

TOSHIBA

Leading Innovation >>>

CryptoVerif の証明能力の改良： 誤った判定の回避

◎ 花谷 嘉一 †
角野 陽輔 †
米山 一樹 †
太田 和夫 †

† 株式会社 東芝
† 電気通信大学

我々の発表論文の関係

DH仮定の
定式化



Gap DH仮定
の定式化

SCIS2009 2009年1月
CryptoVerifを用いたFDH署名の
緊密な安全性証明の検討

FAIS 2007年9月
task-PIOAフレームワークと
Blanchetフレームワークの証明能
力に関する一考察

厳密な
証明への
挑戦

他方式との
比較

異常終了により検証でき
ない場合がある

SCIS2009 2009年1月
安全性検証ツールCryptoVerifの改良:
異常終了に対する一対策

誤った変形により
安全性を誤判定する

今回の発表

発表の流れ

- **CSS2008 で行った発表**
 - 背景
 - FDH署名とGDH署名の安全性証明
 - 考察
- **defined() と otheruses()**
- **対策案と効果の考察**
- **まとめ**

• CryptoVerif ってどれくらい証明能力があるんだろう？

– 既存研究：方式の検証例

- FDH署名, [BR93]のPKE, 鍵交換プロトコルなど・・・ Blanchet ら
- Authenticated Encryption 関係・・・ 荒井ら@信州大
- DH鍵交換(弱い意味での安全性)・・・ 花谷ら

– 既存研究：書き換え規則の定式化

- 落とし戸付き一方向性置換, ランダムオラクル, 各種プリミティブ・・・ Blanchet ら
- CDH仮定, DDH仮定・・・ 花谷ら

GDH署名の安全性証明ができるか調べてみよう！

ランダムオラクルモデル, Gap DH仮定の下で
FDH署名とほぼ同じ手順で証明可能な方式. [BLS01]

ランダムオラクル : Blanchetが定式化済.

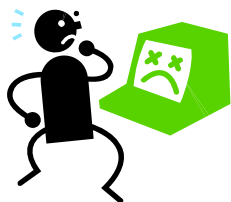
Gap DH仮定 \leftarrow CDH仮定 + DDHオラクル \leftarrow ~~CDH仮定~~



証明能力を探るために、色々な方式の安全性証明をしてみよう！

今回の対象：GDH署名：ランダムオラクルモデル，Gap DH仮定.
FDH署名とほぼ同じ手順で安全性証明可能.

