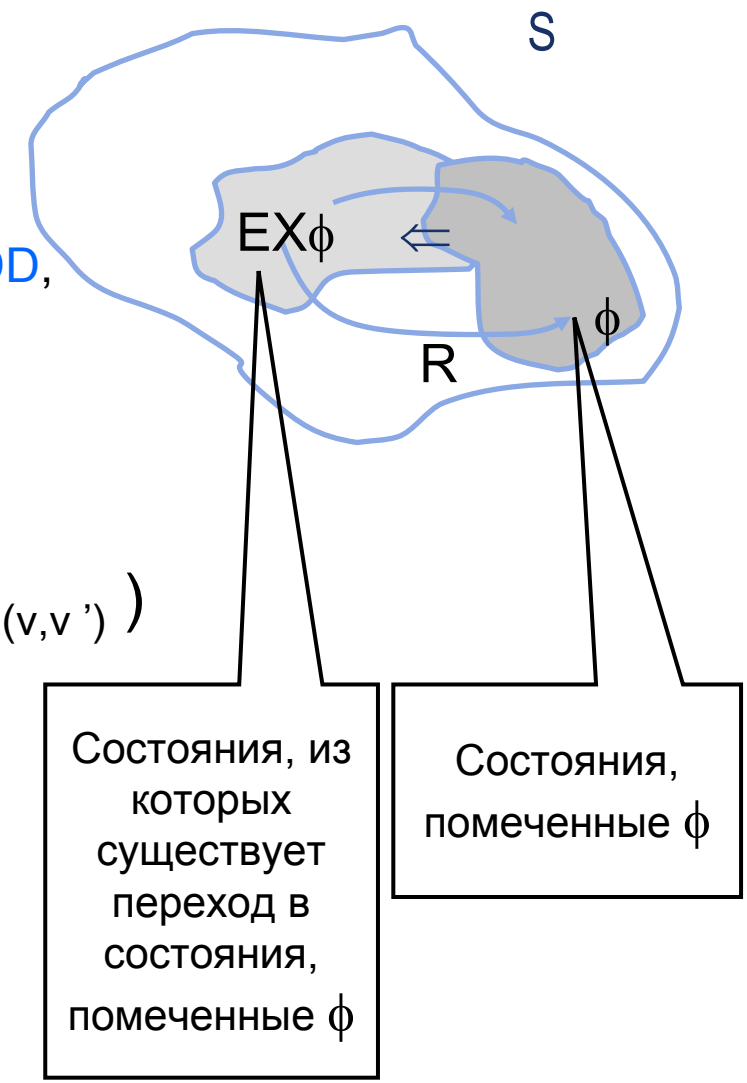
# Вычисление операторов базиса: EX$\phi$

*Синтаксис*: CTL (для одного из базисов):

$\phi ::= p \mid \neg\phi \mid \phi \vee \phi \mid EX\phi \mid EF\phi \mid E[\phi_1 U \phi_2]$

- EX $\phi$ реализуется с помощью операции "Обратный Образ" над булевыми ф-циями в BDD, представляющими ограничение отношения R структуры Крипке на подмножестве $Q_\phi$ тех состояний, в которых истинна формула $\phi$

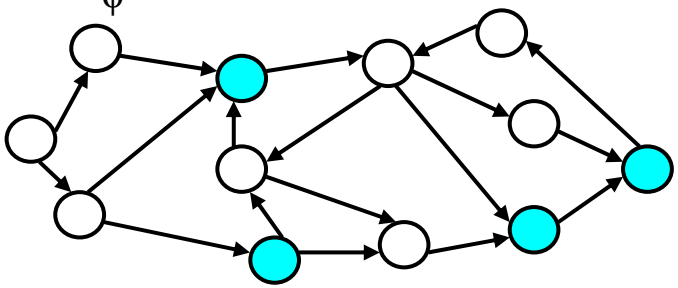
$$\chi_{RImage(Q\phi,R)(v)} = \exists v'.(\chi_{Q\phi(v)} \langle v'/v \rangle \,\&\, \chi_{R(v,v')})$$




Строим 'обратный образ' ограничения отношения R на множестве Q$\phi$ с использованием характеристических двоичных функций

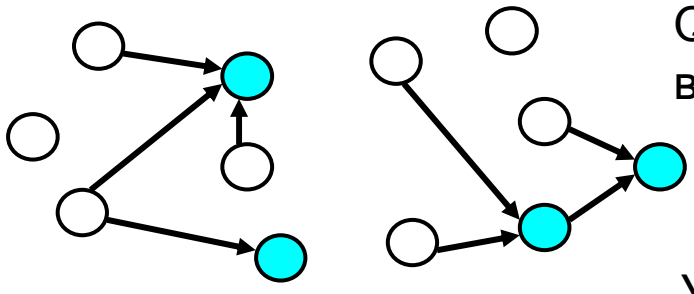
# Пример вычисления оператора EX$\phi$

$Q_\phi$ и R ::



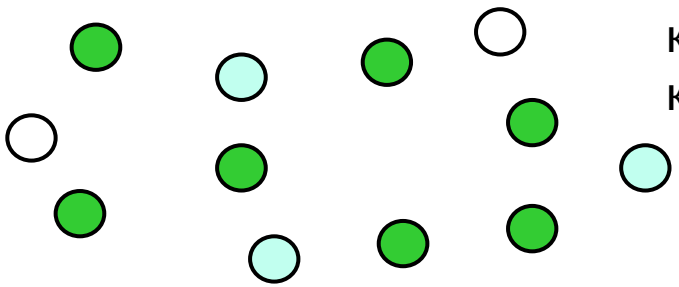
Состояния, удовлетворяющие $\phi$, помечены (множество $Q_\phi$ определено). Отношение R задано

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



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

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



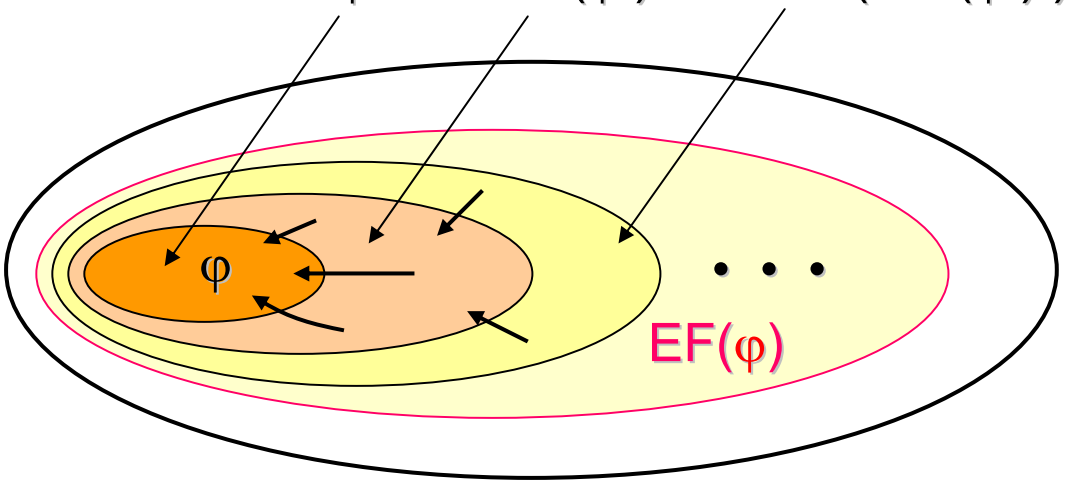
У оставшихся переходов берем только первую компоненту с помощью операции квантификации по переменным второй компоненты. Так получаем множество $Q_{EX\,\phi}$

Всего несколько операций с BDD, представляющими $Q_\phi$ и R

# Определение множества состояний, удовлетворяющих формуле EFφ

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

$EF(\varphi) \equiv$ состояния из которых можно достичь $\varphi$ - состояния, помеченные: $\varphi \cup EX(\varphi) \cup EX(EX(\varphi)) \cup \ldots$




$S_0 = S_\varphi$

$S_1 = S_0 \cup EX\,S_0$

$S_2 = S_1 \cup EX\,S_1$

...

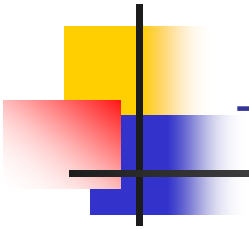
$S_{i+1} = S_i \cup EX\,S_i$

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 иногда будет циклический перебор подмножеств?

