Successful Termination in Timed CSP

Paul Howells
University of Westminster

Mark d'Inverno
Goldsmiths, University of London

Communicating Process Architectures (CPA 2013)

Overview of the Talk

- Motivation & Aims of Paper
- Successful Termination Problems in Original CSP
- Roscoe's "Standard" Solution
- Introduction to CSP$_T$
- An overview of Timed CSP
- Termination Issues in Timed CSP
- Example Termination Axiom
- Conclusions & Future Work

Motivation for the Paper

- Successful termination is important and should be modelled “consistently” within CSP & Timed CSP.
- Continue our investigation of successful termination within the CSP framework, consider how it is or should be modelled within Timed CSP.
- Believe similar issues exist in the various Timed CSP models as existed in the original CSP models.
- Believe it is possible to develop an improved treatment of successful termination within Timed CSP.
- Believe can be achieved by adopting a similar approach to that taken in resolving these issues when developing CSP_T.

Aims of the Paper

To provide an improved treatment of successful termination within Reed and Roscoe's Timed CSP framework.

- Investigate how successful termination is modelled in Reed and Roscoe's Timed CSP.
- Identify & discuss the issues that need to be considered when selecting termination axioms for each Timed CSP model, based on our experiences in defining CSP$_T$.
- Outline what a solution entails by identifying candidate termination axioms for each of the Timed CSP models.

Successful Termination Problems in Original CSP

In the original *failure-divergence* semantic models for CSP, developed by Hoare, Brookes & Roscoe during the 80's, the treatment of successful process termination, as modelled by *SKIP* & $\checkmark$, was incomplete.

Parallel operators: *alphabetised* ($_A\|_B$) & *interleaving* ($|||$), permitted intuitively contradictory processes to be defined.

For example:

$$\begin{aligned}
&(a \to SKIP) ||| (b \to SKIP) \\
&\qquad\qquad \equiv \quad (a \to ((\checkmark \to b \to SKIP) \,\Box\, (b \to \checkmark \to SKIP))) \\
&\qquad\qquad\quad \Box \;\, (b \to ((a \to \checkmark \to SKIP) \,\Box\, (\checkmark \to a \to SKIP)))
\end{aligned}$$

Right hand side $\checkmark$s cannot be interpreted as the successful termination of the left hand side process, since it continues to perform a, b and $\checkmark$ events.

A number of solutions have been proposed but the "standard" solution is due to Roscoe presented in his two books:

- *The Theory and Practice of Concurrency* (1997),
- *Understanding Concurrent Systems* (2010).

Main Features of Roscoe's "Standard" Solution

Roscoe (see books) presents the "standard" version of CSP, this presents one way to solve the problems with ✓ and termination.

- New view of termination as a special *signal* event: ✓ is now non-delayable by the environment.

- Impacts on refusals & failures: if a process has the trace $s^\frown\langle ✓ \rangle$, it has the failure (s, Σ).

- Wants the law: $P; SKIP \equiv P$, which does not hold if $P = Q \,\Box\, SKIP$ is allowed.

  Solves with *sliding choice* operator $\rhd$:

  $$P \,\Box\, SKIP = P \rhd SKIP \qquad (\Box - SKIP \text{ resolve})$$

- If ✓ occurs is final event of a trace, for both non-divergent and divergent traces.

- Above results in a modified collection of process axioms.

- Uses "distributed" (asynchronous) parallel termination semantics.

Introduction to CSP$_T$

Aim: provide a more robust treatment of termination through the consistent and special handling of ✓ by the language (processes and operators) and semantics (failures and divergences).

- Based on Brookes and Roscoe's *improved failure-divergence model* for CSP.

- CSP$_T$ defined by adding a new process axiom that captured our view of termination to original process axioms.

- View of tick (✓) is consistent with Hoare's, i.e. that it is a normal event, and not a *signal* event.

- Three new forms of generalised parallel operators were defined, each with a different form of termination semantics:
  - Synchronous termination: $P \|_{\Delta} Q$
  - Asynchronous termination: $P \|\|_{\Theta} Q$
  - Race termination: $P |_{\Theta} Q$

- Replaced the original interleaving ($\|\|$), synchronous ($\|$) & alphabetised (${}_A\|_B$) parallel operators with the synchronous ($\|_{\Delta}$), asynchronous ($\|\|_{\Theta}$) & race ($|_{\Theta}$) operators.

CSP$_T$ Termination Axiom

View of successful termination captured by:

> A process's trace satisfies the ✓*-requirement* if a ✓ only occurs at the end of the trace.

