Architettura del nodo di Massa

Nota

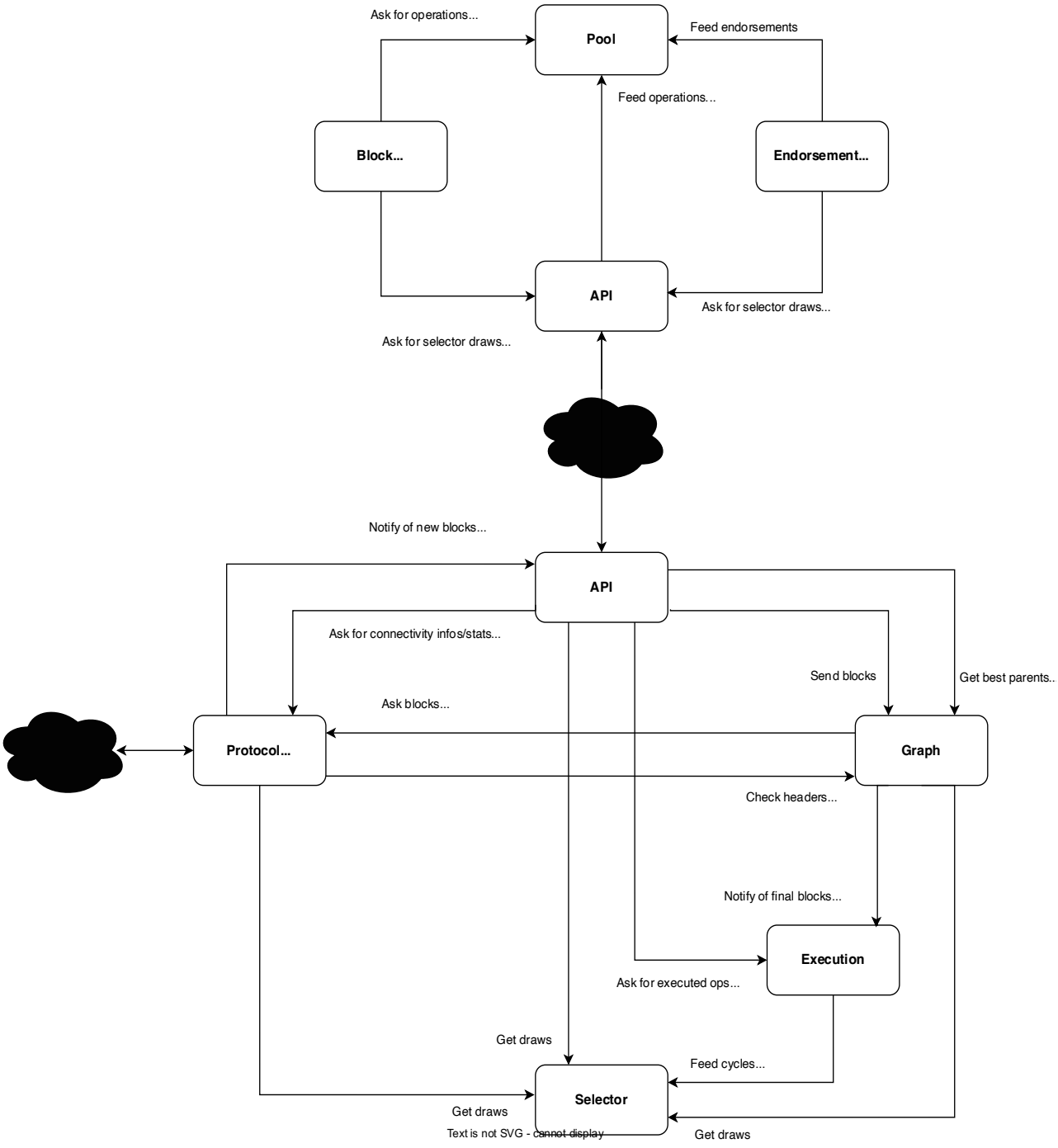
Questa sezione presuppone alcune conoscenze di base del protocollo Massa. Se non hai familiarità con i concetti di base di Massa, leggi prima la sezione Concetti di base.

