ブロックチェーン公開講座 第3回

ビットコインその2

芝野恭平

東京大学大学院工学系研究科技術経営戦略学専攻
ブロックチェーンイノベーション寄付講座
特任研究員
shibano@tmi.t.u-tokyo.ac.jp

トランザクション

概要

- UTXOモデル
- トランザクションデータ
  - トランザクションID
  - インプット，アウトプット
  - 手数料
- 鍵の使い方
  - ロッキングスクリプト，アンロックスクリプト
- 様々なトランザクション5種類
  - Pay-to-Public-Key-Hash (P2PKH)
  - Pay-to-Public-Key
  - データアウトプット (OP_RETURN)
  - Multi-Signature
  - Pay-to-Script-Hash (P2SH)

トランザクションのライフサイクル

組成

- どのトランザクションからどこのアドレスに送金するか
- 署名

ブロードキャスト

- ビットコインネットワークに送信
- 複数のノードに送信
- トランザクションは機密情報が含まれていないのでオープンにしてもよい
  - クレジットカードとは異なる

伝搬

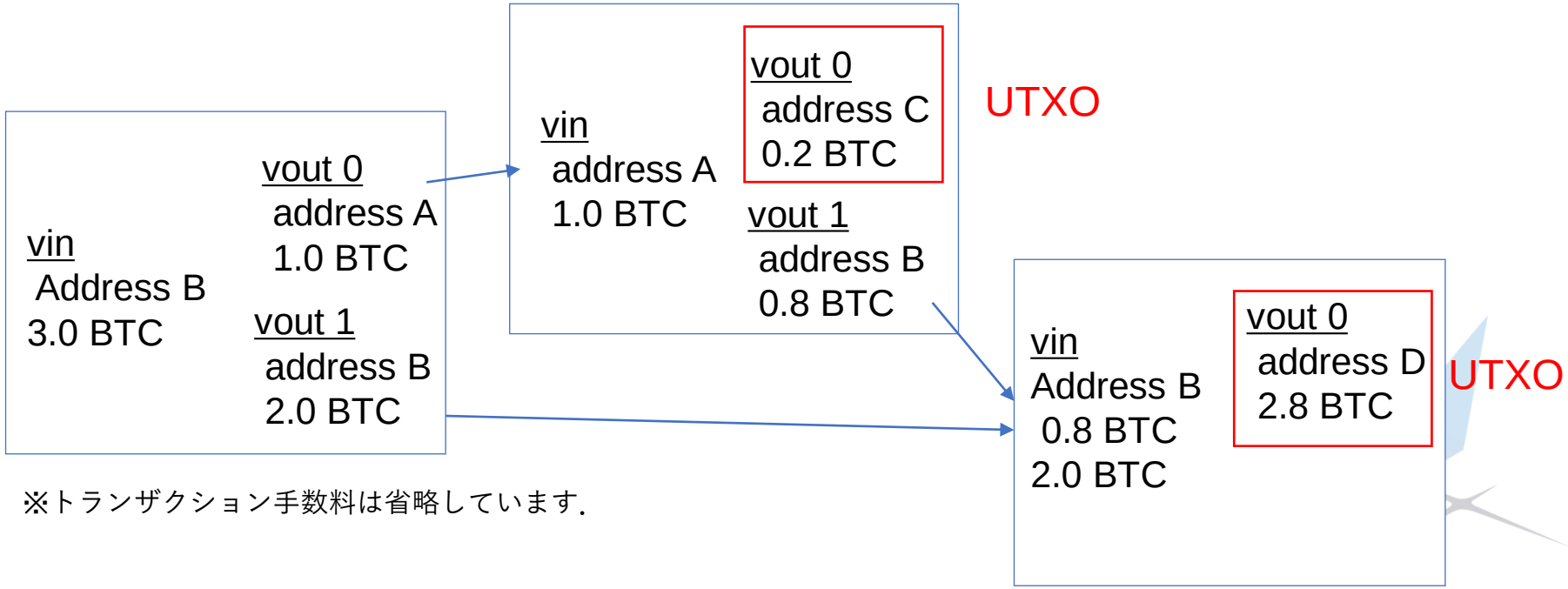
- 受け取ったノードはトランザクションを検証する
- 有効な場合ほかのノードに伝搬
  - 隣接ノード（接続されたすべてのノード）へ伝搬
  - トランザクションプールに入れる
- 無効な場合破棄

記録

- マイニングノードにより新しいブロックに取り込まれる

トランザクション間の関係例

- トランザクションはブロックに含まれる，送金情報
- どのトランザクションからどのアドレスに対していくら送金したか
- 未処理のトランザクションアウトプット（UTXO）から新しくトランザクションを生成できる
  - 元のトランザクションで送金された額を残高として，さらにほかのアドレスに送金する



※トランザクション手数料は省略しています.

