ICALP 2023 – Paderborn Germany

Regular Methods for Operator Precedence Languages

Thomas A. Henzinger [1]

Pavol Kebis [1,2]

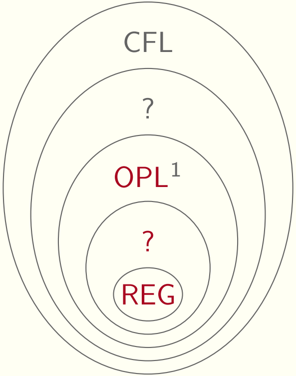
Nicolas Mazzocchi [1]

N. Ege Saraç [1]

1 Institute of Science and Technology, Austria

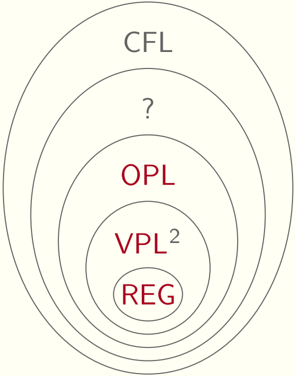
2 University of Oxford, United Kingdom

Operator Precedence Languages



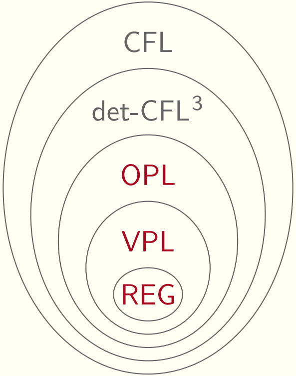
[1] R. W. Floyd. *Syntactic Analysis and Operator Precedence*. Journal of the ACM 10, 1963

Operator Precedence Languages



[2] R. Alur, P. Madhusudan. *Visibly pushdown languages*. STOC 2004

Operator Precedence Languages



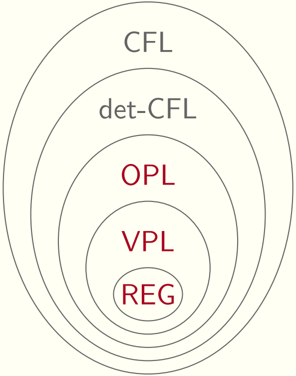
[3] Géraud Sénizergues. *L(A)=L(B)? decidability results from complete formal systems*. ICALP 2002

Operator Precedence Languages

Closures[4]

	$\cup$	$\cap$	$\neg$
REG	✓	✓	✓
VPL	✓	✓	✓
OPL	✓	✓	✓
det-CFL	×	×	✓
CFL	✓	×	×

- $\cup$ union
- $\cap$ intersection
- $\neg$ complement



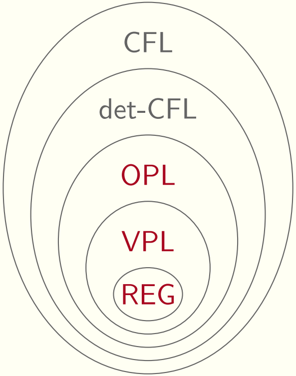
[4] S. Crespi-Reghizzi et al. *Algebraic Properties of OPLs.* Information and Control 37, 1978.

Operator Precedence Languages

Closures

	$\cup$	$\cap$	$\neg$
REG	✓	✓	✓
VPL	✓	✓	✓
OPL	✓	✓	✓
det-CFL	×	×	✓
CFL	✓	×	×

$\cup$ union
$\cap$ intersection
$\neg$ complement



Decidability[5]

	$\varnothing$	$=$	$\subseteq$
REG	✓	✓	✓
VPL	✓	✓	✓
OPL	✓	✓	✓
det-CFL	✓	✓	×
CFL	✓	×	×

$\varnothing$ emptiness
$=$ equivalence
$\subseteq$ inclusion

[5] V. Lonati et al. *OPLs: Their automata-theoretic and logic characterization*. SICOMP 44, 2015

Example of System

```
call_A() {
    try
        call_B()
    catch
        call_Err()
    return_A
}
```

```
call_B() {
    select (*)
        call_C()
        call_B()
        throw
    return_B
}
```

```
call_C() {
    return_C
}

call_Err() {
    return_Err
}
```

Example of System

```
call_A() {
    try
        call_B()
    catch
        call_Err()
    return_A
}
```

```
call_B() {
    select (*)
        call_C()
        call_B()
        throw
    return_B
}
```

```
call_C() {
    return_C
}

call_Err() {
    return_Err
}
```

Specifications

- throw always preceded by try
- catch if and only if throw before
- call always eventually ended by return or throw

Example of System

```
call_A() {
    try
        call_B()
    catch
        call_Err()
    return_A
}
```

```
call_B() {
    select (*)
        call_C()
        call_B()
        throw
    return_B
}
```

```
call_C() {
    return_C
}

call_Err() {
    return_Err
}
```

Specifications

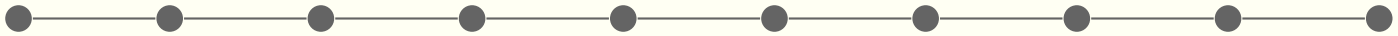
- throw always preceded by try
- catch if and only if throw before
- call always eventually ended by return or throw

Model-Checking

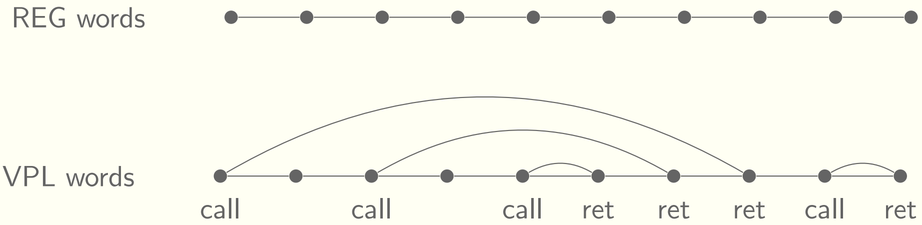
$$L_{\text{System}} \subseteq L_{\text{Specification}}$$

- undecidable for det-CFL

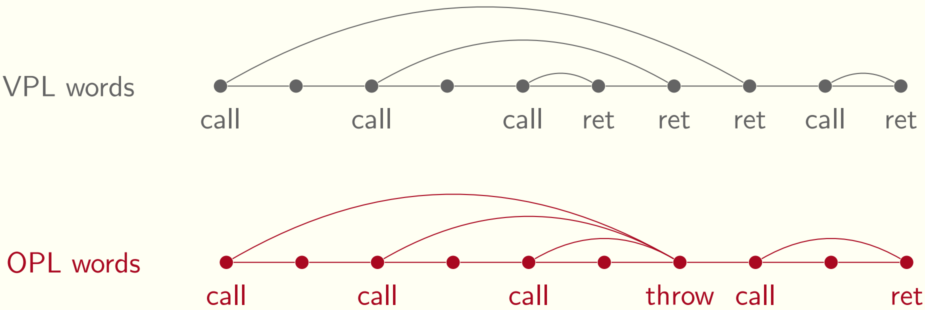
Structured Words

REG words 

Structured Words



Structured Words

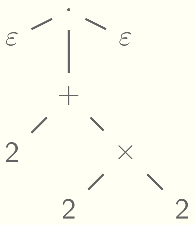


Operator Precedence Words

Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

$$2 + 2 \times 2$$

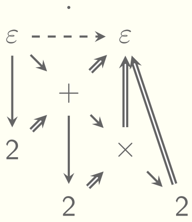


Operator Precedence Words

Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

$$2 + 2 \times 2$$



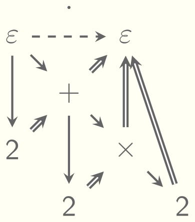
Operator Precedence Words

Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

	2	$+$	$\times$	ε
2	-	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$
$\times$	$\lessdot$	$\gtrdot$	$\doteq$	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

$$2 + 2 \times 2$$



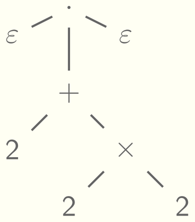
Operator Precedence Words

Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

	2	$+$	$\times$	ε
2	-	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$
$\times$	$\lessdot$	$\gtrdot$	$\doteq$	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$



Operator Precedence Words

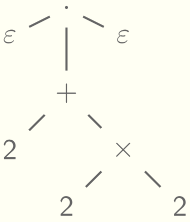
Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

	2	$+$	$\times$	ε
2	-	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$
$\times$	$\lessdot$	$\gtrdot$	$\doteq$	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$



Operator Precedence Words

Operator Precedence Alphabet

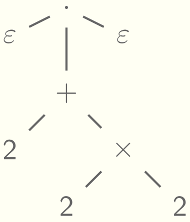
- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

	2	$+$	$\times$	ε
2	-	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$
$\times$	$\lessdot$	$\gtrdot$	$\doteq$	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \gtrdot \varepsilon$$



Operator Precedence Words

Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

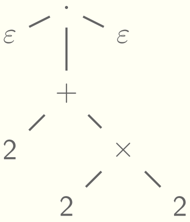
	2	$+$	$\times$	ε
2	-	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$
$\times$	$\lessdot$	$\gtrdot$	$\doteq$	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \varepsilon$$

$$\varepsilon \lessdot 2 \gtrdot + \lessdot \lessdot 2 \gtrdot \times \lessdot 2 \gtrdot \gtrdot \varepsilon$$

$$\varepsilon \lessdot \lessdot 2 \gtrdot + \lessdot \times \gtrdot \gtrdot \varepsilon$$



Operator Precedence Words

Operator Precedence Alphabet

- a **yields** precedence to b, denoted $a \lessdot b$,
- a **takes** precedence over b, denoted $a \gtrdot b$,
- a **equals** in precedence with b, denoted $a \doteq b$.

