REVISTA DE LA ESCUELA DE FÍSICA
UNIVERSIDAD NACIONAL AUTÓNOMA DE HONDURAS
Sección: Investigación • REF-UNAH 14, 11 - 23 (2020) • DOI: 10.5377/ref.v8i1.10085

Aplicación del Algoritmo Cuántico de Grover al problema de Optimización de Calendarización de Agentes (Sin restricciones).

Application of Grover's Quantum Algorithm on Agent's Scheduling Optimization Problem (Without Constraints).

A. FLORES[1]

Recibido: 21 de septiembre de 2019 / Aceptado: 29 de mayo de 2020

[1]Escuela de Física, Facultad de Ciencias, Universidad Nacional Autónoma de Honduras. Tegucigalpa (Honduras). email: aldoflorescordova@gmail.com

El trabajo presenta una curiosa aplicación del algoritmo cuántico de Grover para intentar solucionar el problema de optimización de calendarización de agentes. Comienza con el planteamiento del problema de calendarización y posteriormente desarrolla el algoritmo en detalle. Da especial relevancia al proceso de inversión sobre la media, mostrando de manera explicita la evolución de los estados cuánticos en cada iteración del algoritmo. Finalmente, expone los limites teóricos del algoritmo.

The following paper presents a curious use of Grover's quantum algorithm to try to solve the optimization problem of agents schedulling. It begins with the statement of the schedulling problem and later it developes the algorithm in depth. Gives special attention to the process of inversion over the mean, showing explicitly the evolution of the quantumn states on every iteration of the algorithm. Finally, exposes the theoretical limits of the algorithm.

PALABRAS CLAVES

optimización, algoritmo grover, computación cuántica

KEYWORDS

optimization, grover algorithm, quantum computing

I | INTRODUCCIÓN

UN grupo de agentes recibe llamadas a lo largo del día. Existen horas pico y horas con bajo nivel de llamadas por lo que el número de agentes requeridos fluctua. A esto se le suma la disponibilidad de los agentes que en su mayoría cuentan con restriciones personales. La tarea de Manejo de Personal consiste en diseñar horarios teniendo en cuenta esos y otros puntos, lo que se resume en:

1. El número de agentes calendarizados por hora debe acoplarse a las proyecciones.
2. A un agente no se le puede dar un horario fuera de su disponibilidad [1].
3. Existe un número máximo de horas semanales que un agente puede trabajar [2](ej. 48 horas).
4. Existe un número mínimo de horas semanales que un agente debe trabajar [3](ej. 20 horas).

En este trabajo no se tendrá en cuenta los puntos 3 y 4 dejandolos para una próxima ocasión. El punto 2 puede ser introducido empleando una nomenclatura para los estados los cuales serán considerados indistinguibles entre ellos.

II | DETALLES TÉCNICOS

1 | Planteamiento

Para resolver el problema lo reduciremos a encontrar el horario de los agentes por dia, de esta manera las proyecciones por dia solo dependen de la hora. La función de proyecciones puede ser escrita como $P = P(t)$, donde $0 \leq t < 24$.

El horario por dia de un agente estará dado por:

$$x_i(t, \alpha_i, \beta_i) = \mathcal{U}(t - \alpha_i) - \mathcal{U}(t - \beta_i) \tag{1}$$

Donde $a_i \leq \alpha_i < \beta_i \leq b_i$ [4] [5] [6].

Para obtener la distribución de N agentes a lo largo del dia escribimos:

$$X(t, \alpha_0, \beta_0, .., \alpha_{N-1}, \beta_{N-1}) = \sum_{i=0}^{N-1} x_i(t, \alpha_i, \beta_i) \tag{2}$$

La solución al problema la obtenemos al encontrar los α_i y β_i que hagan que la función $X(t)$ tienda a $P(t)$ o de manera equivalente se debe minimizar (Figura 1):

$$A(\alpha_0, \beta_0, .., \alpha_{N-1}, \beta_{N-1}) = \int_0^{24} |P(t) - X(t, \alpha_0, \beta_0, .., \alpha_{N-1}, \beta_{N-1})| dt \tag{3}$$

[1] Limitación impuesta por el agente

[2] Limitación impuesta por ley

[3] Limitación impuesta por la empresa

[4] $\mathcal{U}$ es la función de heaviside

[5] α_i y β_i son las horas de entrada y de salida del i-ésimo agente respectivamente.

[6] $[a_i, b_i]$ representa el intervalo de disponibilidad del i-ésimo agente.

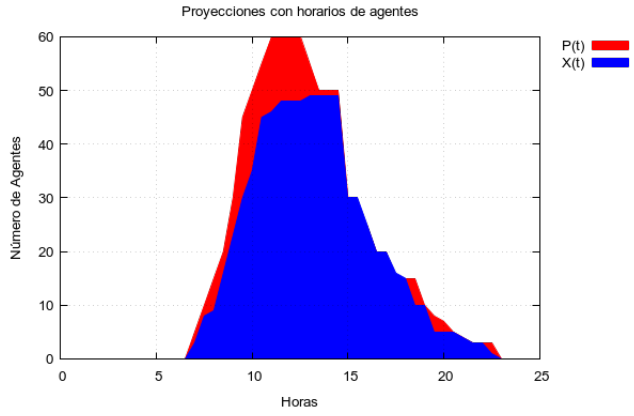


Figura 1: Comparacion entre proyecciones(Agentes necesitados) y calendarizacion(Agentes calendarizados). Una calendarizacion perfecta equivale a que ambas sean iguales o que el area bajo la curva de la diferencia entre ambas sea cero.

