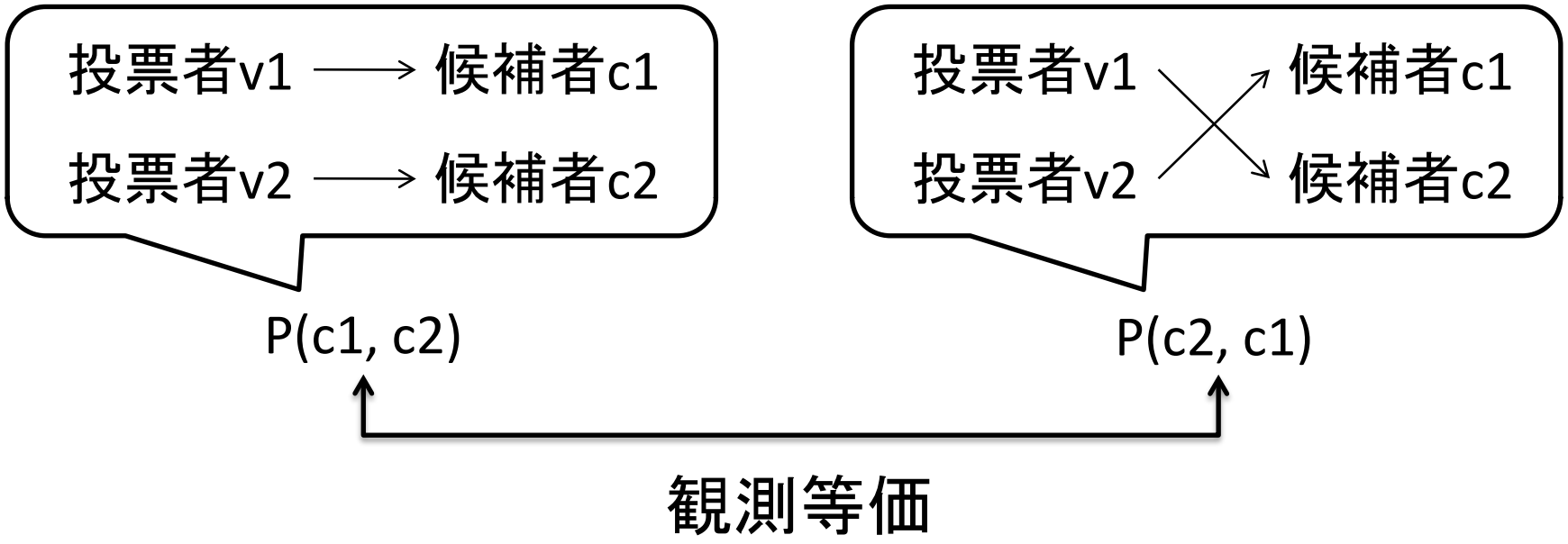
投票プロトコルの匿名性の自動検証

櫻田英樹
日本電信電話株式会社
NTT コミュニケーション科学基礎研究所

投票プロトコルの匿名性

- 同じ開票結果を持つ2つの投票の組み合わせを攻撃者が区別できない（観測等価である）。
- 観測等価性の自動検証は一般には困難。



