SecHack365

stoq-qtos

学習駆動コース 坂井ゼミ 30Ss 松田 心杏

概要

```
Hurumy@Ponkotsu stoq % ./stoq "Hello world" "" | ./qtos ""
Hello world^C
Hurumy@Ponkotsu stoq % ./stoq "Hello world" "H" | ./qtos ""
zAC{Z_BeX?^C
Hurumy@Ponkotsu stoq % ./stoq "Hello world" "H" | ./qtos ""
?4s)?}^C
Hurumy@Ponkotsu stoq % ./stoq "Hello world" "H" | ./qtos "H"
Hello world^C
```

実際にコマンドを実行した時のスクリーンショット。%以後が実際に入力したコマンド。qtosは入力を中断しない限り永遠に受け付け続けるため、Ctrl+Cで中断している。

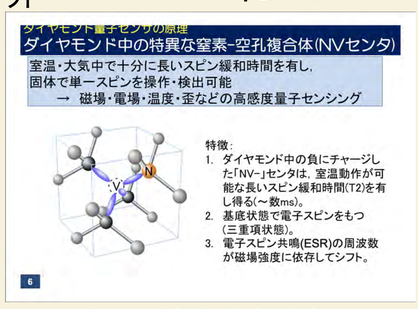
stoq/qtosは、**BB84をモデルとした量子通信エミュレータ**です。ゲームなどに量子通信システムを組み込む際に利用されることを想定し、**高速・軽量・明瞭**であることを特徴とします。プロトコルの骨子のみをシンプルにモデル化することで、触れた人が量子通信や量子回路についてのアイデアを得られることを目標に制作しました。文字列を量子ビット列にするコマンド **stoq**、それを復号するコマンド **qtos** の二つをまとめて扱っています。上の画像のように、stoq コマンドに暗号化したい文字列とかけるゲートを送って暗号化し、それの暗号化したパケットと復号のために使用するゲートをqtosに送って復号します。ゲートは複数同時にかけることができます。復号のための鍵となるゲートの設定が誤っていると、**意味のない文字列**になってしまいます。

BB84とは？

BB84とは、1984年に提唱された量**子鍵配送プロトコル**[1]です。光子などの粒子の偏光や位相といった情報に鍵を載せて通信します。

背景

このようなものを作ろうと思ったきっかけは、わたしが2024年の夏にダイヤモンドを利用した**量子制御を実際にやってみる**という合宿に参加したことです。物理学科3年の学生向けで、量子力学を既に履修し終わっている班のメンバー6人で十日十晩自主ゼミを行ったにも関わらず、実験や実験準備一つとっても非常に難しく、**到底議論し尽くすことができませんでした。**この体験に非常にショックを感じ、なんとかならないだろうかと思いました。そこでモンティ・ホール問題のことを思い出します。モンティ・ホール問題は当時の数学者ですら大いに間違ったのでした。我々は生まれついて素朴な確率の感覚を持っていません。そういった感覚を子どもの頃からゲームを通して培い、そのような**新人類の出現に期待**するのはどうだろうか？適切な教材を用意し、身体的に理解してもらう方法を考えたくなりました。そこで、物理学の知識がなくても簡単に組み込め・楽しめる量子暗号通信ライブラリを作ろう！という考えに至りました。量子コンピュータを学ぶ上で自分にとって一番障害になっていたのは、量子回路を使って特に作れるものがないということです。そこで、暗号通信ができたら楽しみながら量子回路に触れられると考えました。



[2]わたしが実際に実験で作成した
ダイヤモンド量子センサの原理 を説明する図。

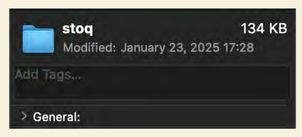
3つの特徴

3つ目の特徴が「明瞭さ」です。明瞭とはよくモデル化されてシンプルで見通しが良い、という意味です。このモジュールは**現状4種類のゲート(H、X、Y、Z)しか持ちません。**そして追加する予定もありません。その方が情報量が多すぎず、とっつきやすいと考えたからです。ビジュアルモデルも敢えて含めず、使い方も非常に単純にし、暗号生成機であることに徹しています。だからこそ応用しやすく、様々なソフトウェアに組み込みやすいと考えています。

さらにもう一種類の明瞭さとして、**「利用者に知識がなくても自分のソフトウェアで利用できる」**という観点があると思います。ここにも留意して、コマンドという考えうる限り最も独立したモジュールとして用意することによって、呼び出し元の言語やFFIなどについて一切考えることなく、シェルが存在し、C++のコンパイルができれば**様々な環境から**このモジュールを利用することができます。

