Teoría de Juegos Cuánticos: Dilema del Prisionero Cuantizado

License MIT

python 3.8+

PyQuil 4.0+

📋 Descripción del Proyecto

Este repositorio contiene una implementación computacional rigurosa del **Dilema del Prisionero Cuántico**, basado en el trabajo seminal de Eisert, Wilkens y Lewenstein (1999). El proyecto demuestra cómo el entrelazamiento cuántico puede modificar fundamentalmente la estructura de equilibrios en teoría de juegos.

Objetivos Académicos

1. **Implementar** el protocolo cuántico de Eisert et al. usando PyQuil
2. **Validar** la recuperación del límite clásico (γ=0)
3. **Analizar** la transición del régimen clásico al cuántico
4. **Demostrar** la existencia de estrategias cuánticas que dominan las clásicas
5. **Visualizar** y comparar pagos en diferentes configuraciones

🎓 Fundamentos Teóricos

El Dilema del Prisionero Clásico

Dos jugadores eligen simultáneamente entre Cooperar (C) o Traicionar (D). La matriz de pagos satisface:

$$T > R > P > Sy2R > T + S$$

donde:

- **T** (Tentación) = 5: pago al traidor cuando el otro coopera
- **R** (Recompensa) = 3: pago por cooperación mutua
- **P** (Castigo) = 1: pago por traición mutua
- **S** (Sucker) = 0: pago al cooperador cuando el otro traiciona

Paradoja: El equilibrio de Nash (D,D) con pago (1,1) es subóptimo comparado con (C,C) → (3,3).

Cuantización del Juego

El protocolo cuántico introduce tres modificaciones clave:

1. **Entrelazamiento inicial:** Un árbitro aplica el operador J(γ) que crea correlaciones cuánticas

$$J(\gamma) = exp(i\gamma/2(\sigma_x \otimes \sigma_x + \sigma_\gamma \otimes \sigma_\gamma))$$

2. **Estrategias cuánticas:** Los jugadores aplican operadores unitarios U(θ,φ) ∈ SU(2)
3. **Desentrelazamiento:** El árbitro aplica J†(γ) antes de medir

Resultado Clave

Con entrelazamiento máximo (γ=π/2), existe una estrategia cuántica Q que:

- vs Q: ambos obtienen (3,3) - óptimo de Pareto
- vs D: Q evita la explotación
- Rompe el dilema: Q domina a D

🔧 Instalación

Requisitos

```
python >= 3.8
numpy >= 1.20
matplotlib >= 3.3
pyquil >= 4.0
```

Setup

```bash
# Clonar el repositorio
git clone https://github.com/0xGeN02/QuantumPrisionersDilemma.git
cd QuantumPrisionersDilemma

# Crear entorno virtual
python -m venv .venv
source .venv/bin/activate  # En Windows: .venv\Scripts\activate

# Instalar dependencias
pip install -r requirements.txt

# Ejecutar Jupyter
jupyter notebook QuantumPrisonersDilemma.ipynb
```

📊 Estructura del Notebook

El notebook está organizado en 12 secciones:

1. **Introducción Teórica** - Fundamentos del dilema clásico y cuántico
2. **Fundamentos Matemáticos** - Formalismo de espacios de Hilbert y protocolo EWL
3. **Metodología Computacional** - PyQuil y configuración del simulador
4. **Parámetros del Juego** - Definición de matriz de pagos y estrategias
5. **Implementación del Circuito** - Construcción del circuito cuántico
6. **Simulación y Muestreo** - Método de Monte Carlo cuántico
7. **Cálculo de Pagos** - Análisis de utilidad esperada
8. **Análisis Comparativo** - Evaluación de todas las estrategias
9. **Interpretación de Resultados** - Comparación con teoría
10. **Estudio Paramétrico** - Transición clásico-cuántico (γ)
11. **Validación** - Verificación de límite clásico y equilibrios
12. **Conclusiones** - Resultados, implicaciones y trabajo futuro

🧪 Experimentos Implementados

Experimento 1: Caso de Prueba

- Estrategias: C vs D
- Objetivo: Verificar funcionamiento básico
- Resultado esperado: Comportamiento cuántico con $\gamma=\pi/2$

Experimento 2: Análisis Exhaustivo

- Estrategias: {C, D, Q} × {C, D, Q} (9 combinaciones)
- Objetivo: Construir matriz de pagos completa
- Visualización: Gráfico de barras comparativo

