

# Reward Sharing

Nhan Cao

--

[nhancv92@gmail.com](mailto:nhancv92@gmail.com)

**Keywords**—Dividend, Reward distribution, Reflection, Smart contract

## I. CONTEXT

| <b>BUY FROM DEX</b> |                                                                        |                                                                                                                      |               |                        |                                                      |                                 |
|---------------------|------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|---------------|------------------------|------------------------------------------------------|---------------------------------|
| 1                   | BOB                                                                    | Buy 103 > Tax 3 > Pool = 3, balance = 100, total staked = 100                                                        |               |                        |                                                      |                                 |
| 2                   | ADAM                                                                   | Buy 206 > Tax 6 > Pool = 9, balance = 200, total staked = 300                                                        |               |                        |                                                      |                                 |
| 3                   | CAROL                                                                  | Buy 309 > Tax 9 > Pool = 18, balance = 300, total staked = 600                                                       |               |                        |                                                      |                                 |
| 4                   | ERIC                                                                   | Buy 103 > Tax 3 > Pool = 21, balance = 100, total staked = 700                                                       |               |                        |                                                      |                                 |
| <b>FORMULA</b>      |                                                                        | Total buy: $103 + 206 + 309 + 103 = 700 + 21 = 721$                                                                  |               |                        |                                                      |                                 |
| 1                   | BOB                                                                    | $100 * 3/100$                                                                                                        | $100 * 6/300$ | $100 * 9/600$          | $100 * 3/700$                                        | $3.0 + 2.0 + 1.5 + 0.43 = 6.93$ |
| 2                   | ADAM                                                                   | 0                                                                                                                    | $200 * 6/300$ | $200 * 9/600$          | $200 * 3/700$                                        | $0.0 + 4.0 + 3.0 + 0.86 = 7.86$ |
| 3                   | CAROL                                                                  | 0                                                                                                                    | 0             | $300 * 9/600$          | $300 * 3/700$                                        | $0.0 + 0.0 + 4.5 + 1.28 = 5.78$ |
| 4                   | ERIC                                                                   | 0                                                                                                                    | 0             | 0                      | $100 * 3/700$                                        | $0.0 + 0.0 + 0.0 + 0.43 = 0.43$ |
|                     |                                                                        |                                                                                                                      |               |                        | Total = $6.93 + 7.86 + 5.78 + 0.43 = 21$ = pool size |                                 |
| <b>TRANSFER</b>     |                                                                        |                                                                                                                      |               |                        |                                                      |                                 |
| 1                   | BOB                                                                    | Buy 103 > Tax 3 > Pool = 3, balance = 100, total staked = 100                                                        |               |                        |                                                      |                                 |
| 2                   | ADAM                                                                   | Buy 206 > Tax 6 > Pool = 9, balance = 200, total staked = 300                                                        |               |                        |                                                      |                                 |
| 3                   | BOB send 50 coins to ADAM => CONVERT TO BUY CASE: ADAM BUY 50 from BOB |                                                                                                                      |               |                        |                                                      |                                 |
| 3.1                 | BOB                                                                    | Balance = $100 - 50 = 50$                                                                                            |               |                        |                                                      |                                 |
| 3.2                 | ADAM                                                                   | Same BUY 50 from DEX > Tax = $1.46 > \text{Pool} = 10.46$ , balance = $200 + 48.54 = 248.54$ , total staked = 298.54 |               |                        |                                                      |                                 |
| <b>FORMULA</b>      |                                                                        |                                                                                                                      |               |                        |                                                      |                                 |
| 1                   | BOB                                                                    | $100 * 3/100$                                                                                                        | $100 * 6/300$ | $50 * 1.46/298.54$     | $3.0 + 2.0 + 0.24 = 5.24$                            |                                 |
| 2                   | ADAM                                                                   | 0                                                                                                                    | $200 * 6/300$ | $248.54 * 1.46/298.54$ | $0.0 + 4.0 + 1.22 = 5.22$                            |                                 |
|                     |                                                                        |                                                                                                                      |               |                        | Total = $5.24 + 5.22 = 10.46$ = pool size            |                                 |
| <b>SELL TO DEX</b>  |                                                                        |                                                                                                                      |               |                        |                                                      |                                 |
| 1                   | BOB                                                                    | Buy 103 > Tax 3 > Pool = 3, balance = 100, total staked = 100                                                        |               |                        |                                                      |                                 |
| 2                   | ADAM                                                                   | Buy 206 > Tax 6 > Pool = 9, balance = 200, total staked = 300                                                        |               |                        |                                                      |                                 |
| 3                   | BOB SELL 50 coins => CONVERT TO TRANSFER CASE: BOB TRANSFER to LP      |                                                                                                                      |               |                        |                                                      |                                 |
| 3.1                 | BOB                                                                    | Sell 50 > Tax 1.5 > Pool 10.5, balance = 50, total staked = 250                                                      |               |                        |                                                      |                                 |
| <b>FORMULA</b>      |                                                                        |                                                                                                                      |               |                        |                                                      |                                 |
| 1                   | BOB                                                                    | $100 * 3/100$                                                                                                        | $100 * 6/300$ | $50 * 1.5/250$         | $3.0 + 2.0 + 0.3 = 5.3$                              |                                 |
| 2                   | ADAM                                                                   | 0                                                                                                                    | $200 * 6/300$ | $200 * 1.5/250$        | $0.0 + 4.0 + 1.2 = 5.2$                              |                                 |
|                     |                                                                        |                                                                                                                      |               |                        | Total = $5.3 + 5.2 = 10.5$ = pool size               |                                 |