- ・ Gap DH仮定の定式化
- ・ 指数法則の定式化
- ・ GDH署名に対する偽造ゲームの定式化



Gap DH仮定を適用する前に 安全性証明がついてしまった。
明らかに 誤った証明！

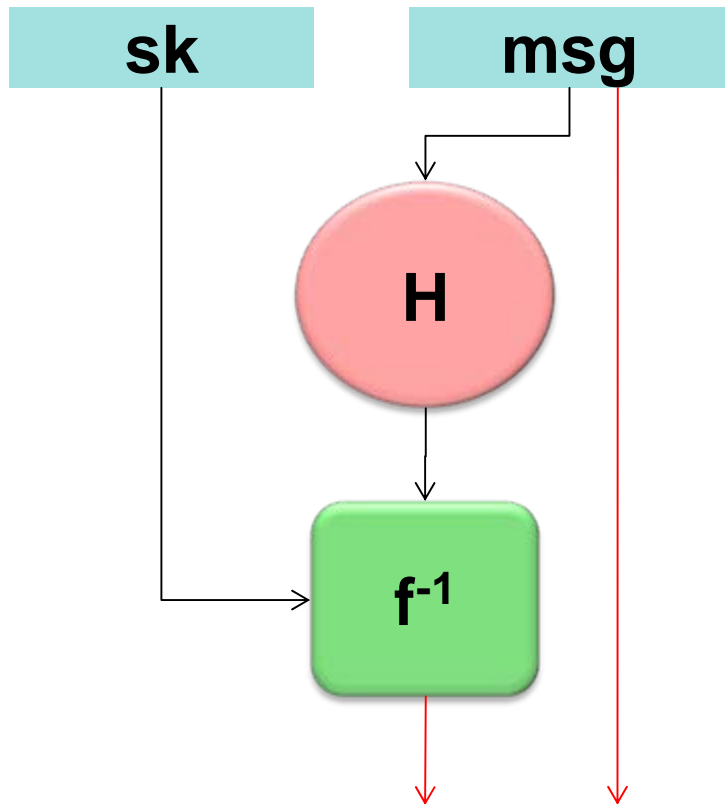


CryptoVerifで用いるランダムオラクルの記述に問題がありそうだ。
取り敢えず問題を回避する方法は見つけた。

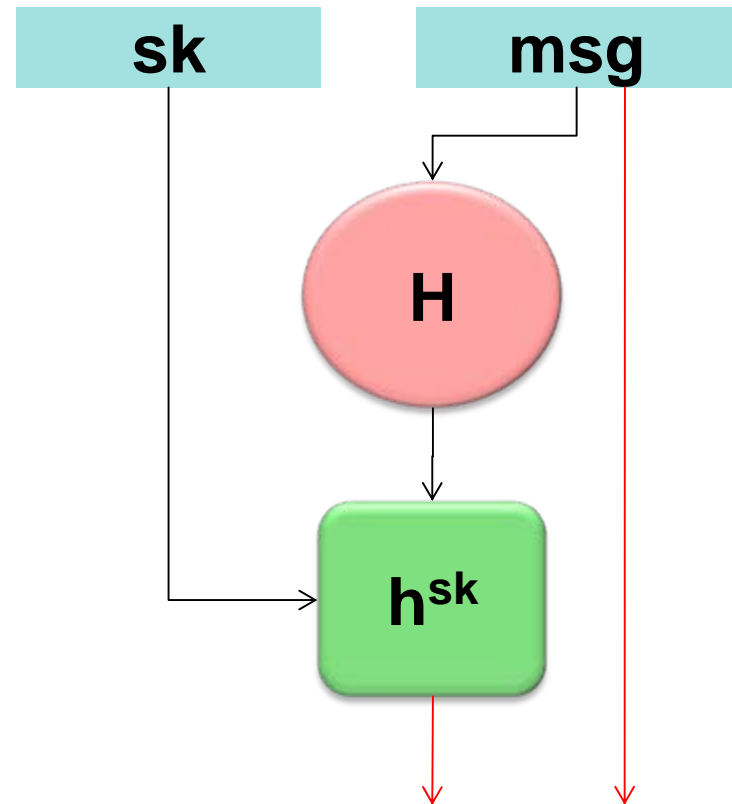
発表の流れ

- **CSS2008 で行った発表**
 - 背景
 - FDH署名とGDH署名の安全性証明
 - 考察
- **defined() と otheruses()**
- **対策案と効果の考察**
- **まとめ**