	2	$+$	$\times$	ε
2	-	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$
$\times$	$\lessdot$	$\gtrdot$	$\doteq$	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Chain

- $a_0 \ldots a_n$ is a **simple chain** when $a_0 \lessdot a_1 \doteq \ldots \doteq a_{n-1} \gtrdot a_n$ $\quad {}^{a_0}[a_1 \ldots a_{n-1}]^{a_n}$
- $a_0 u_0 \ldots b_{n-1} u_{n-1} b_n$ is a **nested chain** when ${}^{b_0}[b_1 \ldots b_{n-1}]^{b_n}$ and $u_i \neq \varepsilon \Rightarrow {}^{b_i}[u_i]^{b_{i+1}}$

Operator Precedence Automata

Like pushdown automata

- Q set of states, q_0 initial state, F finial states
- $\widehat{\Sigma}$ words alphabet, Γ stack alphabet
- Δ transition relation from $Q \times \widehat{\Sigma} \times \Gamma$ to a finite subset of $Q \times \Gamma^*$

Operator Precedence Automata

Like pushdown automata

- Q set of states, q_0 initial state, F finial states
- $\widehat{\Sigma}$ words alphabet, Γ stack alphabet
- Δ transition relation from $Q \times \widehat{\Sigma} \times \Gamma$ to a finite subset of $Q \times \Gamma^*$

With input-driven transitions

- Push: $b \lessdot a$
- Shift: $b \doteq a$
- Pop: $b \gtrdot a$
- Empty stack: Always push on $\perp$

Operator Precedence Automata

Like pushdown automata

- Q set of states, q_0 initial state, F finial states
- $\widehat{\Sigma}$ words alphabet, Γ stack alphabet
- Δ transition relation from $Q \times \widehat{\Sigma} \times \Gamma$ to a finite subset of $Q \times \Gamma^*$

With input-driven transitions

- Push: $(q, \langle b, p \rangle \theta) \xrightarrow{a} (q', \langle a, q \rangle \langle b, p \rangle \theta)$ $b \lessdot a$
- Shift: $b \doteq a$
- Pop: $b \gtrdot a$
- Empty stack: Always push on $\perp$

Operator Precedence Automata

Like pushdown automata

- Q set of states, q_0 initial state, F finial states
- $\widehat{\Sigma}$ words alphabet, Γ stack alphabet
- Δ transition relation from $Q \times \widehat{\Sigma} \times \Gamma$ to a finite subset of $Q \times \Gamma^*$

With input-driven transitions

- Push: $(q, \langle b, p\rangle\theta) \xrightarrow{a} (q', \langle a, q\rangle\langle b, p\rangle\theta)$ — $b \lessdot a$
- Shift: $(q, \langle b, p\rangle\theta) \overset{a}{\dashrightarrow} (q', \langle a, p\rangle\theta)$ — $b \doteq a$
- Pop: — $b \gtrdot a$
- Empty stack: Always push on $\bot$

Operator Precedence Automata

Like pushdown automata

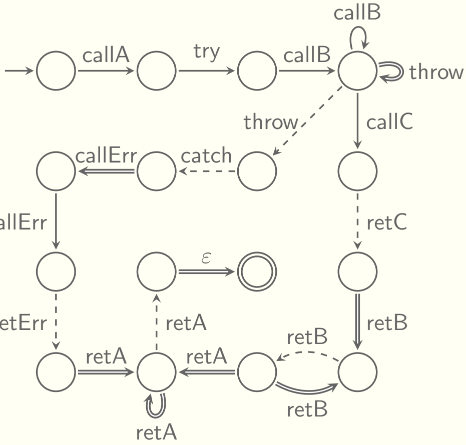
- Q set of states, q_0 initial state, F finial states
- $\widehat{\Sigma}$ words alphabet, Γ stack alphabet
- Δ transition relation from $Q \times \widehat{\Sigma} \times \Gamma$ to a finite subset of $Q \times \Gamma^*$

With input-driven transitions

- Push: $(q, \langle b, p\rangle\theta) \xrightarrow{a} (q', \langle a, q\rangle\langle b, p\rangle\theta)$ $\quad b \lessdot a$
- Shift: $(q, \langle b, p\rangle\theta) \dashrightarrow^{a} (q', \langle a, p\rangle\theta)$ $\quad b \doteq a$
- Pop: $(q, \langle b, p\rangle\theta) \overset{a}{\Longrightarrow} (q', \theta)$ without consuming the input letter $\quad b \gtrdot a$
- Empty stack: Always push on $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

$\perp$

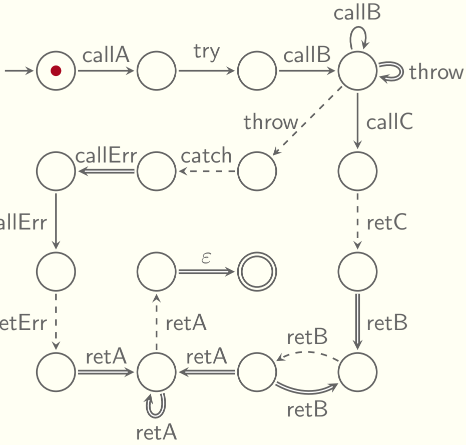
Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

$\perp$



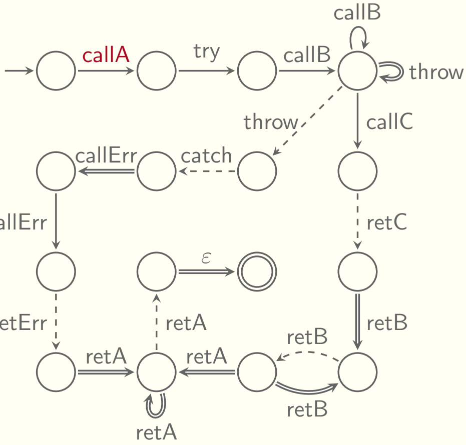
Example of Modelization

▸▸ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

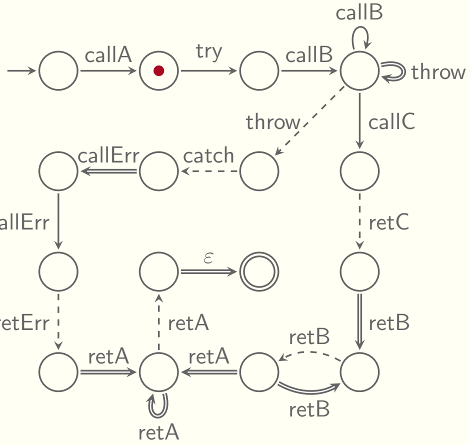
Stack

callA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

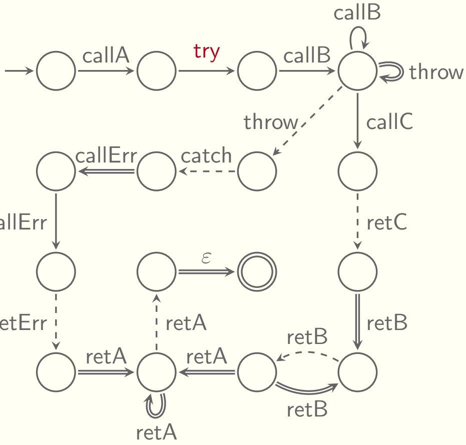
callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

