XCM v2概览与去中心化的Liquid Staking方案

平行链开发经验及工具分享

江成 2021年9月25日

Parallel Finance
Institutional-grade DeFi Parachain

江成

全栈，区块链工程师

ParallelFl 核心开发者

https://github.com/GopherJ

目录

ParallelFI 介绍

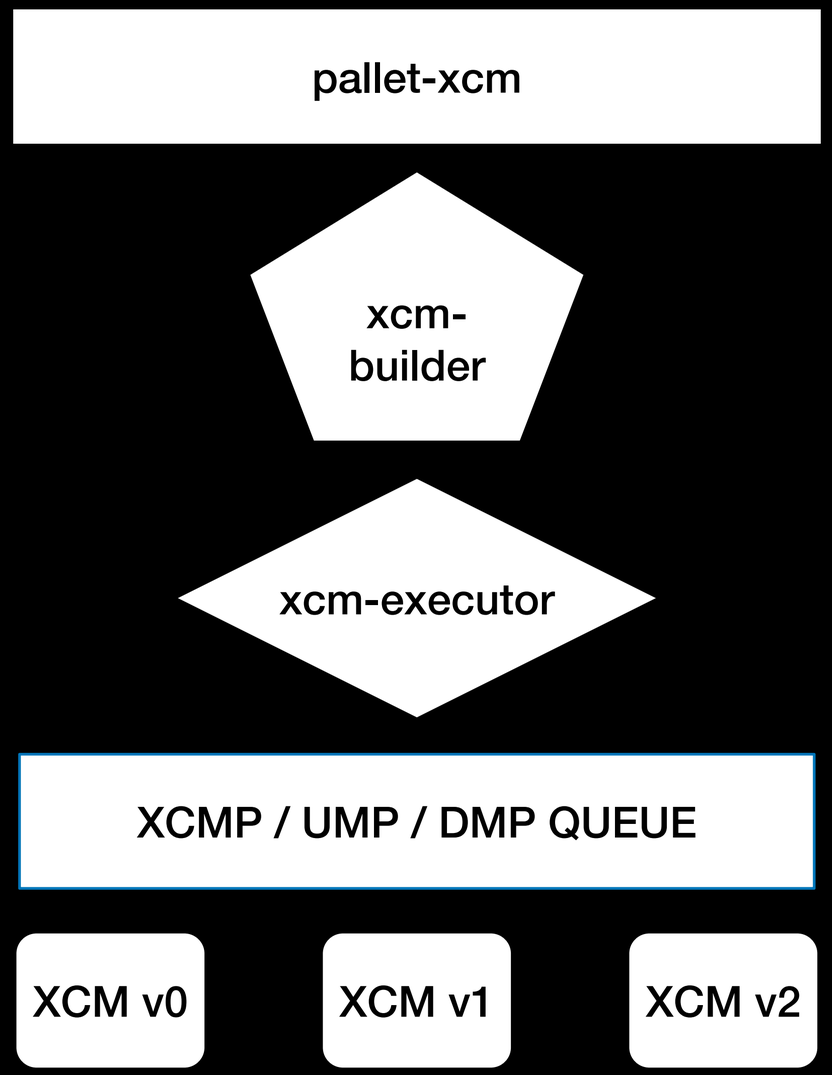
- 春季 Substrate Hackathon 时建立的团队
- 4月份成立Parallel Finance，总部位于美国
- 团队成员主要位于中国，美国，欧洲，印度，俄罗斯
- 2200万美元A轮融资，波卡生态中最大的一轮融资

Parallel Finance = Lending + Liquid Staking + AMM + Auction Loan + Cross chain wallet

Part 1

XCM v2概览

XCM v0 的回顾




```
WithdrawAsset {
    assets,
    effects: vec![DepositReserveAsset | InitiateReserveWithdraw {
        assets: vec![MultiAsset::All],
        dest | reserve,
        effects: vec![BuyExecution{..}, DepositAsset{..}],
    }],
}
```

```
WithdrawAsset {
    assets: vec![MultiAsset::ConcreteFungible {
        id: MultiLocation::Null,
        amount,
    }],
    effects: vec![
        BuyExecution {
            fees: MultiAsset::All,
            weight,
            debt,
            halt_on_error: false,
            xcm: vec![Transact {
                origin_type: OriginKind::SovereignAccount,
                require_weight_at_most,
                call,
            }],
        },
        DepositAsset {
            assets: vec![MultiAsset::All],
            dest: MultiLocation::X1(Junction::AccountId32 {
                network: NetworkId::Any,
                id: HOLDING_ACCOUNT,
            }),
        },
    ],
}
```

XCM v1, v2 的新特性

- 类型结构简化 (v1)
- 版本自动协商 (v1添加类型，v2添加实现)
- Extrinsics回调 (v2)
- VM (v2)

XCM v1, v2 的新特性

类型结构简化

- MultiAsset -> AssetId + Fungibility + WildMultiAsset (v1)
- MultiLocation X1…X8 -> MultiLocation { parents: u8, interior: Junctions } (v1)
- Xcm, Order -> Instruction (v2)
- BuyExecution weight + debt -> WeightLimit::Limited(weight) (v2)