Experimento 3: Transición γ

- Barrido: $\gamma \in [0, \pi/2]$ con 10 puntos
- Enfoque: Estrategia (D,D)
- Análisis: Evolución de pagos vs entrelazamiento

Experimento 4: Validación

- Modos: Clásico ($\gamma=0$) vs Cuántico ($\gamma=\pi/2$)
- Objetivo: Verificar recuperación del límite clásico
- Test: Condiciones de equilibrio de Nash

📈 Resultados Principales

Límite Clásico (γ=0)

```
(C,C) → (3, 3)  ✓ Óptimo de Pareto
(C,D) → (0, 5)  ✓ Explotación
(D,C) → (5, 0)  ✓ Explotación
(D,D) → (1, 1)  ✓ Equilibrio de Nash (subóptimo)
```

Régimen Cuántico ($\gamma=\pi/2$)

```
(C,C) → (3, 3)  ✓ Mantiene óptimo
(D,D) → Variable (asimétrico debido a entrelazamiento)
(Q,Q) → (3, 3)  ✓ Nuevo equilibrio óptimo
```

Observación Clave

El entrelazamiento modifica sustancialmente los pagos, permitiendo estrategias cuánticas que escapan del dilema clásico.

🔬 Metodología

Simulador

- **PyQuil WavefunctionSimulator**: Simulación exacta sin ruido
- **Método**: Monte Carlo con N=1000-10000 shots
- **Error estadístico**: $\sigma/\sqrt{N}$

Circuito Cuántico

```
|0�_A ─────[J(γ)]───[RY(θ_A)]───[J†(γ)]───[M]───
|0�_B ─────[J(γ)]───[RY(θ_B)]───[J†(γ)]───[M]───
```

- **J(γ)**: Aproximado como CNOT·RX(γ)·CNOT
- **Estrategias**: Parametrizadas por ángulo θ en RY(θ)
- **Medición**: Base computacional {|00�, |01�, |10�, |11�}

📚 Referencias

Artículos Principales

1. **Eisert, J., Wilkens, M., & Lewenstein, M.** (1999). Quantum games and quantum strategies. *Physical Review Letters*, 83(15), 3077.

arXiv:quant-ph/9806088
2. **Meyer, D. A.** (1999). Quantum strategies. *Physical Review Letters*, 82(5), 1052. arXiv:quant-ph/9804010
3. **Marinatto, L., & Weber, T.** (2000). A quantum approach to static games of complete information. *Physics Letters A*, 272(5-6), 291-303.

Libros de Texto

- **Nielsen, M. A., & Chuang, I. L.** (2010). *Quantum Computation and Quantum Information*. Cambridge University Press.
- **Flitney, A. P., & Abbott, D.** (2002). An introduction to quantum game theory. *Fluctuation and Noise Letters*, 2(04), R175-R187.

Documentación Técnica

- **Rigetti Computing** (2023). *PyQuil Documentation*. pyquil

🤝 Contribuciones

Este proyecto es parte de un trabajo académico. Si deseas contribuir o reportar errores:

1. Fork el repositorio
2. Crea una rama para tu feature (`git checkout -b feature/AmazingFeature`)
3. Commit tus cambios (`git commit -m 'Add AmazingFeature'`)
4. Push a la rama (`git push origin feature/AmazingFeature`)
5. Abre un Pull Request

📝 Licencia

Este proyecto está bajo la licencia MIT. Ver `LICENSE` para más detalles.

👨‍🎓 Autor

Manuel Mateo Delgado-Gambino López

Ingeniería en Sistemas Inteligentes
UiE
@0xGeN02

🙏 Agradecimientos

- Rigetti Computing por PyQuil
- Eisert, Wilkens y Lewenstein por el protocolo original
- UiE por el apoyo institucional

Citación:

```
@misc{quantum_prisoners_dilemma_2025,
  author = {Manuel Mateo Delgado-Gambino López},
  title = {Teoría de Juegos Cuánticos: Una Implementación del Dilema del Prisionero Cuanti
  year = {2025},
  publisher = {@0xGeN02},
  url = {https://github.com/0xGeN02/QuantumPrisionersDilemma}
}
```

📊 Estado del Proyecto

- [x] Implementación del circuito básico
- [x] Validación del límite clásico
- [x] Análisis paramétrico completo
- [x] Visualizaciones académicas
- [x] Documentación exhaustiva
- [ ] Ejecución en hardware real
- [ ] Extensión a N jugadores
- [ ] Análisis con ruido y decoherencia

Última actualización: Diciembre 2025