La sezione descrive l'architettura globale di un nodo Massa, da zero.

Questo è il diagramma dell'architettura dei moduli software coinvolti nella costruzione, nell'approvazione e nella propagazione di blocchi.

La parte inferiore corrisponde a un singolo processo in esecuzione in un nodo ed è responsabile dell'esecuzione e della costruzione del consenso. Il pool e le factories, denominate "Factory", possono potenzialmente essere in esecuzione in un processo diverso o far parte del nodo.

Nel complesso, ciascuno dei moduli qui descritti viene eseguito all'interno di uno o più thread collegati al rispettivo processo eseguibile (NB: la separazione fabbrica/nodo non è ancora implementata, ma sarà presto).



Spiegheremo di seguito i diversi moduli presenti in questo diagramma e simuleremo la produzione di un'operazione per mostrare come naviga attraverso i diversi moduli per capire meglio come vengono prodotti e propagati i blocchi.

Modulo Bootstrap

Il modulo bootstrap è responsabile della sincronizzazione iniziale del nodo con il resto della rete. È responsabile del download dell'elenco dei peer, del grafico corrente dei blocchi, del libro mastro(ledger), del pool asincrono, dello stato del Proof-of-Stake e delle ultime operazioni eseguite.

Il bootstrap verrà eseguito da un server elencato nella configurazione del nodo. Bootstrap è il punto di ingresso della rete, quindi devi stare attento a quale nodo ti connetti per evitare di scaricare dati dannosi.

Modulo API

Il modulo API è la finestra pubblica del nodo per il resto del mondo. Consente interazioni con client esterni o factories tramite protocolli JSON RPC e gRPC.

L'API include interfacce per fare quanto segue:

- Pubblicare una nuova operazione da un client
- Interrogare la rete sui saldi o sullo stato del libro mastro
- Consentire la sincronizzazione tra i nodi del pool remoto/fabbrica e i nodi di consenso, inviando/chiedendo blocchi, migliori genitori, disegni, ecc.

Modulo protocollo/rete

Il modulo Protocollo/Rete implementa il protocollo che collega i nodi di consenso. Questo protocollo è supportato da un livello di trasporto binario e ottimizzato. Il modulo Protocollo/Rete trasmetterà tutte le operazioni/creazioni di blocchi e la loro propagazione, in modo che tutti gli altri nodi della rete possano sincronizzare il loro stato interno, seguendo un tipo di protocollo di sincronizzazione basato su gossip. Il tipo di messaggi che possono essere inoltrati tramite il modulo Protocol/Network includono:

- Propagazione di blocchi/operazioni/approvazione (entrando o uscendo dal nodo)
- Richieste di ban dei nodi
- Informazioni/statistiche sulla connettività.

Modulo selettore, proof_of_stake sybil di stake

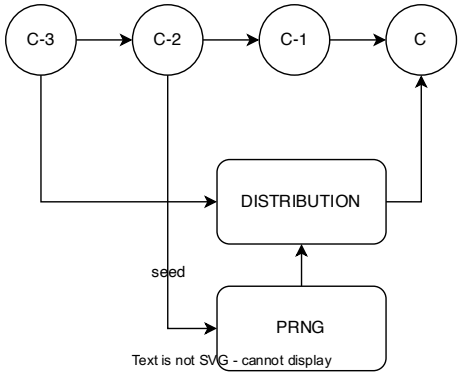
Ogni 0.5s, un nuovo slot diventa attivo per ricevere un nuovo blocco. Un meccanismo di selezione deterministа assicura che uno dei nodi della rete sia eletto per avere la responsabilità di costruire il blocco per quello slot.

Questo meccanismo deve avere diverse proprietà chiave:

- Dovrebbe essere resistente alla sybil, in modo che non sia possibile aumentare le proprie probabilità di essere eletti creando più cloni di se stessi (sybil) senza un costo uguale o superiore al costo di aumentare le proprie probabilità solo per se stessi;
- Dovrebbe essere deterministico, in modo che tutti i nodi della rete siano d'accordo sul risultato della selezione in un dato momento;
- Dovrebbe essere equo, in modo che ogni partecipante abbia una probabilità ben definita di essere selezionato in qualche modo proporzionale al costo della partecipazione e i disegni convergano verso questa distribuzione di probabilità nel tempo.

