Optimización de la Transmisión de Estados Cuánticos en Cadenas de Qubits usando Deep Reinforcement Learning y Algoritmos Genéticos.

Sofía Perón Santana (0009-0005-1545-0690)[1,2], Ariel Fiuri (0009-0008-9762-9134)[2], Omar Osenda (0000-0003-3251-7031)[1,2], and Martín Domínguez (0000-0002-1815-0117)[2]

[1] Instituto de Física Enrique Gaviola (CONICET-UNC)
[2] Facultad de Matemática, Astronomía, Física y Computación (UNC)
sofia.peron@mi.unc.edu.ar , ariel.fiuri@unc.edu.ar
omar.osenda@unc.edu.ar , martin.dominguez@unc.edu.ar

Abstract. La transferencia de estado cuántico (QST) a través de cadenas de espín homogéneas desempeña un papel crucial en la construcción de hardware cuántico escalable. Un protocolo básico de transmisión de estados cuánticos prepara un estado en un qubit y lo transfiere a otro a través de un canal, buscando minimizar el tiempo y evitar pérdida de información. La fidelidad del proceso se mide mediante funciones proporcionales a la probabilidad de transición entre ambos estados. Abordamos este problema de optimización mediante pulsos magnéticos constantes y dos estrategias complementarias: aprendizaje por refuerzo profundo, donde un agente aprende secuencias de pulsos mediante recompensas, y algoritmos genéticos, que desarrollan soluciones candidatas mediante selección y mutación. Analizamos la eficiencia de ambos métodos y su capacidad para incorporar restricciones físicas.

Keywords: transmisión de estados cuánticos, aprendizaje reforzado profundo, algoritmo genético

Quantum state transfer (QST) via homogeneous spin chains plays a crucial role in building scalable quantum hardware. A basic quantum state transmission protocol prepares a state in one qubit and transfers it to another through a channel, seeking to minimize the time and avoid information loss. The fidelity of the process is measured by functions proportional to the transition probability between both states. We approach this optimization problem using constant magnetic pulses and two complementary strategies: deep reinforcement learning, where an agent learns pulse sequences through rewards, and genetic algorithms, which develop candidate solutions through selection and mutation. We analyze the efficiency of both methods and their ability to incorporate physical constraints.

Keywords: quantum state transfer, deep reinforced learning, genetic algorithm

1 Introducción

La transmisión de estados cuánticos es un área que lleva más de veinte años de desarrollo (Perón Santana et al., 2025 y las referencias citadas). El protocolo básico es muy simple, se prepara un vector de estado inicial en un extremo de una cadena de qubits y se pretende obtener dicho estado en el otro extremo de la cadena. En Mecánica Cuántica, por vector de estado se entiende un vector *complejo* que pertenece a un espacio vectorial, el cual contiene todos los estados posibles. El estado inicial evoluciona siguiendo el **Hamiltoniano** del sistema, el cual contiene todas las interacciones entre los elementos de la cadena y los controles externos aplicados a la misma. Las interacciones dependen de la física del problema, pero en general se pueden escribir como las interacciones entre dipolos magnéticos.

La ecuación que gobierna la evolución temporal de un vector de estado $\psi(t)$ es la ecuación de Schrödinger

$$i\frac{d\psi(t)}{dt} = H\psi(t), \tag{1}$$

donde el Hamiltoniano H es una matriz. Si el Hamiltoniano es independiente del tiempo, la solución es trivial

$$\psi(t) = \exp(iHt)\psi(0) = U(t)\psi(0). \tag{2}$$

Si el Hamiltoniano depende del tiempo, pero sólo a través de cantidades que son constantes en el tiempo durante un intervalo, la solución se puede generalizar en forma simple

$$\begin{aligned}\psi(t) &= \exp(iH_n\tau_n)\exp(iH_{n-1}\tau_{n-1})\ldots\exp(iH_2\tau_2)\exp(iH_1\tau_1)\psi(0) &(3)\\ &= U_nU_{n-1}\ldots U_2U_1\psi(0) = U\psi(0) &(4)\end{aligned}$$

donde H_n es el Hamiltoniano independiente del tiempo válido en el intervalo de tiempo τ_n y t es el tiempo $t = \tau_1 + \tau_2 + \ldots + \tau_n$.

