비트코인: 개인 대 개인 전자화폐 시스템

사토시 나카모토
www.bitcoin.org

요약. 전적으로 개인 대 개인 버전의 전자 화폐를 사용하면 금융 기관을 거치지 않고 한쪽에서 다른 한쪽으로 직접 온라인 결제를 실현할 수 있다.디지털 서명digital signature 은 그 해결책의 일부로 만약 이중지불double-spending 을 방지하기 위해 여전히 신뢰할만한 제3자가 필요하게 된다면 주요 이점이 사라진다. 그래서 우리는 네트워크를 이용해 이중지불 문제의 해결책을 제안한다. 네트워크는 지속적인 해시Hash 기반의 작업증명Proof of Work 체인에 거래들을 해시화 하여 타임스탬프Timestamp 를 찍어 해당 작업증명을 다시 수행하지 않고는 변경할 수 없는 기록을 생성한다. 가장 긴 체인은 일련의 거래 사건들을 증명할 뿐 아니라, 이것이 가장 큰 CPU파워 풀에서 생성했음을 증명한다. 네트워크에 참여하는 과반수 CPU파워가 네트워크를 공격하지 않는 데 협력하는 노드에 의해 통제되는 한, 가장 긴 체인을 생성하여 공격자를 앞지를 것이다. 이 네트워크 자체는 최소한의 구조만 요구한다. 노드들은 최선의 노력 기반으로 메시지가 전송되며 자신들이 네트워크를 떠나 있을 때 생성된 가장 긴 작업증명 체인을 거래의 증명으로 받아들임으로써 마음대로 네트워크를 떠나거나 재 합류할 수 있다.

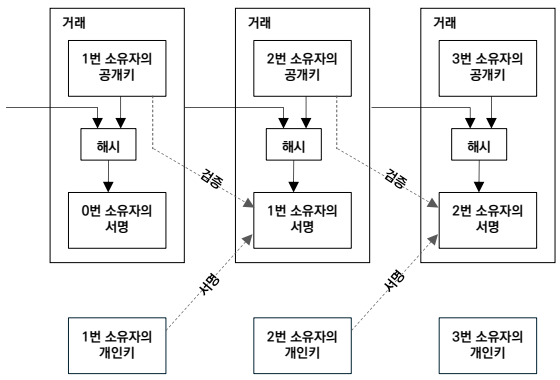
1. 서론

인터넷 상거래는 전자 결제를 처리하는 신뢰할 수 있는 제3자 역할을 하는 금융기관에 거의 전적으로 의존해 왔다. 이 시스템은 대부분의 거래에서 충분히 잘 작동하지만, 여전히 신뢰 기반 모델의 내재적인 취약점을 안고 있다. 금융기관이 분쟁을 중재해야 하므로 실제로 완전히 되돌리지 못하는 거래non-reversible는 가능하지 않다. 중재 비용은 거래 비용을 증가시켜 실질적으로 최소 거래 규모를 제한하고 일상적인 소액 거래의 가능성을 가로막으며, 되돌릴 수 없는 서비스에 대한 지급을 환불불가 조건으로 하는 등 더 큰 비용이 발생한다. 거래를 되돌릴 수 있는 가능성이 있으면 신뢰할 만한 제3자의 필요성이 더 커진다. 판매자는 고객을 의심해야 하며, 필요 이상으로 많은 정보를 요구할 수 있다. 어느 정도의 사기는 피할 수 없는 것으로 간주한다. 실물 화폐를 사용하면 이러한 비용과 결제의 불확실성을 피할 수 있지만, 신뢰할 수 있는 제3자 없이 온라인을 통해 결제할 수 있는 방법은 존재하지 않는다.

