Complejidad de N-queens Completion

Alexander Baylon, Juan Flores, Jorge Huanca, Carl Villachica

Universidad Católica San Pablo

12 de abril de 2022

Overview

Overview

1. Introducción
2. Definición del problema
3. Secuencia de Reducciones
4. Soluciones
5. Conclusiones

Completación N-Reinas

Al hablar de acerca de la teoría de complejidad, es muy poco habitual no hablar sobre los grupos de problemas importantes como lo son P, NP, NP-Completo, NP-hard.
A continuación presentamos el problema llamado Completación de N-Reinas el cual es un NP-Completo .

Overview

Definición del problema

Definición 1

Una reina es un par de numeros enteros (α, β) con $0 \leq \alpha, 0 \leq \beta$.
Para el resto de estas definiciones se asume que $q = (\alpha, \beta)$ y donde sea apropiada una segunda reina $q_1 = (\alpha_1, \beta_1)$.

Problema 1. **n-Queens**

PROBLEMA: $\langle n \rangle$ donde n es un entero ≥ 1.
SOLUCIÓN: Un conjunto de Q de reinas las cuales caben en el tablero de tamaño n tal que: $|Q| = n$; y por cada dos reinas distintas $q_1, q_2 \in Q$, q_1 no ataca a q_2. Adicionalmente, para cada dos reinas distintas $(\alpha_1, \beta_1), (\alpha_2, \beta_2) \in Q$, ambas $\alpha_1 + \beta_1 \neq \alpha_2 + \beta_2 \pmod{n}$ y $\alpha_1 - \beta_1 \neq \alpha_2 - \beta_2 \pmod{n}$, entonces se dice que Q es una solución "modular" del problema de n-Queens.

Definición del problema

Problema 2. **n-Queens Completion**

PROBLEMA: $M_2 = \langle n, P \rangle$ donde n es un entero y P es un conjunto de Q de reinas las cuales caben en el tablero de tamaño n y no existen dos reinas en P que tengan la misma columna o fila.
SOLUCIÓN: Un conjunto S_2 de reinas el cual es una solución al problema de n-Queens para n tal que $P \subseteq S_2$.

Problema 3.

PROBLEMA: $M_3 = \langle n, P, C, R \rangle$ donde $\langle n, P \rangle$ es una instancia del *Problema 2*, y C, R son conjuntos de enteros tal que $|R| = |C|$ y por cualquier reina $(\alpha_1, \beta_1), (\alpha_2, \beta_2) \in P, \alpha \notin C$ y $\beta \notin R$.
SOLUCIÓN: Un conjunto S_3 de reinas tal que: $|S_3| = |C| + |P|$; $P \subseteq S_3$; no existen dos reinas en S_3 que se ataquen una a la otra; y para cada par $(\alpha, \beta) \in S_3 \backslash P$ tenemos $\alpha \in C, \beta \in R$.

Definición del problema

Problema 4. **Diagonales Excluidas**

PROBLEMA: $M_4 = \langle n, C, R, D^-, D^+ \rangle$ donde $M_3 = \langle n, \{\}, C, R, \rangle$ es una instancia del *Problema 3* D^-, D^+ son un conjunto de enteros con $D^- \subseteq \{-(n-1), ..., n-1\}$, $D^+ \subseteq \{0, ..., 2n-2\}$.
SOLUCIÓN: Un conjunto de reinas S_4 el cual es una solución a M_3 y adicionalmente: para cualquier reina $(\alpha, \beta) \in S_4$ tenemos que $\alpha - \beta \notin D^-$, $\alpha + \beta \notin D^+$.

Problema 5.