Considered which processes this requirement should apply to:

- only non-divergent processes
- divergent & non-divergent processes
- only to the non-divergent traces of both divergent & non-divergent processes.

Selecting the third approach, led to the following termination axiom:

$$t \neq \langle\,\rangle \;\wedge\; (s^\frown\langle\checkmark\rangle^\frown t, \varnothing) \in F \;\Rightarrow\; s \in D \qquad \text{(T1)}$$

where s and t are traces, F and D are the failure and divergence sets respectively of a process.

This axiom means that if a process indicates that it has terminated (by means of the ✓) but continues to perform events (t), then it must have started diverging before it performed the ✓ (i.e. $s \in D$).

Timed CSP

Timed CSP was developed by Reed and Roscoe, in the late 80's, taking *time* as the non-negative reals: $\mathit{TIME} = [0, \infty)$.

Only needed to add the delayed form of the *SKIP* process: $\mathit{WAIT}\, t$, $(t \geq 0)$.

Reed's hierarchy of semantic models for Timed CSP:

$$\begin{tikzcd}
 & & TM_{FS} \arrow[d] \arrow[ddll] \arrow[ddrr] & & \\
 & & M_{FS} \arrow[dl] \arrow[dr] & & \\
TM_{F} \arrow[r] \arrow[ddrr] & M_{F} \arrow[dr] & & M_{S} \arrow[dl] & TM_{S} \arrow[l] \arrow[ddll] \\
 & & M_{T} & & \\
 & & TM_{T} \arrow[u] & &
\end{tikzcd}$$

There are several new notions that are central to the semantics of Timed CSP:

- *timed events* & *timed traces*,
- *timed refusal sets* & *timed failures*,
- *stability values*

Timed Events & Traces

Timed event is an ordered pair (t, a), where $a \in \Sigma$ and $t \in \mathit{TIME}$.

Timed trace is a finite sequence of timed events.

The events in the sequence are ordered chronologically.

For example, the process:

$\mathit{WAIT}\ 1; (a \rightarrow b \rightarrow \mathit{STOP})$

two possible traces are:

$\langle (t, a) \rangle$ for $1 \leq t$.

$\langle (2, a), (3, b) \rangle$

but since a can not occur before time 1

$\langle (0, a), (2, b) \rangle$

is not.

Timed Refusals & Failures

A CSP *failure*, (s, X), means the refusal set X may be refused *after* the process has performed the trace s.

In Timed CSP a *timed failure* $(s, \aleph)$, represents what a process may refuse:

- after the timed trace s,
- but also what can be refused during the trace s.

E.g. before the first event is performed, during the time between consecutive events or after the final event of the trace.

A timed *refusal token*: is one of these "snap shot" pieces of refusal information (with timings) at various stages during the execution of the associated timed trace.

A timed *refusal set*, $\aleph$, is a union of: "initial", "intermediate" and "final" *refusal tokens*.

A timed *failure*, $(s, \aleph)$, is then straightforwardly defined as a timed trace combined with a timed refusal.

Process performs the timed trace s while refusing sets of events during the time intervals described by the timed refusal $\aleph$.

Stability

Stability is used to model the internal activity of a process.

Dual of divergence as used in CSP.

A process is *stable* once it has ceased all internal activity.

A stable process cannot change state without performing an external event.

The *stability value*, α, associated with an observation (timed trace or failure) of a process is the earliest time by which all internal activity of the process is *guaranteed* to have stopped.

A process which diverges has a stability value of ∞.

TM_S stability value associated with every timed trace: (s, α).

TM_{FS} stability value associated with every timed failure: $(s, \alpha, \aleph)$.

<ins>Termination Issues in Timed CSP</ins>

Termination is such a basic property of a process that it should be captured by a process axiom.

Issues to be considered when defining a Timed CSP termination axiom:

- Ensure $\checkmark$s only occur as the last event in a timed trace.
  (Requires a timed trace version of our $\checkmark$-requirement.)

- The most significant new feature is stability & how it is used to model divergence versus a divergence trace.
  So problem traces resulting from divergence e.g. $s^\frown\langle\checkmark\rangle^\frown t$, no longer an issue.

- Stability at termination:
  Implicit notion of *"immediate stability at termination"*.
  Should it be zero or something else?

- Davies & Schneider's *timeout* & *interrupt* operators: rely on the race termination semantics of $|||$.
  (So need to add a timed version of $|_{\varnothing}$.)