Il modo in cui la resistenza sybil si ottiene qui è attraverso il meccanismo di proof of stake. I nodi che vogliono partecipare alla lotteria della creazione di blocchi dovranno mettere in stake dei "rolls" che acquistano con le monete Massa. Se cercano di imbrogliare creando blocchi falsi o più blocchi sullo stesso slot, lo stake gli verrà tolto (slashing) e subirebbero una perdita. La "superficie" probabilistica di un partecipante è uguale alla so stake totale, il che rende inutile la creazione di account sybil perché la posta in gioco dovrebbe comunque essere divisa tra loro.

Il metodo utilizzato per disegnare un nodo eletto per un dato slot è semplicemente un'estrazione casuale da una distribuzione in cui gli indirizzi sono ponderati in base alla quantità di stake (=rolls) che detengono. Lo schema seguente illustra come vengono costruiti il seed e la distribuzione di probabilità, in base ai cicli passati (sono necessari due cicli per l'aggiornamento della distribuzione per garantire che la finalizzazione del saldo sia avvenuta e che la quantità di rolls sia accurata):



Il modulo selettore è incaricato di calcolare la formula e rispondere alle richieste relative a quale nodo viene scelto per un determinato slot nel presente o nel passato. Il modulo di esecuzione (vedi sotto) è incaricato di alimentare il modulo Selettore con gli aggiornamenti relativi ai saldi, necessari per calcolare le estrazioni.

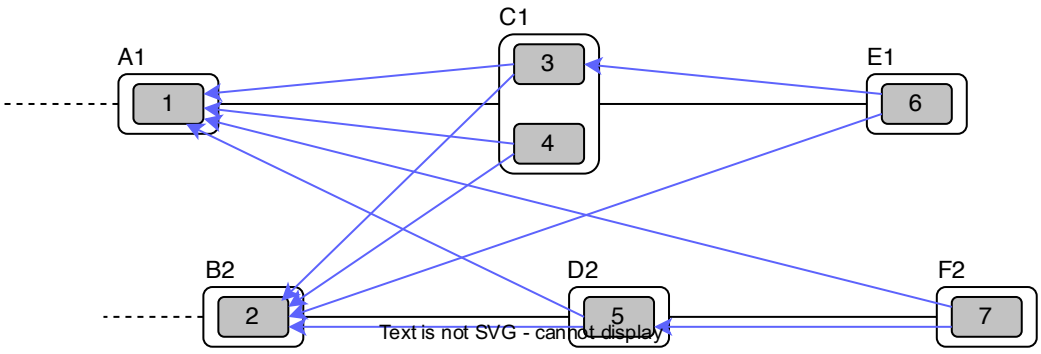
Modulo grafico/consenso

Il Modulo di Consenso è il cuore della macchina della Rete Massa. È incaricato di integrare i blocchi proposti nei rispettivi slot e di verificare l'integrità del risultato. Non abbiamo ancora parlato dei vari vincoli riguardanti la creazione di blocchi e in particolare di come i genitori devono essere selezionati. Nelle blockchain tradizionali, il genitore di un blocco è semplicemente il blocco valido precedente nella catena. Nel contesto della rete Massa e delle catene parallele nei 32 threads, identificare il genitore corretto in un dato thread richiede una strategia più sofisticata che coinvolga la nozione di “block cliques” o gruppi di blocchi.