try callA $\perp$

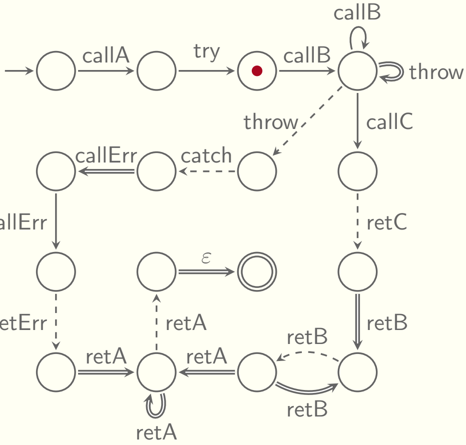


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

try callA $\perp$

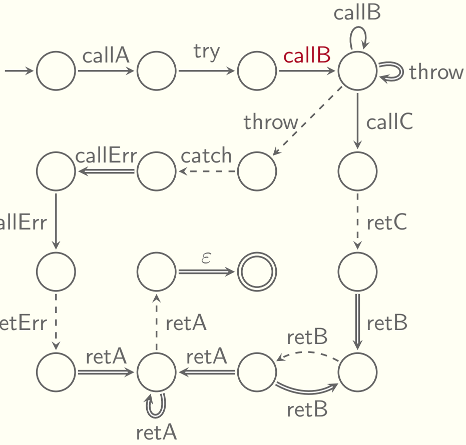


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

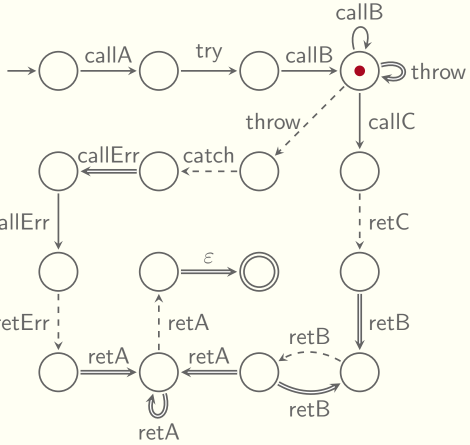
Stack

callB try callA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

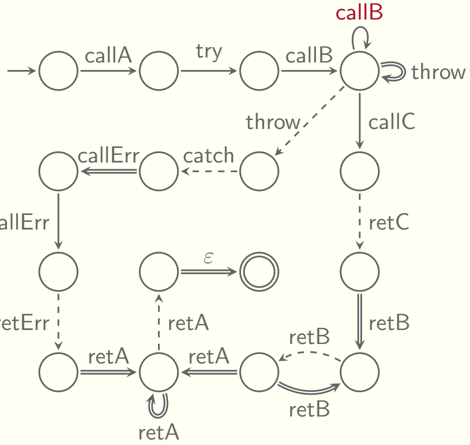


Stack

callB try callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

callB callB try callA $\perp$

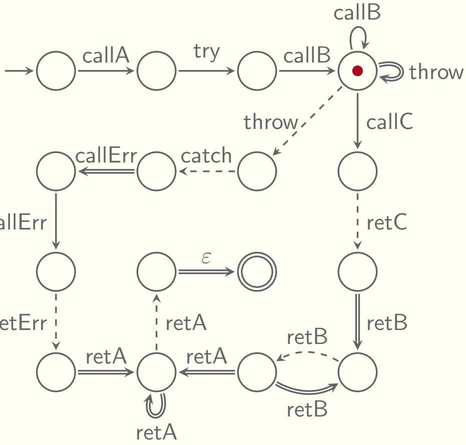
Example of Modelization

skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callB callB try callA $\perp$