2 | Qbits

Un qbit es la unidad de información básica en un computador cuántico de forma análoga al bit en un computador clásico. Un qbit no es un ente físico concreto, ya que puede ser representado en diferentes sistemas fisicos. Por ejemplo a los bits nos gusta representarlos con potenciales electricos, siendo un potencial alto un 1 y un potencial bajo un 0. Para el qbit podriamos pensar(por razones pedagógicas) en un electrón que tiene un spin, el cual puede estar en dos estados: arriba o abajo. En la vida real no se usan spines sino que sistemas basados en la polarización de fotones, procesadores de estado solido, y combinación de nucleos atómicos (BONILLO, 2013) entre muchas otras propuestas. Nos abstraeremos de la implementación física de estos sistemas y nos concentraremos en sus propiedades.

Lo que hace especial al qbit es una propiedad llamada sobreposición la cual permite al qbit estar en una combinación de sus dos estados puros. Por ejemplo, el spin del electrón está en una sobreposición de sus estados arriba y abajo, unicamente al realizar una medición es que el spin colapsará en uno de esos dos valores. Dicho esto es importante hacer notar que un algoritmo cuántico puede hacer uso de la sobreposición unicamente si "no mide"(o en palabras más técnicas, mantinene la coherencia) los qbits hasta el final para poder extraer la respuesta.

$$|q\rangle = j\,|1\rangle + k\,|0\rangle \tag{4}$$

Siendo $j,k \in \mathcal{C}$ y cumpliendo $|j|^2 + |k|^2 = 1$.

3 | Complejidad Computacional

Para el lector que se pregunta si es buen uso de su tiempo el leer un articulo orientado a la solución de un problema tan banal como el de calendarización, dejeme preguntarle si preferiría leer un árticulo de cura del cancer, o de resolución de las ecuaciones no lineales de la teoria general de la relatividad de Einstein, o de como forzar numeros de tarjetas de credito. Lo cierto es que todos estos problemas son esencialmente el mismo (Cook, 1971), como lo demostró Stephen Cook ya que todos ellos caen en

6-qbit	decimal	interpretación	6-qbit	decimal	interpretación
$\|000000\rangle$	0	0:00	$\|011000\rangle$	24	12:00
$\|000001\rangle$	1	0:30	$\|011001\rangle$	25	12:30
$\|000010\rangle$	2	1:00	$\|011010\rangle$	26	13:00
$\|000011\rangle$	3	1:30	$\|011011\rangle$	27	13:30
$\|000100\rangle$	4	2:00	$\|011100\rangle$	28	14:00
$\|000101\rangle$	5	2:30	$\|011101\rangle$	29	14:30
$\|000110\rangle$	6	3:00	$\|011110\rangle$	30	15:00
$\|000111\rangle$	7	3:30	$\|011111\rangle$	31	15:30
$\|001000\rangle$	8	4:00	$\|100000\rangle$	32	16:00
$\|001001\rangle$	9	4:30	$\|100001\rangle$	33	16:30
$\|001010\rangle$	10	5:00	$\|100010\rangle$	34	17:00
$\|001011\rangle$	11	5:30	$\|100011\rangle$	35	17:30
$\|001100\rangle$	12	6:00	$\|100100\rangle$	36	18:00
$\|001101\rangle$	13	6:30	$\|100101\rangle$	37	18:30
$\|001110\rangle$	14	7:00	$\|100110\rangle$	38	19:00
$\|001111\rangle$	15	7:30	$\|100111\rangle$	39	19:30
$\|010000\rangle$	16	8:00	$\|101000\rangle$	40	20:00
$\|010001\rangle$	17	8:30	$\|101001\rangle$	41	20:30
$\|010010\rangle$	18	9:00	$\|101010\rangle$	42	21:00
$\|010011\rangle$	19	9:30	$\|101011\rangle$	43	21:30
$\|010100\rangle$	20	10:00	$\|101100\rangle$	44	22:00
$\|010101\rangle$	21	10:30	$\|101101\rangle$	45	22:30
$\|010110\rangle$	22	11:00	$\|101110\rangle$	46	23:00
$\|010111\rangle$	23	11:30	$\|101111\rangle$	47	23:30

Tabla 1: Interpretación de Qbits

lo que se conoce como clase de Complejidad NP-Completo. Es decir si encontramos una solución en tiempo polinomial a cualquier problema NP-Completo, habremos solucionado todos ellos. Es por esa razón que hay tanto interés en la comunidad actual (Ikeda, Nakamura, y Humble, 2019). Desgraciadamente este no es el espacio para hablar de tan profundo tema por lo que anexo el siguiente libro para el lector interesado: (Johnson, 1979).

Lo unico que se debe saber es que en computación no solo importa resolver el problema, sino que tambien importa el tiempo que nos toma resolverlo. Si un algoritmo resuelve un problema en un tiempo que crece polinomialmente a sus entradas, decimos que es eficiente. Por otro lado si un algoritmo resuelve un problema en un tiempo que crece exponencialmente a sus entradas decimos que es ineficiente. Todos los problemas NP-Completos conocidos se resuelven en un tiempo exponencial.

