Ethereum Yellow Paper

2. The Blockchain Paradigm

1. $$\sigma_{t+1} = \Upsilon(\sigma_t, T)$$

   - σ_t : world-state at time t
   - T : transaction
   - Υ : Ethereum state transition function

2. $$\sigma_{t+1} = \Pi(\sigma_t, B)$$

   $$B = (\ldots, (T_0, T_1, \ldots), \ldots)$$

   $$\Pi(\sigma, B) = \Omega(B, \Upsilon(\Upsilon(\sigma, T_0), T_1)\ldots)$$

   - B : block
   - Ω : block-finalisation state transition func.
   - Π : block-level state transition function

4. Blocks, State and Transactions

4.1. World State

1. σ : address $\mapsto$ account_state

- address : 160-bit identifier
- account_state : RLP-serialised data structure

2. $\sigma[a] = (\sigma[a]_n, \sigma[a]_b,$
$\sigma[a]_s, \sigma[a]_c)$

- $\sigma[a]_n$: nonce ; a scalar value equal to
  - the number of transactions sent by a
  - the number of contract-creation made by a, if a is a contract
- $\sigma[a]_b$: balance ; a scalar value equal to the number of Wei owned by a
- $\sigma[a]_s$: a 256-bit hash of the root node of a trie that encodes the storage contents - a mapping between 256-bit integer values : storageRoot

4.1. cont'd

- $\sigma[a]_c$: codeHash ; the hash of the EVM code of a

$KEC(b) = \sigma[a]_c$: $b :=$ the code of a

a : simple account $:= \sigma[a]_c = KEC(())$

1. $L_s(\sigma) = \{p(a) : \sigma[a] \neq \emptyset\}$

$p(a) = (KEC(a), RLP((\sigma[a]_n, \sigma[a]_b, \sigma[a]_s, \sigma[a]_c)))$

2. $EMPTY(\sigma, a)$

$= \sigma[a]_c = KEC(()) \wedge \sigma[a]_n = 0 \wedge \sigma[a]_b = 0$

3. $DEAD(\sigma, a) = \sigma[a] = \emptyset \vee EMPTY(\sigma, a)$

4.2. The Transaction

T: transaction $:=$ T: message call
$\lor$ T: contract creation

1. $$L_T(T) = \begin{cases} (T_n, T_p, T_g, T_t, T_v, p, T_w, T_r, T_s), & T_x = 0 \\ (T_c, T_n, T_p, T_g, T_t, T_v, p, T_A, T_y, T_r, T_s), & T_x = 1 \end{cases}$$

$$p = \begin{cases} T_i, & T_t = \emptyset \\ T_d, & T_t \neq \emptyset \end{cases}$$

4.3. The Block

1. $$B = (B_H, B_T, B_U)$$

2. $$L_B(B) = (L_H(B_H), \tilde{L}_T^*(B_T), L_H^*(B_U))$$

$$L_H(H) = (H_p, H_o, H_c, H_r, H_t, H_e, H_b, H_d, H_i, H_l, H_g, H_s, H_x, H_m, H_n)$$

$$\tilde{L}_T(T) = \begin{cases} L_T(T), & T_x = 0 \\ (T_x) \cdot \mathrm{RLP}(L_T(T)), & T_x \neq 0 \end{cases}$$

$\cdot$: byte array concatenation

$f^*((x_0, x_1, \ldots)) := (f(x_0), f(x_1), \ldots)$: f: function

6. Transaction Execution

1. $\sigma' = \Upsilon(\sigma, T)$

6.1. Substate

1. $A = (A_s, A_l, A_t, A_r, A_a, A_K)$

2. $A^0 = (\varnothing, (), \varnothing, 0, \pi, \varnothing)$

6.2. Execution

1. σ_0 : checkpoint state ;

$$\sigma_0 = \sigma \text{ except :}$$
$$\sigma_0[S(T)]_b = \sigma[S(T)]_b - T_g T_p$$
$$\sigma_0[S(T)]_n = \sigma[S(T)]_n + 1$$

2. σ_p : post-execution provisional state

$$(\sigma_p, g', A, z) = \begin{cases} \Lambda_4(\sigma_0, A^*, S(T), S(T), g, \\ \quad T_p, T_v, T_i, 0, \varnothing, \top), & T_t = \varnothing \\ \Theta_4(\sigma_0, A^*, S(T), S(T), T_t, T_t, \\ \quad g, T_p, T_v, T_v, T_d, 0, \top), & T_t \neq \varnothing \end{cases}$$