## II. ALGORITHM

|                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Global                                      | <pre>// Global Variables r: map (uint -&gt; uint) // Portion reward mapping l: uint = 0 // Map length. urc: map (address -&gt; uint) // User's reward accumulation up: map (address -&gt; uint) // User's portion reward offset. ud: map (address -&gt; uint) // User's debt t: uint // total staked pool: uint // total pool reward distributed: uint // distributed reward fee: uint // fee percent</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Transfer (from_address, to_address, amount) | <pre># ----- # Token transfer function: transfer(from_address, to_address, amount) // Update portion anytime coin changes: buy, sell, transfer tp: address // tax payer  tax = fee * amount; after_tax = amount - tax; pool = pool + tax;  if (is_buy) {     t = t + after_tax;     tp = to_address;  } else if (is_transfer) {     t = t - tax;     tp = to_address;      urc[from_address] = urc[from_address] + get_portion_reward(from_address);     up[from_address] = l + 1;  } else {     t = t - amount;     tp = from_address;  }  // Update user offset and portion first urc[tp] = urc[tp] + get_portion_reward(tp); up[tp] = l + 1;  // Update reward portion r[l+1] = r[l] + tax / t; l = l + 1;  // Call native transfer super.transfer(from_address, address(this), tax); super.transfer(from_address, to_address, after_tax);</pre> |
| get_portion_reward (user_address)           | <pre># ----- # Get user's portion reward amount function: get_portion_reward(user_address)</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|                                           |                                                                                                                                                                                                                                                                                                                                     |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                           | // Get pending reward of user address<br>portion = up[user_address];<br>if (portion == 0) portion = 1;<br>return (r[l] - r[portion - 1]) * balanceOf(user_address);                                                                                                                                                                 |
| get_max_reward<br>(user_address)          | # -----<br># Get user's max reward amount function:<br>get_max_reward(user_address)<br>// Get max reward of user address<br>return get_portion_reward(user_address) + urc[user_address];                                                                                                                                            |
| get_pending_reward<br>(user_address)      | # -----<br># Get user's pending reward amount function:<br>get_pending_reward(user_address)<br>// Get pending reward of user address<br>total = get_max_reward(user_address);<br>if (total > ud[user_address]) return total - ud[user_address];<br>return 0;                                                                        |
| withdraw_pending_reward<br>(user_address) | # -----<br># Withdraw user's pending reward amount function:<br>withdraw_pending_reward(user_address)<br>// Withdraw reward<br>reward = get_pending_reward(user_address);<br>if (reward > 0) {<br>distributed = distributed + reward;<br>ud[user_address] = ud[user_address] + reward;<br><br>... do something with reward ...<br>} |