III | ALGORITMO CUÁNTICO

Se representan los 48 diferentes estados que corresponden a cada media hora desde el inicio del dia en un 6-qbit en analogía a un 6-bit de computación clásica [7].

[7] En computación clásica al usar un 6-bit se representa uno y solo uno de los enteros del 0-63. En cambio un 6-qbit representa todos ellos a la vez debido a la superposición de estados cuánticos.

$$|\psi\rangle = \prod_{i=0}^{5} |q_i\rangle \tag{5}$$

Se muestra en la tabla 1, con respectivas correspondencias. El estado inicial consta de $2 \times N$ 6-qbits [8]

$$|\psi_0\rangle = (|000000\rangle \otimes |000000\rangle)^{(N)} = (\left|0^{(6)}\right\rangle \otimes \left|0^{(6)}\right\rangle)^{(N)} \tag{6}$$

Los $6 \times N$ q-bits de la izquierda representan las horas de entrada (α_i) y los otros $6 \times N$ q-bits de la derecha representan las horas de salida(β_i).

Al aplicar $6 \times 2 \times N$ puertas de Hadamard a cada qbit[9] se obtiene una sobreposición de todos los estados con igual probabilidad de ser medidos (Strubell, 2011).

$$\begin{aligned}
|\Phi_0\rangle =& \mathcal{H}^{(12N)} |\psi_0\rangle \\
&= \mathcal{H}^{(12N)} (\left|0^{(6)}\right\rangle \otimes \left|0^{(6)}\right\rangle)^{(N)} \\
&= (\mathcal{H}^{(6)} \left|0^{(6)}\right\rangle \otimes \mathcal{H}^{(6)} \left|0^{(6)}\right\rangle)^{(N)} \\
&= (\frac{1}{\sqrt{2^6}} \sum_{\alpha_i=\{0,1\}^6} |\alpha_i\rangle \otimes \frac{1}{\sqrt{2^6}} \sum_{\beta_i=\{0,1\}^6} |\beta_i\rangle)^{(N)} \\
&= \frac{1}{2^{6N}} (\sum_{\alpha_0=\{0,1\}^6} |\alpha_0\rangle \otimes \sum_{\beta_0=\{0,1\}^6} |\beta_0\rangle) \otimes ... \otimes (\sum_{\alpha_{N-1}=\{0,1\}^6} |\alpha_{N-1}\rangle \otimes \sum_{\beta_{N-1}=\{0,1\}^6} |\beta_{N-1}\rangle) \\
&= \frac{1}{2^{6N}} \prod_{i=0}^{N-1} (\sum_{\alpha_i=\{0,1\}^6} |\alpha_i\rangle \otimes \sum_{\beta_i=\{0,1\}^6} |\beta_i\rangle) \\
&= \frac{1}{2^{6N}} \prod_{i=0}^{N-1} \sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6} |\alpha_i\rangle \otimes |\beta_i\rangle
\end{aligned} \tag{7}$$

Se aplica operador oráculo que tiene el siguiente comportamiento (Strubell, 2011):

$$O(\prod_{i=0}^{N-1} |\alpha_i\rangle |\beta_i\rangle) = (-1)^{f(\alpha_0,\beta_0,...,\alpha_{N-1},\beta_{N-1})} \prod_{i=0}^{N-1} |\alpha_i\rangle |\beta_i\rangle \tag{8}$$

donde la función f es:

$$f(\alpha_0, \beta_0, ..., \alpha_{N-1}, \beta_{N-1}) = \mathcal{U}(A_{max} - A(\alpha_0, \beta_0, ..., \alpha_{N-1}, \beta_{N-1})) \tag{9}$$

A_{max} es un limite que corresponde a un porcentaje del area bajo la funcion de proyeccion. Por ejemplo si $A_{max} = p_0 \int_0^{24} |P(t)|dt$ el [10] algoritmo encontrará los estados $\prod_{i=0}^{N-1} |\alpha_i\rangle |\beta_i\rangle$ que se acoplen de un $(100 - p_0)\,\%$ a un $100\,\%$ a las proyecciones.

[8] La notación es una abreviación del producto tensorial de $12 \times N$ espacios de hilbert correspondiente a los qbits. Escrito de manera explicita es $\underbrace{|0\rangle \otimes |0\rangle \otimes ... \otimes |0\rangle}_{12N \text{ veces}}$

[9] La forma explicita de la expresión $\mathcal{H}^{(m)} \left|0^{(m)}\right\rangle$. es $\underbrace{\mathcal{H}|0\rangle \otimes \mathcal{H}|0\rangle \otimes .... \otimes \mathcal{H}|0\rangle}_{m \text{ veces}}$

[10] $0 < p_0 < 1$

La función A se define de la misma manera a como se definio en (3) pero multiplicando los α_i y β_i por un factor de escala dado por $t_{min} = 0.5$. Aplicando O al estado en sobreposición [11] $|\Phi_0\rangle$:

$$\begin{aligned} O(|\Phi_0\rangle) &= O(\frac{1}{2^{6N}} \prod_{i=0}^{N-1} \sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6} |\alpha_i\rangle \otimes |\beta_i\rangle) \\ &= \frac{1}{2^{6N}} O(\prod_{i=0}^{N-1} [\sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6}] \prod_{i=0}^{N-1} [|\alpha_i\rangle \otimes |\beta_i\rangle]) \\ &= \frac{1}{2^{6N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6}] O(\prod_{i=0}^{N-1} [|\alpha_i\rangle \otimes |\beta_i\rangle]) \\ &= \frac{1}{2^{6N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6}] [(-1)^{f(\alpha_0,\beta_0,\ldots,\alpha_{N-1},\beta_{N-1})} \prod_{i=0}^{N-1} |\alpha_i\rangle \otimes |\beta_i\rangle] \end{aligned} \tag{10}$$