FDH署名



GDH署名



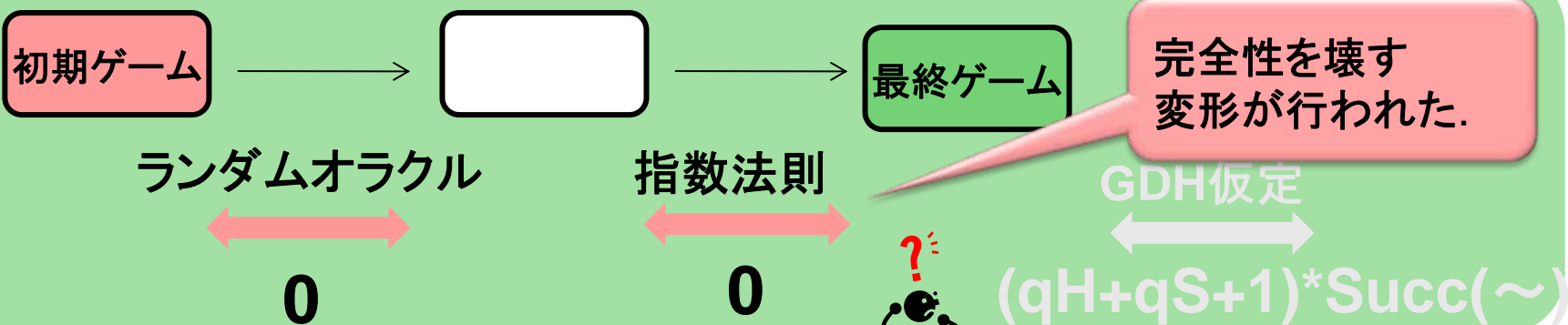
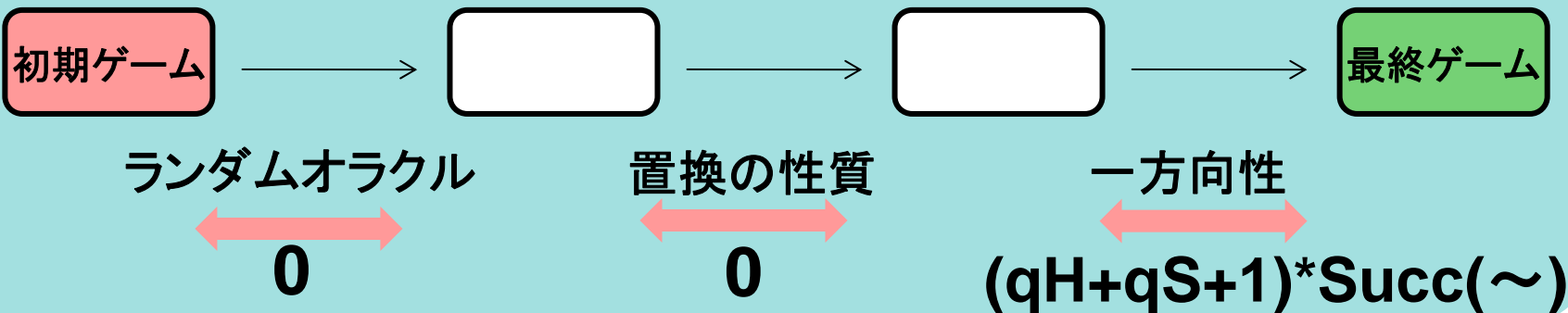
計算量的仮定は異なるが、
FDH署名と同様に安全性証明可

FDH署名のための規則

ランダムオラクル : 0
置換の性質 : 0
一方向性 : $nk*nf*Succ(\sim)$

GDH署名のための規則

ランダムオラクル : 0
指数法則 : 0
Gap DH仮定 : $nk*nf*Succ(\sim)$



発表の流れ

- **CSS2008 で行った発表**
 - 背景
 - FDH署名とGDH署名の安全性証明
 - 考察
 - 行われた変形
 - 変形の意味
 - 原因とその回避
- **defined() と otheruses()**
- **対策案と効果の考察**
- **まとめ**

ランダムオラクルに対して行われた変形 (CSS2008@沖縄)

0

```
!!_19 <= qH
in(c5[!_19], x: bitstring);
out(c6[!_19], hash(x))
```

ランダムオラクル



指数法則



1.5

```
!!_19 <= qH
in(c5[!_19], x: bitstring);
let x_28: bitstring = cst_bitstring in
find suchthat defined(y_37, m', x_24, r_23) && otheruses(r_23) && (x = m') then
  out(c6[!_19], f(pkgen(r), y_37))
orfind @i_34 <= qS suchthat defined(y_35[@i_34], m[@i_34], x_26[@i_34], r_25[@i_34]) && otheruses(r_25[@i_34]) && (x = m[@i_34]) then
  out(c6[!_19], f(pkgen(r), y_35[@i_34]))
orfind @i_33 <= qH suchthat defined(y_39[@i_33], x[@i_33], x_28[@i_33], r_27[@i_33]) && otheruses(r_27[@i_33]) && (x = x[@i_33]) then
  out(c6[!_19], f(pkgen(r), y_39[@i_33]))
else
```

↓ x が問合せ済なら 以前の r を返す

```
new y_39: modq; ← x が初出なら r をランダムに選択して返す
let r_27: typeg = cst_typeg in
out(c6[!_19], f(pkgen(r), y_39))
```

