Céu: A Reactive language for Wireless Sensor Networks

LabLua
PUC-Rio

Francisco Sant'Anna
(fsantanna@inf.puc-rio.br)

Overview of Céu

- Reactive
  - environment in control: *events*
- Imperative
  - sequences, loops, assignments
- Concurrent
  - multiple lines of execution: *trails*
- Synchronous
  - trails synchronize at each external event
- Deterministic
  - always yields the same outcome for a given timeline

Céu and TinyOS

- Céu is implemented on top of nesC/TinyOS
  - TinyOS **events** -> Céu **external input events**
    - Radio.receive -> Radio_recv
  - TinyOS **commands** -> Céu **external output events**
    - Leds.set -> Leds_set

”Sum” Example

```
0 => acc;

(
   ( (~Radio_recv,acc)->add ~> acc )*
||
   ( ~acc ~> Leds_set )*
)
```

Céu Syntax

```
e ::=  e ; e              (sequence)
    |  e ? e : e          (conditional)
    |  e || e             (parallel or)
    |  e && e             (parallel and)
    |  e*                 (loop)
    |  e^                 (loop break)
    |  { e }              (scope block)
    |  @{ e }             (asynchronous block)
    |  ID                 (variable read)
    |  e => ID            (variable attribution)
    |  ~ID                (event await)
    |  ~TIME              (time await)
    |  ~> ID              (event trigger, 0 params)
    |  e ~> ID            (event trigger, 1 param)
    |  e -> ID            (operation call, 1 param)
    |  (e,e) -> ID        (operation call, N params)
```

Synchronous Execution

- Time: discrete sequence of external input events
  - sequence: only one event reacts at a time
  - discrete: a reaction executes in bounded time

1) Await next event and awake awaiting trails.
2) Active trails execute without interruption. (*Reaction Chain)*
3) Goto 1.

Blink

```
(
      ( ~250ms  ; ~>Leds_led0Toggle )*
  ||
      ( ~500ms  ; ~>Leds_led1Toggle )*
  ||
      ( ~1000ms ; ~>Leds_led2Toggle )*
)
```

Radio - Init

```
(
    ~1s
 ||
    ( ~>Radio_start => radio_err ?
        ~> radio_err
    :
        ( ~Radio_startDone => radio_err ?
            ~> radio_err
        :
            0^
        )
    ) ;
    ~~
)*
```

Radio - Using

```
(
    RADIO_START() -- copy init
 ||
    (~radio_err ; ~>Leds_led0Toggle)*
) ;

...                 -- use the radio
```

www.lua.inf.puc-rio.br/~francisco/ufrj/tarefas.html

Temporal Analysis

- Bounded execution
  - *Loops:*
    - `(1)*` `(~A||v)*` `(v?1:~A)*`
    - `(~A)*` `(~A&&v)*` `(~A?1:0)*`
  - *Operations:* *`(1,2) -> add`*
- Determinism
  1) `1=>v && 2=>v` `(vs: ~A=>v && ~B=>v)`
  2) `1~>A && 2~>B`
  3) `(1 || 2) => v`
  4) `(1^ && 2^)* => v`

Céu -> DFA

- Céu programs are converted to DFAs
- Céu is static
- Detects:
  - concurrent access to variables or events
  - concurrent *par/or* termination
  - concurrent loop escape
  - unreachable expressions

DFA example

```
    (~A ; ~A ; 1=>v)*
||
    (~A ; ~A ; ~A ; v)*
```