トランザクションデータ形式

- トランザクションデータの形式：固定長データ形式，hexadecimal notation

トランザクションID：
0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

0100000001186f9f998a5aa6f048e51dd8419a14d8a0f1a8a2836dd734d2804fe65fa35779000000008b483045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e381301410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adfffffffff0260e3160000000000001976a914ab68025513c3dbd2f7b92a94e0581f5d50f654e788acd0ef8000000000001976a9147f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a888ac00000000

Example 1. Alice’s transaction, serialized and presented in hexadecimal notation

見やすく変換


```
{
  "version": 1,
  "locktime": 0,
  "vin": [
 {
"txid": "7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",
"vout": 0,
"scriptSig" : "3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813[ALL]
0484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4
"sequence": 4294967295
 }
  ],
  "vout": [
 {
"value": 0.01500000,
"scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG
 },
 {
"value": 0.08450000,
"scriptPubKey": "OP_DUP OP_HASH160 7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8 OP_EQUALVERIFY OP_CHECKSIG",
 }
  ]
}
```


60e3160000000000が送金額
小さい数から並んでいる（リトルエンディアン）．
0x16e360=1,500,000 satoshi = 0.015 BTC
（1億 satoshi = 1 BTC)

トランザクションID

- トランザクションを一意に識別する識別子
- トランザクションデータをダブルハッシュ（SHA256を2回）した値バイトオーダー逆順にしたもの

- トランザクションハッシュとも呼ばれる.
- 前ページの例だと以下：

トランザクションID：
0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

0100000001186f9f998a5aa6f048e51dd8419a14d8a0f1a8a2836dd734d2804fe65fa35779000000008b483045022100884d142d86652a
3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f
6e381301410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4
938de5cc17b4a10fa336a8d752adffffffff0260e31600000000001976a914ab68025513c3dbd2f7b92a94e0581f5d50f654e788acd0ef800
0000000001976a9147f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a888ac00000000

```
shibano@DESKTOP-940THE0:~$ bx sha256
0100000001186f9f998a5aa6f048e51dd8419a14d8a0f1a8a2836dd734d2804fe65fa35779000000008b483045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb64
98c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e381301410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5
412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adffffffff0260e31600000000001976a914ab68025513c3dbd2f7b92a94e0581f5d50f654e788acd0ef80000
00000001976a9147f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a888ac00000000|bx sha256
f2c245c38672a5d8fba5a5caa44dcef277a52e916a0603272f91286f2b052706
```

バイトオーダーを逆順にするとトランザクションIDと一致していることが確認できる.

トランザクションのデータ構造


```
{
  "version": 1,
  "locktime”: 0,
  “vin”: [
 {
“txid”: “7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",
"vout": 0,
"scriptSig" :
"3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813[
ALL] 0484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf",
"sequence": 4294967295
 }
  ],
  "vout": [
 {
"value": 0.01500000,
"scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG"
 },
 {
"value": 0.08450000,
"scriptPubKey": "OP_DUP OP_HASH160 7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8 OP_EQUALVERIFY OP_CHECKSIG",
 }
  ]
}
```


この例では，1つのInput，2か所のOutputで構成されます．

TransactionのInputとOutput

“vin”: [
{
“txid”:
“7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18"",
"vout": 0,
・・・
],

"vout": [
{
"value": 0.01500000,
・・・
},
{
"value": 0.08450000,
・・・
]

インプットの
TransactionID

インプットトランザクション内のアウトプットインデックス

ブロックエクスプローラーでトランザクションを見てみる

https://www.blockchain.com/ja/btc/tx/0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

送金額合計は
0.015 + 0.0845 = 0.0995 BTC
足らない0.0005BTCはトランザクション手数料となる.

