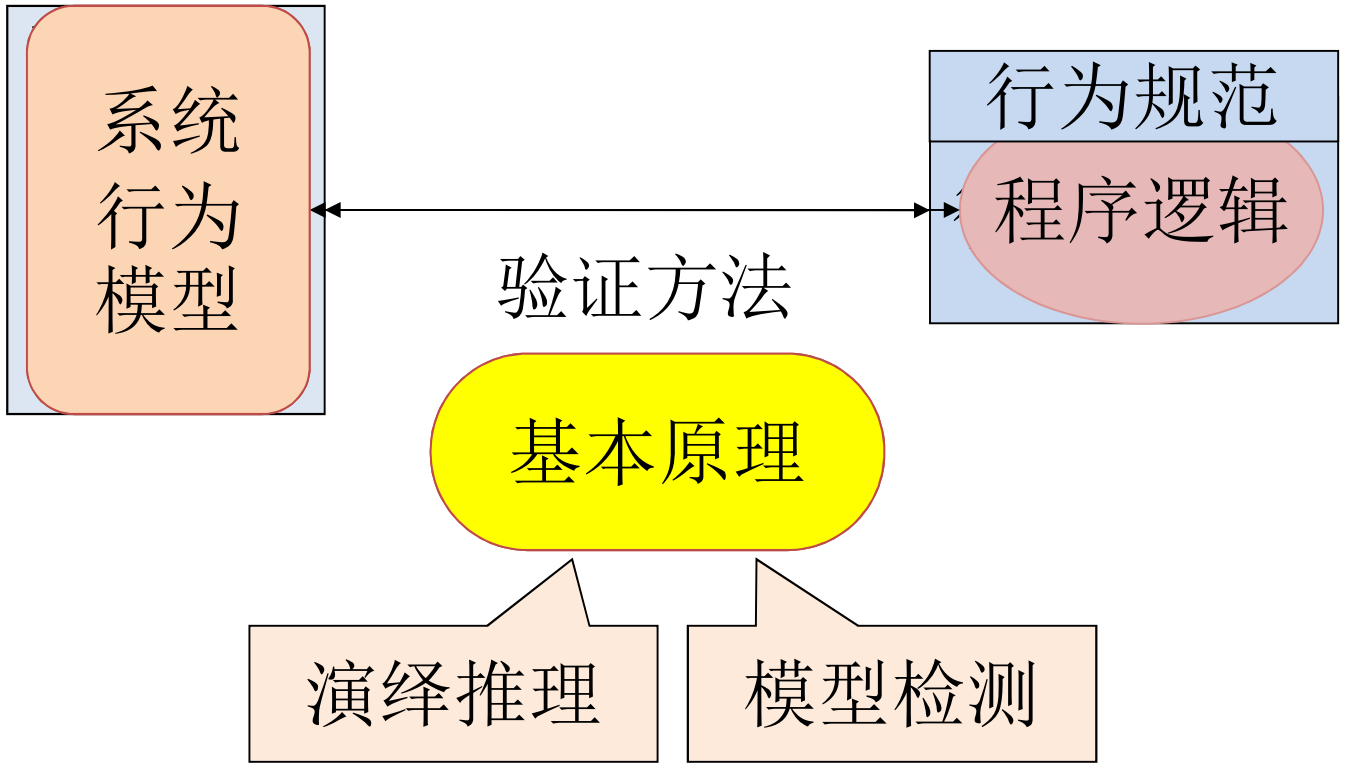
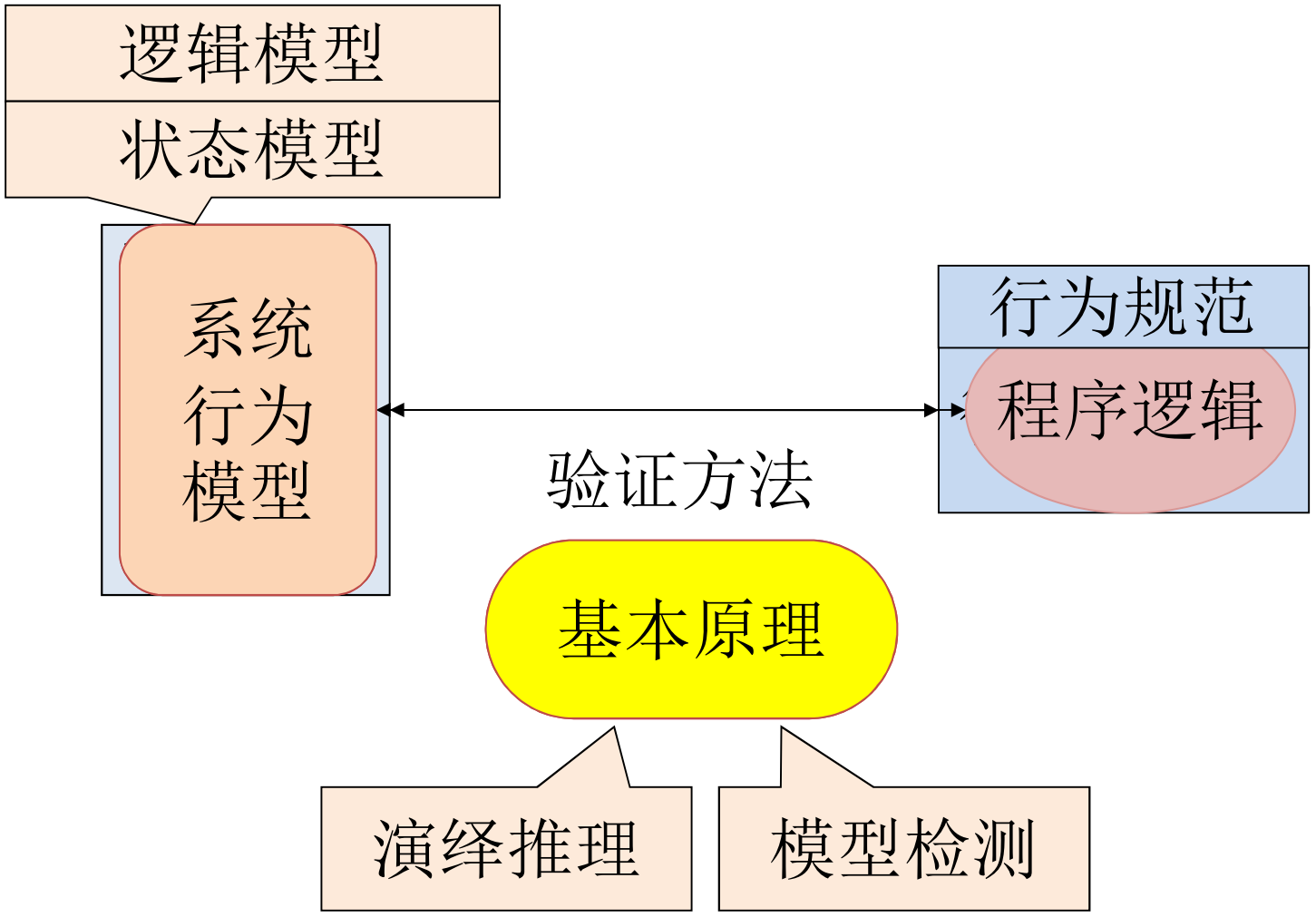
基于一阶逻辑的迁移系统

中国科学院软件研究所
计算机科学国家重点实验室
张文辉
http://lcs.ios.ac.cn/~zwh/

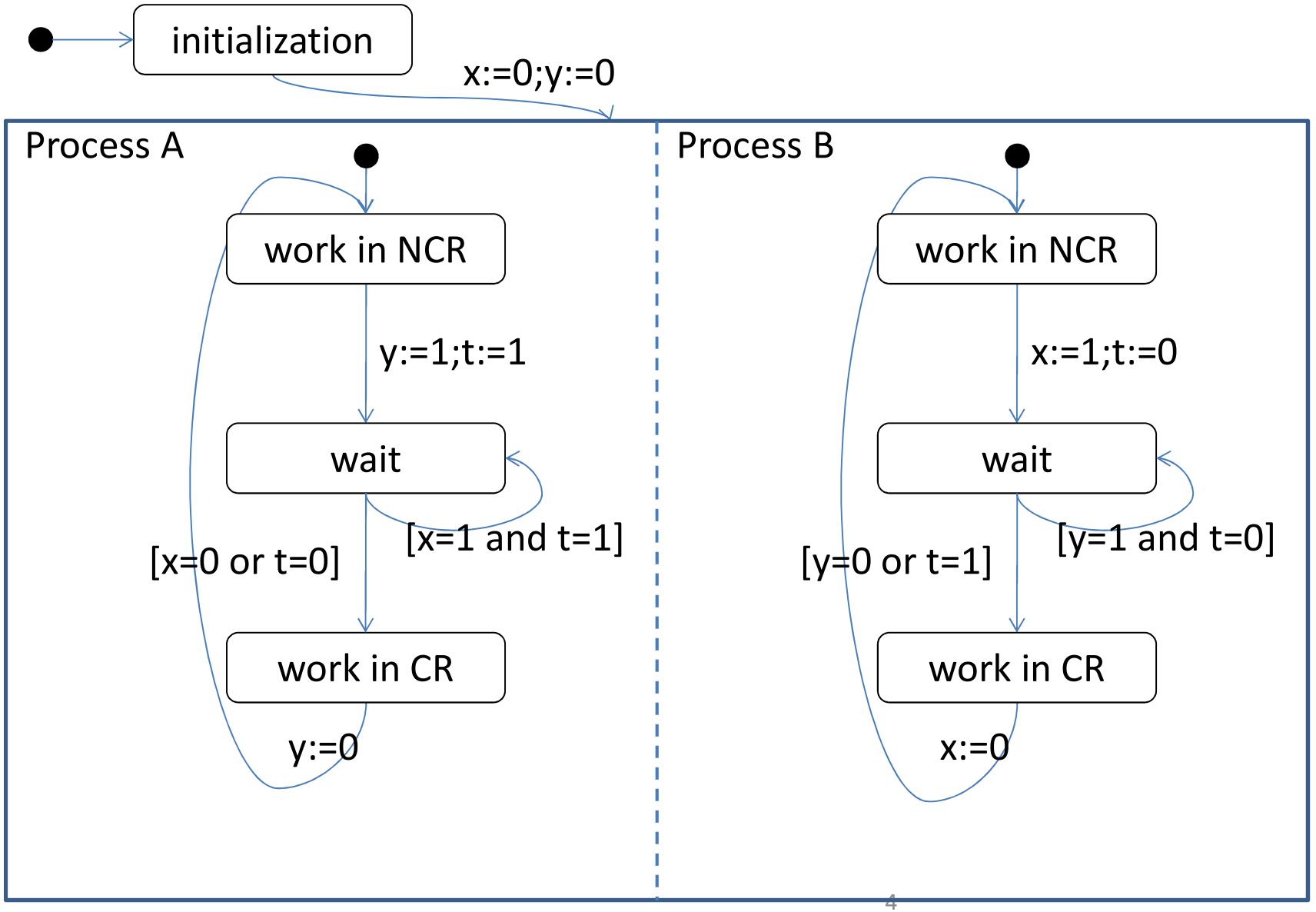
课程内容



课程内容



例子-互斥：状态图(State Diagram)



例子-互斥：算法

VAR: a: {NCR,wait,CR}; b: {NCR,wait,CR};
x: 0..1; y: 0..1; t: 0..1;
INIT: a=NCR; b=NCR;
x=0; y=0;

Process A:

a=NCR	→ (a,y,t):=(wait,1,1);
a=wait∧(x=0∨t=0)	→ (a):=(CR);
a=wait∧¬(x=0∨t=0)	→ (a):=(wait);
a=CR	→ (a,y):=(NCR,0);

Process B:

b=NCR	→ (b,x,t):=(wait,1,0);
b=wait∧(y=0∨t=1)	→ (b):=(CR);
b=wait∧¬(y=0∨t=1)	→ (b):=(wait);
b=CR	→ (b,x):=(NCR,0);

例子-互斥：算法

S: { <a,b,x,y,t> | a,b ∈ {NCR,wait,CR}, x,y,t ∈ {0,1} }
I: { <NCR,NCR,0,0,t> | t ∈ {0,1} }

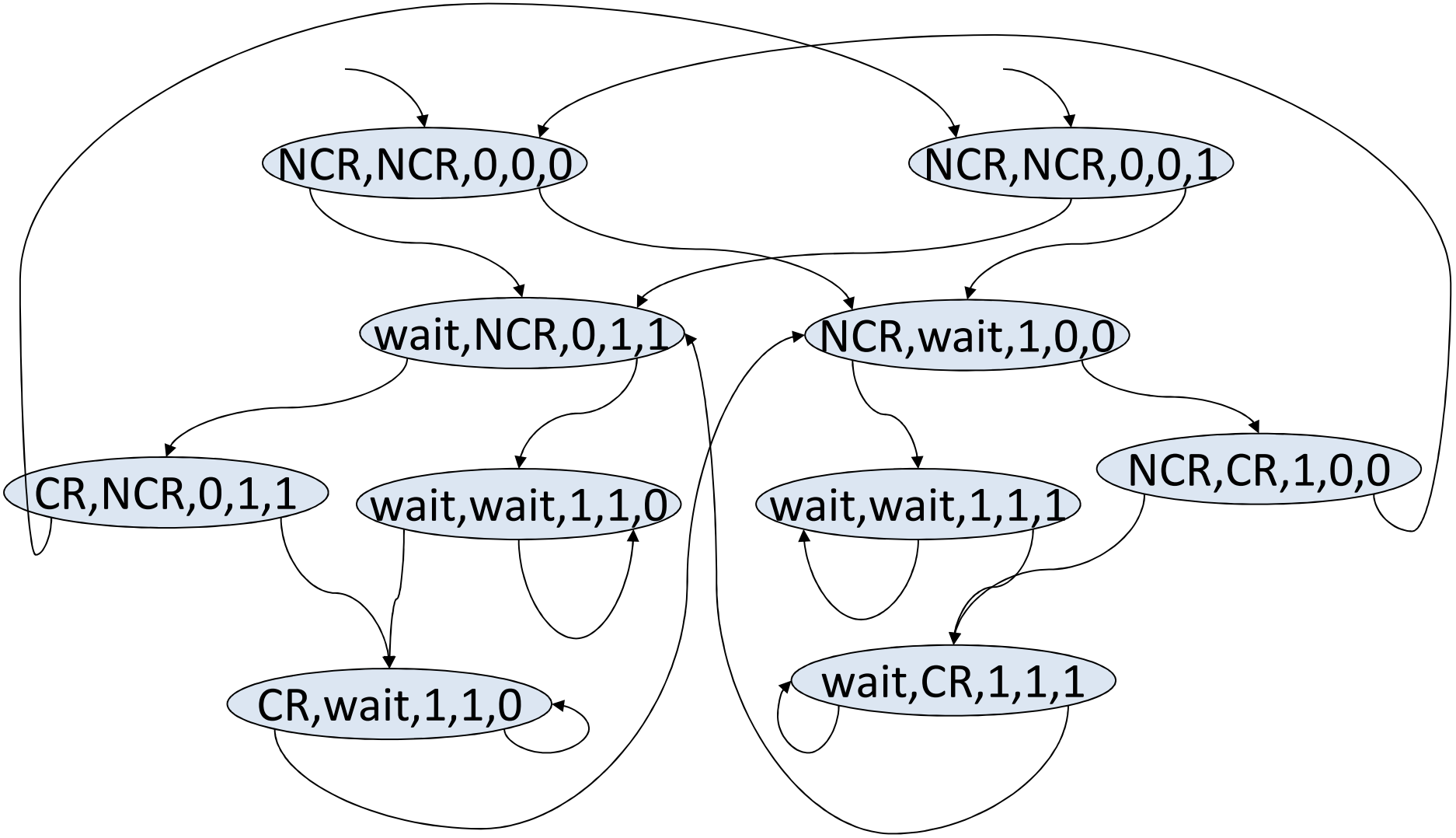
Process A:

{ (<NCR,b,x,y,t>,<wait,b,x,1,1>) | b ∈ {NCR,wait,CR} and x,y,t ∈ {0,1} }
{ (<wait,b,x,y,t>,<CR,b,x,y,t>) | b ∈ {…} and x,y,t ∈ {…} and x=0∨t=0 }
{ (<wait,b,x,y,t>,<wait,b,x,y,t>) | b ∈ {…} and x,y,t ∈ {…} and ¬(x=0∨t=0) }
{ (<CR,b,x,y,t>,<NCR,b,x,0,t>) | b ∈ {…} and x,y,t ∈ {…} }

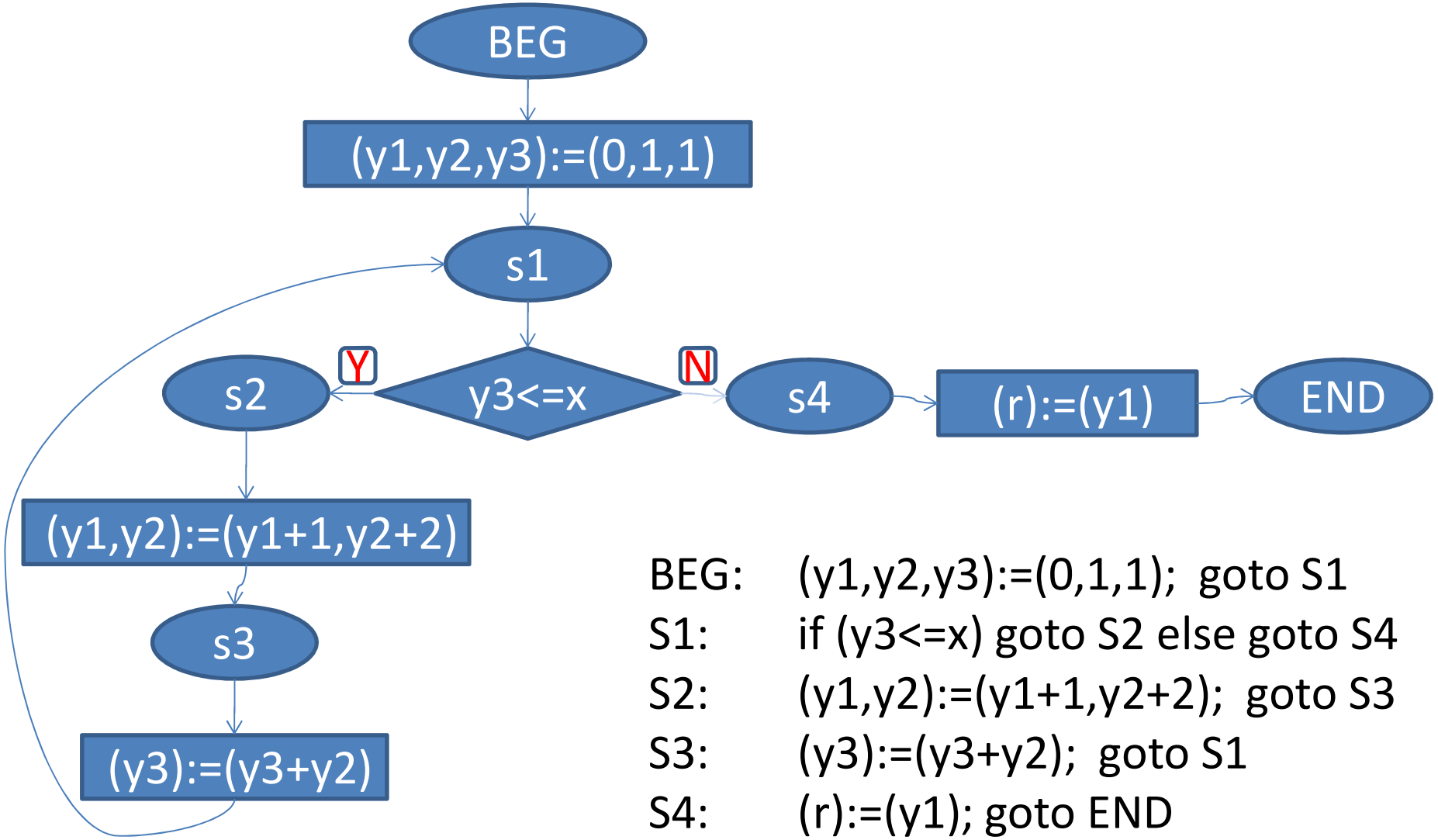
Process B:

