

量子暗号のための プロトコル等価性検証ツール

久保田 貴大^{*}, 角谷 良彦^{*},
加藤 豪[†], 河野 泰人[†], 櫻田 英樹[†]

^{*}東京大学情報理工学系研究科,

[†]NTTコミュニケーション科学基礎研究所

背景

- 暗号安全性証明の検証は難しい
- 量子暗号でもそうである
- 検証のための形式体系が提案されているが、実際には、形式体系の適用は手作業では非常に煩雑である
- 形式検証のためには、検証ツールが開発されることが望ましい

研究の概要

- 量子プロセス計算qCCS [FDY11] における双模倣関係を自動検証するツールを開発した
- ShorとPreskillによるBB84の安全性証明[SP00] にツールを適用した

本発表の構成

- 量子プロセス計算qCCS
- Shor-PreskillによるBB84の安全性証明
- 検証ツール
- 実験結果
- まとめと今後の課題

量子ビットと密度行列

- 量子ビットの状態は, \mathbb{C}^2 (二次元ヒルベルト空間) の単位ベクトルとして表される
 - $|0\rangle, |1\rangle$ は標準基底, 縦ベクトル
 - $\alpha|0\rangle + \beta|1\rangle$ ($|\alpha|^2 + |\beta|^2 = 1$)
- 量子ビット列の状態は, 量子ビットの状態のテンソル積として表される
 - $|0110\rangle \equiv |0\rangle \otimes |1\rangle \otimes |1\rangle \otimes |0\rangle$
- 量子ビット列の状態が, 確率 p_i で $|\psi_i\rangle$ であるという確率分布は, 密度行列 $\sum_i p_i |\psi_i\rangle\langle\psi_i|$ で表される
 - $1/2(|0\rangle\langle 0| + |1\rangle\langle 1|)$

qCCSの文法[FDY11]

量子通信

$$P ::= \text{nil} \mid \tau.P \mid c?x.P \mid c!e.P \mid \mathbf{c}?q.P \mid \mathbf{c}!q.P \\ \mid \text{if } b \text{ then } P \mid \mathcal{E}[\tilde{q}].P \mid M[\tilde{q}; x].P \mid P||P \mid P \setminus L$$

量子演算

観測

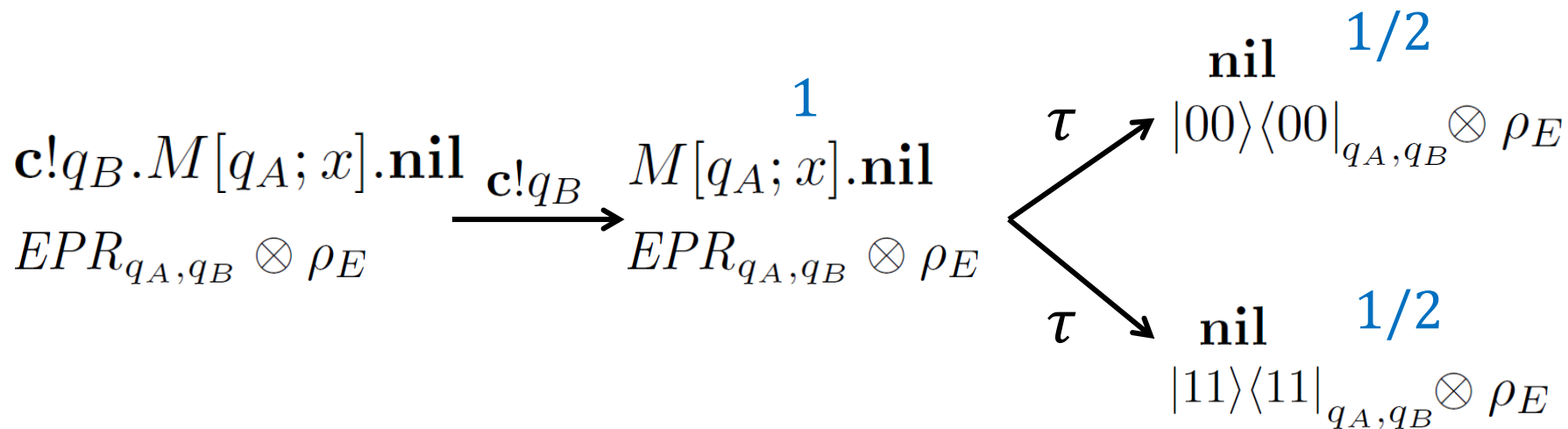
- ひとつのqubit型自由変数には, 2次元ヒルベルト空間が対応する
- 各qubitの状態をあらわす密度行列を ρ とする
以後, 組 $\langle P, \rho \rangle$ のことをコンフィグレーションとよぶ
- $P||Q$ は, $qv(P) \cap qv(Q) = \emptyset$ のときのみ定義される
- 量子通信で空間をやりとりできる