если пришли к пределу, будет ли это предельное множество тем, что мы хотели?



# Теория неподвижной точки преобразований множеств

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

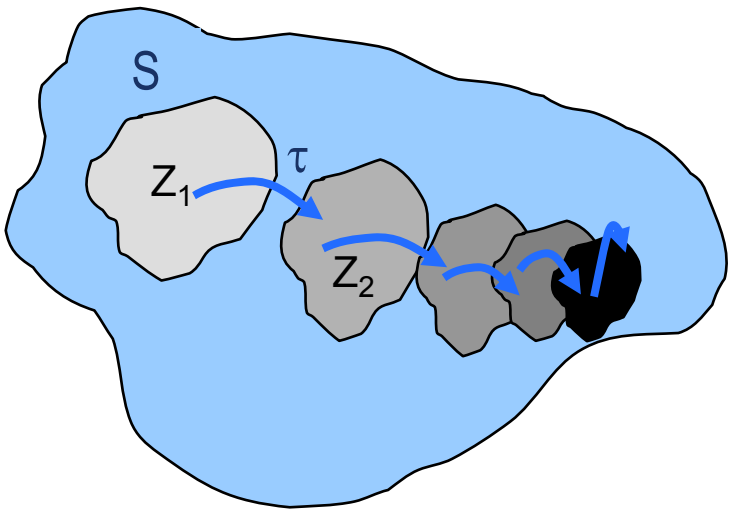
Ответ на эти вопросы дает теория неподвижной точки операторов на множествах

Теория фиксированной (неподвижной) точки нужна нам для обоснования алгоритмов для темпоральных операторов - $EF\phi$, $EG\ \phi$ и других

# Теория неподвижной точки (Fixpoint)

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

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




$$\tau: 2^S \to 2^S \quad \tau(Z_1) = Z_2$$

Что будет, если оператор $\tau$ на $2^S$ применить несколько раз?

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

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

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

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

# Операторы на конечных множествах

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

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

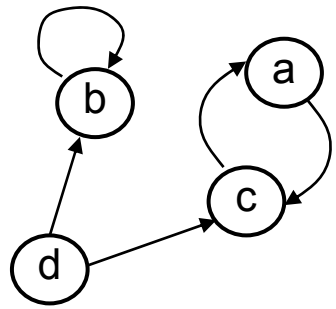
$\tau_1(\{a, d\}) = \{a, b, d\}$

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

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

$\tau_1(S) = S$ FP

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



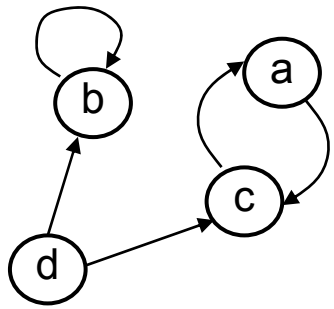

У $\tau_1$ несколько неподвижных точек

# Операторы на конечных множествах

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

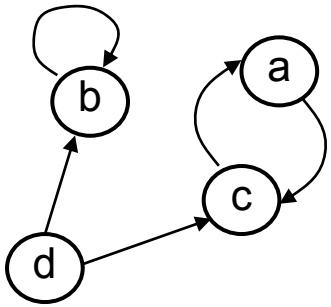
$\tau_2(Q) = \{c\} \cup \{a\} \cap Q$

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

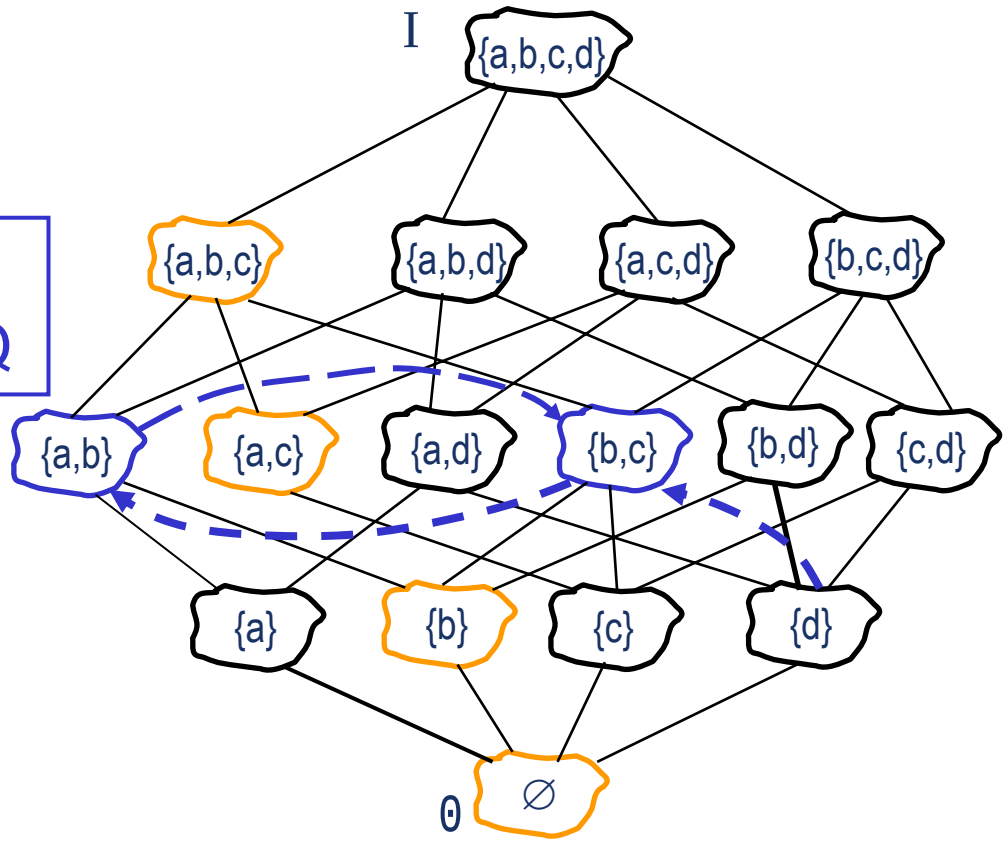



У $\tau_2$ две неподвижные точки

# Операторы на конечных множествах




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




$\tau_3(Q)=Q'$, где $Q'$ – преемники $Q$

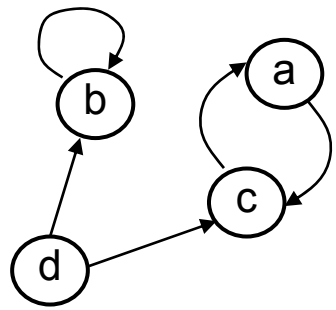
$\tau_3(\{d\}) = \{b, c\}$
$\tau_3(\{b, c\}) = \{a, b\}$
$\tau_3(\{a, b\}) = \{b, c\}$
$\tau_3(\{a, c\}) = \{a, c\}$ FP

Существуют множества, применение $\tau_3$ к которым приводит к циклическому повторению

# Операторы на конечных множествах

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

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

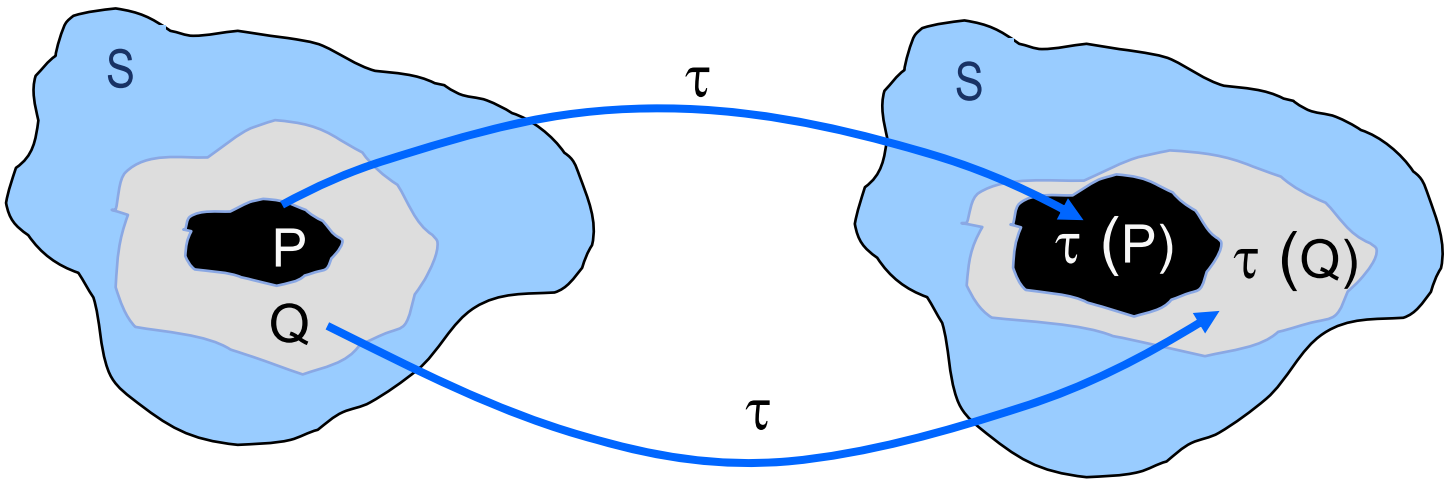



$\tau_4 ( \{d\} ) = \{d\}$
$\tau_4 ( \{c,d\} ) = \{c, d\}$ FP
$\tau_4 ( \{a,c\} ) = \varnothing$
$\tau_4 (\varnothing ) = \varnothing$ FP
$\tau_4 ( \{a\} ) = \{a\}$