신뢰 대신 암호화 증명cryptographic proof을 기반으로 하는 전자 결제 시스템이 필요하며, 이를 통해 신뢰할 수 있는 제3자 없이도 두 당사자가 직접 거래할 수 있다. 거래를 전산적으로 되돌릴 수 없게 한다면 판매자를 사기로부터 보호할 수 있고, 구매자를 보호하기 위한 일상적인 제3자 예치 방법을 쉽게 구현할 수 있을 것이다. 백서에서 개인 대 개인 분산 타임스탬프 서버를 사용해 거래의 시간 순서에 대한 전산적 증명을 생성하는 이중 지불 문제에 대한 해결책을 제안한다. 정직한 노드가 공격자 노드보다 더 많은 CPU 성능을 집단적으로 통제하는 한 안전하다.

2. 거래

우리는 전자 화폐electronic coin를 디지털 서명의 체인으로 정의한다. 각 화폐 소유자는 자신에게 그 화폐를 보낸 직전 거래 명세the previous transaction 및 그 화폐를 받는 다음 소유자의 공개키the public key of the next owner 를 해시 처리한 값에 디지털 방식으로 서명하고 이를 화폐 끝에 추가해 다음 소유자에게 송금한다. 수취인은 그 서명을 검증해 화폐 소유권의 체인을 검증할 수 있다.

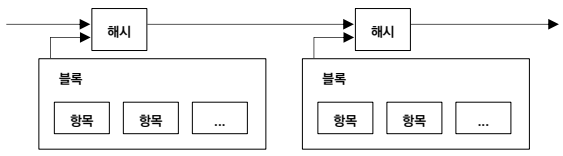


이 과정의 문제는 화폐 소유자 가운데 이중 지불하지 않았는지 수취인이 검증할 수 없다는 점이다. 일반적인 해결책은 조폐국처럼 신뢰할 수 있는 중앙 기관을 두어 모든 거래에서 이중 지불을 검증하는 것이다. 각 거래 후에는 화폐를 조폐국에 반환하여 새 화폐를 발행해야 하며, 조폐국에서 직접 발행한 화폐만 이중 지불이 발생하지 않았다고 신뢰할 수 있다. 이 해결책의 문제점은 전체 통화체계의 운명이 조폐국을 운영하는 회사에 달려 있으며, 모든 거래가 은행처럼 조폐국을 거쳐야 한다는 것이다.

우리는 직전 화폐 소유자가 앞선 어떤 거래에도 서명하지 않았다는 사실을 수취인에게 알릴 방법이 필요하다. 이런 목적에 따라 가장 앞선 거래 하나만 인정하고, 뒤따르는 이중 지불 시도를 무시한다. 이중 지불 거래가 없음을 확인하는 유일한 방법은 모든 거래를 파악하는 것이다. 조폐국 기반 모델에서는 조폐국이 모든 거래를 파악하고 어떤 거래가 먼저 도착했는지 결정했다. 신뢰할 수 있는 제3자 없이 이를 수행하려면 거래가 공개적으로 알려져야 하며[1], 참여자들이 수취 순서에 대한 단일 이력에 동의할 수 있는 시스템이 필요하다. 수취인은 각 거래가 일어날 때마다 다수의 노드가 가장 먼저 받은 거래라는 데 동의했다는 증명이 필요하다.

3. 타임스탬프 서버

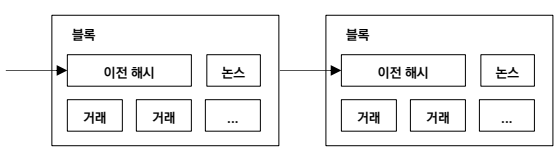
우리가 제안하는 해결책은 타임스탬프 서버에서 시작한다. 타임스탬프 서버는 타임스탬프가 찍힌 항목들의 블록 해시를 가져와서 신문이나 인터넷 게시물과 같이 널리 게시하는 방식으로 작동한다[2-5]. 타임스탬프는 그 데이터가 해시에 들어가기 위해 명백히 그 당시에 존재했음을 증명한다. 각 타임스탬프는 그 해시에 이전 타임스탬프를 포함한 체인을 형성하게 되고, 이후에 추가되는 타임스탬프는 그 앞의 타임스탬프를 강화한다.



