

The logo for AGIR (Analyse Globalisée des Données d'Imagerie Radiologique) is displayed in large, bold, orange capital letters. It is positioned on the left side of the slide, partially overlapping a white rounded rectangle. The background of the slide features a blurred image of a medical scan with various colored overlays (green, yellow, red) on a dark background.

AGIR

Analuse Globalisée des Données d'Imagerie Radiologique

Implementing Turing machines with the ScufI language

Tristan Glatard*, Johan Montagnat
CNRS I3S / INRIA Sophia-Antipolis

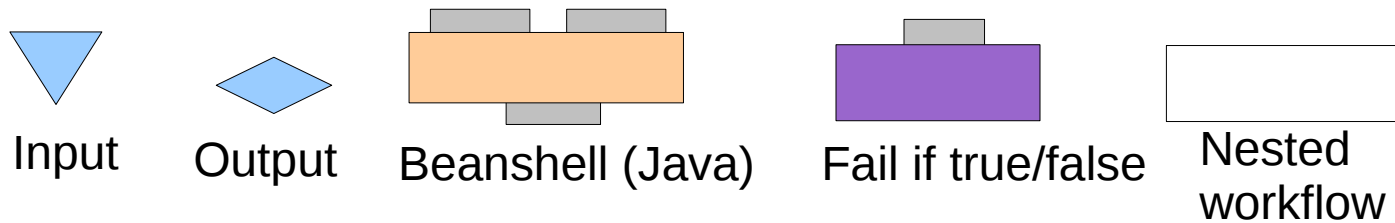
3rd International Workshop on Workflow Systems in e-Science
in conjunction with CCGrid'08 - Lyon



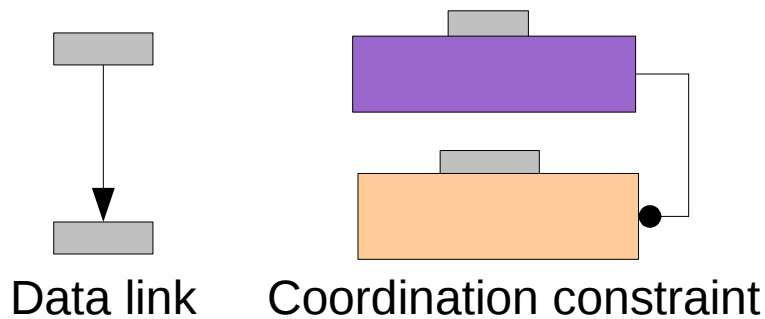
*now at the University of Amsterdam

- **Scufl is becoming a *de-facto* standard**
 - Due to Taverna's spread
 - Because of its simplicity
- **Study the expressiveness**
 - Is Scufl sufficient for describing complex applications ?
 - Is a (painful) language shift to be scheduled in the future ?
- **Detail a Scufl use-case**
 - Identify reusable patterns
 - Show implementation tricks
- **Identify potential improvements**
 - Expressiveness limitations
 - Definition flaws
 - Minimal subset of required processors