ラベル付き状態遷移

- $\langle P, \rho \rangle \xrightarrow{\alpha} \mu$
- コンフィギュレーション $\langle P, \rho \rangle$ は
行動 α をおこなって,
確率分布 μ に遷移する

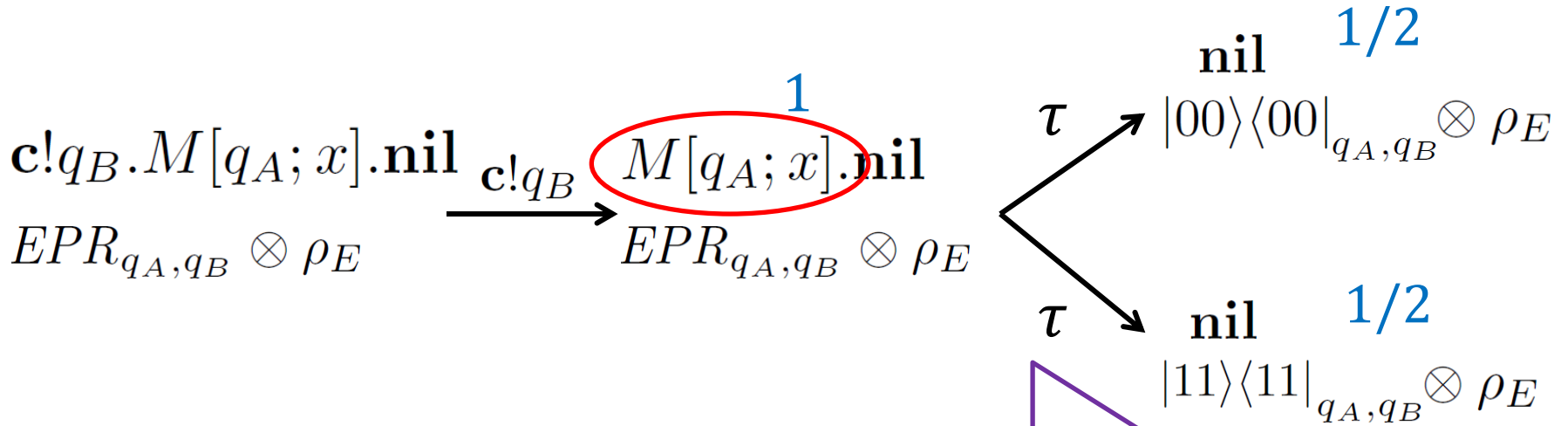
状態遷移の例

$$\langle \mathbf{c!}q_B.M[q_A; x].\mathbf{nil}, EPR_{q_A, q_B} \otimes \rho_E \rangle$$



状態遷移の例

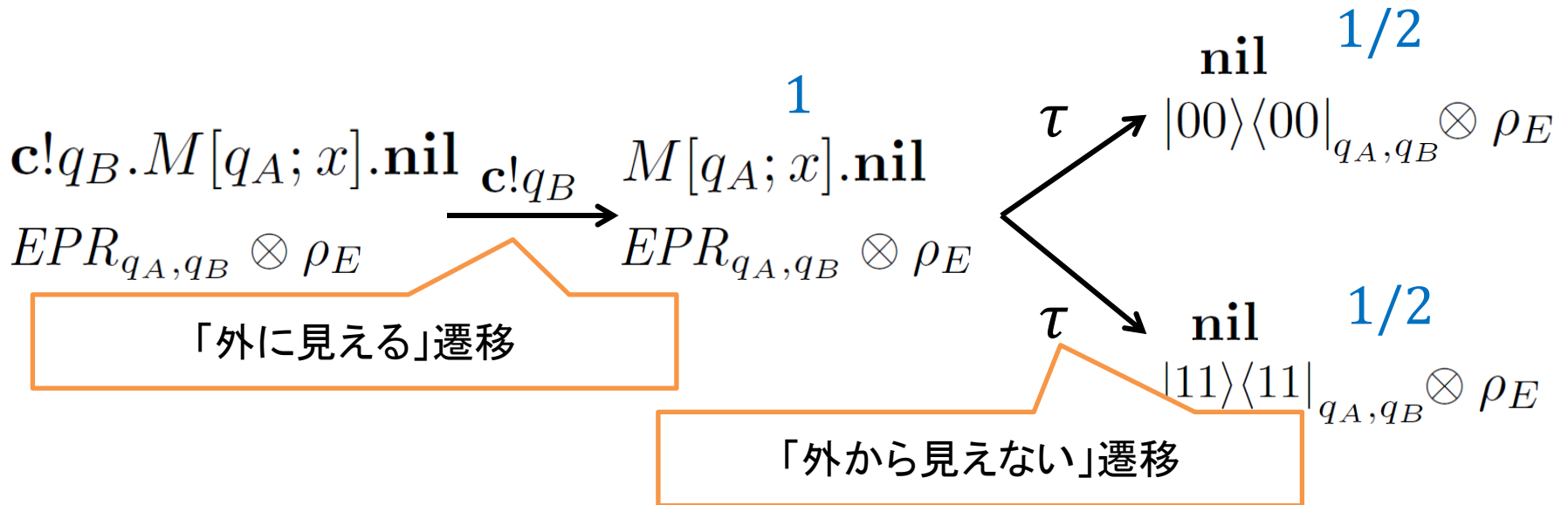
$$\langle \mathbf{c!}q_B.M[q_A; x].\mathbf{nil}, EPR_{q_A, q_B} \otimes \rho_E \rangle$$



