Synthesizing Safe Smart Contracts using Session Types

ANONYMOUS AUTHOR(S)

Abstract

CCS Concepts: •**Software and its engineering** → **General programming languages;** •**Social and professional topics** → *History of programming languages;*

Additional Key Words and Phrases: keyword1, keyword2, keyword3

ACM Reference format:
Anonymous Author(s). 2017. Synthesizing Safe Smart Contracts using Session Types. *PACM Progr. Lang.* 1, 1, Article 1 (January 2017), 3 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

$$E : \text{Append-only list}$$
$$R_i : \text{A Queue for actor } i$$

$$\begin{array}{lcl} \langle cmd \rangle & ::= & \text{publish } \langle V \rangle \\ & | & \text{yield; take } \langle T \rangle \\ & | & \text{if * then } \langle cmdList \rangle \text{ else } \langle cmdList \rangle \\ & | & \text{while * do } \langle cmdList \rangle \end{array}$$

$$\frac{-}{(E, (i, L, M) :: R, \Sigma, \text{take L}.P) \rightsquigarrow (E, R, \Sigma, P)}\, TAKE \qquad \frac{\Sigma \vdash v \rightsquigarrow x}{(E, R, \Sigma, \text{publish v}.P) \rightsquigarrow (x :: E, R, \Sigma, P)}\, PUB$$

$$\frac{L' \neq L}{(E, (i, L', M) :: R, \Sigma, \text{take L}.P) \rightsquigarrow (E, R, \Sigma, \text{take L}.P)}\, DROP \qquad \frac{}{(E, R, \Sigma, \text{yield; take T}.P) \rightsquigarrow (E, R, \Sigma, \text{takes T}.P)}\, YI$$

$$\begin{array}{lcl} \langle cmd \rangle & ::= & \text{send } \langle V \rangle \text{: } \langle T \rangle \\ & | & \text{read latest } \langle T \rangle \\ & | & \text{deq } \langle T \rangle \\ & | & \text{if * then } \langle cmdList \rangle \text{ else } \langle cmdList \rangle \\ & | & \text{while * do } \langle cmdList \rangle \end{array}$$

$$\frac{\forall M', (L, M') \notin E'}{(E'.(L, M).E, R_i, \Phi, \text{read latest L}.P) \rightsquigarrow (E, R_i, \Phi, P)}\, RL \qquad \frac{\Sigma \vdash v \rightsquigarrow x}{(E, R_i, \Phi, \text{send v: L}.P) \rightsquigarrow (E, (i, L, x) :: R_i, \Phi, P)}\, SEND$$

$$\frac{}{((T, M) :: E, R_i, \Phi, \text{deq T}.P) \rightsquigarrow (E, R_i, \Phi, P)}\, DEQ \qquad \frac{-}{-}\, YIELD$$

A note.

2017. 2475-1421/2017/1-ART1 $15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

1 NOTATIONS

We write _ to denote an immaterial value, which is implicitly existentially quantified, and $\perp$ to denote the undefined value. We denote the *size* (number of elements) of a set A by $|A|$. We write $f : A \rightarrow B$ and $f : A \rightharpoonup B$ to denote a *total*, respectively, *partial*, function from A to B. We denote the *domain of definition* and *range* of a function $f : A \rightharpoonup B$ by $\mathrm{dom}(f)$ and $\mathrm{range}(f)$, respectively, i.e., $\mathrm{dom}(f) = \{a \in A \mid f(a) \neq \perp\}$ and $\mathrm{range}(f) = \{b \in B \mid \exists a.\ f(a) = b\}$. We write $f : A \rightharpoonup_{\mathit{fin}} B$ to denote that f has a finite domain. We denote the set of natural numbers (including zero) by $\mathbb{N}$. We write $\{m..n\}$, for some $m, n \in \mathbb{N}$, to denote the set of integers $\{i \in \mathbb{N} \mid m \leq i \wedge i \leq n\}$. A *sequence* $\pi = a_1, \ldots, a_n$ *over a set* A is a function $\pi : \{1..n\} \rightarrow A$, from $\{1..n\}$, for some $n \in \mathbb{N}$, to A. We denote the *length* of π by $|\pi| = |\,\mathrm{dom}(\pi)|$, and its ith element, for $i \in \{1..|\pi|\}$, by $\pi(i)$. We denote the *empty sequence* by ϵ, and the concatenation of sequences π_1 and π_2 by $\pi_1 \cdot \pi_2$. We denote the set of sequences over a set A by $\overline{A}$.

REFERENCES