{ (<a,NCR,x,y,t>,<a,wait,1,y,0>) | a ∈ {NCR,wait,CR} and x,y,t ∈ {0,1} }
{ (<a,wait,x,y,t>,<a,CR,x,y,t>) | a ∈ {…} and x,y,t ∈ {…} and y=0∨t=1 }
{ (<a,wait,x,y,t>,<a,wait,x,y,t>) | a ∈ {…} and x,y,t ∈ {…} and ¬(y=0∨t=1) }
{ (<a,CR,x,y,t>,<a,NCR,0,y,t>) | a ∈ {…} and x,y,t ∈ {…} }

Kripke Structure



例子-整树平方根：设计

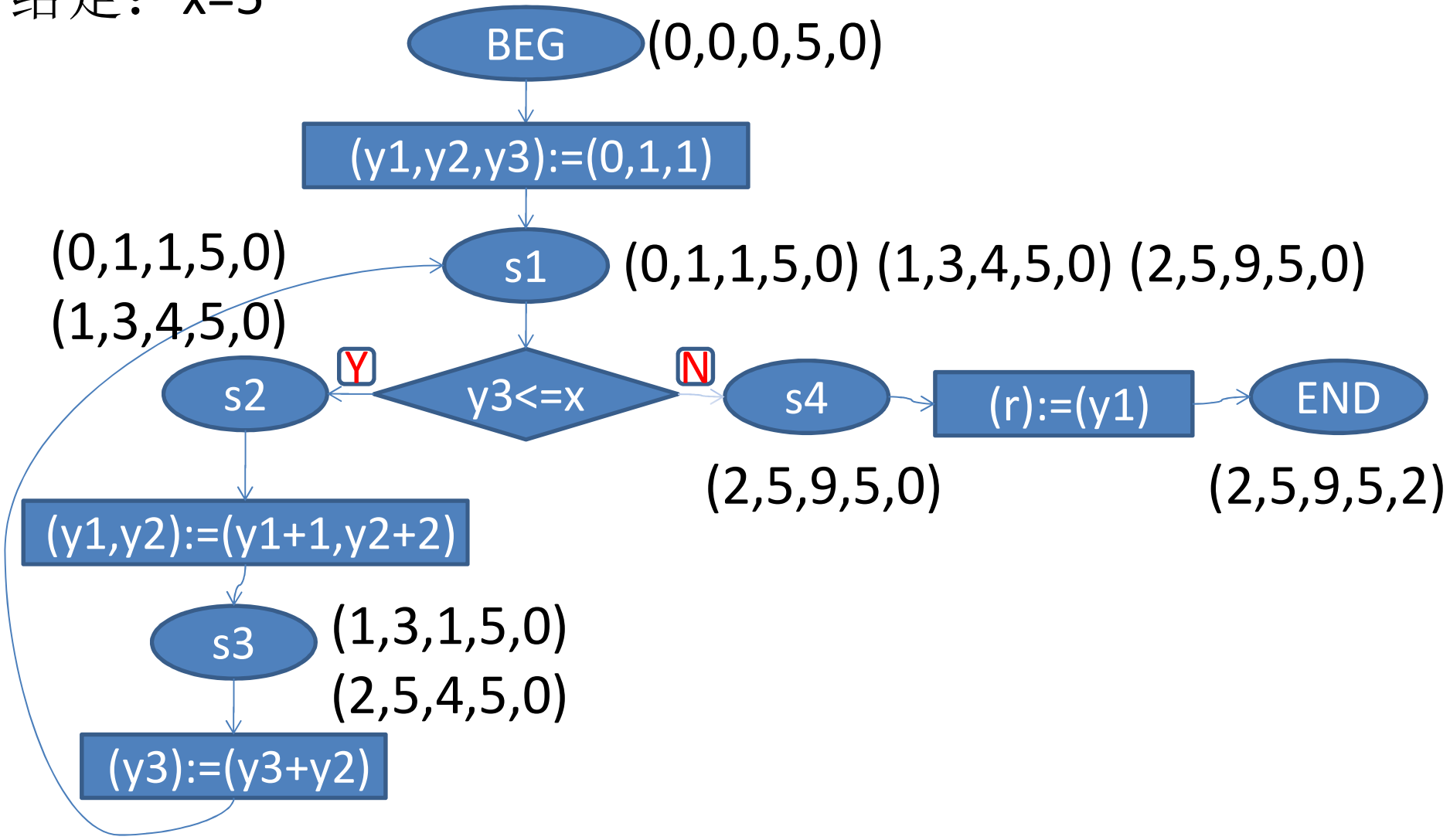


BEG: (y1,y2,y3):=(0,1,1); goto S1
S1: if (y3<=x) goto S2 else goto S4
S2: (y1,y2):=(y1+1,y2+2); goto S3
S3: (y3):=(y3+y2); goto S1
S4: (r):=(y1); goto END

例1a - 整数平方根(计算过程实例)

状态：LAB+(y1,y2,y3,x,r)

给定：x=5



Motivation

迁移系统和计算的高层描述

基础知识

- 集合、关系、函数、谓词逻辑

logic

谓词逻辑：公式

- B=(F,P)
- 项
  - 个体变元
  - 个体常元
  - 函数符号(n元)
- 公式
  - 谓词符号(n元)、量词符号、联结词符号、辅助符号

由B=(F,P)生成的公式集合

WFF – Well formed formulas

QFF – Quantifier-free formulas

谓词逻辑：解释

用F(n)表示F中n元函数符号集合

用P(n)表示P中n元谓词符号集合

解释 $I=(D,I_0)$

– 论域D

– I_0：$F(n) \rightarrow (D^n \rightarrow D)$　　n元函数

– I_0：$P(n) \rightarrow (\text{PowerSet}(D^n))$　　n元关系

谓词逻辑：赋值

赋值/状态　　σ ： $(X \to D)$

值：　　$t, I(t), I(t)(\sigma)$

真假值：　　$\varphi, I(\varphi), I(\varphi)(\sigma), \sigma \mid =_I \varphi$

谓词逻辑：替换

替换

- $\varphi' = \varphi[x_1/t_1]\ldots[x_n/t_n]$
- $\sigma' = \sigma[x_1/I(t_1)(\sigma)]\ldots[x_n/I(t_n)(\sigma)]$

定理：$I(\varphi')(\sigma) = I(\varphi)(\sigma')$

例子

- φ: x>y
- I
- σ: (x,y)=(5,2)

- $I(\varphi)(\sigma)$ = 5>2

- $\varphi' = \varphi[x/x+1] = (x+1)>y$
- $I(\varphi')(\sigma)$ = 5+1>2
- σ': (x,y)= $\sigma[x/I(x+1)\sigma]$=(6,2)
- $I(\varphi)(\sigma')$ = 6>2
- $I(\varphi')(\sigma) = I(\varphi)(\sigma')$

小结

- $B=(F,P)$：　　基本符号集
- t, φ：　　项、公式

- $I=(D,I_0)$：　　解释
- $\sigma \in (X \to D)$：　　赋值/状态
- $I(t)(\sigma)$, $I(\varphi)(\sigma)$：　　值、真假值

- $I(\varphi[x_1/t_1]\ldots[x_n/t_n])(\sigma)$
- $I(\varphi)(\sigma[x_1/I(t_1)(\sigma)]\ldots[x_n/I(t_n)(\sigma)])$

逻辑与程序(赋值语句)

- x:=t

- {σ} x:=t {σ’}
- 语义: $\sigma' = \sigma[x/I(t)(\sigma)]$

- {φ’} x:=t {φ}
- 前后断言定义: $I(\varphi')(\sigma) \rightarrow I(\varphi)(\sigma')$

- 前后断言推理: $\varphi' \rightarrow \varphi[x/t]$
- 最弱前断言φ’= φ[x/t]: $I(\varphi')(\sigma) \leftrightarrow I(\varphi)(\sigma')$

例子

- x:=x+1

- {(5,2)} x:=x+1 {(6,2)}
- 语义: $\sigma' = \sigma[x/I(x+1)(\sigma)]$

- $\{\varphi'\}$ x:=x+1 {x>y}
- 前后断言: $I(\varphi')(\sigma) \rightarrow I(x>y)(\sigma')$

- $\varphi' \rightarrow x+1>y$
- 在φ'的位置可用最弱前断言$\varphi[x/x+1]$即x+1>y

内容：一阶迁移模型

- 卫式迁移模型
- 公平卫式迁移模型及实例
- 流程图模型
- 结构化程序模型

(I) 卫式迁移模型

- 定义
- 性质
- 卫式迁移模型与标号Kripke模型的关系

(I.a) 卫式迁移模型

一阶逻辑的扩充

主要有卫式和赋值

在一阶逻辑的基础上，增加以下辅助符号集：

{:=, →}

迁移

给定B=(F,P)和变量集合V。
定义(B,V)上的迁移如下：

$\phi \rightarrow (x1,\ldots,xn) := (t1,\ldots,tn)$

其中　　$t1,\ldots,tn \in T_B$ 且在t1,...,tn中出现的变量在 V中
$\phi \in QFF_B$且在ϕ中出现的变量在 V中
$x1,\ldots,xn \in V$且这些变量不重复

卫式迁移模型

一个(B,V)上的卫式迁移模型M是一个二元组(T,Θ)

其中

T为(B,V)上迁移的有限集合，

$\Theta \in QFF_B$ 且Θ中出现的变量在 V中，是模型的初始条件

例子-互斥

T:	$a=NCR \rightarrow$	$(y,t,a):=(1,1,wait);$
	$a=wait \wedge (x=0 \vee t=0) \rightarrow$	$(a):=(CR);$
	$a=wait \wedge \neg(x=0 \vee t=0) \rightarrow$	$(a):=(wait);$
	$a=CR \rightarrow$	$(y,a):=(0, NCR);$
	$b=NCR \rightarrow$	$(x,t,b):=(1,0,wait);$
	$b=wait \wedge (y=0 \vee t=1) \rightarrow$	$(b):=(CR);$
	$b=wait \wedge \neg (y=0 \vee t=1) \rightarrow$	$(b):=(wait);$
	$b=CR \rightarrow$	$(x,b):=(0, NCR);$
Θ:	$x=0 \wedge y=0 \wedge a=NCR \wedge b=NCR$	

例子-整树平方根

T:	pc=BEG → (y1,y2,y3,pc):=(0,1,1,S1) pc=S1∧(y3<=x) → (pc):=(S2) pc=S1∧¬ (y3<=x) → (pc):=(S4) pc=S2→ (y1,y2,pc):=(y1+1,y2+2,S3) pc=S3→ (y3,pc):=(y3+y2,S1) pc=S4→ (r,pc):=(y1,END) pc=END→ (pc):=(END)
Θ:	pc=BEG

模型

状态集合

状态迁移关系

计算/运行

系统状态

给定$M=(T,\Theta)$。

假定$B=(F,P)$的解释$I=(D,I_0)$已经给定。

模型的系统状态：变量状态

变量状态空间：V中变量取值的组合

$\Sigma=\{\sigma \mid \sigma(x)\in D,x\in V\}$

状态迁移关系

迁移$\phi \rightarrow (v1,v2,...,vn):=(e1,e2,...,en)$在状态$\sigma$可执行，
当且仅当$\sigma \models_l \phi$。

设t是$\phi \rightarrow (v1,v2,...,vn):=(e1,e2,...,en)$ 为一个迁移。

$\sigma \rightarrow^t \sigma'$当且仅当
t在状态σ可执行且$\sigma'=\sigma[v1/l(e1)(\sigma)]...[vn/l(en)(\sigma)]$

状态迁移关系

$\sigma \rightarrow \sigma'$当且仅当

(1) 存在$t \in T$使得$\sigma \rightarrow^{t} \sigma'$

(2) 不存在可执行迁移且$\sigma = \sigma'$

计算

模型M=(T,Θ)的一条路径是
状态集Σ上的一个无穷序列$\sigma_0\sigma_1$ .....
满足对任意i有$\sigma_i \rightarrow \sigma_{i+1}$

模型M=(T,Θ)的一个计算是
状态集Σ上的一个无穷序列$\sigma_0\sigma_1$ .....
满足$\sigma_0 |=_l \Theta$且对任意i有$\sigma_i \rightarrow \sigma_{i+1}$

模型M的计算集合记为[[M]]

可达状态集

定义 $\rightarrow^*$ 为 $\rightarrow$ 的传递自反闭包

模型M在初始变量状态为σ时
运行可到达的状态的集合为 $\{ \sigma' \mid \sigma \rightarrow^* \sigma' \}$

模型M的可达状态集合 $\{ \sigma' \mid \sigma \rightarrow^* \sigma',\ \sigma \models_I \Theta \}$ 记为rh(M)

模型

给定模型$M=(T,\Theta)$：

状态集：Σ

迁移关系：$\rightarrow$

初始状态集：$\{\sigma \mid \sigma|=_{|}\Theta\}$

标号函数：

例子-互斥

T:	a=NCR→	(y,t,a):=(1,1,wait);
	a=wait∧(x=0 ∨ t=0) →	(a):=(CR);
	a=wait ∧¬(x=0∨t=0) →	(a):=(wait);
	a=CR →	(y,a):=(0, NCR);
	b=NCR →	(x,t,b):=(1,0, wait);
	b=wait ∧(y=0 ∨ t=1) →	(b):=(CR);
	b=wait ∧¬ (y=0 ∨ t=1) →	(b):=(wait);
	b=CR →	(x,b):=(0, NCR);
Θ:	x=0 ∧ y=0 ∧ a=NCR ∧ b=NCR	

B=(F,P) F={0,1,NCR,wait,CR} P={=} V={a,b,x,y,t}	I=(N,I_0) I_0(0) = **0** I_0(1) = **1** I_0(=) = **=** I_0(NCR) = **0** I_0(wait) = **1** I_0(CR) = **2**

例子-整树平方根

T:	pc=BEG →	(y1,y2,y3,pc):=(0,1,1,S1)
	pc=S1∧(y3<=x) →	(pc):=(S2)
	pc=S1∧¬ (y3<=x) →	(pc):=(S4)
	pc=S2→	(y1,y2,pc):=(y1+1,y2+2,S3)
	pc=S3→	(y3,pc):=(y3+y2,S1)
	pc=S4→	(r,pc):=(y1,END)
	pc=END→	(pc):=(END)
Θ:	pc=BEG	

B=(F,P)

F={+,0,1,BEG,S1,…,S4,END}

P={<=}

V={y1,y2,y3,x,r,pc}

I=(N,I_0)

I_0(+) =	**+**	I_0(BEG) =	**0**
I_0(0) =	**0**	I_0(S1) =	**1**
I_0(1) =	**1**	…	
I_0(<=) =	≤	I_0(END) =	**5**

(I.b) 正确性相关性质

φ是模型M的安全性质当且仅当$\forall \pi \in [[M]].\forall k.\pi_k \models_l \varphi$。

φ是模型M的必达性质当且仅当$\forall \pi \in [[M]].\exists k.\pi_k \models_l \varphi$。

模型性质的分析

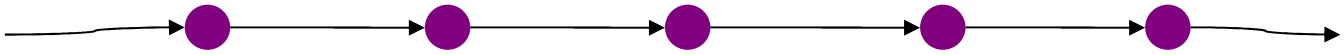
基于规则和推理

基于状态搜索算法(针对有穷状态系统)

(I.c) 与 标号Kripke结构的等价性

x=1 x=1

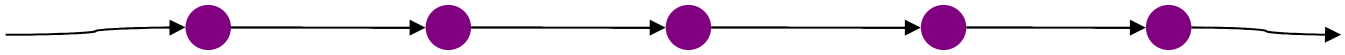
y=1 y=2



p: x=1

q: y=2

{p} {p,q}



卫式迁移系统– 标号Kripke结构: 等价性定义

给定模型M=(T,Θ)和标号Kripke结构K=(S,R,I,L)

设AP={p1,…,pn}, BP={Ψ1,…, Ψn } $\subseteq$ QFF

设 ζ ： AP→BP 为一一对应关系

$\forall \pi \in [[K]], \exists \pi' \in [[M]], \forall k.(\pi_k \models p \leftrightarrow \pi'_k \models \zeta(p))$ 且

$\forall \pi \in [[M]], \exists \pi' \in [[K]], \forall k.(\pi_k \models \Psi \leftrightarrow \pi'_k \models \zeta^{-1}(\Psi))$

则称M与K为 ζ 计算等价。

卫式迁移系统–标号Kripke结构: 性质

定理：

设$\phi \in L(BP)$。
若M与K为ζ计算等价，
则
ϕ是M的安全性质当且仅当$\zeta^{-1}(\phi)$是K的安全性质，
ϕ是M的必达性质当且仅当$\zeta^{-1}(\phi)$是K的必达性质

卫式迁移系统–标号Kripke结构: 构造

给定模型M=(T,Θ)：

给定BP={Ψ_1,…, Ψ_n }

定义AP={p_1 ,…,p_n } 并定义 ζ：AP→BP 为 ζ (p_i)= Ψ_i

定义标号Kripke结构K(M)如下：

状态集：Σ

迁移关系：→

初始状态集：{ σ | σ|=$_l$Θ }

标号函数：p_i ∈L(σ) 当且仅当σ|=$_l\Psi_i$

卫式迁移系统–标号Kripke结构: 具体构造

给定模型M=(T,Θ)：

(B,V)

B=(F,P); V={v1,…,vn}

σ: V → D

σ: (V1 → D1) ∪…∪ (Vn → Dn)　　按不同变量类型

卫式迁移系统–标号Kripke结构: 具体构造

状态集:

用(a1,…,an)代表σ: σ(v1)=a1, …, σ(vn)=an

Σ= {(a1,…,an) | ai $\in$D }

Σ= {(a1,…,an) | a1 $\in$D1, …, an $\in$Dn}

卫式迁移系统–标号Kripke结构: 具体构造

迁移关系

$((a1,\ldots,an),(a1',\ldots,an')) \in \rightarrow$

$\Leftrightarrow$

$\sigma \rightarrow \sigma'$

$\sigma(v1)=a1, \ldots, \sigma(vn)=an$

$\sigma'(v1)=a1', \ldots, \sigma'(vn)=an'$

卫式迁移系统–标号Kripke结构: 具体构造

初始状态集:

$\{(a1,\ldots,an) \mid \sigma(v1)=a1, \ldots, \sigma(vn)=an, \sigma \models_l \Theta\}$

标号函数： $p_i \in L(\sigma)$ 当且仅当$\sigma \models_l \Psi_i$

卫式迁移系统–标号Kripke结构:

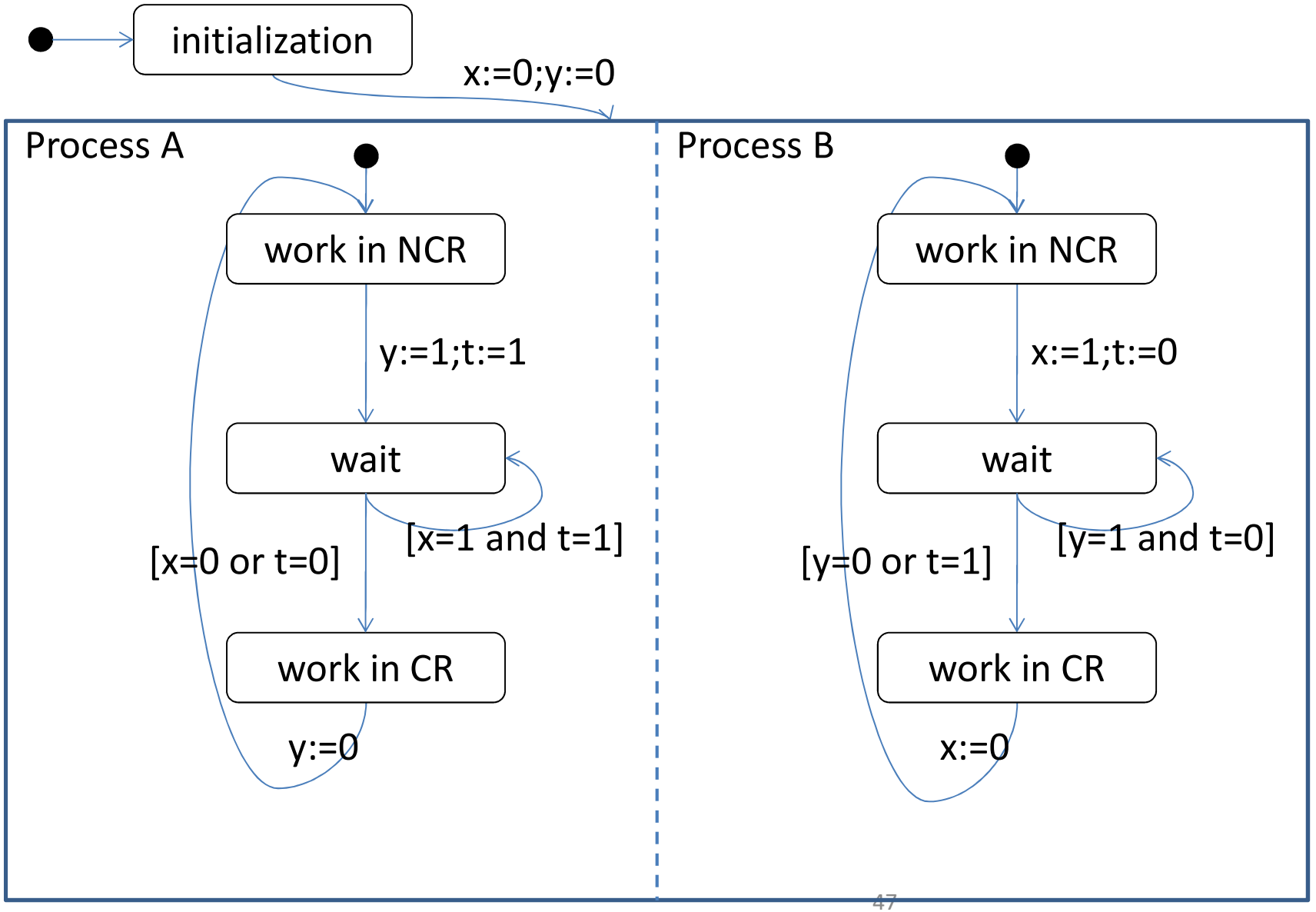
定理：

M与K(M)为ς计算等价.

推论：

ϕ是M的安全性质当且仅当ς^{-1}(ϕ)是K(M)的安全性质
ϕ是M的必达性质当且仅当ς^{-1}(ϕ)是K(M)的必达性质

例子-互斥：状态图(State Diagram)



例子-互斥

A:

a=NCR$\rightarrow$	(y,t,a):=(1,1,wait);
a=wait$\wedge$(x=0 $\vee$ t=0) $\rightarrow$	(a):=(CR);
a=wait $\wedge\neg$(x=0$\vee$t=0) $\rightarrow$	(a):=(wait);
a=CR $\rightarrow$	(y,a):=(0, NCR);

B:

b=NCR $\rightarrow$	(x,t,b):=(1,0, wait);
b=wait $\wedge$(y=0 $\vee$ t=1) $\rightarrow$	(b):=(CR);
b=wait $\wedge\neg$ (y=0 $\vee$ t=1) $\rightarrow$	(b):=(wait);
b=CR $\rightarrow$	(x,b):=(0, NCR);

例子-互斥

A:		初始状态:
a=NCR→	(y,t,a):=(1,1,wait);	x=0
a=wait∧(x=0 ∨ t=0) →	(a):=(CR);	y=0
a=wait ∧¬(x=0∨t=0) →	(a):=(wait);	
a=CR →	(y,a):=(0, NCR);	a=NCR
B:		b=NCR
b=NCR →	(x,t,b):=(1,0, wait);	
b=wait ∧(y=0 ∨ t=1) →	(b):=(CR);	
b=wait ∧¬ (y=0 ∨ t=1) →	(b):=(wait);	
b=CR →	(x,b):=(0, NCR);	

例子-互斥

迁移关系		初始状态
a=NCR$\rightarrow$	(y,t,a):=(1,1,wait);	x=0
a=wait$\wedge$(x=0 $\vee$ t=0) $\rightarrow$	(a):=(CR);	y=0
a=wait $\wedge\neg$(x=0$\vee$t=0) $\rightarrow$	(a):=(wait);	
a=CR $\rightarrow$	(y,a):=(0, NCR);	a=NCR
		b=NCR
b=NCR $\rightarrow$	(x,t,b):=(1,0, wait);	
b=wait $\wedge$(y=0 $\vee$ t=1) $\rightarrow$	(b):=(CR);	
b=wait $\wedge\neg$ (y=0 $\vee$ t=1) $\rightarrow$	(b):=(wait);	
b=CR $\rightarrow$	(x,b):=(0, NCR);	

例子-互斥

T:		Θ:
a=NCR→	(y,t,a):=(1,1,wait);	
a=wait∧(x=0 ∨ t=0) →	(a):=(CR);	x=0 ∧y=0∧
a=wait ∧¬(x=0∨t=0) →	(a):=(wait);	a=NCR ∧b=NCR
a=CR →	(y,a):=(0, NCR);	
b=NCR →	(x,t,b):=(1,0, wait);	
b=wait ∧(y=0 ∨ t=1) →	(b):=(CR);	
b=wait ∧¬ (y=0 ∨ t=1) →	(b):=(wait);	
b=CR →	(x,b):=(0, NCR);	

B=(F,P)

F={0,1,NCR,wait,CR}

P={=}

V={a,b,x,y,t}

I=(N,I_0)

I_0(0) = **0**

I_0(1) = **1**

I_0(=) = **=**

I_0(NCR) = **0**

I_0(wait) = **1**

I_0(CR) = **2**

状态集(状态空间)

状态集:

$V=\{a,b,x,y,t\}$

$\Sigma=\{ (a,b,x,y,t) \mid a,b \in\{0,1,2\}, x,y,t \in\{0,1\} \}$

迁移关系(动态生成)

V={a,b,x,y,t}

NCR,NCR,0,0,0 → ?

(0,0,0,0,0) → ?

a=NCR→ (y,t,a):=(1,1,wait) 可执行?

(0,0,0,0,0) |= a=NCR ?

σ'= (I(wait)(σ),0,0,I(1)(σ),I(1)(σ)) = (1,0,0,1,1)

所以: (0,0,0,0,0) → (1,0,0,1,1)

迁移关系

V={a,b,x,y,t}

NCR,NCR,0,0,0 → ?

(0,0,0,0,0) → ?

b=NCR → (x,t,b):=(1,0, wait) 可执行?

(0,0,0,0,0) |= b=NCR ?

σ'= (0,I(wait)(σ),I(1)(σ),0,I(0)(σ)) = (0,1,1,0,0)

所以: (0,0,0,0,0) → (0,1,1,0,0)

初始状态集

$\Theta \equiv\ x=0 \wedge y=0 \wedge a=NCR \wedge b=NCR$

初始状态集

$= \{ \sigma \mid \sigma \models \Theta \}$

$= \{ (0,0,0,0,t) \mid t \in \{0,1\} \}$

$= \{ (0,0,0,0,0), (0,0,0,0,1) \}$

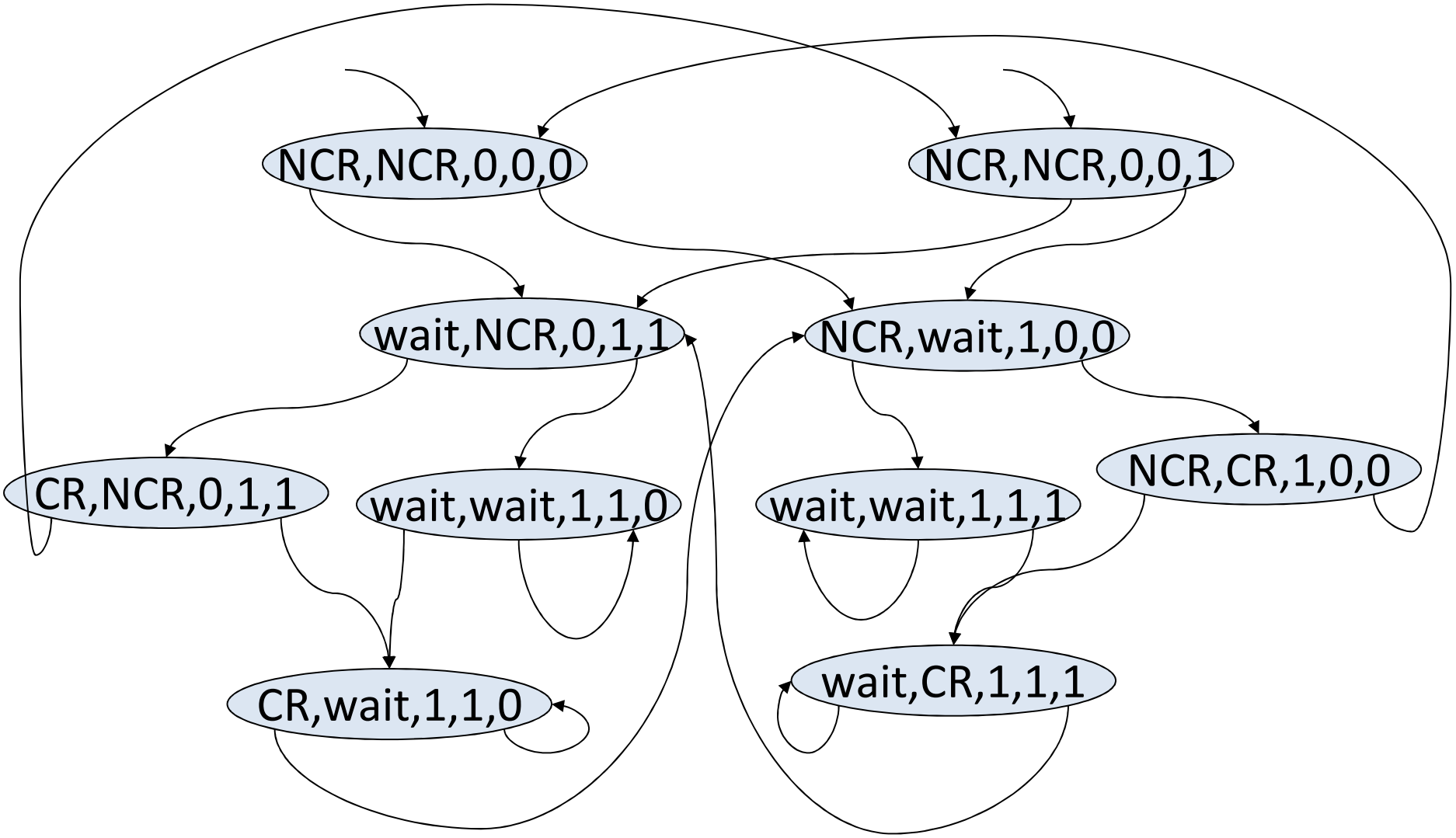
标号

BP={ a=NCR,a=wait,a=CR,b=NCR,b=wait,b=CR,
x=0,x=1,y=0,y=1,t=0,t=1 }

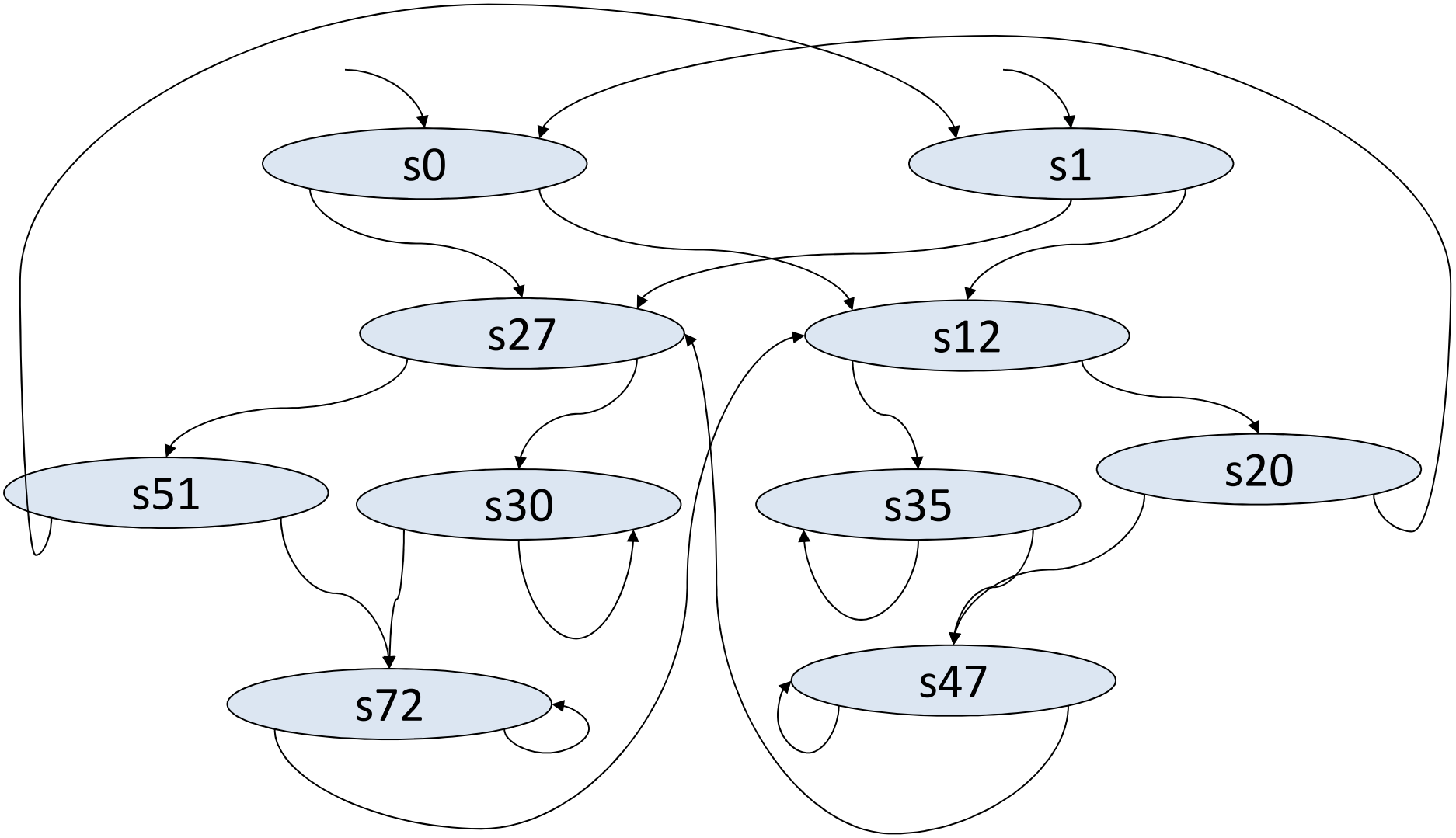
AP={ a=NCR,a=wait,a=CR,b=NCR,b=wait,b=CR,
x=0,x=1,y=0,y=1,t=0,t=1 }

L(NCR,NCR,0,0,0)={a=NCR,b=NCR,x=0,y=0,t=0}
L(NCR,NCR,0,0,1)={a=NCR,b=NCR,x=0,y=0,t=1}
…

Kripke 结构


NCR,NCR,0,0,0
NCR,NCR,0,0,1
wait,NCR,0,1,1
NCR,wait,1,0,0
CR,NCR,0,1,1
wait,wait,1,1,0
wait,wait,1,1,1
NCR,CR,1,0,0
CR,wait,1,1,0
wait,CR,1,1,1

Kripke 结构：(a,b,x,y,t) = 24a+8b+4x+2y+t



标号

BP={ a=NCR,a=wait,a=CR,b=NCR,b=wait,b=CR,
x=0,x=1,y=0,y=1,t=0,t=1 }

AP={ p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12 }

L(s0)={p1,p4,p7,p9,p11}

L(s1)={p1,p4,p7,p9,p12}

…

结论

ζ (p1) = (a=NCR)

ζ (p2) = (a=wait)

…

根据前述构造方法， M 与 K(M) 为 ζ 计算等价。

¬(a=CR∧b=CR)是M的安全性质当且仅当¬(p4∧p6)是K(M)的安全性质

(a=CR∨b=CR)是M的必达性质当且仅当(p4∨p6)是K(M)的必达性质

(II) 公平卫式迁移模型

- 定义
- 性质
- 公平卫式迁移模型与公平标号Kripke模型的关系

(II.a)公平卫式迁移模型

卫式迁移模型的扩充

给定B=(F,P)和变量集合V。

公平卫式迁移模型是一个三元组(T,Θ,Φ)
其中(T,Θ)是(B,V)上一个卫式迁移模型,
$\Phi = \{\varphi_1, \ldots, \varphi_n\}$其中$\varphi_i \in QFF_B$且在$\varphi_i$中出现的变量在 V中

例子-互斥

T:	$a=NCR \rightarrow$	$(y,t,a):=(1,1,wait);$
	$a=wait \wedge (x=0 \vee t=0) \rightarrow$	$(a):=(CR);$
	$a=wait \wedge \neg(x=0 \vee t=0) \rightarrow$	$(a):=(wait);$
	$a=CR \rightarrow$	$(y,a):=(0, NCR);$
	$b=NCR \rightarrow$	$(x,t,b):=(1,0, wait);$
	$b=wait \wedge (y=0 \vee t=1) \rightarrow$	$(b):=(CR);$
	$b=wait \wedge \neg (y=0 \vee t=1) \rightarrow$	$(b):=(wait);$
	$b=CR \rightarrow$	$(x,b):=(0, NCR);$
Θ:	$x=0 \wedge y=0 \wedge a=NCR \wedge b=NCR$	
F:	$\{ \neg(a=NCR), \neg(a=wait \wedge (x=0 \vee t=0)), \neg(a=CR),$	
	$\neg(b=NCR), \neg(b=wait \wedge (y=0 \vee t=1)), \neg(b=CR) \}$	