Как разобраться в неподвижных точках??

# Монотонные операторы на конечных множествах

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

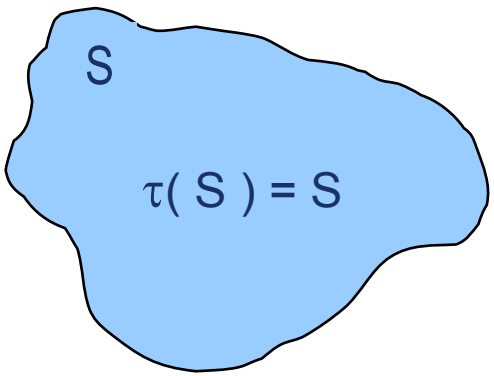



Композиция монотонных операторов -- монотонный оператор

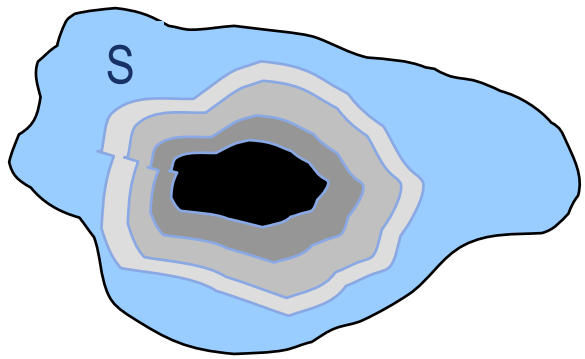
# Теорема Тарского о неподвижной точке

Что, если **МОНОТОННЫЙ** оператор $\tau$ 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 $\tau$,
$\tau^k(S) = S$




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

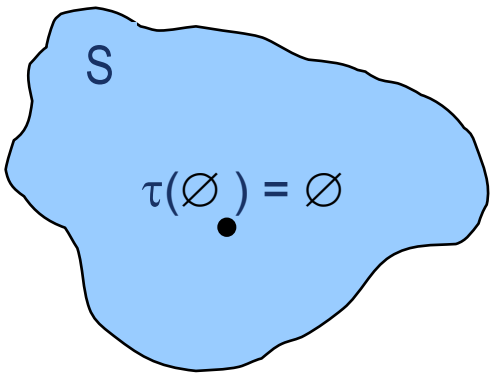
До каких пор?

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

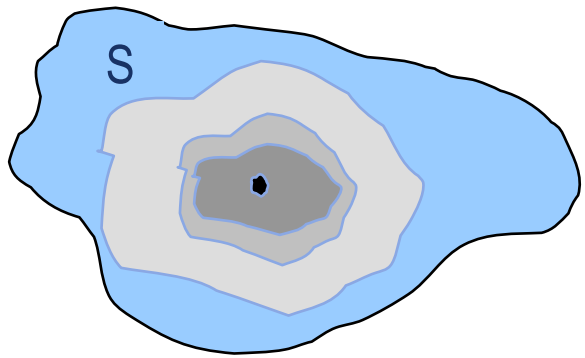
# Теорема Тарского о неподвижной точке

Что, если монотонный оператор $\tau$ 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 $\tau$,
$\tau^k(\varnothing) = \varnothing$




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?

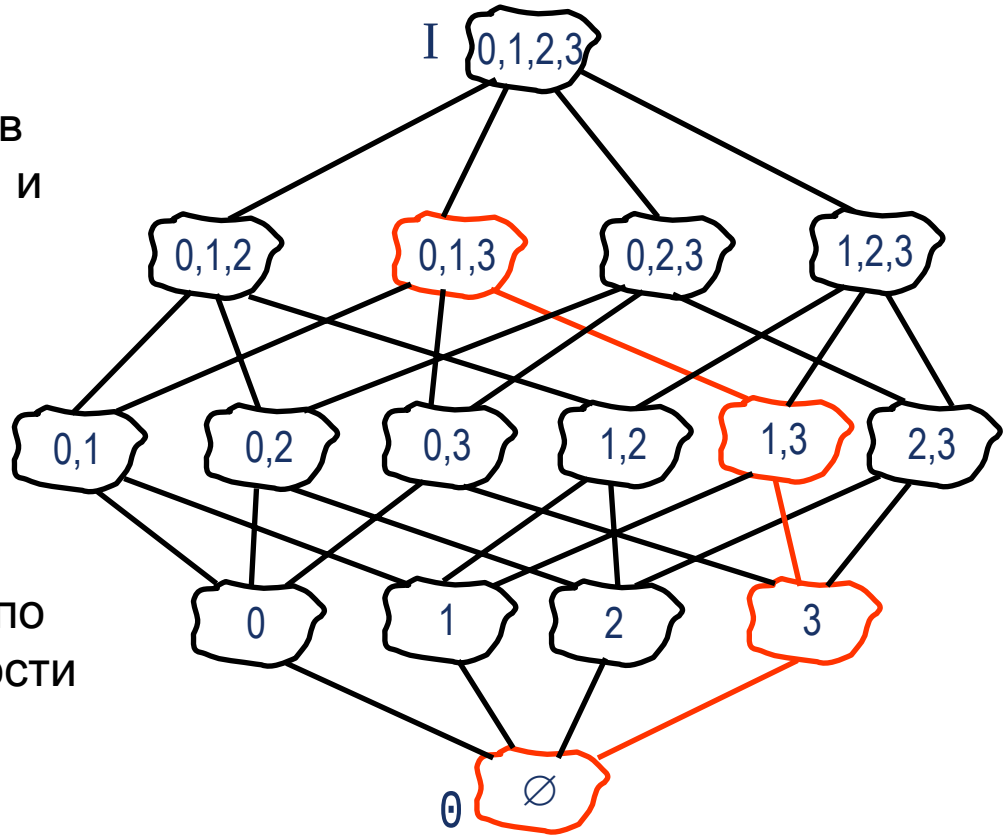
Поскольку множество S конечное, то $\varnothing$, $\tau(\varnothing)$, $\tau^2(\varnothing)$ … $\tau^k(\varnothing)$ придет к FP

# 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

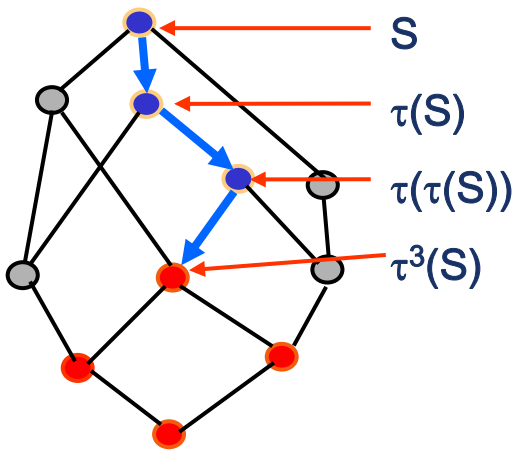
$$A \leq B \text{ iff } A \subseteq B$$

# 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 максимальную, и минимальную FP, которые находятся простыми алгоритмами