Vamos a considerar que el Hamiltoniano se puede escribir como

$$H_j = \sum_{i=1}^{N-1}(\sigma_i^x\sigma_{i+1}^x + \sigma_i^y\sigma_{i+1}^y) + \sum_{k=1}^{N} B_{k,j}\sigma_k^z = H_{XX} + \sum_{k=1}^{N} B_{k,j}\sigma_k^z, \tag{5}$$

donde $B_{k,j}$ es el control que se aplica sobre el qubit $k-$ésimo, las matrices complejas $\sigma_i^x, \sigma_i^y, \sigma_i^z$ son las matrices de Pauli. Para convertir el Hamiltoniano en una matriz de dimensión $2^N \times 2^N$ se usa el producto tensorial usual. Los controles B_{jk} pueden configurarse de distinta forma, aquí estudiaremos la implementación de Zhang et al., 2018, que considera controles en los primeros y últimos tres elementos de la cadena. Esto da un total de 16 acciones posibles: todos los controles distintos de cero, las combinaciones de 1,2 o 3 controles distintos de cero en alguno de los dos extremos y todos los controles iguales a cero. En la figura 1a, se muestra en forma de circuito cuántico el protocolo de transmisión y se denota

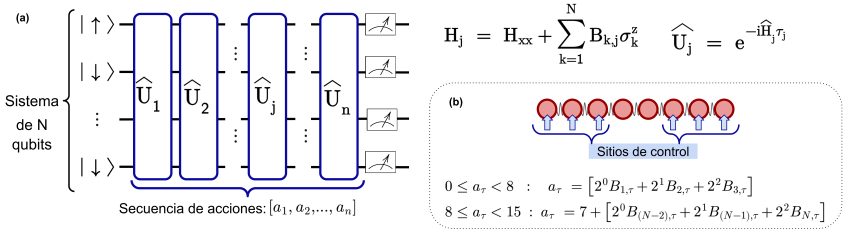

Fig. 1. (a) Representación en forma de circuito cuántico del protocolo de QST. Se inicializa un estado de N qubits con una única excitación y se aplica una serie de *acciones* $[a_1, a_2, ..., a_A]$ que representan compuertas actuando sobre los distintos qubits. (b) Esquema de los sitios de control con los que se efectúan las acciones sobre el sistema. Si B_{jk} es distinto de cero esto implica que el control sobre el k-ésimo qubit en el j-ésimo intervalo de tiempo está encendido.

cada operación U_k como una compuerta que actúa sobre todos los qubits. Un esquema de los sitios de control se muestra en el apartado (b). Note que los campos de control son las *acciones*, tanto el algoritmo genético como el aprendizaje reforzado profundo son usados para determinar secuencias de acciones para obtener el vector de estado inicial en el otro extremo de la cadena.

Si el vector de estado inicial preparado en un extremo de la cadena es ψ_0 y el vector de estado *target* al final de las acciones está dado por ψ_f, transmitir con alta fidelidad significa que hay que maximizar la probabilidad

$$P = |(\psi_f, U\psi_0)|^2, \tag{6}$$

donde se usa el producto interno usual entre matrices y vectores complejos.

2 Métodos

Aprendizaje por refuerzo

En los algoritmos de aprendizaje por refuerzo (RL), un *agente* aprende por medio de *recompensas* que premian o castigan la capacidad para realizar determinada tarea (François-Lavet et al., 2018).

En nuestro trabajo implementamos un modelo del problema físico, el que a su vez constituye el *entorno* con el cual el agente interactúa para generar experiencia y aprendizaje a partir de esta. Este agente tiene la posibilidad de aplicar *acciones* (descriptas en la sección anterior) al entorno y obtiene en contrapartida información de *estado* del mismo y la *recompensa* obtenida por la aplicación de dicha acción. Usamos una función de recompensa relacionada a la *fidelidad* de transmisión de información del sistema influenciado por las diferentes acciones, la cual nos guía en ese proceso. El estado obtenido es el vector de estado de la cadena de qubits. El aprendizaje se logra entrenando el agente usando *diferencia temporal* (QN) en una versión *profunda* (Deep QN) y con una política de

exploración $\epsilon - greedy$. La *función de valor* es aproximada por una red neuronal *fully connected* que es entrenada por *lotes* a partir de la experiencia generada en el proceso y que recibe como entrada el estado del entorno y como salida los valores aproximados de la función de valor para ese estado en cada una de las acciones (Sutton and Barto, 2018).