El operador de difusión $\mathcal{D}_0$ se define como: $\mathcal{D}_0 = 2|\Phi_0\rangle\langle\Phi_0| - I$. Este operador es aplicado sobre $O|\Phi_0\rangle$ para completar la primera iteración del algoritmo. El estado resultante es:

$$\begin{aligned} |\Phi_1\rangle &= (2|\Phi_0\rangle\langle\Phi_0| - I) O(|\Phi_0\rangle) \\ &= \frac{2}{2^{18N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6} \sum_{\alpha_i'=\{0,1\}^6} \sum_{\beta_i'=\{0,1\}^6}] ((-1)^{f(\boldsymbol{\alpha'},\boldsymbol{\beta'})} \prod_{i=0}^{N-1} [|\alpha_i\rangle |\beta_i\rangle]) \\ &- \frac{1}{2^{6N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6}] ((-1)^{f(\boldsymbol{\alpha},\boldsymbol{\beta})} \prod_{i=0}^{N-1} [|\alpha_i\rangle |\beta_i\rangle]) \end{aligned} \tag{11}$$

Para medir la probabilidad de elegir una respuesta correcta [12] primero es necesario obtener la función de probabilidad de medir un estado cualquiera $\prod_{i=0}^{N-1} |\alpha_i\rangle |\beta_i\rangle$:

$$\begin{aligned} \mathcal{P}(\boldsymbol{\alpha},\boldsymbol{\beta}) &= \left| (\prod_{i=0}^{N-1} \langle\alpha_i| \langle\beta_i|) |\Phi_1\rangle \right|^2 \\ &= |\frac{2}{2^{18N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i''=\{0,1\}^6} \sum_{\beta_i''=\{0,1\}^6} \sum_{\alpha_i'=\{0,1\}^6} \sum_{\beta_i'=\{0,1\}^6}] ((-1)^{f(\boldsymbol{\alpha'},\boldsymbol{\beta'})} \prod_{i=0}^{N-1} [\delta(\alpha_i - \alpha_i'')\delta(\beta_i - \beta_i'')]) \\ &- \frac{1}{2^{6N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i''=\{0,1\}^6} \sum_{\beta_i''=\{0,1\}^6}] ((-1)^{f(\boldsymbol{\alpha''},\boldsymbol{\beta''})} \prod_{i=0}^{N-1} [\delta(\alpha_i - \alpha_i'')\delta(\beta_i - \beta_i'')])|^2 \\ &= \left| \frac{2}{2^{18N}} \prod_{i=0}^{N-1} [\sum_{\alpha_i'=\{0,1\}^6} \sum_{\beta_i'=\{0,1\}^6}] ((-1)^{f(\boldsymbol{\alpha'},\boldsymbol{\beta'})}) - \frac{1}{2^{6N}} (-1)^{f(\boldsymbol{\alpha},\boldsymbol{\beta})} \right|^2 \end{aligned} \tag{12}$$

Al introducir s como el número de soluciones que existen, i.e. el número de estados que cumplen con el

[11] La notación $\prod_{i=0}^{N-1} [\sum_{\alpha_i=\{0,1\}^6} \sum_{\beta_i=\{0,1\}^6}]$ indica la yuxtaposicion de las sumatorias $\sum_{\alpha_0=\{0,1\}^6} \sum_{\beta_0=\{0,1\}^6} \ldots \sum_{\alpha_{N-1}=\{0,1\}^6} \sum_{\beta_{N-1}=\{0,1\}^6}$

[12] Es decir la probabilidad de que el estado cuantico dado por $\prod_{i=0}^{N-1} |\alpha_i\rangle |\beta_i\rangle$ tenga observables α_i y β_i tales que $A_{max} > A(\boldsymbol{\alpha},\boldsymbol{\beta})$

criterio $A_{max} > A(\alpha,\beta)$ se tiene:

$$\mathcal{P}(\alpha,\beta) = \left| \frac{2}{2^{18N}}(2^{12N} - 2s) - \frac{1}{2^{6N}}(-1)^{f(\alpha,\beta)} \right|^2 \tag{13}$$

Si ξ corresponde a una respuesta correcta se deduce que:

$$\mathcal{P}(\xi) = \left| \frac{2}{2^{18N}}(2^{12N} - 2s) + \frac{1}{2^{6N}} \right|^2 \tag{14}$$

Por lo que la probabilidad de obtener alguna respuesta correcta, siendo que las respuestas correctas son equiprobables entre ellas, es:

$$\mathcal{P}(\sum \xi) = s \cdot \left| \frac{2}{2^{18N}}(2^{12N} - 2s) + \frac{1}{2^{6N}} \right|^2 \tag{15}$$

