Protocol Π_{Chan}

```
Initialisation:
    $State \leftarrow$ INIT

On TOP UP, CHECK TOP UP by $\mathcal{E}$, act as $\mathcal{F}_{\text{Chan}}$ (Fig. 3, lines 4-8 and 9-15
respectively)

On (OPEN, $c_F$, $pk_{A,out}$, $pk_{B,out}$) by $\mathcal{E}$:
    ensure $State =$ TOPPED UP
    $State \leftarrow$ OPENING BASE CHANNEL
    do LN (other box)

On (CHECK FUNDING) by $\mathcal{E}$:
    ensure $State =$ WAITING FOR LEDGER
    send (READ) to $\mathcal{G}_{\text{Ledger}}$ and assign reply to $\Sigma$
    ensure $F \in \Sigma$
    $c_A \leftarrow c$; $c_B \leftarrow 0$ // $c$ received in OPEN
    $State \leftarrow$ OPEN BASE
    output (OPEN SUCCESS) to $\mathcal{E}$

On (PAY, x) by $\mathcal{E}$:
    ensure $State \in \{$OPEN BASE, OPEN VIRTUAL$\}$
    ensure $c_A \geq x$
    do LN payment (these channels won't be async) (balance change here)
    output (OK) to $\mathcal{E}$

On (BALANCE) by $\mathcal{E}$:
    ensure $State \in \{$OPEN BASE, OPEN VIRTUAL$\}$
    output (BALANCE, $(c_A, c_B, \text{locked}_A, \text{locked}_B)$) to $\mathcal{E}$

On (CLOSE) by $\mathcal{E}$:
    if $State =$ OPEN BASE then
        prepare $C$ TODO
        send (SUBMIT, $C$) to $\mathcal{G}_{\text{Ledger}}$
    else if $State =$ OPEN VIRTUAL then
        TODO
    end if
```

Fig. 1.

Protocol $\mathit{\Pi}_{\text{Chan}}$ – virtual

```
// notification to funder
// trust that $Alice$ has $c$ in her channel
On (FUND YOU, $c$, $Bob$) by $Charlie$ as input:
    ensure $State = \text{INIT}$
    $State \leftarrow$ OPENING VIRTUAL CHANNEL
    do LN with $Bob$ – TODO
    output (OK) to $Charlie$

On (FUND, $c$, hops, sub_parties = (fundee, counterparty), outer_parties = ($Charlie$, $Dave$) by $\mathcal{E}$:
    ensure $State \in \{\text{OPEN BASE}, \text{OPEN VIRTUAL}\}$
    do the same as in $\mathcal{F}_{\text{Chan}}$ (Fig. 6, lines 9-24) TODO: make sure it makes sense
    do VChan() with hops – TODO // $P_{i-1}P_i, P_iP_{i+1}$ and all $P_1P_n$ held by BOTH $R_{i-1}$ and $L_i$. $P_{i-1}P_i$ held only by $R_{i-1}$, $P_iP_{i+1}$ held only by $L_i$. This (probably) ensures that only relevant parties can close their channels (with the exception of honest $R_{i-1}$ wanting to leave channels virtual but corrupted $L_i$ demoting them to base, which however doesn't cost funds to anyone), but that they have minimal impact to the decisions of ajdacent channels. All $P_{i-1}P_i$ inputs must be signed by $R_{i-1}$ and all $P_iP_{i+1}$ inputs by $L_i$.
    output (OK) to $\mathcal{E}$

// notification to fundee
On (ALLOW FUND, ...) by $Charlie$, act as $\mathcal{F}_{\text{Chan}}$ (Fig 6, line 26):
```

Fig. 2.

Functionality $\mathcal{F}_{\text{Chan}}$ – init & top up

```
Initialisation: // runs on first activation
    $\mathit{State} \leftarrow \textsc{init}$
    $(\text{locked}_A, \text{locked}_B) \leftarrow (0, 0)$

On ($\textsc{top up}$, $c_{\min}$) by $\mathit{Alice}$:
    ensure $\mathit{State} = \textsc{init}$
    $\mathit{State} \leftarrow \textsc{sent key}$
    $(\mathit{sk}, \mathit{pk}) \leftarrow \textsc{KeyGen}()$
    output ($\textsc{public key}$, $\mathit{pk}$) to $\mathit{Alice}$

On ($\textsc{check top up}$) by $\mathit{Alice}$:
    ensure $\mathit{State} = \textsc{sent key}$
    send ($\textsc{read}$) to $\mathcal{G}_{\text{Ledger}}$ as $\mathit{Alice}$ and assign reply to $\Sigma$
    ensure $\exists \text{tx} \in \Sigma, c_{\text{on}} : c_{\text{on}} \geq c_{\min} \wedge (c_{\text{on}}, \mathit{pk}) \in \text{tx.outputs}$
    base_output $\leftarrow (c_{\text{on}}, \mathit{pk})$ of tx
    $\mathit{State} \leftarrow \textsc{topped up}$
    output ($\textsc{topped up}$) to $\mathit{Alice}$
```

Fig. 3.