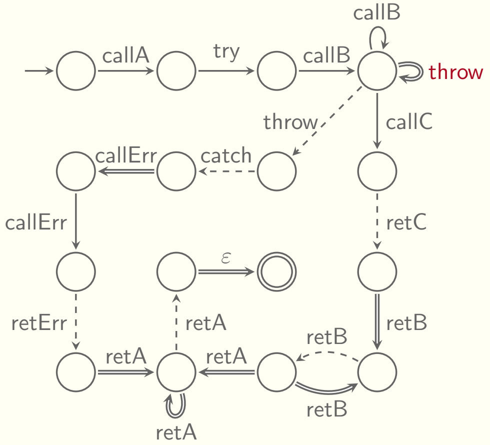


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

~~callB~~ callB try callA $\perp$

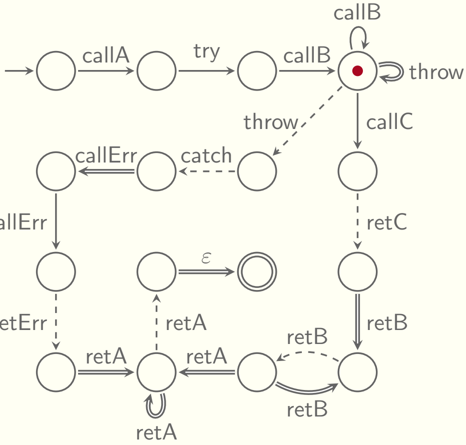


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

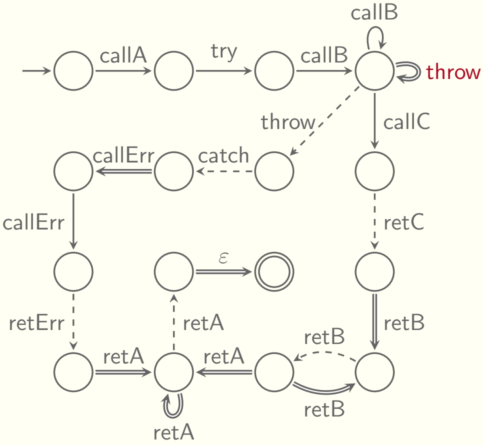
Stack

callB try callA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

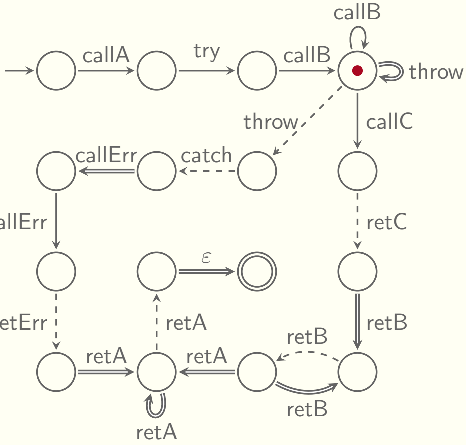


Stack

$\cancel{callB}$ try callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

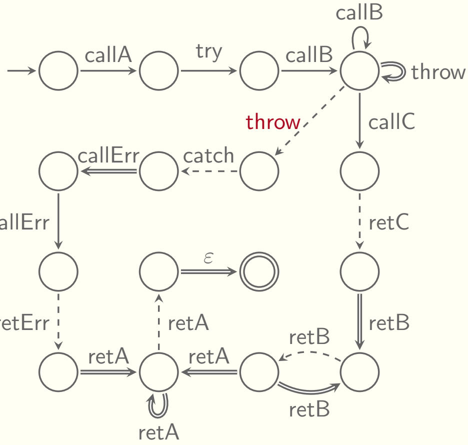


Stack

try callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



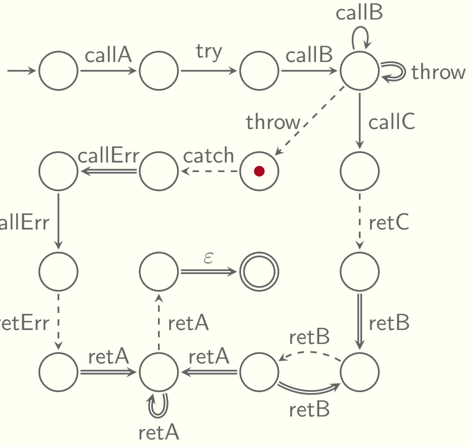
Stack

throw callA $\perp$

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

throw callA $\perp$

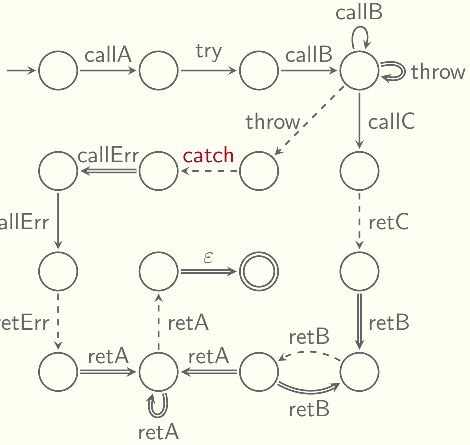


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

catch callA $\perp$

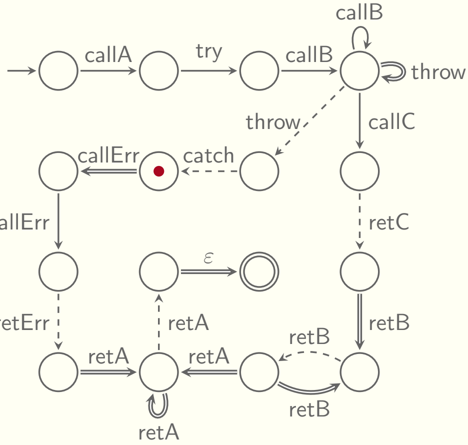


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

catch callA $\perp$

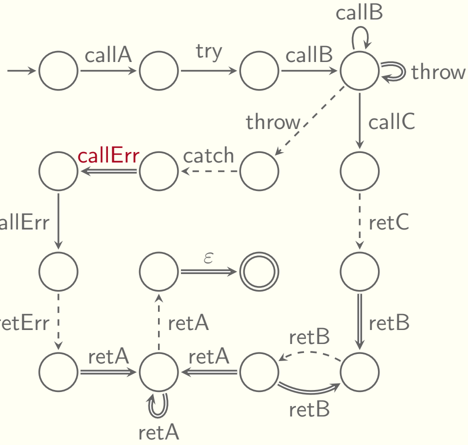


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

~~catch~~ callA $\perp$

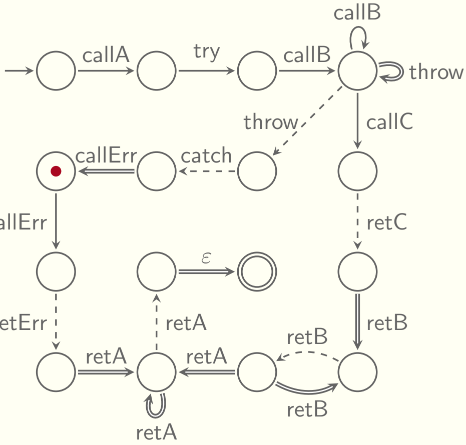


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

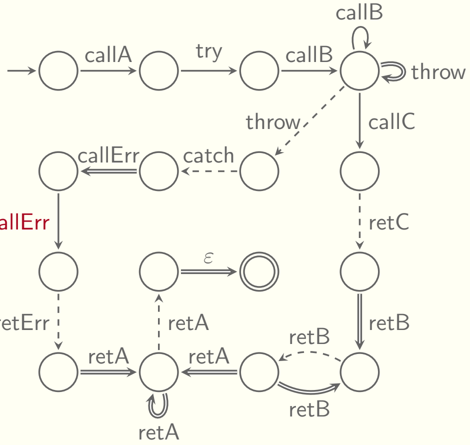
callA $\perp$



Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

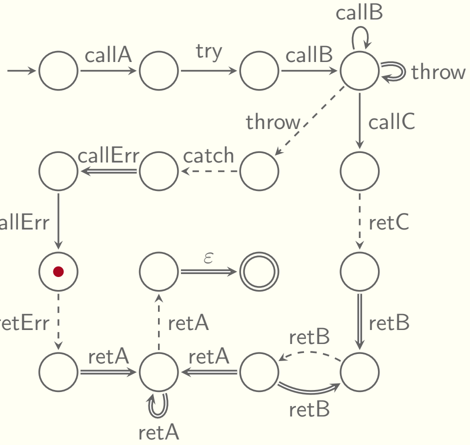
callErr callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callErr callA $\perp$

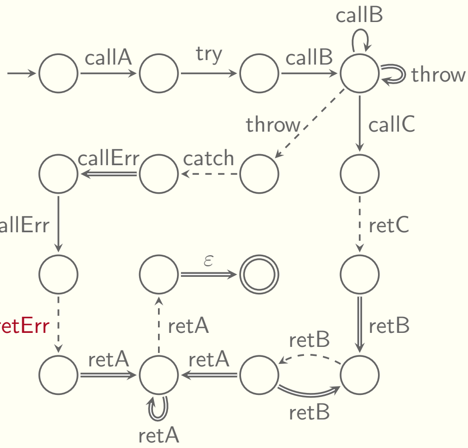


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

retErr callA $\perp$

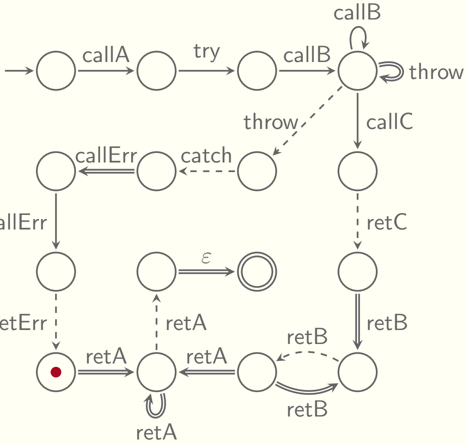


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

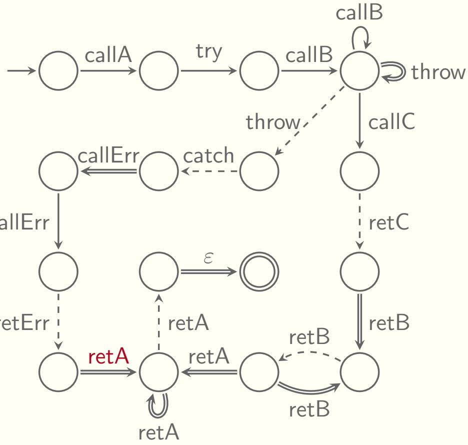
retErr callA $\perp$



Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

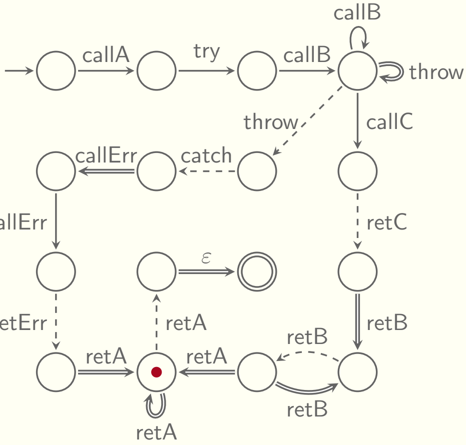
~~retErr~~ callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