関連研究

- 投票プロトコル自動検証（Kremer, Ryan ’05）。

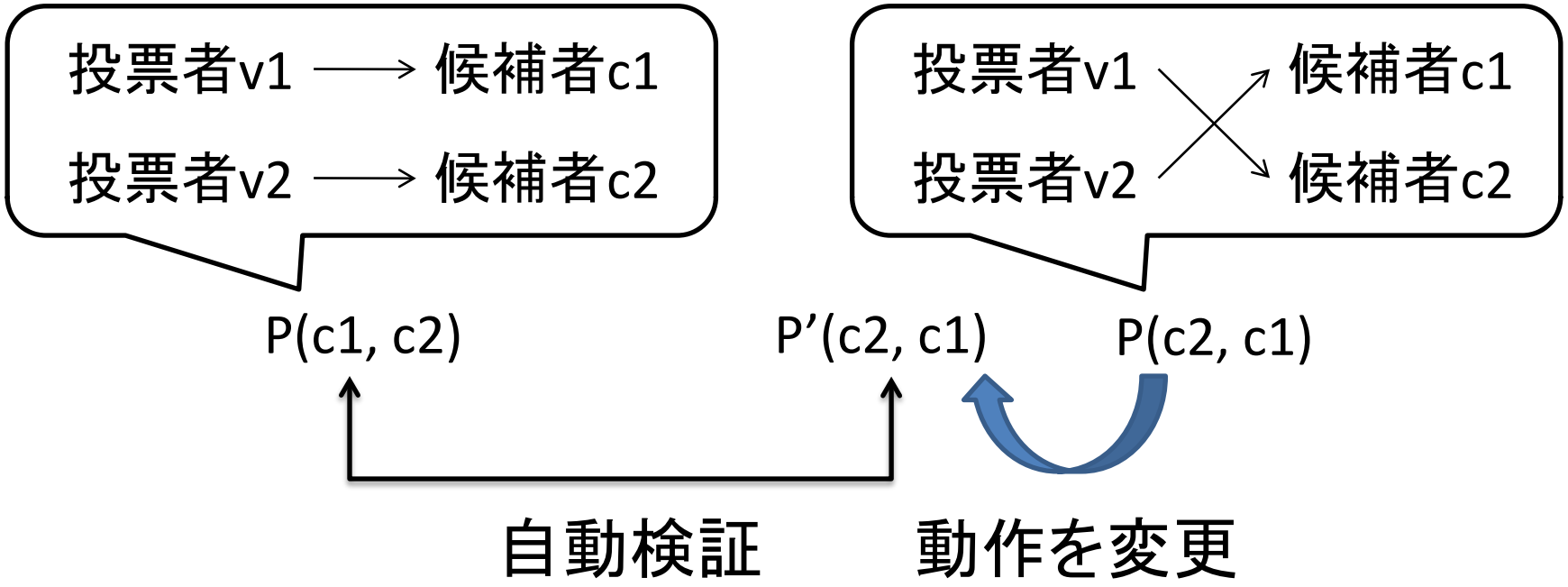
  FOO（藤岡ら, ’92）の様々な性質を自動検証ツールProverifを用いて検証。ただし、匿名性は手作業で安全性証明を記述して検証。
- Proverif (Blanchet, ’01～)