DRL es cada vez más utilizado para estudiar el control de sistemas cuánticos, ver referencias (Bukov et al., 2018) y (Sgroi et al., 2025). Una descripción más detallada del algoritmo usado puede encontrarse en (Zhang et al., 2018) y sus respectivas citas. La versión empleada en este trabajo sigue el patrón básico descrito en (Doshi, 2021). Para la implementación de las redes neuronales profundas necesarias se utilizó TensorFlow. Nuestra implementación necesita una serie de hiperparámetros que definen la arquitectura, conectividad y número de neuronas por capa, además de los parámetros necesarios en el Q-learning, el learning rate, α, el discount factor. γ, y la función de recompensa (ver aquí).

Algoritmo genético

Los Algoritmos Genéticos (AG) son métodos de optimización estocástica basados en poblaciones. Operan sobre una población $\mathcal{P}$ de individuos, donde cada individuo representa una solución candidata. La calidad de cada individuo se evalúa con una función de aptitud f. El propósito de la evolución es aumentar la aptitud (Mirjalili, 2019).

Cada generación del AG consta de: (i) selección de individuos con probabilidades proporcionales a su aptitud; (ii) cruce de pares parentales mediante operadores de cruce, generando nuevos individuos; y (iii) mutación mediante un operador aleatorio que perturba genes seleccionados. La población se actualiza iterativamente, preservando los individuos más aptos, hasta que se cumple un criterio de convergencia (número máximo de generaciones o umbral de fidelidad).

Nuestro estudio considera como genes a las distintas acciones. Una secuencia completa de acciones representa un individuo de la población. A su vez, la aptitud de cada individuo, es decir, su *fitness* está representada por una función proporcional a la fidelidad de transmisión. En particular, para comparar con el algoritmo de Deep Reinforcement Learning, diseñamos una *fitness* inspirada en la función recompensa usada por Zhang et al., 2018 de este tipo de algoritmos que también es proporcional a la probabilidad de transición dada por (6).

Dado que hay un número discreto de acciones posibles, elegimos una mutación de tipo *swap* que intercambia los valores de dos genes elegidos de forma aleatoria. Respecto al cruce, elegimos usar uno de tipo uniforme que intercambia valores aleatoriamente entre los individuos parentales en cada uno de los sitios.

Los parámetros usados, además de la *fitness* utilizada pueden consultarse aquí.

3 Resultados

Además de reproducir los resultados del trabajo de (Zhang et al., 2018), gracias a que los autores proveyeron algunos de sus códigos fuente, exploramos distintos

conjuntos de parámetros a fin de mejorar sus resultados. En dicho trabajo el DRL es usado para detectar el *quantum speed limit*, no para obtener la máxima transmisión de estados posible. Se pudo comprobar que los resultados para la fidelidad de transmisión no son del todo buenos y que el agente no queda entrenado para proveer secuencias que resulten en una buena transmisión. Sin embargo el presente estudio sugiere que cambiando acciones, hiper parámetros y recompensa puede ser posible mejorar los valores de la probabilidad de transmisión. Para el AG usamos la librería PyGAD (Gad, 2024). La figura 2 muestra la comparación de la probabilidad de transmisión obtenida usando ambos métodos en cadenas de qubits de distinto largo.