例子：在 Parachain(2000) 上表示 Parachain(1001) 上的 30 个同质化 Token

V0

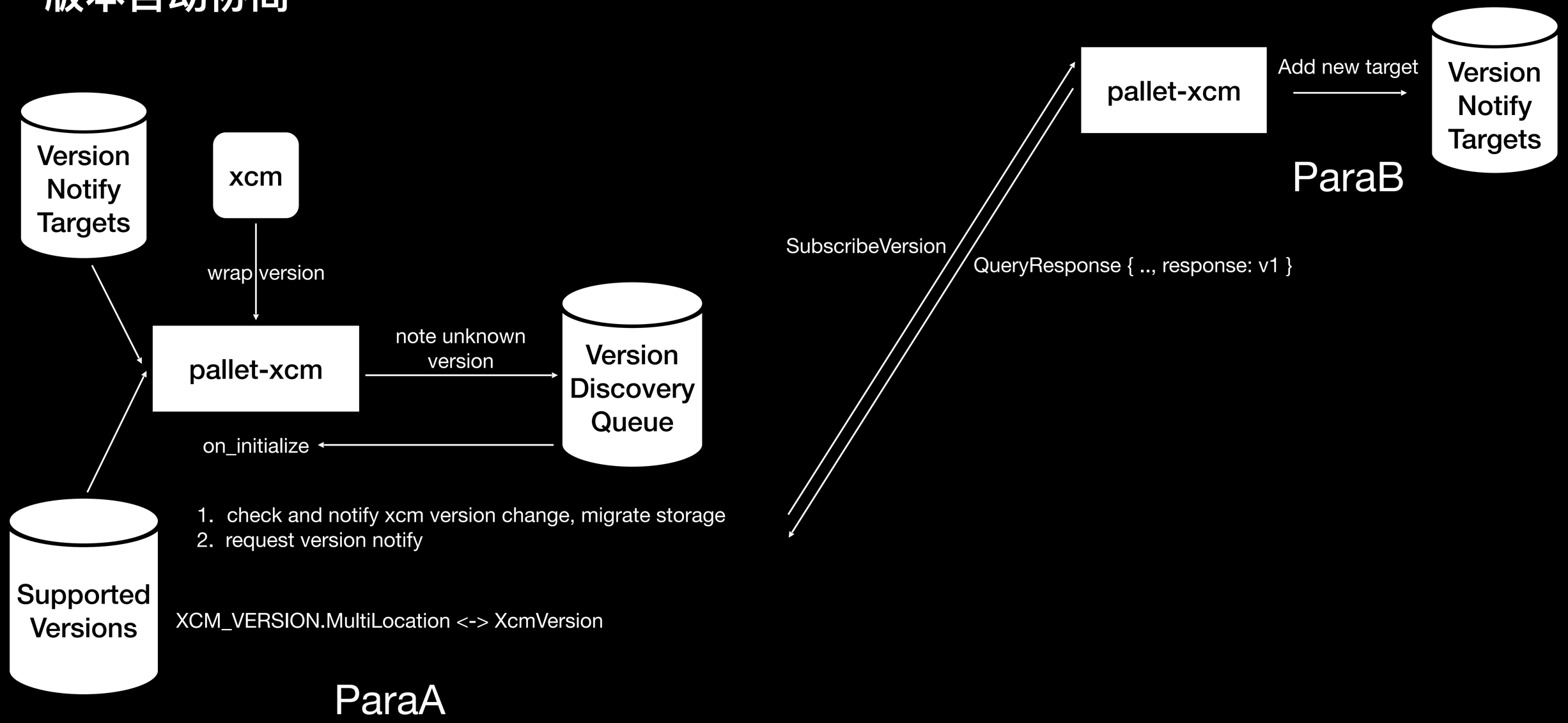
```
MultiAsset::ConcreteFungible {
    id: MultiLocation::X2(Junction::Parent, Junction::Parachain(1001)),
    amount: 30
}
```

V1, V2

```
MultiAsset {
    id: AssetId::Concrete(MultiLocation { parents: 1, interior: Junctions::X1(Junction::Parachain(1001)) }),
    fun: Fungibility::Fungible(30)
}
```