Processors

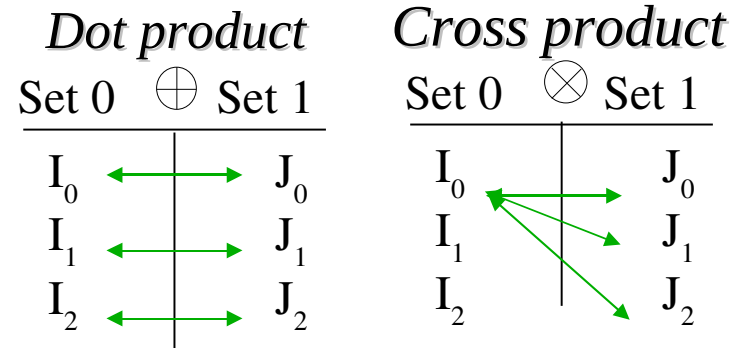


Links



Control operators

– Scufl is a parallel language



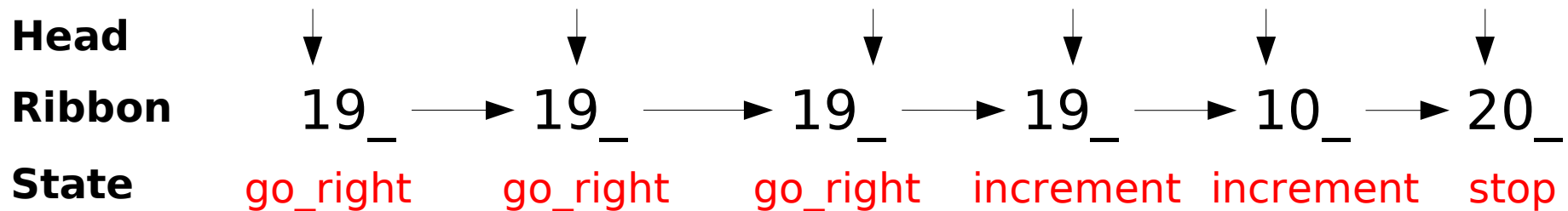
- **Components**
 - Ribbon with symbols from a finite alphabet
 - Init and stop states from a finite set
 - Head positioned on a ribbon cell
 - Transition function
- **Principle**
 - (1) Head reads current symbol
 - (2) Transition function produces new state, symbol and head shift
 - (3) Head writes symbol on ribbon and moves
 - (4) State is updated
- **Church-Turing hypothesis**
 - Every computable function can be computed by such a machine