  自動検証ツール。認証プロトコルの安全性など様々な性質の検証に用いられる。制限された形の観測等価性を検証できる。

⇒Proverifを利用して匿名性を自動検証したい

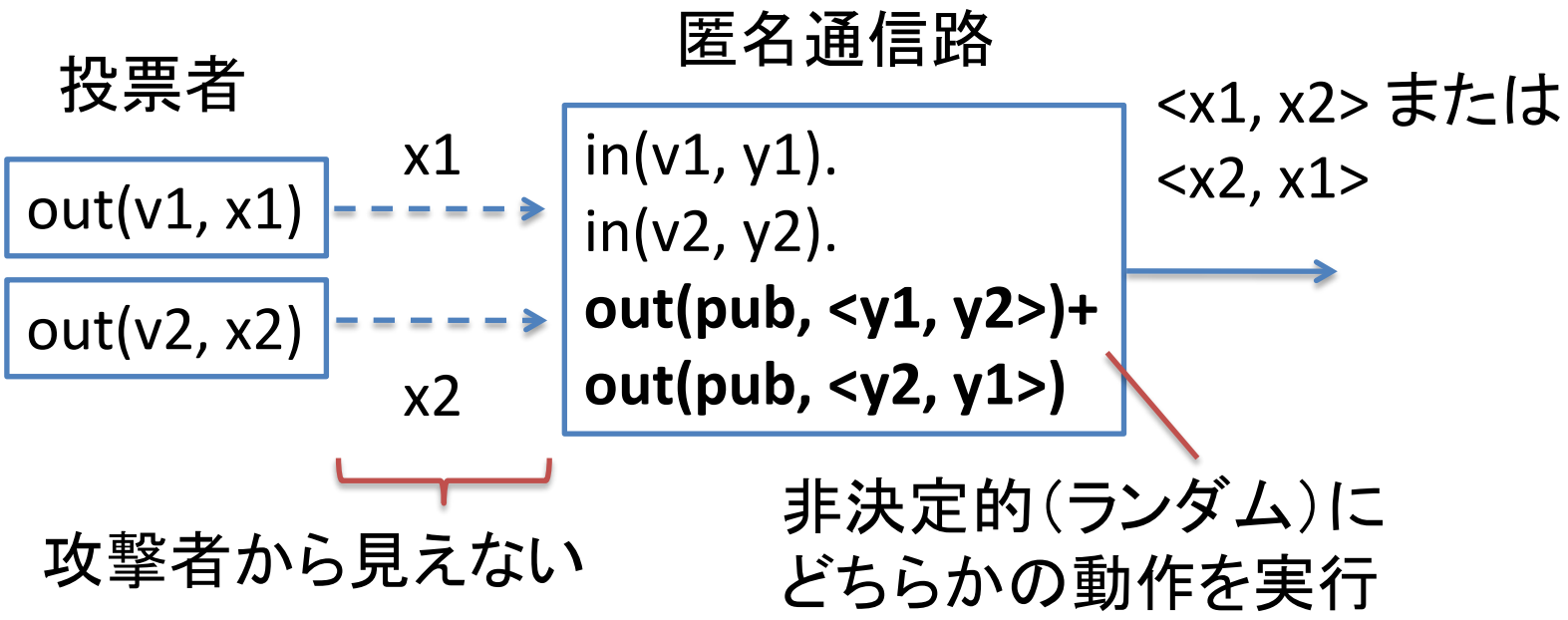
本研究のアプローチ

- プロトコルの一部（匿名通信路）を変更し、Proverifを利用して自動検証。
- 変更前後の観測等価性は自明。