Para contextualizar este resultado, hay que suponer el numero de agentes sea 60, dando un total de $2^{12\cdot 60}$ posibles combinaciones(estados). Tambien hay que suponer que la densidad de estados solución es del 1 % [13], la probabilidad de encontrar alguna respuesta correcta despues de la primera iteración del algoritmo es:

$$\mathcal{P}(\sum \xi) = (0.01 \cdot 2^{12\cdot 60}) \cdot \left| \frac{2}{2^{18\cdot 60}}(2^{12\cdot 60} - 2(0.01 \cdot 2^{12\cdot 60})) + \frac{1}{2^{6\cdot 60}} \right|^2 = 8.7616\,\% \tag{16}$$

No esta de más decir que este ejemplo es demasiado idealizado al tener un 1 % de probabilidad inicial de elegir la respuesta correcta. Hubiera sido más sencillo usar un computador clásico para hacer una busqueda aleatoria hasta encontrar la respuesta pero el ejemplo nos ayuda a entender el aumento en la amplitud de probabilidad que generá el algoritmo con una iteración.

Generalizando la ecuación 11 se obtiene la relación de recurrencia:

$$|\Phi_{k+1}\rangle = (2\,|\Phi_k\rangle\,\langle\Phi_k| - I)O(|\Phi_k\rangle) \tag{17}$$

IV | CANTIDAD DE ITERACIONES

Hasta el momento se ha encontrado una expresión que nos permite calcular los estados en cada iteración. ¿Cuántas iteraciones del algoritmo son necesarias para tener certeza de medir la respuesta correcta al medir el estado final?

El algoritmo de grover hace uso del operador de difusión para aumentar la magnitud de los vectores estado asociados a la respuesta correcta. Esta operación no es mas que una inversión sobre la media de todos los estados.

Para poder encontrar cuantas iteraciones son necesarias,se analiza el algoritmo con este nuevo enfoque. Dado dos conjuntos de números $\mathcal{A},\mathcal{B}$, la inversión sobre la media mapea un número $x \in \mathcal{A}$ a otro $x' \in \mathcal{B}$ tal que:

$$x' = 2\bar{x} - x \tag{18}$$

[13] Es verdad que este algoritmo requiere tener cierta noción del numero de respuestas que tiene el problema, pero se puede jugar con escenarios del tipo "peor de los casos"para obtener un limite inferior en el numero de respuestas correctas.

i	Cantidad		Magnitud		Signo		$\bar{z}_i^*$
	$\mathcal{R}e$	Im	$V_i(n,s)$	$W_i(n,s)$	$\mathcal{R}e$	Im	
0			$\frac{1}{\sqrt{n}}$	$\frac{1}{\sqrt{n}}$			$\frac{n-2s}{n^{3/2}}$
1			$\frac{n-4s}{n^{3/2}}$	$\frac{3n-4s}{n^{3/2}}$			$\frac{n^2-8ns+8s^2}{n^{5/2}}$
2	$\sqrt{n-s}$	$\sqrt{s}$	$\frac{n^2-12ns+16s^2}{n^{5/2}}$	$\frac{5n^2-20ns+16s^2}{n^{5/2}}$	$+$	$-$	$\frac{(n-2s)(n^2-16ns+16s^2)}{n^{7/2}}$
3			$\frac{n^3-24n^2s+80ns^2-64s^3}{n^{5/2}}$	$\frac{7n^3-56n^2s+112ns^2-64s^3}{n^{5/2}}$			$\frac{n^4-32n^3s+160n^2s^2-256ns^3+128s^4}{n^{7/2}}$
4			$\frac{(n-4s)(n^3-36n^2s+96ns^2-64s^3)}{n^{7/2}}$	$\frac{(3n-4s)(3n^3-36n^2s+96ns^2-64s^3)}{n^{7/2}}$			$\frac{(n-2s)(n^4-48n^3s+304n^2s^2-512ns^3+256s^4)}{n^{9/2}}$
5			$\frac{n^5-60n^4s+560n^3s^2-1792n^2s^3+2304ns^4-1024s^5}{n^{9/2}}$	$\frac{11n^5-220n^4s+1232n^3s^2-2816n^2s^3+2816ns^4-1024s^5}{n^{9/2}}$			$\frac{(n^2-8ns+8s^2)(n^4-64n^3s+320n^2s^2-512ns^3+256s^4)}{n^{11/2}}$

Tabla 2: Primeras cinco iteraciones del algoritmo

Siendo $\bar{x}$ la media definida en $\mathcal{A}$. Una importante cualidad de esta operación es que preserva la sumatoria de los elementos en $\mathcal{A}$ y más importante aun, la sumatoria cuadrada. Es decir, si cada número correspondiese a la componente de un vector multidimensional, este operador preserva la norma. Estamos ante un operador unitario.[14]

En el algoritmo el conjunto de números corresponde a las magnitudes de los vectores estado. Se supone un número de estados posibles n y un número de soluciones s, inicialmente las magnitudes de los vectores estado eran iguales entre si (7) y luego de aplicar el operador oráculo (10) se obtiene el siguiente estado:

$$O|\Phi_0\rangle = \sqrt{n-s}\frac{1}{\sqrt{n}}\left|\bar{\xi}\right\rangle - \sqrt{s}\frac{1}{\sqrt{n}}|\xi\rangle \tag{19}$$

$|\xi\rangle$ y $\left|\bar{\xi}\right\rangle$ son los estados unitarios asociados a respuestas correctas y los estados asociados a respuestas incorrectas, respectivamente. Debido a que las componentes se mantienen como números reales se representan estos estados con números complejos.

$$z_0^* = \sqrt{n-s}\frac{1}{\sqrt{n}} - i\sqrt{s}\frac{1}{\sqrt{n}} \tag{20}$$

Cuando $n >> s$, la componente real(respuestas incorrectas) será mucho más grande que la componente imaginaria(respuestas correctas), por tanto habrá más probabilidad de elegir erroneamente. Gráficamente esto quiere decir que el número complejo se encuentra mas cercano al eje real. Al efectuar el algoritmo estamos rotando este número un ángulo de $\pi/2$.