トランザクション手数料

- トランザクション手数料はトランザクションの作成者によって決められる.
  - ほとんどのウォレットは，トランザクション手数料を自動的に計算して入れる.
  - 手動で決めることもできる.
  - トランザクションの大きさに基づいて計算され，送金金額とは関係ない

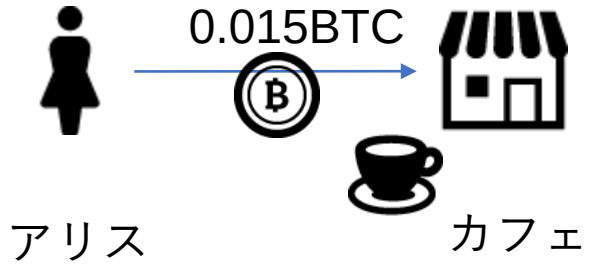
- 手数料はマイナーに報酬として与えられる.

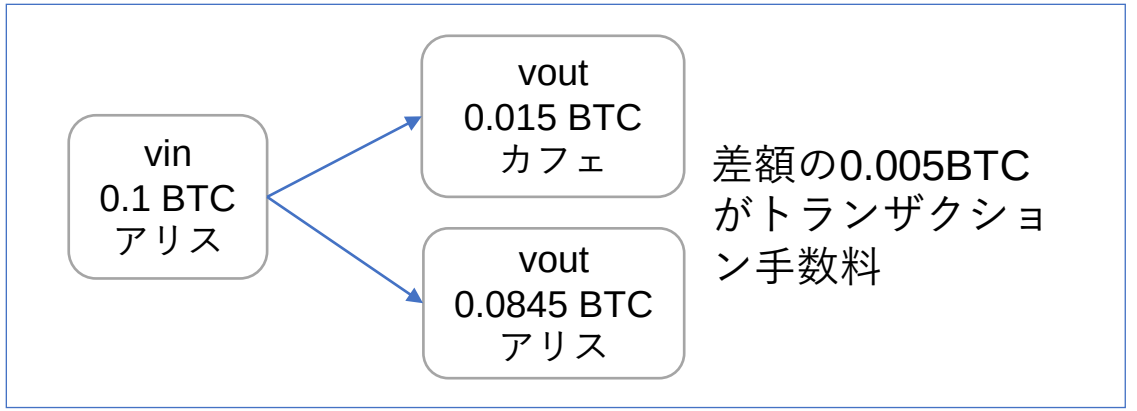
- マイナーは，どのトランザクションをブロックに含めるかを手数料によって決めている
  - ブロックサイズ 1MB
  - トランザクションの大きさ
  - 手数料

- サイズあたりの手数料を高く設定すればするほどブロックに取り込まれるのが速くなる傾向がある.

