choreography extraction

luís cruz-filipe

(joint work with kim skak larsen, fabrizio montesi & larisa safina)

department of mathematics and computer science
university of southern denmark

cl seminar
november 13th, 2020

the goal

extract a choreography from an implementation of a distributed system (if possible)

unsatisfactory state-of-the-art

- too complex (encode, apply algorithm, reencode, optimize, reencode, recompute, get unreadable result)
- too complex (super-factorial)
- kind of *ad-hoc* (non-standard languages)

the goal

extract a choreography from an implementation of a distributed system (if possible)

unsatisfactory state-of-the-art

- too complex (encode, apply algorithm, reencode, optimize, reencode, recompute, get unreadable result)
- too complex (super-factorial)
- kind of *ad-hoc* (non-standard languages)

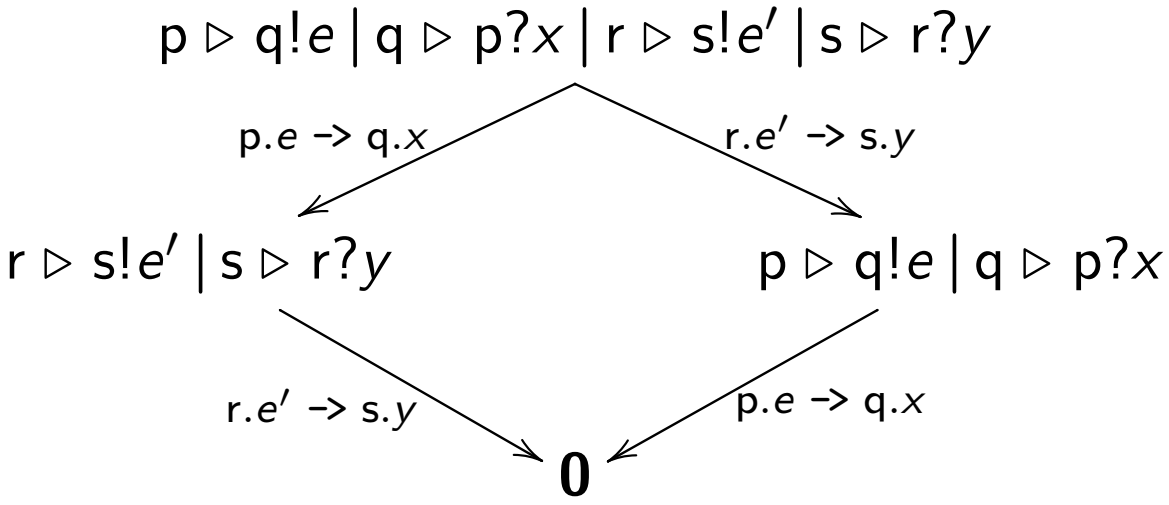
our privileged position

- nice languages for choreographies and processes
- intuitive description of what extraction should be (some kind of adjoint to epp)
- cool idea (symbolic execution)

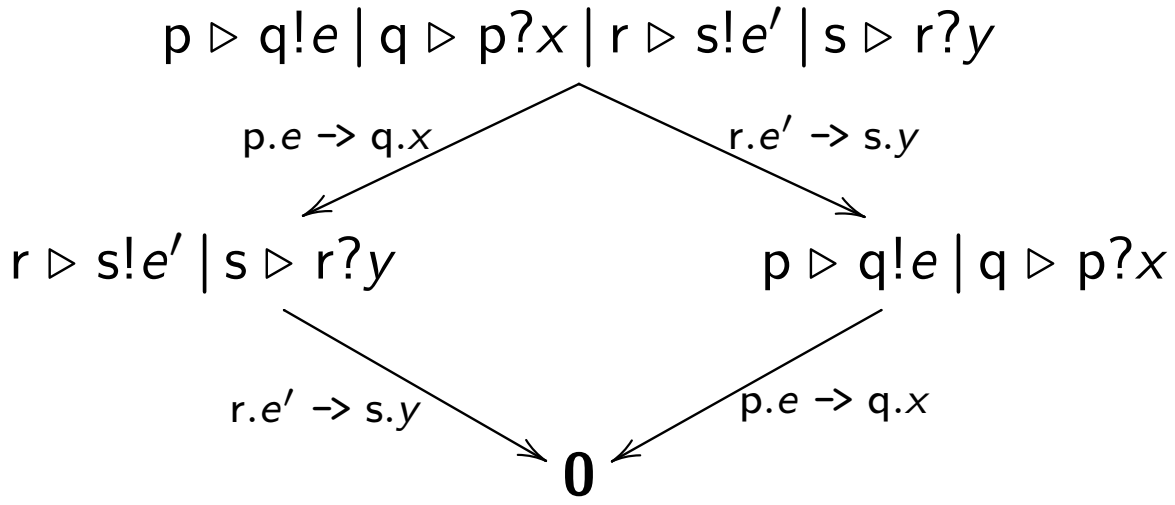
extraction by example (i/iii)

