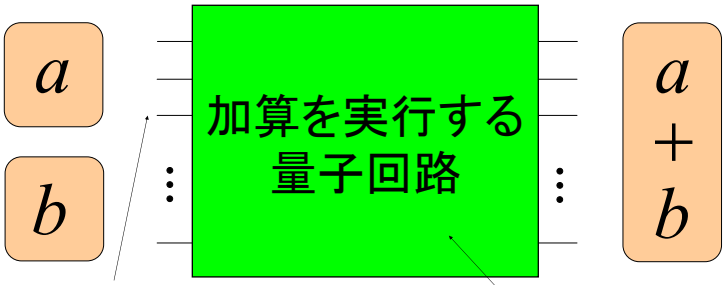
加算を実行する効率的な量子回路

入力



出力

線の数＝量子ビット数

基本的な演算により構成

どのような研究？

- 超高速計算に必要な加算演算を高速かつ少ない量子ビットで実行する方法（量子回路）を提案
- 量子コンピュータの計算能力と暗号の安全性との関係を詳細に分析するための研究

n：加算する数の長さ

	補助量子ビット数	演算数	計算ステップ数
従来回路(2006)	$O(n)$	$O(n)$	$O(\log n)$
提案回路	$O(n/\log n)$	$O(n)$	$O(\log n)$

評価尺度
- 補助量子ビット数＝作業領域として使われる量子ビットの数
- 演算数＝回路を構成する基本的な演算の数
- 計算ステップ数＝並列に実行可能な演算を1グループとしたときのグループの数

従来回路

繰り上がりがあるかどうかを少ない計算ステップ数で判定し
和を計算する Carry-Lookahead（CL）法を採用

出力

$$a + b = s_8 s_7 s_6 s_5 s_4 s_3 s_2 s_1 s_0$$

各桁ごとに並列処理

$C[7,0]$ $C[5,0]$ $C[6,0]$ $C[8,0]$ $C[3,0]$

$C[8,4]$ $C[4,0]$

$C[8,6]$ $C[6,4]$ $C[4,2]$ $C[2,0]$

$C[8,7]$ $C[7,6]$ $C[6,5]$ $C[5,4]$ $C[4,3]$ $C[3,2]$ $C[2,1]$ $C[1,0]$

入力

$$a = a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0$$
$$b = b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$$

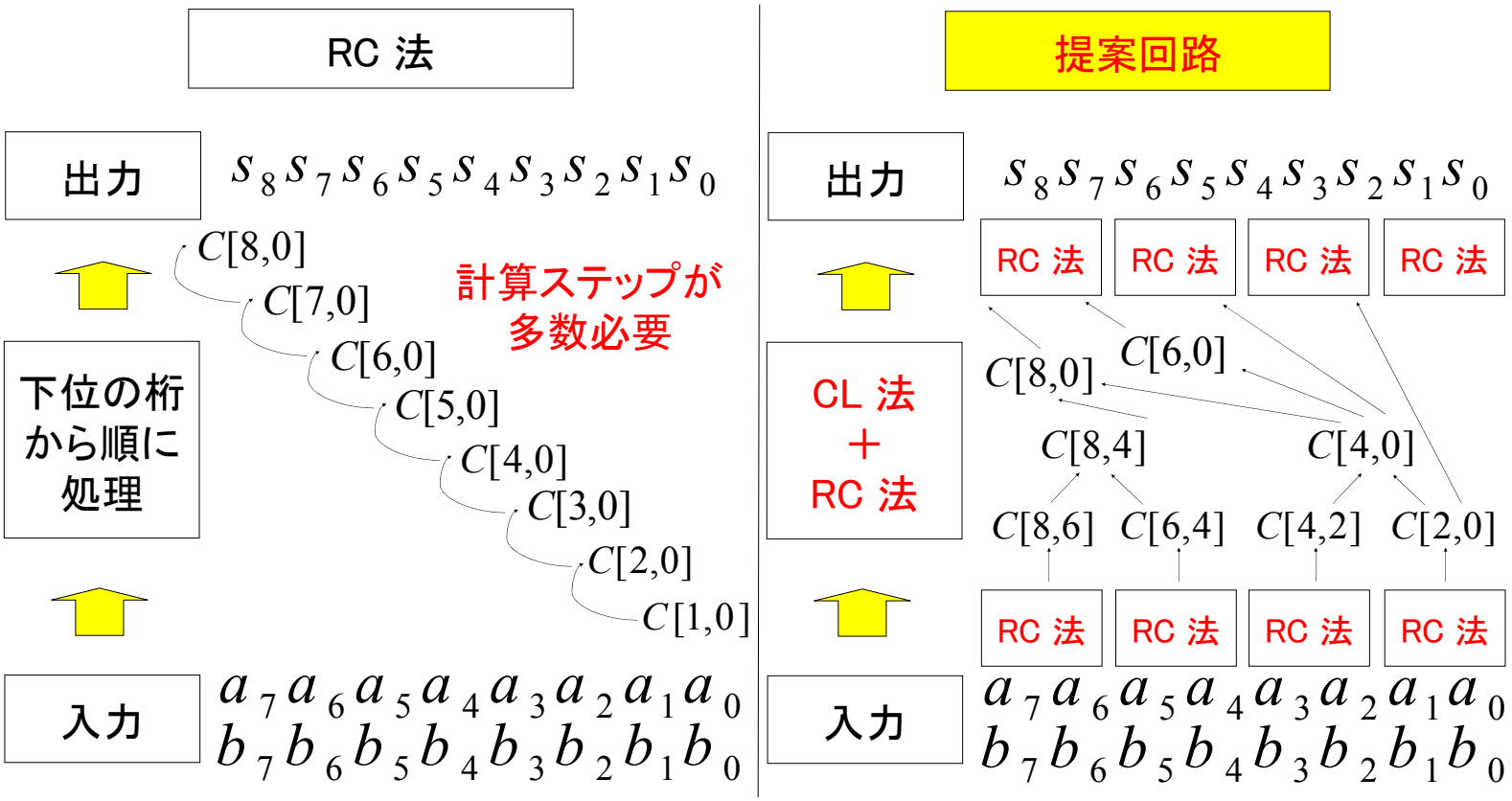
$C[j,i]$

i 桁目から j 桁目の間で繰り上がりがあるかどうかの情報

計算の途中経過を記録する補助量子ビットが多数必要

提案回路の構成のアイデア

計算ステップ数の少ない CL 法と少ない量子ビットしか使わない
Ripple-Carry（RC）法を組み合わせる



RSA 暗号の安全性

少ない量子ビットしかもたない量子コンピュータで
RSA 暗号を高速に破ることができる

因数分解問題を安全性の基礎とする暗号

因数分解アルゴリズムを
実行する量子回路

n：因数分解する数の長さ

	量子ビット数	演算数	計算ステップ数
従来回路の応用	$4n$	$O(n^3 \log n)$	$O(n^2 \log n)$
提案回路の応用	$2n+O(n/\log n)$	$O(n^3 \log n)$	$O(n^2 \log n)$
量子ビット数 最小の回路（2006）	$2n+2$	$O(n^3 \log n)$	$O(n^3)$

さらに詳細に分析し，量子コンピュータに耐え得る暗号の構築に利用