4. 작업증명

개인 대 개인 기반으로 분산 타임스탬프 서버를 구현하려면 신문이나 인터넷 게시물이 아닌 아담 백Adam Back의 해시캐시Hash Cash[6]와 유사한 작업증명 시스템을 사용해야 한다. 작업증명은 SHA-256과 같은 해시 처리를 거쳐 결과값이 0비트zero bits 여러 개로 시작하는 특정 값을 찾는 작업을 수반한다. 평균적으로 작업증명의 소요 시간은 특정 값이 요구하는 0비트 개수에 따라 기하급수적으로 증가하며, 작업증명은 단 한번의 해시 함수를 실행하여 검증할 수 있다.

타임스탬프 네트워크의 경우, 블록의 해시에 필요한 0비트 자릿수에 해당하는 값을 발견될 때까지 블록의 임의의 값 논스nonce를 증가시켜 가며 작업증명을 실행한다. 작업증명을 충족시키기 위한 CPU 자원이 소모된 후에는 작업을 다시 실행하지 않고는 블록을 변경할 수 없다. 이후에 생성되는 블록은 그 뒤에 연속적으로 연결되므로, 블록을 변경하려면 그 뒤에 있는 모든 블록을 다시 실행해야 한다.



작업증명은 다수결 의사 결정에서 대표성을 결정하는 문제도 해결한다. 만약 과반수가 IP 주소 하나당 1표에 기반한다면, 많은 IP를 할당할 수 있는 사람이 이를 장악할 수 있다. 작업증명은 본질적으로 CPU 하나당 1개의 투표권이다. 과반수에 의한 결정은 가장 긴 체인으로 대표되며, 가장 많은 작업증명 노력이 투입된 것이다. 대부분의 CPU 파워가 정직한 노드에 의해 통제된다면, 정직한 노드가 통제하는 체인은 가장 빠르게 늘어나며 경쟁관계에 있는 다른 체인들을 앞지르게 될 것이다. 공격자가 과거 블록을 변경하려면 해당 블록과 그 이후에 연결된 모든 블록의 작업증명을 다시 수행한 다음 정직한 노드의 작업증명을 따라잡고 체인의 길이를 앞질러야 한다. 후속 블록이 추가될수록 느린 공격자가 따라잡을 확률은 기하급수적으로 줄어든다는 것을 보여주겠다.

시간이 지남에 따라 노드 운영 하드웨어 속도가 증가하고 관여도interest가 변화하면 이를 보정하기 위한 작업증명 난이도difficulty는 시간당 생성되는 평균 블록 수를 목표로 하는 이동 평균에 의해 결정된다. 블록이 너무 빨리 생성되면 작업증명 난이도가 증가한다.

5. 네트워크

네트워크를 실행하는 단계는 다음과 같다:

1) 새로운 거래가 모든 노드에 전파된다.
2) 각 노드는 새로운 거래를 하나의 블록에 수집한다.
3) 각 노드는 블록에 맞는 난이도의 작업증명을 찾는다.
4) 한 노드가 작업증명의 값을 찾으면 모든 노드에 해당 블록을 전파한다.
5) 노드는 블록에 포함된 모든 거래가 유효하고 아직 사용되지 않은 경우에만 블록을 받아들인다.
6) 노드는 자신이 받아들였음을 나타내기 위해 앞선 블록의 해시를 직전 해시로 사용하여 체인에서 다음 블록을 생성하는 작업을 수행한다

노드는 항상 가장 긴 체인을 올바른 체인으로 간주하고 그것을 이어 나가는 작업을 계속한다. 만약 두 노드가 동시에 서로 다른 버전의 다음 블록을 전파하는 경우, 일부 노드가 그 중 하나를 먼저 받는다. 이 경우 노드는 먼저 받은 블록을 가지고 작업하지만, 다른 분기branch도 보관해 그 쪽이 더 길어질 경우를 대비한다. 다음 작업증명의 값이 발견되고 어느 한쪽의 분기가 더 길어지면 두 가지의 묶음tie이 풀어지고, 다른 쪽 분기에서 작업하던 노드는 다른 분기로로 전환한다.