$$\mathsf{p} \triangleright \mathsf{q}!e \mid \mathsf{q} \triangleright \mathsf{p}?x \mid \mathsf{r} \triangleright \mathsf{s}!e' \mid \mathsf{s} \triangleright \mathsf{r}?y$$

extraction by example (i/iii)



extraction by example (i/iii)

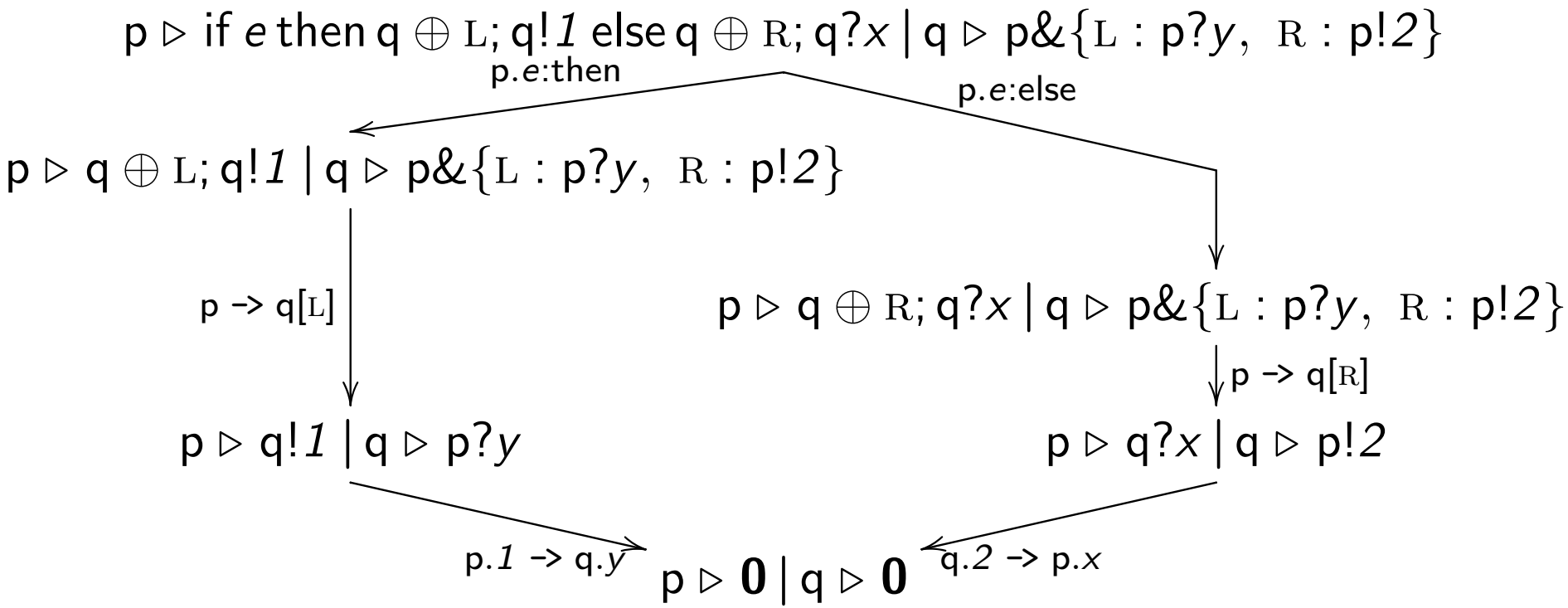


extracted choreography: p.e -> q.x; r.e' -> s.y or r.e' -> s.y; p.e -> q.x

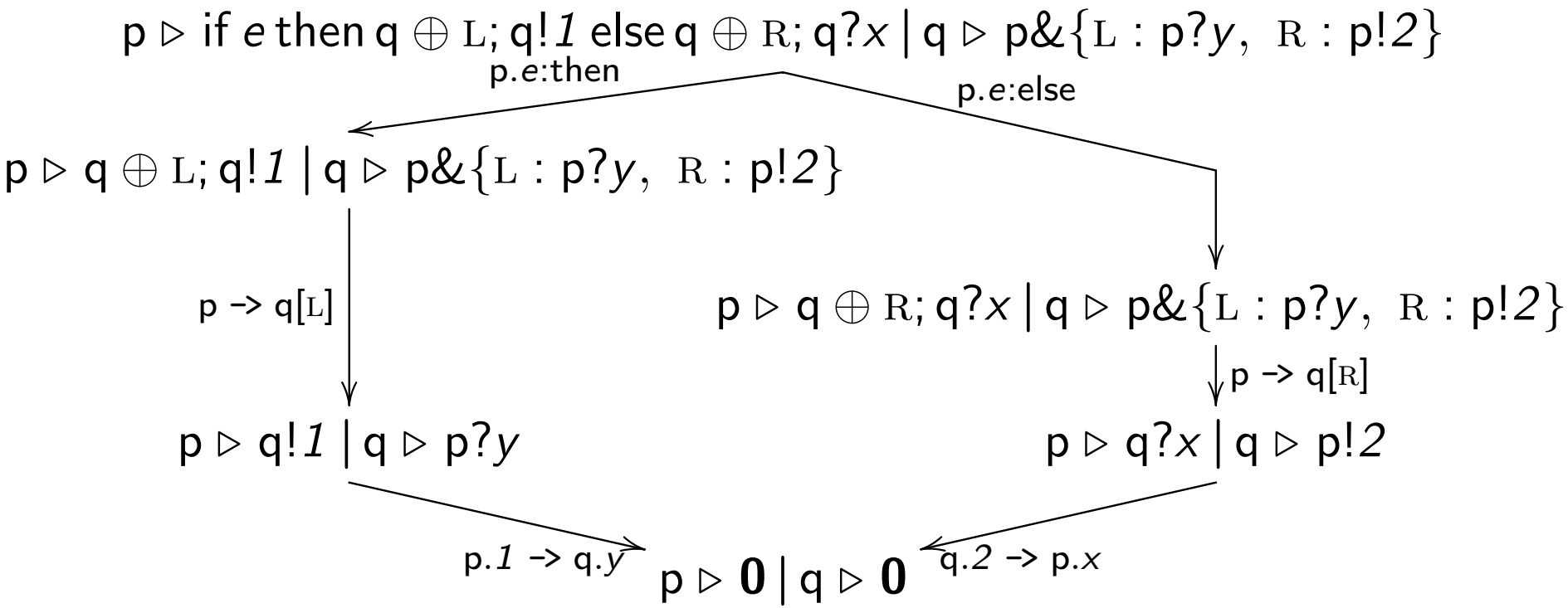
extraction by example (ii/iii)

$$\mathsf{p} \triangleright \mathsf{if}\ e\ \mathsf{then}\ \mathsf{q} \oplus \mathrm{L}; \mathsf{q}!1\ \mathsf{else}\ \mathsf{q} \oplus \mathrm{R}; \mathsf{q}?x \mid \mathsf{q} \triangleright \mathsf{p}\&\{\mathrm{L} : \mathsf{p}?y,\ \ \mathrm{R} : \mathsf{p}!2\}$$

extraction by example (ii/iii)



extraction by example (ii/iii)