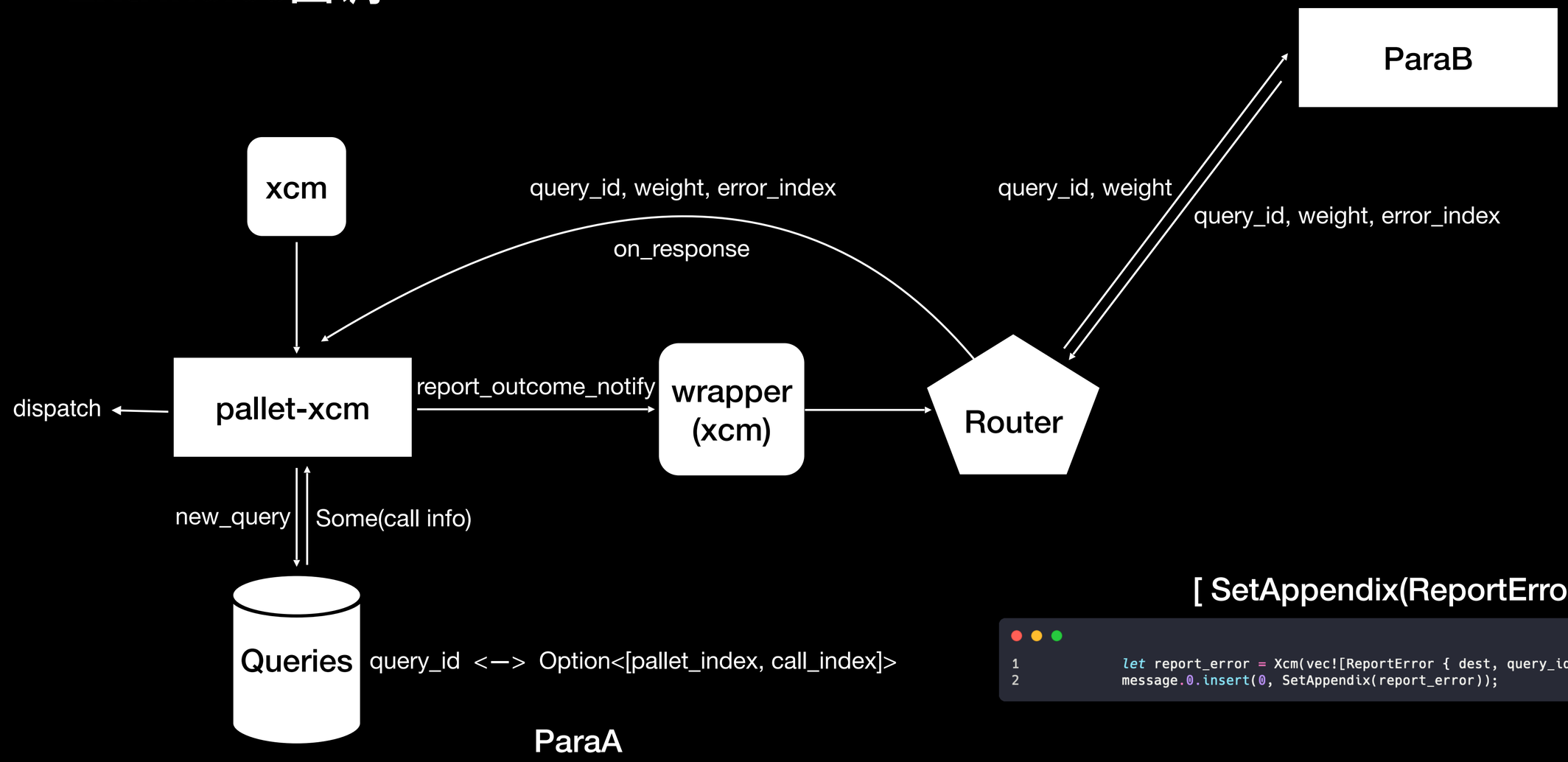
XCM v1, v2 的新特性

版本自动协商



XCM v1, v2 的新特性

Extrinsics回调



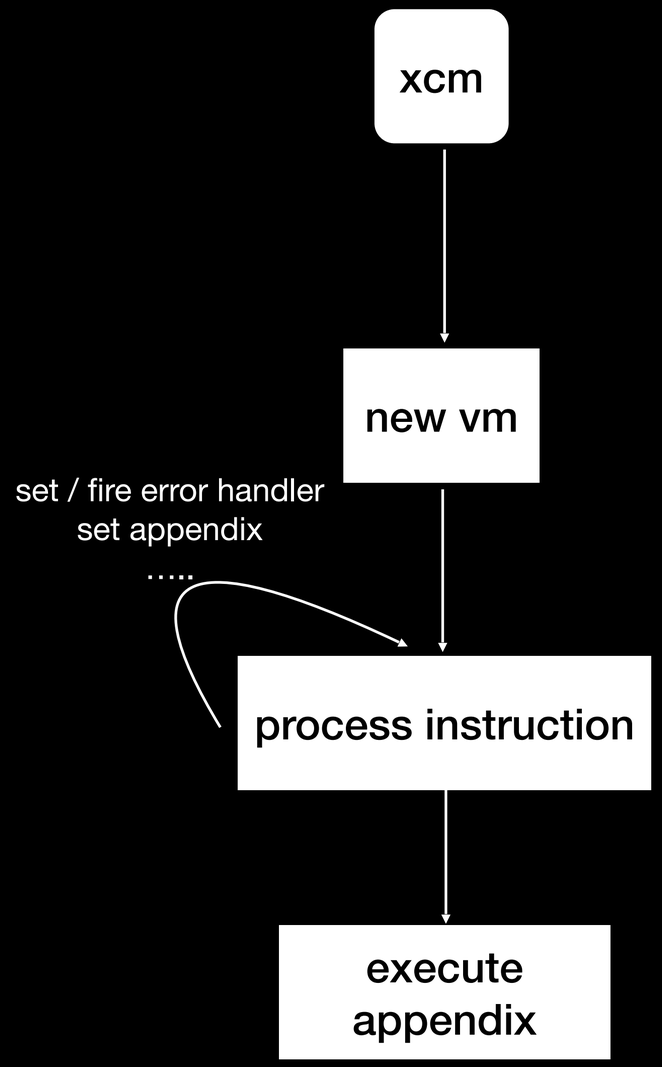
[SetAppendix(ReportError), …]

```
let report_error = Xcm(vec![ReportError { dest, query_id, max_response_weight }]);
message.0.insert(0, SetAppendix(report_error));
```

XCM v1, v2 的新特性

VM

```rust
/// The XCM executor.
pub struct XcmExecutor<Config: config::Config> {
    pub holding: Assets,
    pub origin: Option<MultiLocation>,
    pub original_origin: MultiLocation,
    pub trader: Config::Trader,
    /// The most recent error result and instruction index into the fragment in which it occurred,
    /// if any.
    pub error: Option<(u32, XcmError)>,
    /// The surplus weight, defined as the amount by which `max_weight` is
    /// an over-estimate of the actual weight consumed. We do it this way to avoid needing the
    /// execution engine to keep track of all instructions' weights (it only needs to care about
    /// the weight of dynamically determined instructions such as `Transact`).
    pub total_surplus: u64,
    pub total_refunded: u64,
    pub error_handler: Xcm<Config::Call>,
    pub error_handler_weight: u64,
    pub appendix: Xcm<Config::Call>,
    pub appendix_weight: u64,
    _config: PhantomData<Config>,
}
```



