Circus Time with Reactive Designs

Kun Wei, Jim Woodcock and Ana Cavalcanti
Department of Computer Science
University of York

August 27, 2012

Outline

- Introduction to *Circus* family
- *Circus Time*
  - Summary
  - Observation
  - Syntax
  - Healthiness conditions
- Reactive designs
  - Sequential composition
  - Hiding
  - Recursion
- Conclusion and future work

Circus **Family**

Circus is a combination of Z, CSP, guarded commands and refinement calculus, and has developed into a family of languages for specification, programming and verification. Its semantics is based on UTP.

Circus Time is an extension of a subset of *Circus* with some timed operators added to the notion of actions in *Circus*.

A new *Circus Time* has been recently developed to provide more time operators such as (hard) deadlines.

Summary of *Circus Time*

- A discrete-time model.
- Semantics is based on UTP.
- An extension to *Circus* and original *Circus Time* (Sherif and He).
- *Miracle* (the top element in this complete lattice) is adopted and therefore defines deadline operators.
- More time operators from Timed CSP, but not included in the original *Circus Time*.
- Reactive-design semantics to each action.

Observation in *Circus Time*

Observational variables:

- ok,ok': *boolean*
- wait,wait': *boolean*
- tr,tr':$seq^{+}(seq\ Event)$
- ref,ref':$seq^{+}(\mathbb{P}\ Event)$
- state, state':$N \pfun value$

For example,

$$tr' = \langle \langle a \rangle, \langle b, c \rangle, \langle d \rangle, \langle e, f \rangle, ... \rangle$$
$$ref' = \langle r_1, r_2, r_3, r_4, ... \rangle$$

Time operators in *Circus Time*

$$
\begin{aligned}
P ::= & \mathit{Skip} \mid \mathit{Stop} \mid \mathit{Miracle} \mid \mathit{Chaos} \mid N := e \\
& \mid c.e \rightarrow P \mid P \,;\, Q \mid P \lhd b \rhd Q \mid b \,\&\, P \\
& \mid P \,\Box\, Q \mid P \sqcap Q \mid P \,[\![\, s_1 \mid \{\!|\, CS \,|\!\} \mid s_2 \,]\!]\, Q \mid P \setminus CS \\
& \mid \mathit{Wait}\ d \mid \mathit{Wait}\ d_1..d_2 \mid \mu\, N \bullet P \\
& \mid P \rhd\!\{d\}\, Q \mid P \blacktriangleright d \mid P \blacktriangleleft d \mid c.e@t \rightarrow P
\end{aligned}
$$

- $\mathit{Wait}\ d$: wait for d time units
- $\mathit{Wait}\ d_1..d_2$: non-deterministic wait
- $P \rhd\!\{d\}\, Q$: P if no observable event happens within d, otherwise Q will take place
- $P \blacktriangleright d$: P **must** terminate within d
- $P \blacktriangleleft d$: observable events in P **must** happen within d
- $c.e@t \rightarrow P$: t records the amount of time which has elapsed between the start and the occurrence of $c.e$

Healthiness conditions

$$\mathbf{R1_t}(X) \mathrel{\widehat{=}} X \wedge RT$$

$$\mathbf{R2_t}(X) \mathrel{\widehat{=}} \exists\, r \bullet X[\langle\langle\rangle\rangle, diff(tr', tr), \langle r\rangle, ref' - front(ref)/tr, tr', ref, ref']$$

$$\mathbf{R3_t}(X) \mathrel{\widehat{=}} \mathbb{I}_t \lhd wait \rhd X$$

$$\mathbf{R_t} = \mathbf{R1_t} \circ \mathbf{R2_t} \circ \mathbf{R3_t}$$

$$\mathbf{CSP1_t}(X) \mathrel{\widehat{=}} X \vee (\neg\, ok \wedge RT)$$

$$\mathbf{CSP2_t}(X) \mathrel{\widehat{=}} X\,;\, J$$

$$\mathbf{CSP3_t}(X) \mathrel{\widehat{=}} Skip\,;\, X \qquad \mathbf{CSP4_t}(X) \mathrel{\widehat{=}} X\,;\, Skip \qquad \mathbf{CSP5_t}(X) \mathrel{\widehat{=}} X \,|||\, Skip$$

$$RT \mathrel{\widehat{=}} tr \preccurlyeq tr' \wedge front(ref) \leq ref' \wedge \#ref' = \#tr' \wedge \#ref = \#tr$$

$$\mathbb{I}^{-ok} \mathrel{\widehat{=}} \left(\begin{array}{l} tr' = tr \wedge front(ref') = front(ref) \wedge wait' = wait \wedge \\ state' = state \wedge \#ref' = \#tr' \wedge \#ref = \#tr \end{array} \right)$$

$$\mathbb{I} \mathrel{\widehat{=}} (ok' = ok \wedge \mathbb{I}^{-ok})$$

$$\mathbb{I}_t \mathrel{\widehat{=}} (\neg\, ok \wedge RT) \vee (ok' \wedge \mathbb{I}^{-ok})$$

Reactive designs

A CSP process can be obtained by applying the theory of reactive processes to the theory of designs. That is, for a reactive design, the design describes the behaviour when its predecessor has terminated or not diverged, and the other situations of its behaviour are left to the theory of reactive processes.