観測したときだけ確率分岐

状態遷移の例

$$\langle \mathbf{c!}q_B.M[q_A; x].\mathbf{nil}, EPR_{q_A, q_B} \otimes \rho_E \rangle$$

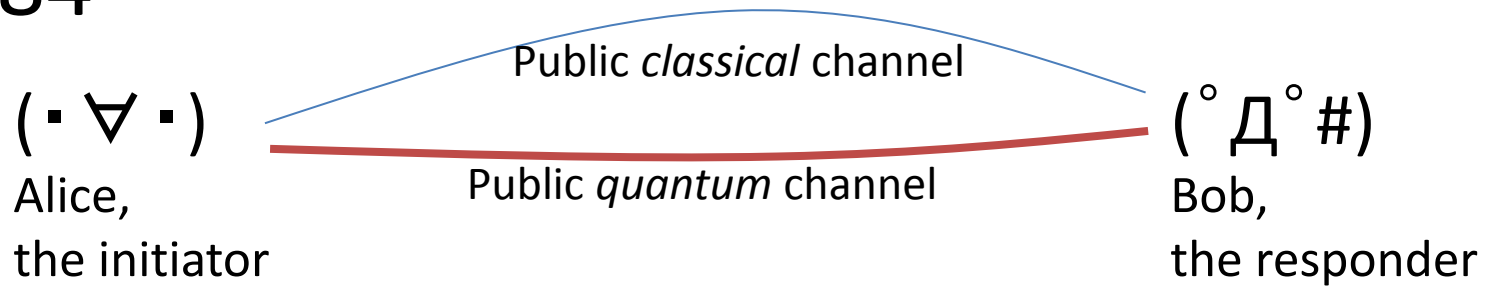


双模倣関係

- ふたつのコンフィグレーション $\langle P, \rho \rangle, \langle Q, \sigma \rangle$ が
外から見て同じように振る舞うという関係
- 一方のプロセスが遷移可能なら, 他方でも
 τ 遷移を除いて同様の遷移が可能であり,
かつ, 各ステップにおいて,
外の人アクセスできる量子状態が同じ
- $\langle P, \rho \rangle \approx \langle Q, \sigma \rangle$ と書く

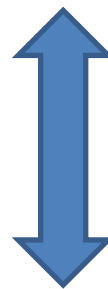
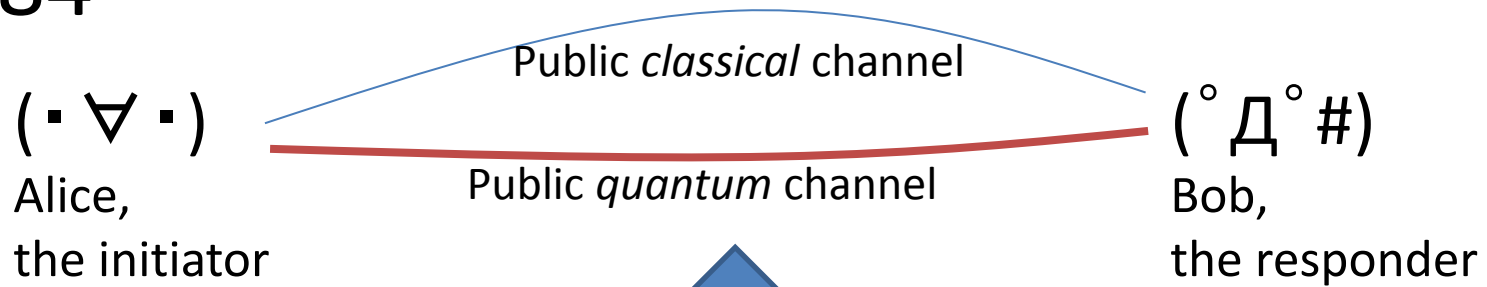
BB84の安全性証明の概要[SP00]