Part 2

去中心化的 Liquid Staking 方案

What is & why Liquid Staking

- 波卡使用 NPOS 保证网络安全
- 用户可以质押自己的 Token 提名验证者来获得奖励
- Token 质押期间不具有流动性，无法获得可流动的 “票据”
- 只有波卡网络知道这笔锁住的 Token

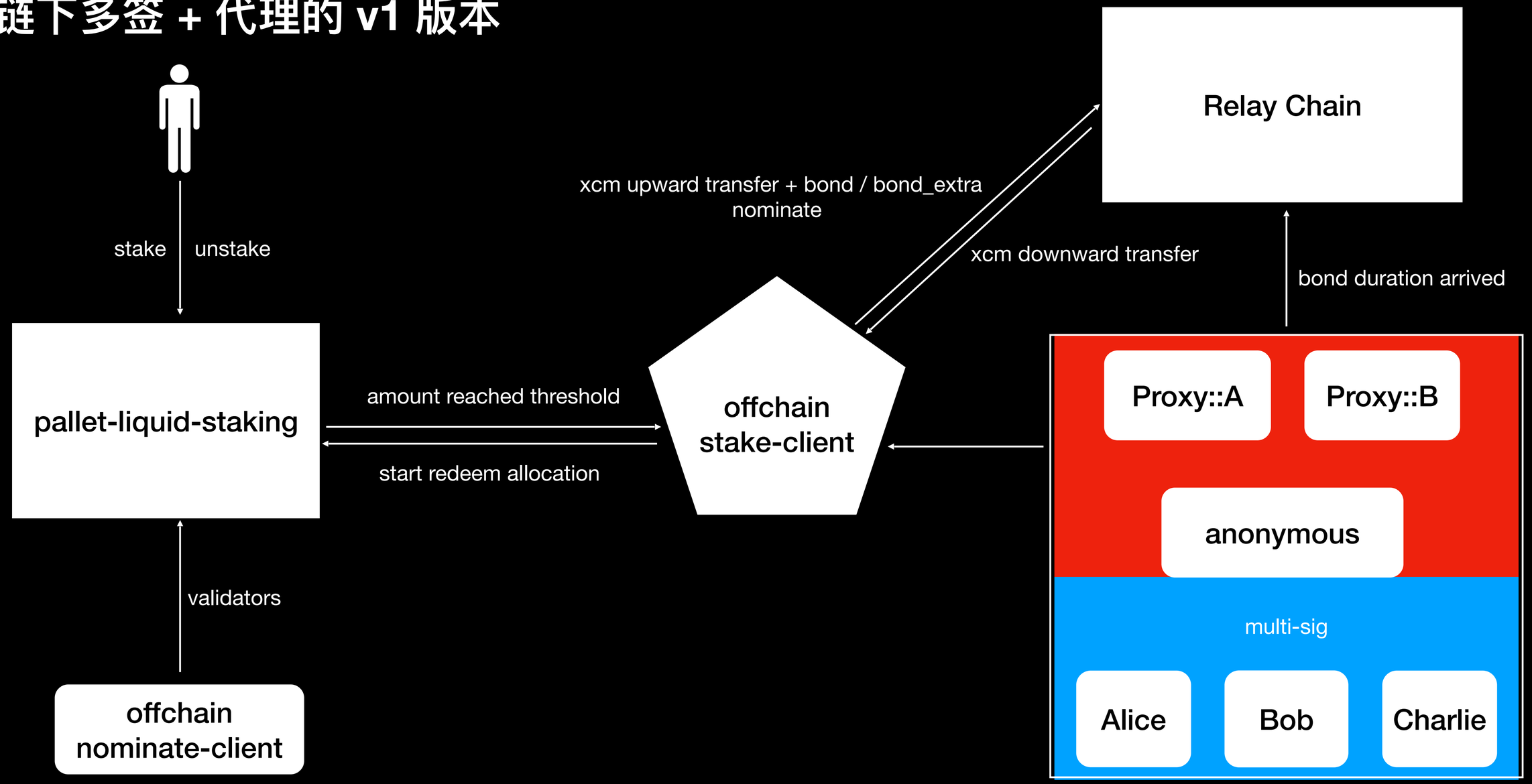
问题：为什么不提供票据释放流动性呢？让用户能自由的在各个平台交易“票据”

