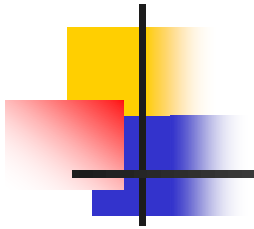
Лекция 14

Верификация параллельных программных и аппаратных систем

Курс лекций

Шошмина Ирина Владимировна

Карпов Юрий Глебович



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. Система верификации Spin и язык Promela. Примеры верификации
8. Структура Крипке как модель реагирующих систем
9. Темпоральные свойства систем
10. Применения метода верификации model checking
11. BDD и их применение
12. Символьная проверка моделей
13. Количественный анализ дискретных систем
14. **Верификация систем реального времени**

AG(*дернул* → AF*раскрылся*)

Общие положения

- Model Checking и структуры Крипке позволяют анализировать свойства систем без явного указания времени. ***Для систем реального времени такие требования бесполезны!***
- Системы реального времени – такие, в которых явные ***значения временных интервалов между событиями существенны***, они влияют на свойства системы.
- Safety-critical systems – пропуск временной границы между двумя событиями может привести к аварии:
  - ***требование*** CTL к системе управления парашютом "AG(*дернул* ⇒ AF*раскрылся*) – *если дернул за кольцо, то когда-нибудь в будущем парашют раскроется*"
  - ***требование*** к подушке безопасности автомобиля "Airbag must be open within 5ms after the car crashes"

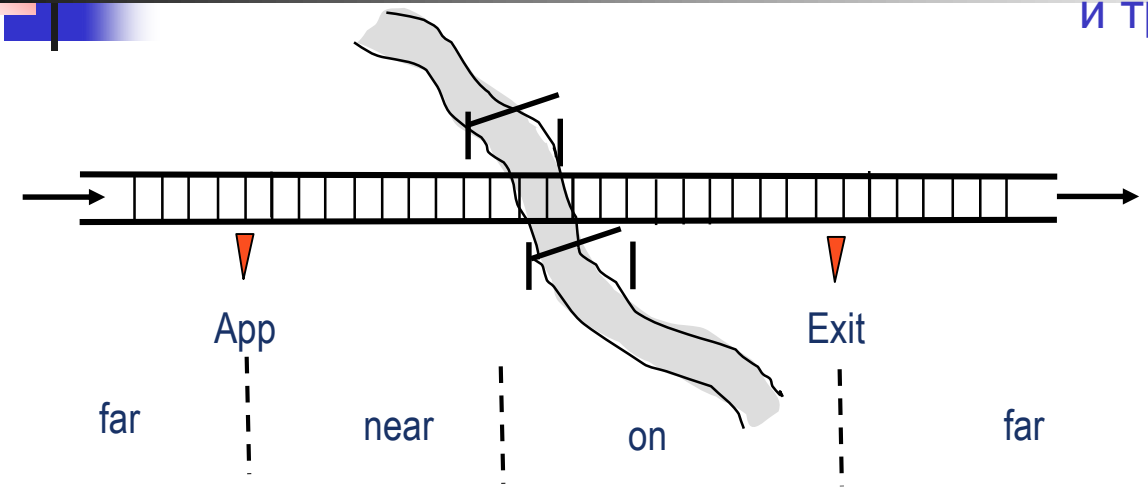
дернул кольцо

когда-нибудь в будущем раскрылся

- Необходимо расширение базовых моделей, используемых в Model Checking: введение реального времени. Впервые такие модели введены в 1994 г.
- Сейчас проблема широко исследуется, существуют пакеты верификации систем реального времени, например, UPPAAL и KRONOS.

Пример: ж.д. переезд

Нужно формально представить и систему, и требования к ее поведению в реальном времени



- Описание системы управления:
  - *Ж.д. переезд оснащен шлагбаумом. Поезд извещает контроллер шлагбаума о своем приближении >= 2 мин до пересечения переезда, и уйдет не более, чем через 5 мин после пересечения.*
  - *Через 1 мин после получения извещения контроллер начинает закрывать шлагбаум. На закрывание нужна 1 минута.*
  - *Не позже, чем через 1 минуту после того, как поезд прошел, контроллер начинает поднимать шлагбаум, на что требуется от 1 до 2 минут.*
- Доказать:
  - *Всегда, когда поезд проходит переезд, шлагбаум закрыт .*
  - *Шлагбаум закрыт не более, чем на 10 минут.*

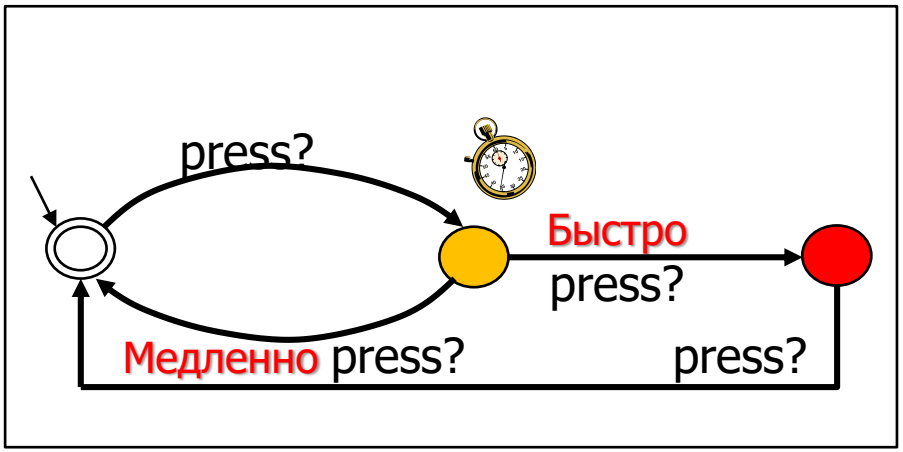
Идея: производство интеллектуальных выключателей

- Однократное нажатие – бледный свет лампы.
- *Двойное быстрое нажатие* – яркий свет.
- Следующее нажатие – погасить лампу.

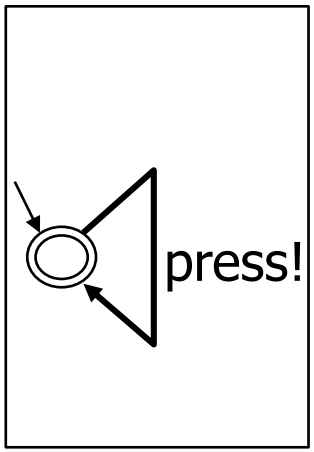
Как построить?

Как формализовать?

Нажатие клавиши – мгновенное событие



Лампа



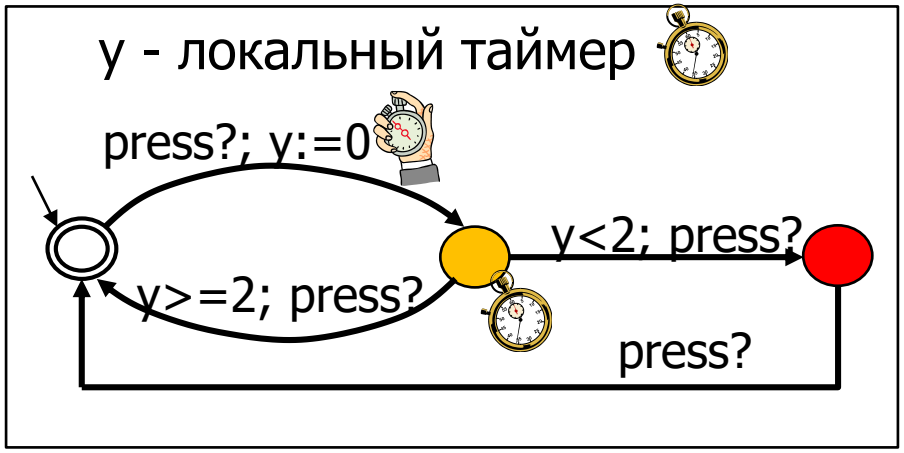
Пользователь

Поставить локальный таймер и в условиях перехода из одного состояния в другое учесть время – значение таймера.

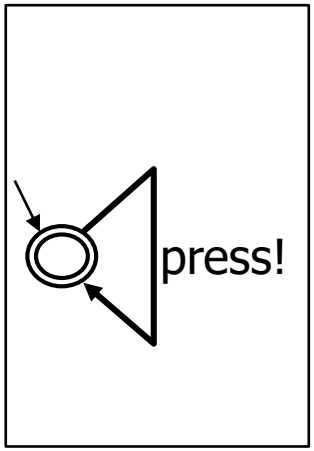
Формализация: временн'ые автоматы (Timed Automata)

- Необходим формализм, позволяющий конечным способом задать поведение, зависящее от реального времени

Вводилось много разных формализмов, чтобы адекватно представить поведение дискретных систем реального времени.
Timed automata оказался очень удачным



Лампа



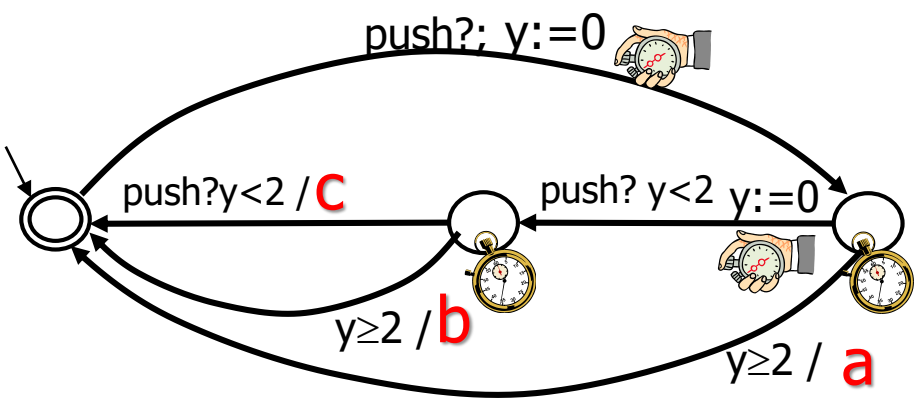
Пользователь

Решение: ввести локальные таймеры *и защиты переходов*, являющиеся условиями над значениями локальных таймеров.

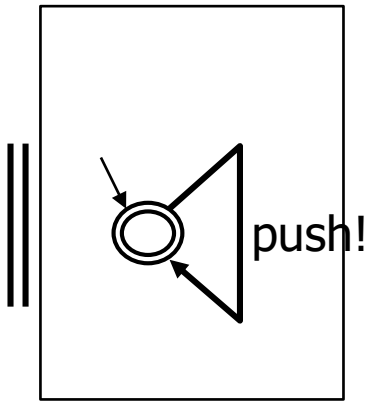
Интеллектуальная кнопка мобильника

- Как построить автомат с поведением:
- Одно нажатие – a
- Двойное быстрое нажатие – b
- Тройное быстрое нажатие – c
- Два медленных нажатия – aa

push



Клавиатура



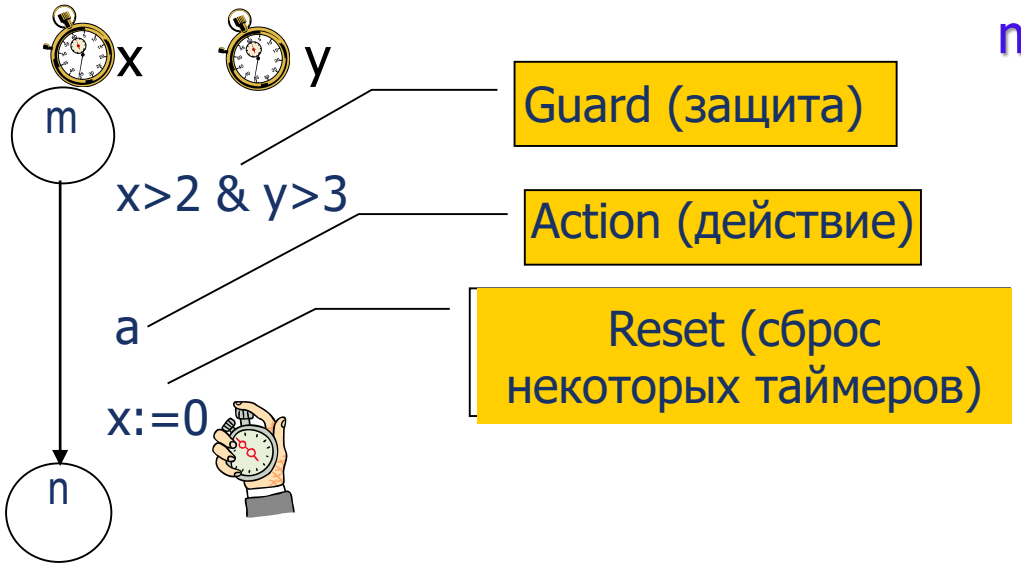
Пользователь

Временн'ые автоматы - определение

Timed Automaton – это:

конечный автомат (система переходов с помеченными локациями)

+ конечное число локальных таймеров
+ условие на переходах (Guard), зависящее от значений часов
+ сброс некоторых таймеров на переходах



m и n – не состояния, а локации

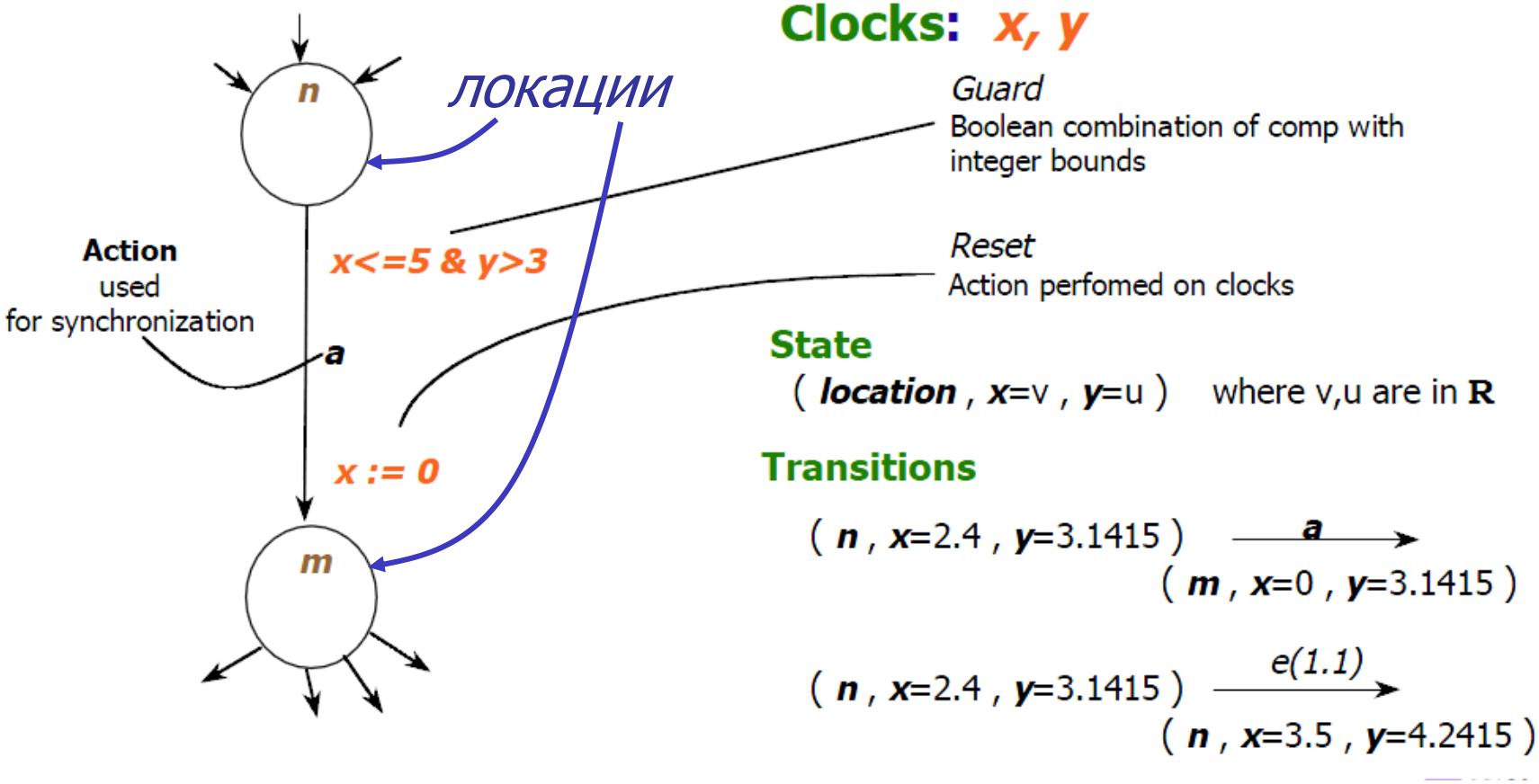
Состояние – это локация + значения всех параметров. Во временн'ом автомате показания таймеров также являются параметрами

Здесь возможные состояния:

(m, x=0.1, y= 2.24)
(m, x=2.2, y= 4.34939)
(n, x=0, y= 4.34939)
(n, x=0.21, y= 4.55939) и т.д.