BB84



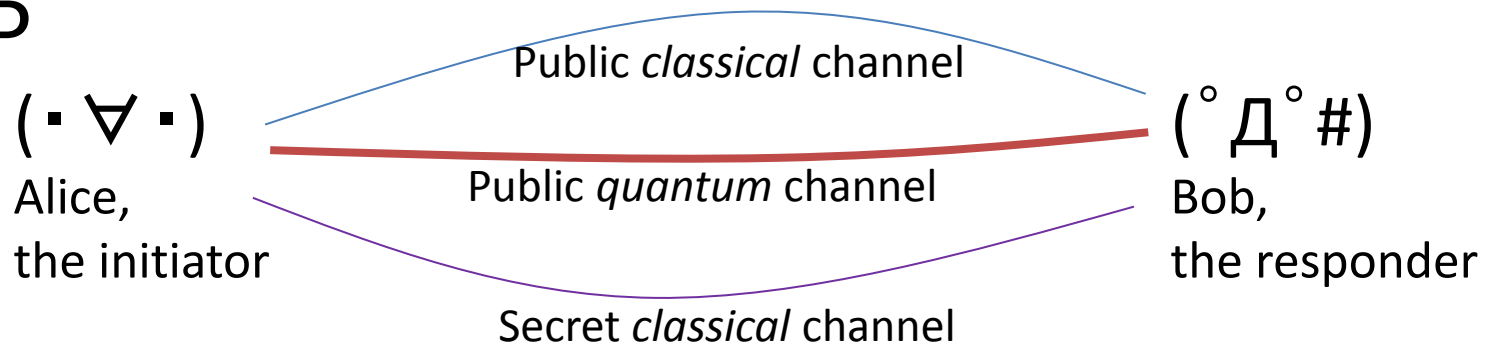
BB84の安全性証明の概要[SP00]

BB84



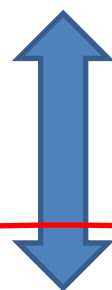
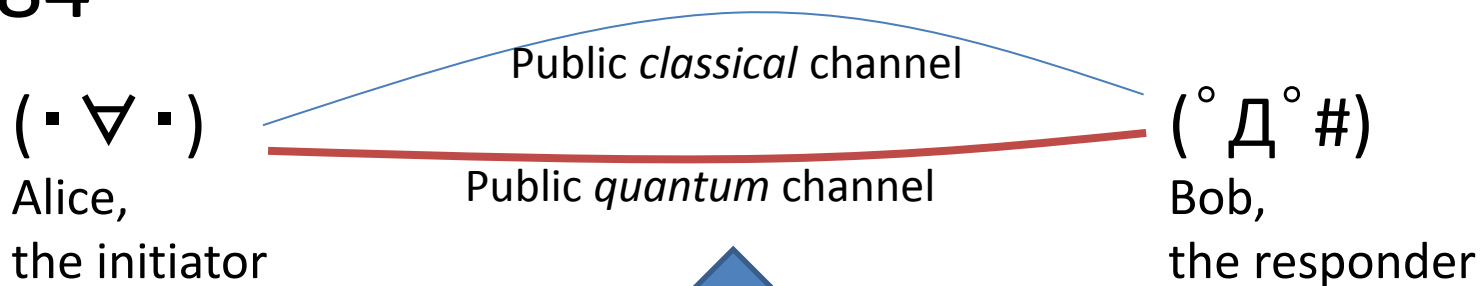
外から見て区別できないことを示す

EDP



BB84の安全性証明の概要[SP00]