Suppose P is an action of *Circus Time*,

$$P = \mathbf{R_t}(\neg\, P_f^f \vdash P_f^t)$$

where P_b^a is an abbreviation of p[a,b/ok',wait].

Reactive design for sequential composition

Original

$$P \,;\, Q \mathrel{\widehat{=}} \exists\, obs_0 \bullet P[obs_0/obs'] \wedge Q[obs_0/obs]$$

Reactive design

$$P \,;\, Q = \mathbf{R_t} \left(\begin{array}{c} \neg\, (\mathbf{R1_t}(P_f^f) \,;\, \mathbf{R1_t}(true)) \wedge \neg\, (\mathbf{R1_t}(P_f^t) \,;\, \mathbf{R1_t}(\neg\, wait \wedge Q_f^f)) \\ \vdash \\ \mathbf{R1_t}(P_f^t) \,;\, \mathbf{R1_t}(\mathbb{I} \lhd wait \rhd Q_f^t) \end{array} \right)$$

For example,

$$c.e \rightarrow Miracle = \mathbf{R_t}(true \vdash wait' \wedge {\frown}/tr' = {\frown}/tr \wedge possible(tr, tr', c))$$

Reactive design for hiding

hiJaC

Original

$$
\begin{aligned}
P \setminus CS &\mathrel{\widehat{=}} \mathbf{R_t}(\exists\, s, r \bullet P[s, r/tr', ref'] \wedge L_t)\,;\, Skip \\
L_t &\mathrel{\widehat{=}} diff\,(tr', tr) = diff\,(s, tr) \downarrow_t (\Sigma - CS) \wedge \\
&\qquad\qquad r - front(ref) = ((ref' - front(ref)) \cup_t CS) \\
diff\,(tr', tr) &\mathrel{\widehat{=}} \langle tr'(\#tr) - last(tr)\rangle \frown tail(tr' - front(tr))
\end{aligned}
$$

Reactive design

$$
P \setminus CS = \mathbf{R_t}\left(\begin{array}{c} \neg\,((\exists\, s, r \bullet P_f^f[s, r/tr', ref'] \wedge L_t)\,;\, \mathbf{R1_t}(true)) \\ \vdash (\exists\, s, r \bullet P_f^t[s, r/tr', ref'] \wedge L_t) \end{array}\right)
$$

Reactive design for recursion

hiJaC

Original

$$\mu X \bullet F(X) \mathrel{\widehat{=}} \bigsqcap \{X \mid F(X) \sqsubseteq X\}$$

Reactive design

$$\mu(X, Y) \bullet (\mathbf{R_t}(F(X, Y) \vdash G(X, Y))) = \mathbf{R_t}(\mu(X, Y) \bullet (F(X, Y) \vdash G(X, Y)))$$

$$\mu(X, Y) \bullet (F(X, Y) \vdash G(X, Y)) = P(Q) \vdash Q$$

where $P(Y) = \nu X \bullet F(X, Y)$

and $Q = \mu Y \bullet (P(Y) \Rightarrow G(P(Y), Y))$

Reactive design for recursion

Theorem

Let D and E be monotonic functions. If there exists a function R such that $R \circ D = E \circ R$, then $R(\mu\, D) = \mu\, E$.

Proof

$$\begin{aligned} \textit{Let}\quad & D(X \vdash Y) = F(X, Y) \vdash G(X, Y) \\ & E(\mathbf{R_t}(X \vdash Y)) = \mathbf{R_t}(F(X, Y) \vdash G(X, Y)) \end{aligned}$$

$$\begin{aligned} & \mathbf{R_t} \circ D = E \circ \mathbf{R_t} \\ =& \mathbf{R_t} \circ D(X \vdash Y) = E \circ \mathbf{R_t}(X \vdash Y) \\ =& \mathbf{R_t}(F(X, Y) \vdash G(X, Y)) = \mathbf{R_t}(F(X, Y) \vdash G(X, Y)) \\ =& \mathit{true} \end{aligned}$$

Case study: who introduces divergence?

$$(\mu\, X \bullet c \rightarrow X) \setminus \{c\} = \mathit{Chaos}$$

The UTP book says we should allow tr to range over infinite as well as finite. The infinite trace consists of all the c is not rule out by the weakest fixed point, and $ok' = \mathit{false}$ is not rule out as well. And the possible divergence is turned into *Chaos* by *Skip*.

- A simple prefix never diverges.
- Recursion is calculated by reactive designs.
- Hiding may introduce divergence.

Conclusion and future work

- We have developed the reactive design semantics to three important CSP operators, sequential composition, hiding and recursion.
- Compared to the original CSP semantics in UTP, the reactive designs provides us with a more concise, readable and uniform semantics, which can help us to exactly understand the behaviours of some subtle processes.
- Reactive designs also expose the pre-postcondition semantics.

Future work:

- Mechanisation of the semantics of *Circus Time* in a theorem prover.