流动性释放的类似案例：

Bifrost SALP 协议释放 Crowdloan 期间锁住 Token 的流动性

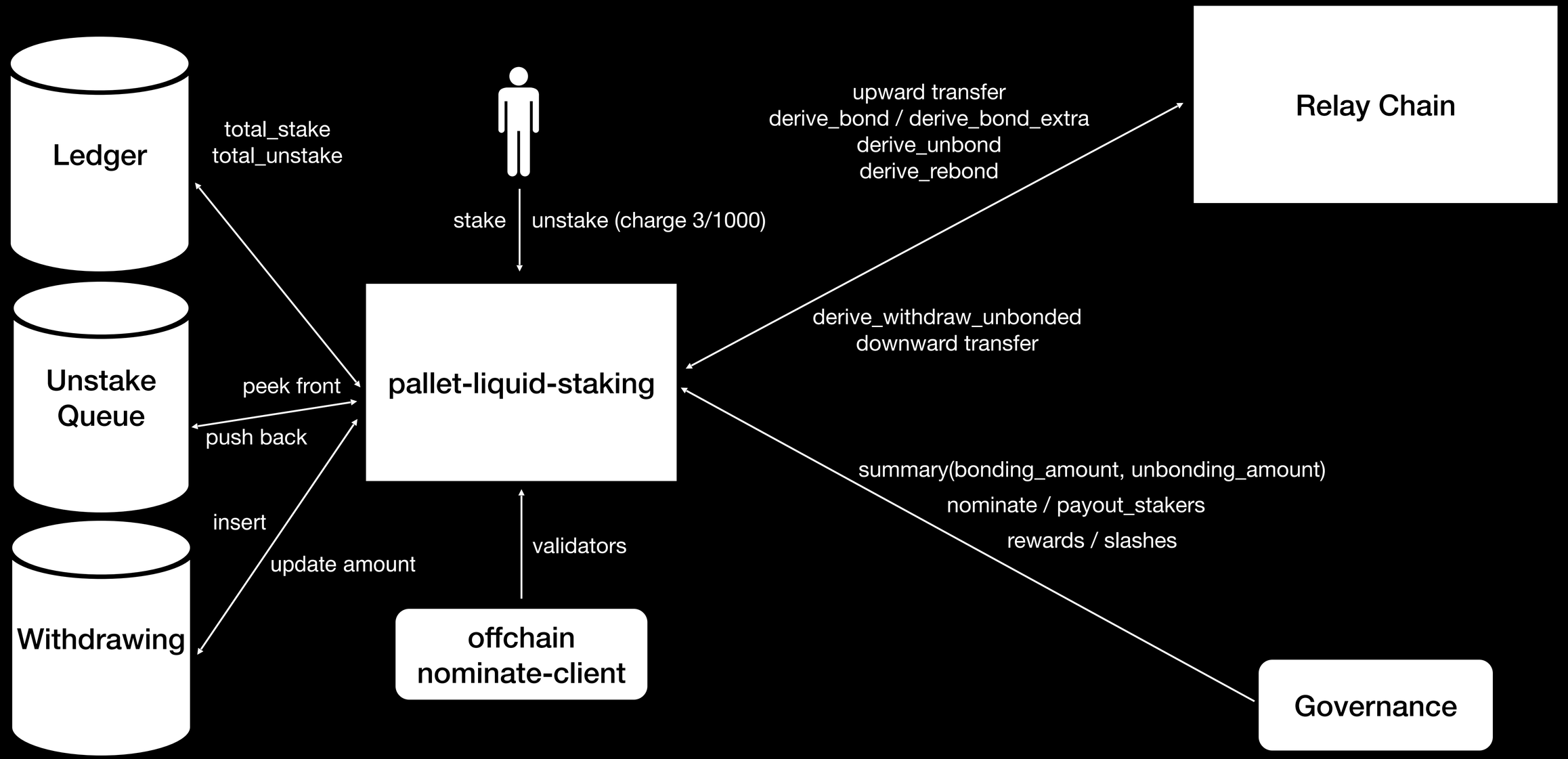
Liquid Staking 方案探索

链下多签 + 代理的 v1 版本



Liquid Staking 方案探索

基于 Transact 和衍生账户的 v2 版本



Liquid Staking 验证人选举

$$Score = R \cdot (crf \cdot (1 - CR) + nf \cdot \frac{1}{N} + epf \cdot \frac{EEP}{EEPA}) \cdot SR$$

R: Reputation, 0 or 1
CR: Commission Rate
N: Nomination of one validator
EEP: Average Era Points of one validator in the past 28 eras.
EEPA: Average Era Points of All validators in the past 28 eras.
crf: A constant shows how much influence of the Commission Rate of a validator. The default value is 100.
nf: A constant shows how much influence of the Nomination of a validator. The default value is 1000.
epf: A constant shows how much influence of the Era Points of a validator. The default value is 10.
SR: Slash Record, default 1, set to 0 if ever slashed in the past month

基于 Transact 和衍生账户的 Liquid Staking 实现

中继链方法定义

pallet_staking

```rust
/// Relaychain staking.bond_extra call arguments
#[derive(Clone, Encode, Decode, RuntimeDebug)]
pub struct StakingBondExtraCall<T: Config> {
    /// Rebond amount
    #[codec(compact)]
    pub value: BalanceOf<T>,
}

#[derive(Encode, Decode, RuntimeDebug)]
pub enum StakingCall<T: Config> {
    #[codec(index = 0)]
    Bond(StakingBondCall<T>),
    #[codec(index = 1)]
    BondExtra(StakingBondExtraCall<T>),
    #[codec(index = 2)]
    Unbond(StakingUnbondCall<T>),
    #[codec(index = 3)]
    WithdrawUnbonded(StakingWithdrawUnbondedCall),
    #[codec(index = 5)]
    Nominate(StakingNominateCall<T>),
    #[codec(index = 18)]
    PayoutStakers(StakingPayoutStakersCall<T>),
    #[codec(index = 19)]
    Rebond(StakingRebondCall<T>),
}
```

pallet_utility

```rust
/// Relaychain utility.batch_all call arguments
#[derive(Encode, Decode, RuntimeDebug)]
pub struct UtilityBatchAllCall<RelaychainCall> {
    /// calls
    pub calls: Vec<RelaychainCall>,
}

#[derive(Encode, Decode, RuntimeDebug)]
pub enum UtilityCall<RelaychainCall> {
    #[codec(index = 1)]
    AsDerivative(UtilityAsDerivativeCall<RelaychainCall>),
    #[codec(index = 2)]
    BatchAll(UtilityBatchAllCall<RelaychainCall>),
}
```

pallet_balances

```rust
/// Relaychain balances.transfer_keep_alive call arguments
#[derive(Clone, Encode, Decode, RuntimeDebug)]
pub struct BalancesTransferKeepAliveCall<T: Config> {
    /// dest account
    pub dest: <T::Lookup as StaticLookup>::Source,
    /// transfer amount
    #[codec(compact)]
    pub value: BalanceOf<T>,
}

/// Relaychain balances.transfer_all call arguments
#[derive(Clone, Encode, Decode, RuntimeDebug)]
pub struct BalancesTransferAllCall<T: Config> {
    /// dest account
    pub dest: <T::Lookup as StaticLookup>::Source,
    pub keep_alive: bool,
}

#[derive(Encode, Decode, RuntimeDebug)]
pub enum BalancesCall<T: Config> {
    #[codec(index = 3)]
    TransferKeepAlive(BalancesTransferKeepAliveCall<T>),
    #[codec(index = 4)]
    TransferAll(BalancesTransferAllCall<T>),
}
```