На переходе: *защита, действие, сброс некоторых таймеров*

Пример перехода

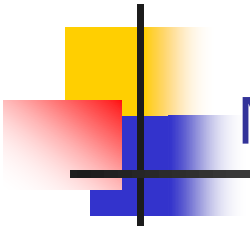


Переходы делятся на два типа:

*а) **переход по событиям** (обычно с изменением локации)*

*б) **переход по времени** (оставаясь в той же локации)*

Переходы между локациями время не увеличивают

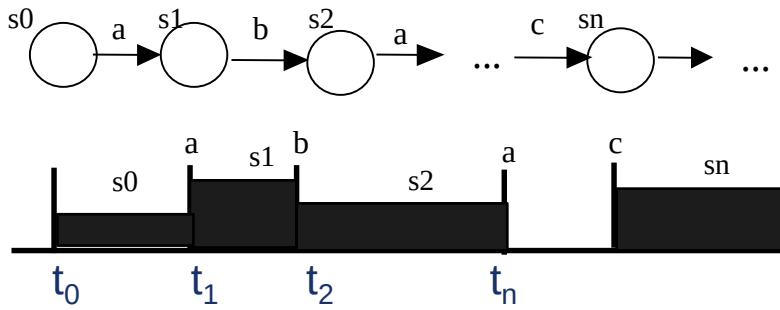


Мгновенность событий

- Будем считать, что конкретное событие в жизни временных автоматов происходит мгновенно, т. е. является элементарным действием, не имеющим протяженности во времени
- Протяженное, т. е. требующее времени, действие будем рассматривать как пару событий,
  - первое из которых отмечает начало действия,
  - а второе – его завершение.
  - Продолжительность действия определяется интервалом между наступлением его события-начала и наступлением его события-завершения; в течение всего этого времени могут происходить другие события
- Два протяженных действия могут перекрываться по времени, если начало каждого из них предшествует завершению другого.

Как описать переходы временн'ого автомата?

Поведение не временн'ого автомата:



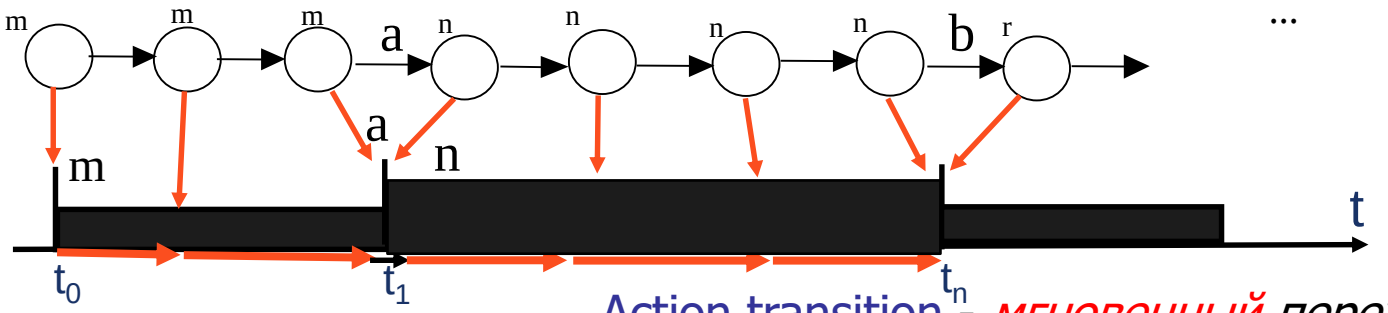
Состояние – что это?

Состояние=локация+конкретные значения таймеров

Число состояний бесконечно!

Поведение временн'ого автомата:

Переходы мгновенны



$(m, x=2.4, y=3.25) \xrightarrow{a} (n, x=0, y=3.25)$

Action transition - *мгновенный переход в новое состояние, значения всех таймеров не меняются кроме тех, которые явно сбрасываются*

$(m, x=1.0, y=1.85) \xrightarrow{1.4} (m, x=2.4, y=3.25)$

Timed transition – *переход по времени – значения всех таймеров синхронно увеличиваются*

Итак, все *clock constrains* представляем только с целыми границами

Формальное определение временн'ого автомата

Обозначим Φ(X) – **множество clock constraints.** Это выражения вида:

$\alpha ::= x<k \mid x>k \mid \neg\alpha \mid \alpha\wedge\alpha$, ***k – рациональное число*** (***возможна любая точность***)

Конечное множество рациональных чисел можно заменить целыми, введя множитель – НОК всех дробных (рациональных) значений и изменив масштаб.

ТА – это шестерка (Q, q_0, Σ, X, inv, E), где:

Q- конечное множество ***локаций***, включающее начальную локацию q_0,

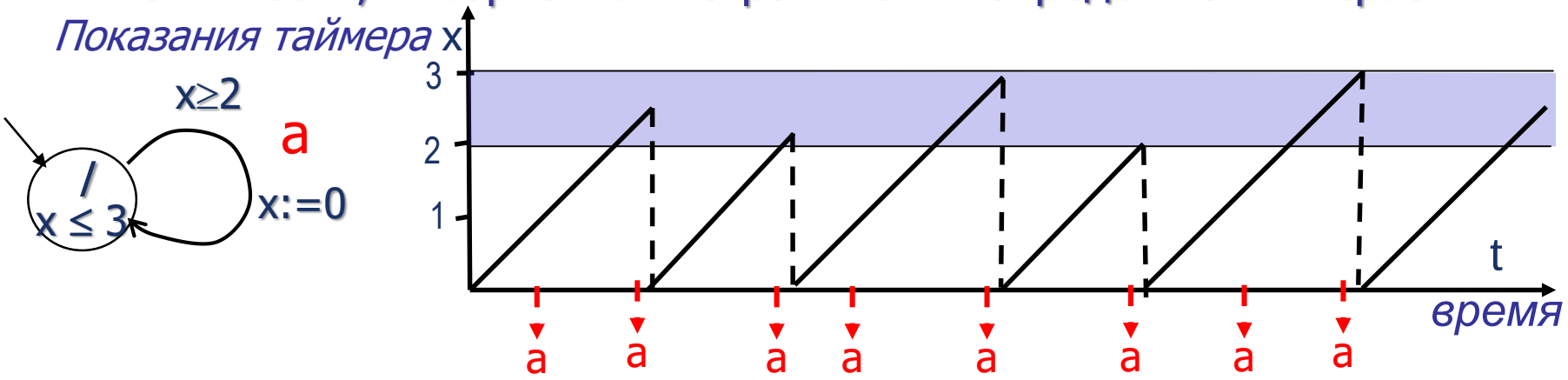
Σ - множество пометок;

X – конечное множество таймеров (clocks);

inv: $Q \to \Psi(X)$ – с каждой локаций связано ограничение (инвариант, clock constraint, ограничение на значения таймеров в локации);

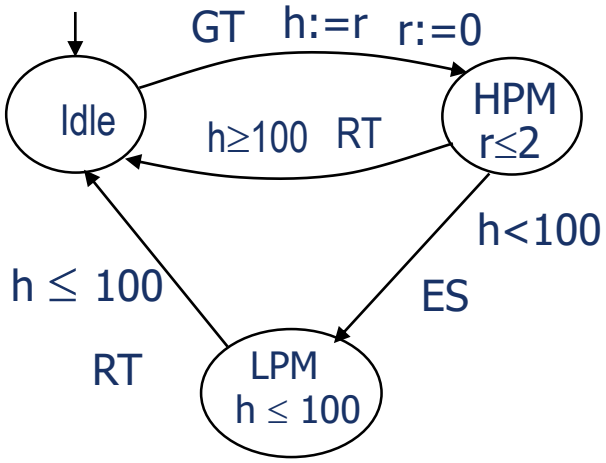
$E \subseteq Q \times \Sigma \times 2^X \times \Phi(X) \times Q$ – переходы ***Φ(x) – защита перехода***

Фактически, инварианты и ограничения определяют интервал



Пример: Реальные спецификации строятся как ТА. Спецификация FDDI протокола (передатчик)

Из руководства по протоколу FDDI



GT – *get token*

RT – *return token*

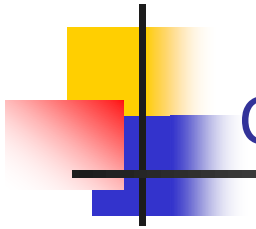
ES – *только низкоприоритетные*

HPM – *high priority msg*

LPM – *low priority msg*

FDDI (англ. **Fiber Distributed Data Interface** — Волоконно-оптический интерфейс передачи данных) — стандарт протокола передачи данных в локальной сети

- В режиме **Idle** станция ожидает токен. **GT** – это действие get token – прибытие токена
- Таймер r считает время, прошедшее после последнего прибытия токена
- По приходе токена, таймеру h присваивается значение таймера r, таймер r сбрасывается; переходим в режим **HPM**
- В режиме **HPM** станция посылает высокоприоритетные сообщения. Это может длиться не дольше 2 е.в.
- Если прошло более 100 е.в. с момента принятия токена, станция возвращается в режим **Idle**, возвращается токен. В противном случае она переходит в режим **LPM** пересылки низкоприоритетных сообщений
- В режиме **LPM** станция посылает низкоприоритетные сообщения до тех пор, пока они не кончатся, но не дольше, чем 100 е.в., после чего возвращается в режим **Idle**, возвращается токен.



Семантика временного автомата

Введем понятие "интерпретации часов v": $X \to R^+$ - присваивание каждым часам неотрицательного значения; v+d означает увеличение всех часов x с v(x) до v(x)+d

Семантической моделью временного автомата A является бесконечный временной граф переходов $S(A)=(\Sigma, Q, Q_0, R)$, где:

Σ- множество действий,

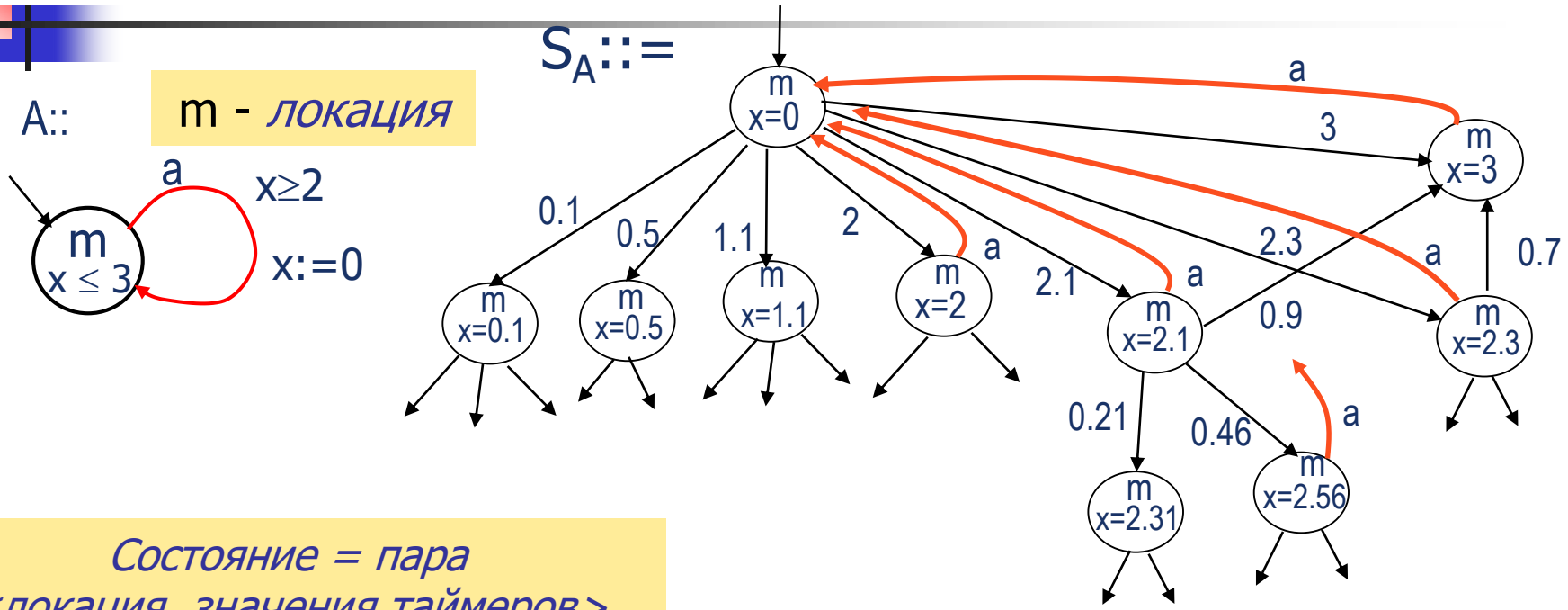
Q – множество состояний, каждое состояние – пара (l,v), где l – локация, а v: $X \to R^+$ "интерпретации" часов ; Таких состояний бесконечное количество (континуум!)

R – множество переходов двух видов:

Задержка: $(l,v) \longrightarrow (l,v+d)$, $d \in R_{\geq 0}$; при условии выполнения инвариантов;

Действие: $(l,v) \longrightarrow (l',v')$, v': показания сбрасываемых часов 0, остальных - то же

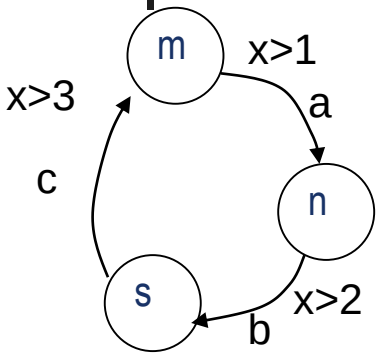
Временн’ой граф переходов – семантическая модель



Состояние = пара
<локация, значения таймеров>

Семантическая модель временн’ого автомата – временн’ой граф переходов – имеет **континуум** состояний и переходов.

Исключение нереалистичных переходов временного автомата

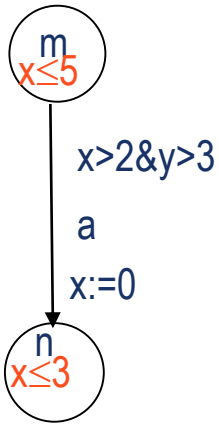


1 проблема: парадокс Зенона – бесконечное число действий в конечный промежуток времени

$$(m, x=3.4) \xrightarrow{0.1} (m, x=3.5) \xrightarrow{a} (n, x=3.5) \xrightarrow{0.01} (n, x=3.51) \xrightarrow{b} (s, x=3.51)$$

$$\xrightarrow{0.001} (s, x=3.511) \xrightarrow{c} (m, x=3.511) \xrightarrow{0.0001} (m, x=3.5111) \xrightarrow{a} \dots$$

Решение: Non-Zeno автоматы -- только конечное число действий м.б. выполнено автоматом за конечный промежуток времени



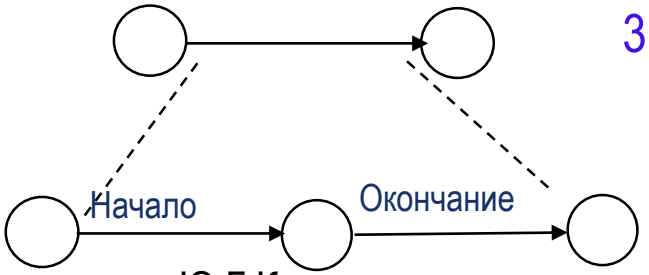
2 проблема: бесконечное нахождение в одной локации

$$(m, x=2.4, y=3.25) \xrightarrow{1.2} (m, x=3.6, y=4.45) \xrightarrow{2} (m, x=5.6, y=4.65)$$

$$\xrightarrow{3} (m, x=8.6, y=7.65) \xrightarrow{10} (m, x=18.6, y=17.65) \xrightarrow{100} \dots$$

Решение: Timed Safety Automaton = TA + Инварианты

Инварианты гарантируют прогресс (если нужно!)



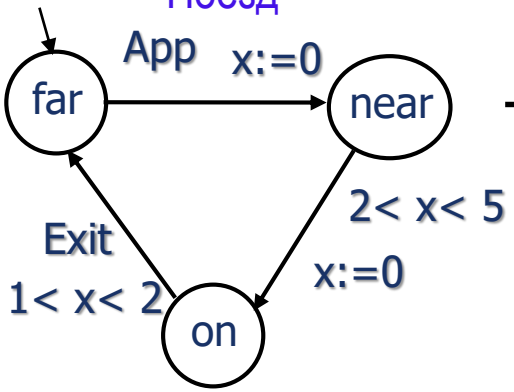
3 проблема:

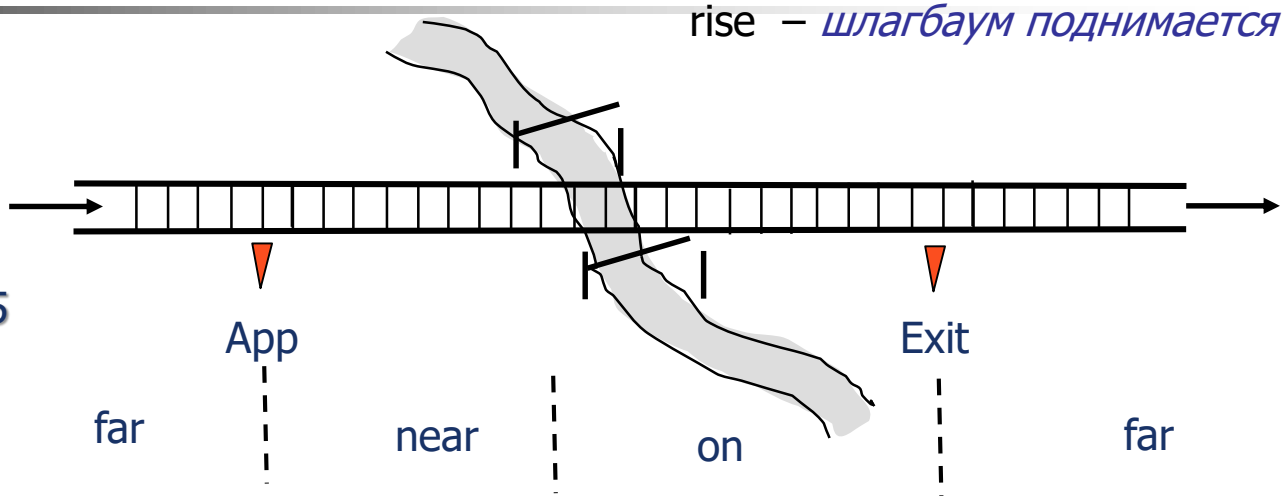
Что делать, если само действие не мгновенно? Решение. Вводить мгновенные события начала и окончания действия и промежуточную локацию его выполнения