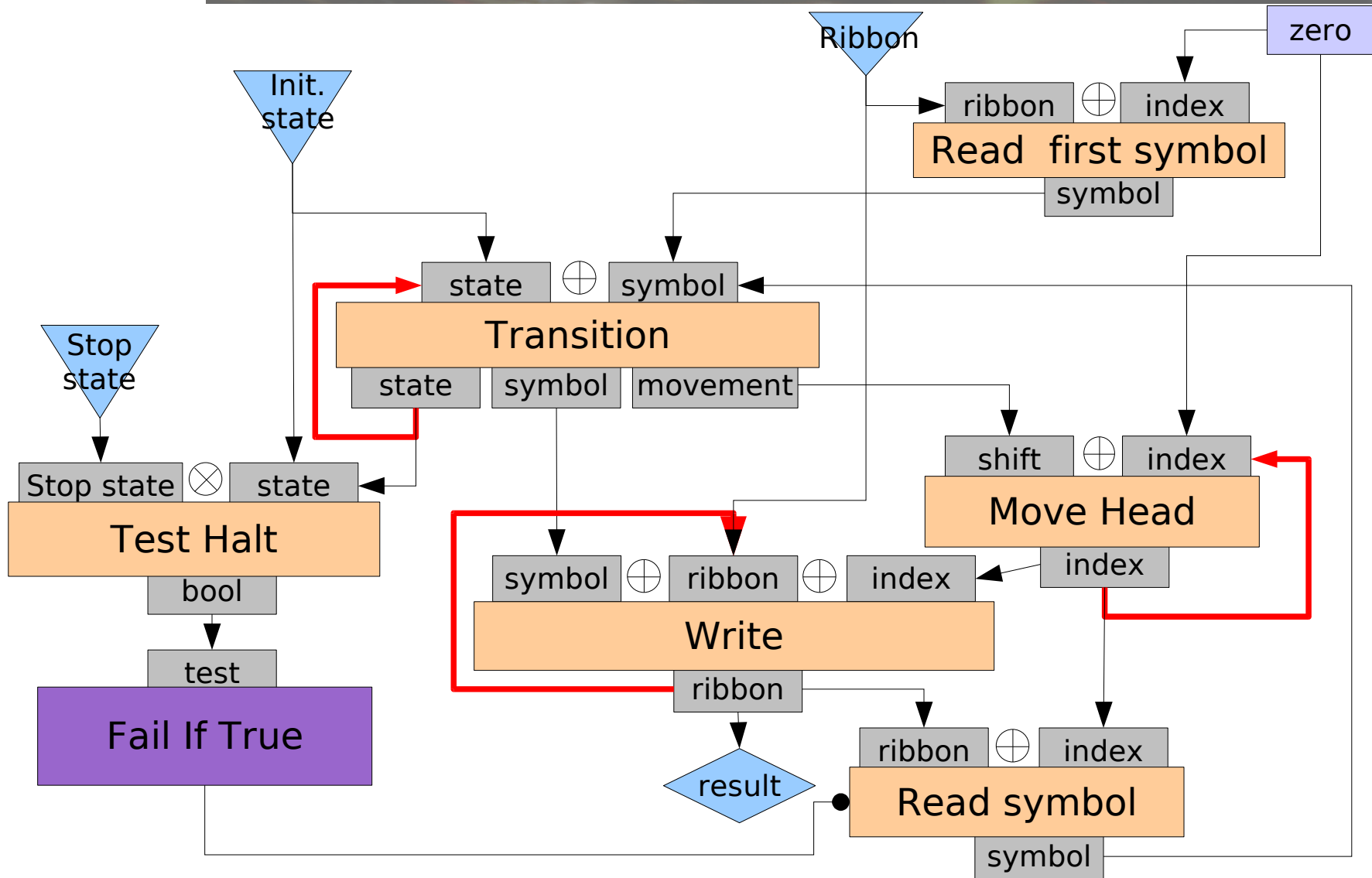
- **Machine definition**

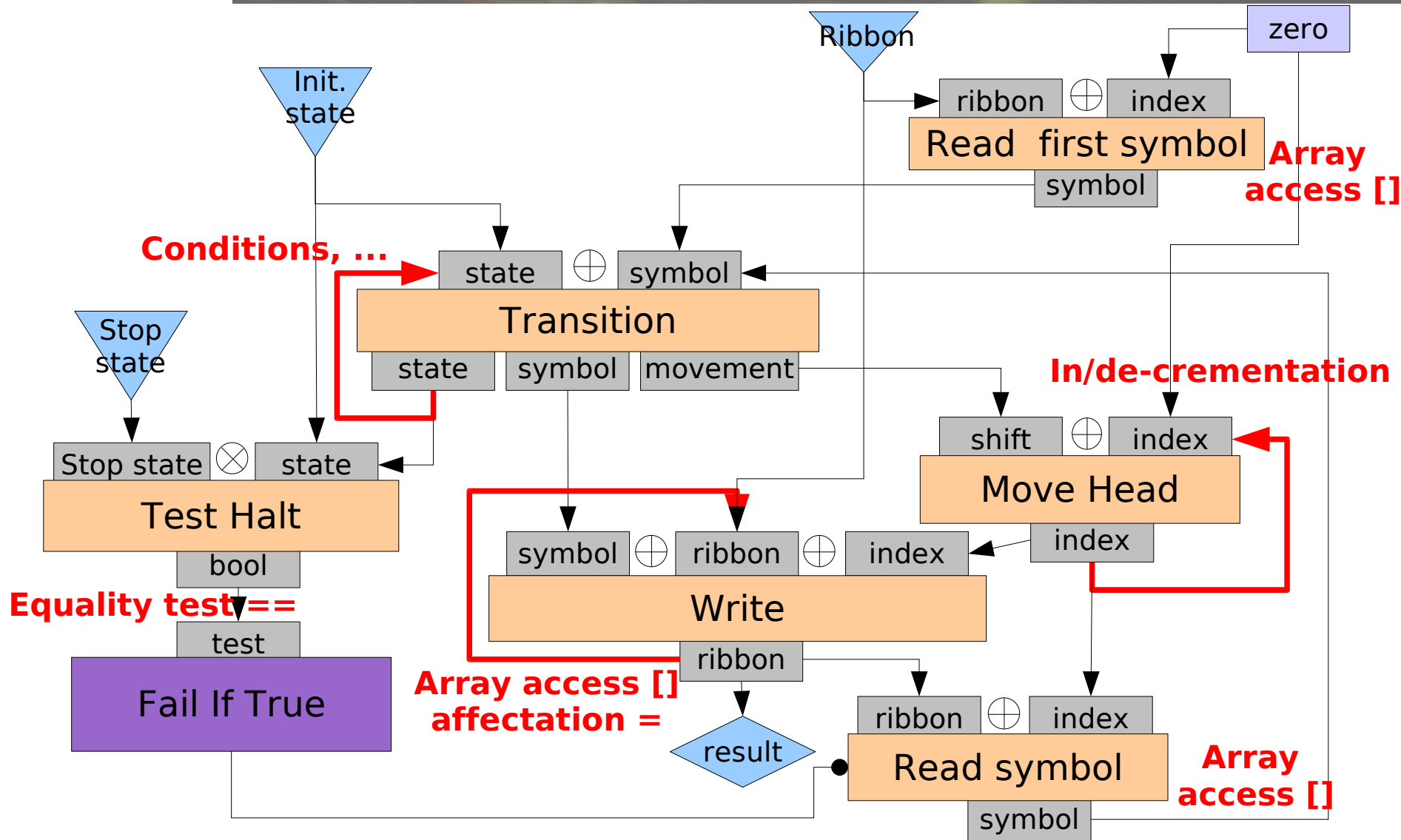
- Symbols: {_,0,1,2,3,4,5,6,7,8,9}
- States: {go_right (start), increment, stop}
- Transitions