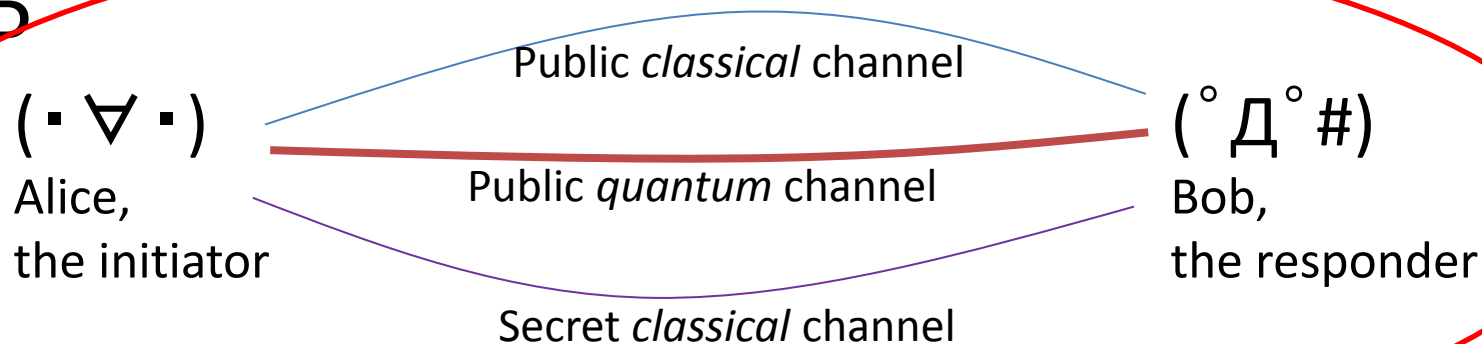
BB84



外から見て区別できないことを示す

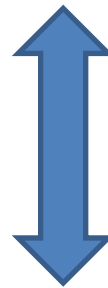
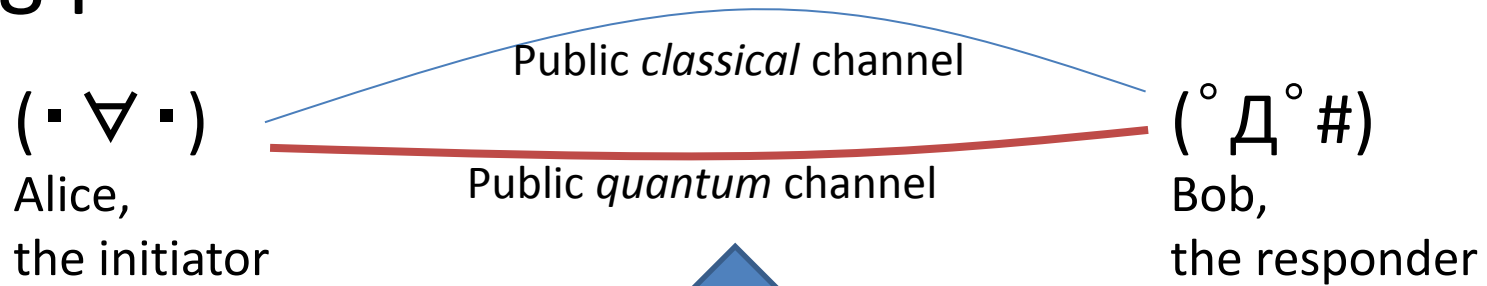
安全であることを示す

