ReverCSP: Time-travelling in CSP computations

Carlos Galindo[1] Naoki Nishida[2] Josep Silva[1] Salvador Tamarit[1]

[1] Departamento de Sistemas Informàticos y Computación
Universitat Politècnica de València, Spain

[2] Graduate School of Informatics
Nagoya University, Japan

12th International Conference on Reversible Computation
July 9th, 2020

Motivation

- Communicating Sequential Processes (CSP): a formal language to describe concurrency.
- Uses: security, livelock analysis, deadlock analysis...
- Debugging CSP: errors cannot be easily reproduced, have to be logged/traced.

CSP Syntax

$$
\begin{array}{lcll}
Proc & ::= & Q & \text{(process call)} \\
 & | & x \rightarrow Proc & \text{(prefixing)} \\
 & | & c?u \rightarrow Proc & \text{(input)} \\
 & | & c!u \rightarrow Proc & \text{(output)} \\
 & | & Proc_1 \sqcap Proc_2 & \text{(internal choice)} \\
 & | & Proc_1 \Box Proc_2 & \text{(external choice)} \\
 & | & Proc_1 \,|||\, Proc_2 & \text{(interleaving)} \\
 & | & Proc_1 \underset{\{\mathtt{x}\}}{||} Proc_2 & \text{(synchronized parallelism)} \\
 & | & Proc_1 \,;\, Proc_2 & \text{(sequential composition)} \\
 & | & Proc \backslash X & \text{(hiding)} \\
 & | & Proc[\![f]\!] & \text{(renaming)} \\
 & | & SKIP & \text{(skip)} \\
 & | & STOP & \text{(stop)}
\end{array}
$$

CSP Syntax

$$
\begin{array}{lcll}
Proc & ::= & Q & \text{(process call)} \\
 & | & x \rightarrow Proc & \text{(prefixing)} \\
 & | & c?u \rightarrow Proc & \text{(input)} \\
 & | & c!u \rightarrow Proc & \text{(output)} \\
 & | & Proc_1 \ \sqcap \ Proc_2 & \text{(internal choice)} \\
 & | & Proc_1 \ \Box \ Proc_2 & \text{(external choice)} \\
 & | & Proc_1 \ ||| \ Proc_2 & \text{(interleaving)} \\
 & | & Proc_1 \underset{\{\mathtt{x}\}}{||} Proc_2 & \text{(synchronized parallelism)} \\
 & | & Proc_1 \ ; \ Proc_2 & \text{(sequential composition)} \\
 & | & Proc \backslash X & \text{(hiding)} \\
 & | & Proc[\![f]\!] & \text{(renaming)} \\
 & | & SKIP & \text{(skip)} \\
 & | & STOP & \text{(stop)}
\end{array}
$$

CSP Syntax

Example

Example

$$\begin{aligned}
\mathtt{MAIN} &= \mathtt{Q} \underset{\{\mathtt{a}\}}{||} \mathtt{P} \\
\mathtt{Q} &= \mathtt{a} \rightarrow \mathtt{b} \rightarrow \mathtt{SKIP} \\
\mathtt{P} &= \mathtt{R} \underset{\{\mathtt{a}\}}{||} \mathtt{a} \rightarrow (\mathtt{b} \rightarrow \mathtt{SKIP} \sqcap \mathtt{Q}) \\
\mathtt{R} &= \mathtt{a} \rightarrow \mathtt{SKIP}
\end{aligned}$$

- Possible traces: $\{\langle\rangle, \langle \mathtt{a} \rangle, \langle \mathtt{ab} \rangle, \langle \mathtt{abb} \rangle\}$
- $\langle \mathtt{abb} \rangle$: `bb` can be emitted on `Q` and then `P` or vice-versa.
- CSP debugging tools: traces → tracks → R-tracks

CSP Tracks

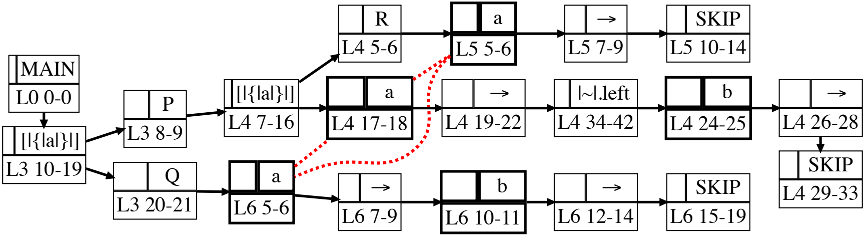
Example

$$\mathtt{MAIN} = \mathtt{Q} \underset{\{\mathtt{a}\}}{||} \mathtt{P}$$

$$\mathtt{Q} = \mathtt{a} \rightarrow \mathtt{b} \rightarrow \mathtt{SKIP}$$

$$\mathtt{P} = \mathtt{R} \underset{\{\mathtt{a}\}}{||} \mathtt{a} \rightarrow (\mathtt{b} \rightarrow \mathtt{SKIP} \sqcap \mathtt{Q})$$