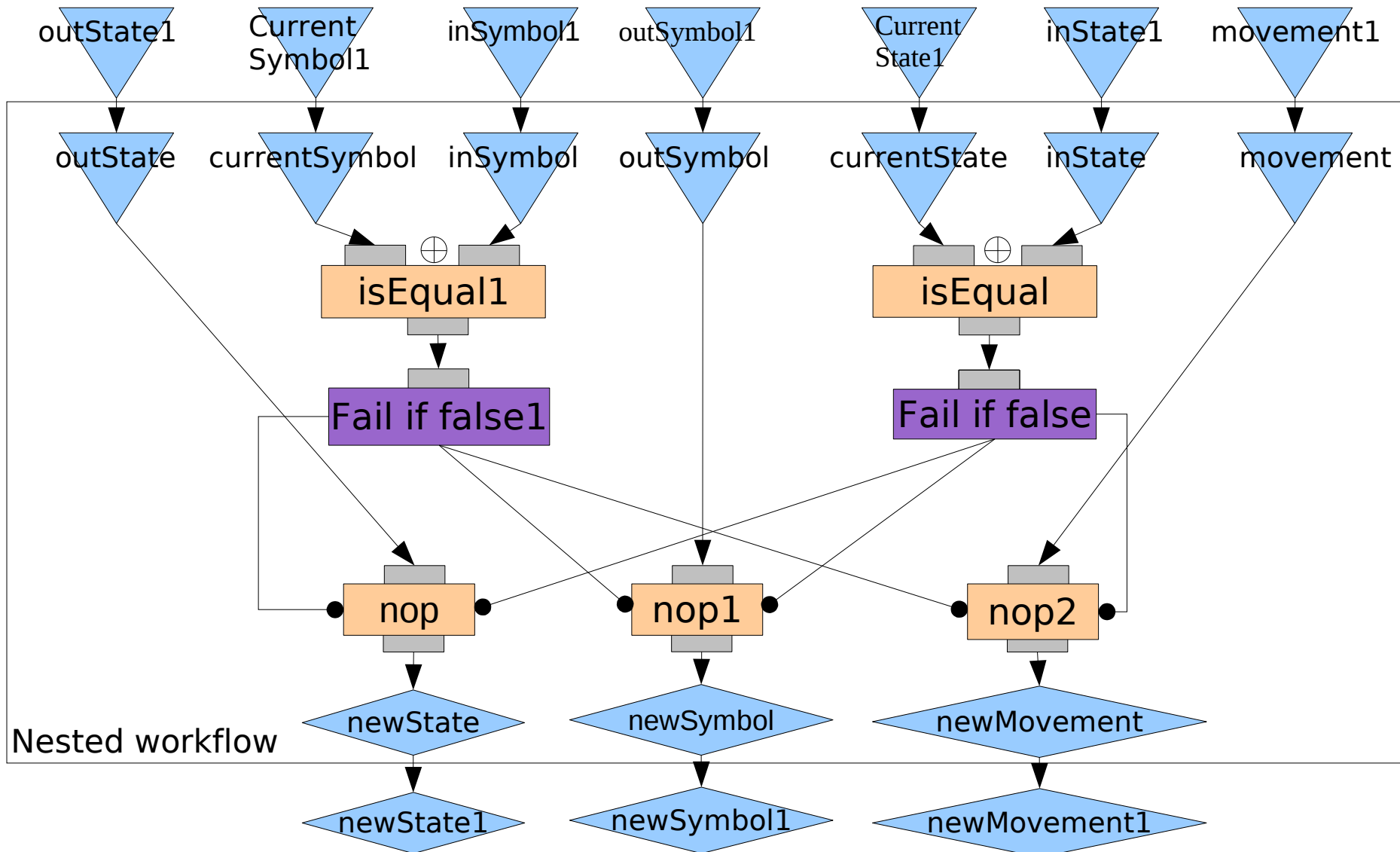
State	Symbol	State	Symbol	Movement
go_right	*	go_right	*	right
go_right	_	increment	_	left
increment	x in [0,8]	stop	x+1	none
increment	9	increment	0	left