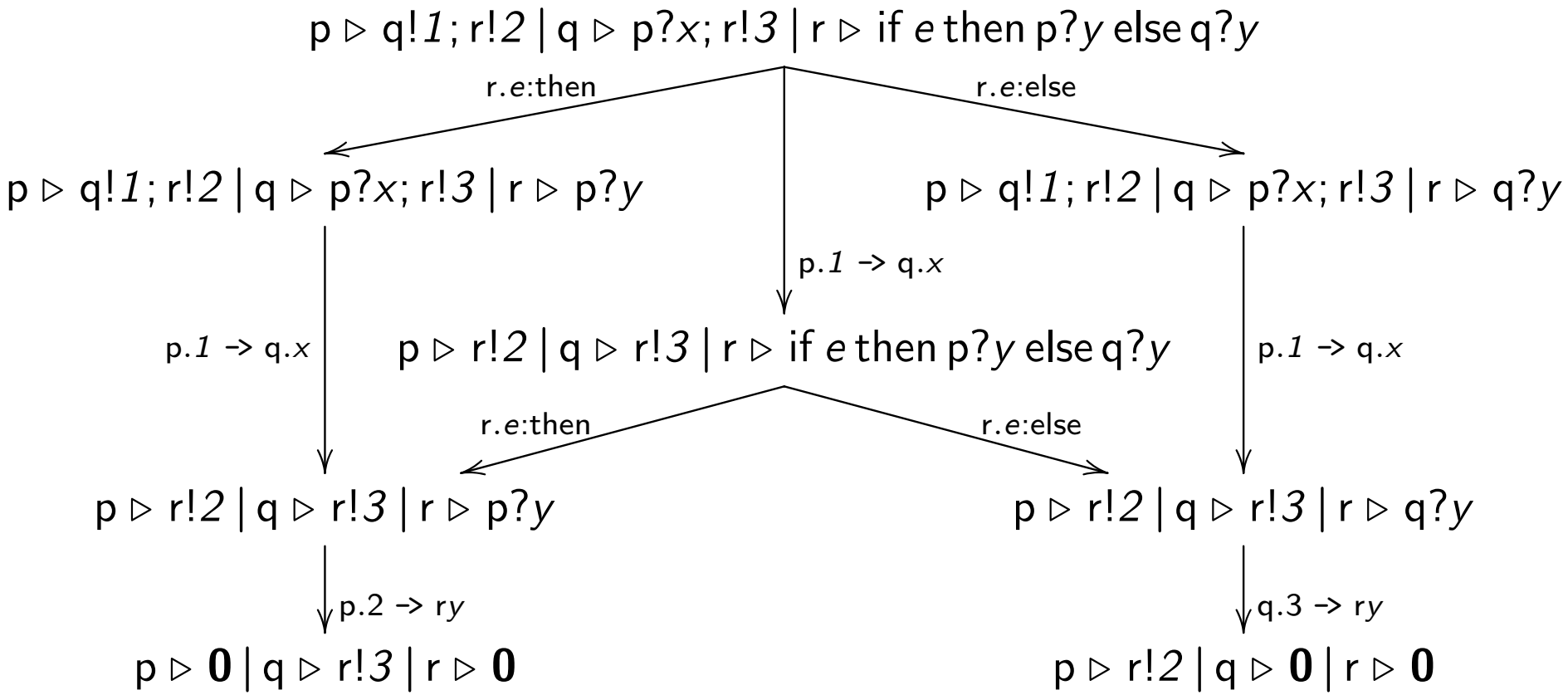


extracted choreography:
if p.*e* then (p -> q[L]; p.*1* -> q.*y*) else (p -> q[R]; q.*2* -> p.*x*)

