Leafy automata for higher-order concurrency

A. Dixon
Warwick

R. Lazic
Warwick

A. S. Murawski
Oxford

I. Walukiewicz
CNRS & Bordeaux

Finitary Idealized Concurrent Algol (FICA)

Idealized Concurrent Algol combines shared-memory concurrency with higher-order computation.

$$M||N \qquad \lambda x.M \qquad MN$$

$$M := N \qquad !M \qquad \text{newvar}\, x = i \,\text{in}\, M$$

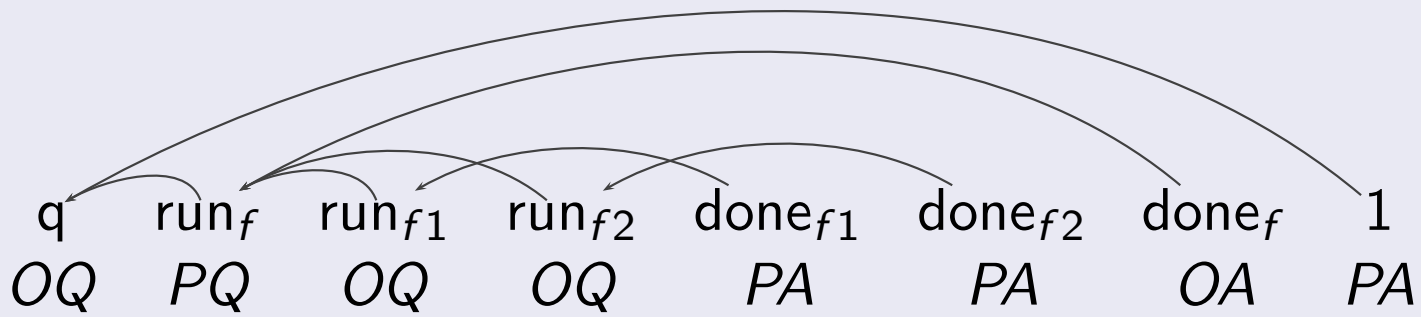
$$\text{grab}(M) \qquad \text{release}(M) \qquad \text{newsem}\, x = i \,\text{in}\, M$$

finitary = finite datatypes + looping (but no recursion)

Game semantics of ICA [Ghica & M., FOSSACS 2004]

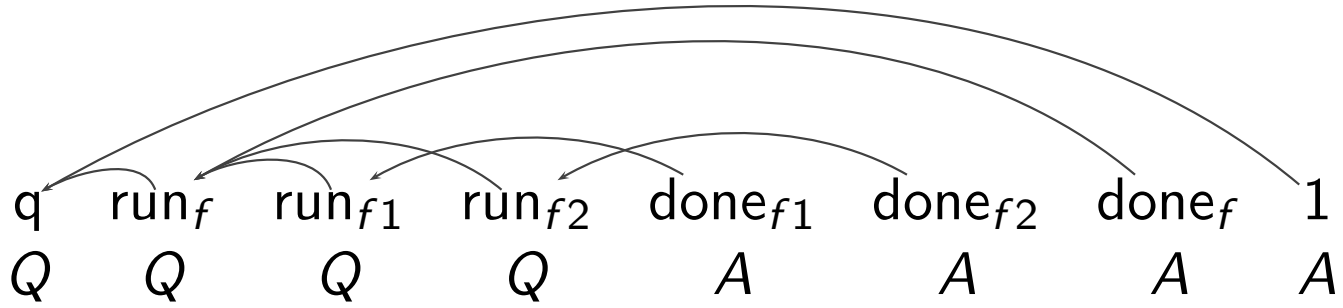
Game semantics views computation as interaction (play) between the context (O) and the program (P). Plays are sequences of moves connected by pointers. Programs are interpreted as strategies (sets of plays).

$$\lambda f^{\mathsf{com}\to\mathsf{com}\to\mathsf{com}}.\quad \mathsf{newvar}\ \ x := 0\ \ \mathsf{in}\ \ f(x := 1)(x := 2);\ \ !x$$



