Slide 1

A formal analysis framework for plan execution languages

Gilles Dowek, César Muñoz, and Corina Pasareanu

Slide 2

I. Plan execution languages

PLEXIL, plans and plan execution

A language used to express the mission of space vehicles

Drive(1); Right(120); Drive(1); Right(120); Drive(1)

Several processes executed in parallel: drive, heat, ...

Slide 3

Plans hand written or generated (planning, optimization)

Autonomous reaction to external events

when T $\leq 0^{\circ}$ do heat

Idle process waiting for activation

Close to the (semi-formal) languages used to monitor vehicles

Plan execution languages and synchronous languages

Synchronous execution

Not very different from Esterel-like synchronous languages

Slide 4

Yet some differences :

- tree like organization of processes : processes create children processes
- strong emphasis on events occurring in the (asynchronous) external world and that trigger processes

PLEXIL-like languages in a nutshell

Slide 5

A program (plan) is a tree

Each process has a status Waiting, Executing, Finishing, ...

All nodes have a start, a repeat until and an end condition (boolean expression)

Leaves are commands Drive(1) and assignments `x := 1`

Internal nodes contribute only to the “control flow” : create other nodes

Expressions may include variables and lookups

A formal semantics

Slide 6

Important to know what programs do

For simulation

For verification

To formally prove (PVS) properties of the language itself (compositionality, determinism, ...)

A quite confuse (?) informal description due to the complexity of the language itself

Slide 7

II. Small step operational semantics

Slide 8

I transforms to V in several small execution steps

First define a relation $\longrightarrow$ describing small execution steps

Then use $\longrightarrow$ to define the big execution step relation $\hookrightarrow$

An example

Slide 9

$$2 * 4 \longrightarrow 8$$

$$8 + 3 \longrightarrow 11$$

An example

From $\longrightarrow$ define a relation $\rhd$ that allows to reduce in subterm

E.g. from

$$2 * 4 \longrightarrow 8$$

Slide 10

deduce

$$(2 * 4)+3 \rhd 8+3$$

Then define $\rhd^*$ (several steps one after another)

Then define $a \rhd^{\downarrow} b$ as $a \rhd^* b$ and b irreducible

And $\hookrightarrow$ as $\rhd^{\downarrow}$

An example

$$(2 * 4) + 3 \rhd 8 + 3 \rhd 11$$

Slide 11

Hence

$$(2 * 4) + 3 \hookrightarrow 11$$

Operations of relations

Slide 12

From $\longrightarrow$ to $\rhd$: reduce a subexpression

From $\rhd$ to $\rhd^*$, $\rhd^\downarrow$, ...

Slide 13

III. The case of process languages

Slide 14

Not always termination, not always an output value

But a behavior: a sequence of states of the interacting atomic processes

Small steps operational semantics is appropriate: behaviors are (finite or infinite) reduction sequences

Slide 15

The Chemical reaction model of Banâtre and Le Métayer

The state of the interacting processes is described by a finite multiset of atomic process states

If four atomic processes that are in state A, B, C and C the solution is the multiset $\{A, B, C, C\}$ written $(A \mid B \mid C \mid C)$

small steps are described by rewrite rules *e.g.*

$$CH_4 \mid O_2 \mid O_2 \longrightarrow CO_2 \mid H_2O \mid H_2O$$

Slide 16

An atomic left hand side describes a spontaneous evolution

$$CO_3H^- \longrightarrow CO_2 + OH^-$$

A compound left hand side describes an interaction

$$CH_4 \mid O_2 \mid O_2 \longrightarrow CO_2 \mid H_2O \mid H_2O$$

An puzzle for family meetings:

Take the rule

$$n \mid p \longrightarrow n \text{ if } p \text{ is a multiple of } n$$

What is the reduced form of $2 \mid 3 \mid 4 \mid \ldots \mid 100$?

Modeling memory

In small step operational semantics assignment is often a burden (explicit memory)

$$\langle [x = 3, y = 6], (x := 4; s) \rangle \longrightarrow \langle [x = 4, y = 6], s \rangle$$

Slide 17

In the Chemical model memory (or each memory cell) is a process

$$Val(x, 3) \mid Val(y, 6) \mid x := 4 \longrightarrow Val(x, 4) \mid Val(y, 6)$$

Modeling external events

Natural processes are processes

$$Position(n) \mid Advance(p) \longrightarrow Position(n + p)$$

Slide 18

$$Advance(p) \longrightarrow Engine(p)$$

$$Position(n) \mid Engine(p{+}1) \longrightarrow Position(n{+}1) \mid Engine(p)$$

$$Engine(0) \longrightarrow$$

Slide 19

III. Synchrony

Slide 20

Do we need a big step execution relation?