模型

状态集合

状态迁移关系

公平计算、公平路径、公平状态

计算

模型M=(T,Θ)的一条路径是
状态集Σ上的一个无穷序列$\sigma_0\sigma_1$ .....
满足对任意i有$\sigma_i \rightarrow \sigma_{i+1}$

模型M=(T,Θ)的一个计算是
状态集Σ上的一个无穷序列$\sigma_0\sigma_1$ .....
满足$\sigma_0 |=_l \Theta$且对任意i有$\sigma_i \rightarrow \sigma_{i+1}$

公平计算

模型M=(T,Θ,Φ)的一条公平路径是
(T,Θ)的一条路径$\sigma_0\sigma_1$ .....
满足对任意i 存在无限多个k, $\sigma_k|=_l \varphi_i$

模型M=(T,Θ,Φ)的一个公平计算是
(T,Θ)的一个计算$\sigma_0\sigma_1$ .....
满足对任意i 存在无限多个k, $\sigma_k|=_l \varphi_i$

模型M的公平计算集合记为[[M]]

公平状态与公平可达状态集合

s是模型$M=(T,\Theta,\Phi)$的一个公平状态，
当且仅当存在从s出发的公平路径。

模型M的公平可达状态集合记为$rh_F(M)$

(II.b) 正确性相关性质

φ是模型M的公平安全性质当且仅当$\forall \pi \in [[M]].\forall k.\pi_k \models_l \varphi$。

φ是模型M的公平必达性质当且仅当$\forall \pi \in [[M]].\exists k.\pi_k \models_l \varphi$。

模型性质的分析

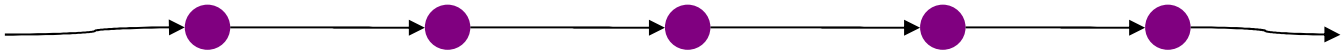
基于规则和推理

基于状态搜索算法(针对有穷状态系统)

(II.c) 与 公平标号Kripke结构的等价性

x=1　　x=1

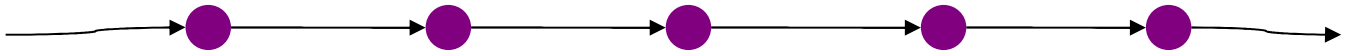
y=1　　y=2



p: x=1

q: y=2

{p}　　{p,q}



卫式迁移系统–公平标号Kripke结构: 等价

给定模型M=(T,Θ,Φ)和公平标号Kripke结构K=(S,R,I,L,F)

设AP={p1,…,pn}, BP={Ψ1,…, Ψn } $\subseteq$ QFF

设 ζ ： AP→BP 为一一对应关系

$\forall \pi \in [[K]], \exists \pi' \in [[M]], \forall k.(\pi_k \models p \leftrightarrow \pi'_k \models \zeta(p))$ 且

$\forall \pi \in [[M]], \exists \pi' \in [[K]], \forall k.(\pi_k \models \Psi \leftrightarrow \pi'_k \models \zeta^{-1}(\Psi))$

则称M与K为 ζ 计算等价。

卫式迁移系统–公平标号Kripke结构: 等价

定理：

设$\phi \in L(BP)$。
若M与K为ζ计算等价，
则
ϕ是M的公平安全性质当且仅当$\zeta^{-1}(\phi)$是K的公平安全性质，
ϕ是M的公平必达性质当且仅当$\zeta^{-1}(\phi)$是K的公平必达性质

卫式迁移系统–公平标号Kripke结构: 构造

给定模型$M=(T,\Theta,\Phi)$：

给定$BP=\{\Psi_1,\ldots,\Psi_n\}$

定义$AP=\{p_1,\ldots,p_n\}$ 并定义 ζ：$AP \rightarrow BP$ 为 $\zeta(p_i)=\Psi_i$

定义公平标号Kripke结构K(M)如下：

状态集：Σ

迁移关系：$\rightarrow$

初始状态集：$\{\sigma \mid \sigma|=_l\Theta\}$

标号函数：$p_i \in L(\sigma)$ 当且仅当$\sigma|=_l\Psi_i$

公平约束：$\zeta^{-1}(\Phi)$

卫式迁移系统–公平标号Kripke结构: 构造

给定模型M=(T,Θ,Φ)：

(B,V)

B=(F,P); V={v1,…,vn}

σ: V → D

σ: (V1 → D1) ∪…∪ (Vn → Dn)　　按不同变量类型

卫式迁移系统–公平标号Kripke结构:构造

状态集:

用(a1,…,an)代表σ: σ(v1)=a1, …, σ(vn)=an

Σ= {(a1,…,an) | ai $\in$D }

Σ= {(a1,…,an) | a1 $\in$D1, …, an $\in$Dn}

卫式迁移系统–公平标号Kripke结构:构造

迁移关系

$((a1,\ldots,an),(a1',\ldots,an')) \in \rightarrow$

$\Leftrightarrow$

$\sigma \rightarrow \sigma'$

$\sigma(v1)=a1, \ldots, \sigma(vn)=an$

$\sigma'(v1)=a1', \ldots, \sigma'(vn)=an'$

卫式迁移系统–公平标号Kripke结构:构造

初始状态集:

$\{(a1,\ldots,an) \mid \sigma(v1)=a1, \ldots, \sigma(vn)=an, \sigma \models_l \Theta\}$

标号函数： $p_i \in L(\sigma)$ 当且仅当$\sigma \models_l \Psi_i$

公平约束： $\zeta^{-1}(\Phi)$

卫式迁移系统–公平标号Kripke结构: 构造

定理：

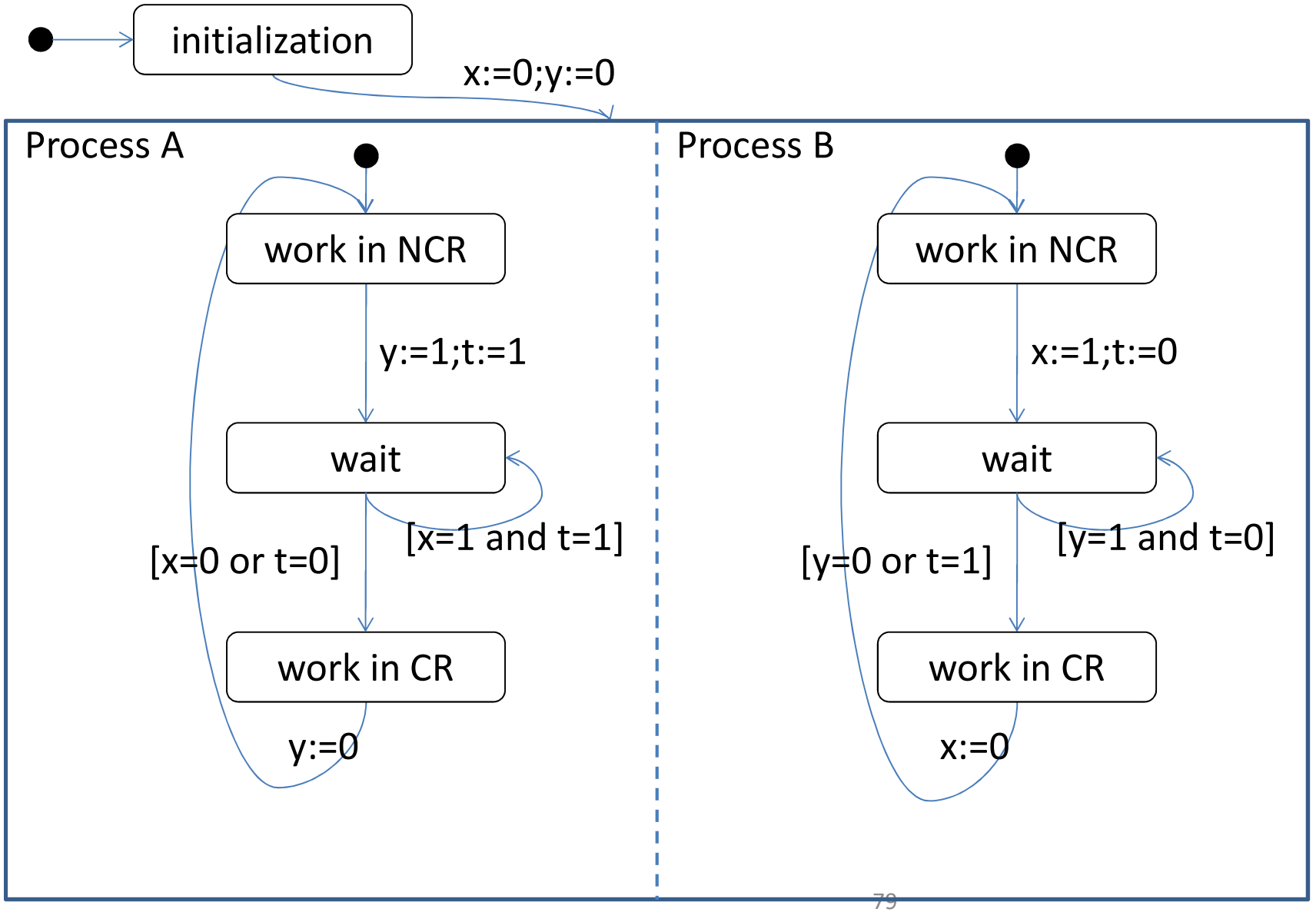
M与K(M)为ς计算等价.

推论：

ϕ是M的公平安全性质当且仅当ς^{-1}(ϕ)是K(M)的公平安全性质

ϕ是M的公平必达性质当且仅当ς^{-1}(ϕ)是K(M)的公平必达性质

例2-互斥：状态图(State Diagram)



例子-互斥

T:	a=NCR$\rightarrow$	(y,t,a):=(1,1,wait);
	a=wait$\wedge$(x=0 $\vee$ t=0) $\rightarrow$	(a):=(CR);
	a=wait $\wedge\neg$(x=0$\vee$t=0) $\rightarrow$	(a):=(wait);
	a=CR $\rightarrow$	(y,a):=(0, NCR);
	b=NCR $\rightarrow$	(x,t,b):=(1,0, wait);
	b=wait $\wedge$(y=0 $\vee$ t=1) $\rightarrow$	(b):=(CR);
	b=wait $\wedge\neg$ (y=0 $\vee$ t=1) $\rightarrow$	(b):=(wait);
	b=CR $\rightarrow$	(x,b):=(0, NCR);
Θ:	x=0 $\wedge$ y=0 $\wedge$ a=NCR $\wedge$ b=NCR	
F:	{ $\neg$(a=NCR), $\neg$(a=wait$\wedge$(x=0 $\vee$ t=0)), $\neg$(a=CR), $\neg$(b=NCR), $\neg$(b=wait $\wedge$(y=0 $\vee$ t=1)), $\neg$(b=CR) }	

标号

BP={ a=NCR,a=wait,a=CR,b=NCR,b=wait,b=CR,
x=0,x=1,y=0,y=1,t=0,t=1 }

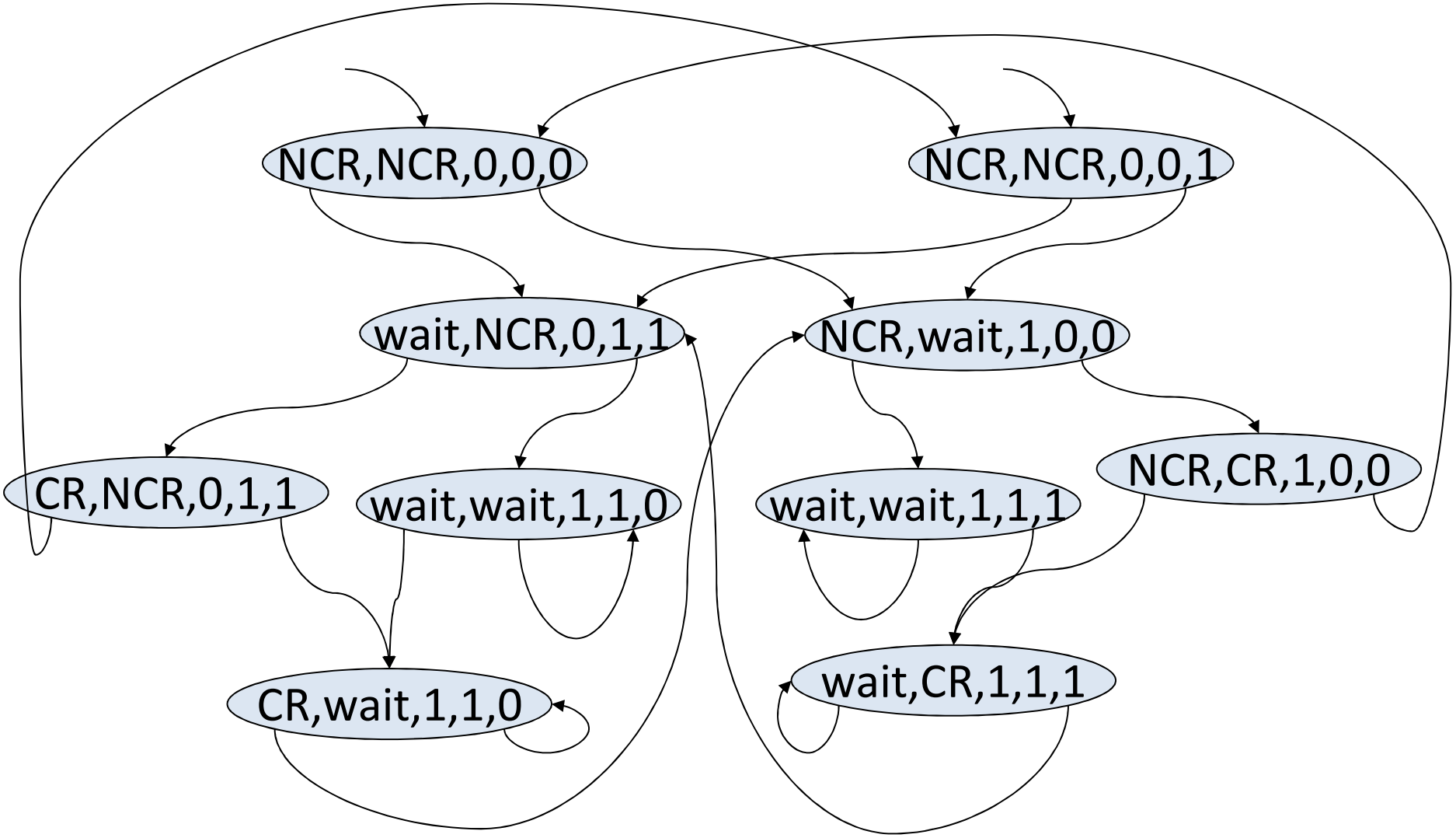
AP={ a=NCR,a=wait,a=CR,b=NCR,b=wait,b=CR,
x=0,x=1,y=0,y=1,t=0,t=1 }

L(NCR,NCR,0,0,0)={a=NCR,b=NCR,x=0,y=0,t=0}
L(NCR,NCR,0,0,1)={a=NCR,b=NCR,x=0,y=0,t=1}
…

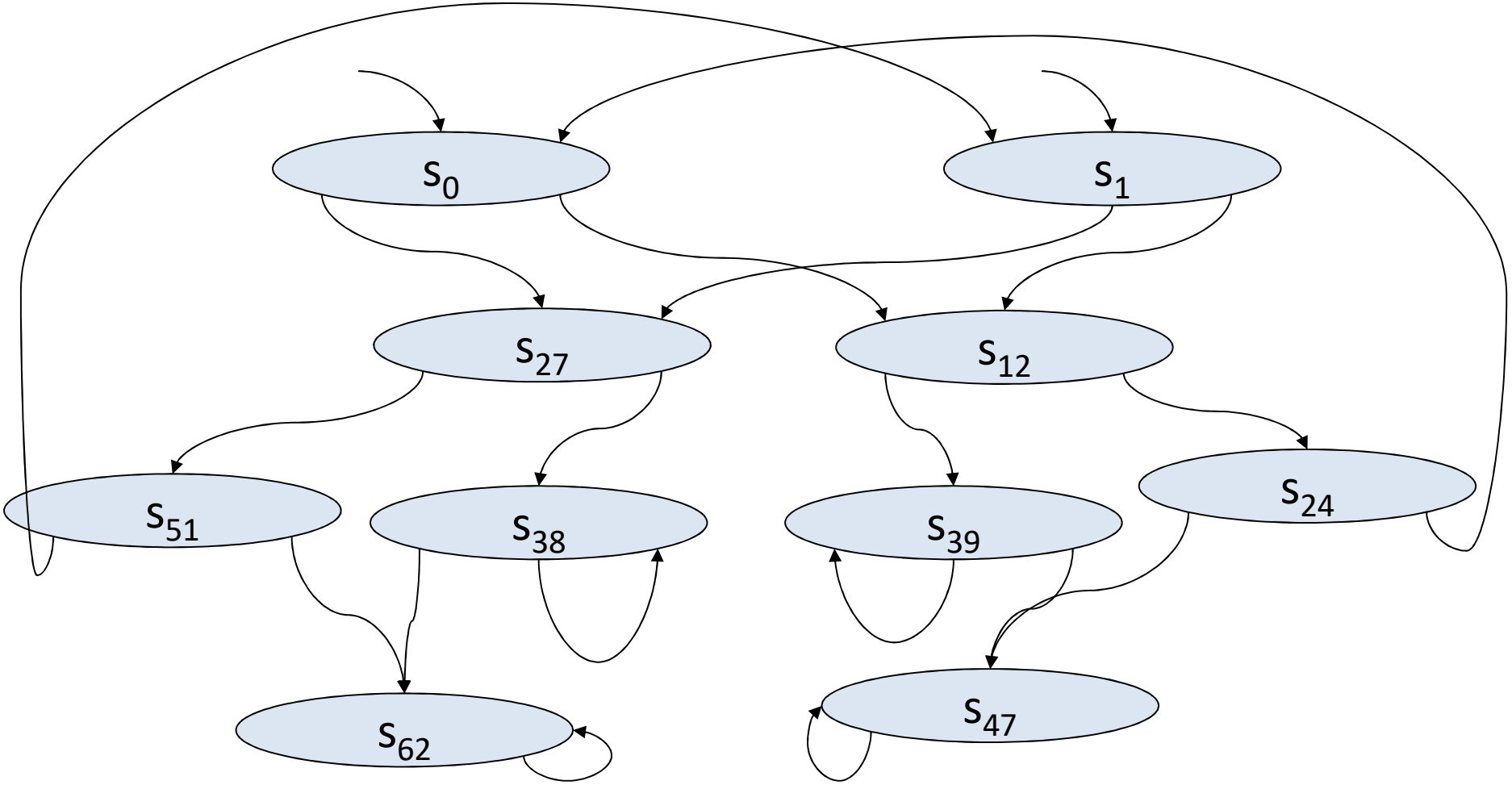
公平约束

$$
\begin{aligned}
F=\{ & \\
& \neg(a=NCR), \\
& \neg((x=0 \vee t=0) \wedge a=wait), \\
& \neg(a=CR), \\
& \neg(b=NCR), \\
& \neg((y=0 \vee t=1) \wedge b=wait), \\
& \neg(b=CR) \\
\} &
\end{aligned}
$$