Termination Axiom for *TM*$_{FS}$

Timed Failures-Stability model *TM*$_{FS}$ models processes by sets of timed failures paired with associated stabilities.

E.g. $(s, \alpha, \aleph)$, with timed trace s, stability value α & timed refusal set $\aleph$.

$$(s, \alpha, \aleph) \in \mathcal{S} \wedge \checkmark \in \Sigma(s) \qquad (TA_{FS})$$
$$\Rightarrow \quad s = s' ^\frown \langle(\alpha, \checkmark)\rangle \wedge \checkmark \notin \Sigma(s') \wedge (s, \alpha, \aleph \cup \aleph_1) \in \mathcal{S}$$

where $\Sigma(s)$ is the set of events in s, the time interval covered by the refusal set $\aleph_1$ is from α to ∞.

TA$_{FS}$ means that if a process can perform the timed trace s with a stability value of α while refusing $\aleph$ and a $\checkmark$ has occurred in the trace then:

- the $\checkmark$-requirement is satisfied
- the $\checkmark$ occurred at the time of stability α
- that from time α it can henceforth refuse all further events.

Conclusions

- Begun the process of providing an improved treatment of successful termination in Timed CSP.
- Identified a number of issues that need to be considered when choosing a termination axiom for each of the four Timed CSP models.
- Identified stability as the most significant new issue.
- Proposed termination axioms for each of the four Timed CSP models.

For more details on CSP$_T$ and a comparison with Roscoe's standard CSP & other solutions see our two previous papers:

A CSP model with flexible parallel termination semantics, Formal Aspects of Computing, 21, No. 5, pp421–449, 2009. DOI: 10.1007/s00165-008-0098-z

Specifying Termination in CSP, Theoretical Computer Science, 2013. DOI: 10.1016/j.tcs.2013.05.008

Further Work

- Do the maths & add the axioms to each of the 4 timed models.
- Define timed versions of $\|_{\Delta}$, $|||_{\Theta}$ and $|_{\Theta}$, for each of the timed models, as replacements for existing operators.
- Are there any more issues that should be taken into account when considering the termination of Timed CSP processes?
- Are there alternative axioms, as there were when considering CSP$_T$?
- Are there different types of successful termination?

Appendix A: Roscoe's New CSP Process Axioms

(See Roscoe's latest book.)

$$\mathsf{traces}_\perp(P) \;=\; \{\, t \mid (t,X) \in F \,\}$$

$$\text{[is non-empty and prefix closed.]} \qquad \text{(F1)}$$

$$(s,X) \in F \wedge Y \subseteq X \;\Rightarrow\; (s,Y) \in F \qquad \text{(F2)}$$

$$(s,X) \in F \wedge (\forall\, a \in Y : s^\frown\langle a\rangle \notin \mathsf{traces}_\perp(P)) \;\Rightarrow\; (s, X \cup Y) \in F \qquad \text{(F3)}$$

$$s^\frown\langle \checkmark \rangle \in \mathsf{traces}_\perp(P) \;\Rightarrow\; (s,\Sigma) \in F \qquad \text{(F4)}$$

$$s \in D \wedge t \in \Sigma^* \;\Rightarrow\; s^\frown t \in D \qquad \text{(D1)}$$

$$s \in D \;\Rightarrow\; (s,X) \in F \qquad \text{(D2)}$$

The axioms (F1) to (F3) and (D2) are similar to (N1) to (N4) and (D2) respectively and so (N5) is no longer necessary.

Roscoe states that the new axioms (F4) and (D1) reflect the special role of ✓ and that he does not wish to distinguish between how processes behave after successful termination.

Axiom (F4) means that if a process can terminate then it can refuse to do anything but terminate.

Axiom (D1) provides divergence closure as ✓ can only occur as a final event.

Appendix B: CSP$_T$ Process Axioms

(D1) – (N5) are the original *process axioms* taken from Brookes & Roscoe *An improved failures model for Communicating Sequential Processes*, plus our CSP$_T$ *Termination* axiom (T1).

$$s \in D \Rightarrow s^\frown t \in D \tag{D1}$$

$$s \in D \Rightarrow (s^\frown t, X) \in F \tag{D2}$$

$$(\langle\,\rangle, \varnothing) \in F \tag{N1}$$

$$(s^\frown t, \varnothing) \in F \Rightarrow (s, \varnothing) \in F \tag{N2}$$

$$(s, X) \in F \land Y \subseteq X \Rightarrow (s, Y) \in F \tag{N3}$$

$$(s, X) \in F \land (\forall c \in Y : (s^\frown\langle c\rangle, \varnothing) \notin F) \Rightarrow (s, X \cup Y) \in F \tag{N4}$$

$$(\forall Y \in \mathbb{F}(X) : (s, Y) \in F) \Rightarrow (s, X) \in F \tag{N5}$$

$$t \neq \langle\,\rangle \land (s^\frown\langle\checkmark\rangle^\frown t, \varnothing) \in F \;\Rightarrow\; s \in D \tag{T1}$$

(D1) states that the divergence set of a process is suffix closed. This captures the idea that once a process has started to diverge it does so for ever and that it is impossible for the process to recover, i.e. stop diverging, by perform some event, even ✓.