Сети временн'ых автоматов

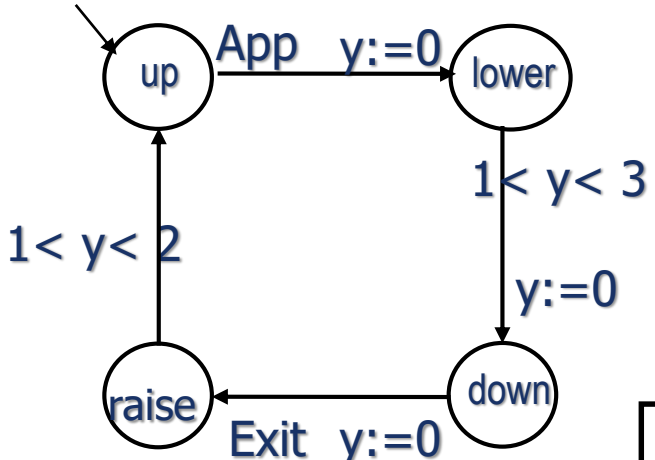
up – *шлагбаум открыт*
down – *шлагбаум закрыт*
lower – *шлагбаум опускается*
rise – *шлагбаум поднимается*

Поезд

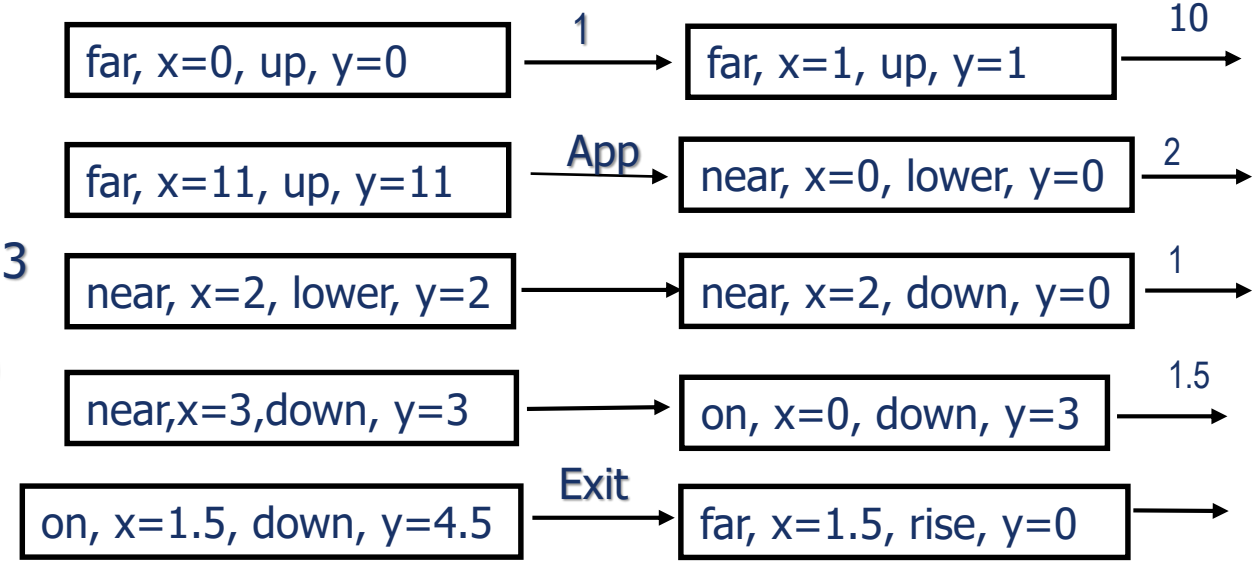


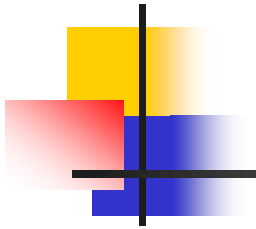


Переезд



Возможные переходы глобальных состояний





Формальной моделью сети временн’ых автоматов является параллельная композиция этих автоматов, которая также является временн’ым автоматом (формальным объектом той же природы).

Параллельная композиция Timed Automata-формально. Параллельная композиция – конструкция того же типа!

L- множество локаций

Σ- множество пометок

X – множество часов

E – множество переходов

$A_1 = (L_1, l_{01}, \Sigma_1, X_1, inv_1, E_1)$,

$A_2 = (L_2, l_{02}, \Sigma_2, X_2, inv_2, E_2)$ $X_1 \cap X_2 = \varnothing$ - таймеры локальные

$A_1 \,||\, A_2 = (L_1 \times L_2, (l_{01}, l_{02}), \Sigma_1 \cup \Sigma_2, X_1 \cup X_2, Inv, E)$,

где: $Inv(s_1,s_2) = inv_1(s_1) \wedge inv_2(s_2)$, а множество переходов

$E \subseteq (L_1 \times L_2) \times \Sigma_1 \cup \Sigma_2 \times 2^X \times \Phi(X) \times (L_1 \times L_2)$ определяется правилами:

1. Если $a \in \Sigma_1 \cap \Sigma_2$, если $(l_1, a, \lambda_1, \varphi_1, l_1') \in E_1$ и $(l_2, a, \lambda_2, \varphi_2, l_2') \in E_2$ то E будет включать переход $((l_1, l_2), a, \lambda_1 \cup \lambda_2, \varphi_1 \wedge \varphi_2, (l_1', l_2'))$ (*a – общее действие*)

2. Если $a \in \Sigma_1 - \Sigma_2$, и $(l_1, a, \lambda, \varphi, l_1') \in E_1$, то для каждого $l_2 \in L_2$ E будет включать переход $((l_1, l_2), a, \lambda, \varphi, (l_1', l_2))$ (локальные действия A1)

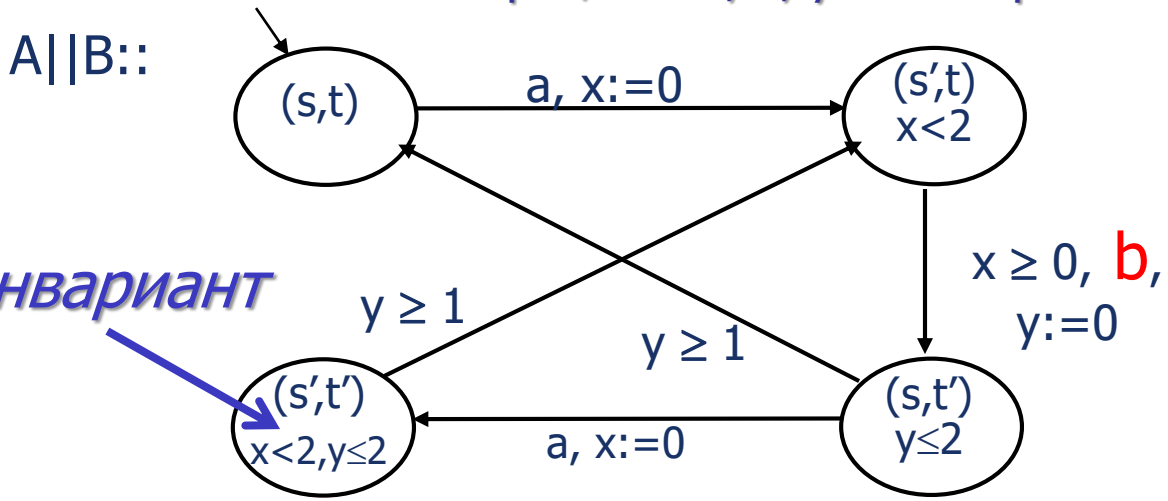
3. Если $a \in \Sigma_2 - \Sigma_1$, и $(l_2, a, \lambda, \varphi, l_2') \in E_2$, то для каждого l_1 E будет включать переход $((l_1, l_2), a, \lambda, \varphi, (l_1, l_2'))$ (локальные действия A2)

Общие действия оба автомата выполняют одновременно (взаимодействие рандеву), и каждый из автоматов выполняет свои локальные действия независимо (интерливинг). || композиция временн'ых автоматов –временн'ой автомат.

Пример параллельной композиции ТА

Общее действие b выполняют оба автомата одновременно;
a – локальное событие процесса A; x, y - таймеры

защита перехода



s, s', t, t' – это локации, не состояния

Интерливинг, чередование: *Общее действие b оба автомата выполняют одновременно (взаимодействие рандеву), а свои локальные действия каждый автомат выполняет независимо: автомат A - действие a в локации s и автомат B – внутренний переход из локации t'*

Верификация временн'ых автоматов

Существует несколько задач и методов верификации временных автоматов:

- использование контрольных автоматов (watchdogs);

- *проверка выполнимости обычных формул темпоральной логики Model checking (в том числе, включающие условия на внутренние таймеры);*

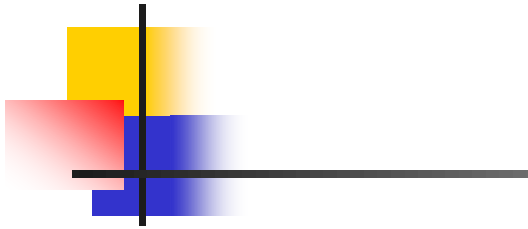
- *анализ достижимости*;

- достижимость за ограниченное время (темпоральное свойство $AF_{<5}\, p$),

- проверка «*бисимуляционной эквивалентности*» заданного автомата и автомата, выражающего требуемые свойства.

Мы рассмотрим, как строить семантическую модель ТА, на которой можно проверить выполнимость формул логики CTL, в которой в качестве атомов используются локации и целые значения локальных таймеров.

$AF\ (@l_2\ \&\ 1<x<2)$ $E\ (\ (1<x)\ U\ (@l_2\ \&\ 1<=x<2)\ )$



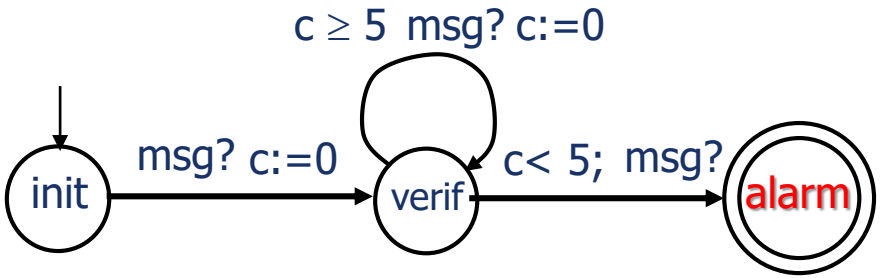
Как проверить выполнение требований к поведению сети временн’ых автоматов?

Примеры проверки свойств ТА с помощью контрольных автоматов (watchdogs)

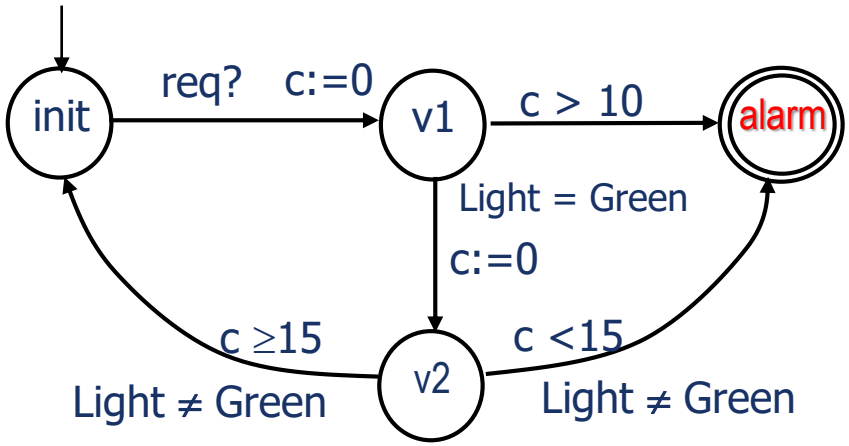
Модель устройства управления || Контрольный автомат

Watchdog перехватывает все события, по которым происходят переходы в УУ

Требование: интервал между двумя сообщениями ≥ 5

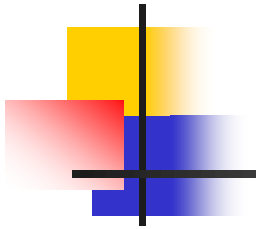


Alarm, если интервал между последовательными сообщениями < 5



Более сложное требование

Если запрос req поступил в систему, то светофор загорится зеленым не позже, чем через 10 единиц времени, и будет гореть зеленым по меньшей мере 15 единиц времени



Темпоральные требования к сети временн’ых автоматов будем выражать формулами логики CTL

Что можно считать атомарными утверждениями в требованиях к поведению временн’ых автоматов?

Неформальное определение семантики формул логики ветвящегося времени CTL

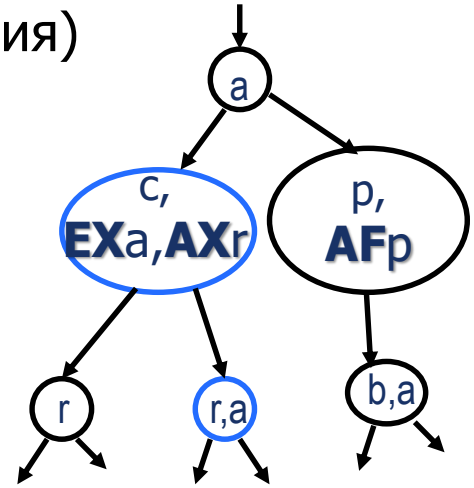
Синтаксис (грамматика):

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

Все формулы CTL – это формулы состояний

AXφ – формула φ выполняется во всех *следующих* состояниях (*непосредственных* потомках данного текущего состояния)

EXφ - формула φ выполняется хотя бы в одном *непосредственном следующем* состоянии

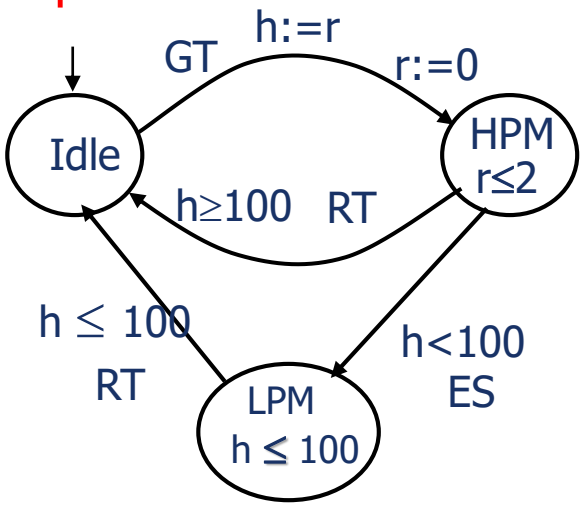


AFφ (**неизбежно** φ) - на всех путях из текущего состояния формула φ *когда-нибудь* выполнится

EFφ (*возможно* φ) - из текущего состояния *существует* путь, на котором формула φ *когда-нибудь* выполнится

Атомарные утверждения логики CTL для Timed Automata

Что является атомарным утверждением в требованиях к поведению временн'ых автоматов?



1. *Режим (локация).*
Автомат находится в режиме (локации) *loc*

@*loc*

2. Clock constraints – произвольные ограничения на значения таймеров

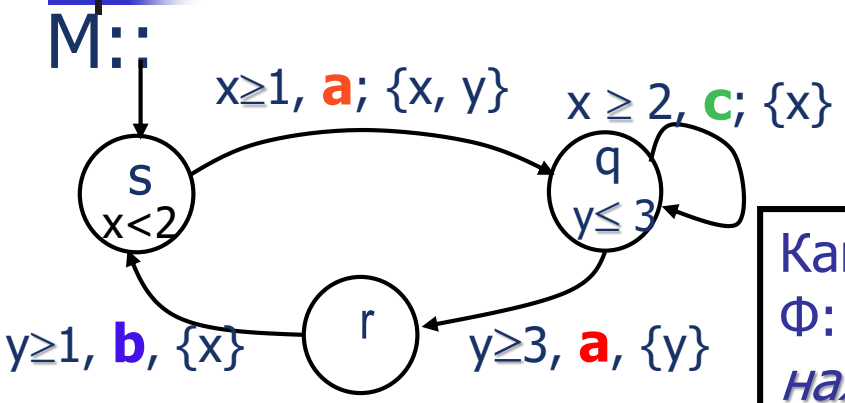
$x < 3;$ $y \geq 2;$ $z \leq 2;$

Границы ТОЛЬКО ЦЕЛЫЕ

Пример требования $EG(\ @Idle \Rightarrow AF\ (@LPM \wedge h<20 \wedge r>h)\ )$

Существует такое вычисление, в любом состоянии которого (EG) если автомат находится в локации (режиме) Idle, то на любом вычислении из этого состояния когда-нибудь в будущем (AF) мы придем в режим LPM, причем значение таймера h будет менее 20, а значение таймера r будет больше значения таймера h.