Kripke 结构


NCR,NCR,0,0,0
NCR,NCR,0,0,1
wait,NCR,0,1,1
NCR,wait,1,0,0
CR,NCR,0,1,1
wait,wait,1,1,0
wait,wait,1,1,1
NCR,CR,1,0,0
CR,wait,1,1,0
wait,CR,1,1,1

Kripke 结构：(a,b,x,y,t) = 24a+8b+4x+2y+t



标号

BP={ a=NCR,a=wait,a=CR,b=NCR,b=wait,b=CR,
x=0,x=1,y=0,y=1,t=0,t=1 }

AP={ p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12 }

L(s0)={p1,p4,p7,p9,p11}

L(s1)={p1,p4,p7,p9,p12}

…

公平约束

F={

¬(p1),

¬((p7∨p11)∧p2),

¬(p3),

¬(p4),

¬((p9∨p12)∧p5),

¬(p6)

}

结论

ζ (p1) = (a=NCR)

ζ (p2) = (a=wait)

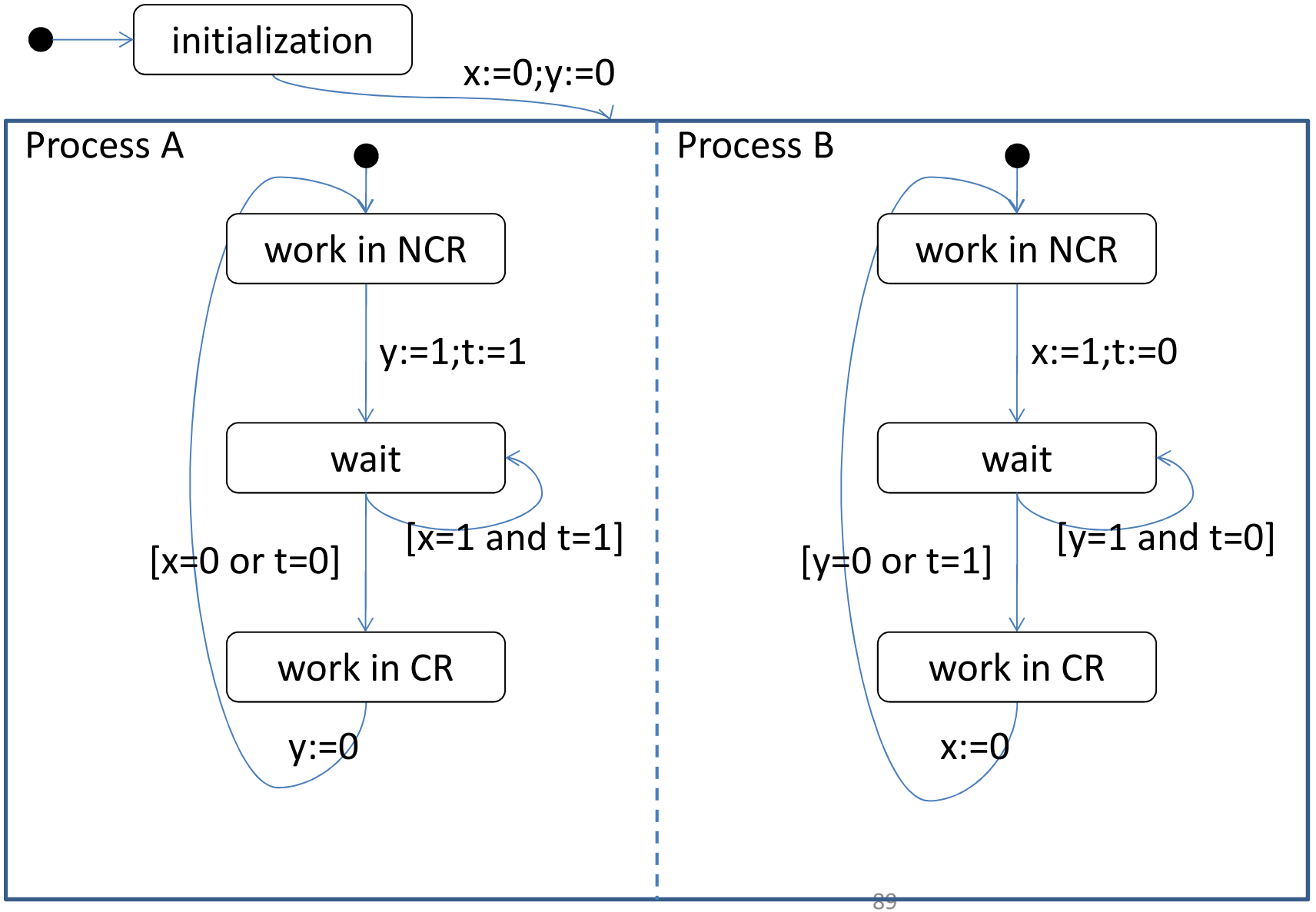
…

根据前述构造方法， M 与 K(M) 为 ζ 计算等价。

x2-me

建模与验证实例

例子-互斥：状态图(使用一个函数符号)



互斥

a=NCR→	(y,t,a):=(1,1,wait);	Θ:
a=wait∧(x=0 ∨ t=0) →	(a):=(CR);	
a=wait ∧¬(x=0∨t=0) →	(a):=(wait);	x=0 ∧y=0∧
a=CR →	(y,a):=(0, NCR);	a=NCR ∧b=NCR
b=NCR →	(x,t,b):=(1,0, wait);	
b=wait ∧(y=0 ∨ t=1) →	(b):=(CR);	
b=wait ∧¬ (y=0 ∨ t=1) →	(b):=(wait);	
b=CR →	(x,b):=(0, NCR);	

必达性质：　　a=CR∨b=CR　　是否成立?

安全性质：　　¬(a=CR∧b=CR)　　是否成立?

互斥(改写)

a=ncr:	(y,t,a):=(1,1,wait);	Θ:
a=wait & (x=0 | t=0):	(a):=(cr);	
a=wait & !(x=0 | t=0):	(a):=(wait);	x=0;
a=cr:	(y,a):=(0,ncr);	y=0;
b=ncr:	(x,t,b):=(1,0,wait);	a=ncr;
b=wait & (y=0 | t=1):	(b):=(cr);	b=ncr;
b=wait & !(y=0 | t=1):	(b):=(wait);	
b=cr:	(x,b):=(0,ncr);	

必达性质：　　$a=cr \vee b=cr$　　是否成立?

安全性质：　　$\neg(a=cr \wedge b=cr)$　是否成立?

互斥

B=(F,P) F={ncr,wait,cr,0,1} P={=} V={a,b,x,y,t}	I=(D,I_0) D={0,1,2,3,……}=N I_0(0)=0 I_0(1)=1 I_0(ncr)=0 I_0(wait)=1 I_0(cr)=2 I_0(=)= =

适用于算法验证的实际建模(1)

有穷状态模型

例子-互斥(变量的实际取值范围)

B=(F,P) F={ncr,wait,cr,0,1} P={=} V={a,b,x,y,t} D0={ncr,wait,cr} D1={0,1}	x:0..1; y:0..1; t:0..1; a:{ncr,wait,cr}; b:{ncr,wait,cr};

例子-互斥(VVM模型)

```
VVM lec4mutex01.vvm
VAR
    x: 0..1;
    y: 0..1;
    t: 0..1;
    a: {ncr,wait,cr};
    b: {ncr,wait,cr};
INIT
    x=0;
    y=0;
    a=ncr;
    b=ncr;
```

```
TRANS
    a=ncr:                  (y,t,a):=(1,1,wait);
    a=wait&(x=0|t=0):  (a):=(cr);
    a=wait&!(x=0|t=0): (a):=(wait);
    a=cr:                   (y,a):=(0,ncr);
    b=ncr:                  (x,t,b):=(1,0,wait);
    b=wait&(y=0|t=1):  (b):=(cr);
    b=wait&!(y=0|t=1): (b):=(wait);
    b=cr:                   (x,b):=(0,ncr);
```

例子-互斥(VVM模型+性质)

```
VVM lec4mutex01.vvm
VAR
    x: 0..1;
    y: 0..1;
    t: 0..1;
    a: {ncr,wait,cr};
    b: {ncr,wait,cr};
INIT
    x=0;
    y=0;
    a=ncr;
    b=ncr;
TRANS
    a=ncr:                  (y,t,a):=(1,1,wait);
    a=wait&(x=0|t=0):  (a):=(cr);
    a=wait&!(x=0|t=0): (a):=(wait);
    a=cr:                   (y,a):=(0,ncr);
    b=ncr:                  (x,t,b):=(1,0,wait);
    b=wait&(y=0|t=1):  (b):=(cr);
    b=wait&!(y=0|t=1): (b):=(wait);
    b=cr:                   (x,b):=(0,ncr);
SPEC
    AG(!(a=cr&b=cr));
    AF((a=cr)|(b=cr));
```

验证

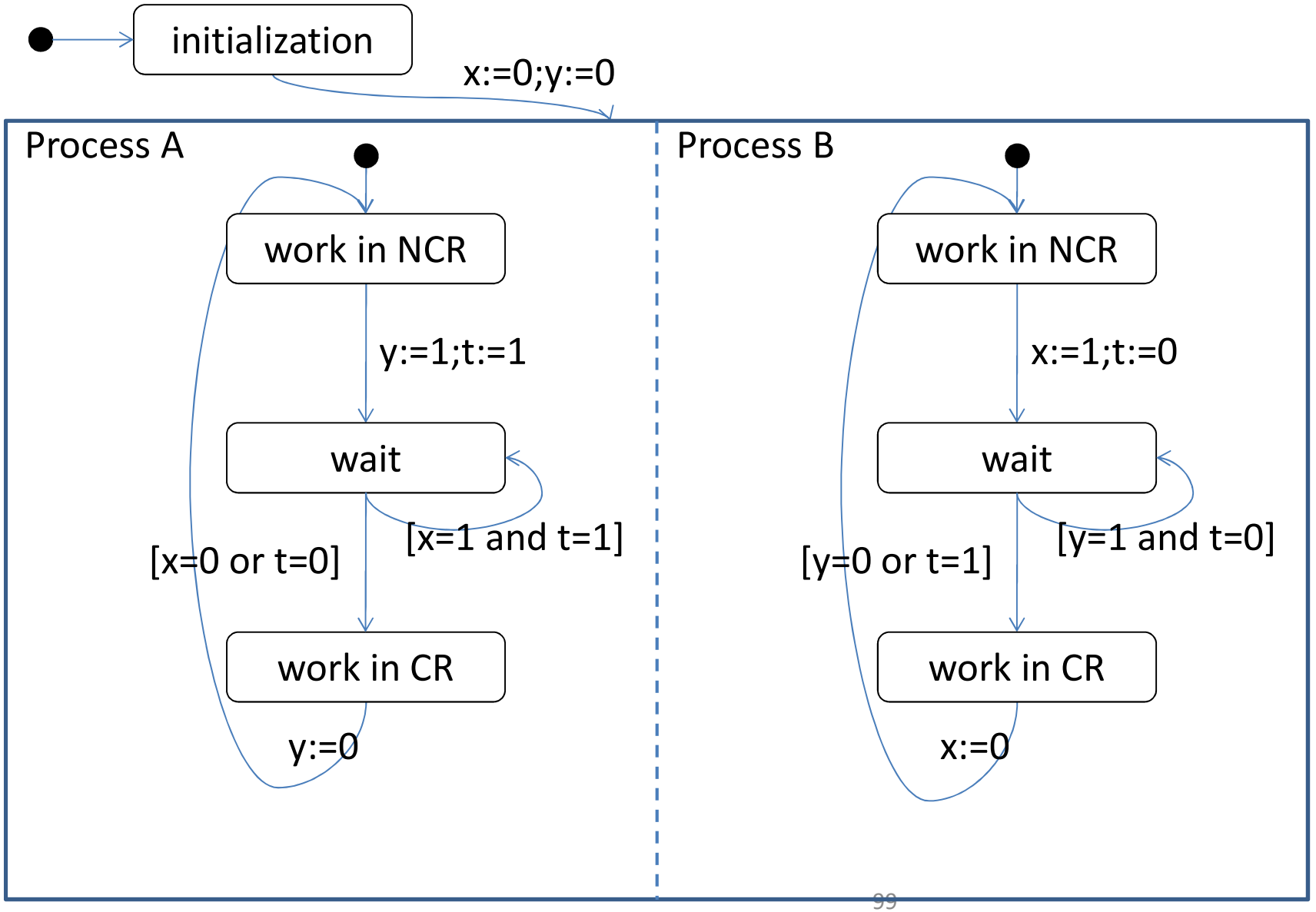
```
verds -ck 1 lec4mutex01.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01.vvm
INFO:       int=i0
PROPERTY:   A G ! ((p0.a = 2 )& (p1.b = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
check:   5
----------
CONCLUSION: TRUE
```

```
verds -ck 2 lec4mutex01.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01.vvm
INFO:       int=i0
PROPERTY:   A F ((p0.a = 2 )| (p1.b = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
The property is false, preparing lec4mutex01.cex ...
CONCLUSION: FALSE
```

实际建模(2)

模块化

例子-互斥：状态图()


initialization
x:=0;y:=0
Process A
work in NCR
y:=1;t:=1
wait
[x=1 and t=1]
[x=0 or t=0]
work in CR
y:=0
Process B
work in NCR
x:=1;t:=0
wait
[y=1 and t=0]
[y=0 or t=1]
work in CR
x:=0

例子-互斥(模块化)

a=ncr: a=wait & (x=0 | t=0): a=wait & !(x=0 | t=0): a=cr:	(y,t,a):=(1,1,wait); (a):=(cr); (a):=(wait); (y,a):=(0,ncr);	a=ncr;

b=ncr: b=wait & (y=0 | t=1): b=wait & !(y=0 | t=1): b=cr:	(x,t,b):=(1,0,wait); (b):=(cr); (b):=(wait); (x,b):=(0,ncr);	b=ncr;

x=0; y=0;
t

例2-互斥(VVM模型)

```
VVM lec4mutex01m.vvm
VAR
      x: 0..1;
      y: 0..1;
      t: 0..1;
INIT
      x=0;
      y=0;
PROC
      p0: p0m();
      p1: p1m();

SPEC
      AG(!(p0.a=cr&p1.b=cr));
      AF((p0.a=cr)|(p1.b=cr));
```

```
MODULE  p0m()
VAR a: {ncr,wait,cr};
INIT a=ncr;
TRANS
      a=ncr:                    (y,t,a):=(1,1,wait);
      a=wait&(x=0|t=0):  (a):=(cr);
      a=wait&!(x=0|t=0): (a):=(wait);
      a=cr:                     (y,a):=(0,ncr);
```

```
MODULE  p1m()
VAR b: {ncr,wait,cr};
INIT b=ncr;
TRANS
      b=ncr:                    (x,t,b):=(1,0,wait);
      b=wait&(y=0|t=1):  (b):=(cr);
      b=wait&!(y=0|t=1): (b):=(wait);
      b=cr:                     (x,b):=(0,ncr);
```

验证

```
verds -ck 1 lec4mutex01m.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01m.vvm
INFO:       int=i0
PROPERTY:   A G ! ((p0.a = 2 )& (p1.b = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
check:   5
----------
CONCLUSION: TRUE
```

```
verds -ck 2 lec4mutex01m.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01m.vvm
INFO:       int=i0
PROPERTY:   A F ((p0.a = 2 )| (p1.b = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
The property is false, preparing lec4mutex01m.cex ...
CONCLUSION: FALSE
```

实际建模(3)

参数化

例子-互斥(模块化)

a=ncr: a=wait & (x=0 | t=0): a=wait & !(x=0 | t=0): a=cr:	(y,t,a):=(1,1,wait); (a):=(cr); (a):=(wait); (y,a):=(0,ncr);	a=ncr;

b=ncr: b=wait & (y=0 | t=1): b=wait & !(y=0 | t=1): b=cr:	(x,t,b):=(1,0,wait); (b):=(cr); (b):=(wait); (x,b):=(0,ncr);	b=ncr;

x=0; y=0;
t

例子-互斥(模块化)

a=ncr: a=wait & (x=0 | t=0): a=wait & !(x=0 | t=0): a=cr:	(y,t,a):=(1,1,wait); (a):=(cr); (a):=(wait); (y,a):=(0,ncr);	a=ncr;

a=ncr: a=wait & (y=0 | t=1): a=wait & !(y=0 | t=1): a=cr:	(x,t,a):=(1,0,wait); (a):=(cr); (a):=(wait); (x,a):=(0,ncr);	a=ncr;

x=0; y=0;
t

例子-互斥(参数化)

a=ncr: a=wait & (x=0 | t=k): a=wait & !(x=0 | t=k): a=cr:	(y,t,a):=(1,1-k,wait); (a):=(cr); (a):=(wait); (y,a):=(0,ncr);	a=ncr;

a=ncr: a=wait & (y=0 | t=k): a=wait & !(y=0 | t=k): a=cr:	(x,t,a):=(1,1-k,wait); (a):=(cr); (a):=(wait); (x,a):=(0,ncr);	a=ncr;

k=0,1; x与y互换

x=0; y=0;
t

例2-互斥(VVM模型)

```
VVM lec4mutex01pm.vvm
VAR
      x: 0..1;
      y: 0..1;
      t: 0..1;
INIT
      x=0;
      y=0;
PROC
      p0: p0m(x,y,0);
      p1: p0m(y,x,1);

SPEC
      AG(!(p0.a=cr&p1.a=cr));
      AF((p0.a=cr)|(p1.a=cr));
```

```
MODULE  p0m(px,py,k)
VAR a: {ncr,wait,cr};
INIT a=ncr;
TRANS
      a=ncr:                    (py,t,a):=(1,1-k,wait);
      a=wait&(px=0|t=k):  (a):=(cr);
      a=wait&!(px=0|t=k): (a):=(wait);
      a=cr:                     (py,a):=(0,ncr);
```

注意:

参数与变量不重名(顺序替换)

验证

```
verds -ck 1 lec4mutex01pm.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01pm.vvm
INFO:       int=i0
PROPERTY:   A G ! ((p0.a = 2 )& (p1.a = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
check:   5
----------
CONCLUSION: TRUE
```

```
verds -ck 2 lec4mutex01pm.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01pm.vvm
INFO:       int=i0
PROPERTY:   A F ((p0.a = 2 )| (p1.a = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
The property is false, preparing lec4mutex01pm.cex ...
CONCLUSION: FALSE
```