● неподвижная точка

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




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

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

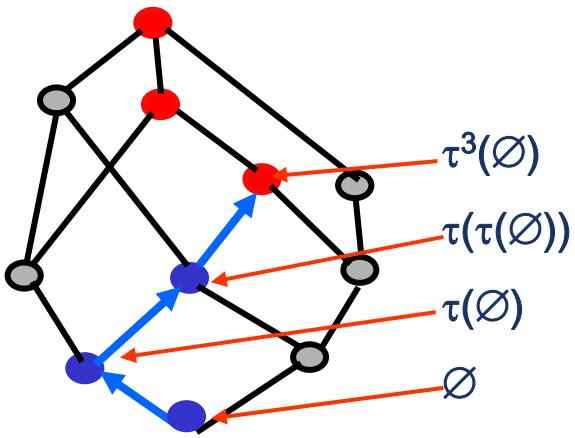
Во многих приложениях важны максимальные FP

# 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 максимальную, и минимальную FP, которые находятся простыми алгоритмами

● неподвижная точка

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



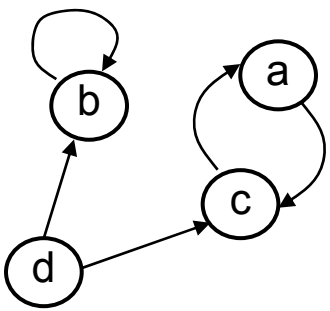

Расширяющий оператор

$\varnothing \subseteq \tau(\varnothing) \Rightarrow \tau(\varnothing) \subseteq \tau(\tau(\varnothing))$

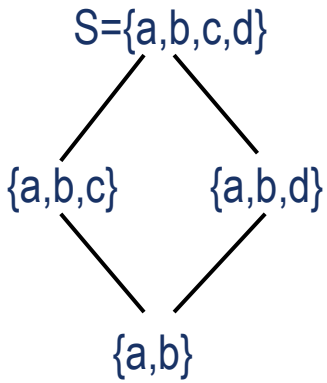
Минимальная неподвижная точка
$\mu S.\tau(S) = \tau^{\infty}(\varnothing)$

Во многих приложениях важны минимальные FP

# Решетки неподвижных точек операторов

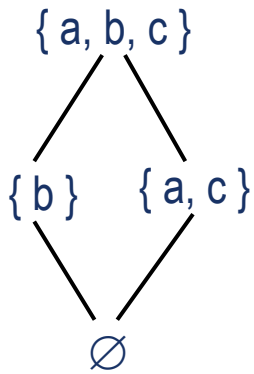



$\tau_1(Q) = \{a, b\} \cup Q$

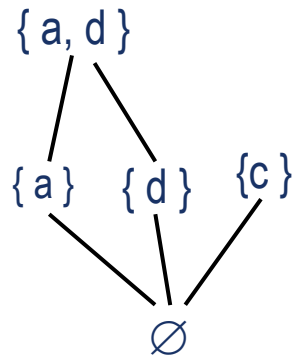



$\tau_2(Q) = \{c\} \cup \{a\} \cap Q$


$\tau_3(Q) = Q'$




$\tau_4(Q) = Q \setminus Q'$



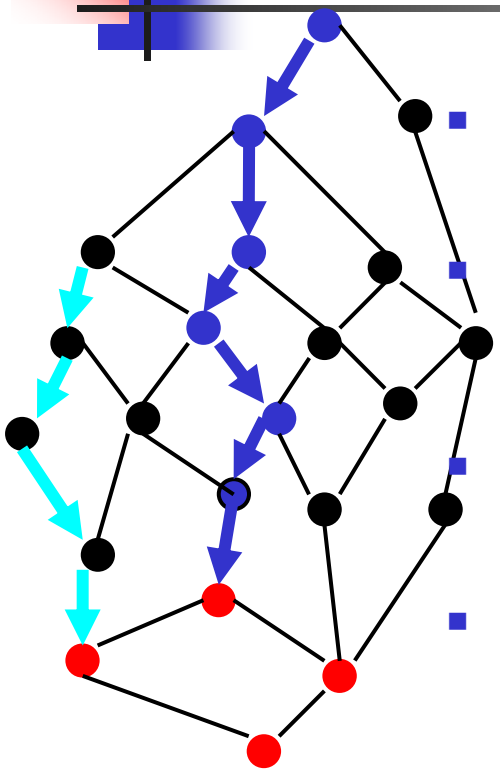

$\tau_1 - \tau_3$ монотонные операторы

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

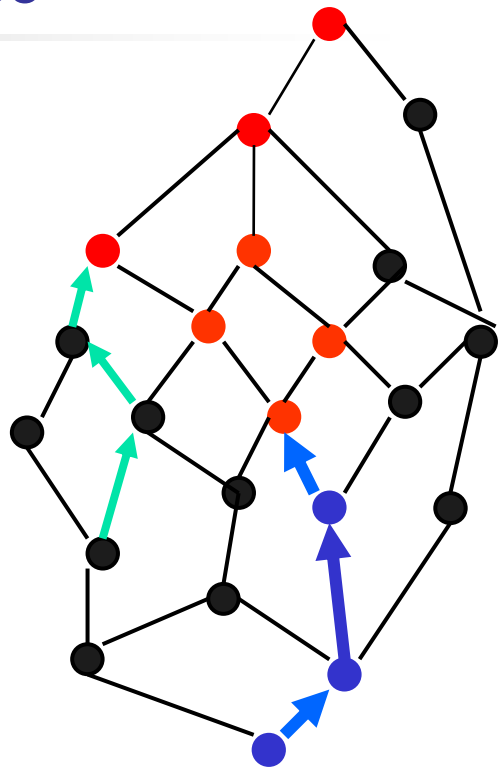
$\tau_4$ – не монотонный оператор

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

# 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
- Максимальная неподвижная точка монотонного оператора может быть найдена как предел $\tau^k(S)$
- Минимальная неподвижная точка монотонного оператора может быть найдена как предел $\tau^k(\varnothing)$



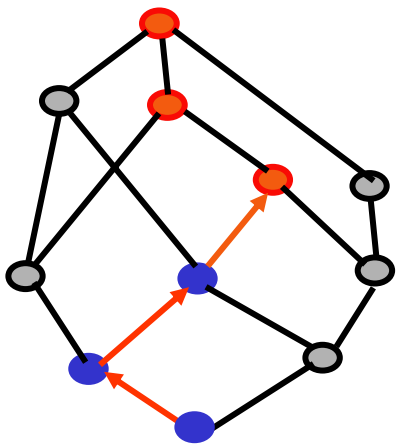
Доказательство. Для любого k, $\tau^k(\varnothing) \subseteq \tau^{k+1}(\varnothing)$ (база: $\varnothing \subseteq \tau(\varnothing)$, шаг индукции – по монотонности $\tau$). Пусть $M = \tau^\infty(\varnothing)$. Поскольку S- конечное множество, то $\tau^k(\varnothing) = \tau^{k+1}(\varnothing)$ с какого-то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

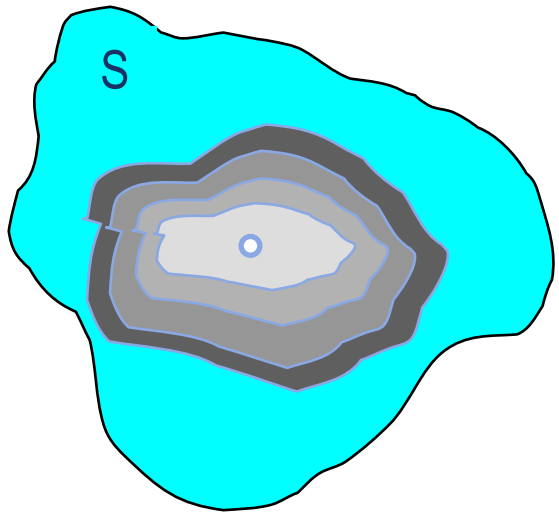
Докажем, что $M \subseteq$ любой другой FP M*. Применим k раз к обеим частям очевидного неравенства $\varnothing \subseteq M^*$ оператор $\tau$: $\tau^k(\varnothing) = M \subseteq \tau^k(M^*) = M^*$ □