전파한 새로운 거래가 반드시 모든 노드에 도달할 필요는 없다. 많은 노드에 도달하기만 하면 얼마 지나지 않아 블록에 포함될 것이다. 또한 블록 전파는 메시지 누락을 허용한다. 노드가 블록을 받지 못한 경우, 다음 블록을 받으면서 해당 블록을 놓쳤다는 사실을 알게 되면서 블록을 요청한다.

6. 인센티브

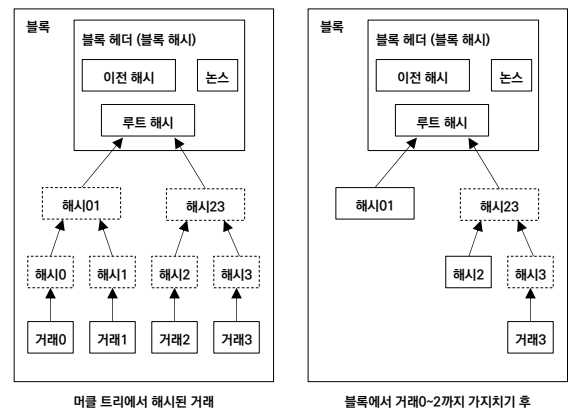
관례상 블록의 첫 번째 거래는 블록 생성한 자에게 소유하는 화폐를 발행하는 특별한 거래다. 이는 노드가 네트워크를 지원할 인센티브를 더해 주며, 화폐를 발행하는 중앙기관 없이 화폐를 유통할 수 있는 방법을 제공한다. 새로운 화폐를 일정량으로 꾸준히 추가하는 것은 금 채굴자가 금을 유통하기 위해 자원을 소비하는 것과 유사하다. 우리의 경우에는 CPU 연산시간과 전기가 소비된다.

이 인센티브는 거래 수수료transaction fee로 충당될 수도 있다. 거래에서 출금액output value이 입금액input value보다 작다면 그 차액이 거래를 포함한 블록의 인센티브 금액에 더해질 거래 수수료다. 미리 정해진 수만큼의 화폐가 유통되면 인센티브는 전적으로 거래 수수료로 전환되어 인플레이션이 전혀 발생하지 않는다.

이 인센티브는 노드가 정직한 참여자로 남아있게 하는데 도움이 될 수 있다. 만약 탐욕스러운 공격자가 모든 정직한 노드보다 더 많은 CPU 파워를 모을 수 있다면, 이를 사용하여 자기가 지불한 금액을 도로 훔치고 사람들을 속일지 새로운 화폐를 만들지 선택해야 한다. 공격자는 CPU 파워를 이용해 시스템과 자신의 부를 훼손하는 것보다 규칙을 따르는 것, 즉 다른 모든 사람을 합친 것보다 더 많은 새로운 화폐를 얻는 것이 자신에게 유리하다는 것을 알게 될 것이다.

7. 디스크 공간 재확보

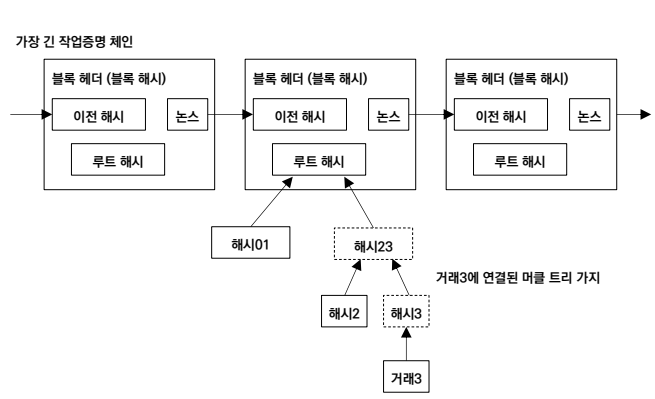
화폐의 최신 거래가 충분한 블록 아래에 묻히면, 그 전에 사용된 거래는 디스크 공간을 절약하기 위해 폐기될 수 있다. 이 블록의 해시를 깨지 않고 가능하게 하기 위해 거래는 루트root만 블록의 해시에 포함된 채로 머클 트리Merkel Tree[7][2][5]로 해시 된다. 오래된 블록은 트리의 분기를 잘라내어 압축할 수 있고 내부 해시는 저장할 필요가 없다.