公平约束

例子-互斥

T:	a=NCR$\rightarrow$	(y,t,a):=(1,1,wait);
	a=wait$\wedge$(x=0 $\vee$ t=0) $\rightarrow$	(a):=(CR);
	a=wait $\wedge\neg$(x=0$\vee$t=0) $\rightarrow$	(a):=(wait);
	a=CR $\rightarrow$	(y,a):=(0, NCR);
	b=NCR $\rightarrow$	(x,t,b):=(1,0, wait);
	b=wait $\wedge$(y=0 $\vee$ t=1) $\rightarrow$	(b):=(CR);
	b=wait $\wedge\neg$ (y=0 $\vee$ t=1) $\rightarrow$	(b):=(wait);
	b=CR $\rightarrow$	(x,b):=(0, NCR);
Θ:	x=0 $\wedge$ y=0 $\wedge$ a=NCR $\wedge$ b=NCR	
F:	{ $\neg$(a=NCR), $\neg$(a=wait$\wedge$(x=0 $\vee$ t=0)), $\neg$(a=CR), $\neg$(b=NCR), $\neg$(b=wait $\wedge$(y=0 $\vee$ t=1)), $\neg$(b=CR) }	

公平约束

!(a=ncr);
!((x=0|t=0)&(a=wait));
!(a=cr);
!(b=ncr);
!((y=0|t=1)&(b=wait));
!(b=cr);

例2-互斥(VVM模型)

```
VVM lec4mutex01pmfair.vvm
VAR
      x: 0..1;
      y: 0..1;
      t: 0..1;
INIT
      x=0;
      y=0;
PROC
      p0: p0m(x,y,0);
      p1: p0m(y,x,1);

SPEC
      AG(!(p0.a=cr&p1.a=cr));
      AF((p0.a=cr)|(p1.a=cr));
```

```
MODULE  p0m(px,py,k)
VAR a: {ncr,wait,cr};
INIT a=ncr;
TRANS
      a=ncr:                    (py,t,a):=(1,1-k,wait);
      a=wait&(px=0|t=k):  (a):=(cr);
      a=wait&!(px=0|t=k): (a):=(wait);
      a=cr:                     (py,a):=(0,ncr);

FAIRNESS
      !(a=ncr);
      !((px=0|t=k)&(a=wait));
      !(a=cr);
```

注意:

参数与变量不重名(顺序替换)

验证

```
verds -ck 1 lec4mutex01pmfair.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01pmfair.vvm
INFO:       int=i0
PROPERTY:   A G ! ((p0.a = 2 )& (p1.a = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
check:   4
----------
…
…
check:   19
----------
CONCLUSION: TRUE
```

```
verds -ck 2 lec4mutex01pmfair.vvm
VERSION:    verds 1.49 - JAN 2017
FILE:       lec4mutex01pmfair.vvm
INFO:       int=i0
PROPERTY:   A F ((p0.a = 2 )| (p1.a = 2 ))
check:   0
----------
check:   1
----------
check:   2
----------
check:   3
----------
…
…
check:   99
----------
CONCLUSION: TRUE
```

(III)顺序流程图模型

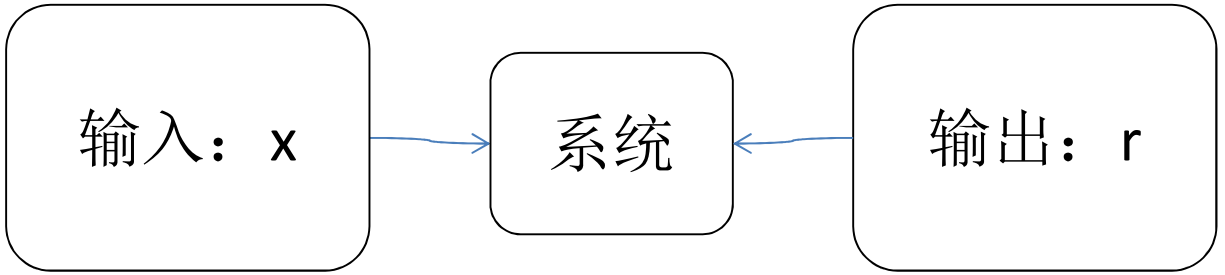
- Motivation

Motivation

Sequential computations:

A restriction of the first order guarded transitions for sequential computations

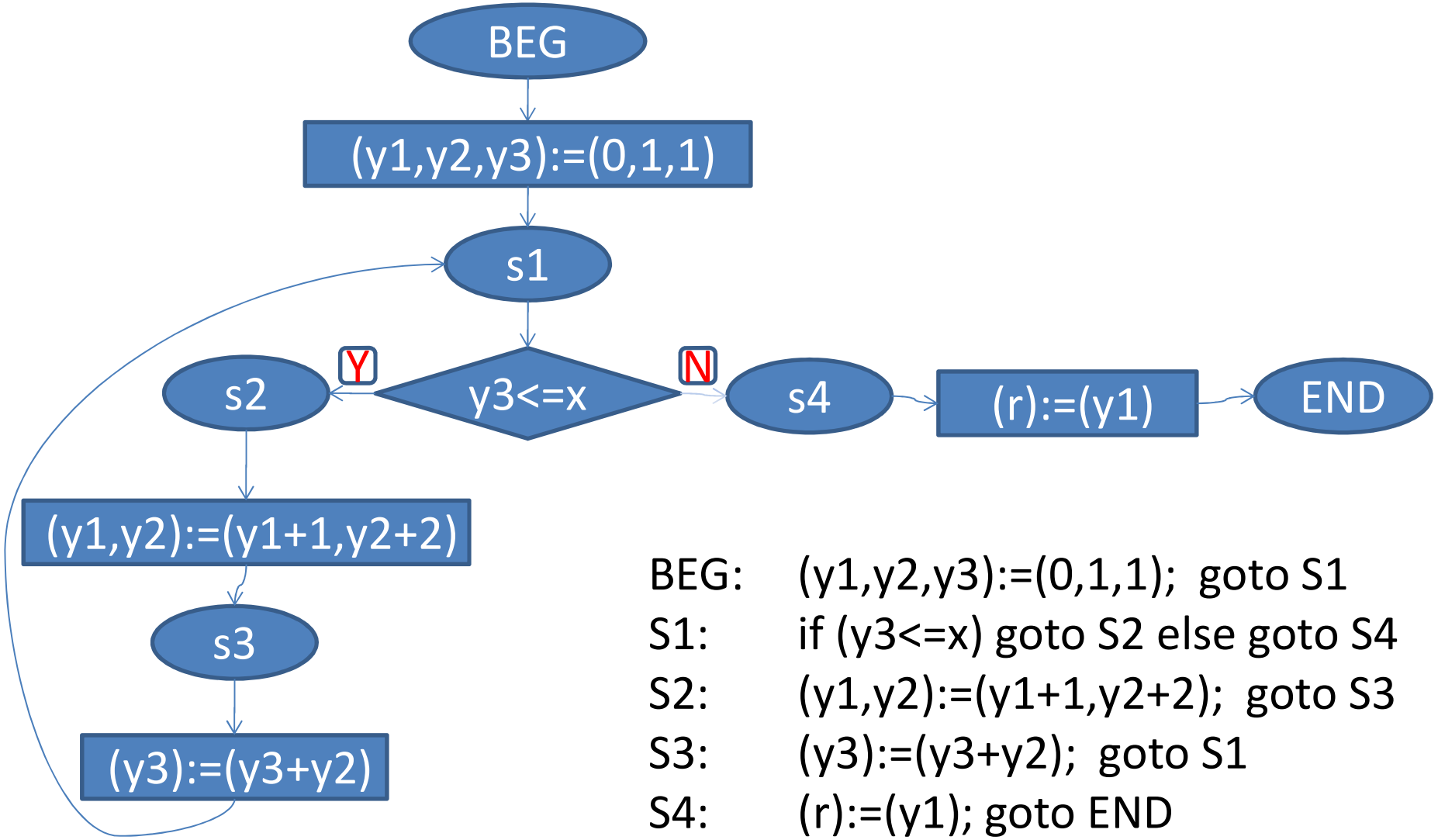
例1 -整树平方根：需求



输入：　　x;
初始条件：x>=0;

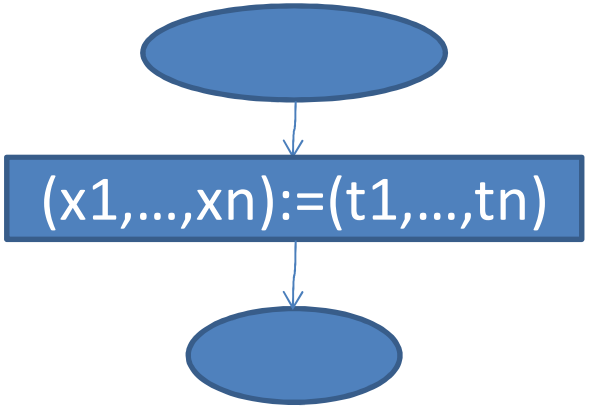
输出：　　r;
输出满足：x>=r*r; x<(r+1)*(r+1);

例1 -整树平方根：设计



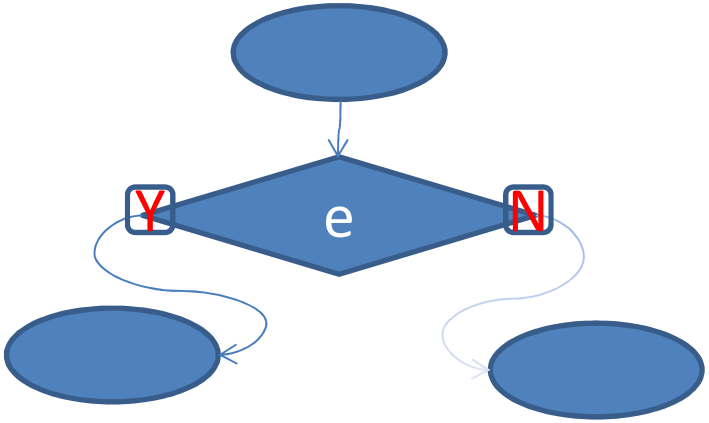
BEG: (y1,y2,y3):=(0,1,1); goto S1
S1: if (y3<=x) goto S2 else goto S4
S2: (y1,y2):=(y1+1,y2+2); goto S3
S3: (y3):=(y3+y2); goto S1
S4: (r):=(y1); goto END

流程图模型基本组成部分



L1: (x1,…,xn):=(t1,…,tn); goto L2

L1: if (e) goto L2 else goto L3



顺序流程图模型

- 定义
- 性质
- 顺序流程图模型与卫式迁移系统的关系

(III.a)顺序流程图模型

一阶逻辑的扩充

主要有两种语句（指令）：赋值语句，条件语句。

在一阶逻辑的基础上，增加以下两个集合的符号：
辅助符号集AX：{ :=, :, ;, goto, if, else }
标号符号集LB：{ BEG, END, ... }

指令

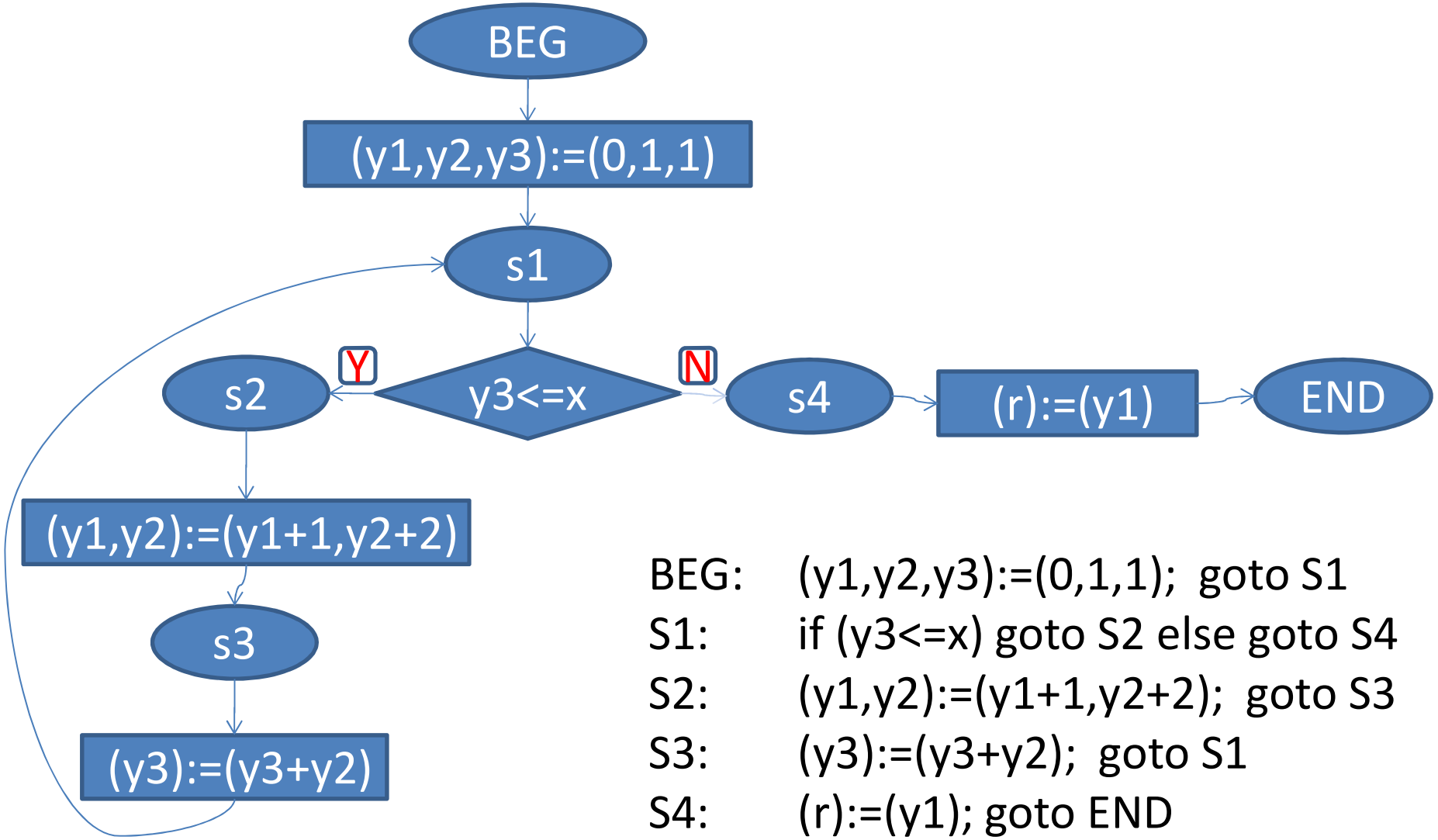
给定B=(F,P)和变量集合V。
定义(B,V)上的流程图模型的两种指令如下：

L1: (x1,...,xn):=(t1,...,tn) ; goto L2
L1: if (e) goto L2 else goto L3

其中　　L1,L2,L3∈LB为标号，L1 ≠ END
t1,...,tn ∈T_B 且在t1,...,tn中出现的变量在 V中
e ∈ QFF_B且在e中出现的变量在 V中
x1,...,xn ∈V且这些变量不重复

冒号左边的标号称为　被定义标号
冒号右边的句子称为　标号定义

例1 -整树平方根：设计



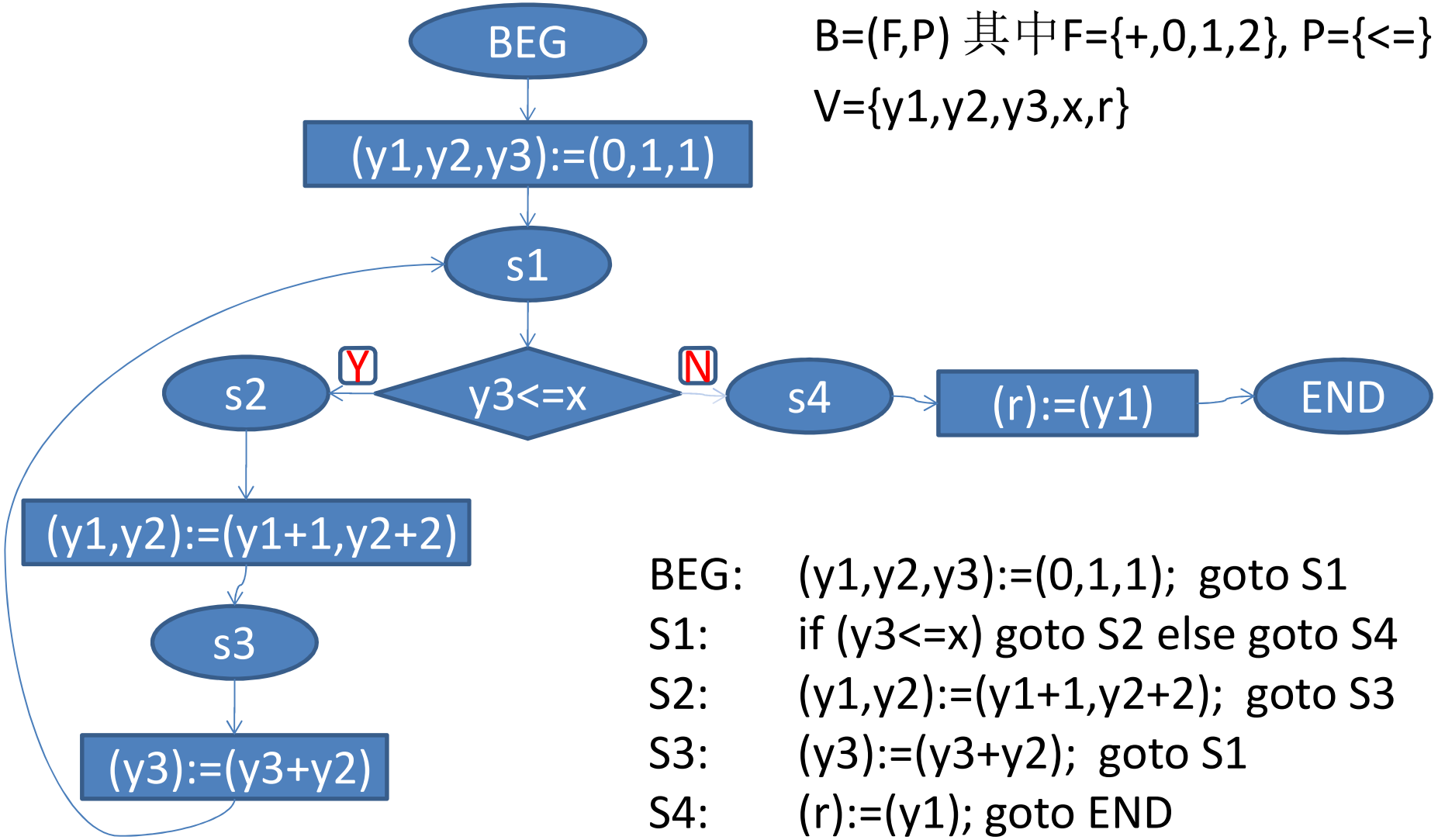
BEG: (y1,y2,y3):=(0,1,1); goto S1
S1: if (y3<=x) goto S2 else goto S4
S2: (y1,y2):=(y1+1,y2+2); goto S3
S3: (y3):=(y3+y2); goto S1
S4: (r):=(y1); goto END