# Вычисление минимальной неподвижной точки Least Fixed Point (LFP)

$\mu S.\ \tau(S) = U^{\infty}_{k=0}\ \tau^k(\varnothing)$

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






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

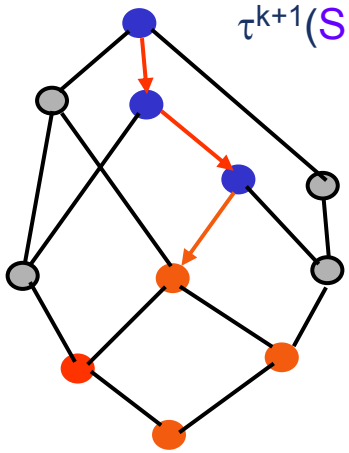
Минимальная неподвижная точка монотонного оператора $\tau$ вычисляется как предел последовательности:

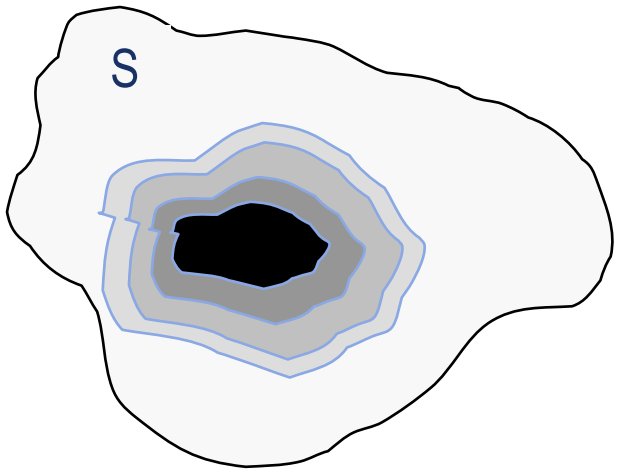
$\varnothing \subseteq \tau(\varnothing) \subseteq \tau(\tau(\varnothing))$ ...

Эта последовательность всегда имеет предел – Least Fixed Point

# Вычисление максимальной неподвижной точки Greatest Fixed Point (GFP)

$\nu S.\ \tau(S) = \cap^{\infty}_{k=0} \tau^k(S)$




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

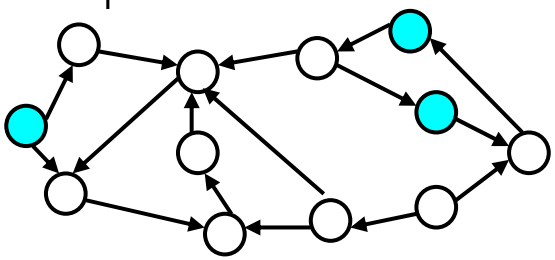
Максимальная неподвижная точка монотонного оператора $\tau$ вычисляется как предел последовательности:
$S \supseteq \tau(S) \supseteq \tau(\tau(S)) \supseteq \ldots$

Эта последовательность всегда имеет предел – Greatest Fixed Point

# Пример: Множество $Q_{EFf}$ как минимальная неподвижная точка оператора

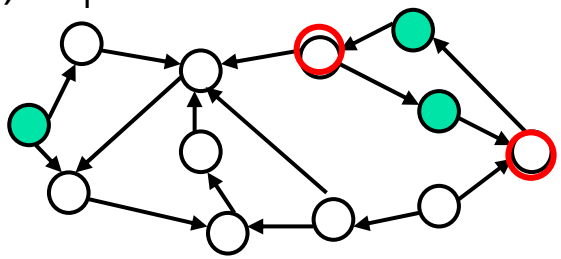
Состояния, которые помечены EFf - это состояния, помеченные f, + все те, из которых за один шаг можно достичь состояния, помеченного EFf, причем нам нужно построить МИНИМАЛЬНОЕ множество таких состояний

$Q_f$ ::



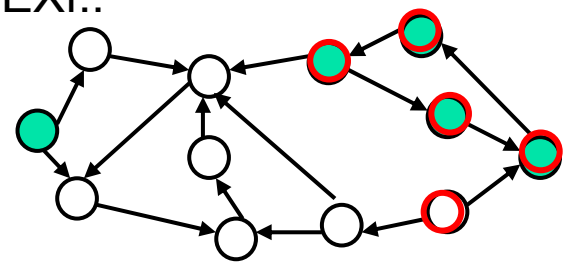
$$\tau(Q) = Q_f \cup EX(Q)$$

$\tau^1(\varnothing)=Q_f$::

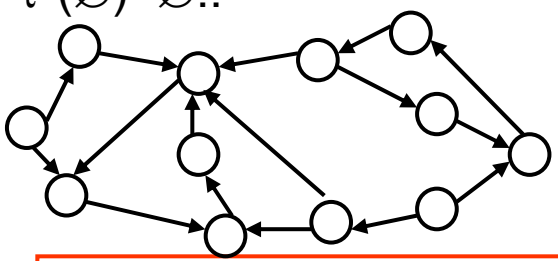


Минимальная FP определяется как предел $\varnothing \subseteq \tau(\varnothing) \subseteq \tau^2(\varnothing) \subseteq \tau^3(\varnothing) \subseteq \ldots$

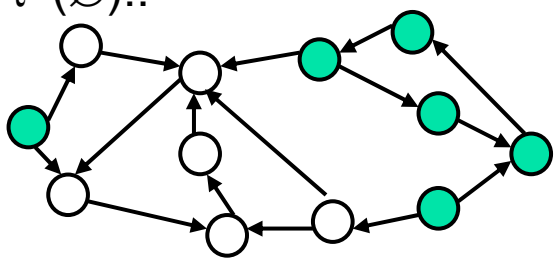
$\tau^2(\varnothing)= Q_f \cup EXf$::



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



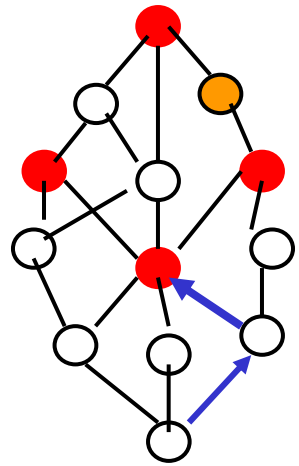
$\tau^3(\varnothing)= Q_f \cup EX\tau^2(\varnothing)= \tau^\infty(\varnothing)$::