Generalizando z_0^* se tiene:

$$z_i^* = \sqrt{n-s}V_i(n,s) - i\sqrt{s}W_i(n,s) \tag{21}$$

La media de z_i^* es definida como:

$$\bar{z_i^*} = \frac{(n-s)V_i(n,s) - sW_i(n,s)}{n} \tag{22}$$

Y análogamente a (18) y (17) obtiene las relaciones de recurrencia:

$$V_{i+1} = 2\bar{z_i^*} - V_i \tag{23}$$

$$W_{i+1} = 2\bar{z_i^*} + W_i \tag{24}$$

Se realiza varias iteraciones del algoritmo y se observa como va cambiando z_i^* con cada una de ellas.

[14] Prueba de que es un operador unitario: $\|\mathcal{B}\|_2 = \sum(x_i')^2 = \sum(2\bar{x} - x_i)^2 = \sum(4\bar{x}^2 - 4\bar{x}x_i + x_i^2) = 4\bar{x}^2 m - 4\bar{x}\sum x_i + \sum(x_i^2) = 4\bar{x}^2 m - 4\bar{x}\bar{x}m + \sum(x_i^2) = \sum(x_i^2) = \|\mathcal{A}\|_2$

Existen algunas observaciones que realizar:

1. De la primera iteración el resultado (15) es recuperado ya que la probabilidad de elegir correctamente en cada iteración es:

$$P_i(\sum \xi) = Im(z_i*)^2 \tag{25}$$

Solo debemos recordar que: $n = 2^{12N}$ siendo N el número de agentes.

2. Todas las funciones $V_i(n,s)$ y $W_i(n,s)$ cumplen la propiedad:

$$(n-s)V_i(n,s)^2 + sW_i(n,s)^2 = 1 \tag{26}$$

Esto no es de sorprender ya que se demostró que la inversión sobre la media preserva la norma.

3. Existe una familia de polinomios asociada a cada $V_i(n,s)$ y $W_i(n,s)$:

$$X_i(n,s) = n^{i+\frac{1}{2}}V_i(n,s) \tag{27}$$

$$Y_i(n,s) = n^{i+\frac{1}{2}}W_i(n,s) \tag{28}$$

Con cada iteración el grado del polinomio aumenta en uno.

Hay una relación entre estos polinomios y los numeros impares. Se hace la siguiente correspondencia: a cada polinomio X_z y Y_z le corresponde el numero $2z+1$. El porque se hace esta corresponcia queda justificado en el siguiente ejemplo:

El polinomio X_7 al ser factorizado en otros polinomios de coeficientes enteros da $(n-4s)(n^2-12ns+16s^2)(n^4-96n^3s+416n^2s^2-576ns^3+256s^4)$. El polinomios X_1 es $n-4s$ y el polinomios X_2 es $n^2-12ns+16s^2$. Observando entonces que X_7 tiene como factores a los polinomios X_1 y X_2. Ahora, el polinomio X_7 le corresponde el 15, y 15 tiene como factores 3 y 5 que corresponden perfectamente con X_1 y X_2.

Esto sería una mera curiosidad si no fuera porque ocurre lo mismo con cualquiera de los polinomios. Si el polinomio corresponde a un numero que se puede factorizar, entonces el polinomio se factoriza en polinomios que correspondan con esos factores numéricos.

El porque la estructura de los números impares aparece en las iteraciones de este algoritmo es algo que se ignora. Y hasta donde corroboró el autor, a la fecha de publicación no hay trabajos que hayan explicitado este detalle. Por ultimo hay que notar que el problema de factorización de enteros en tiempo polinomial sigue estando abierto esperando a ser resuelto, de ahi el interes con estos polinomios que mimetizan esta estructura.[15]

4. Se puede demostrar que:

[15] El problema de factorización de enteros grandes ha sido resuelto usando un algoritmo cuántico(Algoritmo de Shor), pero no hay algoritmo clásico que lo resuelva.

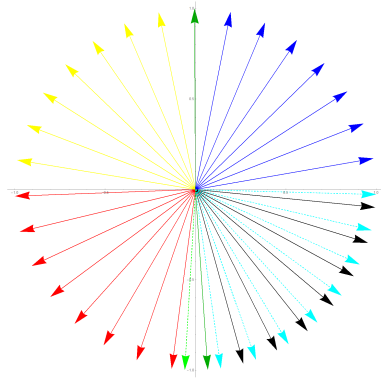


Figura 2: Evolución de z_i* por cada iteración del algoritmo cuando $n = 2^{12\cdot 60}$ y $s = 0.01 \cdot 2^{12\cdot 60}$. Iteraciones (0-6) en negro, (8-15) en rojo, (16-22) en amarillo, (24-30) azul, (31-38) cyan y (7,23,39) verde son las iteraciones en las cuales se maximiza la probabilidad de elegir correctamente.