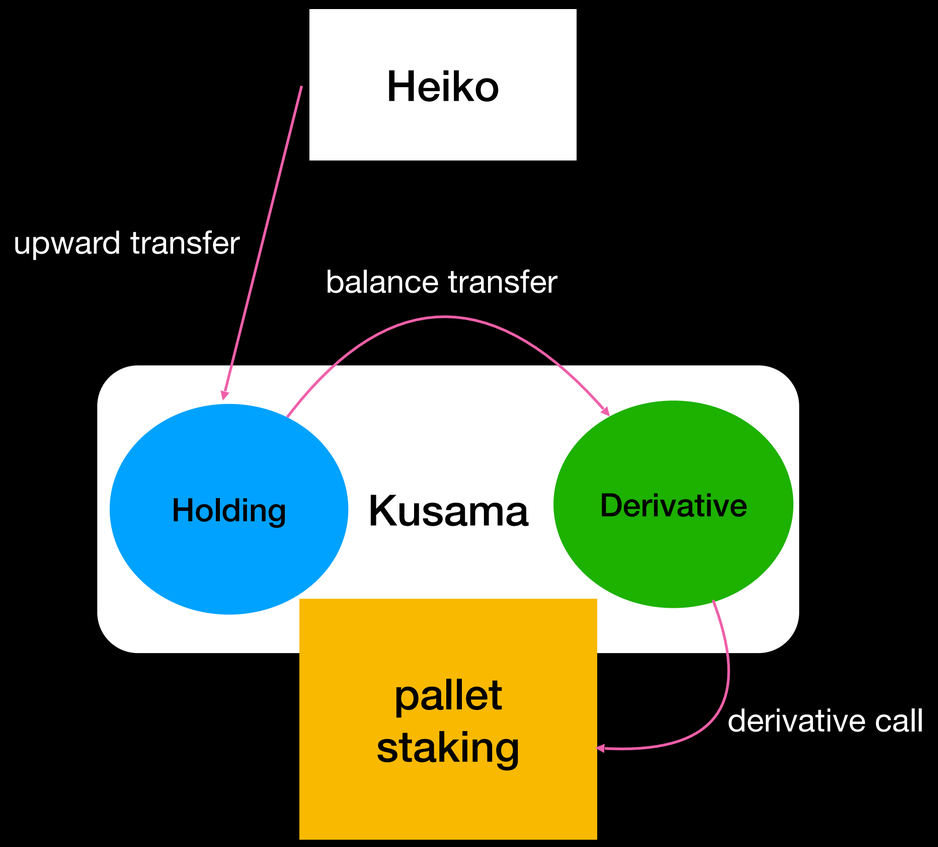
注意：参数的 #[codec(compact)] 需与中继链保持一致，否则会报 WeightNotComputable 错误

基于 Transact 和衍生账户的 Liquid Staking 实现

XCM消息格式

staking.bond

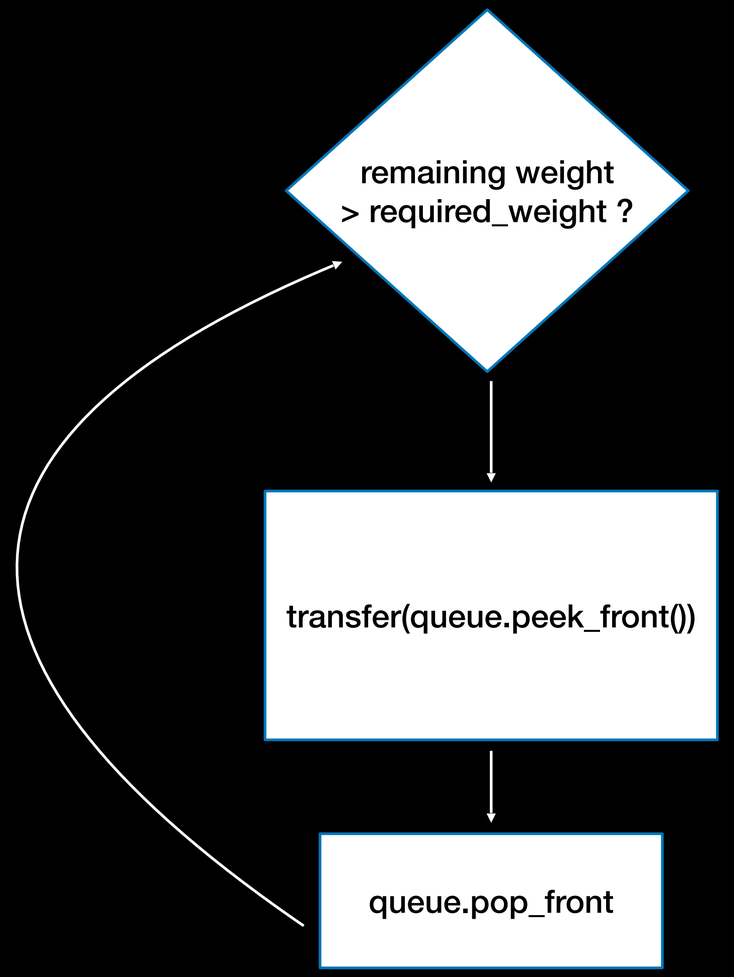
```rust
        let stash = Self::derivative_account_id();
        let controller = stash.clone();

        let call = RelaychainCall::Utility(Box::new(UtilityCall::BatchAll(UtilityBatchAllCall {
            calls: vec![
                RelaychainCall::Balances(BalancesCall::TransferKeepAlive(
                    BalancesTransferKeepAliveCall {
                        dest: T::Lookup::unlookup(stash),
                        value,
                    },
                )),
                RelaychainCall::Utility(Box::new(UtilityCall::AsDerivative(
                    UtilityAsDerivativeCall {
                        index: T::DerivativeIndex::get(),
                        call: RelaychainCall::Staking::<T>(StakingCall::Bond(StakingBondCall {
                            controller: T::Lookup::unlookup(controller.clone()),
                            value,
                            payee,
                        })),
                    },
                ))),
            ],
        })));

        let msg = WithdrawAsset {
            assets: vec![MultiAsset::ConcreteFungible {
                id: MultiLocation::Null,
                amount,
            }],
            effects: vec![
                BuyExecution {
                    fees: MultiAsset::All,
                    weight,
                    debt,
                    halt_on_error: false,
                    xcm: vec![Transact {
                        origin_type: OriginKind::SovereignAccount,
                        require_weight_at_most,
                        call,
                    }],
                },
                DepositAsset {
                    assets: vec![MultiAsset::All],
                    dest: MultiLocation::X1(Junction::AccountId32 {
                        network: NetworkId::Any,
                        id: HOLDING_ACCOUNT,
                    }),
                },
            ],
        };

        T::XcmSender::send_xcm(MultiLocation::X1(Junction::Parent), msg);
```