$$\mathtt{R} = \mathtt{a} \rightarrow \mathtt{SKIP}$$



CSP R-tracks

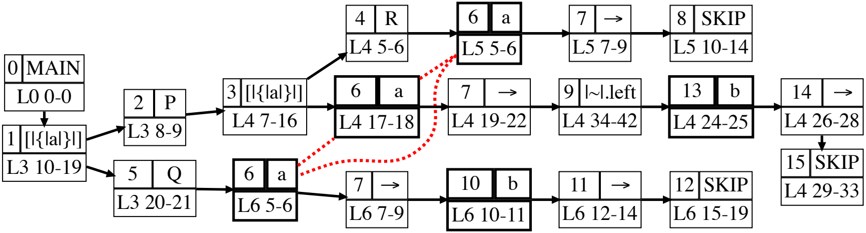
Example

$$\mathtt{MAIN} = \mathtt{Q} \underset{\{\mathtt{a}\}}{||} \mathtt{P}$$

$$\mathtt{Q} = \mathtt{a} \rightarrow \mathtt{b} \rightarrow \mathtt{SKIP}$$

$$\mathtt{P} = \mathtt{R} \underset{\{\mathtt{a}\}}{||} \mathtt{a} \rightarrow (\mathtt{b} \rightarrow \mathtt{SKIP} \sqcap \mathtt{Q})$$

$$\mathtt{R} = \mathtt{a} \rightarrow \mathtt{SKIP}$$



CSP Reversibility with R-tracks

Deterministic reversibility

Replay the original execution, forwards or backwards.
At any state, there is at most one possible execution step in each direction.

Causal-consistent reversibility

Explore any and all executions consistently.

Forward	Only possible if all *causes* of a step have been executed.
Backward	Only possible if all *consequences* of a step have been reversed.

CSP Reversibility with R-tracks

Deterministic reversibility

Replay the original execution, forwards or backwards.
At any state, there is at most one possible execution step in each direction.

Causal-consistent reversibility

Explore any and all executions consistently.

- Forward: Only possible if all *causes* of a step have been executed.
- Backward: Only possible if all *consequences* of a step have been reversed.

CSP R-tracks

Causal consistency

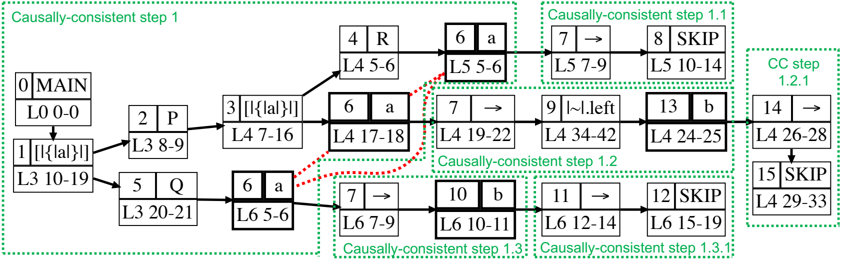
Example

$$\mathtt{MAIN} = \mathtt{Q} \underset{\{\mathtt{a}\}}{||} \mathtt{P}$$

$$\mathtt{Q} = \mathtt{a} \rightarrow \mathtt{b} \rightarrow \mathtt{SKIP}$$

$$\mathtt{P} = \mathtt{R} \underset{\{\mathtt{a}\}}{||} \mathtt{a} \rightarrow (\mathtt{b} \rightarrow \mathtt{SKIP} \sqcap \mathtt{Q})$$

