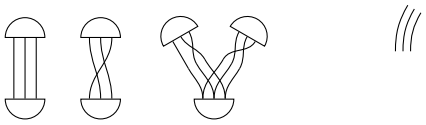
4 Nitro Protocol

Nitro protocol is an extension to Turbo protocol. In Nitro protocol, the outcome of a channel can be either an allocation or a **guarantee**. A guarantee outcome specifies the address of a target allocation channel; the protocol specifies how this guarantee may be used to pay debt on its behalf. When paying debt, a guarantee can be used to alter the payout priority of the allocation outcome of its target address.



We will use the notation $(L|A_1, A_2, \ldots, A_m)$ for a guarantee with target L, which prioritizes payouts to A_1 above A_2, A_2 above A_3, and so on. Any addresses which occur in the outcome of L but not in the guarantee are prioritized after A_m, in the order they occur in the outcome. We say a guarantor channel, G, which targets an allocation channel, L, 'can afford x for A', if A would receive at least x coins, were the coins currently held in A to be paid out according to G's reprioritization of L's allocation.

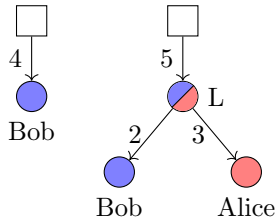
Nitro adds the **claim** operation, $C_{G,A}(x)$, to the existing transfer, deposit and withdraw operations. If G acts as guarantor for L and can afford x for A, then $C_{G,A}(x)$ has the following three effects:

- Reduces the funds held in channel G by x.
- Increases the funds held in channel A by x.
- Reduces the amount owed to A in the outcome of L by x.

Otherwise, the claim operation has no effect.

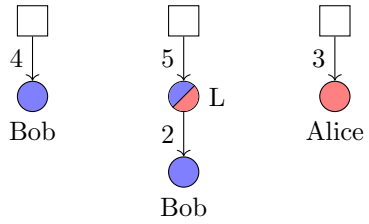
Adjudicator		
Address	Balance	Outcome
G	3	$(L\|\text{Alice})$
L		Bob: 2, Alice: 3

$[\![G : 3 \mapsto (L|\text{Alice}),\ L \mapsto (\text{Bob} : 2, \text{Alice} : 3)]\!]$



Adjudicator		
Address	Balance	Outcome
G		$(L\|\text{Alice})$
L		Bob: 2
Alice	3	

$[\![G \mapsto (L|\text{Alice}),\ L \mapsto (\text{Bob} : 2),\ \text{Alice} : 3]\!]$



Example 4.1. In the following example, we have a guarantor channel, G, which holds 5 coins and guarantees L's allocation, with B as highest priority.

$$C_{G,B}(5)[\![G : 5 \mapsto (L|B), L \mapsto (A : 5, B : 5)]\!] = [\![G \mapsto (L|B), L \mapsto (A : 5), B : 5]\!] \quad (1)$$

Note that after the claim has gone through, L's debt to B has decreased.

We give a python implementation of an adjudicator implementing the Nitro protocol in the appendix.

4.1 Virtual Channels

A virtual channel is a channel between two participants who do not have a shared on-chain deposit, supported through an intermediary. We will now give the construction for the simplest possible virtual channel, between A and B through a shared intermediary, I. Our starting point for this channel is a pair of ledger channels, L and L', with participants $\{A, I\}$ and $\{B, I\}$ respectively.

$$[\![L : x, L' : x]\!],\ [L \mapsto (A : a, I : b)]_{A,I},\ [L' \mapsto (B : b, I : a)]_{B,I} \quad (2)$$

where $x = a + b$. The participants want to use the existing deposits and ledger channels to fund a virtual channel, χ, with x coins.

In order to do this the participants will need three additional channels: a joint allocation channel, J, with participants $\{A, B, I\}$ and two guarantor channels G and G' which target J. The setup is shown in figure 1.

We will cover the steps for safely setting up this construction in section 4.3. In the next section, we will explain why this construction can be considered to fund the channel χ.