基于 Transact 和衍生账户的 Liquid Staking 实现

on_idle 尝试对 unstake 队列 peek_front 并支付 Staking Currency

```rust
fn on_idle(_n: BlockNumberFor<T>, mut remaining_weight: Weight) -> Weight {
    let base_weight = T::WeightInfo::pop_queue();
    if Self::staking_currency().is_none() {
        return remaining_weight;
    }
    loop {
        // Check weight is enough
        if remaining_weight < base_weight {
            break;
        }

        if Self::unstake_queue().is_empty() {
            break;
        }

        // Get the front of the queue.
        let (who, amount) = &Self::unstake_queue()[0];

        if T::Assets::transfer(
            Self::staking_currency(),
            &Self::account_id(),
            who,
            *amount,
            true,
        )
        .is_err()
        {
            // break if we cannot afford this
            break;
        }

        // substract weight of this action if succeed.
        remaining_weight -= base_weight;

        // remove unstake request from queue
        Self::pop_unstake_task()
    }
    remaining_weight
}
```



Part 3

平行链开发经验及工具分享

平行链开发经验及工具分享

parachain-launch

config.yml

```yaml
relaychain:
  image: parallelfinance/polkadot:v0.9.9-1
  chain: westend-local
  runtimeGenesisConfig:
    configuration:
      config:
        validation_upgrade_frequency: 1
        validation_upgrade_delay: 1
  flags:
    - --rpc-methods=unsafe
    - --no-beefy
  nodes:
    - name: alice
    - name: bob
    - name: charlie

parachains:
  - image: parallelfinance/parallel:latest
    chain:
      base: vanilla-dev
      collators:
        - alice
        - bob
        - charlie
      sudo: dave
    id: 2085
    parachain: true
    flags:
      - --rpc-methods=unsafe
      - --force-authoring
    relaychainFlags:
      - --no-beefy
    nodes:
      - flags:
          - --alice
      - flags:
          - --bob
      - flags:
          - --charlie
```

```bash
yarn global add @open-web3/parachain-launch
parachain-launch generate config.yml
cd output
docker-compose up -d --build
```

Makefile

```makefile
.PHONY: shutdown
shutdown:
    docker-compose -f output/docker-compose.yml down --remove-orphans > /dev/null 2>&1 || true
    rm -fr output || true
    docker volume prune -f

.PHONY: launch
launch: shutdown
    parachain-launch generate config.yml && docker-compose -f output/docker-compose.yml up -d --build
```

平行链开发经验及工具分享

rust-analyzer 针对平行链开发的配置优化

.vscode/settings.json | ~/.config/nvim/coc-settings.json

```
1 {
2   "rust-analyzer.experimental.procAttrMacros": false,
3   "rust-analyzer.cargo.runBuildScripts": false,
4   "rust-analyzer.procMacro.enable": false,
5   "rust-analyzer.checkOnSave.enable": false
6 }
```

Makefile

```
1 .PHONY: check
2 check:
3     SKIP_WASM_BUILD= cargo check --all-targets --features runtime-benchmarks --features try-runtime
```

平行链开发经验及工具分享

srtool 确定性编译

scripts/srtool-build.sh

```bash
#!/usr/bin/env bash

DIR=$(cd -P -- "$(dirname -- "$0")" && pwd -P)

cd $DIR/../

set -xe

RUSTC_VERSION=1.53.0-0.9.16;
PACKAGE=${PACKAGE:-heiko-runtime};
BUILD_OPTS=$BUILD_OPTS;

docker run --rm -it \
  -e PACKAGE=$PACKAGE \
  -e BUILD_OPTS="$BUILD_OPTS" \
  -v $PWD:/build \
  -v $TMPDIR/cargo:/cargo-home \
  --network=host \
  paritytech/srtool:$RUSTC_VERSION
```