거래가 들어있지 않은 블록 헤더Block Header는 약 80바이트이다. 블록이 10분마다 생성된다고 가정하면 연간 4.2MB씩 증가 한다.(80바이트 * 6 * 24 * 365 =4.2MB) 2008년 현재 일반적으로 판매되는 컴퓨터에는 2GB 램이 있고 무어의 법칙에 따르면 매년 1.2GB의 성장을 예측하고 있기 때문에 블록 헤더를 메모리에 보관한다 해도 저장공간은 문제가 되지 않는다.

8. 간소화된 지불 검증

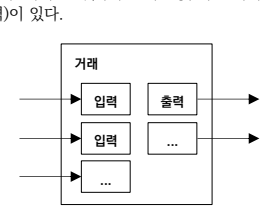
전체 네트워크 노드full network node를 실행하지 않고도 지불을 검증할 수 있다. 사용자는 가장 긴 작업증명 체인의 블록 헤더 사본만 보관하면 되는데, 가장 긴 체인이 맞다고 확신할 때까지 거래와 타임스탬프가 찍힌 블록을 연결하는 머클 트리 분기를 네트워크 노드에 요청하여 얻을 수 있다. 거래를 직접 확인할 수는 없지만, 이를 체인의 한 지점에 연결하면 네트워크 노드가 거래를 받아들였음을 확인할 수 있으며 이후 추가된 블록을 볼 수 있다.



따라서 정직한 노드가 네트워크를 통제하는 한 검증은 신뢰할 수 있지만, 공격자가 네트워크를 압도할 경우 더 취약해진다. 네트워크 노드가 스스로 거래를 검증할 수는 있지만, 공격자가 네트워크를 계속 장악할 수 있는 한 조작된 거래로 속일 수 있다. 이를 방지하기 위한 한 가지 전략은 유효하지 않은 블록을 감지한 네트워크 노드의 경고를 받아들여 사용자 소프트웨어가 받아들여 온전한 블록과 경고를 받은 거래를 다운로드하여 불일치된 부분을 확인하도록 하는 것이다. 지불금을 자주 일어나는 기업은 좀 더 독립적인 보안과 빠른 검증을 위해 자체 노드를 운영하길 원할 것이다.

9. 금액 합치기와 나누기

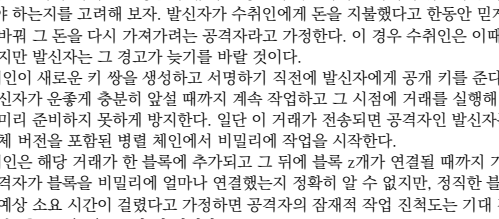
화폐를 개별적으로 처리할 수는 있겠지만, 최소 단위를 송금할 때마다 별도의 거래로 분리하는 것은 번거로운 일이다. 이것을 합치고 나눌 수 있도록 거래에는 여러 개의 입력과 출력이 포함된다. 일반적으로 더 큰 직전 거래의 단일 입력 또는 더 작은 금액을 합친 여러 개의 입력이 있으며, 최대 두 개의 출력(하나는 지불용, 다른 하나는 잔돈이 있는 경우 발신자에게 반환하는 출력)이 있다.



한 거래가 여러 거래에 의존하고 그 거래가 더 많은 거래에 의존하는 부채꼴fan-out 구조는 여기서 문제가 되지 않는다는 점에 유의해야 한다. 거래 기록의 완전한 독립형 사본을 추출할 필요가 전혀 없다.

10. 프라이버시

전통적인 은행 모델은 정보에 대한 액세스를 관련 당사자와 신뢰할 수 있는 제3자로 제한하여 일정 수준의 프라이버시를 달성한다. 모든 거래를 공개 할 필요성 때문에 이 방식은 사용하지 못하지만, 공개 키를 익명으로 유지하여 다른 곳의 정보 흐름을 차단하는 것으로 프라이버시를 유지할 수 있다. 대중은 누군가가 다른 사람에게 송금하는 것을 볼 수 있지만, 그 거래와 연결된 누군가의 정보는 볼 수 없다. 이는 증권 거래소에서 공개되는 정보 수준과 비슷하게, 개별 거래의 시간과 매매 규모를 나타내는 "테이프"는 공개되지만 당사자가 누구인지 알지는 못하는 것이다.