簡略化



2

```
!!_19 <= qH
in(c5[!_19], x: bitstring);
new y_40: Zq;
out(c6[!_19], f(pkgen(r), y_40))
```

2

```
!!_19 <= qH  
in(c5[!_19], x: bitstring);  
new y_40: modq;  
out(c6[!_19], f(pkgen(r), y_40))
```

← 入力に対して、その都度乱数を返す
⇒ 関数じゃない！

ハッシュ関数が 毎回 ランダムな値を返す

⇒ 署名生成時と署名検証時でハッシュ値が異なる

署名検証: 入力: m
 $h \leftarrow H(m)$
 $\sigma \leftarrow h^x$

署名検証: 入力: $(\tilde{m}, \tilde{\sigma})$
 $\tilde{h} \leftarrow H(\tilde{m})$
 $g, v, \tilde{h}, \tilde{\sigma} \stackrel{?}{=} g, g^x, \tilde{h}, \tilde{h}^x$

偽造署名 ⇒ 圧倒的確率で検査式に不合格
偽造に成功する確率は十分小さい。
⇒ 安全な方式だ！

正当な署名 ⇒ 圧倒的確率で検査式に不合格

CryptoVerifでの ランダムオラクルの記述

```
(x:bitstring) nH -> find u <= nH suchthat
  defined(x[u],r[u])
  ∧ otheruses(r[u])
  ∧ x = x[u]
  then r[u]
else
  new r:G; r.
```

r[u]が find の外で使われていない場合 false を返す。
枝刈り用の関数

↓ 指数法則 に基づく変形

```
(x:bitstring) nH -> find u <= nH suchthat
  defined(x[u],r[u],s[u])
  ∧ otheruses(r[u])
  ∧ x = x[u]
  then f(g,s[u])
else
  new s:Zq*; f(g, s).
```

r が find 内ですら
使われない。
⇒ 全く使われなくなる。

	GDH署名	[BR93] PKE	その他 証明例
オリジナル otheruses 有	×	○	○
オリジナル otheruses 無	○	×	○

- **GDH署名の安全性証明に成功**
 - FDH署名の証明結果とほぼ同じ結果を得た.
- **[BR93]のPKEは証明不可能に**

otheruses は不要とまでは言えないようだ.

発表の流れ

- CSS2008 で行った発表
 - 背景
 - FDH署名とGDH署名の安全性証明
 - 考察
- **defined() と otheruses()**
- **対策案と効果の考察**
- **まとめ**

今回の内容

- **otheruses** を適切に使うことで 正しい証明を行う.
 - defined と otheruses の処理内容を明らかにする.
 - [BR93]のPKE,CSS2008の例を 正しく証明できるよう修正を加える.
 - 修正の方針
 - 変形規則の書き方を工夫する. (CSS2008)
 - ★ otheruses の使い方自体を変更する.

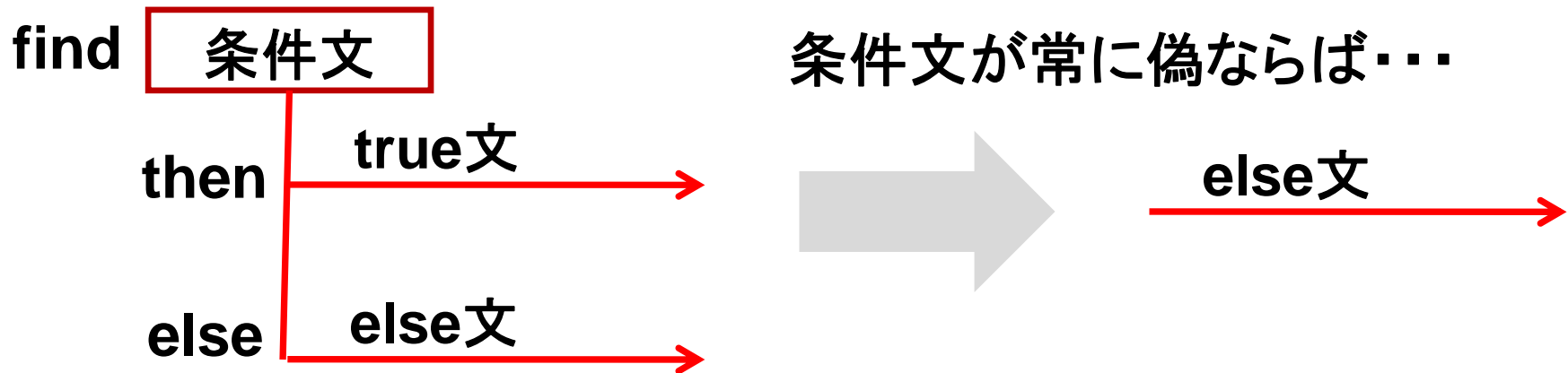
結果: 今回の修正により...