Model checking для временн'ых автоматов



{x, y} = сброс обоих таймеров на переходе

Выполняется ли в M формула темпоральной логики CTL:

$\Phi = AG(\ @s \Rightarrow AF\ (x>2 \wedge @r)\ )$

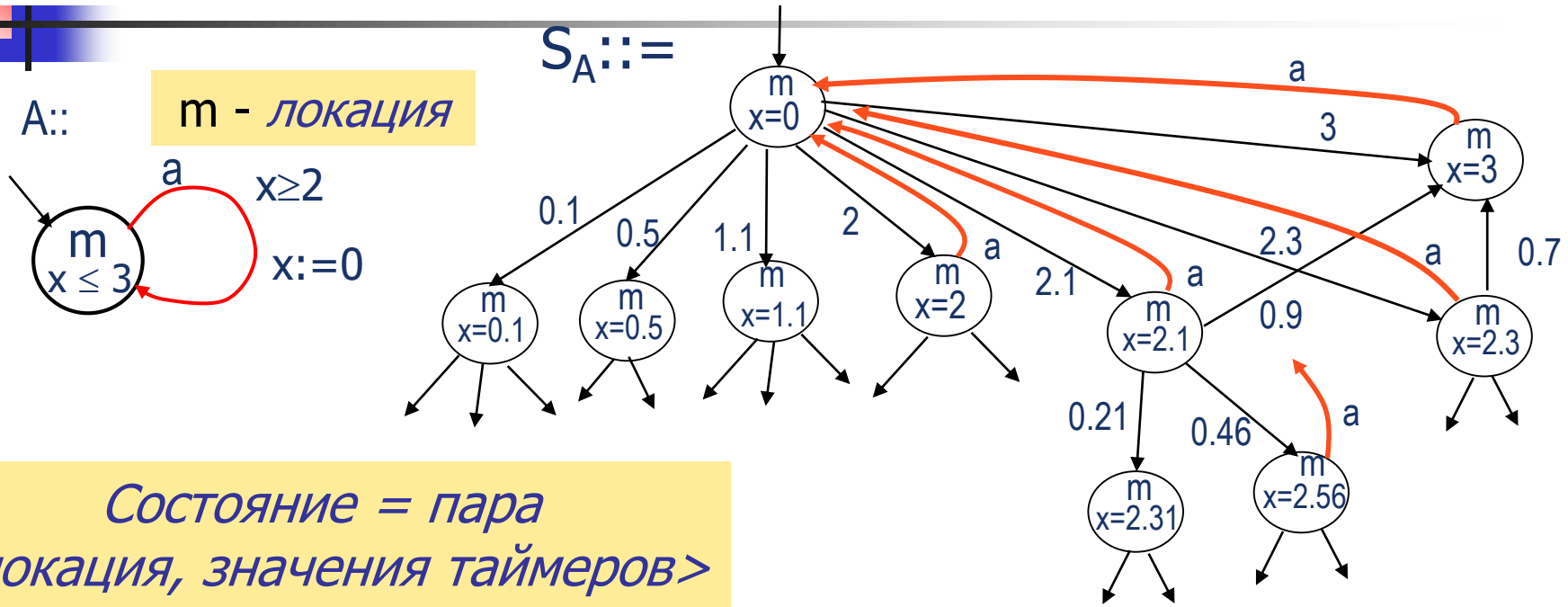
Как для этого TA проверить выполнение Ф: "*на всех вычислениях если M находится в локации s, то по любому пути когда-нибудь в будущем M придет в локацию r со значением таймера x, большим 2*"??

Стандартными методами проверить это нельзя!

Временной автомат (граф локаций) TA не показывает **реальных** траекторий поведения.

- Как проверить выполнение требования?
  Какая модель представляет реальную динамику, поведение такого автомата в РЕАЛЬНОМ ВРЕМЕНИ? Модель должна быть КОНЕЧНОЙ.

Временн'ой граф переходов – семантическая модель



Состояние = пара <локация, значения таймеров>

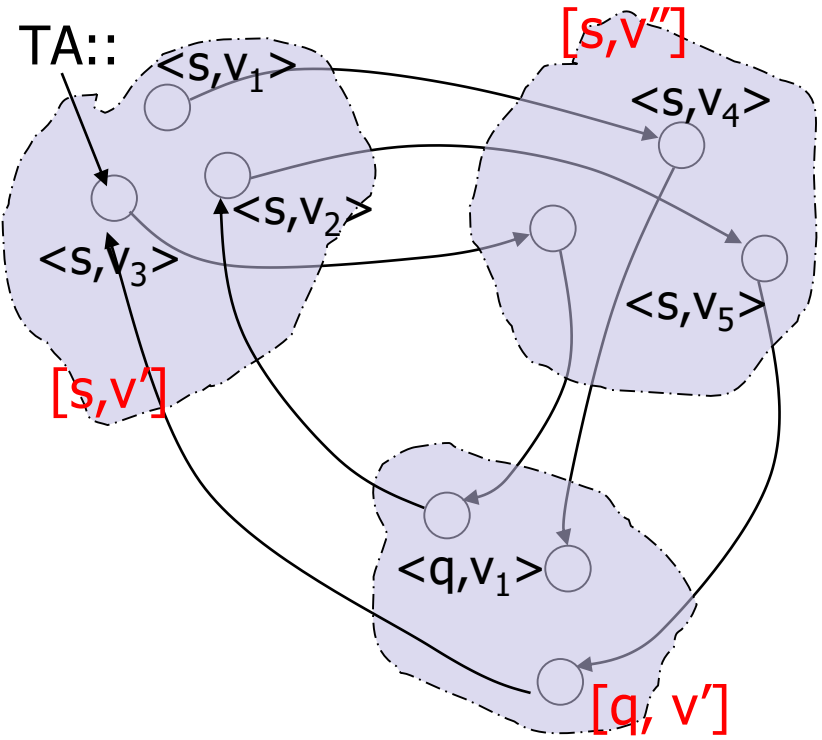
Семантическая модель временн'ого автомата – временн'ой граф переходов – имеет **континуум** состояний и переходов.

Использовать ее как модель для представления РЕАЛЬНОГО поведения невозможно!

Как построить структуру Крипке временн'ого автомата?

Строить классы эквивалентных состояний

Фактор-автомат по модулю отношения эквивалентности π (ТА$_\pi$). Один таймер



Какие состояния ТА будут эквивалентны относительно ЛЮБЫХ проверяемых свойств?

<s, x=0.30875> ≈ ? <q, x=0.30875> **НЕТ**

разные локации

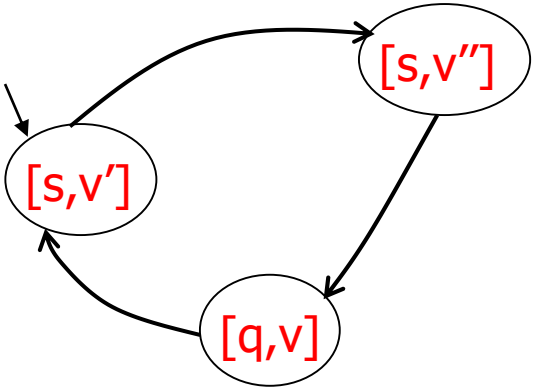
<q, x=0.99999> ≈ ? <q, x=1.0001> **НЕТ**

для первого состояния утверждение x>1 ложно, для второго - истинно

<q, x=0.30875> ≈ ? <q, x=0.9999> **ДА**

во всех защитах – только целые

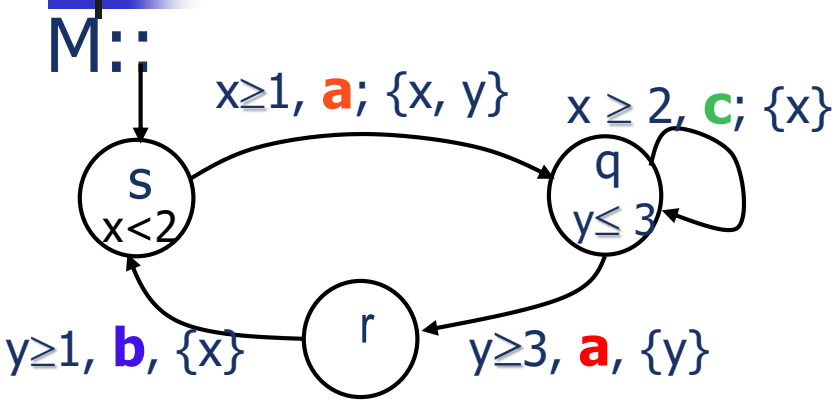
ТА$_\pi$::



q, s – значения локаций и переменных
v, v', v'' – наборы значений таймеров

локаций -2;
классов эквивалентности - 3

Как определить эквивалентность состояний ТА относительно ЛЮБЫХ формул CTL?



Какие состояния (q, v), (q,v') будут эквивалентны относительно требований к ТА, выражаемых формулами CTL?

Выполняется ли в M формула Φ темпоральной логики CTL:

$$\Phi = AG(\ @s \Rightarrow AF\ (x>1 \wedge @r)\ )$$

Когда для любой CTL формулы ϕ выполняется соотношение

$$(q, v) \models \phi \equiv (q, v') \models \phi\ ?$$

- В формулах CTL стоят имена локаций и произвольные утверждения с таймерами, имеющие ТОЛЬКО ЦЕЛОЧИСЛЕННЫЕ значения границ

Эквивалентность ≅ на множестве состояний ТА. Один таймер.

v(x) – значение (valuation) таймера x

Вводим отношение ≅ эквивалентности на множестве наборов значений таймеров.

Для двух интерпретаций таймеров v и v′, **v ≅ v′** iff для любых значений таймера x:

($\lfloor A \rfloor$ - целая часть A, fr(A) – дробная часть A, C_x – максимальное значение x)

1 $v(x)>c_x$ & $v'(x)>c_x$ *все значения таймера x, большие c_x эквивалентны между собой* ИЛИ

$v(x) \le c_x$ & $v'(x) \le c_x$ & $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ =>

2 $fr(v(x)) \neq 0$ & $fr(v'(x)) \neq 0$ *все значения таймера x с одинаковой целой частью и непустой дробной частью эквивалентны*

3 $fr(v(x)) = 0$ & $fr(v'(x)) = 0$ *целые значения таймеров составляют отдельные классы эквивалентности*

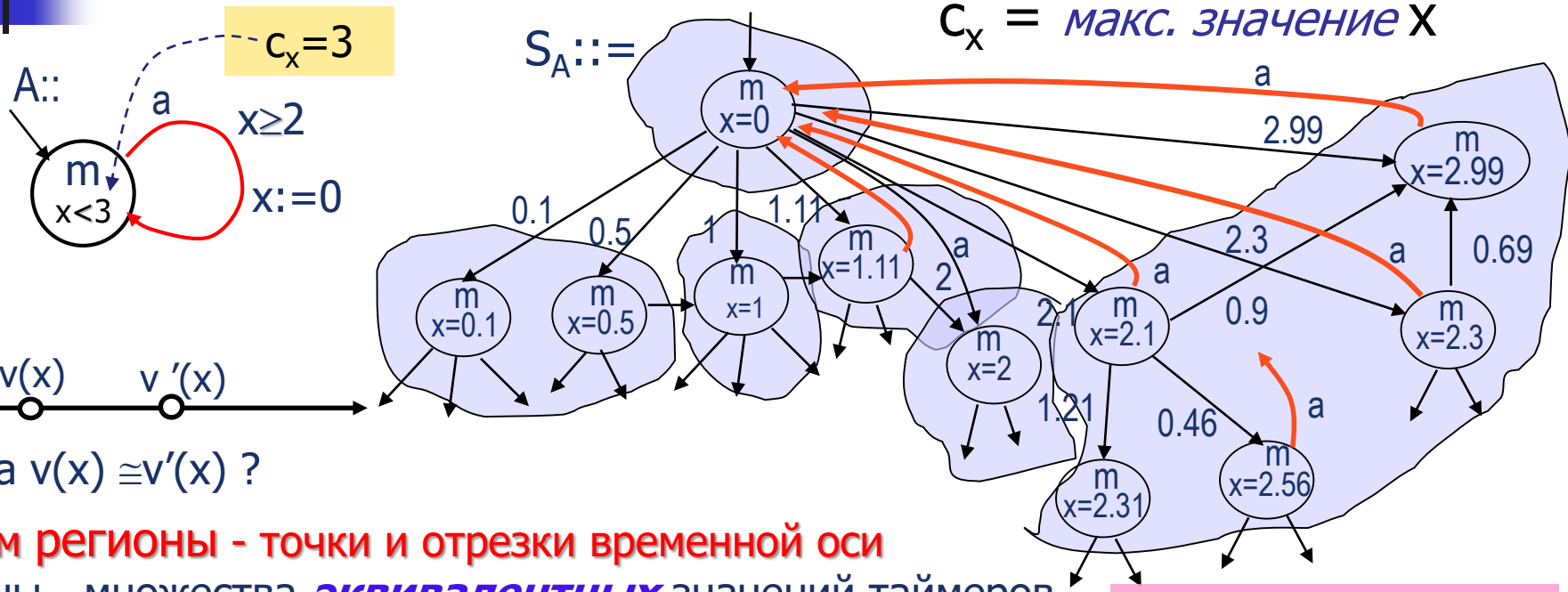
$[v]_{\cong}$ класс эквивалентности, которому принадлежит значение v

Пример: пусть v(x)=0.3; Тогда $[v]_{\cong}$ = 0<x<1;

v(x=1.2) ≅ v(x=1.3) ≅ v(x=1.77) = 1<x<2

Временн'ой регион временн'ого автомата A – класс эквивалентности наборов значений таймеров, индуцированный отношением ≅

Один таймер: конечное представление поведения TA с помощью графа регионов

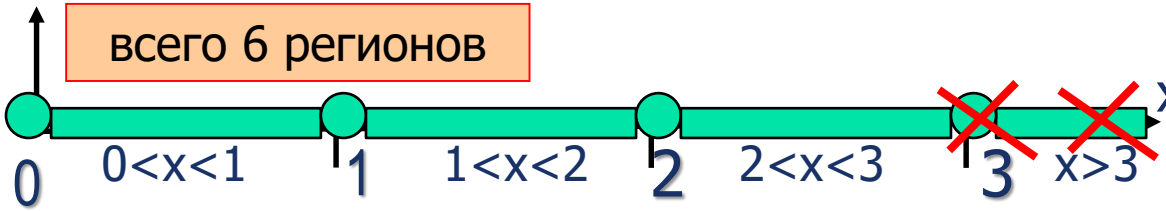


Когда v(x) ≅v′(x) ?

Введем регионы - точки и отрезки временной оси

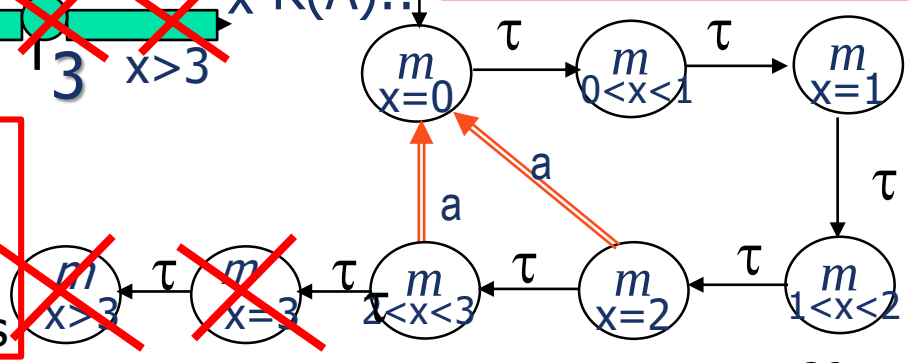
Регионы - множества ***эквивалентных*** значений таймеров

всего 6 регионов



R(A) отражает реальное поведение

x R(A)::

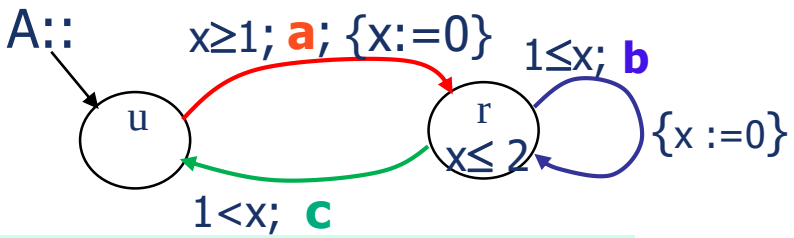


Любые две точки одного и того же временного региона эквивалентны относительно любого свойства, выраженного с помощью clock constraints

Граф регионов – один таймер

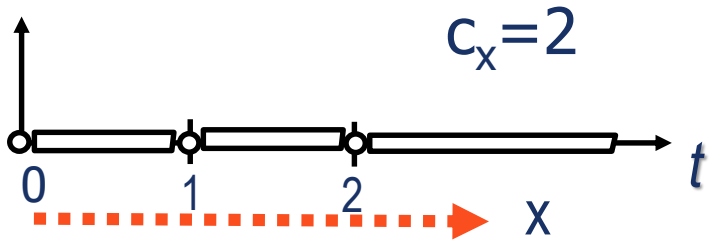
⌊z⌋ - целая часть z

Пример:



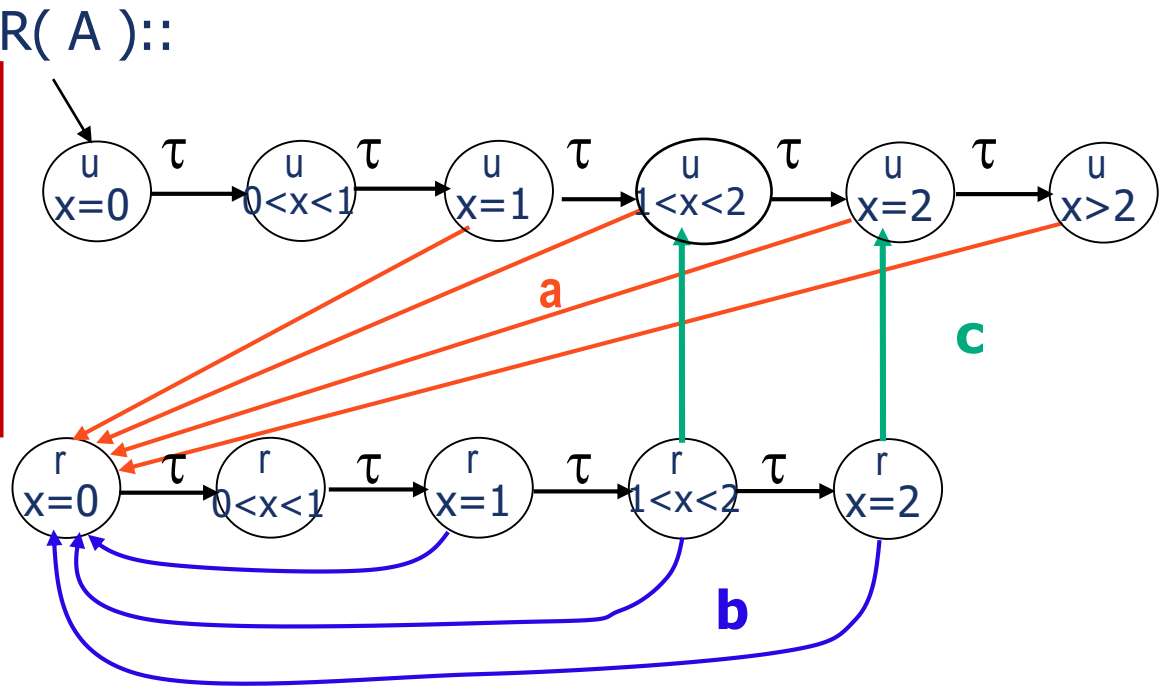
У автомата A 6 регионов

Регионы для одного таймера

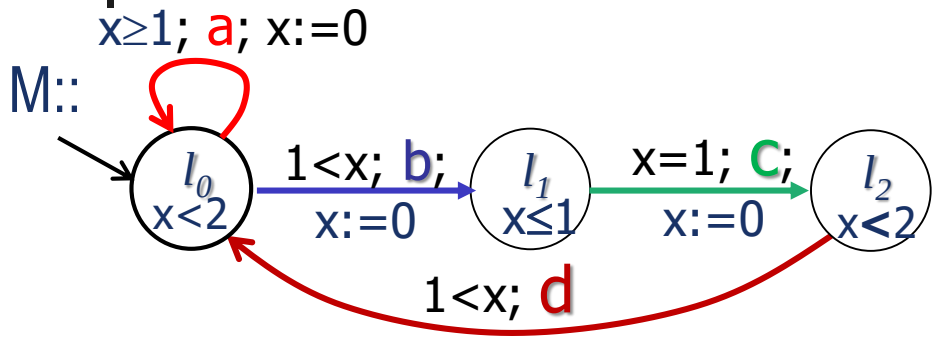


Возрастание значений таймера x

Для анализа любых темпоральных свойств поведения автомата A, выраженных формулами CTL, достаточно анализировать региональный автомат R(A)



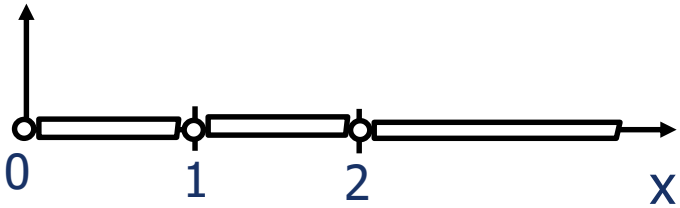
Пример. Временной автомат с одним таймером и соответствующий ему РЕГИОНАЛЬНЫЙ АВТОМАТ



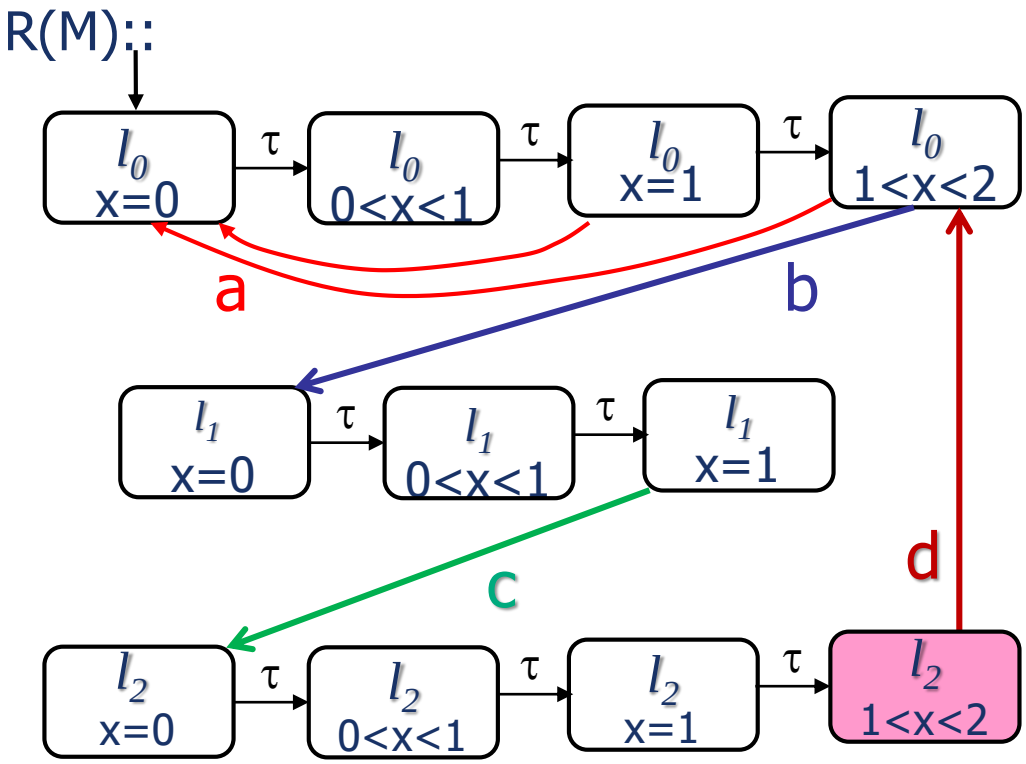
Выполняется ли в M формула CTL:

EF ($x>1 \wedge @l_2$)?

$c_x=2$



множество регионов



Анализ любых свойств поведения временн'ого автомата M, выраженных формулами CTL, можно выполнить на автомате R(M).

Свойства временн'ых регионов и региональных автоматов

Чем удобны временные регионы?

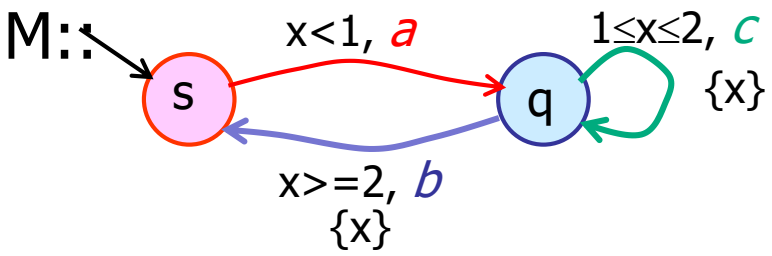
Если все временн'ые соотношения с ~ k устанавливаются для конечного числа таймеров и рациональных k, то множество регионов – классов эквивалентности интерпретаций таймеров - конечно

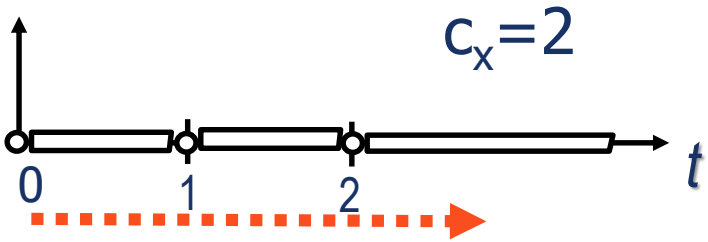
Пусть v_1 и v_2 – два вектора значений таймеров (их интерпретации).
Если $[v_1]_{\cong} = [v_2]_{\cong}$ (т.е. эти векторы – в одном и том же регионе), то для любой формулы f логики CTL $f(v_1) \equiv f(v_2)$.

Любая формула ϕ логики CTL для временн'ого автомата в одной и той же локации одновременно либо выполняется, либо не выполняется при двух интерпретациях таймеров в одном и том же регионе.

Таким образом, фиксировать состояния временн'ого автомата достаточно фиксировать ТОЛЬКО с точностью до временн'ых регионов.

Региональный автомат и временной автомат. Один таймер

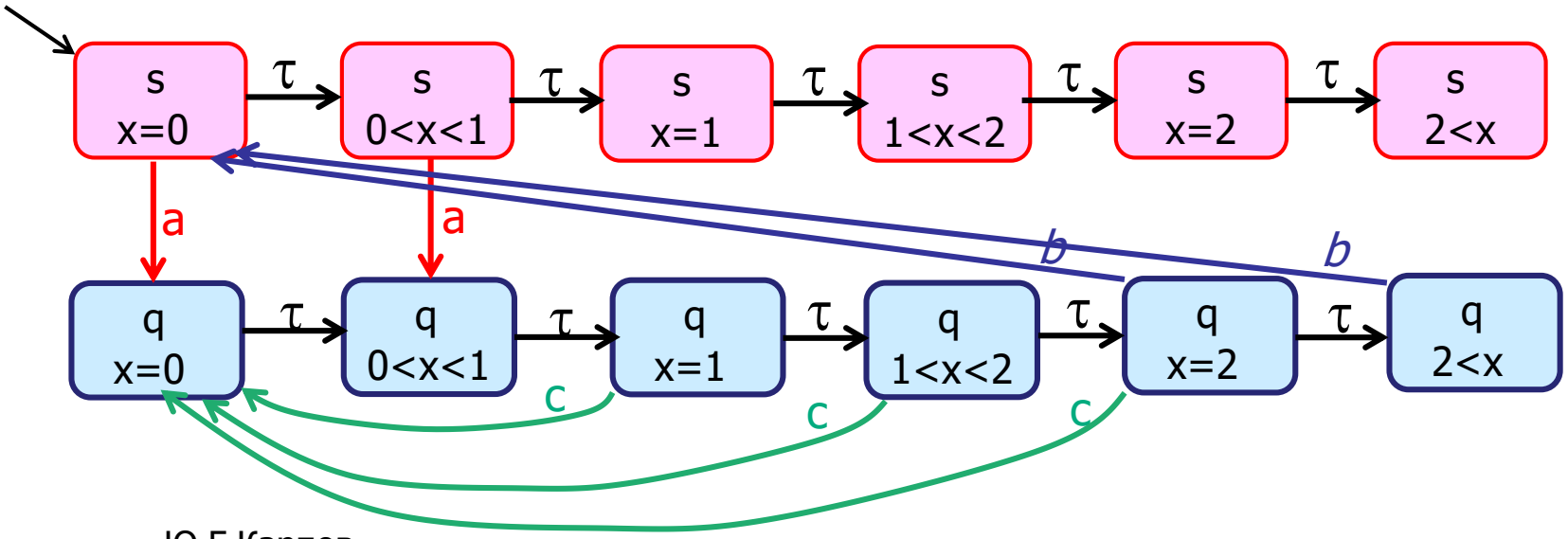




Возрастание значений таймера x

Выполняется ли в M формула CTL:

$\mathbf{E}((@s \wedge x \leq 1)\ \mathbf{U}\ (x>1 \wedge @q))$?



Эквивалентность ≅ на **множестве** наборов значений **таймеров** ТА. Число таймеров >1

Вводим отношение ≅ эквивалентности на множестве наборов значений таймеров.

Для двух интерпретаций таймеров v и v′, **v ≅ v′** iff для любых таймеров x, y:
($\lfloor A \rfloor$ - целая часть A, fr(A) – дробная часть A,
C_X – максимальное значение таймера x, которое встречается в модели ТА)

1. $v(x)>c_x$ & $v'(x)>c_x$ *(все значения таймера x, большие c_x эквивалентны)* И

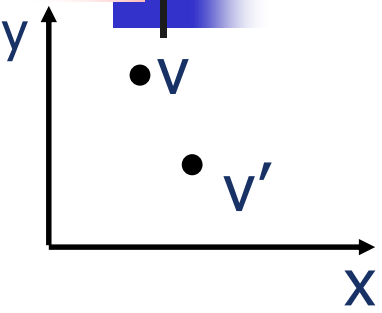
2. $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ *(все значения таймера x с одинаковой целой частью эквивалентны)* И

3. $v(x) \le c_x$ & $v(y) \le c_y$ & $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ & $\lfloor v(y) \rfloor = \lfloor v'(y) \rfloor \Rightarrow$
$(fr(v(x)) < fr(v(y))) \equiv (fr(v'(x)) < fr(v'(y)))$
(если для двух таймеров отношения между их дробными частями не меняются, а целые значения сохраняются, то эти наборы значений эквивалентны)

4. $v(x) \le c_x$ & $v(y) \le c_y$ & $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$ & $\lfloor v(y) \rfloor = \lfloor v'(y) \rfloor \Rightarrow$
$(fr(v(x)) = fr(v(y))) \equiv (fr(v'(x)) = fr(v'(y)))$
(если для двух таймеров отношения между их дробными частями не меняются, а целые значения сохраняются, то эти наборы значений эквивалентны)

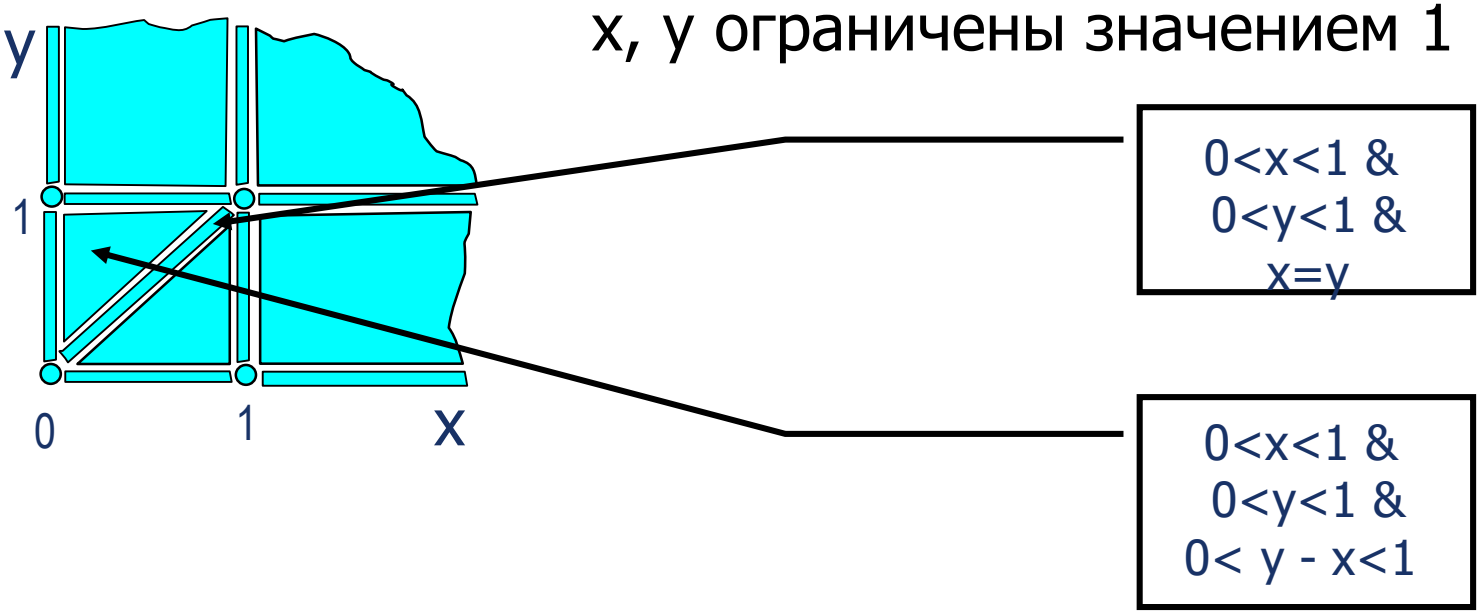
⌊z⌋ - целая часть z

Граф регионов – два таймера - формально



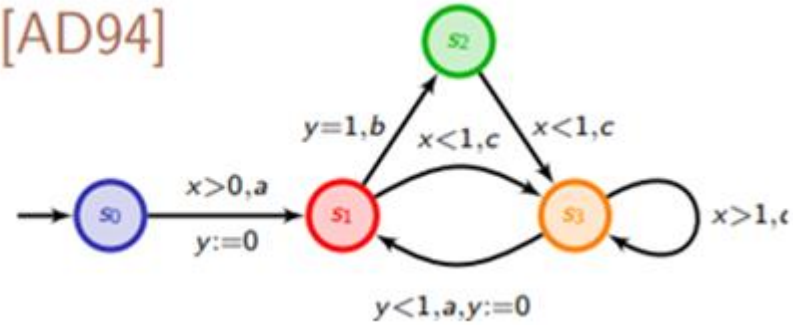
Две интерпретации часов v и v′ эквивалентны (v ≅ v′), iff:

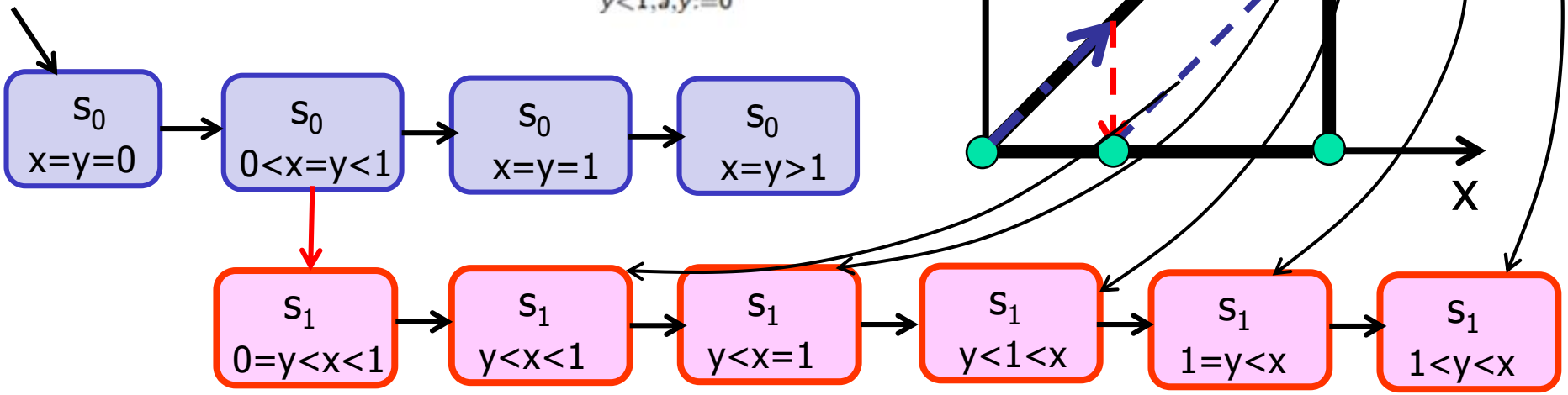
1. $(\forall x)\ v(x)>c_x \ \&\ v'(x)>c_x$
2. $(\forall x,y)\ v(x)\le c_x \& v(y)\le c_y \Rightarrow fr(v(x))\le fr(v(y)) \equiv fr(v'(x))\le fr(v'(y))$
3. $(\forall x)\ \ v(x) \le c_x \ \&\ v'(x) \le c_x \Rightarrow fr(v(x)) = 0 \equiv fr(v'(x)) = 0$

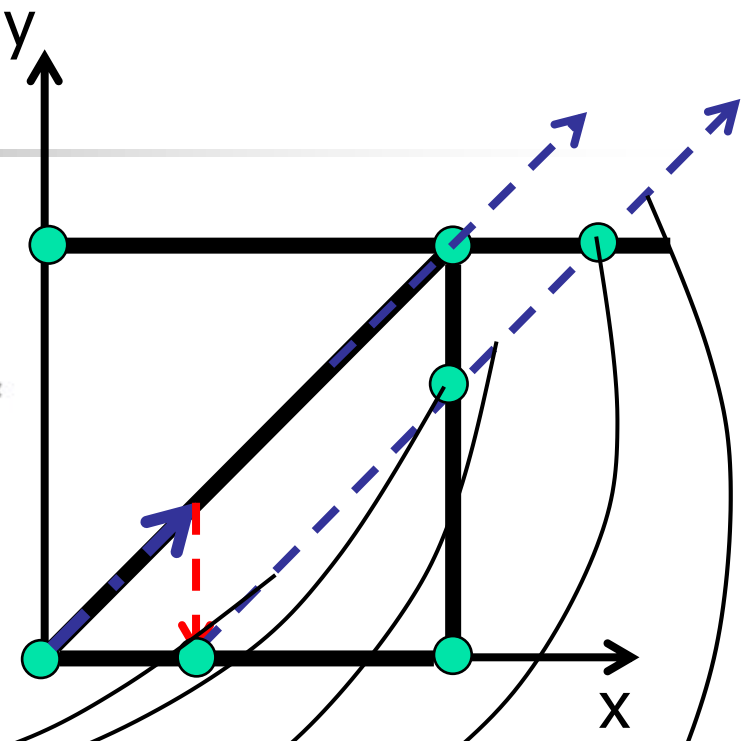


Построение вычисления

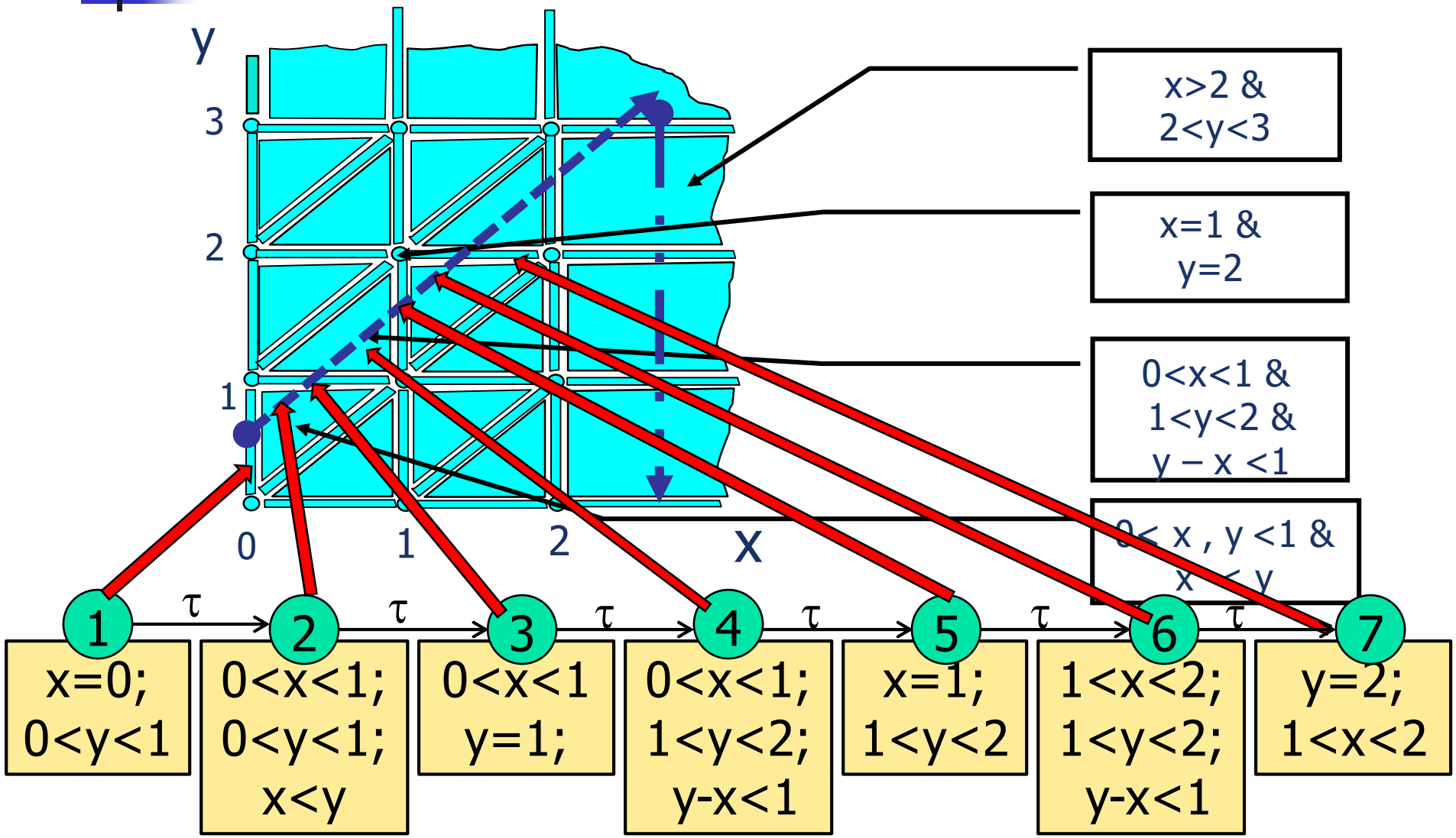
An example [AD94]



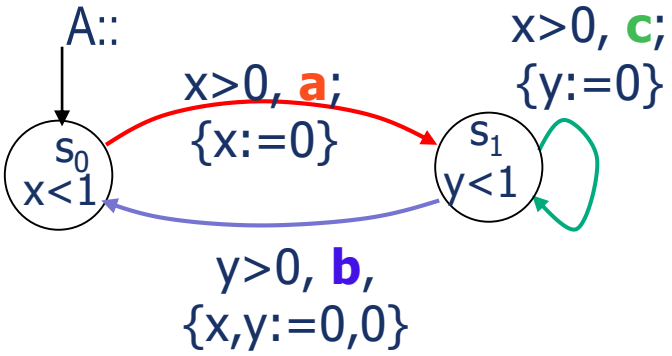


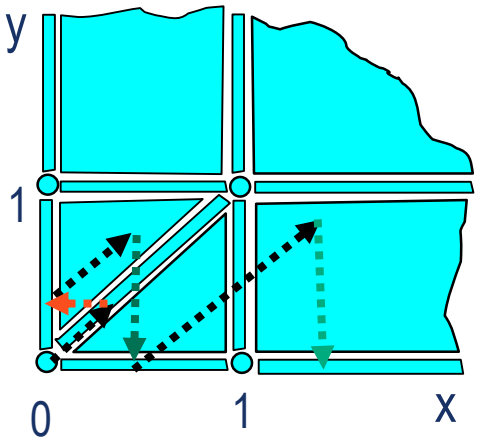


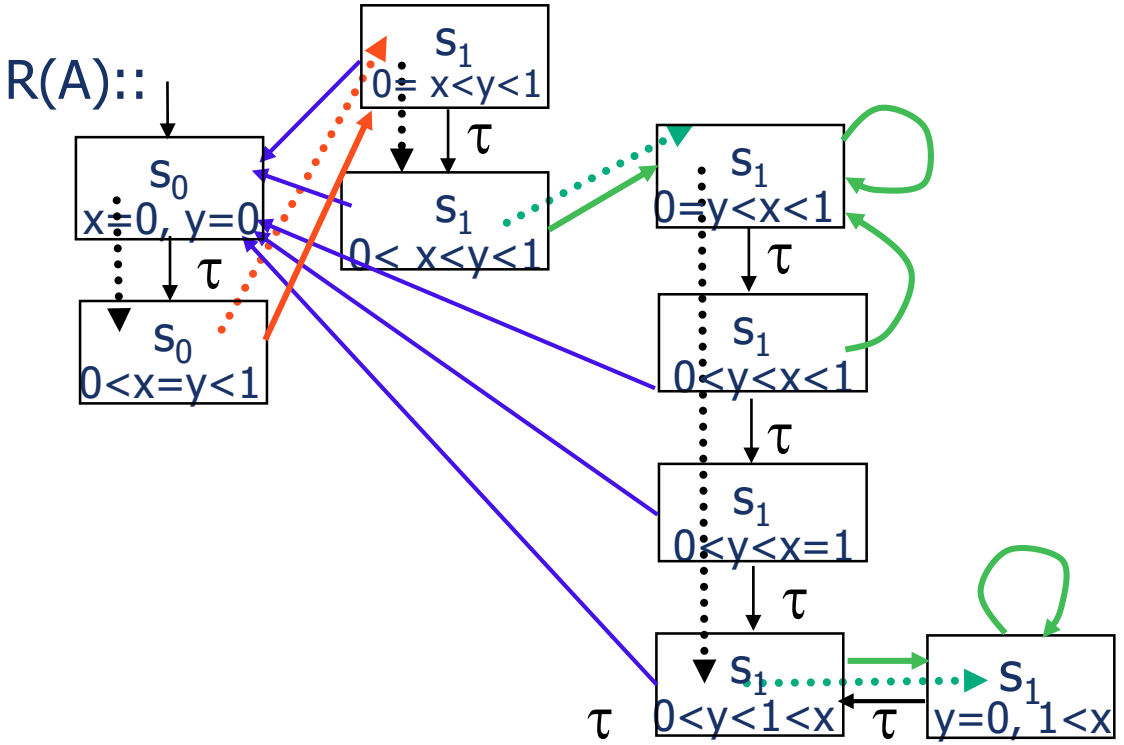
Граф регионов – два таймера; $c_x=2$, $c_y=3$



Граф регионов – два таймера.
Граф регионов может обнаружить скрытую блокировку

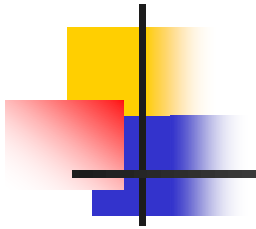






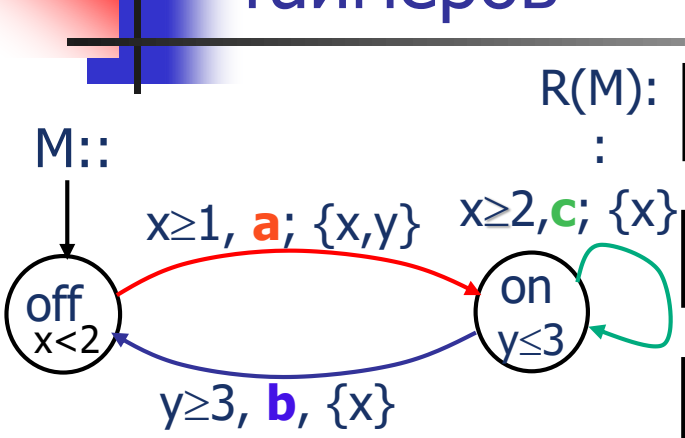
одно из вычислений

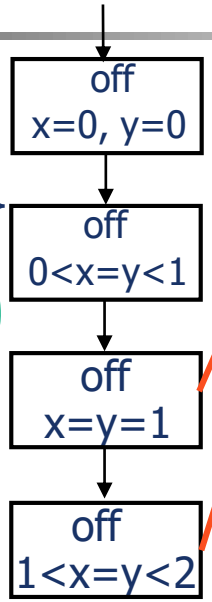
Последовательность возрастаний таймеров

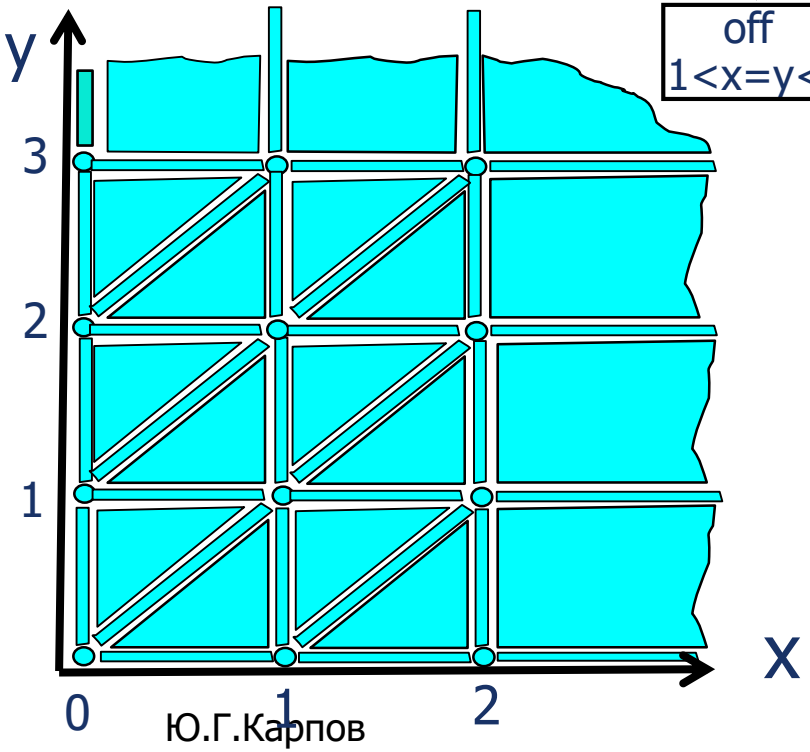


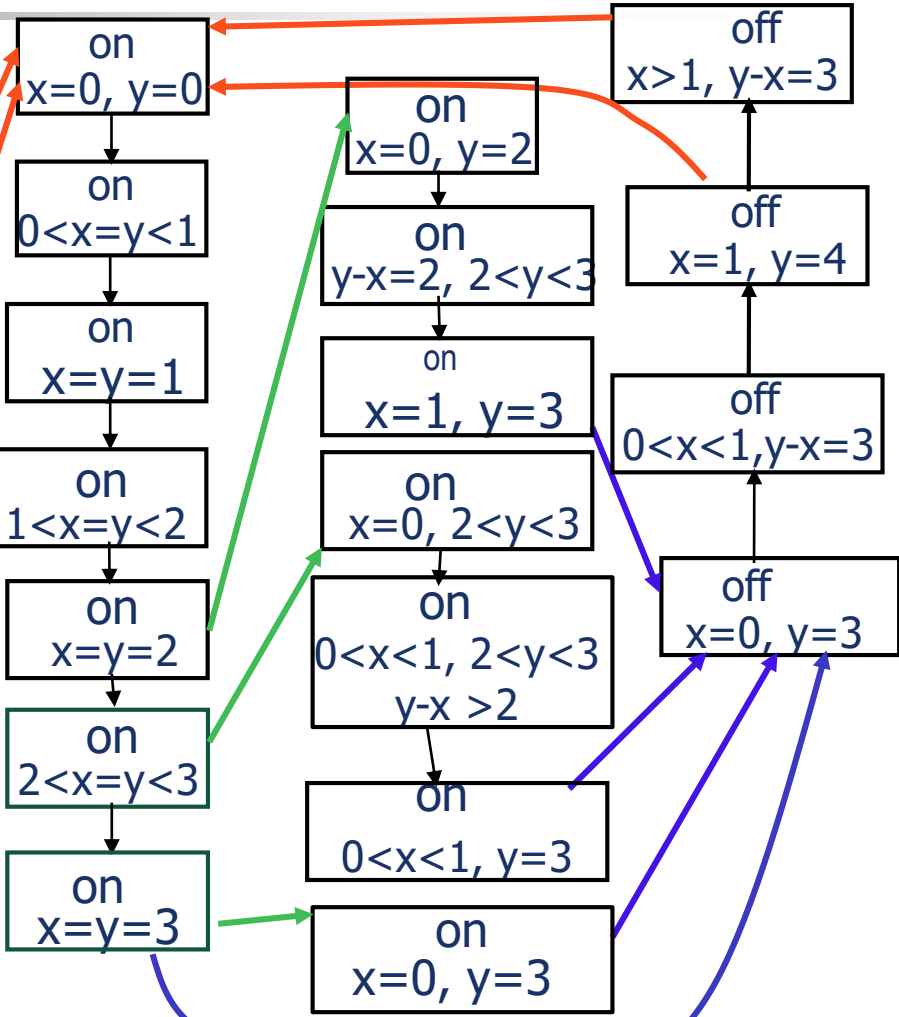
Проверка выполнения формулы CTL для временн’ого автомата A сводится к проверке этой формулы для регионального графа, построенного для автомата A