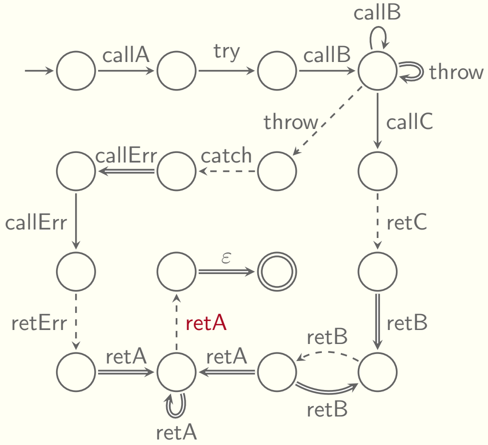
Stack

callA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

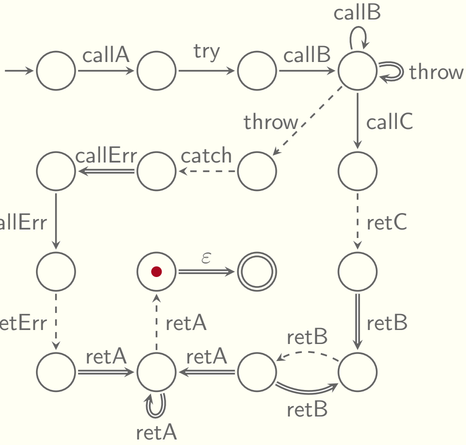


Stack

retA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

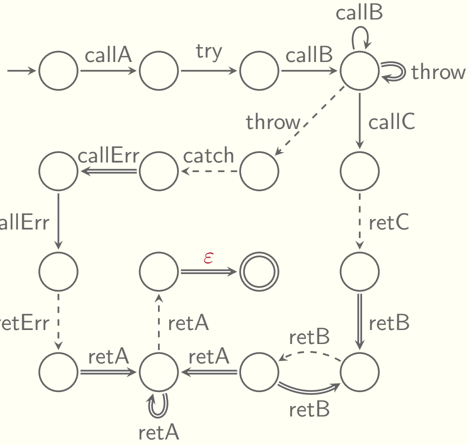


Stack

retA ⊥

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

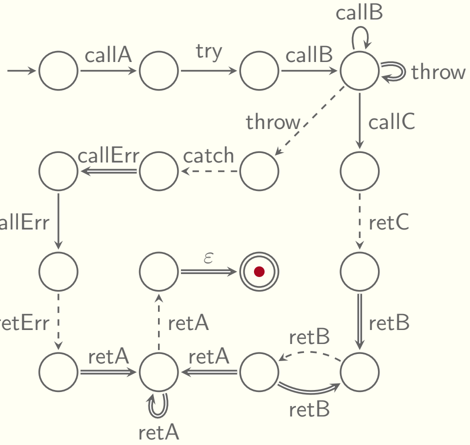


Stack

~~retA~~ $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

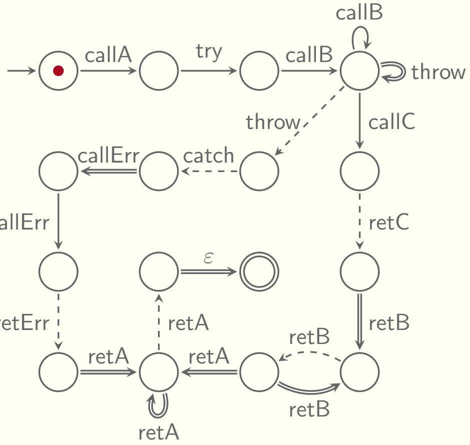


Stack

$\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

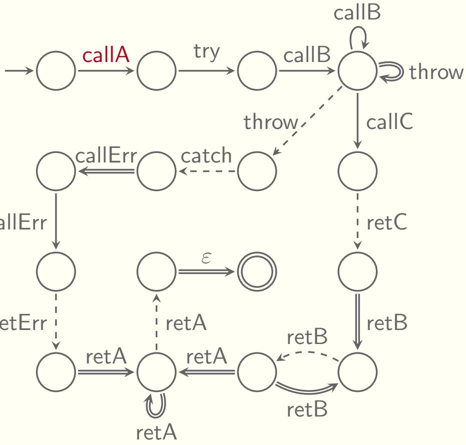
$\perp$

Example of Modelization

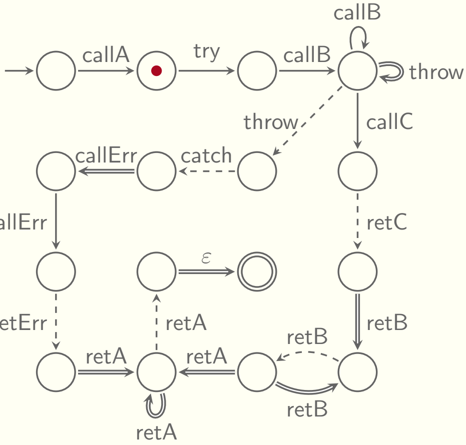
	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callA $\perp$



	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

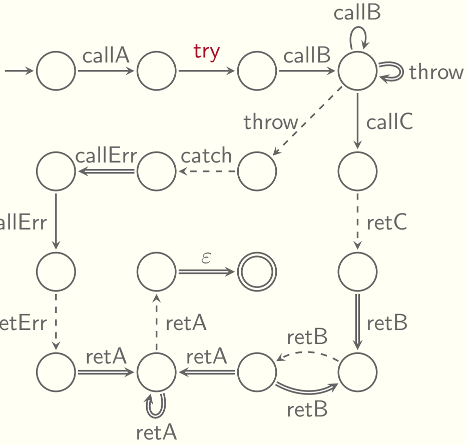


Stack

callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

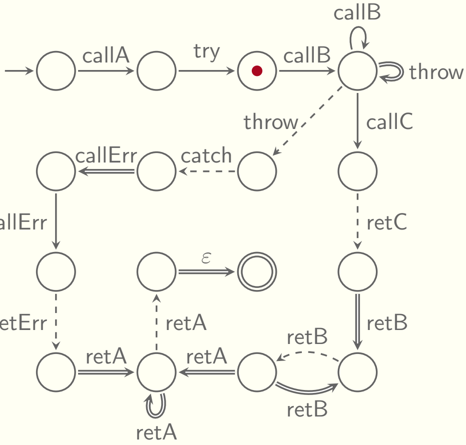
try callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

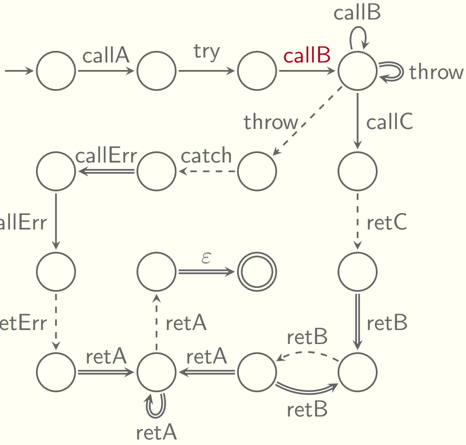
Stack

try callA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

callB try callA $\perp$

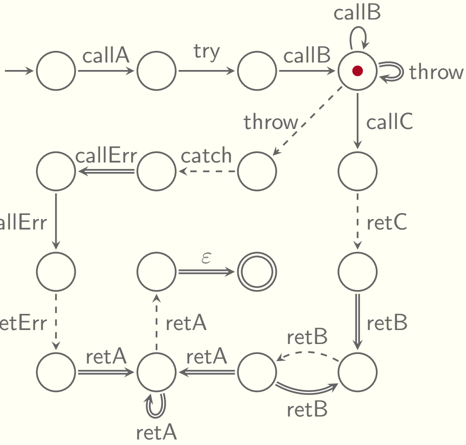
Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

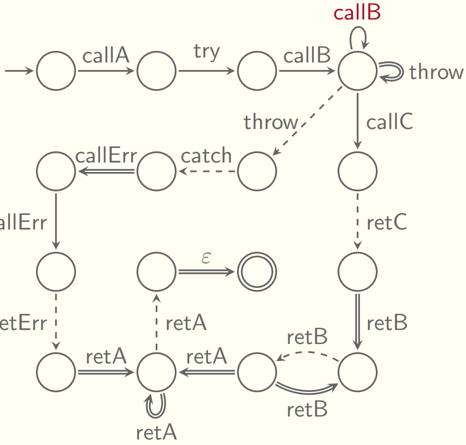
callB try callA $\perp$



Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

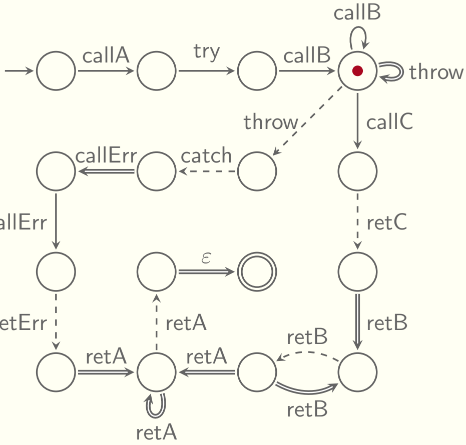
callB callB try callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callB callB try callA $\perp$



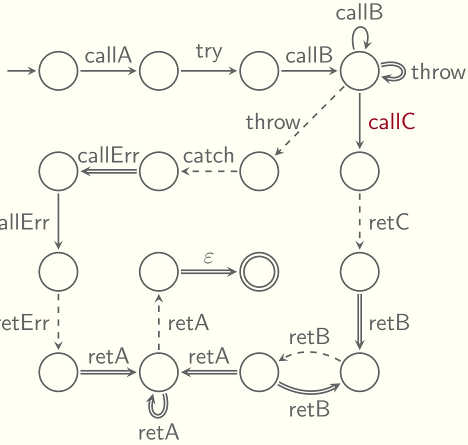
Example of Modelization

skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callC callB callB try callA $\perp$

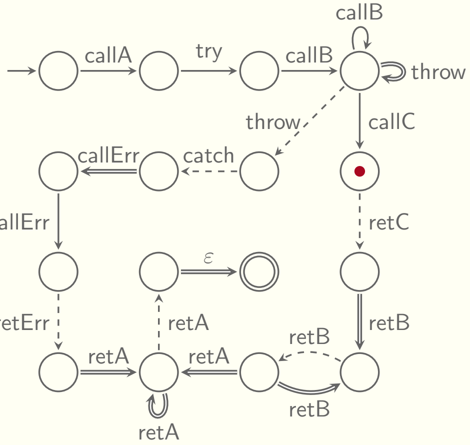


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callC callB callB try callA $\perp$