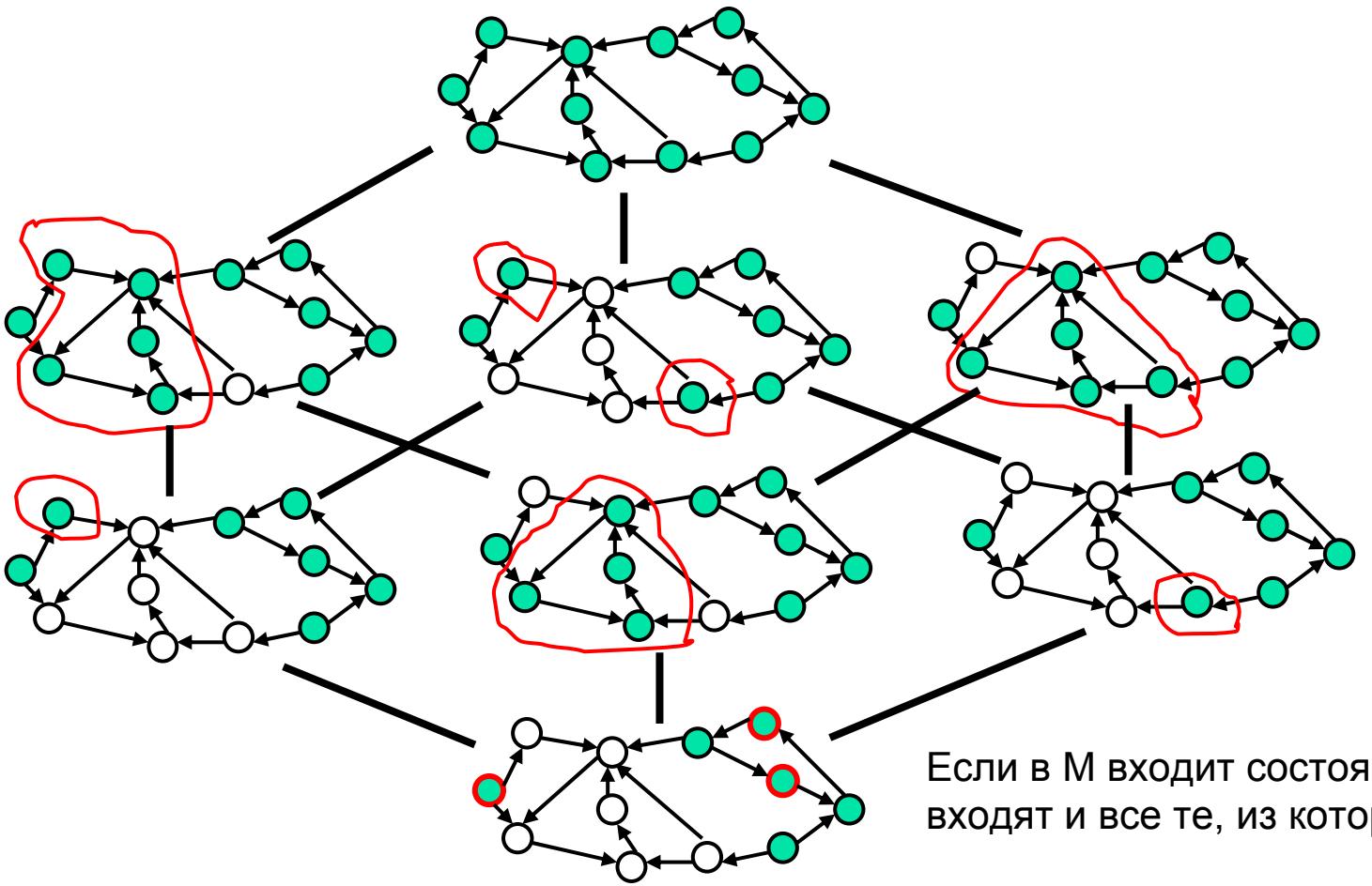


Красным обведены состояния EX(помеч), т.е. те, из которых можно достичь помеченные за 1 шаг

# Решетка неподвижных точек оператора $\tau(Q) = Q_f \cup EX(Q)$

Максимальная неподвижная точка – все множество S



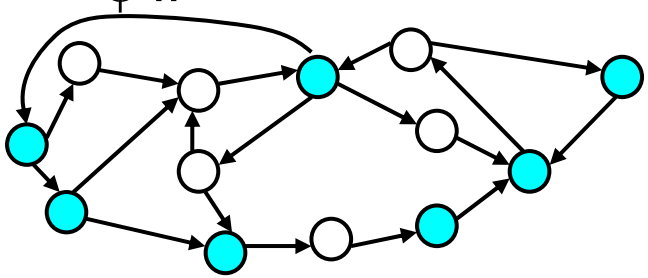


Если в M входит состояние v, то в M входят и все те, из которых v достижимо

Минимальная неподвижная точка – искомое множество EFf

# Пример: максимальная неподвижная точка оператора (2)

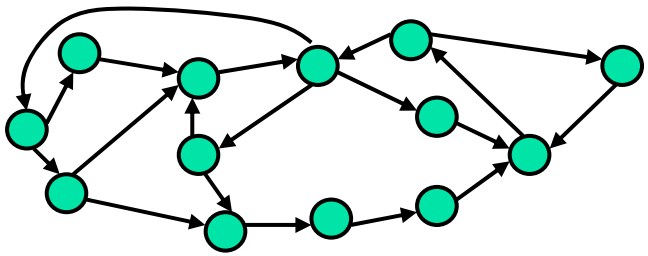
$\varphi$::



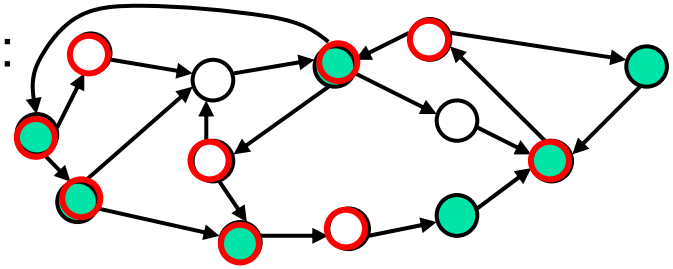
$\tau(Q) = \varphi \cap EXEX\ Q$

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

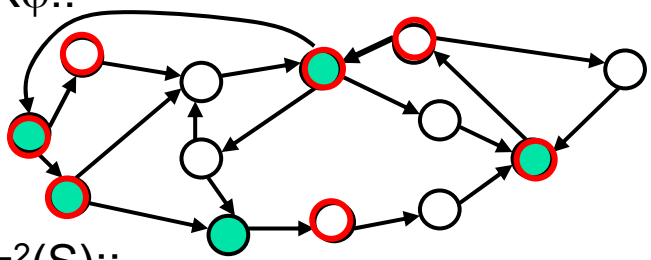
$\tau^0(S)=S$::



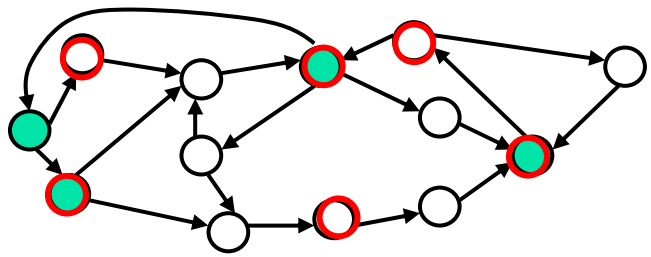
$\tau^1(S)=\varphi$::



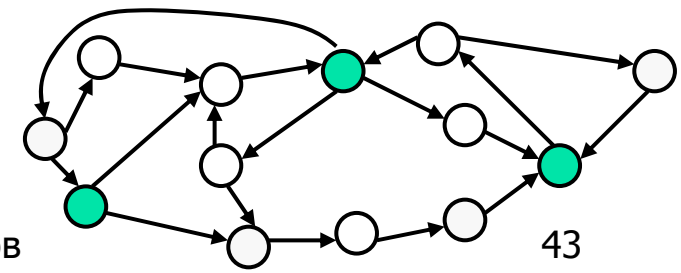
$\tau^2(S)= \varphi \cap EXEX\varphi$::



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



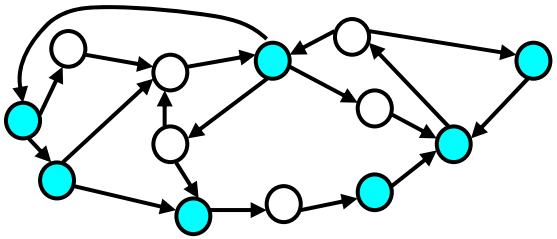
$\tau^4(S) = \varphi \cap EXEX\tau^3(S) = \tau^\infty(S)$::



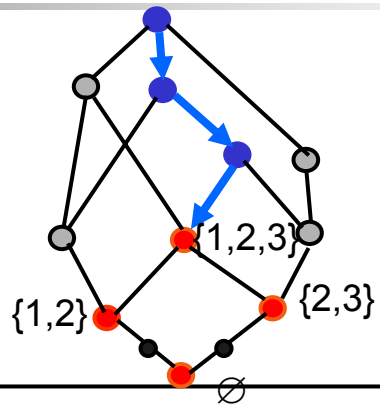
Красным обведены состояния EXEX $\varphi$, т.е. те, из которых можно достичь помеченные $\varphi$ за 2 шага

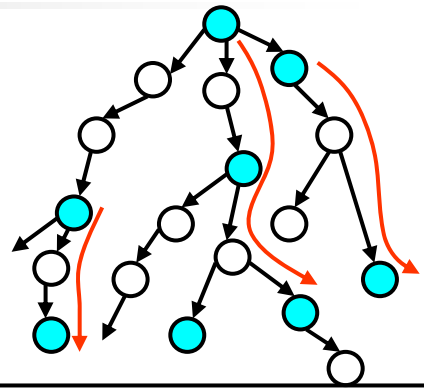
# Решетка неподвижных точек

$Q_\Phi$::