### III. JAVASCRIPT

```
export class AppService {  
    // Native transfer  
    balance: { [index: string]: number } = {};  
  
    _transfer(from: string, to: string, amount: number): void {  
        this.balance[from] -= amount;  
        this.balance[to] += amount;  
    }  
  
    r: { [index: number]: number } = {};  
    l: number = 0;  
    urc: { [index: string]: number } = {};  
    up: { [index: string]: number } = {};  
    ud: { [index: string]: number } = {};  
    t: number = 0;  
    pool: number = 0;  
    distributed: number = 0;  
    fee: number = 3 / 103; // 2.9%  
  
    // reset state  
    reset(): void {  
        this.r = {0: 0};  
        this.l = 0;  
    }  
}
```

```

this.urc = {};
this.up = {};
this.ud = {};
this.t = 0;
this.pool = 0;
this.balance = {'bob': 0, 'adam': 0, 'carol': 0, 'eric': 0};
}

// Update reward in transfer action
token_transfer(type: string, from_address: string, to_address: string, amount: number):
void {
  console.log(`type: ${type}, from: ${from_address}, to: ${to_address}`);
  let tp: string;

  const tax: number = this.fee * amount;
  const after_tax: number = amount - tax;
  this.pool = this.pool + tax;

  if (type === 'buy') {
    this.t = this.t + after_tax;
    tp = to_address;
  } else if (type === 'transfer') {
    this.t = this.t - tax;
    tp = to_address;

    this.urc[from_address] = (this.urc[from_address] ?? 0) +
this.get_portion_reward(from_address);
    this.up[from_address] = this.l + 1;
  } else {
    this.t = this.t - amount;
    tp = from_address;
  }
}

// Update user offset and portion first
this.urc[tp] = (this.urc[tp] ?? 0) + this.get_portion_reward(tp);
this.up[tp] = this.l + 1;

// Update reward portion
this.r[this.l + 1] = this.r[this.l] + tax>this.t;
this.l = this.l + 1;
}

// Call super native transfer
this._transfer(from_address, 'pool', tax);
this._transfer(from_address, to_address, after_tax);
}

// Get user portion reward function
get_portion_reward(user_address: string): number {
  let portion = (this.up[user_address] ?? 0);
  if(portion == 0) {
    portion = 1;
  }
  return (this.r[this.l] - this.r[portion - 1]) * this.balance[user_address];
}

// Get user max reward function
get_max_reward(user_address: string): number {
  return this.get_portion_reward(user_address) + (this.urc[user_address] ?? 0);
}

// Get user's pending reward amount function
get_pending_reward(user_address: string): number {
}

```

```

    const total = this.get_max_reward(user_address);
    if(total > (this.ud[user_address] ?? 0)) {
      return total - (this.ud[user_address] ?? 0);
    }
    return 0;
}

// Withdraw user's pending reward amount function
withdraw_pending_reward(user_address: string): void {
  const reward = this.get_pending_reward(user_address);
  if(reward > 0) {
    this.distributed += reward;
    this.ud[user_address] = (this.ud[user_address] ?? 0) + reward;
    console.log(`=> ${user_address} withdraw: ${reward}`);
  } else {
    console.log(`=> ${user_address} withdraw: empty`);
  }
}

```

#### IV. OUTPUT

```

const app = await NestFactory.createApplicationContext(AppModule);
const appService = app.get(AppService);

const printRewards = (isFull: boolean = false): void => {
  console.log(`Bob reward: ${appService.get_pending_reward('bob')}, balance: ${appService.balance['bob']} `);
  console.log(`Adam reward: ${appService.get_pending_reward('adam')}, balance: ${appService.balance['adam']} `);
  if(isFull) {
    console.log(`Carol reward: ${appService.get_pending_reward('carol')}, balance: ${appService.balance['carol']} `);
    console.log(`Eric reward: ${appService.get_pending_reward('eric')}, balance: ${appService.balance['eric']} `);
  }
}

const printPoolSizeAndStaked = (isDistributed: boolean = false): void => {
  console.log('Pool size:', appService.pool);
  console.log('Total staked:', appService.t);
  if(isDistributed) {
    console.log('Distributed size:', appService.distributed);
  }
}

// -----
appService.reset();
// Buy
console.log('---BUY---');
appService.token_transfer('buy', 'dex', 'bob', 103);
appService.token_transfer('buy', 'dex', 'adam', 206);
appService.token_transfer('buy', 'dex', 'carol', 309);
appService.token_transfer('buy', 'dex', 'eric', 103);
// Log
printPoolSizeAndStaked();
printRewards(true);

// -----
appService.reset();
// Transfer
console.log('---TRANSFER---');