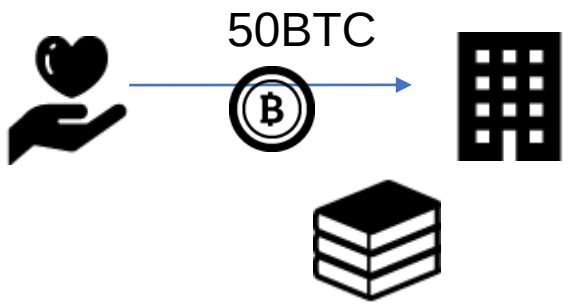
トランザクション手数料の高い・低い例

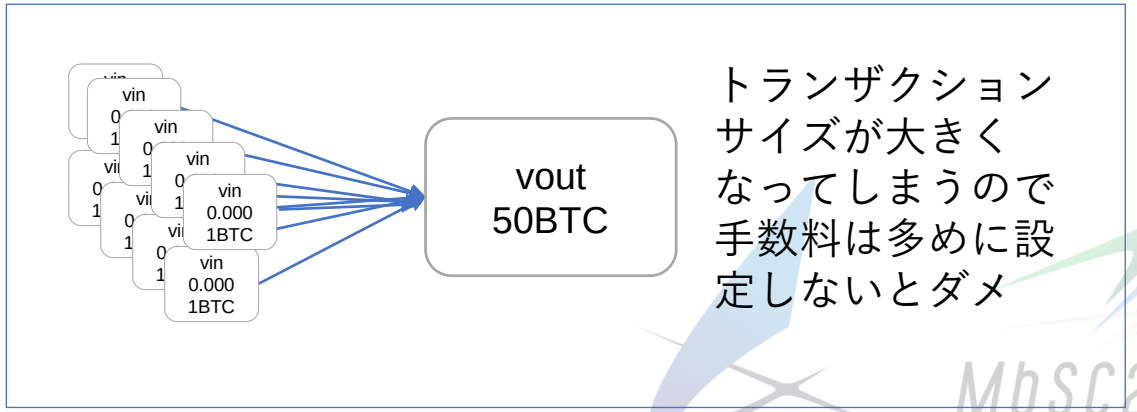
アリスのコーヒー代金支払い





チャリティ団体が募金で教科書を買う

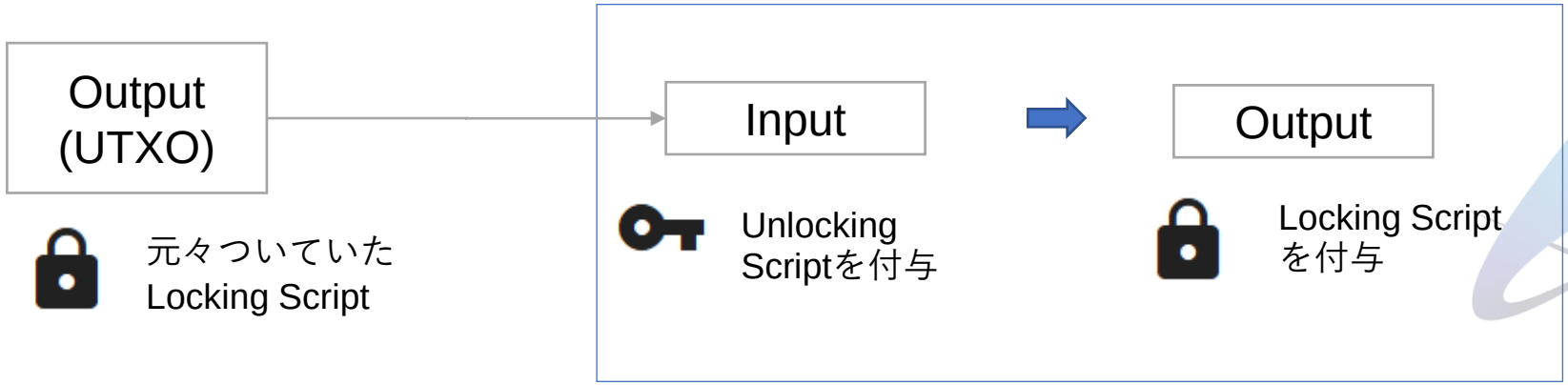




Locking Script / Unlocking Script

- Outputのトランザクションは将来，UTXO（未処理のトランザクション）となり，そのアドレスの所有者が送金できる．
- 所有者「だけ」が送金できるようにしなければならない．
- これをLocking Script / Unlocking Scriptという仕組みで解決する．
  - Pay-to-Public-Key-Hashというのが標準的な送金の仕組み

- OutputトランザクションにはLocking Scriptがついている
- そのトランザクションをInputにするには対応するUnlocking Scriptがないとダメ．
  - 不正なUnlocking Scriptのトランザクションはノード到達時に検証に失敗するため伝搬されないしブロックに取り込まれない．
  - 正しいUnlocking Script作成には秘密鍵が必要



Locking Script/Unlocking Script

Transactionをもう一度見てみる.

{“vin”: [
{
“txid”: “7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",
“scriptSig” : “30450221008・・・・",
}
],
"vout": [
{
"value": 0.01500000,
"scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG"
},
]}

voutにあるvalueは送金金額で，scriptPubKeyというのがLocking Script
vinにあるscriptSigがtxidで指定しているトランザクションのUnlocking Script

Locking, Unlocking ScriptはBitcoin Scriptで書かれている.

Locking script と Unlocking scriptの関係はBitcoin Scriptで以下がTrueになればよい

Inputトランザクションでは参照しているUTXOのLocking Scriptに対してのUnlocking Scriptを書く

Bitcoin Script

- 例えば，以下の例を見てみる．

```
2 3 OP_ADD 5 OP_EQUAL
```

結果はTrueとなる．

Locking Scriptが以下だったとする

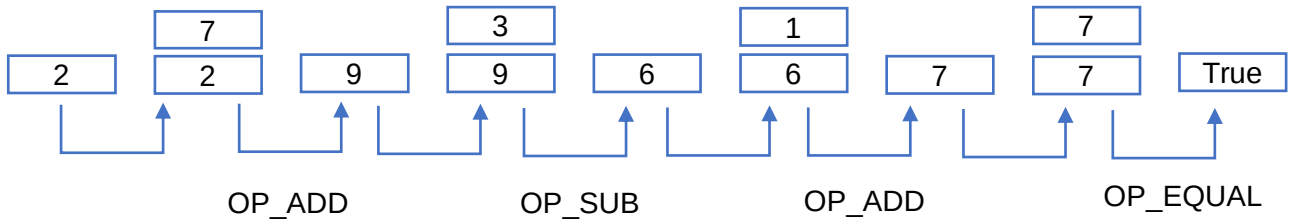
```
7 OP_ADD 3 OP_SUB 1 OP_ADD 7 OP_EQUAL
```