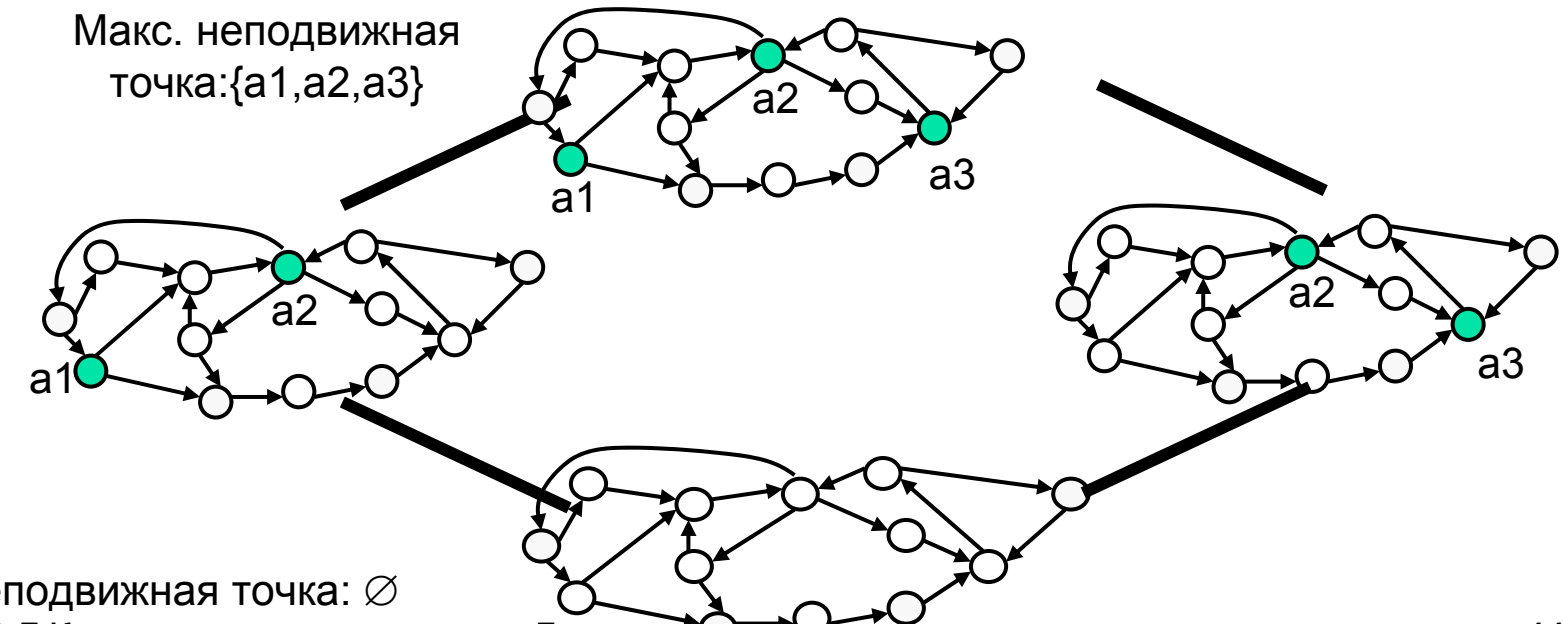
Четыре решения – неподвижные точки



Множества состояний, из которых существует путь, на котором Ф выполняется на каждом расстоянии четной длины от этих состояний, образуют решетку

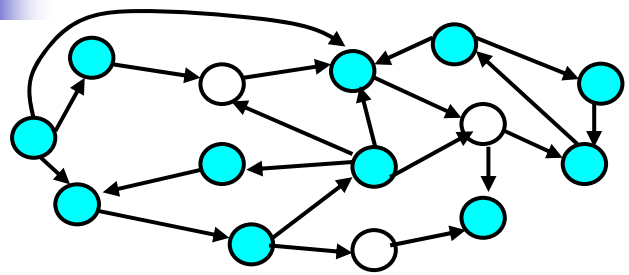
Макс. неподвижная точка:{a1,a2,a3}

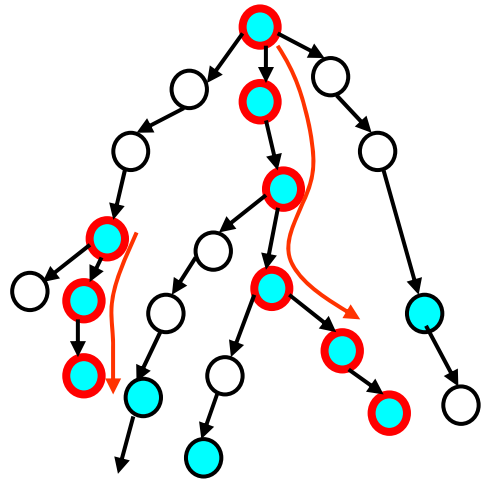



Мин. неподвижная точка: ∅

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

$Q_\varphi$ ::





На структуре Крипке задано множество состояний, в которых истинна формула $\varphi$. Найти множество *M* таких состояний, на которых истинна формула EGФ

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

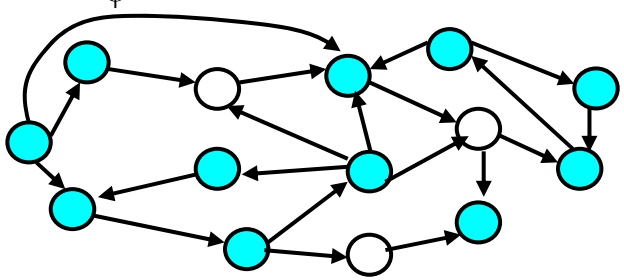
Это значит, что *M* является неподвижной точкой оператора $\tau(Q)=Q_\varphi \cap EXQ$

Оператор $\tau(Q)$ - монотонный, он имеет минимальную и максимальную FP

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

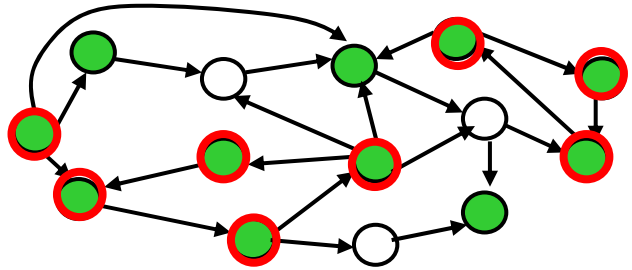
# Пример: максимальная неподвижная точка оператора $\tau(Q) = Q\varphi \cap EX\ Q$

$Q_\varphi$ ::

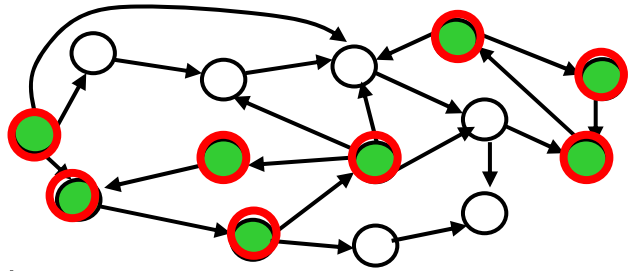


$\tau(Q) = Q_\varphi \cap EX\ Q$

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

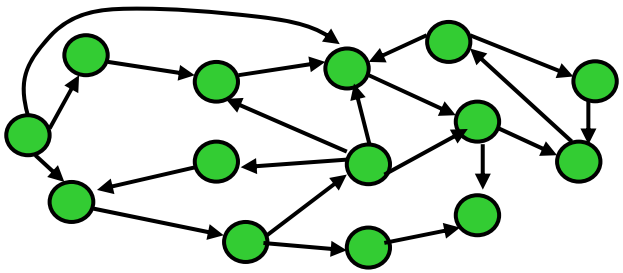


$\tau^2(S)= Q_\varphi \cap EXQ_\varphi$::

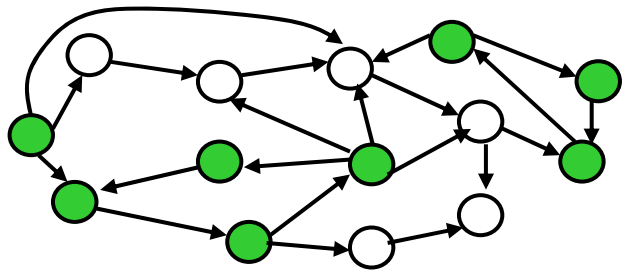


Максимальная FP определяется как предел $S \supseteq \tau(S) \supseteq \tau^2(S) \supseteq \tau^3(S) \supseteq \ldots$