Makefile

```makefile
.PHONY: wasm
wasm:
	./scripts/srtool-build.sh
```

平行链开发经验及工具分享

vim 下基于 codelldb, vimspector, coc-rust-analyzer 调试

~/.vimspector.json

```
1 {
2   "configurations": {
3     "rust - launch": {
4       "adapter": "CodeLLDB",
5       "configuration": {
6         "type": "lldb",
7         "request": "launch",
8         "program": "${Executable}",
9         "args": ["*${Args}"],
10        "sourceLanguages": ["rust"]
11      },
12      "breakpoints": {
13        "exception": {
14          "cpp_throw": "Y",
15          "cpp_catch": "N"
16        }
17      }
18    }
19  }
20 }
```

~/.config/nvim/init.vim

```
1 Plug 'puremourning/vimspector'
2 let g:vimspector_enable_mappings = 'VISUAL_STUDIO'
3 let g:vimspector_install_gadgets = [
4       \ 'CodeLLDB'
5       \ ]
6 nnoremap <F8> :call vimspector#Reset()<CR>
7 let g:vimspector_sign_priority = {
8             \     'vimspectorBP':         15,
9             \ }
10 let g:vimspector_bottombar_height = 5
11 nmap <leader>di <Plug>VimspectorBalloonEval
12 xmap <leader>di <Plug>VimspectorBalloonEval
```

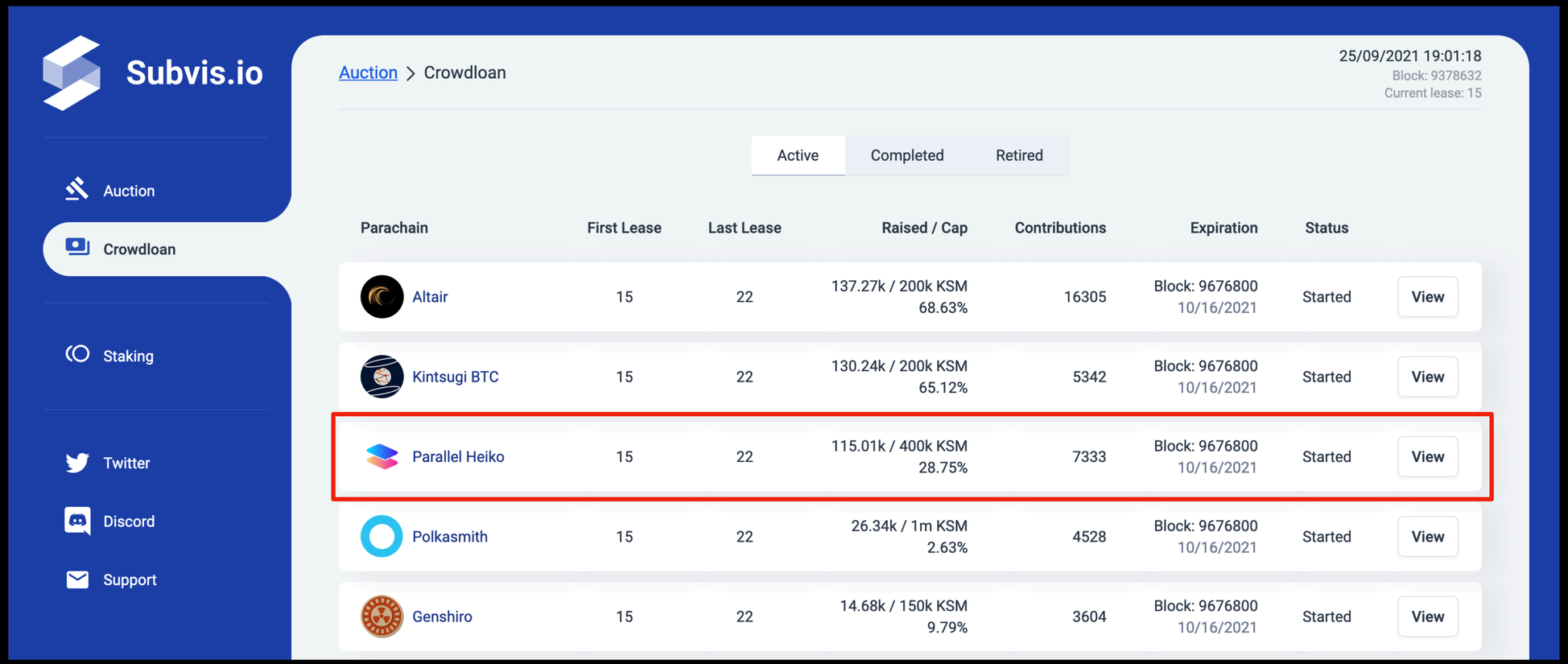
~/.config/nvim/coc-settings.json

```
1 {
2   "rust-analyzer.debug.vimspector.configuration.name": "rust - launch"
3 }
```

引用

- **[XCM part II: Versioning and compatibility]** https://medium.com/polkadot-network/xcm-part-ii-versioning-and-compatibility-b313fc257b83
- **[XCM: The Cross-Chain Message Format]** https://medium.com/polkadot-network/xcm-the-cross-consensus-message-format-3b77b1373392
- **[XCM v1]** https://github.com/paritytech/polkadot/pull/2815
- **[XCM v1 version notificatio stub]** https://github.com/paritytech/polkadot/pull/3766
- **[Automatic version negociation]** https://github.com/paritytech/polkadot/pull/3736
- **[XCM v2: Scripting, Query Responses, Exception handling and error reporting]** https://github.com/paritytech/polkadot/pull/3629

Kusama 插槽众贷



https://app.parallel.fi

Thank You!