$$\mathcal{R}e(z_i*)\mathcal{R}e(z_{i-1}*) + Im(z_i*)\,Im(z_{i-1}*) = 1 - \frac{2s}{n}$$

Ahora, el coseno del ángulo entre dos número complejos w y z se define como:

$$\cos(\theta) = \frac{\mathcal{R}e(w)\mathcal{R}e(z) + Im(w)\,Im(z)}{|w||z|} \tag{29}$$

En este caso combinando ambas ecuaciones se obtiene:

$$\theta = \arccos\left(1 - \frac{2s}{n}\right) \tag{30}$$

Es decir, con cada iteración del algoritmo el número complejo es girado un ángulo θ. Graficamente el problema se resuelve cuando el número complejo a girado aproximadamente $\pi/2$, de esta forma hay mayor probabilidad de elegir la respuesta correcta.[16]

$$\theta \cdot i = \frac{\pi}{2} \longrightarrow i = \frac{\pi}{2\arccos\left(1 - \frac{2s}{n}\right)} \tag{31}$$

Lov Grover,diseñador del algoritmo(Grover, 1996), planteó el límite de iteraciones cuando $s = 1$ en: $i = \frac{\pi}{4}\sqrt{n}$. Si $s = 1$ estos dos resultados coinciden en el límite $n \to \infty$ (ver Figura 3).

$$\lim_{n\to\infty}\left(\frac{\pi}{2\arccos\left(1 - \frac{2}{n}\right)} - \frac{\pi}{4}\sqrt{n}\right) = 0 \tag{32}$$

[16] Esto no es del todo cierto ya que si se sigue iterando eventualmente el estado girará aproximadamente un ángulo $3\pi/2$(o $5\pi/2$, $7\pi/2$,etc) y puede ser el caso que en este estado haya una mayor probabilidad de elegir correctamente. Sin embargo bajo el supuesto de minimizar las iteraciones mientras se maximiza la probabilidad, el girar el estado $\pi/2$ resuelve el problema.

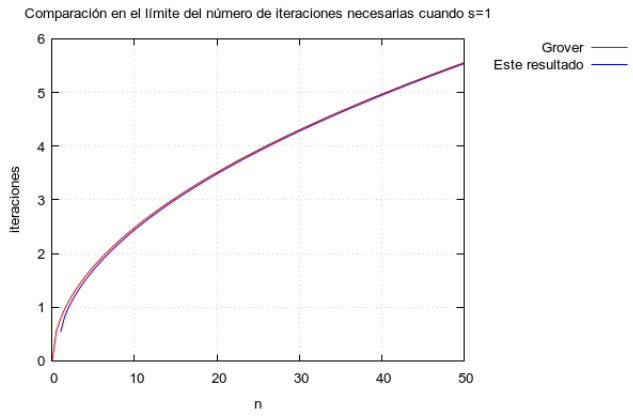


Figura 3: Comparación del número de iteraciones necesarias estimadas originalmente por Grover (rojo) y las calculadas en este trabajo (azul).

V | DISCUSIÓN

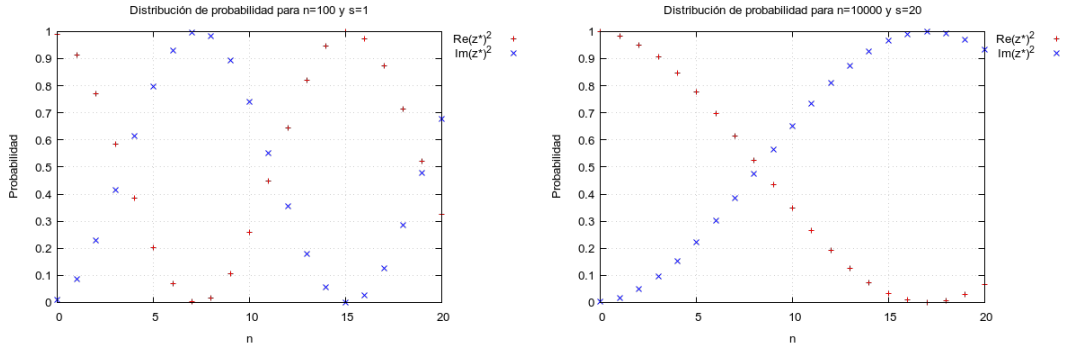
Es interesante que un algoritmo pensando en un inicio para busqueda en base de datos ayude a resolver un problema de optimización. Lo que ocurre es que el algoritmo de Grover en realidad resuelve un tipo de problema más general, todos aquellos que puedan formurlarse en términos de una caja negra (10). Si al lector le parece demasiado artificial la introducción de un operador oráculo que justamente distingue las respuestas correctas, hay que recordar que en computación muchas veces podemos crear funciones que distingan respuestas correctas de incorrectas(y lo hacen en tiempo polinomial) pero no por eso estas funciones son capaces de generar ellas mismas respuestas correctas. Por ejemplo es trivial distinguir si dado un número X este sea factor de un número Y, pero es un problema monumental el encontrar un número X que sea factor de Y si acaso existe.

Generando diferentes valores de n y s, se observa una periodicidad en la probabilidad de elegir correctamente. Tambien se cumple la predicción del número de iteraciones necesarias para alcanzar el primer máximo(31). El pseudoperiodo de la probabilidad de elegir correctamente depende inversamente de n/s, cuando n/s tiende a 0, el periodo(y por tanto el número de iteraciones necesarias para llegar al primer máximo) se alarga.