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



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

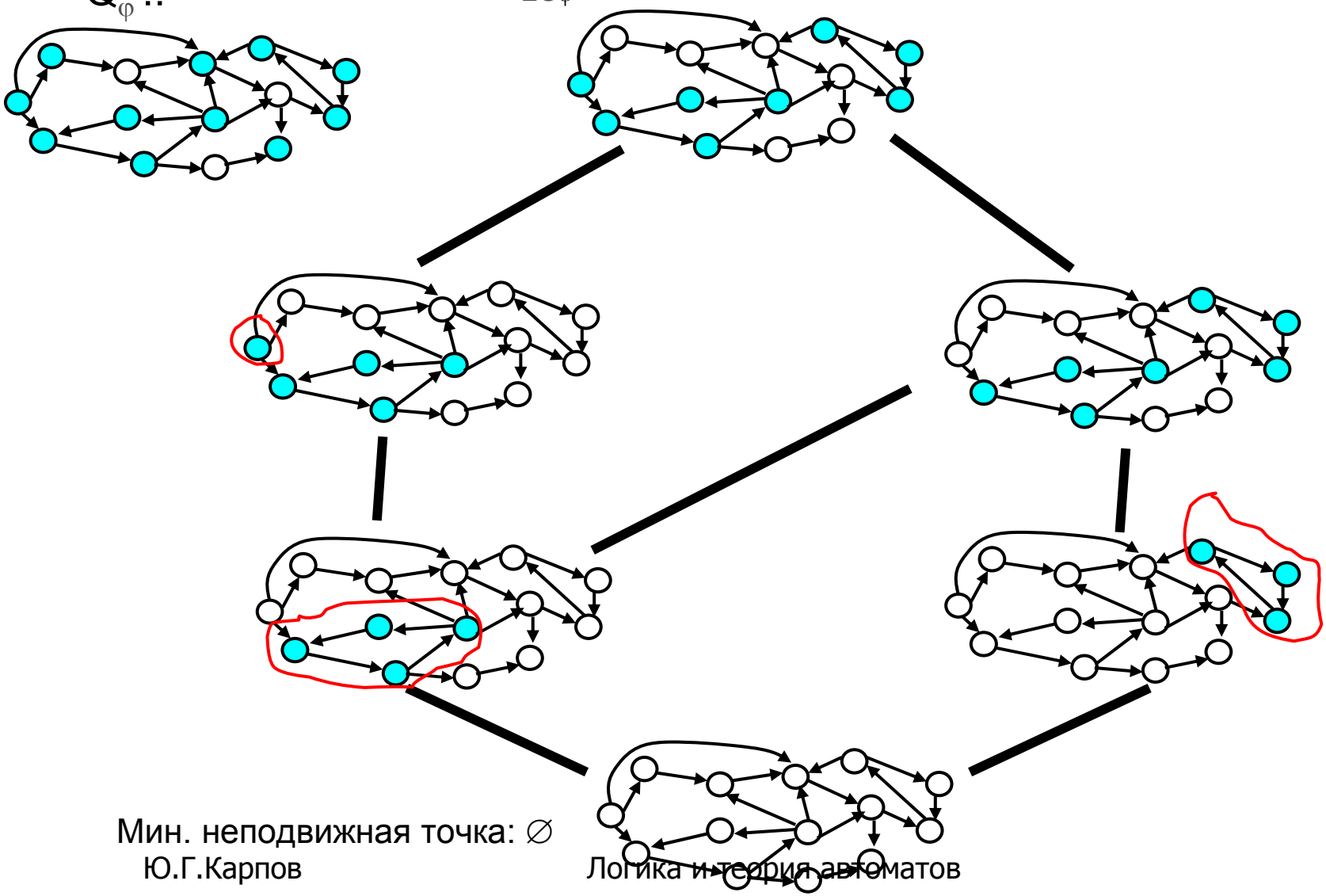


Красным обведены состояния EXΦ, (из которых можно достичь помеченные Φ за 1 шаг

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

$Q_\varphi$ ::

$Q_{EG\varphi}$ - Максимальная неподвижная точка



Мин. неподвижная точка: ∅

# 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

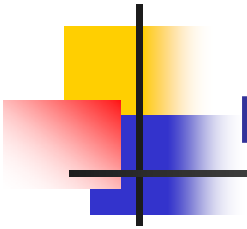
Операции $\cup$, $\cap$, EX и AX

*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

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



# 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

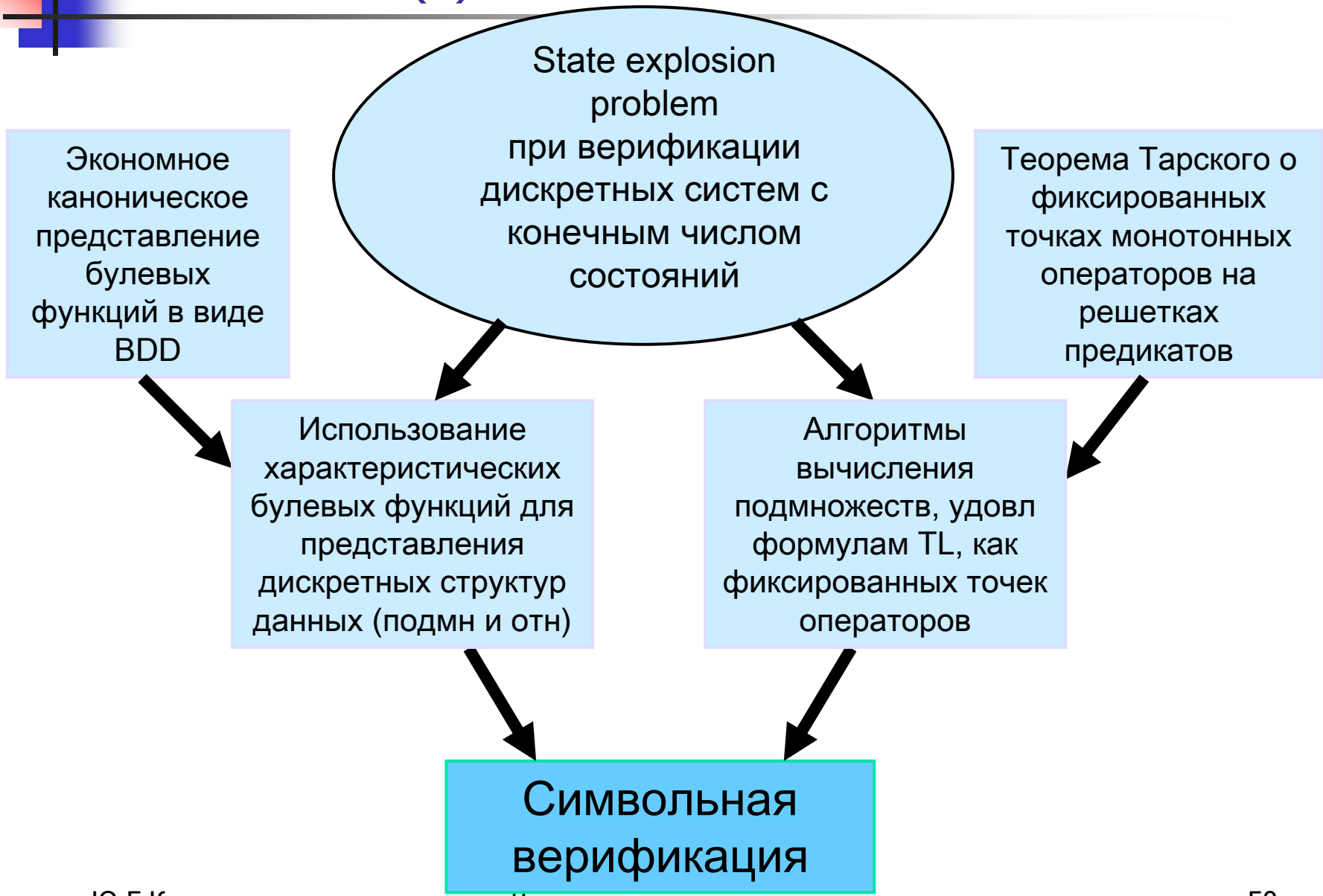
На практике число итераций до достижения FP невелико. Символьный метод позволяет увеличить размерность анализируемых систем от $10^6$ до $10^{20}$ - $10^{30}$. Опубликованы работы, в которых описывался опыт верификации систем с $10^{130}$ состояний и даже $10^{300}$ состояний

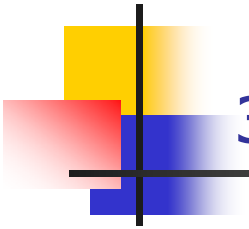
Это комбинаторные числа: число атомов во вселенной менее $10^{80}$

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



# Заключение (1)

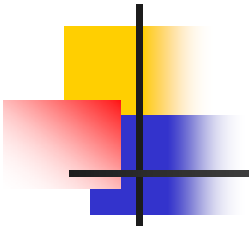




# Заключение (2)

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