- GDH署名の証明に成功
- [BR93]のPKEの証明にも成功.

今回の修正の効果は要検討.

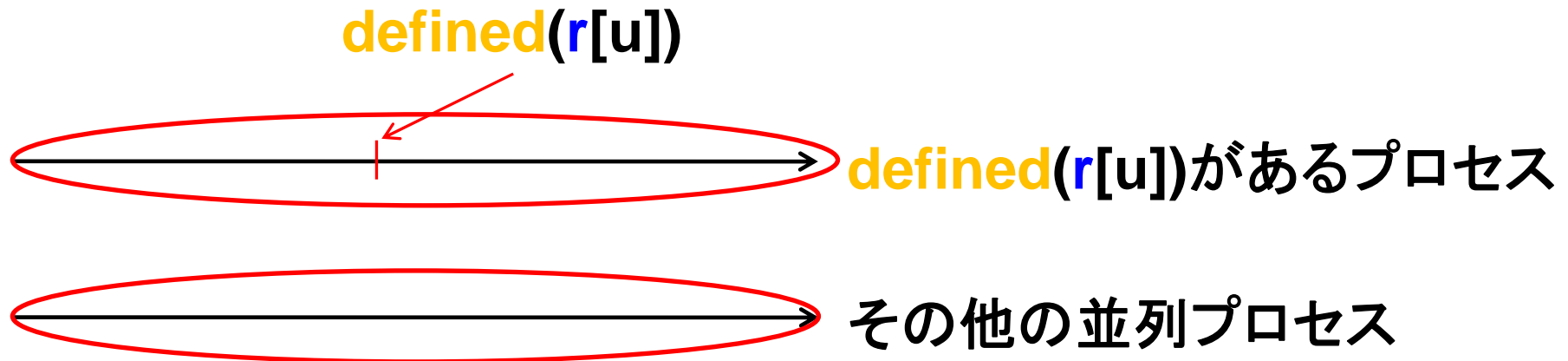
find文と簡約化

$u \leq n$ suchthat
defined(r[u]) && **otheruses**(r[u]) && $x = r[u]$



otheruses は **defined** とセットで用いる.
find 文の簡約化を助けるための関数.