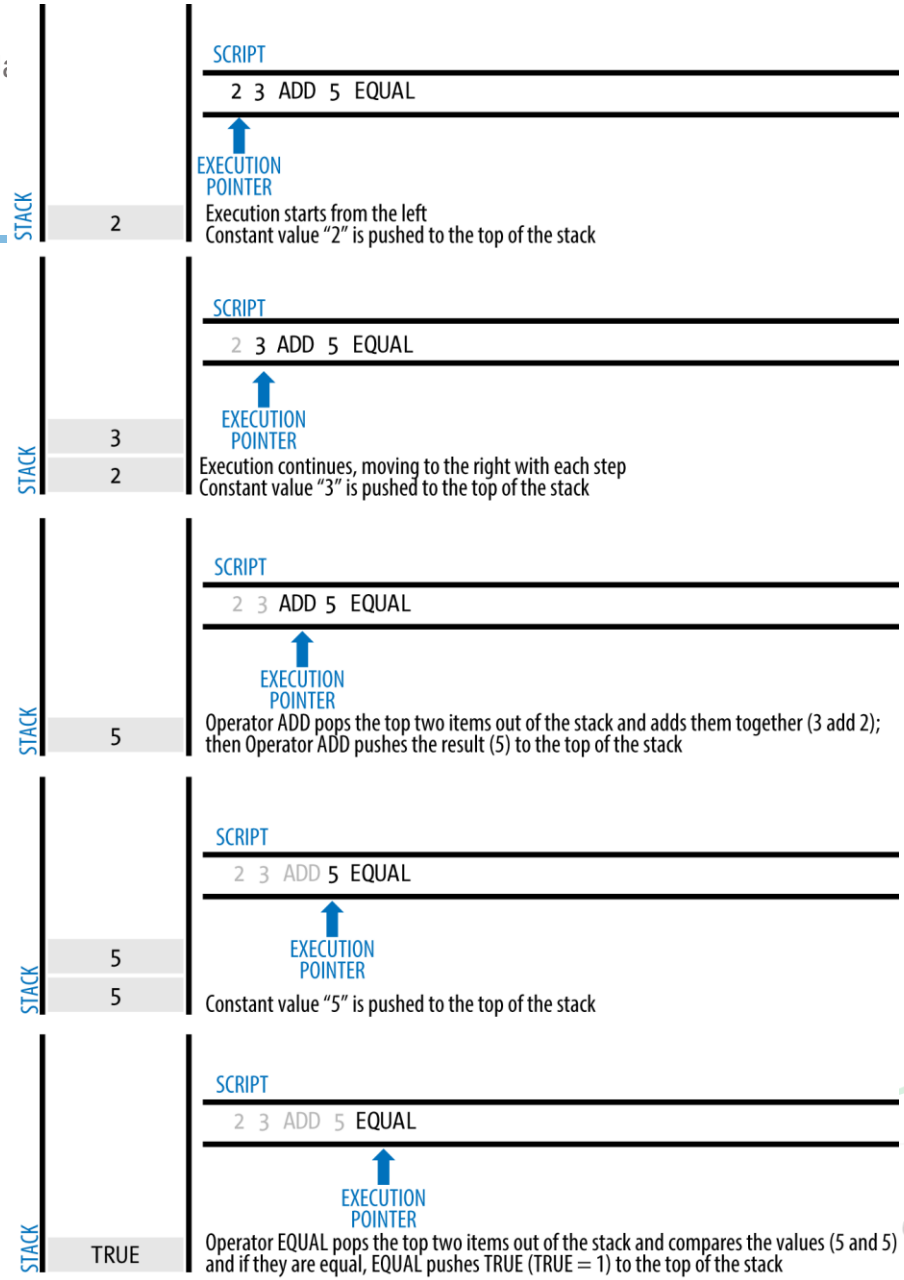
この場合，Unlocking Scriptは次

```
2
```

すなわち，以下のScriptがTrueになる

```
2 7 OP_ADD 3 OP_SUB 1 OP_ADD 7 OP_EQUAL
```





https://github.com/bitcoinbook/bitcoinbook/blob/second_edition_print3/ch06.asciidoc
Figure 4. Bitcoin's script validation doing simple math