EDP



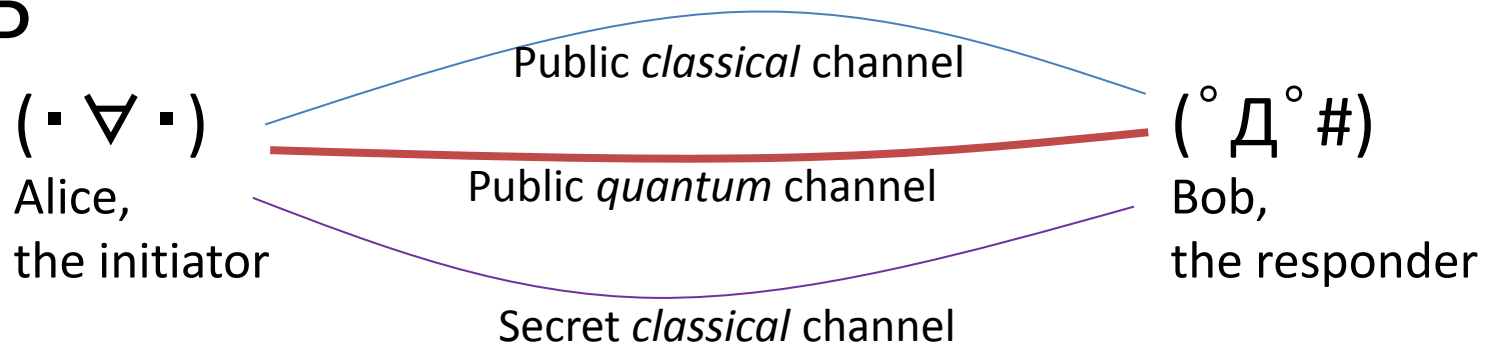
形式検証の概要

BB84



外から見て区別できないことを示す

EDP



形式検証の概要

BB84

コンフィグレーション
として形式化

the initiator

```

(A|B)\{c1, c2, c3, c4, c5, c6\} \rho_A \otimes \rho_B \otimes \rho_E
A ≡ hadamards(q_A^1, r_A^1).
  shuffle(q_A^2, r_A^2).
  c1!q_A^1, ..., q_A^2n, c1?x_A.
  copy(r_A^2, R_A^2). c2!r_A^2, d1!R_A^2.
  copy(r_A^1, R_A^1). c3!r_A^1, d2!R_A^1.
  measure(q_A, ..., q_A, n).
  c4?s_A.abort_alice(q_A, ..., q_A, n, s_A, b_A)
M|b_A; y. c2!y, d1!y. if y then
  css_projection(q_A, n+1, ..., q_A, 2n, u_A, v_A).
  css_decode(q_A, n+1, ..., q_A, 2n, u_A, v_A)
copy(u_A, U_A). c5!u_A, d3!U_A.
copy(v_A, V_A). c6!v_A, A2
+ if -y then ⊥

ρ_A ≡ (|00⟩⟨00| + |00⟩⟨11| + |11⟩⟨00| + |11⟩⟨11|)_{q_A, r_A}^{⊗ 2n} ⊗
(|0⟩⟨0| + |1⟩⟨1|)_{R_A^1}^{⊗ N} ⊗ (|0⟩⟨0| + |1⟩⟨1|)_{R_A^2}^{⊗ N} ⊗
(|0⟩⟨0|)_{R_A^1}^{⊗ N} ⊗ (|0⟩⟨0|)_{R_A^2}^{⊗ N} ⊗ |0⟩⟨0|_{b_A} ⊗
(|0⟩⟨0|)_{U_A}^{⊗ n} ⊗ (|0⟩⟨0|)_{V_A}^{⊗ n} ⊗ (|0⟩⟨0|)_{U_A}^{⊗ n} ⊗ (|0⟩⟨0|)_{V_A}^{⊗ n}
  
```

annel

(° Δ ° #)

Bob,
the responder

双模倣の証明



EDP

(· ∇ ·)

Alice,
the initiator

```

(A|B)\{c1, c2, c3, c4, c5\} \rho_A \otimes \rho_B \otimes \rho_E
A ≡ hadamards(q_A^1, r_A^1).
  shuffle(q_A^2, r_A^2).
  c1!q_A^1, ..., q_A^2n, c1?x_A.
  copy(r_A^2, R_A^2). c2!r_A^2, d1!R_A^2.
  copy(r_A^1, R_A^1). c3!r_A^1, d2!R_A^1.
  c4?s_A.abort_alice(q_A, ..., q_A, n, s_A, b_A)
M|b_A; y. c2!y, d1!y. if y then
  if b_A then
    cnot(u_A, q_A, n+1, ..., q_A, 2n).
    copy(u_A, q_A, n+1, ..., q_A, 2n).
    key(q_A, n+1, ..., q_A, 2n).
    copy(u_A, U_A). c5!u_A, d3!U_A, A2
  + if -y then ⊥

ρ_A ≡ (|00⟩⟨00| + |11⟩⟨11|)_{q_A, r_A}^{⊗ 2n} ⊗
(|0⟩⟨0| + |1⟩⟨1|)_{R_A^1}^{⊗ N} ⊗ (|0⟩⟨0| + |1⟩⟨1|)_{R_A^2}^{⊗ N} ⊗
(|0⟩⟨0|)_{R_A^1}^{⊗ N} ⊗ (|0⟩⟨0|)_{R_A^2}^{⊗ N} ⊗ |0⟩⟨0|_{b_A} ⊗
(∑_{u ∈ C1} |u⟩⟨u|)_{U_A}^{⊗ n} ⊗ (|0⟩⟨0|)_{V_A}^{⊗ n}
  
```

el

(° Δ ° #)

Bob,
the responder

el

これまでの研究 [KKKKS'12]

- 双模倣を手作業で示した
 - 状態遷移が長く, 分岐が多い
 - 攻撃者がアクセスできる量子状態が等しいことは, 元論文の証明と同じように示した
- 検証ツールがあるとよい
 - 遷移が対応することを自動的に検証したい
 - 量子状態の等しさも自動で計算したい

ツールの設計(量子状態の表現)

- 量子状態は記号列で表現
 - 鍵の長さなどを決めるセキュリティパラメタは、自然数の定数記号として扱う

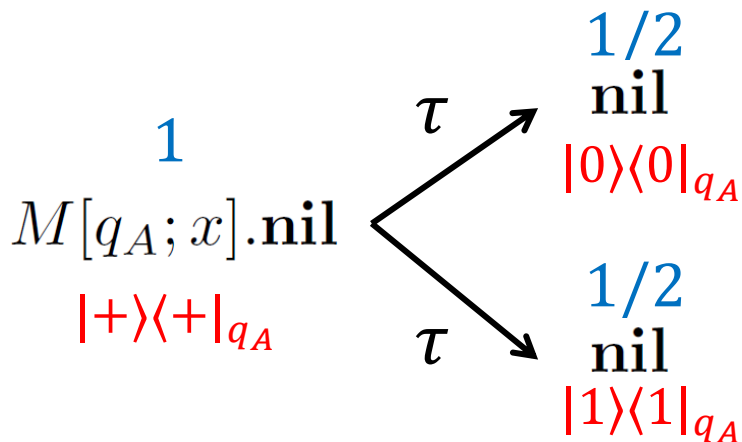
$$(|00\rangle\langle 00| + |00\rangle\langle 11| + |11\rangle\langle 00| + |11\rangle\langle 11|)_{q^A, q'^A}^{\otimes n}$$

n個のEPRペア

```
nat n;  
qvar qA : n;  
qvar qA' : n;  
dsym EPR : n, n;  
...  
EPR[qA, qA']
```