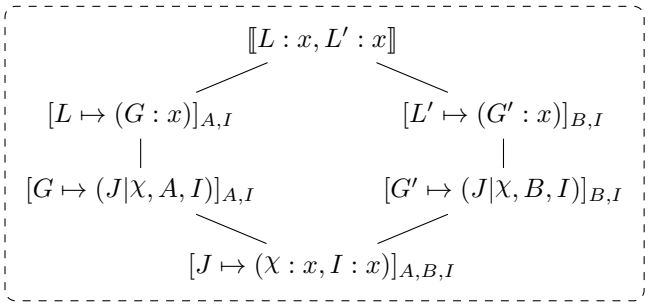


Figure 1: *Virtual channel construction*

4.2 Offloading Virtual Channels

Similarly to the method for ledger channel construction, we will show that the virtual channel construction funds χ by demonstrating how any one of the participants can offload the channel χ, thereby converting it to an on-chain channel that holds its own funds.

We will first consider the case where A wishes to offload χ. A proceeds as follows:

1. A starts by finalizing all their finalizable outcomes on-chain:

$$[\![L : x \mapsto (G : x),\ L' : x, G \mapsto (J|\chi, A, I),\ J \mapsto (\chi : x, I : x)]\!] \tag{3}$$

   Although A has the power to finalize L, G and J, they are not able to finalize L'. Thankfully, this does not prevent them from offloading χ.

2. A then calls $T_{L,G}(x)$ to move the funds from L to G:

$$[\![L' : x, G : x \mapsto (J|\chi, A, I),\ J \mapsto (\chi : x, I : x)]\!] \tag{4}$$

3. Finally A calls $C_{G,A}(\chi)$ to move the funds from G to χ.

$$[\![L' : x, G \mapsto (J|\chi, A, I),\ J \mapsto (I : x),\ \chi : x]\!] \tag{5}$$

As G has χ as top priority, the operation is successful.

By symmetry, the previous case also covers the case where B wants to offload. The final case to consider is the one where I wants to offload the channel and reclaim their funds. This is important to ensure that A and B cannot lock I's funds indefinitely in the channel.

1. I starts by finalizing all their finalizable outcomes on-chain:

$$[\![L : x \mapsto (G : x),\ L' : x \mapsto (G' : x), G \mapsto (J|\chi, A, I),\ G' \mapsto (J|\chi, B, I),\ J \mapsto (\chi : x, I : x)]\!] \tag{6}$$

2. I then transfers funds from the ledger channels to the virtual channels by calling $T_{L,G}(x)$ and $T_{L',G'}(x)$:

$$[\![G : x \mapsto (J|\chi, A, I),\ G' : x \mapsto (J|\chi, B, I),\ J \mapsto (\chi : x, I : x)]\!] \tag{7}$$

3. Then I claims on one of the guarantees, e.g. $C_{G,\chi}(x)$ to offload χ:

$$[\![G \mapsto (J|\chi, A, I),\ G' : x \mapsto (J|\chi, B, I),\ J \mapsto (I : x),\ \chi : x]\!] \tag{8}$$

4. After which, I can recover their funds by claiming on the other guarantee, $C_{G',I}(x)$:

$$[\![G \mapsto (J|\chi, A, I),\ G' \mapsto (J|\chi, B, I),\ \chi : x,\ I : x]\!] \tag{9}$$

Note that I has to claim on both guarantees, offloading χ before being able to reclaim their funds. The virtual channel became a direct channel and the intermediary was able to recover their collateral.

4.3 Opening and Closing Virtual Channels

In this section we present a sequence of network states written in terms of universally finalizable outcomes, where each state differs from the previous state only in one channel. We claim that this sequence of states can be used to derive a safe procedure for opening a virtual channel, where the value of the network remains unchanged throughout for all participants involved. We justify this claim in the appendix.

The procedure for opening a virtual channel is as follows:

1. Start in the state given in equation (2):

$$[\![L : x, L' : x]\!] \tag{10}$$
$$[L \mapsto (A : a, I : b)]_{A,I} \tag{11}$$
$$[L' \mapsto (B : b, I : a)]_{B,I} \tag{12}$$

2. A and B bring their channel χ to the funding point:

$$[\chi \mapsto (A : a, B : b)]_{A,B} \tag{13}$$

3. In any order, A, B and I setup the virtual channel construction:

$$[J \mapsto (A : a, B : b, I : x)]_{A,B,I} \tag{14}$$
$$[G \mapsto (J|\chi, A, I)]_{A,I} \tag{15}$$
$$[G' \mapsto (J|\chi, B, I)]_{B,I} \tag{16}$$

4. In either order switch the ledger channels over to fund the guarantees:

$$[L \mapsto (G : x)]_{A,I} \tag{17}$$
$$[L' \mapsto (G' : x)]_{B,I} \tag{18}$$

5. Switch J over to fund χ:

$$[J \mapsto (\chi : x, I : x)]_{A,B,I} \tag{19}$$

We give a visual representation of this procedure in figure 2.