標準的なトランザクション

- 5つの標準的なトランザクションがある.
  - Locking/Unlocking scriptの種類が5種類

- Pay-to-Public-Key-Hash (P2PKH)
  - 標準的な送金トランザクション
- Pay-to-Public-Key
  - Coinbaseトランザクション
- データアウトプット (OP_RETURN)
  - 支払いとは関係ないトランザクション用
- Multi-Signature
  - 複数人の署名が必要なトランザクション
- Pay-to-Script-Hash (P2SH)
  - より複雑な処理が可能に

Pay-to-Public-Key-Hash -- Locking ScriptとUnlocking Script

Unlocking script　Locking script

がTrueになるように，Inputトランザクションではすでに書かれているLocking Scriptに対してUnlocking Scriptを書く

Aliceがカフェで0.015BTC支払う場合，そしてカフェがそれをUnlockすることを考えてみる．

Locking Script：（Aliceが作成）

OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG

Public Key Hash (160bit)は，アドレスをBase58でデコードして抽出

Unlocking Script：（カフェが作成）

<Cafe Signature> <Cafe Public Key>

つなぎあわせた以下のScriptがTrueになればOK

<Cafe Signature> <Cafe Public Key> OP_DUP OP_HASH160 <Cafe Public Key Hash> OP_EQUALVERIFY OP_CHECKSIG