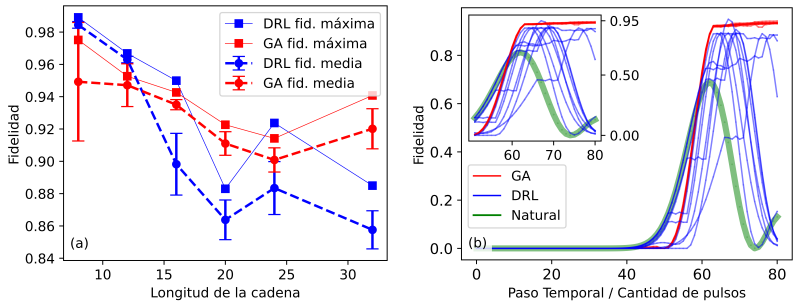

Fig. 2. Panel a): Comparación de los valores de fidelidad obtenidos usando DRL (azul) y algoritmos genéticos (rojo). Teniendo en cuenta que ambos métodos son no deterministas, se toma un promedio sobre 10 ejecuciones. En el caso de DRL, se toman las soluciones correspondientes a los 10 valores más altos de fidelidad logrados por el agente. En el caso del algoritmo genético, se ejecuta el algoritmo 10 veces obteniendo 10 soluciones distintas e independientes. Las curvas se corresponden con los valores de fidelidad medios sobre las 10 ejecuciones y con la fidelidad máxima obtenida en los 10 intentos. Se utiliza mutación de tipo swap, crossover uniforme y una población de 2048 individuos, de la cual se seleccionan 205 padres en cada generación usando *'steady-state-selection'*. Para el algoritmo de DRL, se utilizan los híper-parámetros del trabajo Zhang et al., 2018. Panel b): Se muestra la evolución temporal de la probabilidad de transición para una cadena de 16 espínes para la mejor solución obtenida con ambos algoritmos. Se comparan con la evolución natural (verde), es decir, sin forzamientos.

En el panel a) se observa que para longitudes pequeñas el DRL parece la mejor opción, aunque para longitudes largas el AG muestra claras ventajas. Estas aumentan cuando se observa el panel b). El DRL produce ”picos” en la probabilidad, lo cual limita el tiempo durante el cual la información puede ser adquirida. mientras que el AG obtiene dichos valores durante intervalos considerables, como se muestra en el panel b).

Estos resultados preliminares son interesantes y muestran la necesidad de explorar más la influencia de los hiperparámetros en los resultados obtenidos. Las máximas probabilidades alcanzadas por ambos algoritmos dependen fuertemente

de estas elecciones. Además, utilizando distintas funciones *fitness* (AG) o *recompensas* (DRL) es posible controlar propiedades físicas de la solución obtenida, como su localización. Por otro lado, estamos analizando el efecto de considerar otros grupos de acciones, por ejemplo, utilizando solo controles individuales en cada sitio de la cadena.

Cabe destacar que el algoritmo genético es considerablemente más eficiente en términos de costo computacional ya que, al estar basado en evaluaciones independientes de las distintas secuencias, fue posible paralelizarlo para ejecutar las simulaciones en GPU. De esta manera, pueden obtenerse secuencias que provean fidelidades superiores a un 95 % en minutos. Actualmente, estamos trabajando en formas de optimizar también la implementación del algoritmo de DRL.

Otros gráficos, y algunas extensiones pueden encontrarse en Perón Santana, 2025.

Referencias

Bukov, M., Day, A. G. R., Sels, D., Weinberg, P., Polkovnikov, A., & Mehta, P. (2018). Reinforcement learning in different phases of quantum control. *Phys. Rev. X*, *8*, 031086. https://doi.org/10.1103/PhysRevX.8.031086

Doshi, K. (2021, February). Reinforcement learning explained visually (part 5): Deep q networks. https://tinyurl.com/336cjsux

François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). https://doi.org/10.1561/2200000071

Gad, A. F. (2024). Pygad: An intuitive genetic algorithm python library. *Multimedia Tools and Applications*, *83*(20), 58029–58042.

Mirjalili, S. (2019). *Evolutionary algorithms and neural networks : Theory and applications*. Springer International Publishing.

Perón Santana, S., Domínguez, M., & Osenda, O. (2025). Quantum state transfer performance of heisenberg spin chains with site-dependent interactions designed using a generic genetic algorithm. *Physica Scripta*, *100*(5), 055110.

Perón Santana, S. (2025, June). *Material suplementario jaiio (github)*. https://github.com/sofips/JAIIO_material_suplementario

Sgroi, S., Zicari, G., Imparato, A., & Paternostro, M. (2025). A reinforcement learning approach to the design of quantum chains for optimal energy and state transfer. *Machine Learning: Science and Technology*, *6*(1), 015012. https://doi.org/10.1088/2632-2153/ada71d

Sutton, R. S., & Barto, A. (2018). In *Reinforcement learning: An introduction* (pp. 33–39). MIT Press.

Zhang, X.-M., Cui, Z.-W., Wang, X., & Yung, M.-H. (2018). Automatic spin-chain learning to explore the quantum speed limit. *Phys. Rev. A*, *97*, 052333.