$$\mathtt{R} = \mathtt{a} \rightarrow \mathtt{SKIP}$$



Causally-consistent step 1

0 MAIN L0 0-0

1 [|{|a|}|] L3 10-19

2 P L3 8-9

3 [|{|a|}|] L4 7-16

4 R L4 5-6

6 a L5 5-6

6 a L4 17-18

5 Q L3 20-21

6 a L6 5-6

Causally-consistent step 1.1

7 → L5 7-9

8 SKIP L5 10-14

Causally-consistent step 1.2

7 → L4 19-22

9 |~|.left L4 34-42

13 b L4 24-25

CC step 1.2.1

14 → L4 26-28

15 SKIP L4 29-33

Causally-consistent step 1.3

7 → L6 7-9

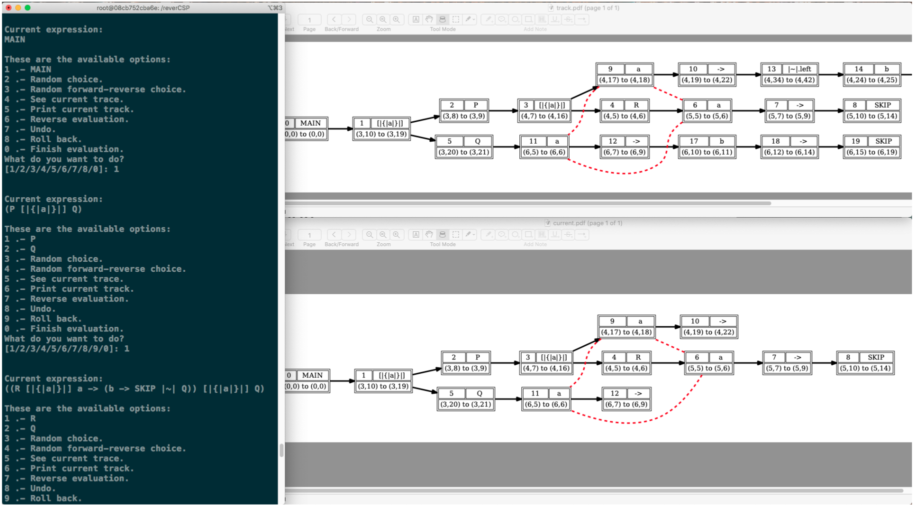
10 b L6 10-11

Causally-consistent step 1.3.1

11 → L6 12-14

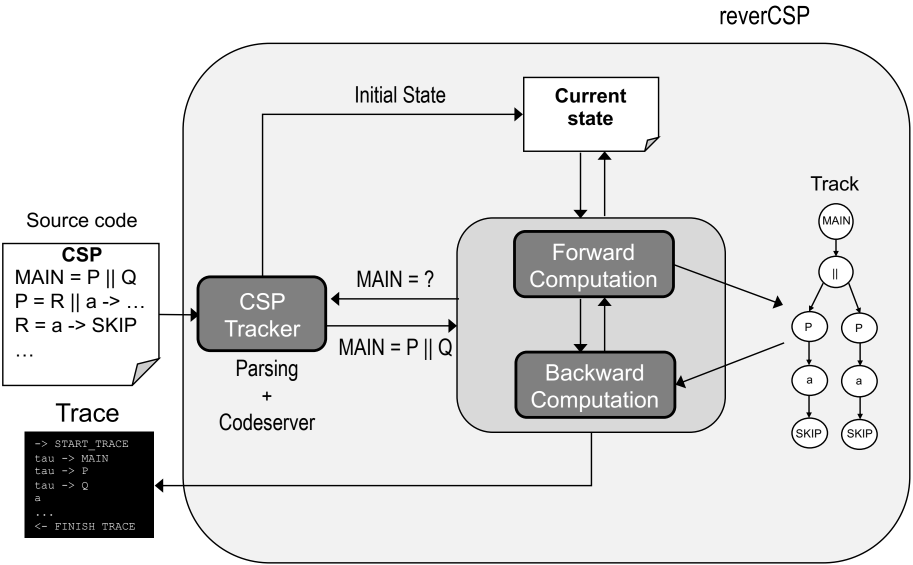
12 SKIP L6 15-19

System demo



```
Current expression:
MAIN

These are the available options:
1 .- MAIN
2 .- Random choice.
3 .- Random forward-reverse choice.
4 .- See current trace.
5 .- Print current track.
6 .- Reverse evaluation.
7 .- Undo.
8 .- Roll back.
0 .- Finish evaluation.
What do you want to do?
[1/2/3/4/5/6/7/8/0]: 1


Current expression:
(P [|{|a|}|] Q)

These are the available options:
1 .- P
2 .- Q
3 .- Random choice.
4 .- Random forward-reverse choice.
5 .- See current trace.
6 .- Print current track.
7 .- Reverse evaluation.
8 .- Undo.
9 .- Roll back.
0 .- Finish evaluation.
What do you want to do?
[1/2/3/4/5/6/7/8/9/0]: 1


Current expression:
((R [|{|a|}|] a -> (b -> SKIP |~| Q)) [|{|a|}|] Q)

These are the available options:
1 .- R
2 .- Q
3 .- Random choice.
4 .- Random forward-reverse choice.
5 .- See current trace.
6 .- Print current track.
7 .- Reverse evaluation.
8 .- Undo.
9 .- Roll back.
```