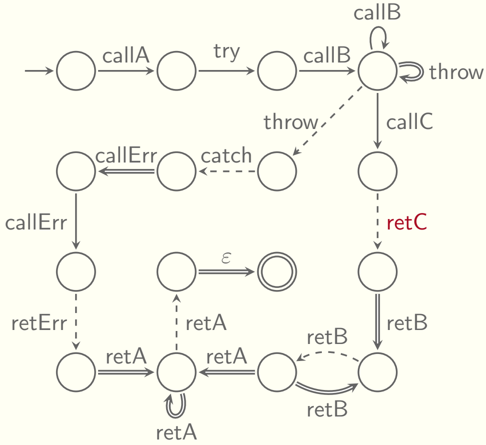


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

retC callB callB try callA $\perp$

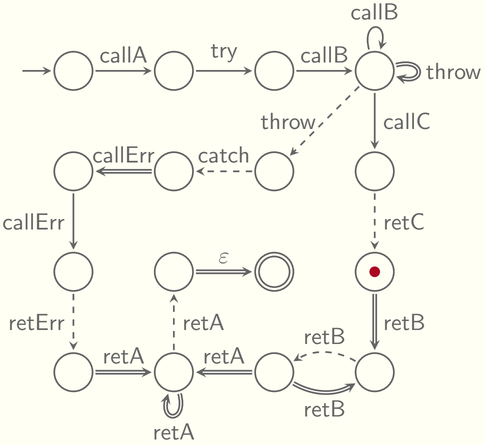


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

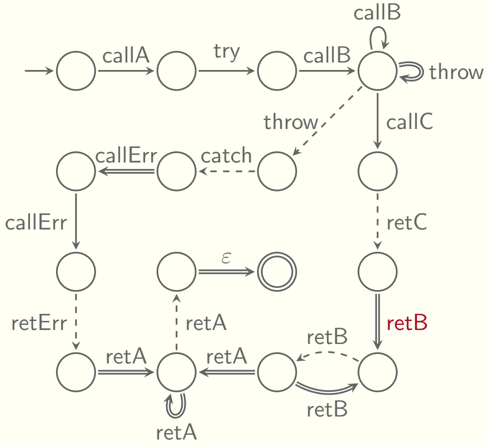
retC callB callB try callA $\perp$



	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

~~retC~~ callB callB try callA $\perp$

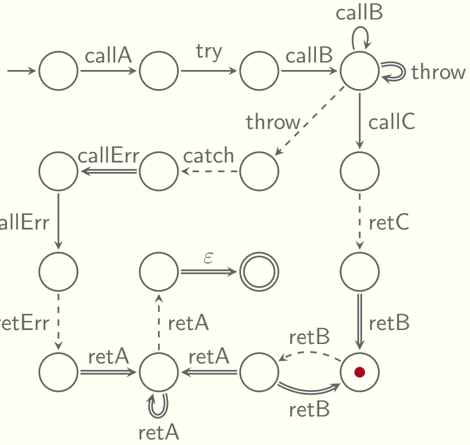


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callB callB try callA $\perp$



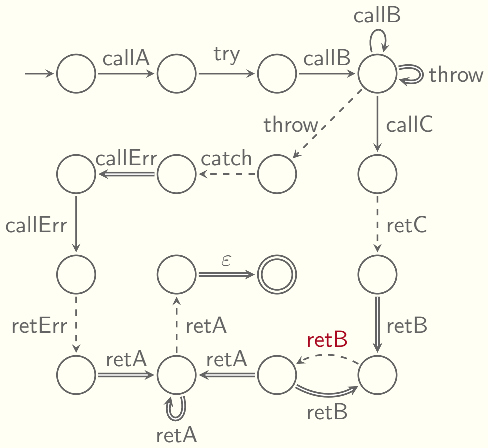
Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

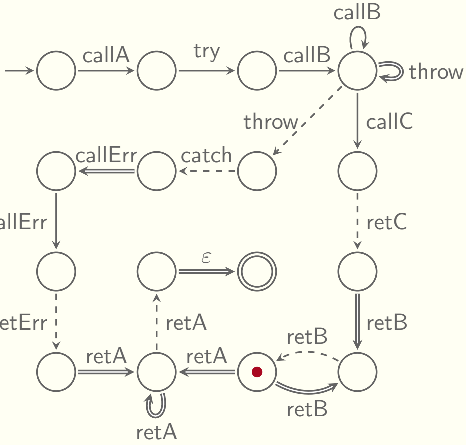
retB callB try callA $\perp$



Example of Modelization

skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

retB callB try callA $\perp$

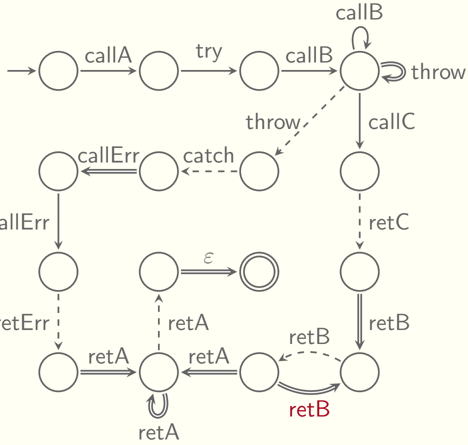
Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

