Blockchain Oracle Problem

Subjects: Computer Science, Information Systems

Contributor: Giulio Caldarelli , alexander higgins

The Blockchain Oracle Problem refers to the inability of confirming the veracity of the data collected by oracles. Also, depending on the type, the chances of malfunction, and deliberate tampering.

Blockchain, Oracles, Smart Contracts

1. Why do we need "Oracles" on the Blockchain?

The term “oracle” comes from Greek mythology and refers to someone able to communicate directly with god and see the future. In ancient stories, people did not have enough information to make decisions and turned to oracles for knowledge beyond their understanding [1]. In the blockchain environment, oracles are systems that provide blockchain with information coming from the real world. If smart contracts do not deal with crypto exchange but with a decentralized mechanism involving weather, stock prices, or political events, a gateway from the external world is needed [2]. As the blockchain problem is to reach consensus, extrinsic information cannot be provided along with transaction data, since other nodes would detect information coming from an “untrusted” source. Therefore, information coming from the real world should come from a third-party univocal source, whose reliability is undisputed for all nodes: the oracle. Unlike Greek mythology, oracles (on the blockchain) do not predict the future but retrieve information from the past. To be precise, oracles, in general, do not insert information into the blockchain directly; conversely, they gather and store data from the real world. When a smart contract concerning extrinsic data is executed, the code then calls for the right information from a trusted oracle like BiteMyCoin. Examples of oracles are IoT systems such as probes and sensors, platforms such as ERP, or in the case of private data, the very human that operates directly on the blockchain. Oracles act as a bridge that can digest external and non-deterministic information into a format that a blockchain can understand [3]. Examples of data gathered by oracles comprise the following:

- Lottery winners;
- Natural disasters along with risk measurements;
- Price and exchange rate of real/crypto assets;
- Static data (e.g., country codes);
- Dynamic data (e.g., time measurements);
- Weather conditions;
- Political events;
- Sporting events;
- Geolocation and traceability information;

- Accidents;
- Events in other blockchains.

To better understand how oracles are necessary for smart contracts, let's consider an example of a crypto swap between waves and Neutrino USD. In that case, critical information for the smart contract to succeed, such as the waves/USD-N exchange rate, is missing (Figure 1). Data such as these are external to the blockchain, and without an oracle that updates rates, the contract could not be executed.

Figure 1. Crypto swap including an oracle.

This (decentralized) oracle's trustworthiness is somehow objective since, even if the oracle sets the price autonomously by merely browsing exchange prices online, any agent can verify if the exchange rate is correct or not. Different is the case where smart contracts operate in a situation in which oracles provide information that is hardly verifiable by the agents (centralized oracles). In those environments, the trustworthiness of oracles is fundamental. If the contracts involve highly valuable agreements, the oracle's chance to be compromised to benefit a particular party increases dramatically [4]. When all the parties cannot verify the oracle's data, and contract value increases, the oracle itself becomes a "problem".

2. The Blockchain Oracle Problem

Regarding blockchain and smart contracts, the oracle problem involves the trustworthiness and reliability of oracles. Curran [3] defined the oracle problem (in the blockchain) as "the security, authenticity, and trust conflict between third-party oracles and the trustless execution of smart contracts". To the best of the author's knowledge, the construct's origin can be spotted in a Reddit post by Dalovindj [5], definitely before the Ethereum environment for smart contracts was launched. The blogger realized that when executing an application on the bitcoin blockchain, regarding crowdfunding or gambling, verifying the reliability of extrinsic information without altering the consensus mechanism was indeed a problem: "*I think of it as 'The Oracle Problem'*".

In his dissertation, Egberts [6] extensively explained the drawbacks of the oracle problem, mainly describing it as a "*two step-back from decentralization*". As oracles are not distributed, they reintroduced the single-point-of-failure. Additionally, since they operate on non-deterministic data, their reliability needs to be trusted, removing trustless peer-to-peer interaction. Their implementation through smart contracts into the blockchain could also jeopardize users' trust who consider the blockchain as more reliable than legacy systems. Brilliantly shown by Antonopoulos

[7], a system built on oracles can also fail in two ways. If the oracle is trusted and cannot be compromised, there is still a chance that the data on which it is working has been altered, and then, although being a trustworthy device, it will feed the smart contracts with data that are untrue.