Functionality $\mathcal{F}_{\text{Chan}}$ – base

```
On (OPEN, $c_F$, $pk_{A,\text{out}}$, $pk_{B,\text{out}}$) by $Alice$:
    ensure $State$ = TOPPED UP
    ensure $c_F \geq c_{\text{on}}$
    $(sk_{A,F}, pk_{A,F}) \leftarrow$ KEYGEN(); $(sk_{B,F}, pk_{B,F}) \leftarrow$ KEYGEN()
    $F \leftarrow$ TX {input: base_output, output: $(c_F, 2/\{pk_{A,F}, pk_{B,F}\})$}
    $F \leftarrow F.\text{sign}(sk)$
    $State \leftarrow$ WAITING FOR LEDGER
    send (OPEN, $c_F, pk_{A,\text{out}}, pk_{B,\text{out}}, F, Alice$) to $\mathcal{A}$ and ensure reply is OK
    output OK to $Alice$

On (CHECK FUNDING) by $Alice$:
    ensure $State$ = WAITING FOR LEDGER
    send (READ) to $\mathcal{G}_{\text{Ledger}}$ as $Alice$ and assign reply to $\Sigma$
    ensure $F \in \Sigma$
    $c_A \leftarrow c$; $c_B \leftarrow 0$
    $State \leftarrow$ OPEN BASE
    output (OPEN SUCCESS) to $Alice$

On (PAY, $x$) by $Dave \in \{Alice, Bob\}$:
    ensure $State \in$ {OPEN BASE, OPEN VIRTUAL}
    ensure $c_D - \text{locked}_D \geq x$
    send (PAY, $x$, $Dave$) to $\mathcal{A}$ and expect reply (OK)
    $c_D \leftarrow c_D - x; c_{\bar{D}} \leftarrow c_{\bar{D}} + x$ // $\bar{D}$ is $Alice$ if $D$ is $Bob$ and vice-versa
    output (PAY SUCCESS) to $Dave$

On (BALANCE) by $Dave \in \{Alice, Bob\}$:
    ensure $State \in$ {OPEN BASE, OPEN VIRTUAL}
    output (BALANCE, $(c_A, c_B, \text{locked}_A, \text{locked}_B)$) to $Dave$
```

Fig. 4.

Functionality $\mathcal{F}_{\text{Chan}}$ – close

```
On (CLOSE) by $Alice$:
    if $State$ = OPEN BASE then
        $C \leftarrow$ TX {input: $F$.out, outputs: $(c_A, pk_{A,\text{out}}), (c_B, pk_{B,\text{out}})$}
        $C \leftarrow C.\text{sign}(\text{sk}_{\text{A,F}}, \text{sk}_{\text{B,F}})$
        $State \leftarrow$ CLOSED
        input (SUBMIT, $C$) to $\mathcal{G}_{\text{Ledger}}$
    else if $State$ = OPEN VIRTUAL then
        $State \leftarrow$ CLOSED
        output (CLOSING, $c_A$, $c_B$) to opener
    end if

On (CLOSING, $c_{\text{left}}$, $c_{\text{right}}$) by $\mathcal{F}_{\text{Chan}}$:
    ensure $State \in$ {OPEN BASE, OPEN VIRTUAL}
    ensure $((c_L, c_R), \text{hops}, (Charlie, Dave), (Frank, George), \text{id}) \in$ funded with $Frank \in \{Alice, Bob\}$
    ensure $c_{\text{left}} \leq c_L + c_R$
    remove entry from funded
    output (CLOSED VIRTUAL, $c_{\text{right}}$, id) to $Frank$

On (CLOSED VIRTUAL, $c_{\text{right}}$, id) by $\mathcal{F}_{\text{Chan}}$:
    ensure $State \in$ {OPEN BASE, OPEN VIRTUAL}
    ensure $(\text{virtual}, c, \mathcal{F}_{\text{Chan}}, Dave, \text{id}) \in$ funded
    ensure $c_{\text{right}} \leq c$
    send (CLOSED) to virtual and expect reply YES
    $c_D \leftarrow c_D + c_{\text{right}}$
    remove entry from funded

On (CLOSED) by $P$:
    if $State$ = CLOSED then
        send (YES) to $P$
    else
        send (NO) to $P$
    end if
```

Fig. 5.