Block cliques

In qualsiasi momento, l'insieme di tutti i blocchi che sono stati prodotti e propagati nella rete costituisce un grafico (più precisamente un grafico aciclico diretto o "DAG"), dove ogni blocco, ad eccezione dei blocchi di genesi, ha 32 genitori. Tutti i ragionamenti seguenti possono essere fatti in linea di principio su questo set sempre più vasto, ma in pratica introdurremo una nozione di blocchi "finalizzati" o "stallati", che possono essere rimossi dal set e che ci permetteranno di lavorare su un sottoinsieme più piccolo di blocchi recenti che non sono né finalizzati né bloccati, quindi blocchi "in sospeso". Questo insieme di blocchi in sospeso è tutto ciò che la rete deve sapere per costruire in modo incrementale un consenso, quindi i blocchi non in sospeso saranno semplicemente dimenticati (questa è una differenza sorprendente con la maggior parte delle altre blockchain che memorizzano in ogni nodo la storia di tutte le transazioni passate). Il vantaggio principale di questa potatura a blocchi è quello di consentire ad alcuni degli algoritmi seguenti, che sono generalmente NP-complete, di funzionare abbastanza velocemente su un sottografo più piccolo e di consentire un'implementazione pratica.

Ecco un esempio semplificato di un grafico di blocchi in sospeso su due threads, con i blocchi 3 e 4 in competizione per lo slot C1 (ad esempio a seguito di un attacco multistaking in cui il produttore di blocchi ha deciso di creare blocchi concorrenti per lo stesso slot). Qui la lettera di uno slot lo identifica, mentre il numero si riferisce al suo numero di thread:



In questa illustrazione abbiamo mostrato solo i link genitori rilevanti in blu, per rendere l'intero diagramma più leggibile, ma in realtà, ogni blocco ha 32 genitori, uno in ciascuno dei 32 threads.

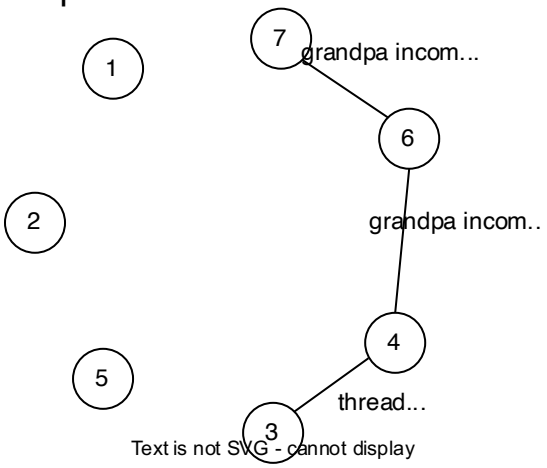
Una nozione importante che useremo tra un attimo è quella dell'incompatibilità tra i blocchi. Escludendo alcuni casi limite con blocchi di genesi, ci sono due fonti di incompatibilità definite per i blocchi:

1) **Incompatibilità del thread**: si verifica quando due blocchi in un dato thread hanno lo stesso padre in quel thread.

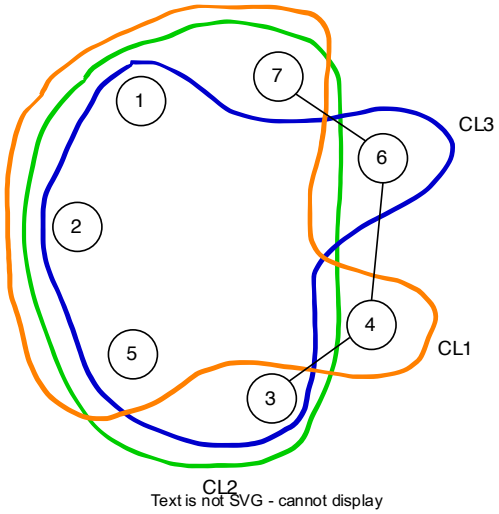
2) **Incompatibilità del nonno**: questo corrisponde a un caso con due blocchi B1 e B2 nei thread t1 e t2, e dove il blocco B1 in t1 ha un genitore in t2 che è un antenato del genitore di B2 in t2, e simmetricamente il genitore di B2 in t1 è un antenato del genitore di B1 in t1.