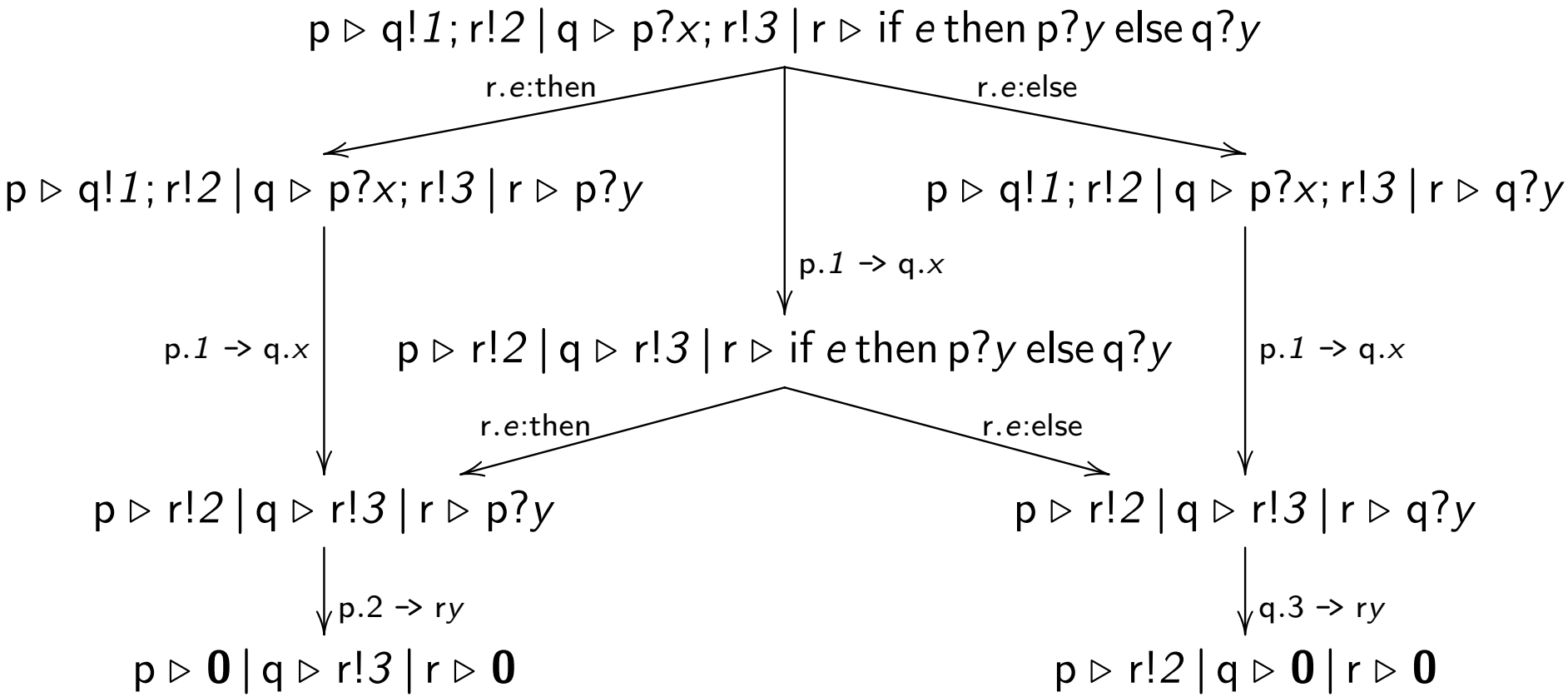
extraction by example (iii/iii)

$$\mathsf{p} \triangleright \mathsf{q}!1\,;\mathsf{r}!2 \mid \mathsf{q} \triangleright \mathsf{p}?x;\mathsf{r}!3 \mid \mathsf{r} \triangleright \text{if } e \text{ then } \mathsf{p}?y \text{ else } \mathsf{q}?y$$

extraction by example (iii/iii)



extraction by example (iii/iii)



all paths in the graph end in a deadlocked state, so the network cannot be extracted to a choreography

properties (finite case)

- termination
- soundness (the extracted choreography is bisimilar to the network)
- completeness (deadlock-free networks are extractable)
- efficient-ish (polynomial in the size of the network, graph size linear in the size of the extracted choreography)

dealing with recursion

properties (general case)

the good ones

- termination
- soundness (the extracted choreography is bisimilar to the network)

properties (general case)

the good ones

- termination
- soundness (the extracted choreography is bisimilar to the network)

the not-so-good ones

- incompleteness (no can do: deadlock-freedom is undecidable)
- horrible complexity (graph size exponential in the size of the extracted choreography)

(still better than previous state-of-the-art)

implementation

creative solutions

- list of bad nodes
- choice paths
- some global variables
- parallelization
- extraction strategies
- livelocks
- three-valued logic for clever backtracking

testing

methodology

1. examples from state-of-the-art
2. reverse epp
3. network fuzzer
4. network scrambler

testing

methodology

1. examples from state-of-the-art
2. reverse epp
3. network fuzzer
4. network scrambler

bottomline

$\rightsquigarrow$ it kind of actually works :-)

status and future work

- journal article in progress, expected submission in the winter
- generalize with process spawning

thank you!