Пример. Региональный Автомат для двух таймеров





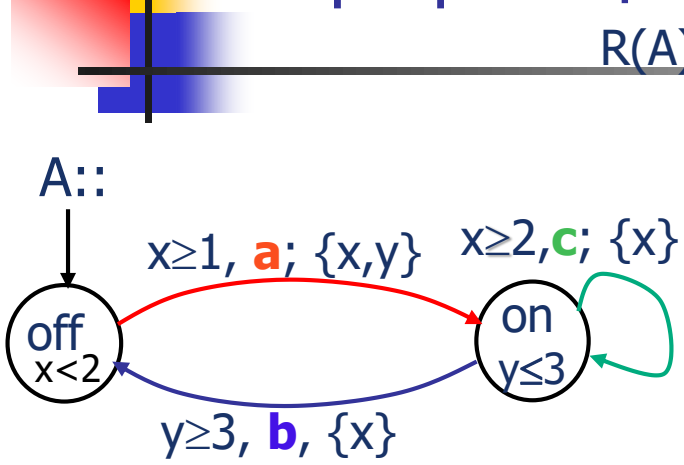


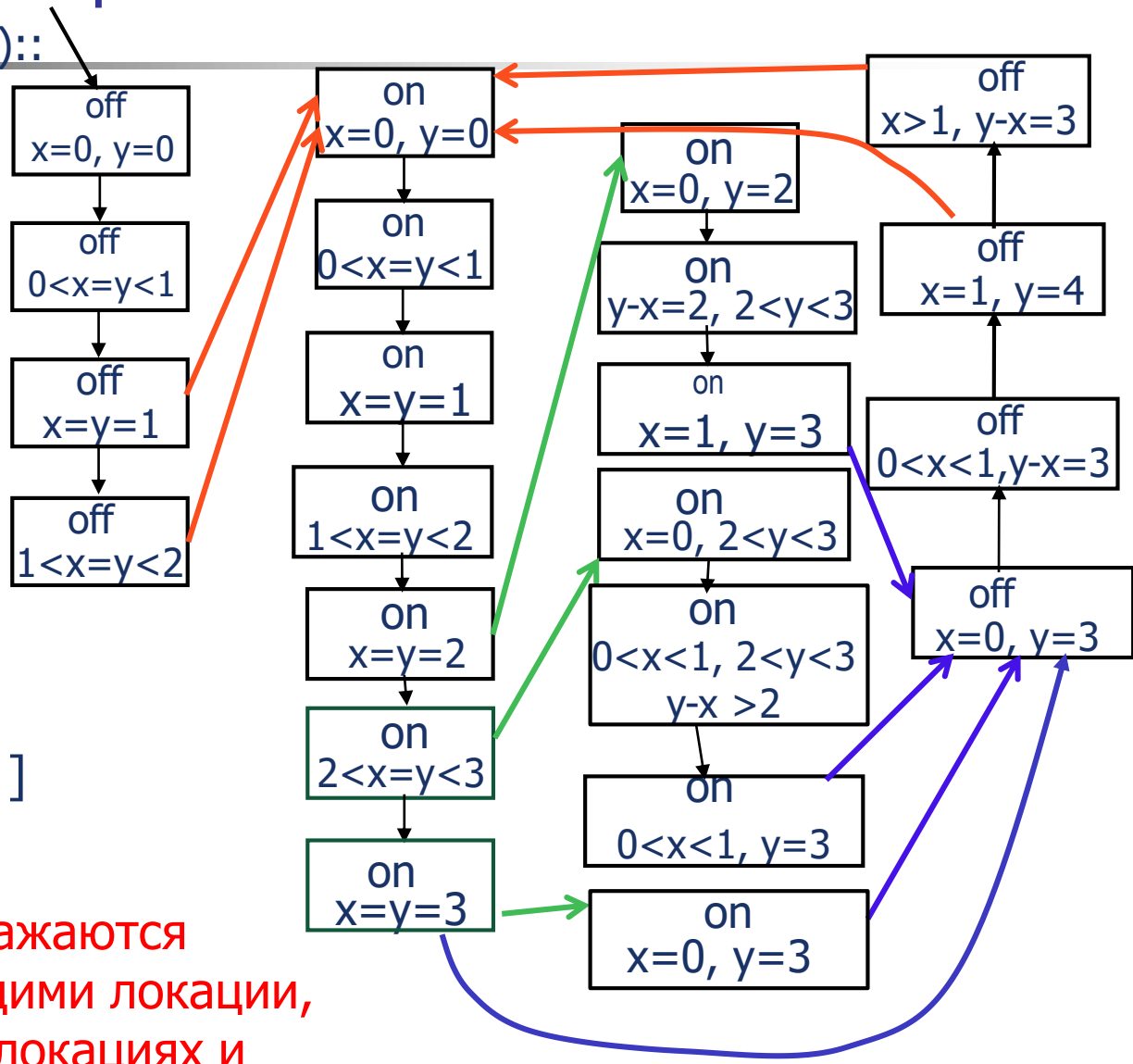


Выполняется ли в M формула CTL:

@EF (x>1∧@*on*)?

Верификация Временн'ых Автоматов: CTL





Примеры свойств:

AX EF (@off & x=y)

AX(@on → AF @off)

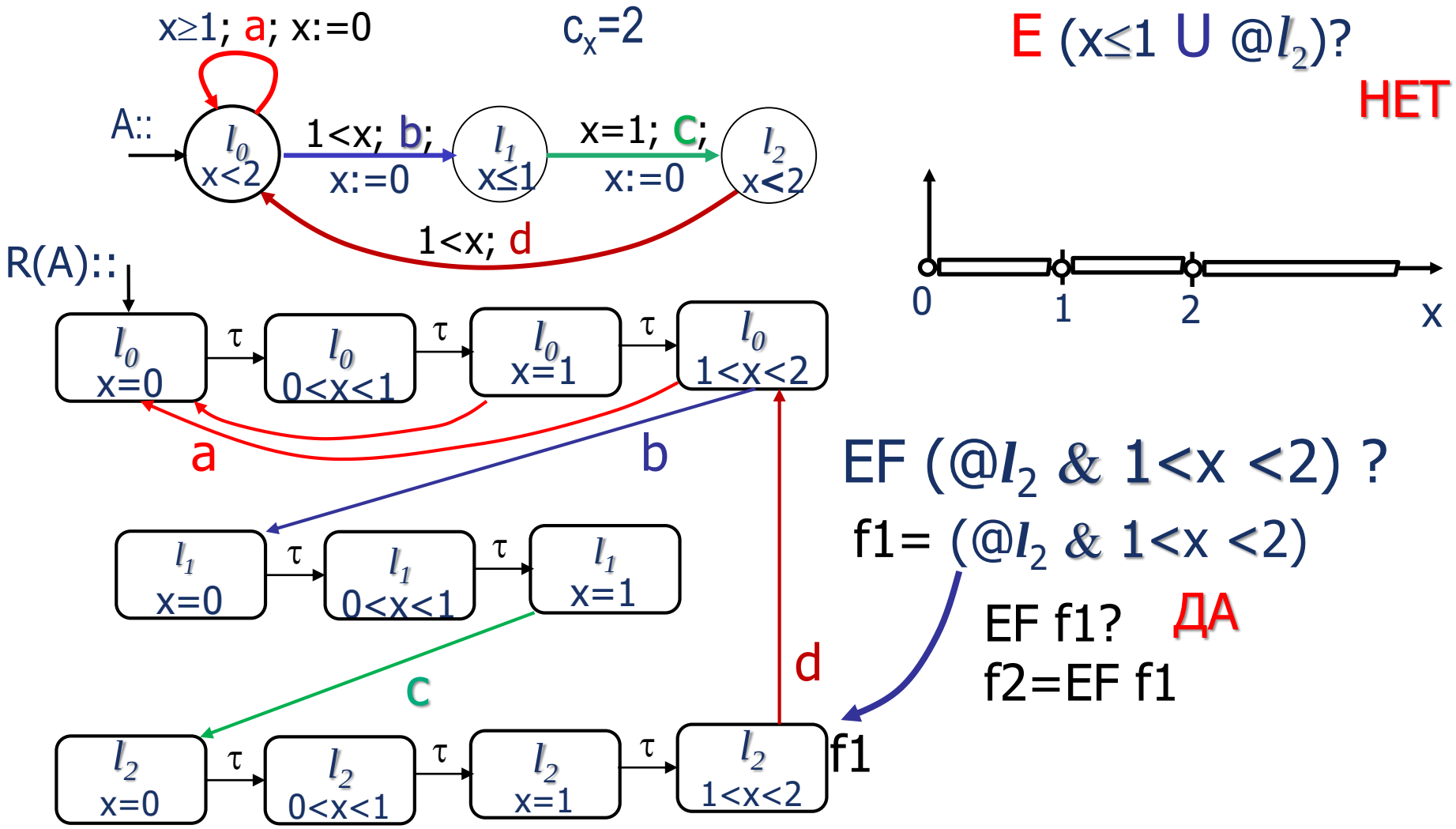
E [@off U (@on & (x ≥ 3))]

AF (x≤y)

Все свойства, которые выражаются формулами CTL, включающими локации, атомарные утверждения в локациях и значения таймеров, проверяются на R(A)

Верификация Временн'ых Автоматов. CTL-формулы

Сводится к верификации графа регионов относительно CTL–формулы



E (x≤1 U @l_2)?

НЕТ

EF (@l_2 & 1<x <2) ?

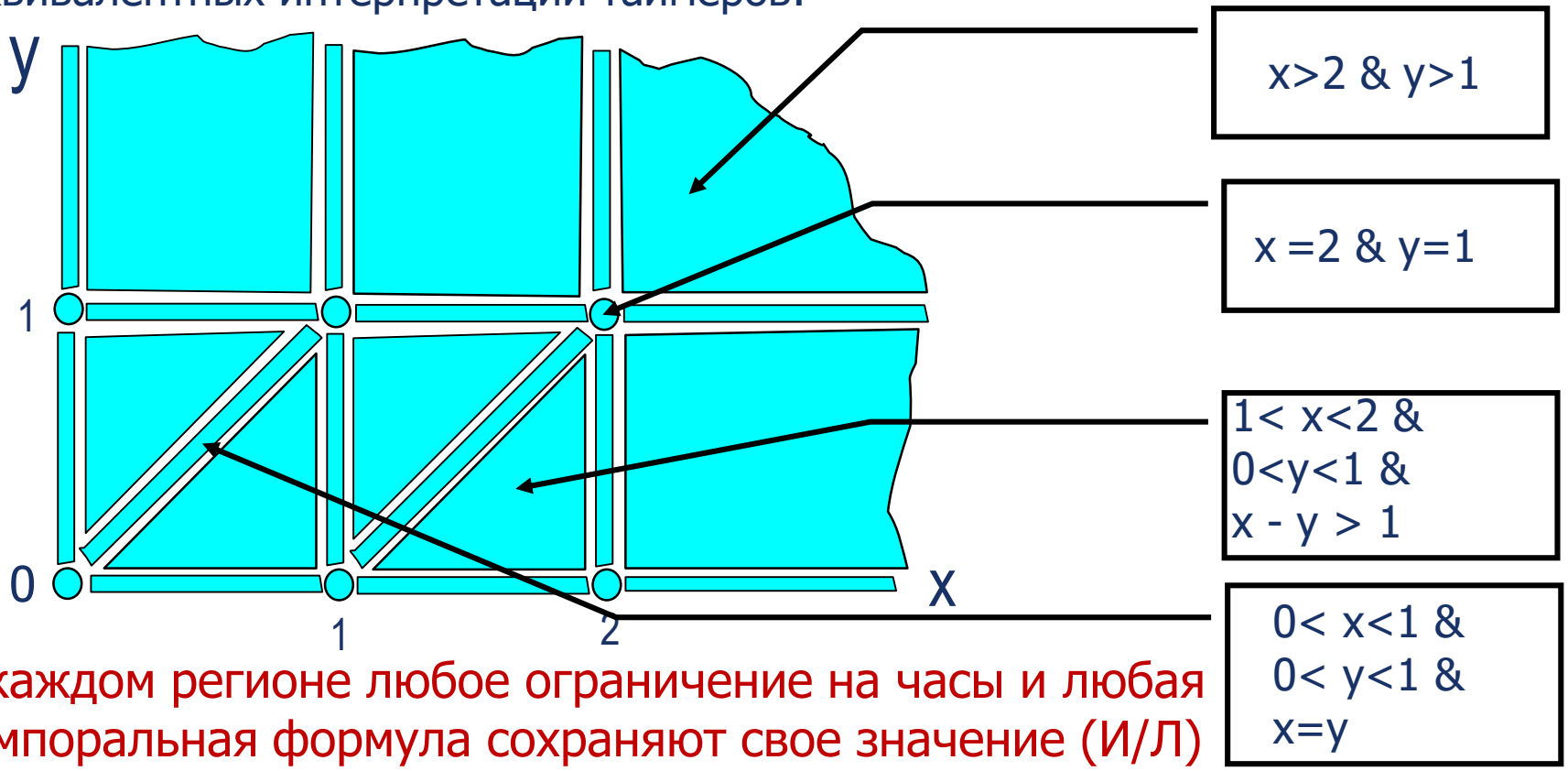
f1= (@l_2 & 1<x <2)

EF f1? ДА

f2=EF f1

Граф регионов – два таймера - размеры

Пример. Два таймера, x и y, с возможными временными ограничениями $x \sim k_1$, $y \sim k_2$, причем $k_1 \in \{0, 1, 2\}$, $k_2 \in \{0, 1\}$. **Всего 28 регионов** эквивалентных интерпретаций таймеров:

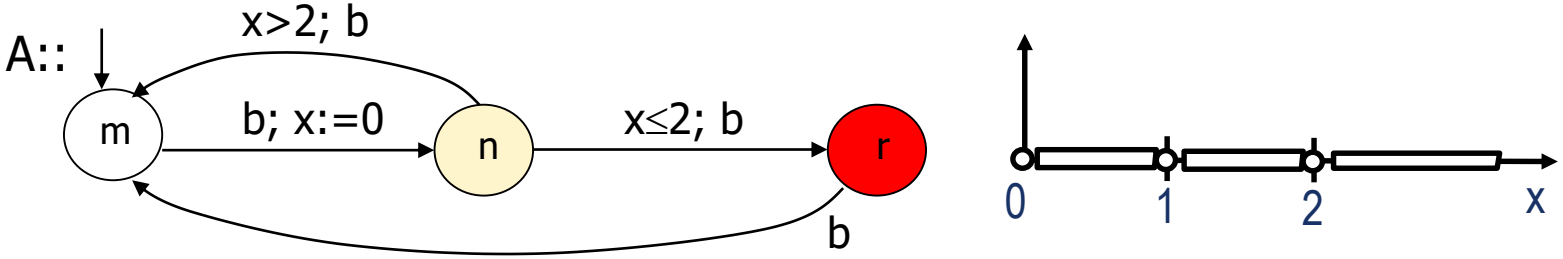


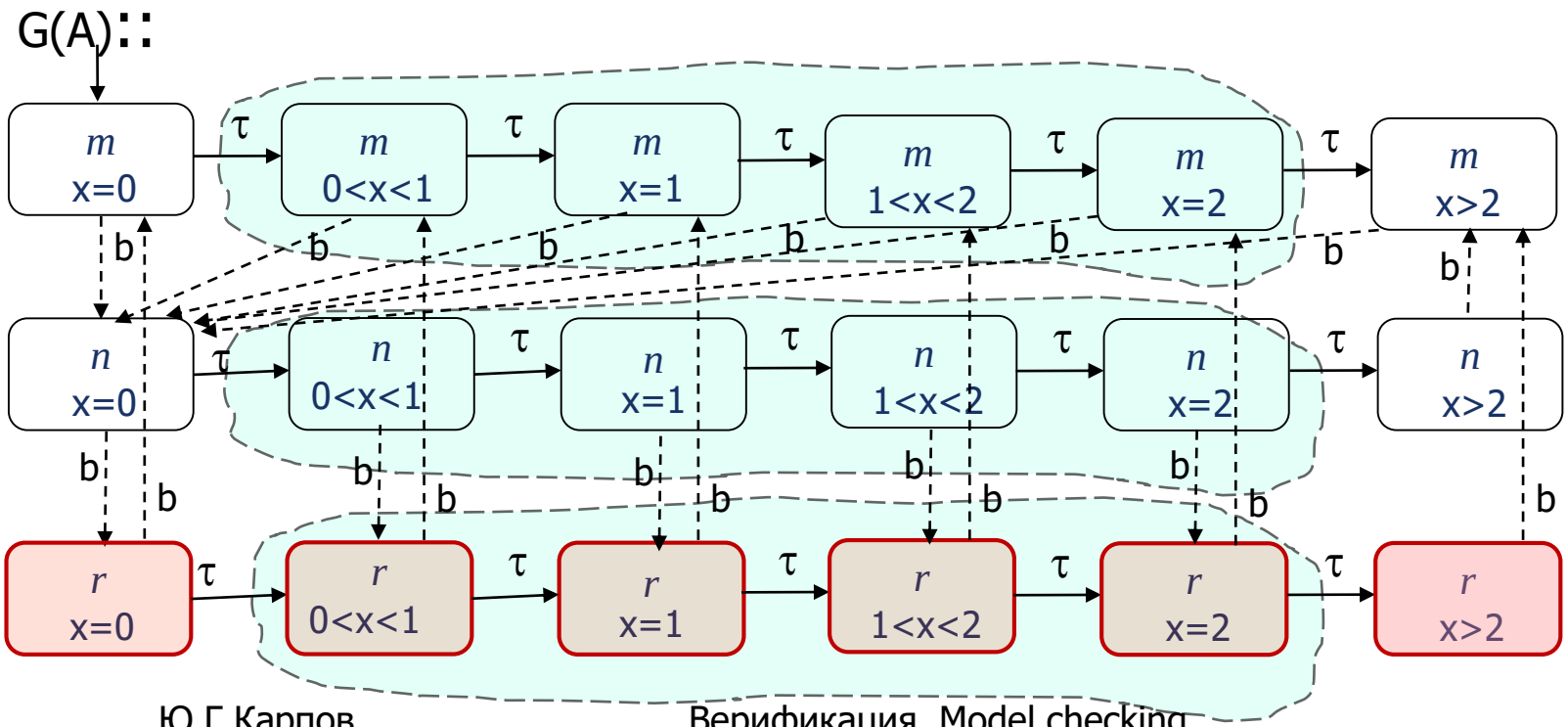
В каждом регионе любое ограничение на часы и любая темпоральная формула сохраняют свое значение (И/Л)