流程图模型

一个(B,V)上的流程图模型T为满足以下条件的指令集合：

标号BEG必须有定义
出现在标号定义中的除END外的标号必须有定义
每个标号最多被定义一次

例1 -整树平方根：设计



B=(F,P) 其中F={+,0,1,2}, P={<=}

V={y1,y2,y3,x,r}

BEG: (y1,y2,y3):=(0,1,1); goto S1

S1: if (y3<=x) goto S2 else goto S4

S2: (y1,y2):=(y1+1,y2+2); goto S3

S3: (y3):=(y3+y2); goto S1

S4: (r):=(y1); goto END

模型

状态集合

状态迁移关系

计算/运行

系统状态

流程图模型的系统状态有两个部分：即标号和变量状态

标号可以理解为模型或系统的控制状态。

变量状态空间：V中变量取值的组合

$\Sigma = \{\sigma \mid \sigma(x) \in D, x \in V\}$

系统的状态空间为标号和V中变量取值的组合$LB \times \Sigma$

状态迁移关系

给定B的一个解释I。给定指令t。

$(L_i,\sigma_i) \rightarrow^t (L_{i+1}, \sigma_{i+1})$当且仅当以下一种情况成立：

t是　　L_i: (v1,...,vn):=(t1,...,tn); goto L_{i+1}　　且

σ_{i+1}是　　σ_i [v1/I(t1)(σ_i)]...[vn/I(tn)(σ_i)]

t是　　L_i: if (e) goto L’ else goto L’’，

σ_{i+1}是　　σ_i　　且

$\sigma_i \models_I$ e则L_{i+1}=L'，否则L_{i+1}=L''

状态迁移关系

给定B的一个解释I。流程图模型T。

$(L_i,\sigma_i) \rightarrow^T (L_{i+1},\sigma_{i+1})$当且仅当以下一种情况成立：

(1) 存在$t \in T$使得$(L_i,\sigma_i) \rightarrow^t (L_{i+1}, \sigma_{i+1})$

(2) L_i =END且$(L_i,\sigma_i) = (L_{i+1},\sigma_{i+1})$

例1 -整树平方根：设计

B=(F,P) 其中 F={+,0,1,2}, P={<=}

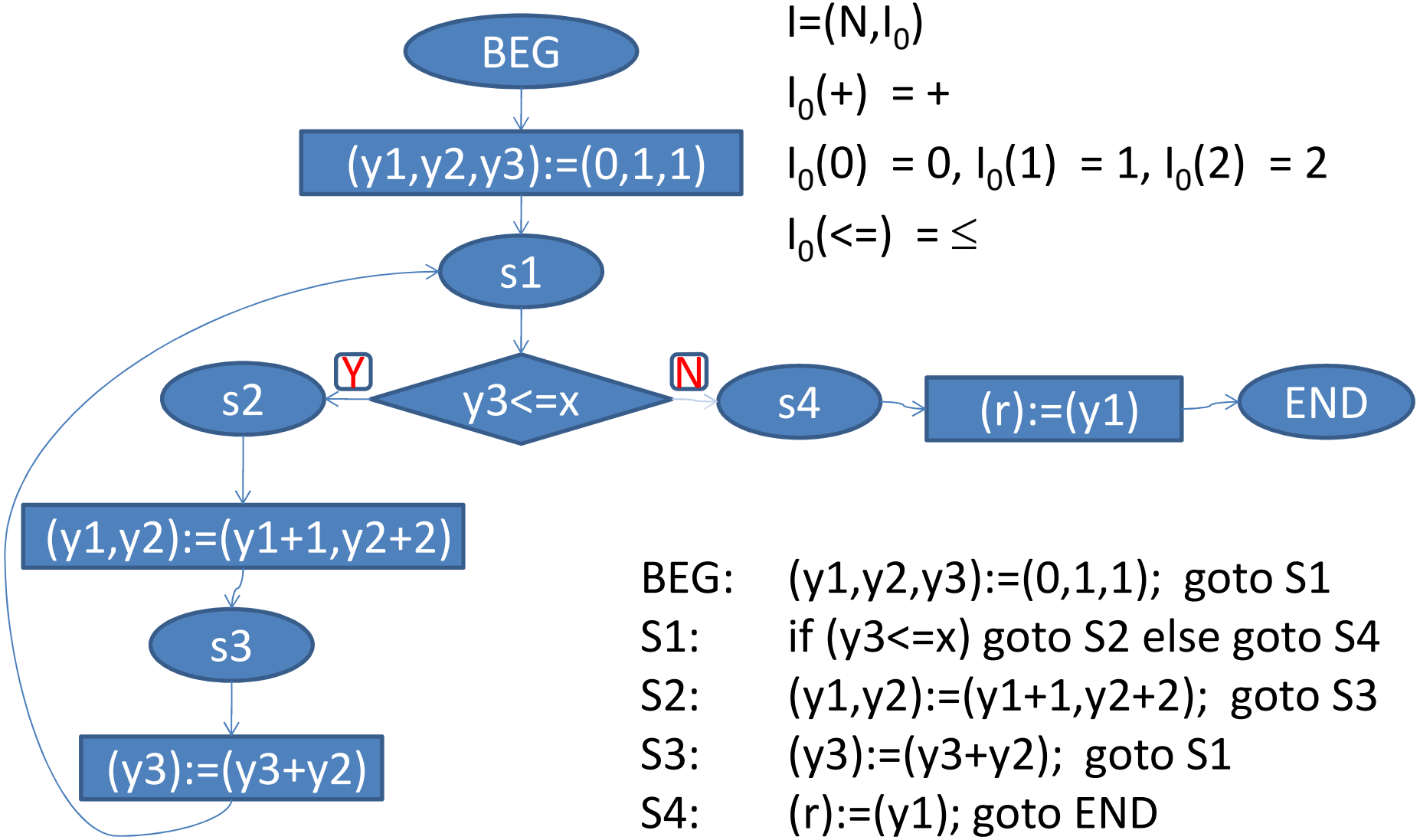
V={y1,y2,y3,x,r}

I=(N,I_0)

I_0(+) = +

I_0(0) = 0, I_0(1) = 1, I_0(2) = 2

I_0(<=) = ≤



BEG: (y1,y2,y3):=(0,1,1); goto S1

S1: if (y3<=x) goto S2 else goto S4

S2: (y1,y2):=(y1+1,y2+2); goto S3

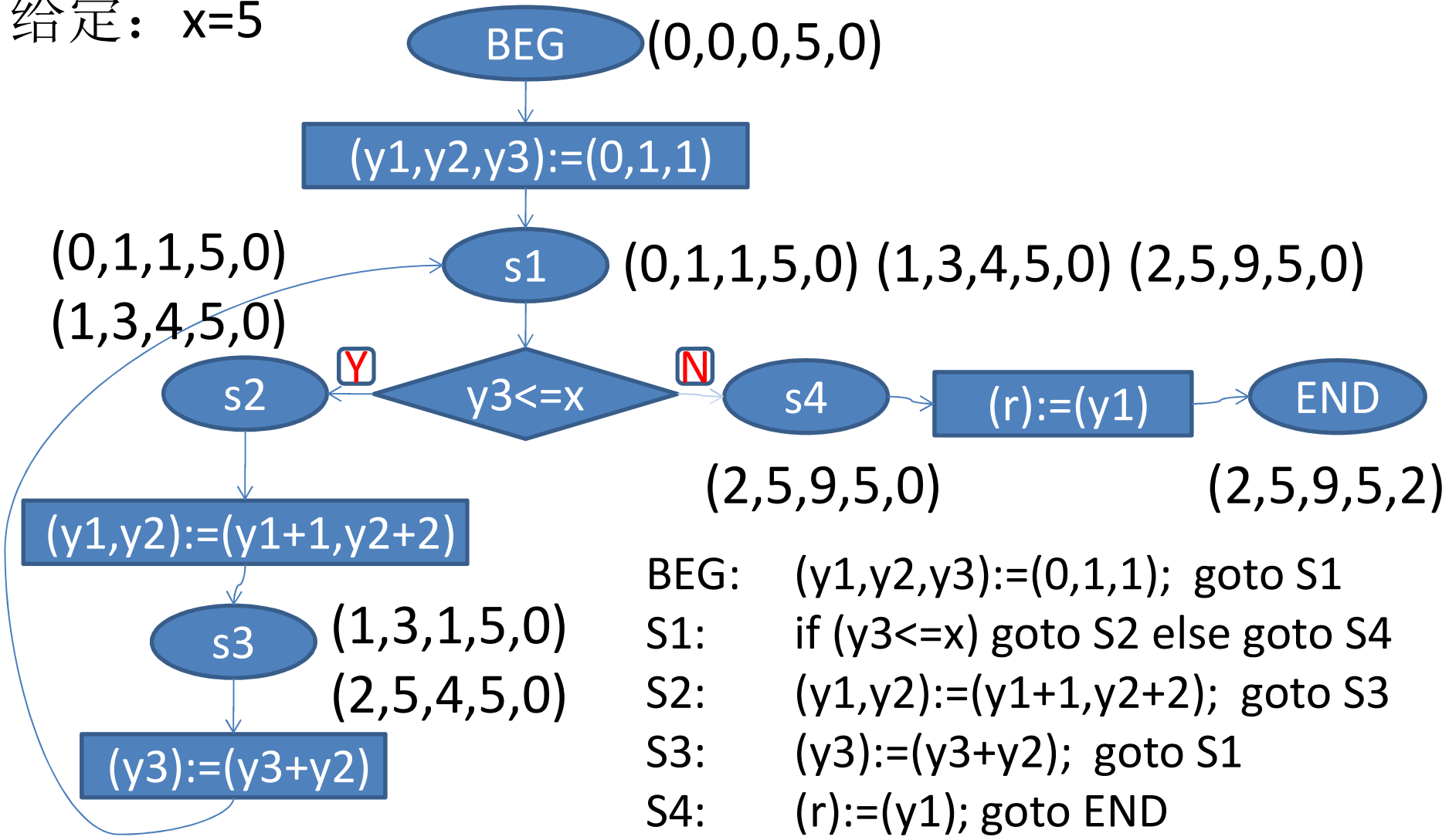
S3: (y3):=(y3+y2); goto S1

S4: (r):=(y1); goto END

例1 - 整树平方根

状态：LAB+(y1,y2,y3,x,r)

给定：x=5



BEG: (y1,y2,y3):=(0,1,1); goto S1

S1: if (y3<=x) goto S2 else goto S4

S2: (y1,y2):=(y1+1,y2+2); goto S3

S3: (y3):=(y3+y2); goto S1

S4: (r):=(y1); goto END

计算

流程图模型T的一个计算是

状态集LB x Σ上的一个无穷序列$(L_0,\sigma_0)(L_1,\sigma_1)$ ……

满足L_0 =BEG且对任意i有$(L_i,\sigma_i) \rightarrow^T (L_{i+1},\sigma_{i+1})$

可达状态集

定义 $\rightarrow^*$ 为 $\rightarrow^T$ 的自反传递闭包

流程图模型T在初始变量状态为σ_0时
运行可到达的状态的集合为 $\{ (L,\sigma) \mid (BEG,\sigma_0) \rightarrow^* (L,\sigma) \}$

流程图模型T在初始变量状态为σ时的计算可终止，
当且仅当存在σ’使得$(BEG,\sigma) \rightarrow^* (END,\sigma')$

模型

给定模型T：

状态集：$LB \times \Sigma$

迁移关系：$\rightarrow^T$

初始状态集：$\{ (BEG,\sigma) \mid \sigma \in \Sigma \}$

标号函数：

(III.b)正确性性质

$A \subseteq LB \times \Sigma$

A是M的安全性质，当且仅当M的所有计算都是A计算。

A是M的必达性质，当且仅当M的所有计算都有A状态出现。

正确性问题(终止性)

设谓词φ为模型T的初始状态满足的条件。

T对于φ的终止性定义如下：

$\varphi(\sigma) \Rightarrow \exists\sigma'.((BEG,\sigma) \rightarrow^{*} (END,\sigma'))$

正确性问题(部分正确性与完全正确性)

设

谓词φ为模型T的初始状态满足的条件，

谓词ψ为对模型T的终止状态的要求。

T对于φ和ψ的部分正确性和完全正确性定义如下：

部分正确性: $\varphi(\sigma) \Rightarrow (\forall\sigma'.(((BEG,\sigma) \rightarrow^{*} (END,\sigma')) \Rightarrow \psi(\sigma')))$

完全正确性: $\varphi(\sigma) \Rightarrow (\exists\sigma'.(((BEG,\sigma) \rightarrow^{*} (END,\sigma')) \wedge \psi(\sigma')))$

完全正确性　＝　部分正确性＋终止性

正确性问题(部分正确性与完全正确性)

相应地可以定义以公式为前后断言的模型正确性。

对于公式φ和ψ，可以把公式通过I解释成为谓词I(φ)和I(ψ)，则有类似的性质描述。

设
公式φ为模型T的初始状态满足的条件，
公式ψ为对模型T的终止状态的要求。
T对于φ和ψ的部分正确性和完全正确性定义如下：

部分正确性: $I(\varphi)(\sigma) \Rightarrow (\forall\sigma'.(((BEG,\sigma) \rightarrow^{*} (END,\sigma')) \Rightarrow I(\psi)(\sigma')))$

完全正确性: $I(\varphi)(\sigma) \Rightarrow (\exists\sigma'.(((BEG,\sigma) \rightarrow^{*} (END,\sigma')) \wedge I(\psi)(\sigma')))$

模型

给定模型T：

状态集：LB x Σ

迁移关系：$\rightarrow^T$

初始状态集：{ (BEG,σ) | $\sigma \in \Sigma$ }

谓词φ为模型T的初始状态满足的条件：

等价于将T的初始状态集限制在{ (BEG,σ) | $\varphi(\sigma)$ }

正确性问题

谓词φ为模型T的初始状态满足的条件：
等价于将T的初始状态集限制在$\{ (BEG,\sigma) \mid \varphi(\sigma) \}$

在这个受限的模型下：

终止性 – 必达性质(Inevitability)：
$\{ (END,\sigma) \mid \sigma \in \Sigma \}$ 是否必然可达？

部分正确性 – 安全性质(Safety)：
是否所有可达状态都满足
$\{ (END,\sigma) \mid \psi(\sigma) \} \cup \{ (L,\sigma) \mid L \neq END, \sigma \in \Sigma \}$ ？

(III.c)与卫式迁移模型的等价

1. 定义等价的含义： LB x Σ -- Σ’

$$((F,P),V) \;\; -- \;\; ((F \cup LB,P), V \cup \{pc\})$$

$$I \qquad\qquad I'$$

2. 定义一个转换方法，证明转换前后是等价的

流程图模型到卫式迁移模型的转换

T={ t1,t2,…,tk }

trans(T) = trans(t1) ∪ … ∪ trans(tn)

trans(L1: (x1,...,xk):=(e1,...,ek) ; goto L2) =

{ pc=L1 → (x1,...,xn,pc):=(e1,...,en,L2) }

trans(L1: if (e) goto L2 else goto L3) =

{pc=L1∧e → (pc):=(L2) } ∪ {pc=L1∧¬e → (pc):=(L3) }

M = (trans(T),pc=BEG)

流程图模型到卫式迁移模型的转换

B=(F,P)

V

LB = { BEG,L1,L2,…,Ln,END }

T={ t1,t2,…,tk }

I=(D,I_0)

B=(F∪{BEG,L1,…,Ln,END},P)

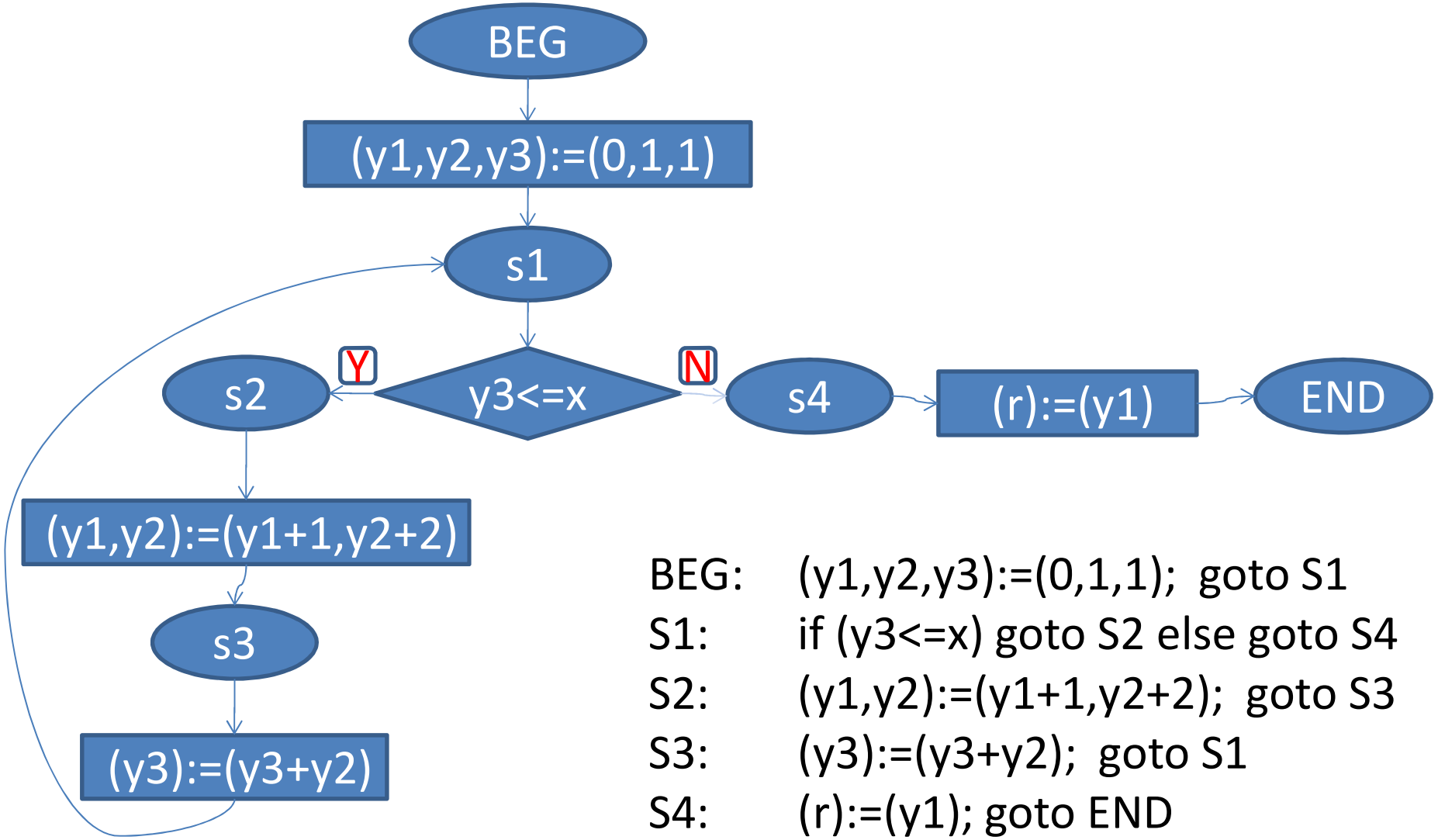
V=V ∪ {pc}

M=(trans(T),pc=BEG)

I=(D∪{0,1,…,n+1},I_0')

I_0'(BEG)=0, I_0'(Li)=i, I_0'(END)=n+1

例1 -整树平方根：设计



BEG: (y1,y2,y3):=(0,1,1); goto S1
S1: if (y3<=x) goto S2 else goto S4
S2: (y1,y2):=(y1+1,y2+2); goto S3
S3: (y3):=(y3+y2); goto S1
S4: (r):=(y1); goto END