Functionality $\mathcal{F}_{\text{Chan}}$ – virtual

```
On (FUND YOU, $c$, $Dave$) by $Charlie$ as input to $Alice$: // $Alice$ is funded by $Charlie$
    ensure $State$ = INIT
    $Bob \leftarrow Dave$
    send (FUND YOU, $c$, $Bob$, $Charlie$, $Alice$) to $\mathcal{A}$ and ensure reply is (OK)
    $c_A \leftarrow c; c_B \leftarrow 0$
    opener $\leftarrow$ $Charlie$
    $State \leftarrow$ OPEN VIRTUAL
    output (OK) to $Charlie$

On (FUND, $c$, hops, sub_parties = (fundee, counterparty), outer_parties = ($Charlie$, $Dave$)) by $Alice$: // we fund another channel
    ensure $State \in \{$OPEN BASE, OPEN VIRTUAL$\}$
    ensure $c_A - \text{locked}_A \geq c$
    input (FUND YOU, $c$, counterparty) to fundee as $Alice$, ensure output is (OK)
    $(L_0, R_0) \leftarrow (Alice, Bob)$
    generate random id
    for all $(L_i, R_i) \in$ hops do // $i \in \{1, \dots, |\text{hops}|\}$
        ensure $R_{i-1} = L_i$
        send (ALLOW FUND, $c$, sub_parties, local_funder = $L_i$, id, $i \overset{?}{=} |\text{hops}|$) to $L_i$ as $Alice$ and ensure reply is (OK)
    end for
    send (FUND $c$, hops, sub_parties = (fundee, counterparty), outer_parties = ($Charlie$, $Dave$), funder = $Alice$) to $\mathcal{A}$ and ensure reply is OK
    for all $(L_i, R_i) \in$ hops do // $i \in \{1, \dots, |\text{hops}|\}$
        send (FUND DONE, id) to $L_i$ as $Alice$ and ensure reply is (OK)
    end for
    $c_A \leftarrow c_A - c$
    add (($c$, 0), hops, sub_parties, outer_parties, id) to funded
    output (OK) to $Alice$

On (ALLOW FUND, $c$, sub_parties, $D$, id, is_last) by $Charlie$:
    ensure $State \in \{$OPEN BASE, OPEN VIRTUAL$\}$
    ensure $D \in \{Alice, Bob\}$
    ensure $c_D - \text{locked}_D \geq c$
    output received message to $D$ and ensure reply is (OK)
    $\text{locked}_D \leftarrow \text{locked}_D + c$
    add (id, is_last) to pending
    send (OK) to $Charlie$

On (FUND DONE, id) by $Charlie$:
    ensure $State \in \{$OPEN BASE, OPEN VIRTUAL$\}$
    ensure (id, is_last) $\in$ pending
    remove (id, is_last) from pending
    if is_last then
        add ((0, $c$), $\bot$, sub_parties.reverse(), ($Dave$, $\bot$), id) to funded
    end if
    send (OK) to $Charlie$
```

Fig. 6.

Simulator $\mathcal{S}$

```
On $(\textsc{open}, c_F, pk_{A,\text{out}}, pk_{B,\text{out}}, F, \mathit{Alice})$ by $\mathcal{F}_{\text{Chan}}$:
    simulate $\mathit{Alice}$ receiving input $(\textsc{open}, c_F, pk_{A,\text{out}}, pk_{B,\text{out}})$ by $\mathcal{E}$
    ensure simulated $\mathit{Alice}$ outputs OK
    send OK to $\mathcal{F}_{\text{Chan}}$

On $(\textsc{pay}, x, \mathit{Dave})$ by $\mathcal{F}_{\text{Chan}}$:
    simulate $\mathit{Dave}$ receiving input $(\textsc{pay}, x)$ by $\mathcal{E}$
    ensure simulated $\mathit{Dave}$ outputs OK
    send OK to $\mathcal{F}_{\text{Chan}}$

On $(\textsc{fund you}, c, \mathit{Bob}, \mathit{Charlie}, \mathit{Alice})$ by $\mathcal{F}_{\text{Chan}}$:
    simulate $\mathit{Alice}$ receiving input $(\textsc{fund you}, c, \mathit{Bob})$ by $\mathit{Charlie}$
    ensure simulated $\mathit{Alice}$ outputs OK to $\mathit{Charlie}$
    send OK to $\mathcal{F}_{\text{Chan}}$
```

Fig. 7.

References