Такое представление конечно, но огромно. Экспоненциальный рост от числа часов и верхней границы k

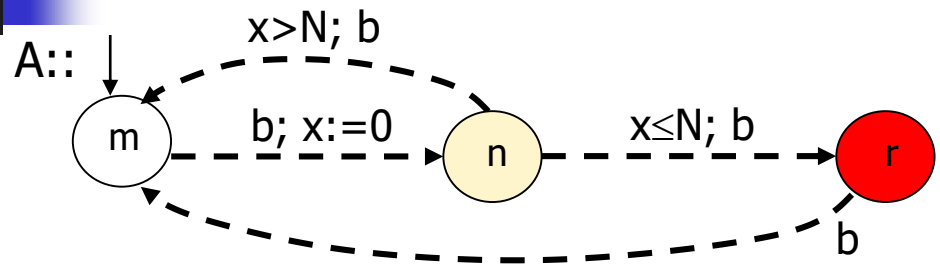
Временные зоны: классы эквивалентности временных регионов (один локальный таймер)

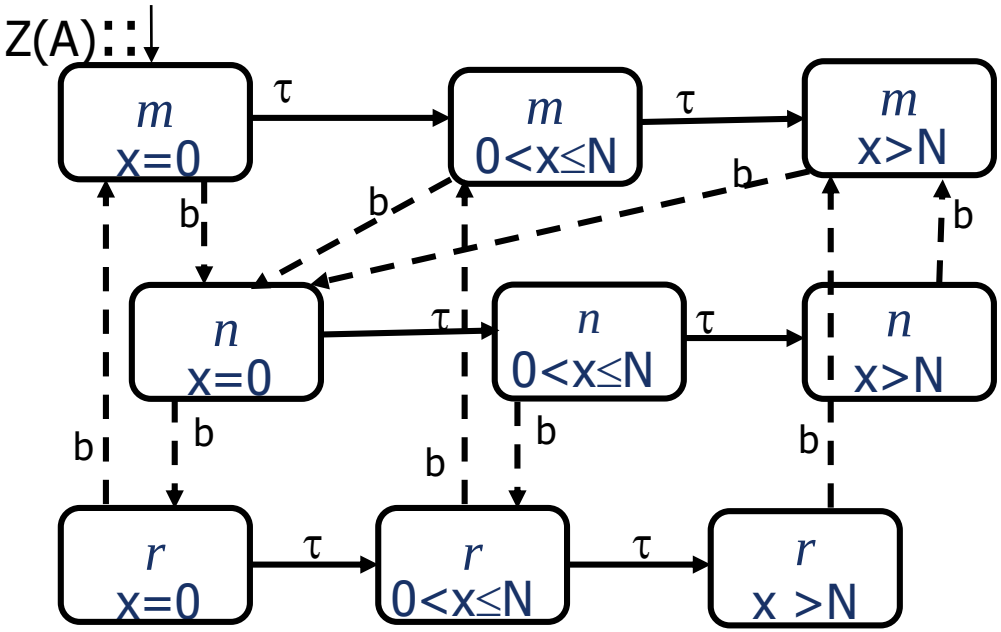
- Многие регионы эквивалентны. Можно выделить ЗОНЫ – классы эквивалентности регионов относительно ограничений и инвариантов временного автомата. В каждой зоне все свойства должны совпадать





Граф зон временного автомата, различающего одинарный и двойной щелчок мыши



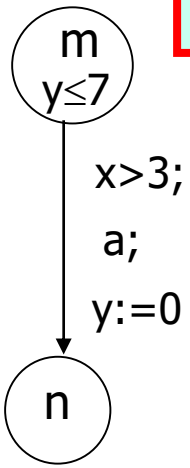


Граф зон строится объединением регионов в классы эквивалентности

- При любом N граф зон этого автомата остается неизменным
- Проверка любой темпоральной формулы CTL может быть выполнена на графе зон, если во временных неравенствах будут только константы 0 и N.

Два локальных таймера: преобразование временных зон на одном переходе временного автомата

Временные зоны: классы эквивалентности временных регионов



Исходная зона: $Z_1 = (1 \le x \le 5) \wedge (2 \le y \le 3)$

Течение времени в локации m, ограниченное инвариантом локации и x-y=const:

$Z_2 = (1 \le x) \wedge (2 \le y \le 7) \wedge (-3 \le y - x \le 2)$

$x-3 \le y \le x+2$

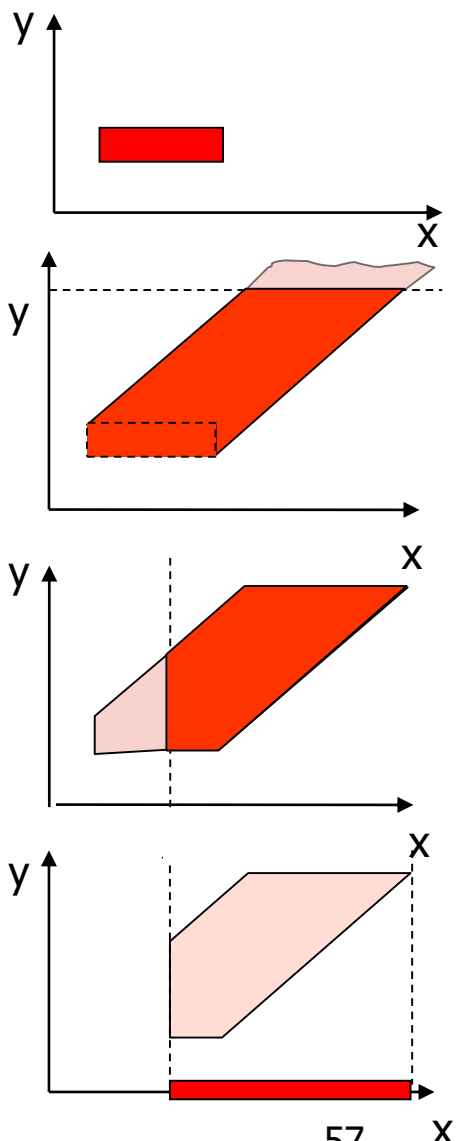
Ограничение (защита) перехода:

$Z_3 = (x > 3) \wedge (2 \le y \le 7) \wedge (-3 \le y-x \le 2)$

Сброс таймера при выполнении перехода:

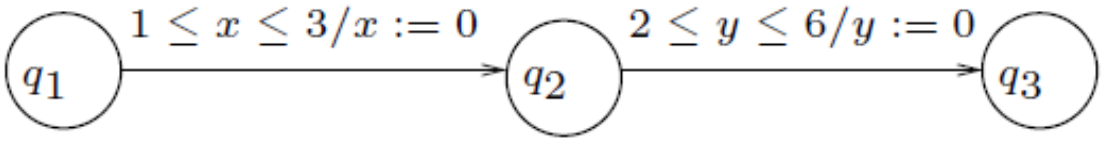
$Z_4 = (3 < x \le 10) \wedge (y = 0)$

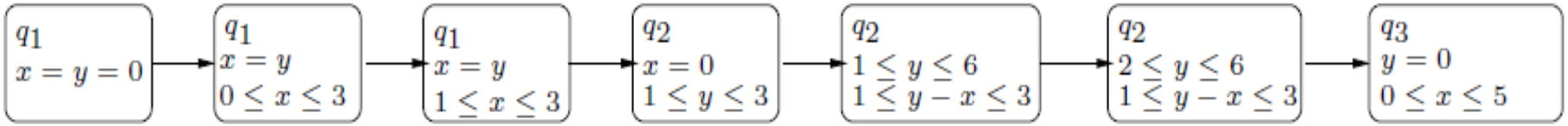
Всевозможные вычисления будут Z1->Z2->Z4

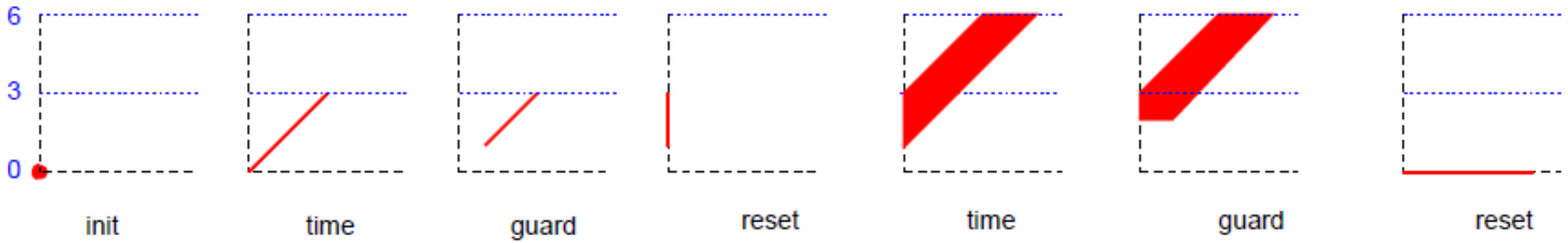


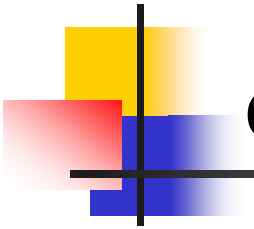
Пример моделирования переходов зональным автоматом

- Построение “зонального” автомата для временного автомата с двумя переходами









Символьная Верификация Временных Автоматов

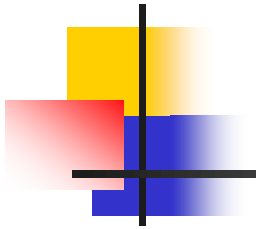
Каждая временная область может быть представлена как логическая формула над линейными *clock constraints* (ограничениями таймеров) – их дизъюнкции и конъюнкции

Идея: нельзя ли анализировать ТА без явного априорного конструирования разбиений на регионы, а просто символически манипулируя всеми ограничениями на показания таймеров?

Такой метод предложили в 1994 г. T. Henzinger и др. – символьная верификация Временн'ых Автоматов

Заключение

- Для систем реального времени требуется анализ временн'ых характеристик поведения систем, чего не может дать обычная CTL.
- Временн'ые автоматы удобны для спецификации широкого класса систем реального времени.
- Можно представить все поведения временного автомата в замкнутом КОНЕЧНОМ виде с помощью регионов, однако такое представление требует огромного объема информации.
- Верификация Временного автомата относительно CTL-формулы сводится к верификации соответствующего графа регионов относительно CTL–формулы (с некоторыми добавлениями).
- Исследования в этой области начались недавно ("*Timed automata are recent models…*" B.Berard, *System and software verification*). Исследования продолжаются, в основном, они направлены на поиск методов более компактного представления временных автоматов.
- Инструменты автоматизированной верификации систем реального времени (KRONOS, UPPAAL, …) сами выполняют построение параллельной композиции временных автоматов, построение графа регионов и проверку формул TCTL для не очень больших систем.



Спасибо за внимание

Timed Temporal Logic – структура формул

TCTL – Timed CTL – естественное расширение операторов U, F, ... логики CTL количественной информацией.

Грамматика TCTL (= CTL + Time):

$\phi ::= p \mid \alpha \mid \neg\phi \mid \phi \wedge \phi \mid z \text{ in } \phi \mid E[\phi \, U \, \phi] \mid A[\phi \, U \, \phi]$

p – атомарный предикат
α - ограничение на таймеры и формульные часы
z – формульные часы
z in ϕ - введение новых часов в формулу ϕ
$E[\phi \, U \, \phi]$, $A[\phi \, U \, \phi]$ – как в CTL

Обозначение: $E[\phi \, U_{\alpha} \psi] \equiv z \text{ in } E[(\phi \,\&\, \alpha) U \, \psi]$

Выводимые операторы $EF_{\alpha}\phi \equiv E[\text{True} \, U_{\alpha} \, \phi]$ и т.д.

Формулы TCTL включают $E[\phi \, U_{\sim k} \psi]$, $A[\phi \, U_{\sim k} \psi]$, $EF_{\sim k}\phi$, $EG_{\sim k}\phi$, $AF_{\sim k}\phi$, $AG_{\sim k}\phi$
где ~ - любой символ из $\{<, \leq, =, \geq, >\}$ и k – рациональное число

Для верификации временного автомата A можно анализировать R(A)

Примеры свойств реального времени

1. [p $U_{<2}$ q] – p истинно непрерывно до тех пор, пока не станет истинно q, и истинность q наступит не позднее, чем через 2 единицы времени

2. AG(problem $\Rightarrow$ $AG_{\geq 5}$ alarm) – как только проблема возникла, сигнал alarm зазвучит сразу и будет звучать не менее 5

3. AG($\neg$far $\Rightarrow$ $AF_{< 7}$ far) – поезд покинет область контроля не позже, чем через 7

4. AG [send(m) $\Rightarrow$ $AF_{<5}$ receive(r_m)] – подтверждение приходит в пределах 5

5. EG [send(m) $\Rightarrow$ $AF_{> 4}$ receive(r_m)] - подтверждение может быть получено более, чем за 4

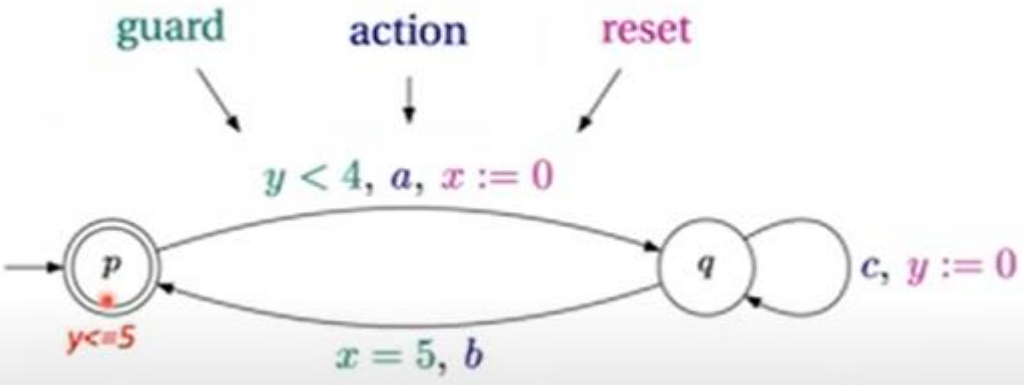
6. AG [$AF_{= 15}$ tick] – тики следуют периодически точно через 15 е.в. (но, кроме того, могут быть и в промежутках)

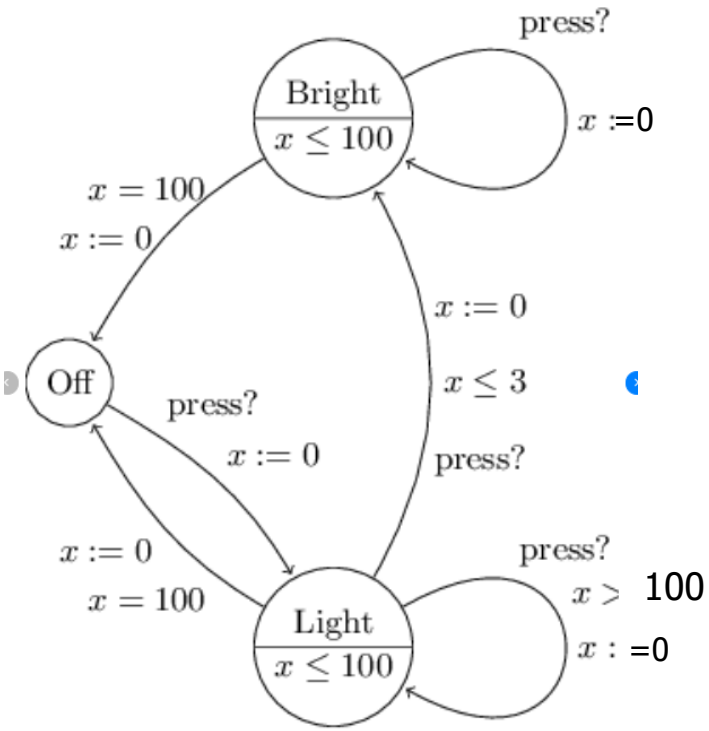
7. AG ($x \leq y$) - таймер x всегда не больше таймера y

8. A [off U $x \geq 3$] по любому пути из начального состояния если светофор выключен, то он будет выключен до тех пор, пока таймер x не будет иметь значение ≥ 3

x, y: clocks

[Alur & Dill - 1990's]




press?
Bright
$x \leq 100$
$x := 0$
$x = 100$
$x := 0$
Off
press?
$x := 0$
$x := 0$
$x \leq 3$
press?
$x := 0$
$x = 100$
press?
$x >$ 100
Light
$x \leq 100$
x : =0