추가 보호수단으로 각 거래마다 새로운 키 쌍을 사용하여 공통 소유자에게 연결되지 않도록 해야 한다. 여러 입금이 동일 소유자의 소유임을 부득이 드러내는 다중입금 거래에서 일부 연결은 여전히 불가피하다. 키의 소유자가 밝혀지면 연결 때문에 동일한 소유자의 다른 거래까지 밝혀질 위험이 있다.

11. 계산

우리는 정직한 체인보다 더 빠른 대체 체인을 생성하려는 공격자의 시나리오를 고려한다. 이 시나리오가 성공하더라도, 시스템이 무에서 가치를 창출하거나 공격자가 소유한 적이 없는 돈을 가져가는 등의 무단 변경을 허용하지 않는다. 노드는 유효하지 않은 거래를 지불로 수락하지 않으며, 정직한 노드는 이러한 거래가 포함된 블록을 절대 수락하지 않는다. 공격자가 할 수 있는 것은 최근에 사용한 돈을 되찾기 위해 자신의 거래 중 하나만 바꾸는 것이다.

정직한 체인과 공격자 체인 사이의 경쟁은 이항 랜덤 워크Binomial Random Walk 로 특징지을 수 있다. 성공 사건은 정직한 체인이 한 블록씩 연장되어 +1만큼 앞서게 되는 것이고, 실패 사건은 공격자 체인이 한 블록씩 연장되어 -1만큼 격차가 줄어드는 것이다.

공격자가 주어진 열세를 따라잡을 확률은 도박꾼의 파산Gambler's Ruin 문제와 유사하다. 신용이 무제한인 도박꾼이 적자 상태에서 시작하여 손익분기점에 도달하려는 시도를 잠재적으로 무한한 수로 수행한다고 가정해 보겠다. 우리는 다음과 같이 그가 손익분기점에 도달할 확률, 즉 공격자가 정직한 체인을 따라잡을 확률을 계산할 수 있다[8]:

p = 정직한 노드가 다음 블록을 찾을 확률
q = 공격자가 다음 블록을 찾을 확률
q_z = 공격자가 블록 z 개 뒤에서 따라잡을 확률

$$q_z = \begin{cases} 1 & if\ p \le q \\ (q/p)^z & if\ p > q \end{cases}$$

p 〉 q라고 가정하면 공격자가 따라잡아야 하는 블록의 수가 증가함에 따라 확률은 기하급수적으로 떨어진다. 공격자가 초반에 운좋게 치고 나가지 않으면, 뒤처질수록 따라잡을 확률은 점점 더 낮아진다.

이제 발신자가 거래를 변경할 수 없다고 충분히 확신하기 전까지 수취인이 얼마나 기다려야 하는지를 고려해 보자. 발신자가 수취인에게 돈을 지불했다고 한동안 믿게 한 다음, 거래를 바꿔 그 돈을 다시 가져가려는 공격자라고 가정한다. 이 경우 수취인은 이때 경고를 받게 되지만 발신자는 그 경고가 늦기를 바랄 것이다.

수취인이 새로운 키 쌍을 생성하고 서명하기 직전에 발신자에게 공개 키를 준다. 이렇게 하면 발신자가 운좋게 충분히 앞설 때까지 계속 작업하고 그 시점에 거래를 실행해 블록의 체인을 미리 준비하지 못하게 방지한다. 일단 이 거래가 전송되면 공격자인 발신자는 자신의 거래 대체 버전을 포함된 병렬 체인에서 비밀리에 작업을 시작한다.