Pay-to-Public-Key-Hash -- Locking ScriptとUnlocking Script

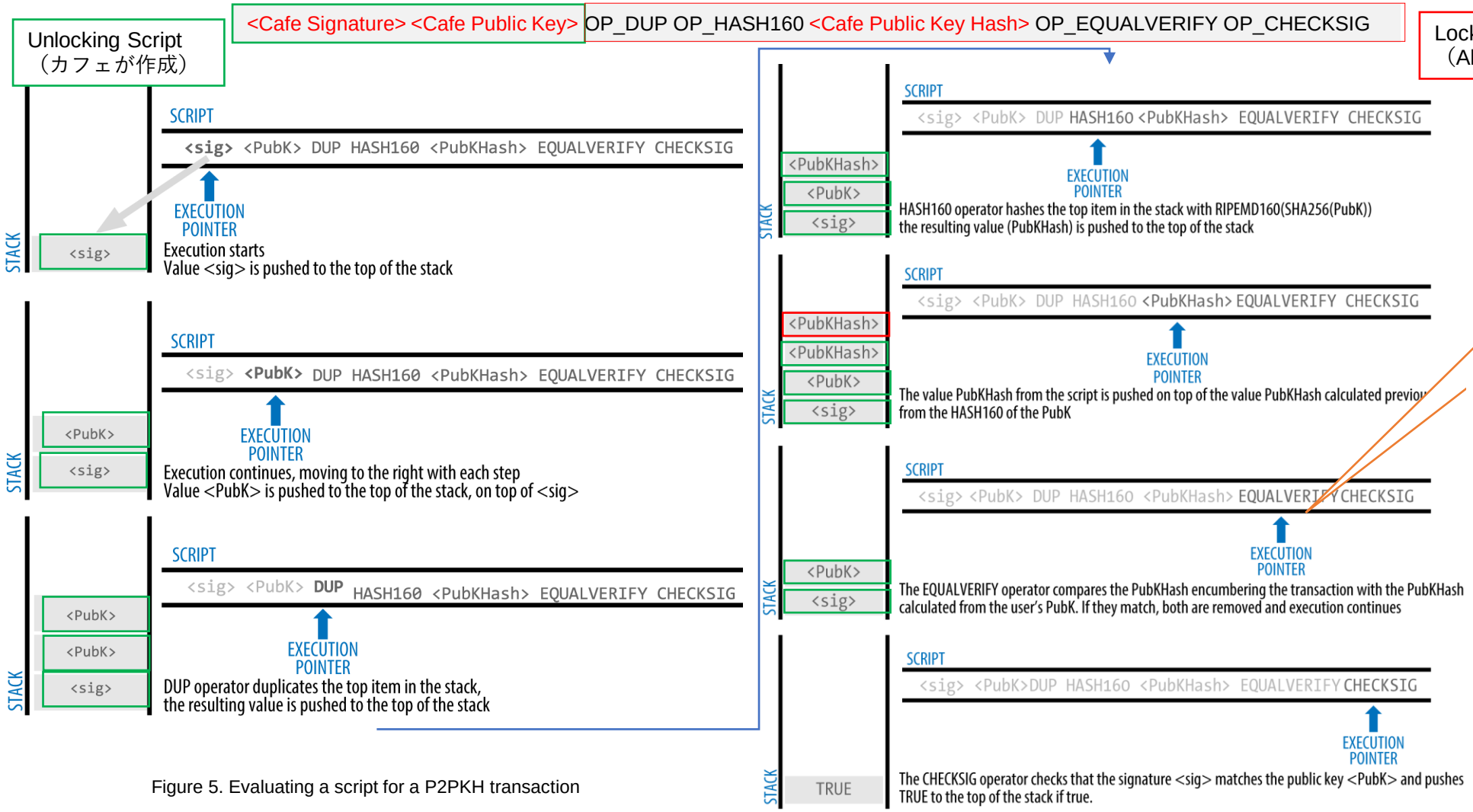


Figure 5. Evaluating a script for a P2PKH transaction

OP_EQUALVERIFY
→ 2つの値が等しかったら処理を続行.
異なっていたら異常として終了.

公開鍵から計算されるHASH160がLocking Scriptに含まれるHASH160（アドレスから抽出）と一致しているか
→ アドレスと対応する公開鍵を署名検証で用いていることを証明.

Unlock Scriptに含まれる署名と公開鍵の検証
→ 公開鍵に対応する秘密鍵の持ち主であることを証明.

Pay-to-Public-Key-Hash -- CHECKSIGでやっていること

- 署名とそれを検証するってどういうことか？
- ECDSAでの署名検証.
  - 公開鍵を用いて，対象メッセージに対する署名が，公開鍵に紐づく秘密鍵でなされていることを検証できる.
  - 参考：https://ja.wikipedia.org/wiki/%E6%A5%95%E5%86%86%E6%9B%B2%E7%B7%9ADSA
- 署名の対象は，そのトランザクションに含まれるInput, Outputを含むデータ（SIGHASH_ALL）
  - 送金先や金額を他の人が変更できないように.
  - 参考：https://en.bitcoin.it/wiki/OP_CHECKSIG

“scriptSig” :

“3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813[ALL]

署名

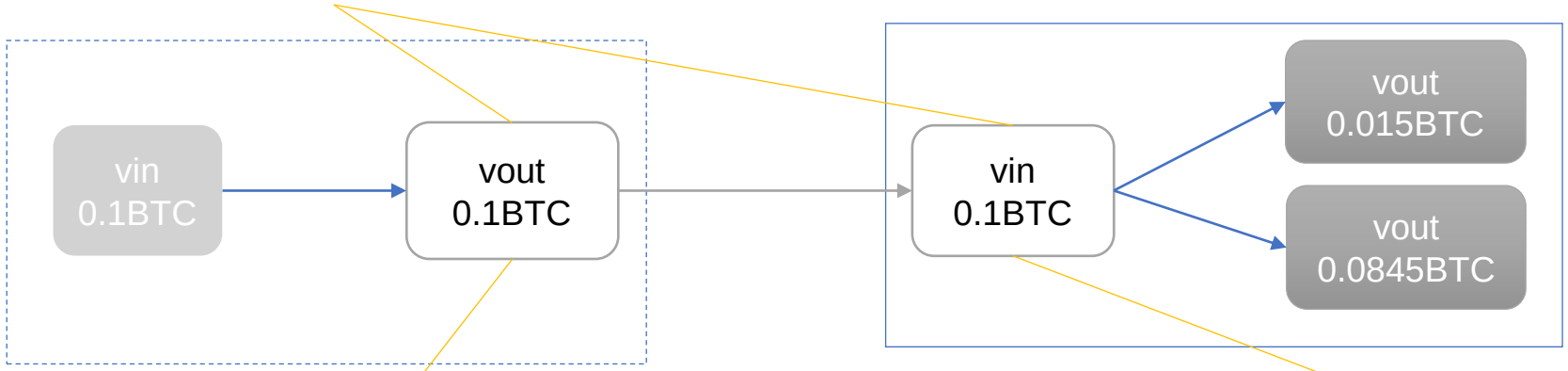
0484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf",

公開鍵

（詳細は割愛します）
要するに，秘密鍵を用いて署名を生成でき，その署名を公開鍵を用いて検証できる.