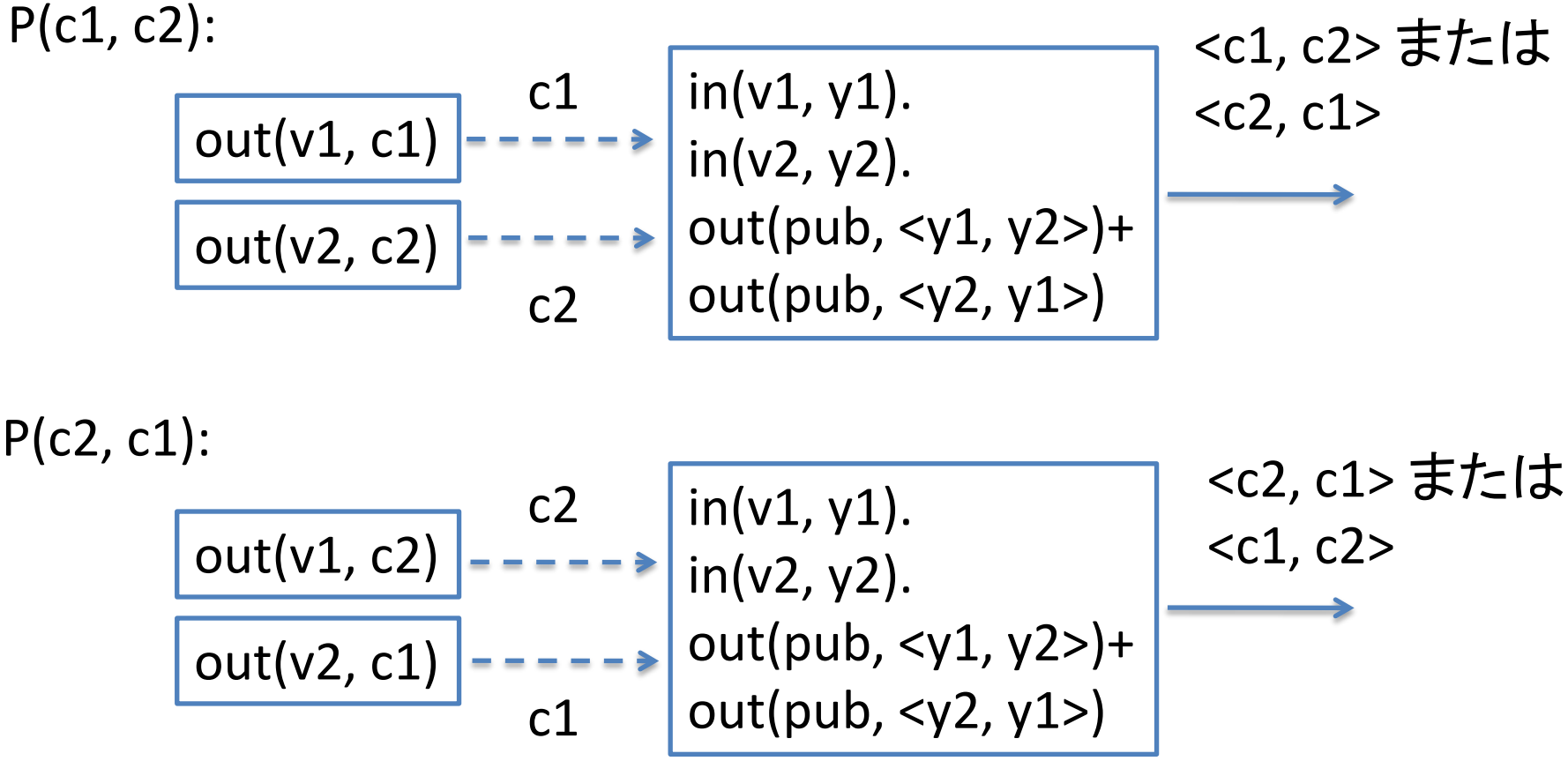


簡単な投票プロトコルの例

P(x1, x2):



2つの投票の組み合わせ



攻撃者が<c1, c2>あるいは<c2, c1>を見ても、
どちらの組み合わせかわからない（識別不能）

Proverifの観測等価性検証方法

- 観測等価性よりも強い等価性を検査:

  非決定的分岐Q+Rで、`+'の左右どちらが実行されるか攻撃者が選ぶことができるとする。

→2つの実行PとP'では同じ分岐が実行される。

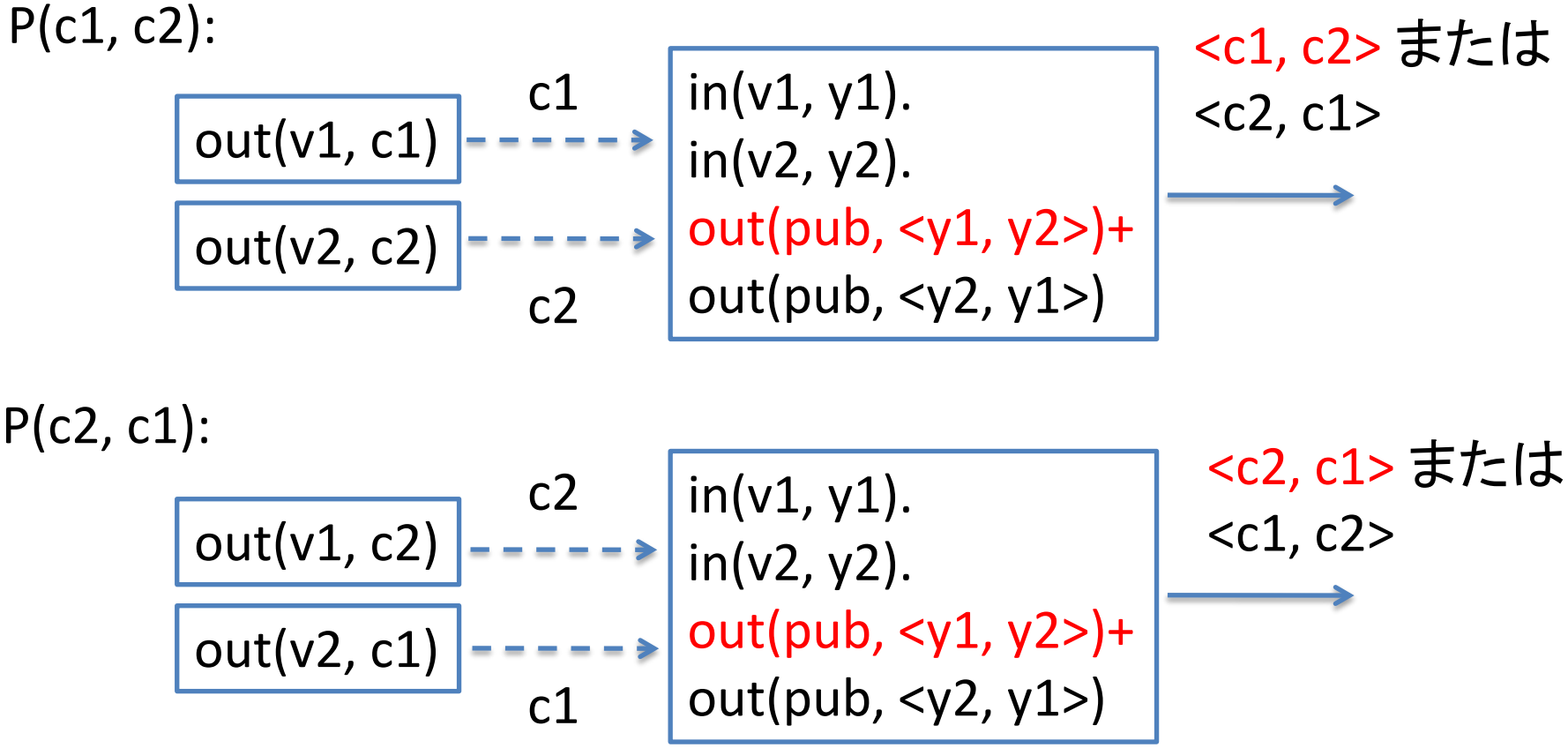
- 実際には、2つの実行を「同時に実行」するプロセスを入力、攻撃者に観測させる。

  例: P(c1, c2)とP(c2, c1)については

  P(diff[c1,c2], diff[c2, c1])