6.2. cont'd

3. σ^* : pre-final state

$\sigma^* = \sigma_p$ except :

$$\sigma^*[S(T)]_b = \sigma_p[S(T)]_b + g^* T_p$$

$$\sigma^*[m]_b = \sigma_p[m]_b + (T_g - g^*) T_p$$

$$: m = B_{Hc}$$

4. σ' : final state

$\sigma' = \sigma^*$ except :

$\sigma'[i] = \emptyset \ : i \in \mathcal{A}_s$

$\sigma'[i] = \emptyset \ : i \in \mathcal{A}_t \ : \mathrm{DEAD}(\sigma^*, i)$

7. Contract Creation

1. $(\sigma', g', A', z, o) = \Lambda(\sigma, A, s, o, g, p, v, i, e, \zeta, w)$

2. $\alpha = ADDR(s, \sigma[s]_n - 1, \zeta, i)$

$$ADDR(s, n, \zeta, i) = \mathcal{B}_{96..255}(KEC(L_A(s, n, \zeta, i)))$$

$$L_A(s, n, \zeta, i) = \begin{cases} RLP(s, n), & \zeta = \varnothing \\ (255) \cdot s \cdot \zeta \cdot KEC(i), & o/w \end{cases}$$

3. $\sigma \to \sigma'$

- $\sigma^* = \sigma$ except:

$$\sigma^*[\alpha] = (1, v + v', TRIE(\varnothing), KEC(()))$$

$$\sigma^*[s] = \begin{cases} \varnothing, & \sigma[s] = \varnothing \wedge v = 0 \\ \alpha^*, & o/w \end{cases}$$

$$\alpha^* = (\sigma[s]_n, \sigma[s]_b - v, \sigma[s]_s, \sigma[s]_c)$$

$$v' = \begin{cases} 0, & \sigma[\alpha] = \varnothing \\ \sigma[\alpha]_b, & o/w \end{cases}$$

- $(\sigma^{**}, g^{**}, A^{**}, o) = \Xi(\sigma^*, g, A^*, I)$

$I_b = i$, $I_d = ()$, $I_x = x \,:\, x \in \{\alpha, o, p, s, v, e, w\}$

7. cont'd :

3. $\sigma \rightarrow \sigma'$: cont'd

$$\sigma' = \begin{cases} \sigma, & \mathcal{F} \vee \sigma^{**} = \varnothing \\ \sigma^{**} \text{ except: } \sigma'[a] = \varnothing, & \mathrm{DEAD}(\sigma^{**}, a) \\ \sigma^{**} \text{ except: } \sigma'[a]_c = \mathrm{KEC}(o), & \text{o/w} \end{cases}$$

$$\mathcal{F} = (\sigma[a] \neq \varnothing \wedge (\sigma[a]_c \neq \mathrm{KEC}(()) \vee \sigma[a]_n \neq 0))$$

$$\vee \ (\sigma^{**} = \varnothing \ \wedge \ o = \varnothing)$$

$$\vee \ g^{**} < c \ : \ c = G_{codedeposit} \times \|o\|$$

$$\vee \ \|o\| > 24576$$

8. Message Call

1. $(\sigma', g', A', z, o) = \Theta(\sigma, A, s, o, r, c, g, p, v, \tilde{v}, d, e, w)$

2. $\sigma \rightarrow \sigma'$

- $\sigma_1' = \sigma$ except:

$$\sigma_1'[r] = \begin{cases} (0, v, \mathrm{TRIE}(\varnothing), \mathrm{KEC}(())), & \sigma[r] = \varnothing \wedge v \neq 0 \\ \varnothing, & \sigma[r] = \varnothing \wedge v = 0 \\ a_1', & \text{o/w} \end{cases}$$

$$a_1' = (\sigma[r]_n, \sigma[r]_b + v, \sigma[r]_s, \sigma[r]_c)$$

- $\sigma_1 = \sigma_1'$ except:

$$\sigma_1[s] = \begin{cases} \varnothing, & \sigma_1'[s] = \varnothing \wedge v = 0 \\ a_1, & \text{o/w} \end{cases}$$

$$a_1 = (\sigma_1'[s]_n, \sigma_1'[s]_b - v, \sigma_1'[s]_s, \sigma_1'[s]_c)$$

8. cont'd

3. $\sigma \rightarrow \sigma'$: cont'd

▸ $(\sigma^{**}, g^{**}, A^{**}, o) = \Xi$:

$$\Xi = \begin{cases} \Xi_{ECREC}(\sigma_1, g, A, I), & c = 1 \\ \Xi_{SHA256}(\sigma_1, g, A, I), & c = 2 \\ \Xi_{RIP160}(\sigma_1, g, A, I), & c = 3 \\ \Xi_{ID}(\sigma_1, g, A, I), & c = 4 \\ \Xi_{EXPMOD}(\sigma_1, g, A, I), & c = 5 \\ \Xi_{BN\_ADD}(\sigma_1, g, A, I), & c = 6 \\ \Xi_{BN\_MUL}(\sigma_1, g, A, I), & c = 7 \\ \Xi_{SNARKV}(\sigma_1, g, A, I), & c = 8 \\ \Xi_{BLAKE2\_F}(\sigma_1, g, A, I), & c = 9 \\ \Xi(\sigma_1, g, A, I), & o/w \end{cases}$$

$: \; I_a = r, \; I_v = \tilde{v}, \; KEC(I_b) = \sigma[c]_c$

$I_x = x \; : \; x \in \{o, p, d, s, e, w\}$

▸ $$\sigma' = \begin{cases} \sigma, & \sigma^{**} = \emptyset \\ \sigma^{**}, & o/w \end{cases}$$

9. Execution Model

1. $(\sigma', g', A', o) = \Xi(\sigma, g, A, I)$:

   $I = (I_a, I_o, I_p, I_d, I_s, I_v, I_b, I_H, I_e, I_w)$

2. Overview

   $\mu = (g, pc, m, i, s)$

▸ $\Xi(\sigma, g, A, I) = (\sigma', \mu'_g, A', o)$:

$(\sigma', \mu', A', \ldots, o) = X((\sigma, \mu, A, I))$

$: \mu = (g, 0, (0, 0, \ldots), 0, ())$

▸ $$X((\sigma,\mu,A,I)) = \begin{cases} (\varnothing, \mu, A, I, \varnothing), & Z(\sigma,\mu,A,I) \\ (\varnothing, \mu', A, I, o), & w = REVERT \\ O(\sigma,\mu,A,I) \cdot o, & o \neq \varnothing \\ X(O(\sigma,\mu,A,I)), & o/w \end{cases}$$

$: o = H(\mu, I)$, $(\ldots, d) \cdot e = (\ldots, d, e)$,

$\mu' = \mu$ except : $\mu'_g = \mu_g - C(\sigma, \mu, A, I)$

9. cont'd

2. Overview : cont'd

$$w = \begin{cases} I_b[\mu_{pc}], & \mu_{pc} < \|I_b\| \\ \text{STOP}, & \text{o/w} \end{cases}$$

- $Z(\sigma, \mu, A, I) = \mu_g < C(\sigma, \mu, A, I)$

$$\vee\ \delta_w = \varnothing \quad \vee\ \|\mu_s\| < \delta_w$$

$$\vee\ (w = \text{JUMP} \ \wedge\ \mu_s[0] \notin D(I_b))$$

$$\vee\ (w = \text{JUMPI} \ \wedge\ \mu_s[1] \neq 0 \ \wedge\ \mu_s[0] \notin D(I_b))$$

$$\vee\ (w = \text{RETURNDATACOPY} \ \wedge\ \mu_s[1] + \mu_s[2] > \|\mu_o\|)$$

$$\vee\ \|\mu_s\| - \delta_w + \alpha_w > 1024$$

$$\vee\ (\neg I_w \wedge W(w, \mu))$$

$$\vee\ (w = \text{SSTORE} \ \wedge\ \mu_g \leqslant G_{callstipend})$$

9. cont'd

2. Overview : cont'd

- $$\mathcal{H}(\mu, I) = \begin{cases} \mathcal{H}_{RETURN}(\mu), & w \in \{RETURN, REVERT\} \\ (), & w \in \{STOP, SELFDESTRUCT\} \\ \varnothing, & o/w \end{cases}$$

- $$O((\sigma, \mu, A, I)) = (\sigma', \mu', A', I)$$

$$\Delta = \alpha_w - \delta_w$$

$$\|\mu'_s\| = \|\mu_s\| + \Delta$$

$$\mu'_s[x] = \mu_s[x - \Delta] \; : \; x \in [\alpha_w, \|\mu'_s\|)$$

Appx H. Virtual Machine Specification

0x54 : SLOAD : $\mu'_s[0] = \sigma[I_a]_s[\mu_s[0]]$

0x55 : SSTORE : $\sigma'[I_a]_s[\mu_s[0]] = \mu_s[1]$