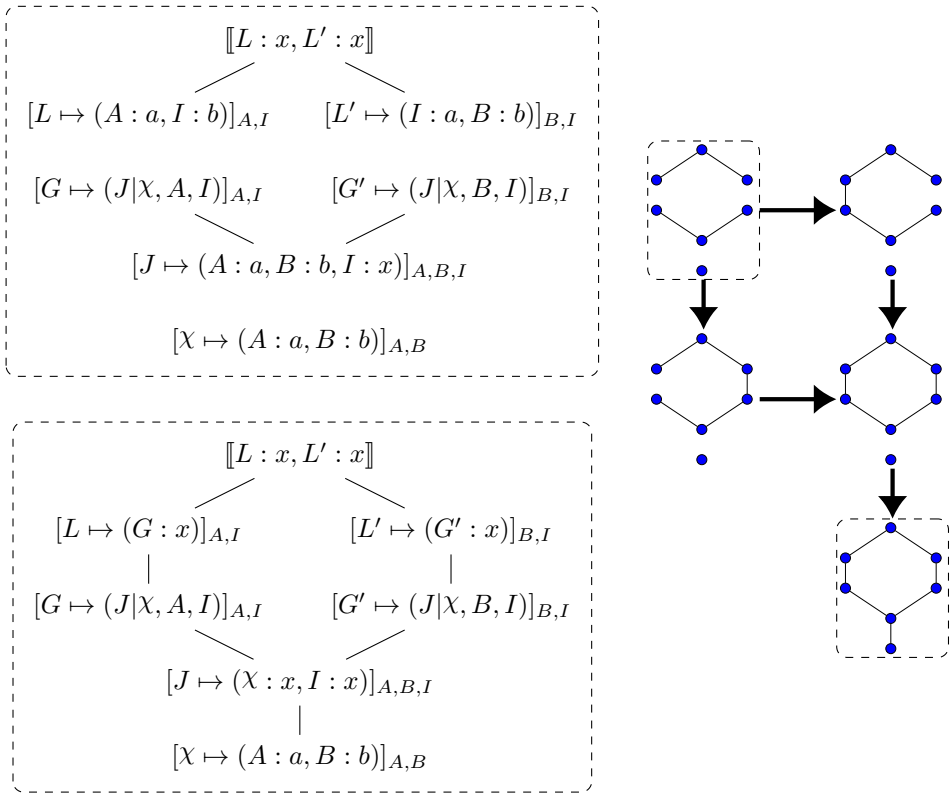
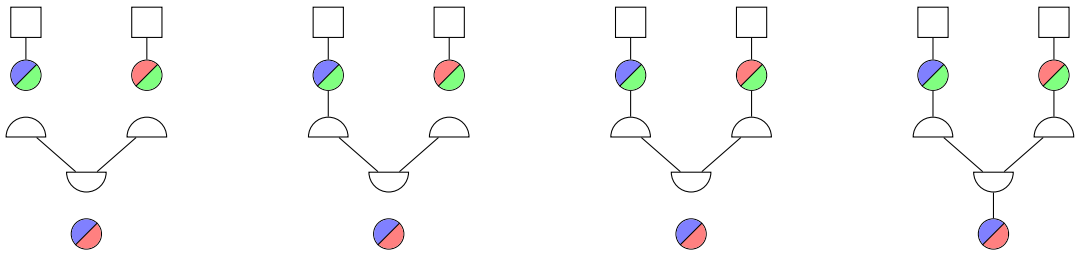


Figure 2: *Opening a virtual channel*



The same sequence of states, when taken in reverse, can be used to close a virtual channel:

1. Participants A and B finalize χ by signing a conclusion proof:

$$[\chi \mapsto (A : a', B : b')]_{A,B} \tag{20}$$

2. A and B sign an update to J to take account of the outcome of χ. I will accept this update, provided that their allocation of x coins remains the same:

$$[J \mapsto (A : a', B : b', I : x)]_{A,B,I} \tag{21}$$

3. In either order switch the ledger channels to absorb the outcome of J, defunding the guarantor channels in the process:

$$[L \mapsto (A : a', I : b')]_{A,I} \tag{22}$$
$$[L' \mapsto (B : b', I : a')]_{B,I} \tag{23}$$

4. The channels χ, J, G and G' are now all defunded, so can be discarded

It is also possible to do top-ups and partial checkouts from a virtual channel.