投票プロトコルを検証してみる

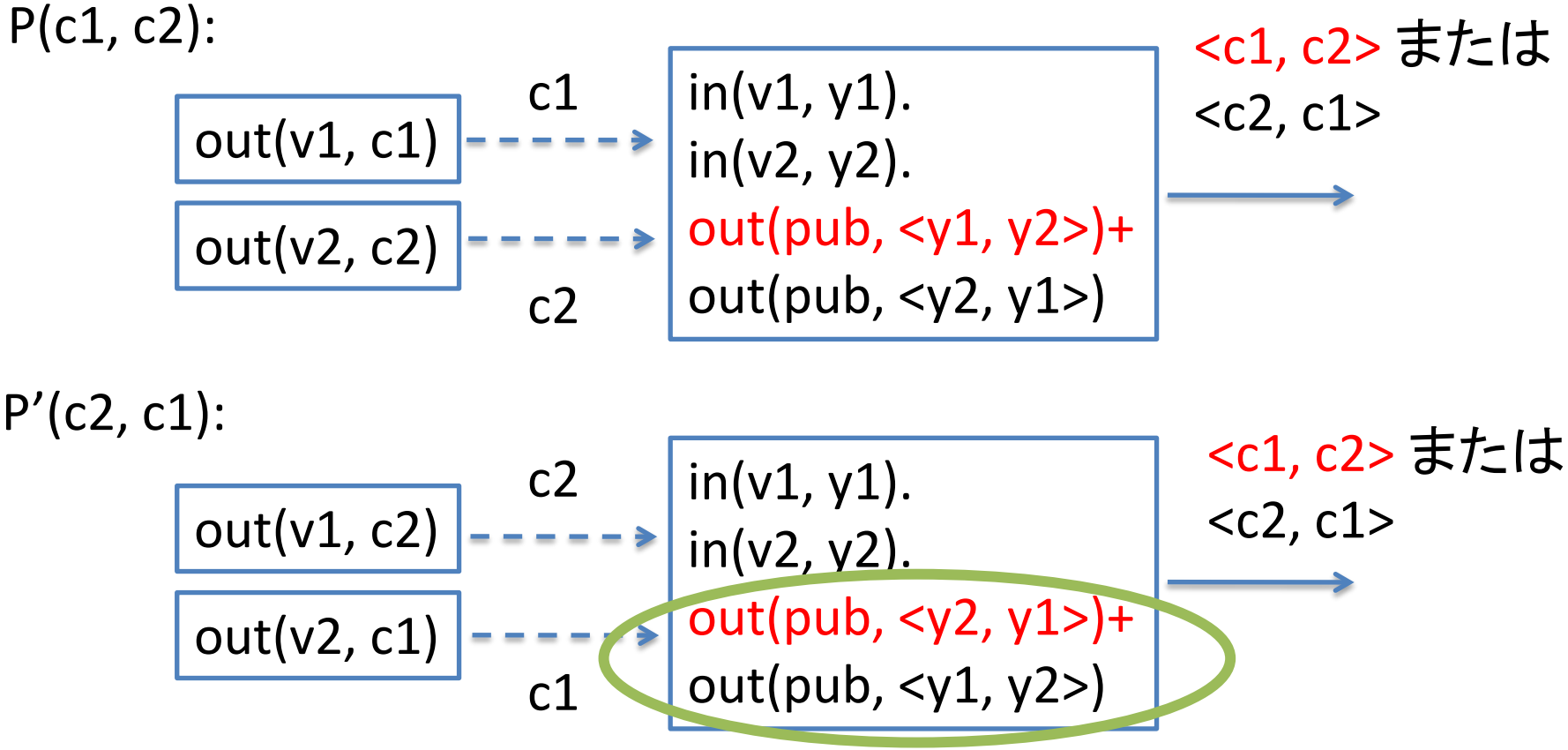
同じ分岐が選ばれると、攻撃者に識別されてしまい、検証失敗。



匿名通信路の動作を変更

２つめの実行の分岐の順序を入れ替える

⇒同じ分岐を選んでも識別できず、検証成功



Proverifの入力・検証

Proverifの入力（2つの実行を一つにしたもの）:

out(v1, diff[c1, c2])

out(v2, diff[c2, c1])

in(v1, y1).
in(v2, y2).
out(pub, <diff[y1, y2], diff[y2, y1]>)+
out(pub, <diff[y2, y1], diff[y1, y2]>)

投票者が投票を送信

out(pub, <diff[diff[c1, c2], diff[c2, c1]], diff[diff[c2, c1], diff[c1, c2]]>)+
out(pub, <diff[diff[c2, c1], diff[c1, c2]], diff[diff[c1, c2], diff[c2, c1]]>)

=

diffが消えるため、同じように分岐しても識別不能

out(pub, <diff[c1, c1], diff[c2, c2]>)+
out(pub, <diff[c2, c2], diff[c1, c1]>)

=

out(pub, <c1, c2>)+
out(pub, <c2, c1>)