PROBLEMA: Un conjunto $M_5 = \{M_{4,a} | 0 \leq a \leq |M_5|\}$, donde cada M_4 es una instancia del *Problema 4* con $D^+ = \{\}$.
SOLUCIÓN: Un conjunto $S_5 = \{S_{4,a} | 0 \leq a \leq |M_5|\}$ donde cada S_4 es una solución a $M_{4,a}$ y adicionalmente: para cualquier $\{S_{4,a}, S_{4,a} \subseteq S_5\}$, y cualquier $(\alpha_a, \beta_a) \in S_{4,a}$, $(\alpha_b, \beta_b) \in S_{4,b}$, tenemos que $(\alpha_a, \beta_a) \neq (\alpha_b, \beta_b)$.

Definición del problema

Problema 6. **1-in-3-SAT Restringido**

PROBLEMA: Un par $M_6 = \langle V, C \rangle$ donde C es un conjunto(de *clausulas*) tal que cada cada $c \in C$ es un conjunto de tres variables, $c = \{v_i, v_j, v_k\}$; y donde $V = \{v | \exists c \in C \cdot v \in c\}$ es el conjunto de todas las variables contenidas en cualquier clausula. Cada variable $v \in V$ ocurre a lo mucho en tres clausulas en C.

SOLUCIÓN: Una asignación de verdad $S_6 : V \to \{true, false\}$ tal que para todo $c = \{v_i, v_j, v_k\} \in C$, S_6 mapea exactamente uno de v_i, v_j, v_k a *true* y los otros dos a *false*.

Prueba.

Porschen, Schmidt, Speckenmeyer, y Wotzlaw (2014, Lemma 4) probaron que este problema es NP-Complete.

Overview

Reducciones

Método

Se realiza una serie de reducciones del *Problema 6* al *Problema 2*. Cada reducción sera *polinomial* y *parsimoniosa*, probando así que el *Problema 2* es *NP-Complete* y *#P-Complete*.

Definición 2 - Reducción del Problema 3 al Problema 2

Esta reducción toma una instancia del *Problema 3* y la $n'-$incrusta en la posición $(0,0)$ de una tabla basada en la *Figura 1*(donde n' se define como parte de la construcción).

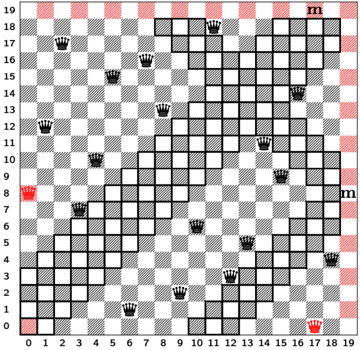

Figura: Figura 1.

Definición 3 - Reducción del Problema 4 al Problema 3

Para la instancia del *Problema 4* $M_4 = \langle n, C, R, D^-, D^+ \rangle$, el conjunto $Q_{D^-} = \{(6n - 3 - d, 3n - 1 - 2d) | d \in D^-\}$, el conjunto $Q_{D^+} = \{(2n - 2 - d, n + 2d) | d \in D^+\}$, y el conjunto $M_3 = \langle 7n - 3, Q_{D^-} \cup Q_{D^+}, \{c + 3n - 2 | c \in C\}, R \rangle$.

Reducciones

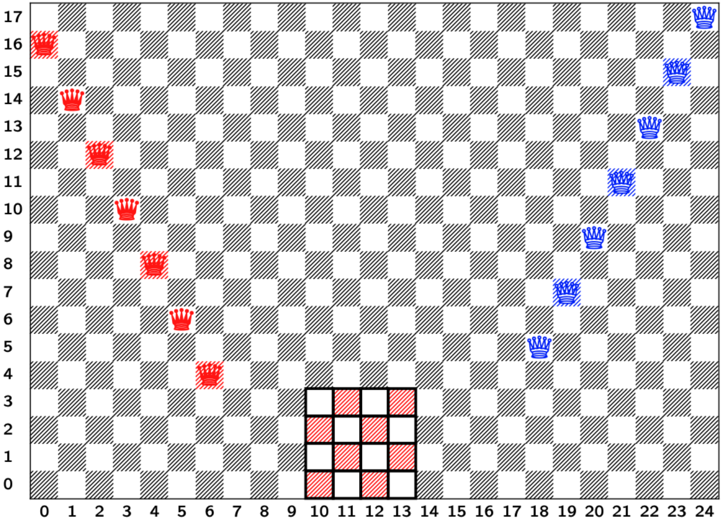

Figura: Figura 2.