~~retB~~ callB try callA $\perp$



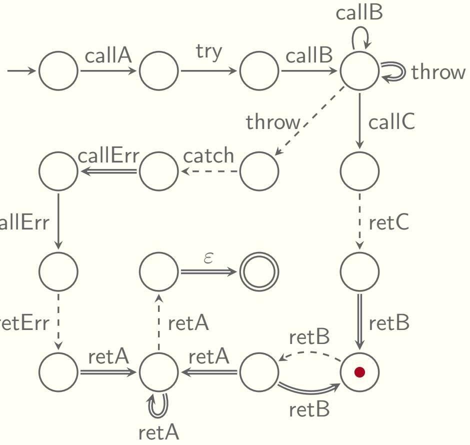
Example of Modelization

⏩ skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

callB try callA $\perp$

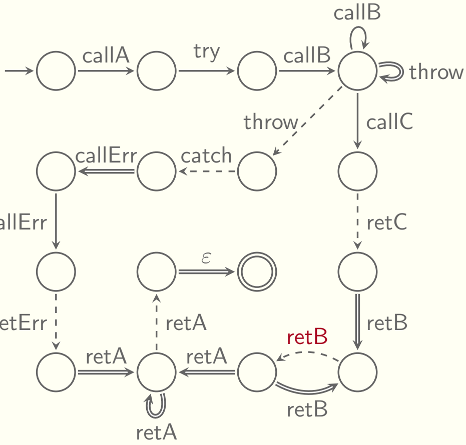


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

retB try callA $\perp$

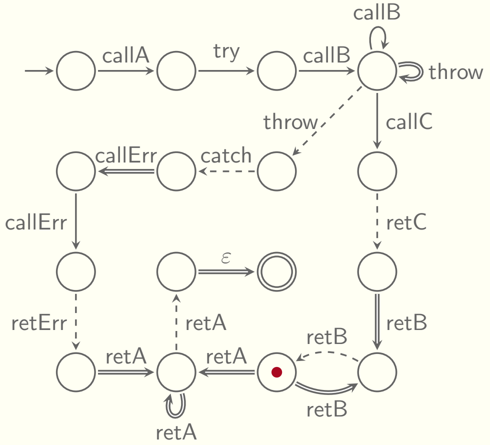


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

retB try callA $\perp$

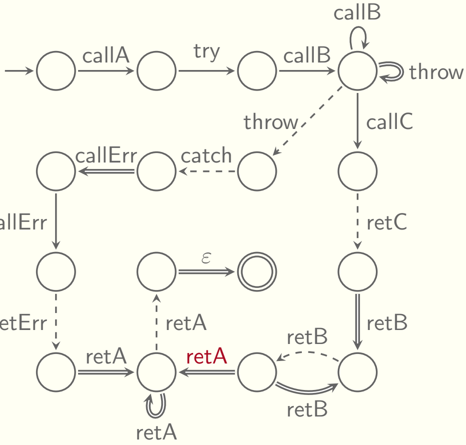


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

~~retB~~ try callA $\perp$

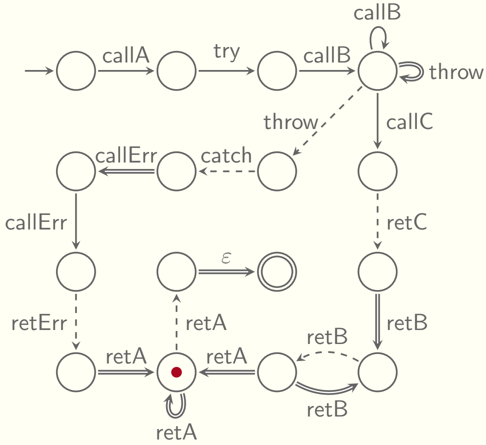


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

try callA $\perp$

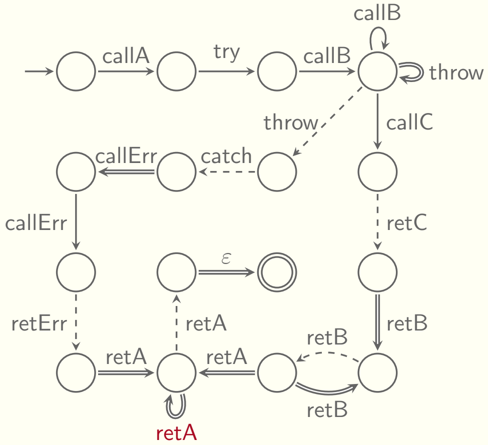


Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

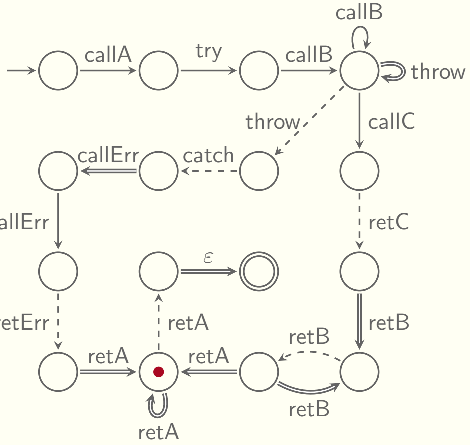
Stack

~~try~~ callA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

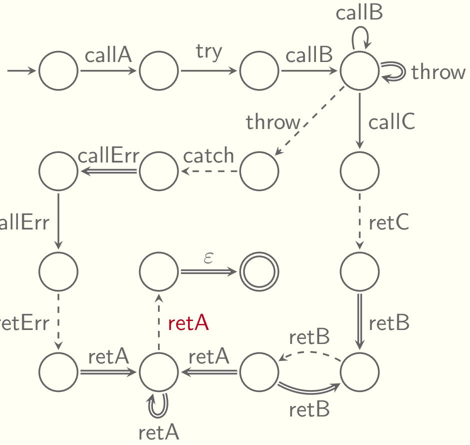


Stack

callA $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

retA $\perp$

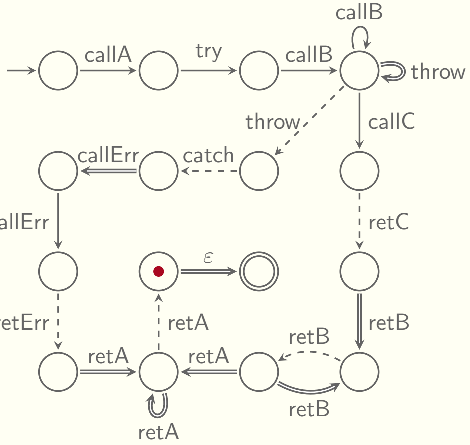
Example of Modelization

skip

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

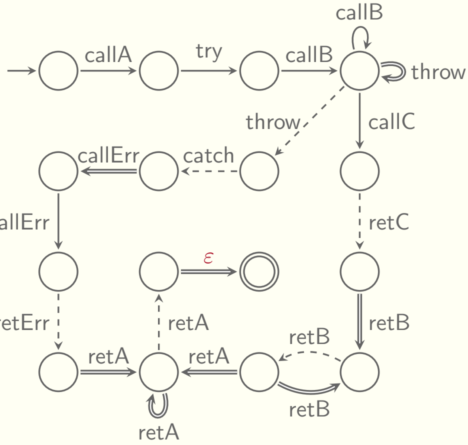
Stack

retA $\perp$



Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$



Stack

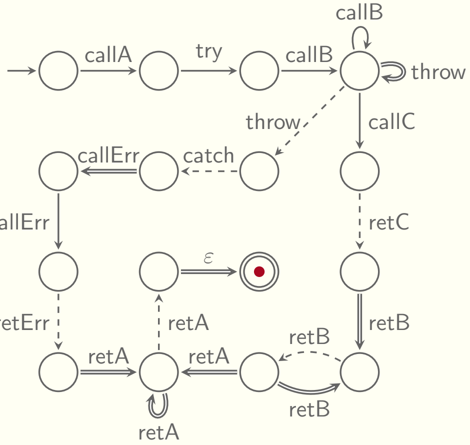
~~retA~~ $\perp$

Example of Modelization

	call	*ret*	*try*	*throw*	*catch*	ε
call	$\lessdot$	$\doteq$	$\lessdot$	$\gtrdot$	-	$\gtrdot$
ret	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
try	$\lessdot$	$\gtrdot$	$\lessdot$	$\doteq$	-	$\gtrdot$
throw	-	-	-	-	$\doteq$	$\gtrdot$
catch	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	-	$\gtrdot$
ε	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$

Stack

$\perp$



Contributions

❶ Characterization by syntactic congruence

L is an OPL $\iff$ $\equiv_L$ has a finite index

❷ Deciding inclusion by synthesizing membership queries: $S \subseteq_{finite} A$

$A \subseteq B$ $\iff$ $\forall w \in S,\ w \in B$

Contributions

❶ Characterization by syntactic congruence

L is an OPL $\iff$ $\equiv_L$ has a finite index

❷ Deciding inclusion by synthesizing membership queries: $S \subseteq_{finite} A$

$A \subseteq B \iff \forall w \in S,\ w \in B$

Complexity[6,7]

Inclusion and equivalence is $\textsc{ExpTime}$-Complete for both OPL and VPL.

[6] R. Alur, P. Madhusudan. *Visibly pushdown languages*. STOC 2004

[7] V. Lonati et al. *OPLs: Their automata-theoretic and logic characterization*. SICOMP 44, 2015

Characterization

Syntactic congruence for REG

$$x \equiv_L y \iff \forall u, v, (uxv \in L \iff uyv \in L)$$

- finite index: $\Sigma^* / \equiv_L$ finite
- language saturation: $x \equiv_L y \implies (x \in L \iff y \in L)$
- monotonic: $x \equiv_L y \implies uxv \equiv_L uyv$

Characterization

Syntactic congruence for REG

$$x \equiv_L y \iff \forall u, v, (uxv \in L \iff uyv \in L)$$

- finite index: $\Sigma^* / \equiv_L$ finite
- language saturation: $x \equiv_L y \implies (x \in L \iff y \in L)$
- monotonic: $x \equiv_L y \implies uxv \equiv_L uyv$

Chain Monotonic relations

A relation $\bowtie$ over $\widehat{\Sigma}^*$ is chain-monotonic when:

$$x \bowtie y \implies u(u_0(x)v_0)v \bowtie u(u_0(y)v_0)v$$

where:

$u_0(x)^{\triangleleft}, u_0(y)^{\triangleleft} \in \widehat{\Sigma}^*_{\geq}$ $\quad$ $(x)^{\triangleright} v_0, (y)^{\triangleright} v_0 \in \widehat{\Sigma}^*_{\leq}$ $\quad$ ${}^{u}[u_0 x v_0]^{v}$ and ${}^{u}[u_0 y v_0]^{v}$

Characterization

▸▸ proof

Syntactic congruence for REG

$$x \equiv_L y \iff \forall u, v, (uxv \in L \iff uyv \in L)$$

- finite index: $\Sigma^* / \equiv_L$ finite
- language saturation: $x \equiv_L y \implies (x \in L \iff y \in L)$
- monotonic: $x \equiv_L y \implies uxv \equiv_L uyv$