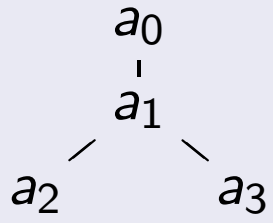
- Questions (Q) = calls
- Answers (A) = returns

We would like to represent plays as data words

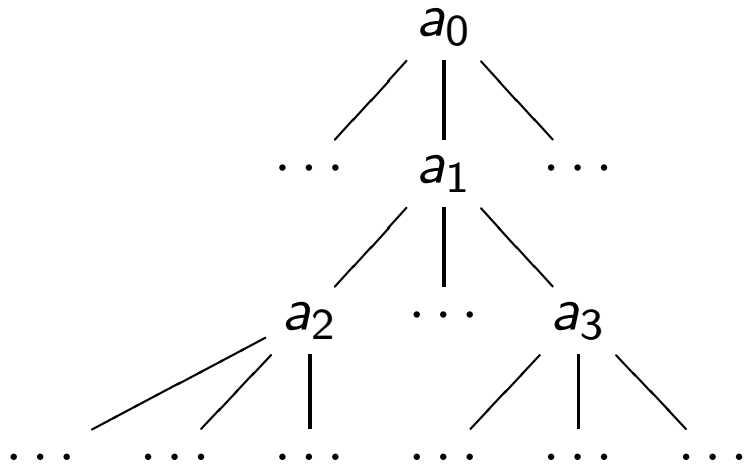


$(\mathsf{q}, a_0)\ (\mathsf{run}_f, a_1)\ (\mathsf{run}_{f1}, a_2)\ (\mathsf{run}_{f2}, a_3)\ (\mathsf{done}_{f1}, a_2)\ (\mathsf{done}_{f2}, a_3)$

- Pointers from answers will be represented using the data value of the corresponding question, e.g. $(\mathsf{done}_{f1}, a_2)$.
- To represent pointers from questions, we assume that the set of data has (infinite) tree structure and rely on the parent-child relationship.



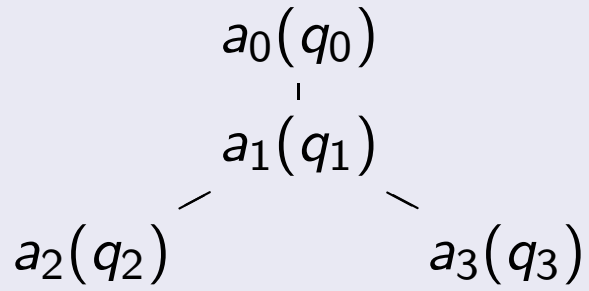
We use a tree-shaped dataset $\mathcal{T}$



- The dataset $\mathcal{T}$ is a countably infinite tree (infinitely many levels, infinite branching).
- To capture plays, we define a special kind of automata, which will accept data words from $(\mathit{Moves} \times \mathcal{T})^*$.

Leafy automata

Configurations of leafy automata will be finite subtrees of $\mathcal{T}$ labelled with states. The empty tree is the initial configuration.