```

```

appService.token_transfer('buy', 'dex', 'bob', 103);
appService.token_transfer('buy', 'dex', 'adam', 206);
appService.token_transfer('transfer', 'bob', 'adam', 50);
// Log
printPoolSizeAndStaked();
printRewards();

// -----
appService.reset();
// Sell
console.log('---SELL---');
appService.token_transfer('buy', 'dex', 'bob', 103);
appService.token_transfer('buy', 'dex', 'adam', 206);
appService.token_transfer('sell', 'bob', 'dex', 50);
// Log
printPoolSizeAndStaked();
printRewards();

// -----
appService.reset();
// Withdraw
console.log('---WITHDRAW---');
appService.token_transfer('buy', 'dex', 'bob', 103);
printRewards();
appService.withdraw_pending_reward('bob'); // = 3

appService.token_transfer('buy', 'dex', 'adam', 206);
printRewards();
appService.withdraw_pending_reward('bob'); // = 3 + 2 - 3 = 2
appService.withdraw_pending_reward('adam'); // = 4

appService.token_transfer('transfer', 'bob', 'adam', 50);
printRewards();
appService.withdraw_pending_reward('bob'); // = 3 + 2 + 0.24 - 5 = 0.24
appService.withdraw_pending_reward('adam'); // = 4 + 1.22 - 4 = 1.22
printPoolSizeAndStaked();

appService.token_transfer('sell', 'bob', 'dex', 50);
printRewards();
appService.withdraw_pending_reward('bob'); // = 0 (balance = 0)
appService.withdraw_pending_reward('adam'); // = 4 + 1.22 + 1.46 - 5.22 = 1.46
printPoolSizeAndStaked();

appService.token_transfer('buy', 'dex', 'adam', 206);
printRewards();
appService.withdraw_pending_reward('bob'); // = 0 (balance = 0)
appService.withdraw_pending_reward('adam'); // = 4 + 1.22 + 1.46 + 6 - 5.22 = 6
printPoolSizeAndStaked(true);

=====

---BUY---
type: buy, from: dex, to: bob
type: buy, from: dex, to: adam
type: buy, from: dex, to: carol
type: buy, from: dex, to: eric
Pool size: 21
Total staked: 700
Bob reward: 6.928571428571429, balance: 100
Adam reward: 7.857142857142857, balance: 200
Carol reward: 5.785714285714285, balance: 300
Eric reward: 0.42857142857142816, balance: 100

```

---TRANSFER---

type: buy, from: dex, to: bob  
type: buy, from: dex, to: adam  
type: transfer, from: bob, to: adam  
Pool size: 10.45631067961165  
Total staked: 298.54368932038835  
Bob reward: 5.2439024390243905, balance: 50  
Adam reward: 5.21240824058726, balance: 248.54368932038835

---SELL---

type: buy, from: dex, to: bob  
type: buy, from: dex, to: adam  
type: sell, from: bob, to: dex  
Pool size: 10.45631067961165  
Total staked: 250  
Bob reward: 5.29126213592233, balance: 50  
Adam reward: 5.165048543689322, balance: 200

---WITHDRAW---

type: buy, from: dex, to: bob  
Bob reward: 3, balance: 100  
Adam reward: 0, balance: 0  
=> bob withdraw: 3  
type: buy, from: dex, to: adam  
Bob reward: 2, balance: 100  
Adam reward: 4.000000000000001, balance: 200  
=> bob withdraw: 2  
=> adam withdraw: 4.000000000000001  
type: transfer, from: bob, to: adam  
Bob reward: 0.24390243902439046, balance: 50  
Adam reward: 1.2124082405872594, balance: 248.54368932038835  
=> bob withdraw: 0.24390243902439046  
=> adam withdraw: 1.2124082405872594  
Pool size: 10.45631067961165  
Total staked: 298.54368932038835  
type: sell, from: bob, to: dex  
Bob reward: 0, balance: 0  
Adam reward: 1.4563106796116507, balance: 248.54368932038835  
=> bob withdraw: empty  
=> adam withdraw: 1.4563106796116507  
Pool size: 11.912621359223301  
Total staked: 248.54368932038835  
type: buy, from: dex, to: adam  
Bob reward: 0, balance: 0  
Adam reward: 6.000000000000002, balance: 448.54368932038835  
=> bob withdraw: empty  
=> adam withdraw: 6.000000000000002  
Pool size: 17.9126213592233  
Total staked: 448.54368932038835  
Distributed size: 17.9126213592233