Architecture



```
CSP
MAIN = P || Q
P = R || a -> …
R = a -> SKIP
…
```

```
-> START_TRACE
tau -> MAIN
tau -> P
tau -> Q
a
...
<- FINISH TRACE
```

Empirical evaluation

Table: Size of the tracks generated with a given runtime

Benchmark	Runtime (ms)	#Nodes	#Edges	Memory Size (KBytes)
ABP.csp	$[_{2208.16}\ 2209.25\ _{2210.34}]$	$[_{1505.61}\ 1506.17\ _{1506.73}]$	$[_{1303.10}\ 1303.63\ _{1304.17}]$	$[_{172.98}\ 173.05\ _{173.11}]$
ATM.csp	$[_{630.17}\ 690.18\ _{750.19}]$	$[_{364.09}\ 405.64\ _{447.19}]$	$[_{300.74}\ 334.67\ _{368.61}]$	$[_{42.61}\ 47.56\ _{52.51}]$
Buses.csp	$[_{126.40}\ 127.19\ _{127.97}]$	$[_{22.00}\ 22.00\ _{22.00}]$	$[_{18.00}\ 18.00\ _{18.00}]$	$[_{2.43}\ 2.43\ _{2.43}]$
CPU.csp	$[_{189.97}\ 190.74\ _{191.51}]$	$[_{87.43}\ 87.76\ _{88.09}]$	$[_{71.23}\ 71.50\ _{71.77}]$	$[_{9.59}\ 9.63\ _{9.67}]$
Disk.csp	$[_{209.07}\ 210.10\ _{211.13}]$	$[_{148.50}\ 148.74\ _{148.98}]$	$[_{123.59}\ 123.78\ _{123.78}]$	$[_{16.72}\ 16.74\ _{16.77}]$
Loop.csp	$[_{2133.02}\ 2133.99\ _{2134.96}]$	$[_{1537.53}\ 1538.34\ _{1539.14}]$	$[_{1230.05}\ 1230.69\ _{1231.35}]$	$[_{191.42}\ 191.53\ _{191.63}]$
Oven.csp	$[_{238.64}\ 241.92\ _{245.20}]$	$[_{157.16}\ 163.37\ _{169.59}]$	$[_{162.68}\ 169.33\ _{175.98}]$	$[_{20.03}\ 20.86\ _{21.69}]$
ProdCons.csp	$[_{2134.59}\ 2135.43\ _{2136.27}]$	$[_{1535.43}\ 1536.09\ _{1536.75}]$	$[_{1228.08}\ 1228.61\ _{1229.15}]$	$[_{189.44}\ 189.53\ _{189.61}]$
ReadWrite.csp	$[_{2148.76}\ 2149.71\ _{2150.65}]$	$[_{1475.85}\ 1476.56\ _{1477.28}]$	$[_{1252.47}\ 1253.34\ _{1254.22}]$	$[_{171.57}\ 171.66\ _{171.76}]$
Traffic.csp	$[_{165.34}\ 166.35\ _{167.36}]$	$[_{61.18}\ 64.37\ _{67.56}]$	$[_{47.73}\ 50.13\ _{52.53}]$	$[_{6.44}\ 6.79\ _{70.30}]$
Average	$[_{1018.41}\ 1025.49\ _{1019.76}]$	$[_{689.478}\ 694.90\ _{700.33}]$	$[_{573.77}\ 578.37\ _{582.98}]$	$[_{115.59}\ 116.72\ _{117.85}]$

Memory usage: $< 144KB/s$

[symmetric 99% confidence intervals]

Source code: https://github.com/tamarit/reverCSP (with benchmarks)

Conclusions

- R-tracks as the extension of tracks with timestamps.
- R-tracks enable reversibility
- Choice between deterministic and causally consistent steps.
- For debugging: track the cause of a bug and easily explore alternative execution paths.
- Backed by formal semantics and proof.
- Freely available online.

  `https://github.com/tamarit/reverCSP`