- **Example**

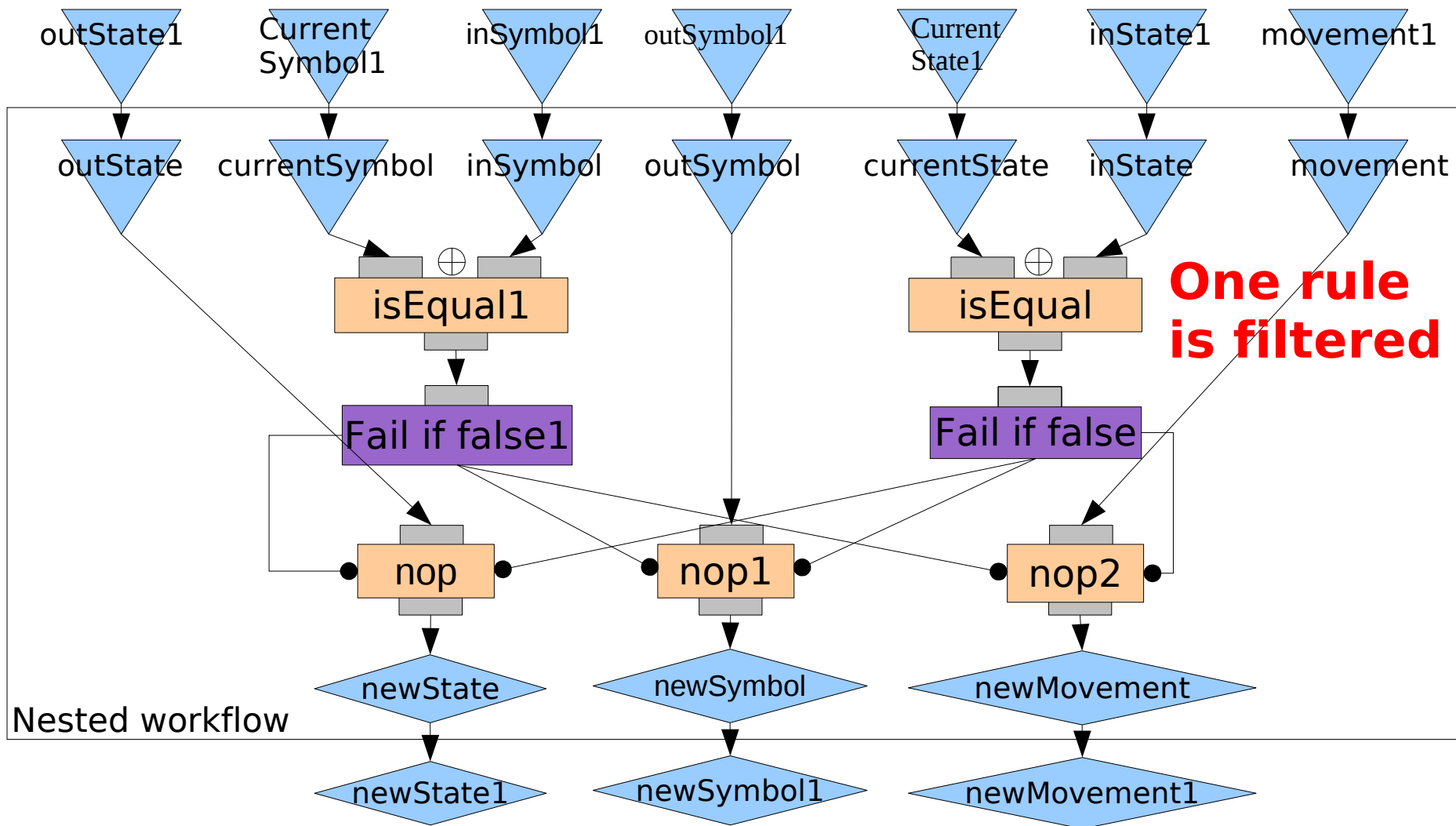


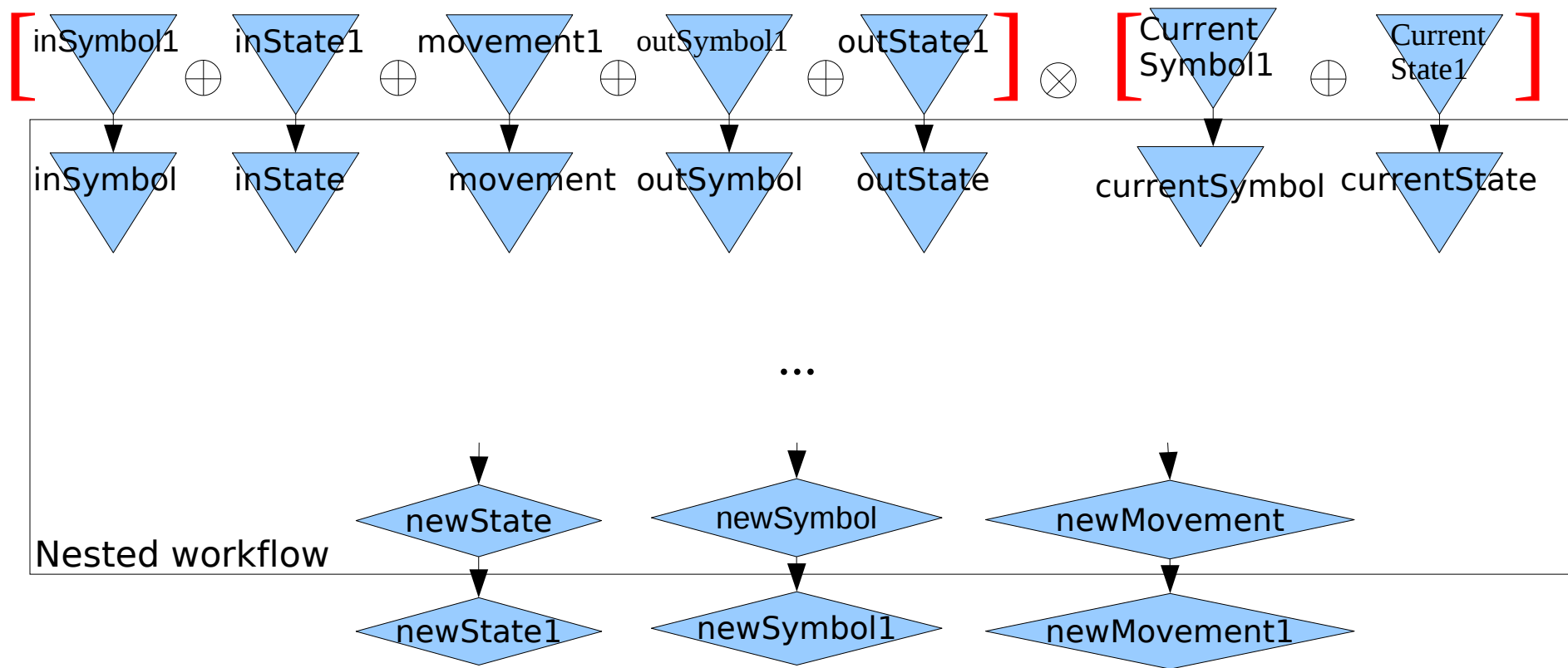






All the transition rules are fed here





- **Self-looping**
 - Maintains a state in the workflow
 - Used as a variable
- **Nested workflows**
 - Improve iteration strategies (see also <http://www.myexperiment.org/workflows/185>)
 - Improve conditionals
- **Limited amount of wrapped Java code**
 - Affectation: =
 - Equality test: ==
 - Array access: [] (might be doable with iteration strategies only)
 - Incrementation

- **Parallel execution**
 - Scufi is a parallel language (iteration is possible)
 - Completely puzzles execution in this case (consider e.g. 2 ribbons)
- **Synchronization between firings**
 - Needed because of self-looping
 - Not explicitly defined in Scufi
 - See Kepler directors ?
- **Code wrapping**
 - Several processors available by default in Taverna (e.g. Fail if true)
 - Only a limited subset of them in the language (=,==,[],++ and --) ?

- **Implemented a universal Turing machine in Scufi**
 - With limited Java inclusion
 - Suggests good expressiveness
- **Identified required patterns and limitations**
 - ✓ Self-looping
 - ✓ Nested workflows
 - ✗ Parallel execution
 - ✗ Synchronization between firings
- **Does not mean that easy implementations are possible**
 - Workflow patterns [van der Aalst]
 - Concrete applications