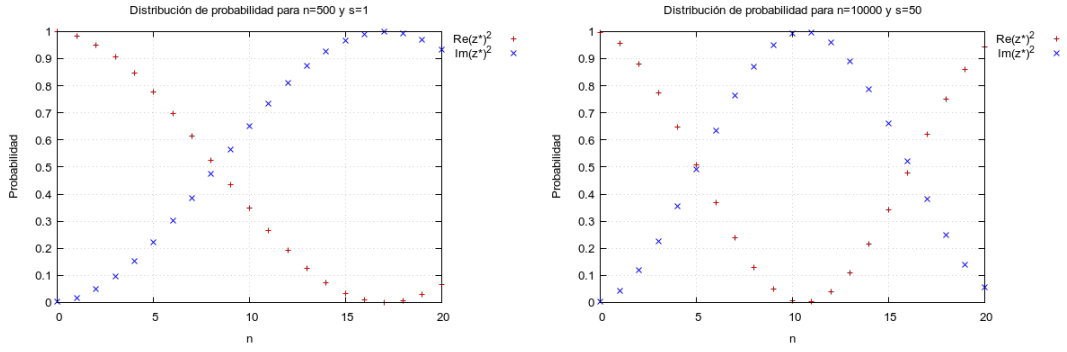
En la Figura 4 la gráfica **a** de la izquierda, coincide con la gráfica **c** de la derecha. Esto es debido a que la fórmula para el cálculo de iteraciones depende unicamente de s/n, el cúal es igual en ambos casos: $1/100 = 100/10000$.

Finalmente hay que preguntarse, ¿se ha encontrado un algoritmo más eficiente? La respuesta es no. Si, se ha encontrado un algoritmo más rapido, es cuadráticamente más rapido.Pero recordando que $n = 2^{12N}$ y suponiendo $n >> s$ el número de iteraciones necesarias es:

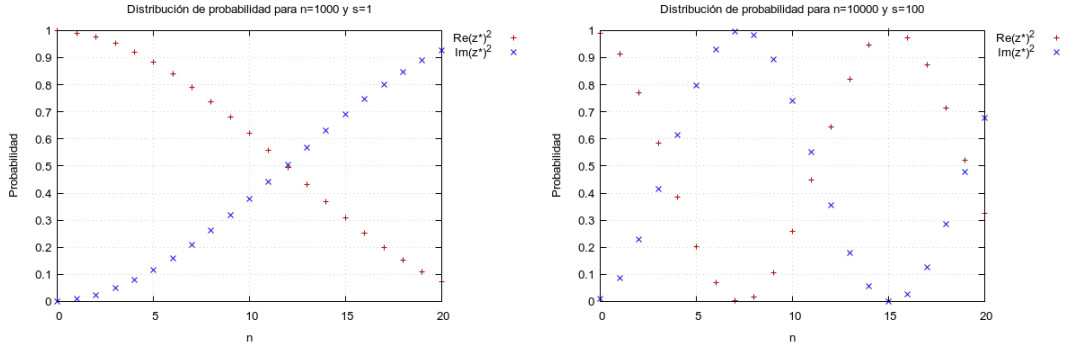
$$i \approx \frac{\pi}{2\arccos\left(1-\frac{2}{2^{12N}}\right)} \approx \frac{\pi}{4}\sqrt{2^{12N}} = \frac{\pi}{4}2^{6N} \tag{33}$$



(a)



(b)



(c)

Figura 4: Distribución de Probabilidad para diferentes valores de n y s.

con tan solo 10 agentes hay que hacer la ingente cantidad de 2^{60} iteraciones. El número de iteraciones sigue dependiendo exponencialmente del número de agentes. Aun así es más rapido que un algoritmo clásico de fuerza bruta que debe realizar 2^{120} iteraciones.

Desgraciadamente el algoritmo de grover es lo más cercano a un algoritmo cuántico óptimo para

problemas de caja negra (Bennett, Bernstein, Brassard, y Vazirani, 1997). Si se desea resolver problemas de forma rapida se deberá explotar la estructura inherente del problema. Quiza en un futuro se encuentre la forma de usar alguna cualidad no descubierta de la mecánica cuántica para generar un algoritmo capaz de resolver problemas NP-Completos (Abrams y Lloyd, 1998).

REFERENCIAS

Abrams, D. S., y Lloyd, S. (1998, Nov). Nonlinear quantum mechanics implies polynomial-time solution fornp-complete and #pproblems. *Physical Review Letters*, *81*(18), 3992–3995. Descargado de http://dx.doi.org/10.1103/PhysRevLett.81.3992

Bennett, C. H., Bernstein, E., Brassard, G., y Vazirani, U. (1997). Strengths and weaknesses of quantum computing.

BONILLO, V. M. (2013). *Principios fundamentales de computación cuántica.*

Cook, S. A. (1971). The complexity of theorem proving procedures. *University of Toronto*.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search.

Ikeda, K., Nakamura, Y., y Humble, T. S. (2019). Application of quantum annealing to nurse scheduling problem.

Johnson, M. R. G. D. S. (1979). *Computers and intractability a guide to the theory of np-completeness* (V. Klee, Ed.). W. H. FREEMAN AND COMPANY.

Strubell, E. (2011). *An introduction to quantum algorithms.*