変更前後の観測等価性

非決定的動作の分岐の順序を変えただけなので、自明に観測等価性が成り立つ

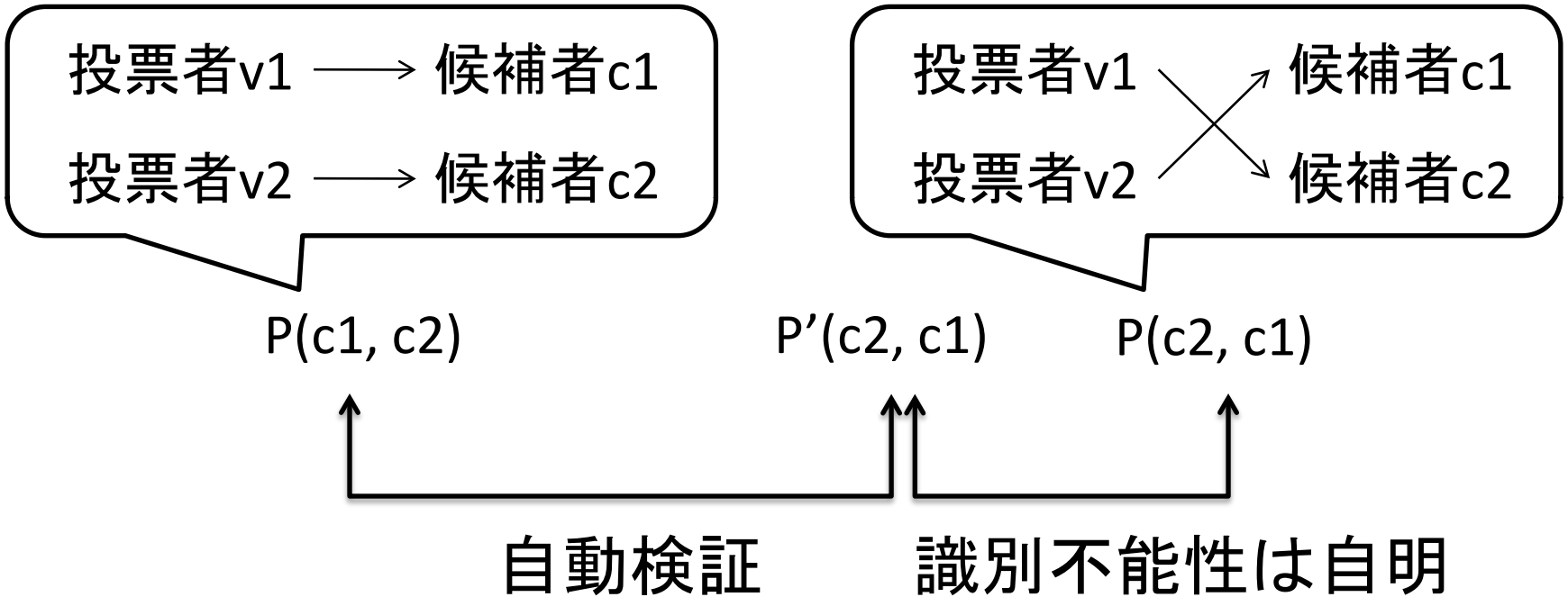
変更前の匿名通信路:

in(v1, y1).
in(v2, y2).
out(pub, <y1, y2>)+
out(pub, <y2, y1>)

変更前の匿名通信路:

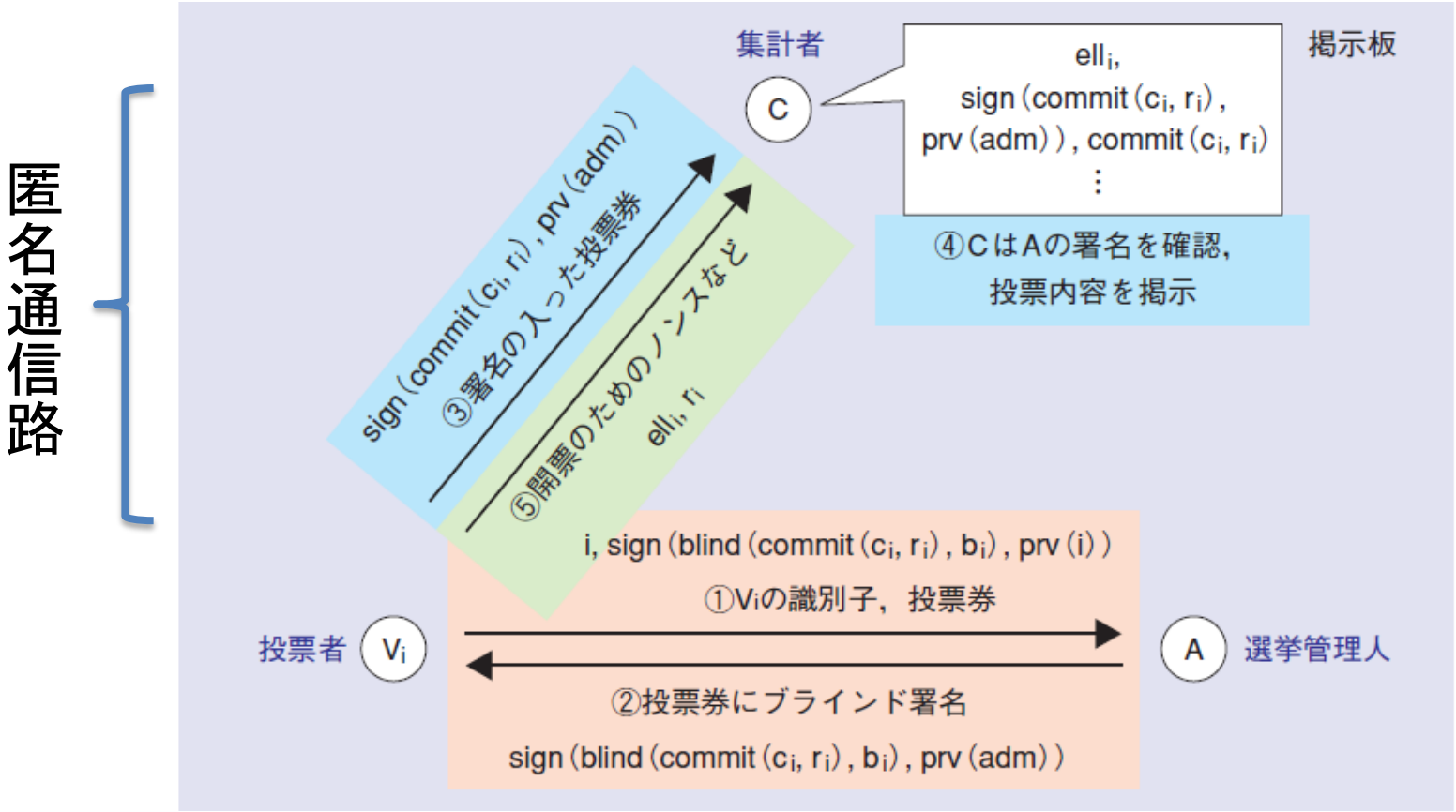
in(v1, y1).
in(v2, y2).
out(pub, <y2, y1>)+
out(pub, <y1, y2>)

全体として安全性検証が完成



FOOの匿名性検証

Proverifを利用して、同様の方法で検証可能



（河辺ら「匿名性とプライバシ保護の数理的技法」, NTT技術ジャーナル2007.6）

まとめ

- プロトコルの匿名性の自動検証方法を提案
  - プロトコルの一部（匿名通信路）を変更し、Proverifを利用して自動検証。
  - 変更前後の観測等価性は自明。
- 匿名投票プロトコルFOOの匿名性検証に適用