(D2) implies that if a process is diverging then it may also fail, i.e. it may refuse any set of events offered to it at any later stage. This captures the totally nondeterministic and chaotic nature of a diverging process in that it is seen as being *catastrophic*. This axiom enforces the consistency requirement between the divergence set and the failure set of a process.

(N1) and (N2) together imply that the traces of a process form a non-empty prefix closed set, i.e. the traces of a process form a *tree*.

(N3) if a process can refuse a set of events X then it can refuse all the subsets of X. If a process is unable to perform any of the events in X then it could not perform a subset of them.

(N4) if at some point a process can refuse the set of events X and there is another set of events Y that it also can can refuse at that point then clearly the process can refuse both of them together, i.e. $X \cup Y$. Basically this means that if it is impossible for a process to perform an event at some point then it can be added to the refusal set at that point.

(N5) means that if a process can refuse all of the finite subsets Y of a (possibly infinite) set X then it can also refuse the set X. This is a *closure* property for refusal sets which allows us to deduce that infinite sets are refusable if all of their finite subsets are refusable.

(T1) for an explanation, see previous slides.

Appendix C: CSP$_T$'s 3 New Parallel Operators

Necessary to define replacements for synchronous ($\|$), interleaving ($|||$) and alphabetised (${}_A\|_B$) parallel operators, as $|||$ & ${}_A\|_B$ do not satisfy (T1).

New operators are *generalised* (or *interface*) style, parameterised by synchronisation sets Δ & Θ.

Three operators have distinct types of parallel termination semantics.

Synchronous ($\|_\Delta$): requires the successful termination of both P & Q, synchronised termination on $\checkmark$ ($\checkmark \in \Delta$).

Asynchronous ($|||_\Theta$): requires the successful termination of both P & Q, terminate asynchronously & do not synchronise on $\checkmark$ ($\checkmark \notin \Delta$).

Race ($|_\Theta$): requires the successful termination of either P or Q, terminate asynchronously & do not synchronise on $\checkmark$ ($\checkmark \notin \Delta$).

Fails to termination only if both P & Q fail to terminate.

Whichever of P or Q terminates first, terminates $P|_\Theta Q$, the other process is aborted.

Appendix D: CSP$_T$ Language

The language for CSP$_T$ is the same as CSP, except that it uses the three new parallel operators: $P \|_{\triangle} Q$, $P |||_{\Theta} Q$ and $P |_{\Theta} Q$ as replacements for $\|$, $|||$ and ${}_A\|_B$.

$$
\begin{aligned}
P ::= {} & \bot \mid \mathit{STOP} \mid \mathit{SKIP} \\
& \mid a \to P \mid P \sqcap P \mid P \,\Box\, P \mid P ; P \mid P \backslash a \mid P[\![R]\!] \\
& \mid \mu\, p.F(p) \mid p \\
& \mid P \|_{\triangle} P \mid P |||_{\Theta} P \mid P |_{\Theta} P
\end{aligned}
$$

where $a \in \Sigma - \{\checkmark\}$ & $F(p)$ is a CSP term used to define recursive processes.

Appendix E: Termination Axiom for TM_S

In Timed Stability model TM_S processes by denoted by sets of pairs (s, α).

A stability value α is associated with a timed trace s.

$$(s, \alpha) \in \mathcal{S} \wedge \checkmark \in \Sigma(s) \;\Rightarrow \tilde{s} = s' \frown \langle(\alpha, \checkmark)\rangle \wedge \checkmark \notin \Sigma(s') \qquad (TA_S)$$

where $\Sigma(s)$ is the set of events in the timed trace s.

TA_S means that if a process can perform the timed trace s with a stability value of α and a ✓ has occurred in the trace then:

- the ✓-requirement is satisfied,
- the ✓ occurred at time α, the time of stability.

This axiom and the trace prefix closure axiom together would ensure that the ✓-requirement was enforced and that stability on termination was maintained.