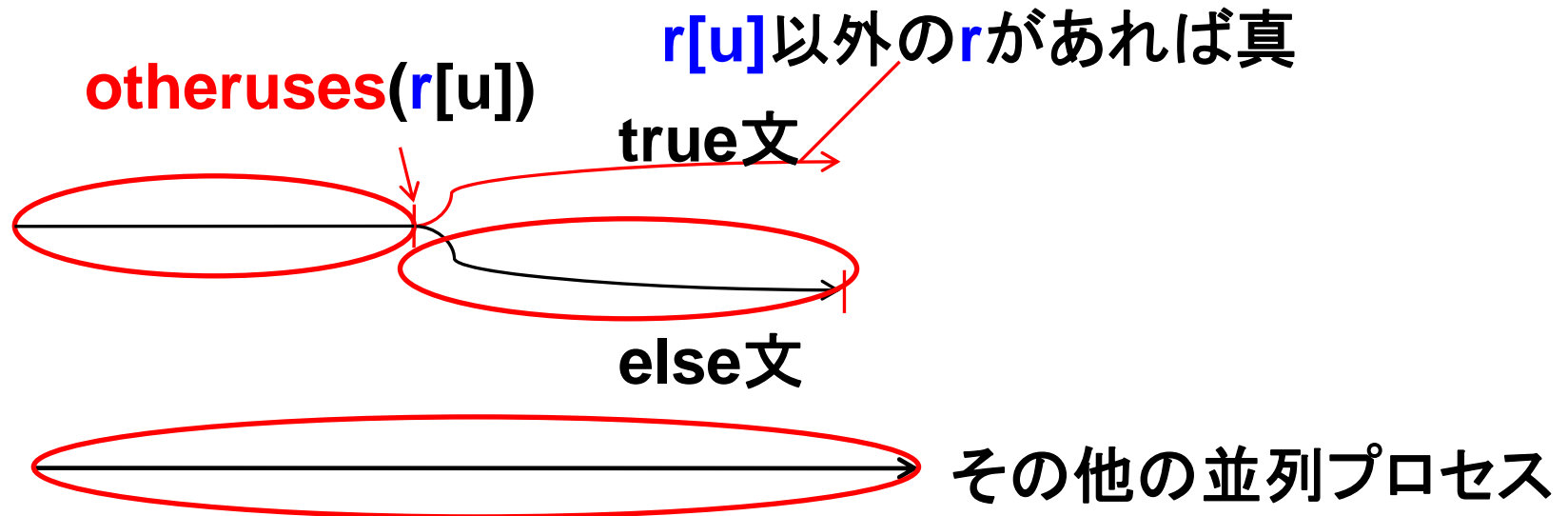
defined(r[u]) の判定の概要



特定の範囲に**r**が存在すれば真, なければ偽と判定

- !**r**
- in(channel, **r**)
- new **r**
- find **r** <= n suchthat defined(b) && otheruses(b) && t
- let **r** = t in p1 else p2

otheruses($r[u]$) の判定の概要

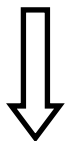


特定の範囲に r が存在すれば真, なければ偽と判定

- `in(channel, r)`
- `if r then p1 else p2`
- `find m <= n suchthat defined($r[m]$) && otheruses($r[m]$) && r`
- `let b = r in p1 else p2`
- `out(channel, r)`
- `event r`

defined と otheruses が関係する書き換え

```
new r:G;  
() → r
```



```
new s:Zq;  
() → f(g,s)
```

```
in(x:bitstring)  
  find u <= nH suchthat  
    defined(x[u],r[u]) ∧ otheruses(r[u])  
    ∧ x = x[u]  
  then out (r[u])  
  else  
    new r:G; out(r).
```

```
in(x:bitstring)  
  find u <= nH suchthat  
    defined(x[u],r[u],s[u]) ∧ otheruses(r[u])  
    ∧ x = x[u]  
  then out (f(g,s[u]))  
  else  
    new s:Zq;  
    let r:G = cst_G in  
    out( f(g,s)).
```

⇒ otheruses(s[u])

検証実験

- Blanchet が与えた証明例 の検証
- 我々が与えた証明例の検証 (CSS2008)

	GDH署名	[BR93] PKE	その他 証明例
オリジナル otheruses 有	×	○	○
オリジナル otheruses 無	○	×	○
修正後 otheruses 有	○	○	○

まとめ

- 目的: CryptoVerif の誤った証明を回避する.
- 方法: ゲームの書き換え規則の適用法に変更を加えた.
 $\text{otheruses}(r[u]) \Rightarrow \text{otheruses}(s[u])$
- 結果: 現在知られている誤った証明例を回避できた.