Not always termination, not always an output value: the trip itself is the goal of the trip

All we need is to define one-tick execution steps and sequences

But still several ways to relate the evolution of subprocesses and that of the global process

From the atomic step relation to micro step relation

From $2 * 4 \longrightarrow 8$

deduce $(2 * 4)+3 \rhd 8+3$

In a similar way, from $A \longrightarrow B$

deduce $\Gamma \mid A \rhd \Gamma \mid B$

Slide 21

One interaction at a time, no parallelism

E.g. if $P \longrightarrow Q$ and $P' \longrightarrow Q'$

$$(P \mid P') \rhd (Q \mid P') \rhd (Q \mid Q')$$

The relation $\rhd$ is the asynchronous evolution relation

Synchrony: the easy case

A and all B_1, ..., B_n its $\longrightarrow$-reducible subprocesses

If there exists C such that

$$A = (B_1 \mid ... \mid B_n \mid C)$$

Slide 22

then we say that the reducible subprocesses do not overlap

In this case we can reduce A to

$$(B'_1 \mid ... \mid B'_n \mid C)$$

Defines a relation $\longrightarrow^{\|}$

Synchrony: the easy case

Slide 23

$$Val(x,2) \mid Val(y,4) \mid x := 0 \mid y := y + 1$$

$$\longrightarrow^{\|}$$

$$Val(x,0) \mid Val(y,5)$$

But if they overlap?

$$Val(x,0) \mid x := 1 \mid x := 2$$

(1) No reduction (or Error)

(2) Priority

Slide 24

A priority relation $\prec$: strict partial order on processes

A reducible subprocess is eligible if it is $\succ$ all the reducible subprocesses it overlaps with

Eligible subprocesses do not overlap

Reduce eligible subprocesses in parallel: another relation $\longrightarrow^{\prec}$

Active parts of reducible expressions

$Val(x,2) \mid Val(y,4) \mid Val(z,6) \mid y := x+1 \mid z := x+3$

Slide 25

irreducible

because

$Val(x,2) \mid Val(y,4) \mid y := x+1 \longrightarrow Val(x,2) \mid Val(y,3)$

but not

$$Val(y,4) \mid y := x+1 \longrightarrow Val(y,3)$$

Active parts of reducible expressions

However in

$Val(x,2) \mid Val(y,4) \mid y := x+1 \longrightarrow Val(x,2) \mid Val(y,3)$

Slide 26

The atomic process $Val(x,2)$ is kept in the right hand side

Needed to reduce but not modified

Active part of the reducible expression

The micro relation: $\Rightarrow$ is $\longrightarrow^{\prec}$

Slide 27

IV. External events

Slide 28

When an external event occurs: $T < 0^\circ$, ...

Some processes start evolving, this evolution must terminate and reach quiescence

Then the system is quite

Until another external event may occur

Two external events never happen simultaneously

i.e. are never processed simultaneously

From the micro step relation to the quiescence relation

Slide 29

The run to quiescence relation is $\Rightarrow_{\downarrow}$

From the quiescence relation to the macro step relation and the execution relation

Slide 30

One macro-step : one event followed by run to quiescence

An *a priori* scenario of external events

The execution relation = iteration of macro step relation

Slide 31

V. What have we been doing ?

Slide 32

Starting from one relation (atomic) we have defined four others

The operations to build relations from relations are a few

$\longrightarrow^n$ n step (macro to execution)

$\longrightarrow^*$ some number of steps

$\longrightarrow^\downarrow$ to irreducible form (micro to quiescence)

$\longrightarrow'$; $\longrightarrow$ (quiescence to macro)

$\longrightarrow_s$ parallel reduction of s-subexpressions (atomic to micro)

The properties of the languages are consequences of properties of these operations

Slide 33

VI. Formally proving properties

Determinism

Determinism: $a \longrightarrow a'$ and $a \longrightarrow a''$ implies $a' = a''$

Slide 34

If $\longrightarrow$ is deterministic, then $\longrightarrow^n$ is deterministic

If $\longrightarrow$ is deterministic, then $\longrightarrow^{\downarrow}$ is deterministic

If $\longrightarrow$ is deterministic, and s is a strategy for $\longrightarrow$, then $\longrightarrow^s$ is deterministic

Notice that $\longrightarrow^*$, for instance, is not

Compositionality

$\longrightarrow$ relation on a set T and $\mid$ binary operation on T

$a \longrightarrow a'$ and $b \longrightarrow b'$ implies $(a \mid b) \longrightarrow (a' \mid b')$

Slide 35

Does not hold in general, but for specific sets A: $\langle a, b \rangle \in A$

Much more interesting because holds only if we have some compatibility between A and $\longrightarrow$...

Some outputs

Better known properties of this language

Participation to the design of the language

Slide 36

Propositions of alternatives (what if run to quiescence does not terminate ?)

A framework to help the design of plan execution languages