Sequential algorithms as spans

Glynn Winskel ??

University of Cambridge Computer Laboratory, England

1 Event structures

An *event structure* comprises $(E, \mathrm{Con}, \leq)$, consisting of a countable set E, of *events* which are partially ordered by $\leq$, the *causal dependency relation*, and a *consistency relation* Con consisting of finite subsets of E, which satisfy

$$\begin{array}{l} \{e' \mid e' \leq e\} \text{ is finite for all } e \in E, \\ \{e\} \in \mathrm{Con} \text{ for all } e \in E, \\ Y \subseteq X \in \mathrm{Con} \Rightarrow Y \in \mathrm{Con}, \quad \text{and} \\ X \in \mathrm{Con} \ \& \ e \leq e' \in X \Rightarrow X \cup \{e\} \in \mathrm{Con}. \end{array}$$

The *configurations*, $\mathcal{C}(E)$, of an event structure E consist of those subsets $x \subseteq E$ which are

Consistent: $\forall X \subseteq x.\ X$ is finite $\Rightarrow X \in \mathrm{Con}$, and

Down-closed: $\forall e, e'.\ e' \leq e \in x \Rightarrow e' \in x$.

We write $\mathcal{C}^o(E)$ for the *finite* configurations of the event structure E.

An event structure is *filiform* if it satisfies in addition that

$$\forall e_1, e_2, e.\ e_1, e_2 \leq e \ \Rightarrow \ e_1 \leq e_2 \text{ or } e_2 \leq e_1 \ .$$

It is *concrete* if inconsistency is determined solely by the following conflict relation: say e_1 and e_2 are in immediate conflict if they are distinct and share an immediate predecessor of odd parity; say a pair is in conflict if they causally depend on a pair in immediate conflict. The parity of an event is the even/oddness of its height starting counting at 1 (so minimal events have height 1 and odd parity).

2 Maps of event structures

For sequential spans we shall use input and output maps of the following kinds. [Pretty sure that both can be obtained as Kleisli maps of monads up to symmetry on event structures with symmetry and rigid maps, and expect appropriate distributive laws so that sequential spans described later can be obtained as general spans of rigid maps....]

Output maps ('partial rigid maps': An *output map* of event structures $f : E \rightharpoonup E'$ is a partial function on events $f : E \rightharpoonup E'$ such that

- for all $x \in \mathcal{C}^o(E)$ its direct image $fx \in \mathcal{C}^o(E')$,
- for all $x \in \mathcal{C}^o(E)$ if $e_1, e_2 \in x$ and $f(e_1) = f(e_2)$ (with both defined), then $e_1 = e_2$, and
- if $e_1 \leq e_2$ & $f(e_1), f(e_2)$ both defined, then $f(e_1) \leq f(e_2)$.

Input maps: An *input map* of event structures $f : E \rightharpoonup E'$ is a partial function on events $f : E \rightharpoonup E'$ such that

- for all $x \in \mathcal{C}^o(E)$ its direct image $fx \in \mathcal{C}^o(E')$
- if $e' \leq f(e)$ (defined), then $\exists! e_1 \leq e.\ f(e_1) = e'$, and
- if $e_1 \leq e_2$ & $f(e_1), f(e_2)$ both defined, then $f(e_1) \leq f(e_2)$.

Note output maps are special input maps. [Believe input maps can be obtained as Kleisli maps wrt a monad up to symmetry on output maps.]

Proposition 2.1 *Given output map out and input map in with common codomain, there is a universal pair, an input map in′ and output map out′, such that*

$$\begin{tikzcd} & \bullet \arrow[dl, dotted, harpoon, "in'"'] \arrow[dr, dotted, harpoon, "out'"] & \\ \bullet \arrow[dr, harpoon, "out"'] & & \bullet \arrow[dl, harpoon, "in"] \\ & \bullet & \end{tikzcd}$$

commutes (an 'input-output pullback').

3 Sequential spans

A sequential span comprises a span

$$\begin{tikzcd} & S \arrow[dl, "in"'] \arrow[dr, "out"] & \\ A & & B \end{tikzcd}$$

where A, B, S are filiform event structures, with A and B concrete, *in* is an input map and *out* is an output map such that

- the span is deterministic: if $Z \subseteq_{\text{fin}} \mathcal{C}^o(S)$ and $in\, Z$ is compatible in $\mathcal{C}^o(A)$ then Z is compatible in $\mathcal{C}^o(S)$.
- for all events $s \in S$ if $in(s)$ is defined then $out(s)$ is undefined (and contrapositively, if $out(s)$ is defined then $in(s)$ is undefined),
- two distinct events of S sharing an immediate predecessor of odd parity are inconsistent (though, generally, inconsistency in S will not be generated from just these),
- satisfying the parity constraints that *in* reverses parity while *out* preserves it.

The parity of an event in A or B is the even/oddness of its height while the parity of an event in S is the even/oddness of its height discounting 'internal' events where both *in* and *out* are undefined. [Believe such spans can be realized as spans of rigid maps with suitable pseudo monads at the feet.]

In a sequential algorithm filling two concurrent output cells can involve investigating the same input cell, the reason why input maps need to break the second 'local injectivity' condition met by output maps.

I believe the following:

By dropping its internal events, a sequential span gives rise to an 'extremal' sequential span with no internal events:

$$\begin{tikzcd} & S \arrow[dl, "in"'] \arrow[dr, "out"] \arrow[d, dotted] & \\ A & S_0 \arrow[l, "in_0"'] \arrow[r, "out_0"] & B \end{tikzcd}$$

where the mediating map is an output map. Up to isomorphism, extremal sequential spans correspond to affine sequential algorithms (as described in Curien or Winskel). The prime configurations of S_0 correspond to 'schedules' in Hyland-Harmer-Mellies.

Sequential spans compose by 'input-output pullback'

$$
\begin{tikzcd}
 & & U \arrow[dl, dotted, "in'"'] \arrow[dr, dotted, "out'"] & & \\
 & S \arrow[dl, "in_1"'] \arrow[dr, "out_1"'] & & T \arrow[dl, "in_2"] \arrow[dr, "out_2"] & \\
A & & B & & C
\end{tikzcd}
$$

U and induce composition of affine sequential algorithms.