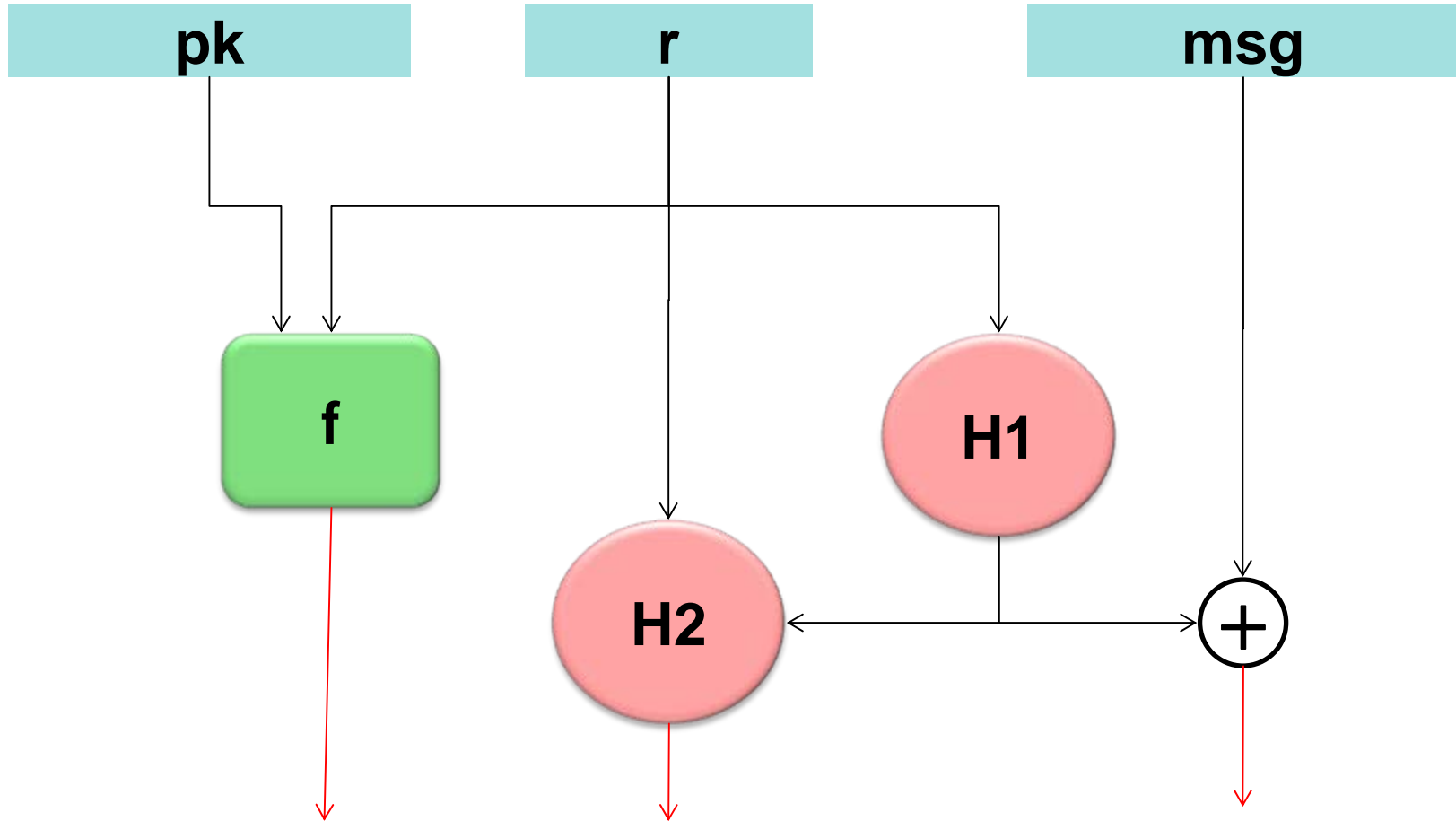
	GDH署名	[BR93] PKE	その他 証明例
オリジナル otheruses 有	×	○	○
オリジナル otheruses 無	○	×	○
修正後 otheruses 有	○	○	○

修正の理論的な正当性は, 未検討.
今後 検討を行う.

TOSHIBA

Leading Innovation >>>

[BR93] の PKE



ランダムオラクルモデル

ランダムオラクル: 出力が一様分布となる理想的なハッシュ関数

ランダムオラクルをシミュレートする関数

```
if  $(m, h) \in H\text{-list}$  then
  return  $(h)$ ;
else
   $r \in_R H$ ;
   $H\text{-list} \leftarrow H\text{-list} \cup \{(m, r)\}$ 
  return  $(r)$ ;
```