On the other hand, if the data are trusted and verified, the oracle may fail to operate correctly on the smart contract either due to malfunction or deliberate tampering. From a game-theoretical approach, it can be shown that the higher the value of the smart contract, the higher the incentive for the system to be compromised [4]. The oracle problem is also triggered in the case of attaching real assets on the blockchain through smart contracts. In a well-known article, Song [8] explained that in decentralized contexts (e.g., blockchain/smart contracts), linking a physical to a digital asset, whether it be fruit, cars, or houses, constitutes a critical issue. Tangible assets are regulated by the jurisdiction in which they reside, meaning that they are subject to something else (in some case, predominant) other than the smart contract. Indeed, this implies trusting something in addition to the smart contract. If, for example, a smart contract involves the property transfer of a house between two agents, the code will indeed swap the certificate between parties.

On the other hand, what happens in the real world may not be affected by the smart contract, as the former owner could refuse to leave the house. Without the involvement of a third party (e.g., government) that supervises the smart contracts, their enforcement is indeed not ensured. The need to trust a third party removes the “killer feature” of trustless applications, which in environments plagued with corruption represents a significant limitation. A considerable attempt to limit the oracle problem was made by Chainlink [9]. The start-up proposed a system of decentralized oracles, based on reputation, to reproduce the consensus mechanism of a blockchain. When deciding which data to upload on the blockchain, it takes into account the majority of oracles with the same data and the reputational level of each oracle. The data confirmed by the majority of the oracle are then uploaded on the chain [10]. This powerful system effectively addresses oracle malfunction or failures; however, deliberate data tampering or collusion could still be performed by the companies controlling the service. When decentralization is not sufficient to address the oracle problem, and data authenticity cannot be objectively verified, a “trust model” is needed for the smart contract environment to keep a certain degree of reliability [11]. As explained in a recent paper, a trust model is an intuitive scheme that outlines the reasons why the smart contract application should be trusted [12]. Failing to address the oracle problem poses a severe threat to investigating and developing real-world blockchain applications.

References

1. Buck, J. Blockchain Oracles Explained. Available online: https://cointelegraph.com/explained/blockchain-oracles-explained (accessed on 1 March 2020).
2. Apla What Is a Blockchain Oracle? Available online: https://blog.apla.io/what-is-a-blockchain-oracle-2ccca433c026 (accessed on 1 March 2020).

3. Curran, B. What Are Oracles? Smart Contracts, Chainlink & “The Oracle Problem. Available online: https://blockonomi.com/oracles-guide (accessed on 29 October 2020).
4. Sztorc, P. The Oracle Problem. Available online: https://www.infoq.com/presentations/blockchain-oracle-problems (accessed on 3 March 2020).
5. Dalovindj, U. The Oracle Problem. Available online: https://www.reddit.com/r/Bitcoin/comments/2p78kd/the_oracle_problem/ (accessed on 2 March 2020).
6. Egberts, A. The Oracle Problem—An Analysis of how Blockchain Oracles Undermine the Advantages of Decentralized Ledger Systems. SSRN Electron. J. 2017.
7. Antonopoulos, A.M. The Killer App: Bananas on the Blockchain? Available online: https://aantonop.com/the-killer-app-bananas-on-the-blockchain (accessed on 3 March 2020).
8. Song, J. The Truth about Smart Contracts. Available online: https://medium.com/@jimmysong/the-truth-about-smart-contracts-ae825271811f (accessed on 2 March 2020).
9. Harper, C. What Is ChainLink? A Beginner’s Guide to Decentralized Oracles. Available online: https://coincentral.com/what-is-chainlink-a-beginners-guide-to-decentralized-oracles/ (accessed on 12 March 2020).
10. Dale, O. What Is Chainlink? Guide to The Decentralized Oracle Network. Available online: https://blockonomi.com/chainlink-guide/ (accessed on 12 March 2020).
11. Antonopoulos, A.M.; Woods, G. Mastering Ethereum—Building Smart Contracts and DAPPS; O’Reilly: Sebastopol, CA, USA, 2018.
12. Caldarelli, G.; Rossignoli, C.; Zardini, A. Overcoming the blockchain oracle problem in the traceability of non-fungible products. Sustainability 2020, 12, 2391.

Retrieved from https://encyclopedia.pub/entry/history/show/131661