Troverai una definizione matematica più formale di queste nozioni di incompatibilità nel whitepaper

Da queste definizioni, è possibile costruire un altro grafico, chiamato grafico di incompatibilità, che collega due blocchi qualsiasi che hanno qualsiasi forma di incompatibilità insieme:



Come potete vedere, alcuni blocchi sono isolati e quindi compatibili con qualsiasi altro, mentre alcuni sono collegati, perché hanno una forma di incompatibilità. Questo ci porta alla nozione di un clique massimo che è un sottoinsieme del grafico di incompatibilità come nessuno dei membri del blocco è incompatibile tra loro (quindi, nessun collegamento interno con il clique), ed è impossibile aggiungere un blocco extra all'insieme senza introdurre incompatibilità. Nell'esempio precedente, ci sono tre gruppi massimi che possono essere costruiti, come illustrato di seguito:



Rappresentano i candidati per estendere l'insieme di blocchi già finalizzati in un insieme coerente di nuovi blocchi. Tutto ciò che dobbiamo aggiungere per essere in grado di costruire una regola di consenso ora è introdurre una metrica deterministica per classificare quei candidati in

modo che i nodi possano decidere in modo indipendente e coerente quale clique è il miglior candidato e continuare a costruire su di essa. In particolare, una volta identificata il miglior clique massimale, diventa banale definire l'elenco dei genitori per un nuovo blocco semplicemente scegliendo il blocco più vecchio da quel clique in ogni thread.

La metrica utilizzata in una blockchain tradizionale per classificare le catene concorrenti è abitualmente la lunghezza della catena, o più precisamente l'ammontare totale di lavoro investito nella catena (anche noto come "consenso Nakamoto"). Nel caso delle clique di blocchi, introdurremo una nozione di idoneità per ogni blocco, e l'idoneità della clique sarà semplicemente la somma di tutte le idoneità dei suoi blocchi. L'idoneità del blocco f(b) è semplicemente definita come 1 + e, con e che rappresenta il numero di approvazioni registrate nel blocco.
Prendendo il clique massimo con la massima idoneità (o qualche selezione deterministica basata su hash in caso di uguaglianza), il modulo Graph/Consensus può definire quella che viene chiamata blockclique al momento attuale.

Blocchi finalizzati, blocchi scaduti

L'insieme di blocchi in sospeso cresce ogni volta che viene prodotto e aggiunto un nuovo blocco all'insieme corrente. Come abbiamo già menzionato, esiste anche un meccanismo di potatura incaricato di ridurre la dimensione del grafo rimuovendo i blocchi considerati definitivi, nonché i blocchi che possono essere considerati obsoleti e che non saranno mai definitivi.
Se un blocco è contenuto solo all'interno di clique che hanno un valore di idoneità inferiore rispetto all'idoneità del blocco nel clique (il clique con il massimo valore di idoneità), meno una costante Δ^0_f allora questo blocco è considerato obsoleto. Inoltre, ogni nuovo blocco che include tra i suoi genitori un blocco obsoleto è anch'esso considerato obsoleto.

Un blocco è considerato definitivo se fa parte di tutti i clique massimali e incluso in almeno un clique in cui la somma totale dell'idoneità di tutti i suoi discendenti è maggiore di Δ^0_f.
Δ^0_f è definito come una costante F moltiplicata per 1+E (E è il numero massimo totale di approvazioni in un blocco, attualmente 16), e F misura efficacemente l'intervallo massimo in blocchi completamente approvati di una blockclique di successo, o il numero di blocchi completamente approvati da cui una clique alternativa può essere più breve della blockclique prima che i suoi blocchi possano essere scartati come obsoleti.

Funzione del modulo grafico/consenso

Il Modulo di Consenso (precedentemente noto come il Grafo) riceve nuove proposte di blocchi, le integra nell'insieme di blocchi in sospeso, aggiorna la blockclique con il metodo spiegato sopra e verifica la legittimità dell'attribuzione genitoriale dei nuovi blocchi. Informa anche altri moduli, come il Modulo di Esecuzione, quando i blocchi sono definitivi e le modifiche corrispondenti al registro implicati dalla loro lista di operazioni dovrebbero essere rese permanenti.
È anche in grado di rispondere alle domande sugli attuali migliori genitori per un nuovo blocco (basato sull'attuale blocco) o l'elenco delle cliques massimali correnti.