수취인은 해당 거래가 한 블록에 추가되고 그 뒤에 블록 z개가 연결될 때까지 기다린다. 그는 공격자가 블록을 비밀리에 얼마나 연결했는지 정확히 알 수 없지만, 정직한 블록이 블록당 예상 소요 시간이 걸렸다고 가정하면 공격자의 잠재적 작업 진척도는 기대 값을 갖는 푸아송 분포Poisson distribution 가 될 것이다:

$$\lambda = z\frac{q}{p}$$

현재 공격자가 여전히 따라잡을 수 있는 확률을 얻기 위해, 우리는 공격자가 달성할 수 있는 각 진척도에 대한 푸아송 밀도Poisson density와 그가 해당 지점에서 따라잡을 수 있는 확률을 곱한다.

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \begin{cases} (q/p)^{(z-k)} & if\ k \le z \\ 1 & if\ k > z \end{cases}$$

해당 분포의 무한한 꼬리를 합산하지 않게 재배열하고...

$$1 - \sum_{k=0}^{z} \frac{\lambda^k e^{-\lambda}}{k!} \left(1 - (q/p)^{(z-k)}\right)$$

C 코드로 바꿔서...

```
#include <math.h>
double AttackerSuccessProbability(double q, int z)
{
        double p = 1.0 - q;
        double lambda = z * (q / p);
        double sum = 1.0;
        int i, k;
        for (k = 0; k <= z; k++)
        {
                double poisson = exp(-lambda);
                for (i = 1; i <= k; i++)
                     poisson *= lambda / i;
                sum -= poisson * (1 - pow(q / p, z - k));
        }
        return sum;
}
```

결과를 실행하면, z 값에 따라 기하급수적으로 떨어지는 확률을 볼 수 있다.

q=0.1		q=0.3		0.1% 미만의 p를 풀면
z=0	P=1.0000000	z=0	P=1.0000000	
z=1	P=0.2045873	z=5	P=0.1773523	P < 0.001
z=2	P=0.0509779	z=10	P=0.0416605	q=0.10 z=5
z=3	P=0.0131722	z=15	P=0.0101008	q=0.15 z=8
z=4	P=0.0034552	z=20	P=0.0024804	q=0.20 z=11
z=5	P=0.0009137	z=25	P=0.0006132	q=0.25 z=15
z=6	P=0.0002428	z=30	P=0.0001522	q=0.30 z=24
z=7	P=0.0000647	z=35	P=0.0000379	q=0.35 z=41
z=8	P=0.0000173	z=40	P=0.0000095	q=0.40 z=89
z=9	P=0.0000046	z=45	P=0.0000024	q=0.45 z=340
z=10	P=0.0000012	z=50	P=0.0000006	

12. 결론

우리는 신뢰에 의존하지 않고 전자 거래를 할 수 있는 시스템을 제안했다. 강력한 소유권을 제공하는 디지털 서명으로 만들어진 화폐라는 일반적인 프레임워크로 시작했지만 이중 지불 방지 없이는 불완전하다. 이를 해결하기 위해 우리는 작업증명을 사용해 거래의 공개 내역을 기록하는 개인 대 개인 네트워크를 제안했고, 정직한 노드가 CPU 파워의 대부분을 제어할 경우 공격자가 이를 변경하는 것이 전산적으로 불가능해 진다. 이 네트워크의 견고함은 비정형적 단순함unstructured simplicity에 있다. 노드는 거의 조율coordination 없이 한꺼번에 작동한다. 메시지가 특정 위치의 경로를 지정받아 가는 게 아니라 단지 최선의 노력을 기반으로 전달되기만 하면 되기 때문에 노드가 식별될 필요가 없다. 노드는 네트워크에서 마음대로 떠났다가 다시 참여할 수 있으며, 자신이 없는 동안 일어난 일의 증명으로 작업증명 체인을 받아들인다. 노드는 자신의 CPU 파워로 투표한 유효한 블록에 대해서는 확장 작업을 통해 수락을 표현하고, 유효하지 않은 블록에 대해서는 작업을 거부함으로써 거부 의사를 표시한다. 필요한 모든 규칙과 인센티브는 합의 작용consensus mechanism을 통해 시행할 수 있다.