تولید کد صوری از درخت نحو سالیدیتی

دانشجو: علی ابراهیمی

استاد راهنما: دکتر فاطمه قاسمی

دانشکده مهندسی برق و کامپیوتر، دانشگاه تهران

مقدمه / خلاصه

در این پروژه، ما تلاش کردیم تا روشي جدید برای تولید کد صوری از درخت نحو کد سالیدیتی با استفاده از قوانین ترجمه و زبان جاوا ارائه دهیم. هدف اصلی این پروژه، تسهیل فرآیند تبدیل کد های سالیدیتی پیش پردازش شده به کد mcrl2 بوده است.

- پروژه: پیاده سازی
- اهداف پروژه: تولید کد mcrl2 از کد سالیدیتی
- دستاورد ها و خروجی ها: تولید ابزاری برای تبدیل کد سالیدیتی به مدلی با زبان mcrl2

روش پیشنهادی

ما در این پروژه از زبان برنامه نویسی Java و ابزار Antlr برای ترجمه کد Solidity به زبان MCRL2 استفاده کرده ایم. برای این منظور، ابتدا گرامر Solidity را با استفاده از Antlr تعریف کردیم. سپس با استفاده از این گرامر، Antlr یك تحلیلگر (parser) و یك تجزیه کننده (lexer) برای زبان Solidity تولید کرد.

با استفاده از تحلیلگر و تجزیه کننده تولید شده توسط Antlr، قادر خواهیم بود کدهای Solidity را تجزیه و تحلیل کنیم. این تحلیلگر و تجزیه کننده قادر به تشخیص ساختارها، نحوهای زبان Solidity و استخراج اطلاعات مهم از قراردادهای هوشمند Solidity هستند.

بعد از تجزیه و تحلیل کدSolidity، میتوانیم با استفاده از AST به دست آمده از بخش قبلی، در زبان برنامه نویسی Java، کدهای Solidity را به زبان MCRL2 ترجمه کنیم. برای این منظور توابعی برای اعمال قواعد ترجمه ذکر شده در بخش قبل، نوشته و آن ها را بر روی AST اعمال می کنیم تا یك AST برای مدل MCRL2 به دست بیاوریم. در نهایت AST جدید را به صورت خروجی کد mcrl2 در محل مناسب ذخیره می کنیم.

برای ارزیابی کارایی و کاربردی بودن روش پیشنهادی، ما چندین نمونه کد سالیدیتی پیش پردازش شده را انتخاب کردیم و با استفاده از الگوریتم تبدیل، آنها را به کد mcrl2 تبدیل کردیم.

سپس نتایج حاصل شده از ابزاری که توسعه دادیم را با خروجی های مورد انتظار به ازای هر کد ورودی، مقایسه کردیم تا از صحت عملکرد این ابزار مطمئن شویم.

نتایج

در شکل زیر، نمونه ای از کد solidity ورودی و کد mcrl2 معادل آن که توسط ابزار توسعه یافته تولید شده است، قابل مشاهده است:

کد سالیدیتی:

```
contract DosAuction {
    address highestBidder = 0x0000000000000000000000000000000000000000;
    uint256 highestBid = 0;

    function bid() public payable {
        require(msg.value > highestBid, "Need to be higher than highest bid");
        if(highestBidder == 0x0000000000000000000000000000000000000000){
            highestBidder = msg.sender;
            highestBid = msg.value;
        }
        else{
            // Refund the old leader, if it fails then revert
            require(payable(highestBidder).send(highestBid), "Failed to send Ether");
            highestBidder = msg.sender;
            highestBid = msg.value;
        }
    }
}
```

خروجی معادل این کد درmcrl2:

```
sort Address;
cons Normal,Attacker,Owner,null,Contract : Address;
map
    equal : Address#Address -> Bool;
var ad1,ad2: Address;
eqn
    equal(Normal,Normal) = true;
    equal(Normal,Attacker) = false;
    equal(Normal,Owner) = false;
    equal(Normal,null) = false;
    equal(Normal,Contract) = false;
    equal(Attacker,Attacker) = true;
    equal(Attacker,Normal) = false;
    equal(Attacker,Owner) = false;
    equal(Attacker,null) = false;
    equal(Attacker,Contract) = false;
    equal(Owner,Owner) = true;
    equal(Owner,Normal) = false;
    equal(Owner,Attacker) = false;
    equal(Owner,null) = false;
    equal(Owner,Contract) = false;
    equal(null,null) = true;
    equal(null,Normal) = false;
    equal(null,Attacker) = false;
    equal(null,Owner) = false;
    equal(null,Contract) = false;
    equal(Contract,Contract) = true;
    equal(Contract,Normal) = false;
    equal(Contract,Attacker) = false;
    equal(Contract,Owner) = false;
    equal(Contract,null) = false;
    ad1 == ad2 = equal(ad1,ad2);


bid(highestBidder:Address,highestBid:Int,balance:Int,value:Int,addr:Address) =
    (value > highestBid) ->
        (bid_firstSuccess.
        ((highestBidder == null && highestBid == 0) ->
            (call_succBid.Auction(addr,value,balance + value))
        <>(highestBidder != null || highestBid != 0) ->
            ((call_transfer_EmptyFallback(highestBidder).bid_secondSuccess.call_succBid.
                Auction(addr,value,value)) +
            (call_transfer_NoFallback(highestBidder).bid_secondFail.call_failBid.
                Auction(highestBidder,highestBid,balance)) +
            (call_transfer_Fallback(highestBidder).bid_secondSuccess.call_succBid.
                Auction(highestBidder,highestBid,balance - highestBid)))
        <> delta))
    <> bid_firstFail.call_failBid.Auction(highestBidder,highestBid,balance);
```

جمع بندی

با استفاده از این روش ترجمه، ما قادر بودیم از قراردادهاي هوشمند Solidity استخراج اطلاعات مهمي را انجام دهیم و ساختارها و نحوهاي زبان Solidity را تشخیص دهیم. همچنین، با ترجمه کدهاي Solidity به زبان MCRL2، امکان تحلیل و بررسي خواص مدل MCRL2 براي قراردادهاي هوشمند فراهم شد.

کاربرد های صنعتی:

از جمله کاربرد های این ابزار توسعه داده شده، می توان به بررسی سناریوهای حمله در قرارداد های هوشمندی که به زبان solidity نوشته شده اند اشاره کرد و از آن برای شناسایی آسیب های موجود در قرارداد های هوشمند و جلوگیری از ضرر و زیان های حاصل از این ضعف ها استفاده کرد.

مراجع اصلی

1. **Solidity official documentation: https://docs.soliditylang.org/en/v0.8.24/**
2. **MCRL2 official documentation: https://www.mcrl2.org/web/index.html**
3. **GitHub repository of project: https://github.com/Mohadeseh-Rafiei/solidity**
4. **Smart contracts security service: https://mythx.io/**