実装の概略

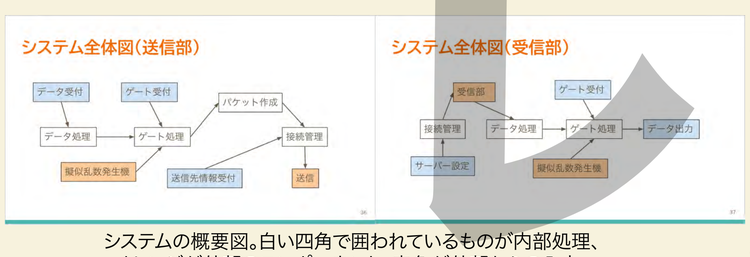
さらに1の高速さともつながりますが、1で説明したのと同じ理由で**動作が非常に軽量**です。量子ビットそのもののエミュレートを真面目に行おうと思ったら、ただでさえ重いゲームには尚更組み込むことができなくなります。このコマンドは先の理由でメモリもほとんど使用しません。ゲームの処理を重くすることなく、あなたのゲームに新たな・未来的な戦略を導入することができます。軽量さを実現するため、実装上も様々な工夫が施されています。例えば行列計算です。量子ビットのエミュレートでは通常行列計算が必要になりますが、これを最も簡単にするため、各ゲートについて**手計算した結果をスカラの式としてコード内にハードコーディング**しています。大変でした。計算ミスがあると考えられるので、ぜひ検算してPRを送ってください。



stoq,qtosのバイナリとソースコードを合わせて134KBしかない。

実装の詳細

コードは**C++**で実装されています。システムの送信部・受信部のワークフローは以下の図の通りです。元々はWEBサーバー、GUIを利用して使っていましたが、利用者の利便性を考え、青い部分をカットして**当初の実装からはかなりダイエットした**形です。結果的にはそれがかなり使いやすさ・とっつきやすさに繋がったと思っています。使い慣れたコマンドの形式で利用できるのは自分としても大変便利でした。



システムの概要図。白い四角で囲われているものが内部処理、
オレンジが外部のコンポーネント、青色が外部からの入力。

パケットの定義も自分で行いました。stoqの出力を吐き出すと下のようになっています。スピンの状態は複素数2つを使って表すことができて、その2つはアルファ・ベータを使って表すことが多いです。そのためこの**a,bはそれぞれが独立した複素数**を表しています。Tupleの一つ目のパラメータがその実数部、二つ目が虚数部の情報を持っています。aを|0>、bを|1>の重みとして書く[3]ことが多いので、このツールの実装もそれに倣っています。しかしこのスクリーンショットを見た方は、「これってもしかして元々”11101101”だったか”00010010”だったのでは？」と思ったかもしれません。まさにその通りで、それが先述した時間計算量O(N)の理由です。しかしこの問題はかけるゲートを変更しながらコマンドを複数回実行することで解決します。それでも最悪O(N)×文章のビット数分bitなのでそこまで遅くはないと考えます。また、数bitごとにゲートを変更していけば実際に判別不可能な暗号になります。

```
QBIT a:(0,-0.707107) b:(0,-0.707107) END
QBIT a:(0,-0.707107) b:(0,-0.707107) END
QBIT a:(0,-0.707107) b:(0,-0.707107) END
QBIT a:(0,-0.707107) b:(0,0.707107) END
QBIT a:(0,-0.707107) b:(0,-0.707107) END
QBIT a:(0,-0.707107) b:(0,-0.707107) END
QBIT a:(0,-0.707107) b:(0,0.707107) END
QBIT a:(0,-0.707107) b:(0,-0.707107) END
```

stoqの出力を実際にターミナルに吐き出したもの。
このパケット形状の定義も自分で行った。

今後の展望

今後の展望として、実際にこれをゲームに組み込んでみて、Moaとして公開するという目標が主にあります。仕上げ方や見せ方、公開手法はかなり悩んだ結果今のようになっているので、これが本当に使う側からしても組み込みやすいのかどうかを確かめたいと考えています。また、盗聴検出は今のところ利用するプラットフォーム側で実装してもらうしかないのが難点だと考えています。その辺りも含めてドキュメントの整備を進める必要もあります。何より一番は、独学で開発を進めたため、専門家の方の意見を聞きたいです。ぜひメールしていただければ幸いです。

参考資料:
[1] 量子鍵配送について教えてください - NTT技術ジャーナル(NTT技術ジャーナル 2025年2月25日)
https://journal.ntt.co.jp/backnumber2/0608/files/jn200608049.pdf
[2] 資料3-1 ダイヤモンド量子センサシステムの可能性(前半) P6
(文部科学省 2025年2月26日)
https://www.mext.go.jp/b_menu/shingi/gijyutu/gijyutu17/010/shiryo/__icsFiles/afieldfile/2016/10/11/1377936_2.pdf
[3] 1-1. 量子ビット
(Quantum Native Dojo! 2025年2月26日)
https://dojo.qulacs.org/ja/latest/notebooks/1.1_qubit_representation.html

NICT National Institute of Information and Communications Technology

作品QR