例子-整树平方根

BEG:	(y1,y2,y3):=(0,1,1); goto S1
S1:	if (y3<=x) goto S2 else goto S4
S2:	(y1,y2):=(y1+1,y2+2); goto S3
S3:	(y3):=(y3+y2); goto S1
S4:	(r):=(y1); goto END

T:

pc=BEG $\rightarrow$	(y1,y2,y3,pc):=(0,1,1,S1)
pc=S1$\wedge$(y3<=x) $\rightarrow$	(pc):=(S2)
pc=S1$\wedge\neg$ (y3<=x) $\rightarrow$	(pc):=(S4)
pc=S2$\rightarrow$	(y1,y2,pc):=(y1+1,y2+2,S3)
pc=S3$\rightarrow$	(y3,pc):=(y3+y2,S1)
pc=S4$\rightarrow$	(r,pc):=(y1,END)

Θ: pc=BEG

(IV) 结构化程序模型

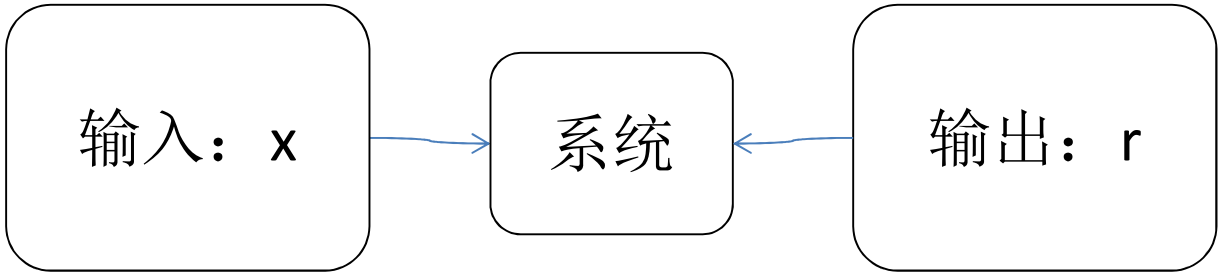
- Motivation

Motivation

Sequential computations:

A structural restriction

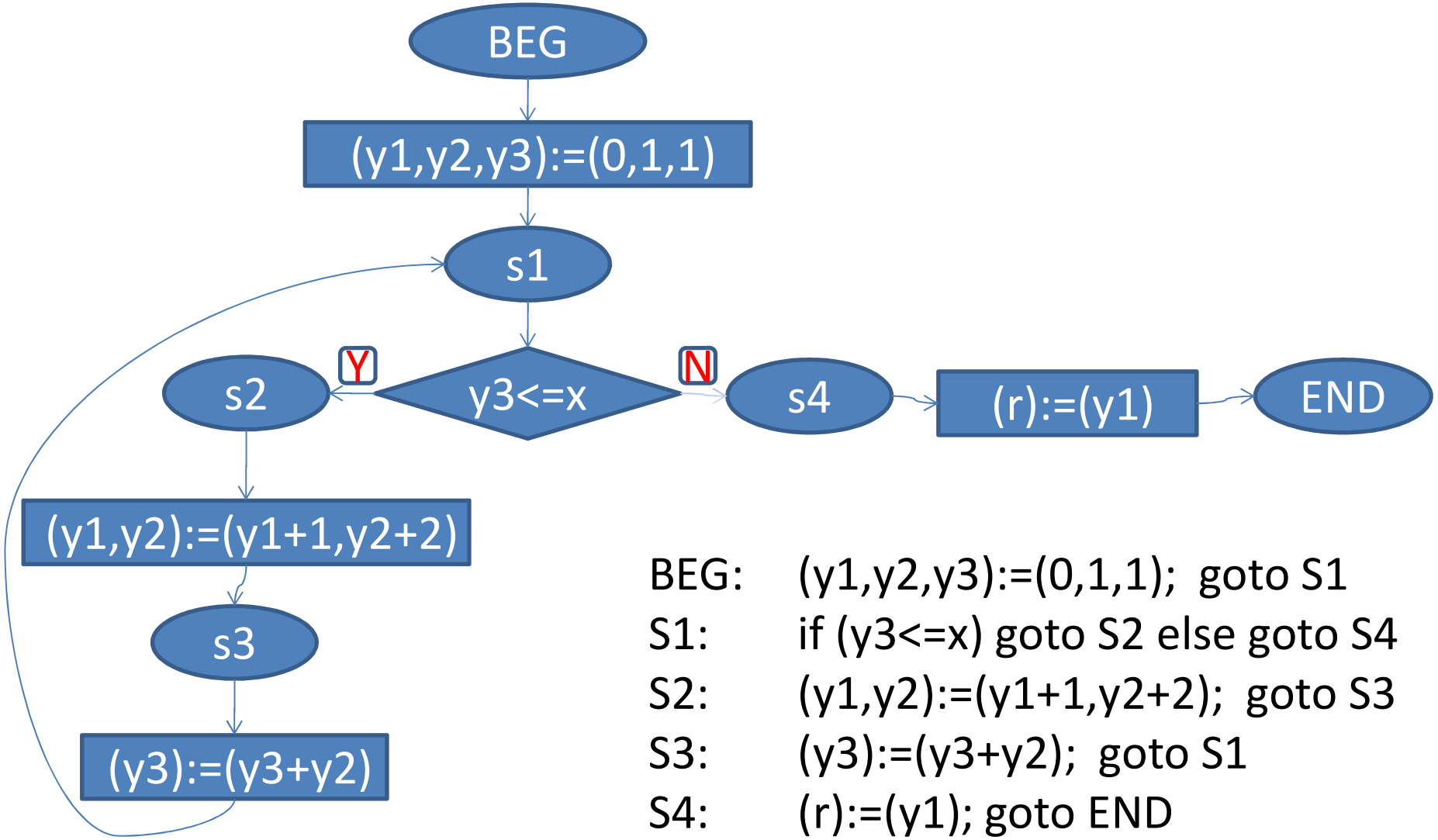
例1 -整树平方根：需求



输入：　　x;
初始条件：x>=0;

输出：　　r;
输出满足：x>=r*r; x<(r+1)*(r+1);

例子-整树平方根：设计



BEG: (y1,y2,y3):=(0,1,1); goto S1
S1: if (y3<=x) goto S2 else goto S4
S2: (y1,y2):=(y1+1,y2+2); goto S3
S3: (y3):=(y3+y2); goto S1
S4: (r):=(y1); goto END

程序代码(1)

```
int isqrt(int x)
{
   int y1,y2,y3,r;

   begin:      y1=0; y2=1; y3=1;  goto s1;
   s1:         if (y3<=x) goto s2; else goto s4;
   s2:         y1=y1+1; y2=y2+2;  goto s3;
   s3:         y3=y3+y2; goto s1;
   s4:         r=y1; goto end;
   end:        return r;
}
```

程序代码(2)

```
int isqrt(int x)
{
        int y1,y2,y3,r;

        y1=0; y2=1; y3=1;
        while (y3<=x) {
                y1=y1+1;
                y2=y2+2;
                y3=y3+y2;
        }
        r=y1;
        return r;
}
```

例1 -整树平方根：结构化程序模型

```
y1=0; y2=1; y3=1;
while (y3<=x) {
        y1=y1+1;
        y2=y2+2;
        y3=y3+y2;
}
r=y1;
```

结构化程序模型

一阶逻辑的扩充

主要有赋值、顺序复合、条件语句和循环语句。

在一阶逻辑的基础上，增加以下辅助符号集：

{:=, ;, if, then, else, fi, while, do, od, ε}

结构化程序模型

- 定义
- 性质
- 结构化程序模型与顺序流程图模型的关系

(IV.a)结构化程序模型

给定B=(F,P)和变量集合V。
(B,V)上的结构化程序模型是一个字符串，
其集合***S***定义如下：

S ::= ε | T; ε

T ::= x:=t | T;T | if (e) then T else T fi | while (e) do T od

其中　　x ∈V 为变量,
　　t ∈T_B 且在t中出现的变量在 V中
　　e ∈ QFF_B且在e中出现的变量在 V中

例1 -整树平方根：软件编码2

```
y1=0; y2=1; y3=1;
while (y3<=x) {
        y1=y1+1;
        y2=y2+2;
        y3=y3+y2;
}
r=y1;
```

```
y1:=0; y2:=1; y3:=1;
while (y3<=x) do
        y1:=y1+1;
        y2:=y2+2;
        y3:=y3+y2
od;
r:=y1;
ε
```

例1 -整树平方根：软件编码2

```
 y1=0; y2=1; y3=1;
while (y3<=x) {
        y1=y1+1;
        y2=y2+2;
        y3=y3+y2;
}
r=y1;
```

```
y1:=0; y2:=1; y3:=1;
while (y3<=x) do
        y1:=y1+1;
        y2:=y2+2;
        y3:=y3+y2
od;
r:=y1;
ε
```

B=(F,P)

F={+,0,1,2}

P={<=,=}

V={y1,y2,y3,x,r}

模型

状态集合

状态迁移关系

计算/运行

系统状态

结构化程序模型的系统状态有两个部分：
模型运行的剩余部分(控制状态)和变量状态

变量状态空间：V中变量取值的组合

$\Sigma = \{\sigma \mid \sigma(x) \in D, x \in V\}$

系统的状态空间为程序和V中变量取值的组合$S \times \Sigma$

状态迁移关系

给定B的一个解释I。给定S。
$(S,\sigma) \rightarrow (S^*,\sigma^*)$当且仅当以下一种情况成立：

S是ε且$S^*=S$且$\sigma^*=\sigma$
S是$x:=t;S^*$ 且$\sigma^*=\sigma[x/I(t)(\sigma)]$

S是if (e) then T else T’ fi; S’且$\sigma \models_I e$且$S^*=T;S'$且$\sigma^*=\sigma$
S是if (e) then T else T’ fi; S’且$\neg(\sigma \models_I e)$且$S^*=T';S'$且$\sigma^*=\sigma$

S是while (e) do T od; S’且$\sigma \models_I e$且$S^*=T;S$且$\sigma^*=\sigma$
S是while (e) do T od; S’且$\neg(\sigma \models_I e)$且$S^*=S'$且$\sigma^*=\sigma$

例1 -整树平方根：软件编码2

B=(F,P)

F={+,0,1,2}

P={<=,=}

V={y1,y2,y3,x,r}

$I=(N,I_0)$

$I_0(+)$ = **+**

$I_0(0)$ = **0**

$I_0(1)$ = **1,** $I_0(2)$ = **2**

$I_0(<=)$ = ≤

$I_0(=)$ = **=**

```
y1:=0; y2:=1; y3:=1;
while (y3<=x) do
        y1:=y1+1;
        y2:=y2+2;
        y3:=y3+y2
od;
r:=y1;
ε
```

计算

结构化程序模型S的一个计算是

状态集$\boldsymbol{S}$ x Σ上的一个无穷序列$(S_0,\sigma_0)(S_1,\sigma_1)$ ……

满足S_0 =S且对任意i有$(S_i,\sigma_i) \rightarrow (S_{i+1},\sigma_{i+1})$

可达状态集

定义 $\rightarrow^*$ 为 $\rightarrow$ 的自反传递闭包

结构化程序模型S在初始变量状态为σ_0时
运行可到达的状态的集合为 $\{ (S',\sigma) \mid (S,\sigma_0) \rightarrow^* (S',\sigma) \}$

模型S在初始变量状态为σ时的计算可终止，
当且仅当存在σ’使得$(S,\sigma) \rightarrow^* (\varepsilon,\sigma')$

模型

给定模型S：

状态集：$S \times \Sigma$

迁移关系：$\rightarrow$

初始状态集：$\{ (S,\sigma) \mid \sigma \in \Sigma \}$

标号函数：

(IV.b)正确性性质

$A \subseteq \boldsymbol{S} \times \Sigma$

A是M的安全性质, 当且仅当M的所有计算都是A计算.

A是M的必达性质, 当且仅当M的所有计算都有A状态出现.

正确性问题(终止性)

设谓词φ为模型S的初始状态满足的条件。

S对于φ的终止性定义如下：

$\varphi(\sigma) \Rightarrow \exists \sigma'.((S,\sigma) \rightarrow^{*} (\varepsilon,\sigma'))$

正确性问题(部分正确性与完全正确性)

设

谓词φ为模型S的初始状态满足的条件，

谓词ψ为对模型S的终止状态的要求。

S对于φ和ψ的部分正确性和完全正确性定义如下：

部分正确性: $\varphi(\sigma) \Rightarrow (\forall\sigma'.(((S,\sigma) \rightarrow^{*} (\varepsilon,\sigma')) \Rightarrow \psi(\sigma')))$

完全正确性: $\varphi(\sigma) \Rightarrow (\exists\sigma'.(((S,\sigma) \rightarrow^{*} (\varepsilon,\sigma')) \wedge \psi(\sigma')))$

完全正确性　＝　部分正确性＋终止性

正确性问题(部分正确性与完全正确性)

相应地可以定义以公式为前后断言的模型正确性。

对于公式φ和ψ，可以把公式通过I解释成为谓词I(φ)和I(ψ)，则有类似的性质描述。

设
公式φ为模型S的初始状态满足的条件，
公式ψ为对模型S的终止状态的要求。
S对于φ和ψ的部分正确性和完全正确性定义如下：

部分正确性: $I(\varphi)(\sigma) \Rightarrow (\forall \sigma'.(((S,\sigma) \rightarrow^{*} (\varepsilon,\sigma')) \Rightarrow I(\psi)(\sigma')))$

完全正确性: $I(\varphi)(\sigma) \Rightarrow (\exists \sigma'.(((S,\sigma) \rightarrow^{*} (\varepsilon,\sigma')) \wedge I(\psi)(\sigma')))$

模型

给定模型S：

状态集：$S \times \Sigma$
迁移关系：$\rightarrow$
初始状态集：$\{ (S,\sigma) \mid \sigma \in \Sigma \}$

谓词φ为模型S的初始状态满足的条件：
等价于将S的初始状态集限制在$\{ (S,\sigma) \mid \varphi(\sigma) \}$

正确性问题

谓词φ为模型S的初始状态满足的条件：
等价于将S的初始状态集限制在$\{ (S,\sigma) \mid \varphi(\sigma) \}$

在这个受限的模型下：

终止性 –必达性质(Inevitability)：
$\{ (\varepsilon,\sigma) \mid \sigma \in \Sigma \}$ 是否必然可达？

部分正确性 – 安全性质(Safety)：
是否所有可达状态都满足
$\{ (\varepsilon,\sigma) \mid \psi(\sigma) \} \cup \{ (S',\sigma) \mid S' \neq \varepsilon, \sigma \in \Sigma \}$ ？

(IV.c)与顺序流程图模型的等价

1. 定义等价的含义： $S \times \Sigma$ -- $LB \times \Sigma$

   (B,V) -- (B,V)

2. 定义一个转换方法，证明转换前后是等价的

结构化程序模型到流程图模型的转换

对结构化程序模型进行标记；
赋予每个语句的入口一个唯一的标识。

labeling(T;ε) = labeling(BEG: T); END: ε

labeling(l0: x:=t) = l0: x:=t

labeling(l0: T1;T2)= = labeling(l0: T1); labeling(l1: T2);

labeling(l0: while (e) do T1 od) = l0: while (e) do labeling(l1: T1); od

labeling(l0: if (e) then T1 else T2 fi) = l0: if (e) then labeling(l1: T1)
else labeling(l2: T2) fi

结构化程序模型到流程图模型的转换

对标记过的结构化程序模型进行转换。

trans(L: ε) = {}

trans(L: x:=t; S) = { L: (x):=(t); goto lb(S) } ∪ trans(S)

trans(L: while (e) do S od; S') = { L: if (e) goto lb(S) else goto lb(S') } ∪
trans(S; L:ε) ∪
trans(S')

trans(L: if (e) then S else S'; S'') = { L: if (e) goto lb(S) else goto lb(S') } ∪
trans(S;lb(S''):ε) ∪ trans(S';lb(S''):ε)∪
trans(S'')

结构化程序模型到流程图模型的转换

B=(F,P), V

S

I=(D,I_0)

B=(F,P), V

T=trans(labeling(S))

I=(D,I_0)

例1 -整树平方根：软件编码2

```
y1=0; y2=1; y3=1;
while (y3<=x) {
        y1=y1+1;
        y2=y2+2;
        y3=y3+y2;
}
r=y1; ε
```

例子-整树平方根

```
BEG:                    (y1,y2,y3):=(0,1,1);  goto S1
S1:                     if (y3<=x) goto S2 else goto S4
S2:                     (y1,y2):=(y1+1,y2+2);  goto S3
S3:                     (y3):=(y3+y2);  goto S1
S4:                     (r):=(y1); goto END
```

```
y1:=0; y2:=1; y3:=1;
while (y3<=x) {
        y1:=y1+1;
        y2:=y2+2;
        y3:=y3+y2;
}
r:=y1; ε
```

例子-整树平方根

```
BEG:                    (y1,y2,y3):=(0,1,1);  goto S1
S1:                     if (y3<=x) goto S2 else goto S4
S2:                     (y1,y2):=(y1+1,y2+2);  goto S3
S3:                     (y3):=(y3+y2);  goto S1
S4:                     (r):=(y1); goto END
```

```
BEG: y1:=0;
L1:   y2:=1;
L2:   y3:=1;
L3:   while (y3<=x) {
L4:                 y1:=y1+1;
L5:                 y2:=y2+2;
L6:                 y3:=y3+y2;
          }
L7:   r:=y1;
END: ε
```

加标号

例子-整树平方根

```
BEG:                    (y1,y2,y3):=(0,1,1);  goto S1
S1:                     if (y3<=x) goto S2 else goto S4
S2:                     (y1,y2):=(y1+1,y2+2);  goto S3
S3:                     (y3):=(y3+y2);  goto S1
S4:                     (r):=(y1); goto END
```

```
BEG: y1:=0;
L1:   y2:=1;
L2:   y3:=1;
L3:   while (y3<=x) {
L4:                 y1:=y1+1;
L5:                 y2:=y2+2;
L6:                 y3:=y3+y2;
          }
L7:   r:=y1;
END: ε
```

加标号

```
BEG:   (y1):=(0);                    goto L1
L1:    (y2):=(1);                    goto L2
L2:    (y3):=(1);                    goto L3
L3:    if (y3<=x) goto L4 else goto L7
L4:    (y1):=(y1+1);                 goto L5
L5:    (y2):=(y2+2);                 goto L6
L6:    (y3):=(y3+y2);                goto L3

L7:    (r):=(y1);                    goto END
```

完成转换

(V) 小结

- 卫式迁移模型
- 公平卫式迁移模型及实例
- 顺序流程图模型
- 结构化程序模型

表达能力问题　表达方式问题

思考题：

(1)卫式迁移模型 — 建模与分析：

定义一个三个进程的互斥协议的(公平)卫式迁移模型。
说明 模型中所用到的各类符号及其解释，
定义该协议应有的安全和必达性质并分析其正确性。

(2)流程图模型与标号Kripke模型的等价 — 定义与构造：

定义一种具备一定合理性的流程图模型与标号Kripke模型的计算等价关系，并
给出一种能够满足前述等价关系的、由流程图模型构造标号Kripke模型的方法。