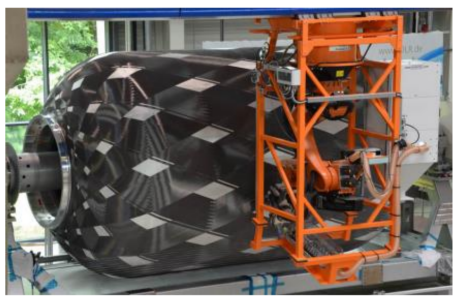
Hypergraph Automata

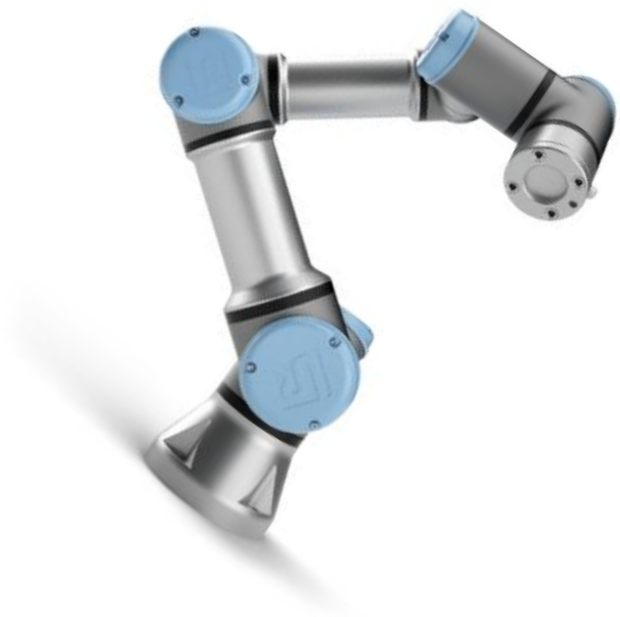
Roland Glück[1]

[1]Center for Lightweight Production Technology
German Aerospace Center

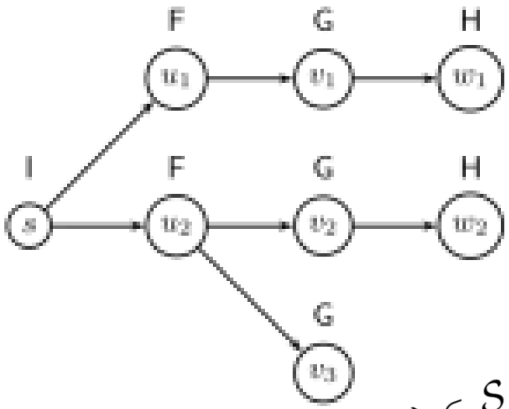
22st International Conference on Relational and Algebraic Methods Computer Science
Będlewo, April 8th, 2026

My Work





After Work



$$(u_1, v_1) \in E_1 \wedge (u_1, u_2) \in S \Rightarrow \exists v_2 : (v_1, v_2) \in S \wedge (u_2, v_2) \in E_2$$

Algorithm 30 Algorithm for computing the bisimulation quotient space

Input: finite transition system TS over AP with state space S
Output: bisimulation quotient space $S/\sim$

$\Pi := \Pi_{AP}$;
while there exists a splitter for Π **do**
  choose a splitter C for Π;
  $\Pi := \mathit{Refine}(\Pi, C)$; (* $\mathit{Refine}(\Pi, C)$ is strictly finer than Π *)
od
return Π

Hypergraphs

Definition

A *hypergraph* is a structure $H = (V, E)$ where V is a set of *nodes* and $E \subseteq V \times 2^V$ is its set of *hyperedges*.

- a hyperedge $(v, \emptyset)$ is called *empty*,
- otherwise *nonempty*

Definition

A *hypergraph automaton* is a structure $HA = (V, E, \delta, \mathrm{s}, \mathrm{F})$ such that

- (V, E) is a finite hypergraph without empty hyperedges, i.e., V is a finite set of states and $E \subseteq V \times 2^V$ is a set of nonemtpy hyperedges,
- $\gamma : E \to V$ is the *goal state function* of a hyperedge,
- $\delta : E \to \Sigma \cup \{\varepsilon\}$ is an *edge labeling function* and
- $\mathrm{s} \in V$ and $\mathrm{F} \subseteq V$ are the *start state* and the set of *final states*, resp.

Searching for Semantics

- traditional finite automata correspond to to regular expressions
- also captured by Kleene Algebra
- intended semantics: CKA (Concurrent Kleene Algebra)
- hyperedges should serve for ||-operator of CKA
- pomsets?

Configurations

- *labeled state*: $(v, V') \in V \times 2^V$
- *configuration*: set of labeled states
- labeled state (v_1, V_1) is *waiting* (v_2, V_2) if $v_2 \in V_1$
- for a configuration C: *waiting states* given by $w(C) =_{def} \bigcup_{(v,V' \in C)} V'$
- for a configuration C: a labeled state $s = (v, V') \in C$ is *free in* C if s is not waiting for a $(w, W') \in C$
- hyperedge $h = (v, V')$ is *enabled in* C if v if free in C

Configuration Transition

- configuration transition $\rightarrow \subseteq 2^C \times \Sigma \cup \{\varepsilon\} \times 2^C$ defined by
- $(C_1, a, C_2) \in \rightarrow \Leftrightarrow_{df} \exists$ free labeled state $(v_1, V_1) \in C_1 \wedge \exists$ hyperedge $h = (v_1, V') : \delta(h) = a \wedge C_2 = C_1 - \{v_1, V_1\} \cup \{(w_1, (W_1 \cup V_1') - (\{w_1\} \cap V_1)) \,|\, (w_1, W_1) \in C_1 \wedge w_1 \in V'\}$
- intuitively, (v_1, V_1) is removed from C_1,
- the set of its labels (modeling the states waiting for this thread) is passed to its successors,
- except if a thread reaches its goal state

Acceptance

Definition

A hypergraph automaton accepts a word $a_1 a_2 \dots a_n$ if there is a sequence $C_0 \xrightarrow{a_1} C_1 \dots \xrightarrow{a_n} C_m$ (where we allow transition labeled with ε which do not contribute to the read word) with $C_0 = \{(s, \emptyset)\}$ and all states in C_m have the form $(v, \emptyset)$ for some $v \in F$.

Questions

- does this make sense at all?
- can it be used to recognize pomsets?
- is this something useful or only a known concept in disguise?
- does it fufil the expectations?