Syntactic congruence for OPL

$$x \equiv_L y \iff x \equiv_{\text{chain}} y \wedge \begin{cases} \forall u, v, u_0, v_0 \in \widehat{\Sigma}^*, \left(u_0 x^{\lessdot} \in \widehat{\Sigma}^*_{\gtrdot} \wedge x^{\gtrdot} v_0 \in \widehat{\Sigma}^*_{\lessdot} \wedge {}^u[u_0 x v_0]^v\right) \\ \qquad \implies \left(u u_0 x v_0 v \in L \iff u u_0 y v_0 v \in L\right) \end{cases}$$

- finite index: $\Sigma^* / \equiv_L$ finite
- language saturation: $x \equiv_L y \implies (x \in L \iff y \in L)$
- chain monotonic: $x \equiv_L y \implies u u_0 x v_0 v \equiv_L u u_0 y v_0 v$ when $u_0 (x)^{\lessdot} \in \widehat{\Sigma}^*_{\gtrdot}$ and $(x)^{\gtrdot} v_0 \in \widehat{\Sigma}^*_{\lessdot}$ and ${}^u[u_0 x v_0]^v$

Deciding Inclusion

With Complementation

$$A \subseteq B \iff A \cap (\neg B) = \varnothing$$

With Antichains[8]

$$A \subseteq B \iff A/{\equiv_B} \subseteq B$$

[8] M. De Wulf et al. *Antichains: A New Algorithm for Checking Universality of Finite Automata*. CAV 2006

Deciding Inclusion

With Complementation

$$A \subseteq B \iff A \cap (\neg B) = \varnothing$$

With Antichains

$$A \subseteq B \iff A/{\equiv_B} \subseteq B$$

Language Abstraction[9]

- well quasi-order
- language saturation
- monotonic

[9] P. Ganty et al. *Complete Abstractions for Checking Language Inclusion*. ACM Trans. 22, 2021

Deciding Inclusion

With Complementation

$$A \subseteq B \iff A \cap (\neg B) = \varnothing$$

With Antichains

$$A \subseteq B \iff A/{\equiv_B} \subseteq B$$

Language Abstraction

- well quasi-order
- language saturation
- monotonic

Key Argument

$$\begin{aligned} & A \nsubseteq B \\ \iff & \exists x,\ x \in A \wedge x \notin B \\ \iff & \forall y,\ x \equiv_B y \implies y \in A \wedge y \notin B \end{aligned}$$

Iterative computation of $A/\equiv_B$

Fixpoint computation for REG

$$(q, u) \rightsquigarrow_A (q', \varepsilon) \iff [u]_{\equiv_B} \in \vec{X}_{q,q'}$$

Fixpoint computation for OPL

Iterative computation of $A/{\equiv_B}$

Fixpoint computation for REG

$$(q, u) \rightsquigarrow_A (q', \varepsilon) \iff [u]_{\equiv_B} \in \vec{X}_{q,q'}$$

- $q = q' \implies [\varepsilon]_{\equiv_B} \in \vec{X}_{q,q'}$
- $\left(\begin{array}{c} [u]_{\equiv_B} \in \vec{X}_{q,p}, \quad [v]_{\equiv_B} \in \vec{X}_{p',q'} \\ ((p, a), (p', \varepsilon)) \in \Delta \end{array} \right) \implies [uav]_{\equiv_B} \in \vec{X}_{q,q'}$

monotonicity

Fixpoint computation for OPL

Iterative computation of $A/{\equiv_B}$

Fixpoint computation for REG

$$(q, u) \rightsquigarrow_A (q', \varepsilon) \iff [u]_{\equiv_B} \in \vec{X}_{q,q'}$$

Fixpoint computation for OPL

$$(q, uc, \langle a, s\rangle\bot) \rightsquigarrow (q', c, \langle b, s\rangle\bot) \iff [u] \in \vec{X}^{a,b,c}_{q,q'}$$

Iterative computation of $A/{\equiv_B}$

Fixpoint computation for REG

$$(q, u) \rightsquigarrow_A (q', \varepsilon) \iff [u]_{\equiv_B} \in \vec{X}_{q,q'}$$

Fixpoint computation for OPL

$$(q, uc, \langle a, s\rangle\bot) \rightsquigarrow (q', c, \langle b, s\rangle\bot) \iff [u] \in \vec{X}^{a,b,c}_{q,q'}$$

- $q = q' \implies [\varepsilon] \in \vec{X}^{a,b,c}_{q,q'}$
- $\left(\begin{array}{c} [u] \in \vec{X}^{a,a',b'}_{q,p}, \quad [v] \in \vec{X}^{b',b,c}_{p',q'} \\ (p, \langle a', s\rangle\bot) \overset{b'}{\dashrightarrow} (p', \langle b', s\rangle\bot) \end{array}\right) \implies [ub'v] \in \vec{X}^{a,b,c}_{q,q'}$ chain-monotonicity
- $\left(\begin{array}{c} [u] \in \vec{X}^{a,b,b'}_{q,p}, \quad [v] \in \vec{X}^{b',c',c}_{p',q'}, \quad b \lessdot b' \\ (p, \bot) \xrightarrow{b'} (p', \langle b', p\rangle\bot),\ (q', \langle c', p\rangle\bot) \overset{c}{\Longrightarrow} (q, \bot) \end{array}\right) \implies [ub'v] \in \vec{X}^{a,b,c}_{q,q'}$

Conclusion

Contributions

1. Characterization by syntactic congruence
2. Deciding inclusion with membership queries

Future works

- Implementation
- Weighted / Omega OPL

Conclusion

Contributions

1. Characterization by syntactic congruence
2. Deciding inclusion with membership queries

Future works

- Implementation
- Weighted / Omega OPL

[1] T. A. Henzinger, P. Kebis, N. Mazzocchi and N. E. Saraç
Regular Methods for Operator Precedence Languages
In *ICALP* proceedings 2023

[2] V. Lonati, D. Mandrioli, F. Panella, and M. Pradella
Operator precedence languages: Their automata-theoretic and logic characterization
In *SICOMP* 44, 2015

[3] P. Ganty, F. Ranzato and P. Valero
Complete Abstractions for Checking Language Inclusion
ACM Trans. 22, 2021

[4] M. De Wulf, L. Doyen, T. A. Henzinger and J.-F. Raskin:
Antichains: A New Algorithm for Checking Universality of Finite Automata
In *CAV* proceedings 2006

Thank you

Appendix

Sketch of Proof

back

From $\equiv_L$ to OPL

Let $L \subseteq \widehat{\Sigma}^*$ such that $\equiv_L$ has a finite index. We construct $A = (Q, q_0, F, \Delta)$ where:

- $Q = \{([u], [v]) : u, v \in \widehat{\Sigma}^*\}$
- $q_0 = ([\varepsilon], [\varepsilon])$
- $F = \{([\varepsilon], [w]) : w \in L\}$
- $\Delta(([u], [v]), a, \langle b, ([u'], [v'])\rangle\theta) \rightarrow (([uva], [\varepsilon]), \varepsilon, <\langle a, ([u'], [v'])\rangle\theta)$ $\quad b \doteq a$
- $\Delta(([u], [v]), a, \langle b, ([u'], [v'])\rangle\theta) \rightarrow (([u'], [v'uv]), a, \theta)$ $\quad b \gtrdot a$
- $\Delta(q, a, \langle b, q'\rangle\theta) \rightarrow (([a], [\varepsilon]), \varepsilon, \langle a, q\rangle\langle b, q'\rangle\theta)$ $\quad b \lessdot a$

From OPL to $\equiv_L$

Sketch of Proof

▸▸ back

From $\equiv_L$ to OPL

From OPL to $\equiv_L$

Let L be a OPL reconized by $A = (Q, q_0, F, \Delta)$. For all $w \in \widehat{\Sigma}^*$ we define:

- $f_w(q) = \{q' \in Q : \exists\theta, (q, w\star, \bot) \rightsquigarrow (q', \star, \theta)\}$
- $\Phi_w(q) = \{\theta : \exists q', (q, w\star, \bot) \rightsquigarrow (q', \star, \theta)\}$
- $g_w(q_1, q_2) = \{p_w : \exists\theta, (q_2, \varepsilon, \theta) \rightsquigarrow (p_w, \varepsilon, \bot) \wedge \theta \in \Phi_w(q_1)\}$

$$x \equiv_A y \iff \big(x \equiv_{\text{chain}} y\big) \wedge \big(f_x = f_y\big) \wedge \big(g_x = g_y\big) \wedge \Big(\bigwedge_{q \in Q} (\Phi_x(q))^\top = (\Phi_y(q))^\top\Big)$$

- $x \equiv_A y \implies x \equiv_L y$
- $\widehat{\Sigma}^* / \equiv_A$ is finite