Modulo di esecuzione

Il modulo di esecuzione è incaricato di eseguire efficacemente le operazioni contenute nei blocchi all'interno dell'attuale blocco, fornito dal modulo Graph/Consensus. Le operazioni in genere modificheranno il ledger, modificando i saldi dei conti o modificando il datastore dei contratti intelligenti dopo l'esecuzione di un certo codice. Dal punto di vista dell'implementazione, le modifiche al ledger sono tuttavia memorizzate come differenza rispetto all'attuale ledger finalizzato, fino a quando i blocchi corrispondenti non vengono contrassegnati come finalizzati dal modulo Graph/Consensus. I creatori di blocchi dovranno in genere interrogare il modulo di esecuzione per controllare i saldi correnti in un determinato slot e verificare se alcune operazioni possono essere eseguite con fondi sufficienti o meno, prima di essere integrate in un nuovo blocco.

Come nota a margine, è anche possibile che i blocchi possano includere operazioni non valide, nel qual caso il modulo di esecuzione le ignorerà semplicemente.
Essendo il manutentore del ledger, il modulo di esecuzione viene anche interrogato sulle informazioni sull'indirizzo in generale, tramite l'API, per qualsiasi modulo che ne abbia bisogno.

Infine, il modulo di esecuzione informerà il modulo selettore quando vengono avviati nuovi cicli man mano che la finalizzazione dei blocchi progredisce.

Modulo Pool

Quando le nuove operazioni in sospeso raggiungono un nodo, non vengono immediatamente elaborate, ma vengono invece memorizzate in un pool di operazioni in sospeso, da utilizzare dal modulo factory. Allo stesso modo, le approvazioni proposte provenienti dalla Endorsement Factory sono bufferate all'interno del pool, per essere integrate in nuovi blocchi dal Block Factory Module.

L'origine delle operazioni in sospeso o delle approvazioni all'interno del pool può essere interna al processo di factory o potrebbe provenire da nodi remoti tramite il modulo API. Allo stesso modo, le approvazioni in sospeso prodotte localmente vengono trasmesse tramite un protocollo di gossip ad altri pool tramite il modulo API.

Si noti che le operazioni memorizzate nel pool vengono naturalmente scartate dopo un certo tempo, poiché le operazioni vengono con una data di scadenza nel campo expiration_period. Tuttavia, alcuni potenziali attacchi possono verificarsi cercando di inondare il pool con operazioni a commissioni elevate che non hanno alcuna possibilità di essere eseguite perché il conto corrispondente non ha i fondi richiesti. Discutere delle contromisure per questo va oltre lo scopo di questa introduzione.

Modulo factory di blocco/approvazione

Il modulo Block Factory è incaricato di creare nuovi blocchi quando l'indirizzo del nodo corrispondente è stato designato per essere il creatore del blocco per un determinato slot. Queste informazioni vengono fornite al modulo di fabbrica dal modulo selettore tramite il modulo API.

Il modulo Block Factory ha anche bisogno di informazioni sui migliori genitori (fatti degli ultimi blocchi in ogni thread nella blockclique) dal modulo Graph/Consensus. Questi genitori saranno inclusi nel blocco appena creato. Le informazioni sul saldo, al fine di valutare la validità delle operazioni in sospeso, sono ottenute dal modulo di esecuzione, che mantiene lo stato del libro mastro dal punto di vista dello slot in cui dovrebbe essere creato il nuovo blocco.

Il modulo Block Factory sceglie le operazioni in sospeso dal modulo Pool. Si noti che la Block Factory interrogherà regolarmente il modulo di esecuzione in modo alle operazioni finalizzate ed eseguite e alle operazioni di pulizia interna che sono state gestite.

Infine, la Block Factory interroga il modulo Pool e sceglierà le approvazioni in sospeso corrispondenti ai migliori genitori selezionati per il blocco.

Con queste informazioni, è in grado di forgiare un nuovo blocco che verrà poi propagato al modulo Graph/Consensus tramite il modulo API, così come ad altri nodi tramite gossip, per mantenere uno stato sincronizzato globale.

Il modulo Endorsement Factory funziona in modo simile, richiedendo al modulo Selector di scoprire quando è stato designato per essere un produttore di approvazione, quindi alimentando nuove approvazioni al modulo Pool e al modulo API per la sincronizzazione globale.