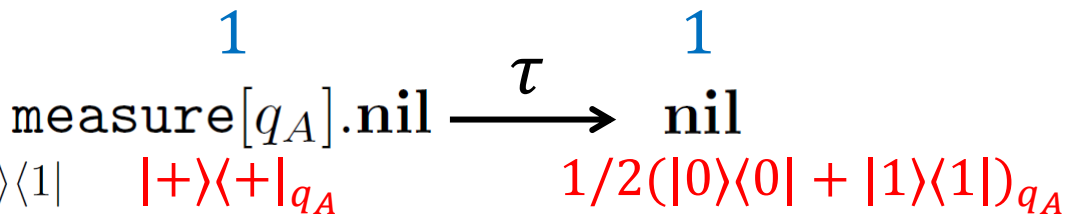
観測の形式化について

構文にある観測 $M[\tilde{q}; x]$ で
書いた場合



観測をあらわす
量子演算で書いた場合

$$\mathcal{E}_{\text{measure}}(\rho) = |0\rangle\langle 0|\rho|0\rangle\langle 0| + |1\rangle\langle 1|\rho|1\rangle\langle 1|$$



$$\begin{array}{l}
 P ::= \text{nil} \mid \tau.P \mid c?x.P \mid c!e.P \mid \mathbf{c}?q.P \mid \mathbf{c}!q.P \\
 \mid \text{if } b \text{ then } P \mid \underline{\mathcal{E}[\tilde{q}].P} \mid \underline{M[\tilde{q}; x].P} \mid P||P \mid P \setminus L \\
 \text{量子演算} \qquad \qquad \qquad \text{観測}
 \end{array}$$

観測の形式化について [KKKKS'12]

- 観測の形式化は二種類あり、使い分ける必要がある
- 攻撃者にとって分岐が見えるような観測だったら $M[\tilde{q}; x]$ で形式化する
 - 観測の結果によって違うラベルで遷移する場合など
- 見えないなら $\mathcal{E}_{\text{measure}}$ で形式化する

$$P ::= \text{nil} \mid \tau.P \mid c?x.P \mid c!e.P \mid \mathbf{c}?q.P \mid \mathbf{c}!q.P \\ \mid \text{if } b \text{ then } P \mid \underline{\mathcal{E}[\tilde{q}].P} \mid \underline{M[\tilde{q}; x].P} \mid P \parallel P \mid P \setminus L$$

量子演算 観測

ツールの設計 (確率分岐)

- 形式化の使い分けを自然に行うため,
 $M[q; x]$ は if 分岐に組み込んだ

$M[q; x].\text{if } x = 1 \text{ then } c!r.\text{nil}$

$EPR_{q,r}$

1/2



if 0 = 1 **then** c!r.nil

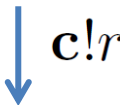
$|00\rangle\langle 00|_{q,r}$

1/2



if 1 = 1 **then** c!r.nil

$|11\rangle\langle 11|_{q,r}$



1/2

nil

$|11\rangle\langle 11|_{q,r}$

ツールの設計 (確率分岐)

- 形式化の使い分けを自然に行うため、 $M[q\tilde{; }x]$ は if 分岐に組み込んだ

if q then c!r.discard(q)

$EPR_{q,r}$

1/2
if q
discard(q,r)
 $|00\rangle\langle 00|_{q,r}$

if q
1/2
c!r.discard(q)
 $|11\rangle\langle 11|_{q,r}$

c!r
1/2
discard(q)
 $|11\rangle\langle 11|_{q,r}$

ツールの設計 (確率分岐)

- 形式化の使い分けを自然に行うため、 $M[q\tilde{; }x]$ は if 分岐に組み込んだ

if q then c!r.discard(q)

$EPR_{q,r}$

if q
↓

discard(q,r)
 $1/2|00\rangle\langle 00|_{q,r}$

if q
↓

c!r.discard(q)
 $1/2|11\rangle\langle 11|_{q,r}$

↓ c!r

discard(q)
 $1/2|11\rangle\langle 11|_{q,r}$

ツールの設計 (確率分岐)

- 形式化の使い分けを自然に行うため,
 $M[q\tilde{;}x]$ は if 分岐に組み込んだ

if q then c!r.discard(q)
EPR[q,r]

if q
↓

discard(q,r)
 $1/2|00\rangle\langle 00|_{q,r}$

if q
↓

c!r.discard(q)
 $1/2|11\rangle\langle 11|_{q,r}$

↓ c!r

discard(q)
 $1/2|11\rangle\langle 11|_{q,r}$

ツールの設計 (確率分岐)