Pay-to-Public-Key-Hashまとめ

送金トランザクション
アドレス：1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK



Output script (Locking script)

OP_DUP OP_HASH160
7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8
OP_EQUALVERIFY OP_CHECKSIG

青：公開鍵ハッシュ＝アドレスの一部

※ 公開鍵ハッシュはアドレス中に埋め込まれているため，送金先アドレスを知っていればロッキングスクリプトの作成ができる．

```
$ bx base58check-decode 1Cdid9KFAaatwczBwBttQcwXYCpvK8h7FK
wrapper
{
    checksum 3326777089
    payload 7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8
    version 0
}
```

Input script (Unlocking script)

4830450221 00884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb 0220 4b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e381301410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf

緑：署名
青：公開鍵

https://btc.com/7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18
https://btc.com/0627052b6f28912f2703066a912ea577f2ce4da4caa5a5fbd8a57286c345c2f2

Pay-to-Public-Key-Hashまとめ

Output script (Locking script)

OP_DUP OP_HASH160
7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc0
25a8 OP_EQUALVERIFY
OP_CHECKSIG

青：公開鍵ハッシュ＝アドレスの一部

Input script (Unlocking script)

4830450221 00884d142d86652a3f47ba4746ec719bbfbd040
a570b1deccbb6498c75c4ae24cb 0220 4b9f039ff08df09cbe9f
6addac960298cad530a863ea8f53982c09db8f6e3813 0141 0
484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade
8416ab9fe423cc5412336376789d172787ec3457eee41c04f
4938de5cc17b4a10fa336a8d752adf

緑：署名
青：公開鍵

4830450221 00884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb 0220 4b
9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813 0141 0484ecc0d46f1918b30
928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f493
8de5cc17b4a10fa336a8d752adf

OP_DUP OP_HASH160
7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8
OP_EQUALVERIFY OP_CHECKSIG

が実行される.

やっていることをまとめると
- Inputスクリプトにある公開鍵がOutputスクリプトのものと同じかどうかがチェックされる
- その公開鍵をもって，署名の検証が可能か
  → 秘密鍵をもってそのTxを作ったかどうかのチェックにつながる

Pay-to-Public-Key

- P2PKHよりシンプルな送金形式
- ロッキングスクリプト内には公開鍵ハッシュではなく公開鍵そのものを使用
  - 送金する側が，送金先アドレスの公開鍵も必要になる
  - アドレスだけでは作れない
- coinbaseトランザクションで使用される.
  - ブロック内に存在するマイナーへのマイニング報酬のトランザクション

Unlocking script　Locking script

ロッキングスクリプト：

<Publick Key A> OP_CHECKSIG

アンロッキングスクリプト：

<Signature from Private Key A>

検証用の結合されたスクリプト：

<Signature from Private Key A>　<Publick Key A> OP_CHECKSIG

データアウトプット（OP_RETURN）

- ビットコインはタイムスタンプ付きの分散台帳
  - タイムスタンプはブロックに含まれます
- 支払い以外の情報の記録でビットコインブロックチェーンを使用したい.
  - ファイルの存在証明など

- ビットコインの開発者の中で意見が分かれる
  - 賛成派：ビットコインの支払い以外にも応用の幅を広げるべき
  - 反対派：そんなことするとブロックチェーンが肥大化してしまう
    - 支払いとは関係ないトランザクションが増える
    - UTXOが増えてしまう
      - UTXOは別DBで管理

- 妥協案としてOP_RETURNコードが導入された

```
OP_RETURN <data>
```

- このdataは80バイト
  - 多くの使用例では，何かのハッシュ値のみ保存する
- OP_RETURNを使用するためのアンロッキングスクリプトは存在しない
  - UTXOにカウントされない

- 本スライドの著作権は，東京大学ブロックチェーンイノベーション寄付講座に帰属しています．自己の学習用途以外の使用，無断転載・改変等は禁止します．
- ただし，
  - Mastering Bitcoin 2nd edition https://github.com/bitcoinbook/bitcoinbook/tree/second_edition_print3
  - ビットコインとブロックチェーン --- 暗号通貨を支える技術 （著：Andreas M. Antonopoulos 訳： 今井崇也，鳩貝 淳一郎）
- を使用した部分の使用のみ，CC BY-SA 4.0 https://creativecommons.org/licenses/by-sa/4.0/deed.ja のライセンスを適用するものとします．