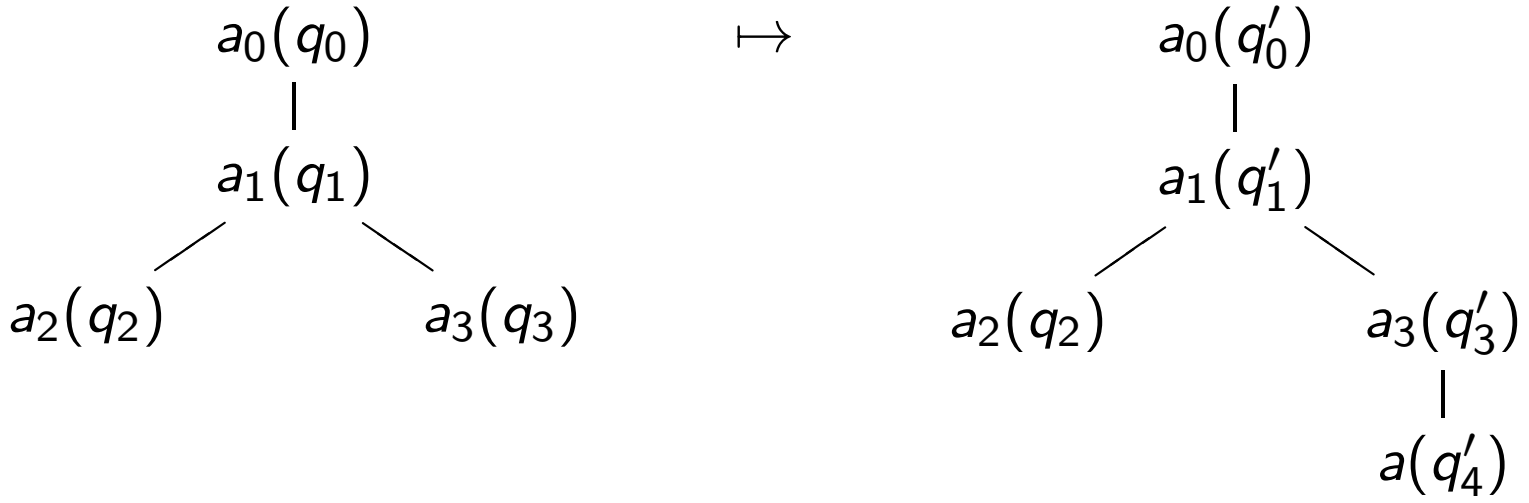


There are two kinds of transitions, each involving a leaf:

- add leaf,
- remove leaf.

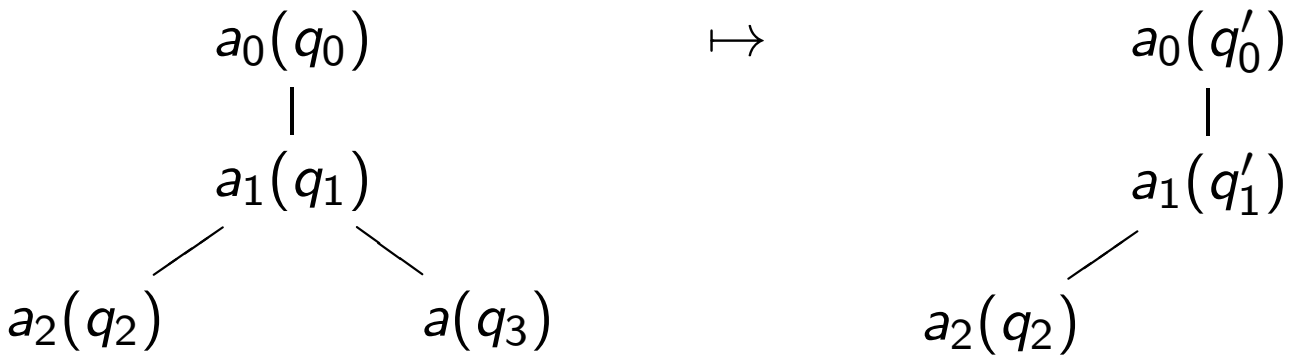
Add leaf while reading (*question*, a)

- a must be fresh (not encountered yet).
- If a is not the root of $\mathcal{T}$, the parent of a in $\mathcal{T}$ (a_3 in the picture) must occur in the current configuration.
- Only the states from the associated branch may be accessed and updated.



Remove leaf while reading $(answer, a)$

- a must occur in the current configuration and *must be a leaf*.
- Only the states from the associated branch may be accessed and updated.



Acceptance

A data word $w = (t_1, a_1) \cdots (t_k, a_k)$ is accepted by a leafy automaton if there exists a sequence of transitions reading w starting from the empty configuration and ending with the empty configuration.

- Emptiness is undecidable already at level 2 (root is level 0).
- At level 1, emptiness is interreducible with the Petri-net reachability problem.
- Equivalence is undecidable at level 1.

Correspondence with FICA

Main results

- For any FICA term, there exists a leafy automaton representing its game semantics.
- For any leafy automaton, there exists a FICA term representing the language of the automaton.

Conclusion

Leafy automata = automata-theoretic formalism for representing higher-order concurrent computation