- 形式化の使い分けを自然に行うため,
 $M[q\tilde{; x}]$ は if 分岐に組み込んだ

if q then c!r.discard(q)

EPR[q,r]

if q
↓

discard(q,r)
proj0[q] (EPR[q,r])

if q
↓

c!r.discard(q)
proj1[q] (EPR[q,r])

c!r
↓

discard(q)
proj1[q] (EPR[q,r])

双模倣の検証

- 状態遷移の対応を調べる
- 外の人(攻撃者)がアクセス可能な量子状態が常に一致しているか確認する

量子状態の等しさの判定

- コンフィグレーション $\langle P, \rho \rangle$ に対し,
外の人(攻撃者)がアクセス可能な量子状態は,
部分トレース $\text{Tr}[qv(P)](\rho)$ で表される
- ツールは, **変数の出現に着目して**,
消去できる記号を消去する (トレースアウト)

$$\text{Tr}[qv(P)](\rho)$$

全体の量子状態 ρ のうち, P に現れる量子変数に
対応する空間を, 攻撃者に見えなくする

トレースアウト

例

$$\text{Tr}[q, r](\text{op1}[q](X[q]*Y[r]*Z[s]))$$

$$\text{Tr}[q, r](\text{op2}[q](U[q]*Z[s]*V[r]))$$

トレースアウト

例

$$\text{Tr}[q, r](\text{op1}[q](X[q]*Y[r]*Z[s])) \rightarrow Z[s]$$

$$\text{Tr}[q, r](\text{op2}[q](U[q]*Z[s]*V[r])) \rightarrow Z[s]$$

ユーザ定義等式の適用

- 左辺にマッチする部分を, 右辺に書き換える

ユーザ定義 : $\text{Tr}[q](\text{EPR}[q, r]) = \text{Tr}[q](\text{PROB}[q, r])$

対象の環境 : $\text{Tr}[q](\text{op}[s, t](\text{EPR}[q, r]*X[s]*Y[t]))$

↓

$\text{Tr}[q](\text{op}[s, t](\text{PROB}[q, r]*X[s]*Y[t]))$

- 判定する機会ごとに,
各等式は高々一度ずつしか適用しない
- 等式適用の手続は必ず停止する

量子状態の等しさの判定

- トレースアウトと等式適用の両方によって、書き換える
- 書き換えたあとの環境どうしが構文的に等しいか判定する
 - 量子演算の記号, 状態の記号を適切に交換して, 等しい記号列にできるかを調べる

$$\begin{aligned} \text{(例)} \quad & E[q](F[r, s](U[q]*V[r]*W[s])) \\ & = \\ & F[r, s](E[q](V[r]*U[q]*W[s])) \end{aligned}$$

ツールの概要

- Ruby 1.9.2で動作確認
- 入力: 双模倣を検証したい二つの
コンフィグレーション $\langle P, \rho \rangle, \langle Q, \sigma \rangle$ と,
ユーザ定義等式の集合
- 出力: true または false

ツールの性質

- どんな入力に対しても停止する
- ツールがtrueを出力したとき, 等式が正しければ, $\langle P, \rho \rangle, \langle Q, \sigma \rangle$ は双模倣である
- ツールがfalseを出力しても, $\langle P, \rho \rangle, \langle Q, \sigma \rangle$ は双模倣かもしれない
 - 双模倣であることを示すための等式が不足している可能性がある

等式のためのオプション

- 量子状態, 部分トレース, プロセスを表示する
 - 部分トレースが等しくならないときなど
- 不足している等式を見つけることや, 等式の誤りを見つけるのに役立つ

実験環境

- Panasonic CF-J9
Intel(R) Core(TM) i5 CPU
M460 @ 2.53GHz, 1GB memory

実験結果

- Shor-Preskillの, BB84とEDPの等価性検証
 - 状態遷移木
 - BB84: 588ノード, 165パス
 - EDP: 621ノード, 165パス
 - 双模倣の検証までに, 15.98秒かかった
 - 再帰手続きは, 951回呼び出された

まとめ

- 量子プロセス計算qCCSの双模倣関係を
検証するツールを開発した
 - 量子状態は記号で表される
 - 量子状態間の等式は人間が与える
 - ツールは等式を使って部分トレースを計算し、
状態遷移の対応をチェックする
- Shor-PreskillによるBB84の安全性証明に
検証ツールを適用した
 - BB84とEDPの等価性の自動検証を行った

今後の課題

- 項書き換え戦略のより詳しい検討
- 確率双模倣
 - EDPの安全性証明の形式化・自動化
- 他のプロトコルへの応用
 - B92, 六状態プロトコルなどは, Shor-Preskillと似た方法による安全性証明がある