Definición 4 - Reducción del Problema 5 al Problema 4

Si $M_5 = \{M_{4,a} \mid a = 0, 1, 2..., k-1\}$ es una instancia del *Problema 5* donde $k = |M_5|$ y cada $M_{4,a} = \langle n_a, C_a, R_a, D_a^-, \{\}\rangle$. Primero se fija $n' = max\{n_a \mid 0 \leq a \leq k\}$.Se posiciona cada artilugio $M_{4,a}$ en la posición (α_a, β_a) en una cuadrícula $n \times n$ y define la instancia del *Problema 4* como $M_4 = \langle n, C, R, D^-, D^+\rangle$.

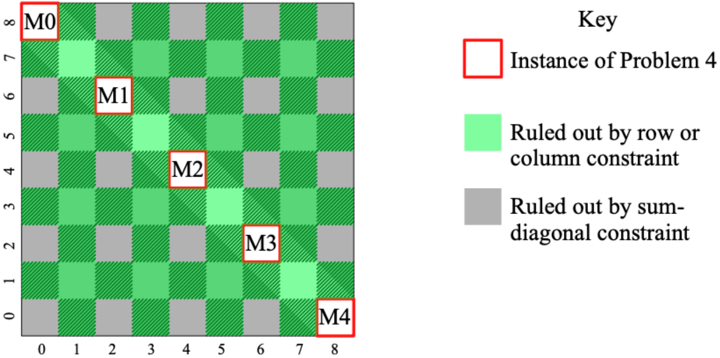

Figura: Figura 3

Definición 5 - **Reducción de 1-in-3-SAT al Problema 5**

Para un entero $\delta \geq 0$ y $G_a = \langle n_a, C_a, R_a, D_a^-, \{\} \rangle$, se define

$$G_a \oplus \delta_a \stackrel{def}{=} \langle n_a + \delta_a, \{c + \delta_a \mid c \in C_a\}, R_a, \{d + \delta_a \mid d \in D_a^-\}, \{\} \rangle$$

G^0 - **Un artilugio para una variable SAT**

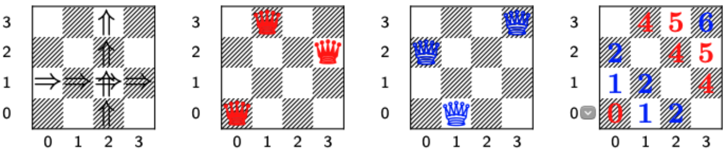

Figura: Figura 3

G^1, G^2, G^3 - Tres artilugios para una clausula 1-in-3-SAT

Se presentan tres artilugios asociados con la clausula $c = \{v_i, v_j, v_k\}$, los cuales son G_c^1, G_c^2, G_c^3.

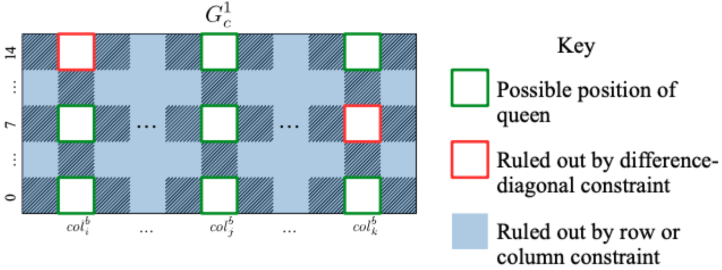

Figura: Figura 3

Prueba

Definición 6
Reducción de 1-in-3-SAT restringido al Problema 5

Se considera una instancia de $M_6 = \langle V, C \rangle$ de 1-in-3-SAT restringido. Para una clausula $c = \{v_i, v_j, v_k \in C\}$ se reescribe $M_{5,c}$ para la instancia del *Problema 5*, $M_{5,c} \stackrel{def}{=} \{G_i^0, G_j^0, G_k^0, G_c^1, G_c^2, G_c^3\}$. Para la reducción de la instancia entera M_6 se establece:

$$M_5 \stackrel{def}{=} \bigcup_{c \in C} M_{5,c}$$

Overview

Solucion Fuerza bruta

Logica

```
Input:
    Permutacion de posicion actual

Output:
    Falso si cumple
    Verdadero si no cumple

    Paso 1:
        a. Se generan todas las permutaciones posibles para el tablero
    Paso 2:
        a. Al restar los indices de dos posiciones , se puede verificar si
        estan en la misma diagonal
    Paso 3:
        a. Si se encuentra fuera de la diagonal se imprime la solucion
```

Figura: Figura 4

Soluciones

Solucion Fuerza bruta

Codigo usado

```
import itertools as it

def es_solucion(perm):
  for(i1, i2) in  it.combinations(range(len(perm)),2):
    if abs(i1 - i2) == abs(perm[i1] - perm[i2]):
      return False
  return True

for perm in it.permutations(range (8)):
  if es_solucion(perm):
    print (perm)
```

Figura: Figura 5

Solucion Backtracking

Logica

```
Input:
    Arreglo A de tamaño n*n
Output:

    Paso 1:
        a. Crear una matriz booleana de tamaño n*n
    Paso 2:
        a. Calcular las diagonales para saber si es valido
    Paso 3:
        a. Agregar valor posible al conjunto
    Paso 4:
        a. Eliminar si no cumple
    Paso 5:
        a. Finalmente se imprime los conjuntos resultados
   O(nM)
```

Figura: Figura 6

Solucion

Solucion Backtracking

Codigo usado

```
def puede_ser_solucion(perm):
  i = len(perm) - 1
  for j in range(i):
    if i - j == abs(perm[i] - perm[j]):
      return False
  return True
def backtracking(perm, n):
  if len(perm) == n:
    print(perm)
    exit()
  for k in range (n):
    if k not in perm:
      perm.append(k)
      if puede_ser_solucion(perm):
        backtracking(perm,n)
      perm.pop()

backtracking(perm = [], n = 20)
```

Figura: Figura 7

Solución

Solucion Quantum

Logica : *Se usa a cada posicion de cada atomo como representacion de una reina y este sera estimulado por un rayo de luz. El ´atomo estar´a bloqueado en movimiento en los ejes Y y Z. Luego el sistema pasa estado solido, por la interaccion de los ´atomos, el cual es interpretado como la solucion del problema. Todo este proceso se aprecia en la Figura 8.*

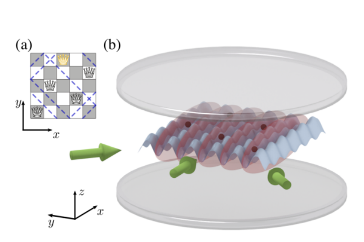

Figura: Figura 8

Conclusiones

Conclusiones

1. Hemos demostrado que el 'n-Queens Completion' es NP-Completo.
2. La demostración se basa en una reducción del problemas al 3-SAT (revisar secuencia de reducciones).
3. El problema presentado es uno de los más importantes dentro de la Inteligencia Artificial.
4. La solución propuesta usando backtracking es más eficiente que el algoritmo basado en fuerza bruta, además este representa perfectamente la definición de un algoritmo pseudo-polinomial.
5. Las computadoras cuánticas aún son muy inestables, pero se encuentran en una constante mejora; por lo que se estima que en dos décadas, las computadoras cuánticas sean estables y económicamente accesibles como las computadoras personales actuales.

Referencias

Complexity of n-Queens Completion

Ian P. Gent, Christopher Jefferson, Peter Nightingale.

Journal of Artificial Intelligence Research, vol. 59, pp. 815-848, 2017.

Fin