.UBAfiuba
FACULTAD DE INGENIERÍA

Teoría de Algoritmos
(75.29) Curso Buchwald - Genender

Trabajo Práctico 3
Problemas NP-Completos

25 de noviembre de 2023

Stephanie Izquierdo
104196

Tomás Macía
99248

1. Análisis del problema

1.1. Problema

La Selección Argentina de Lionel Scaloni fue campeona de la Copa del Mundo 2022. Para dicho evento armó una lista de 26 jugadores convocados, jugadores para distintas posiciones y el recambio necesario para afrontar los 7 partidos que lo llevaron a la consagración. Pero el armado de esta lista no es algo sencillo. En Argentina, país de tradición futbolera, todos opinamos sobre fútbol y creemos saber más que los propios protagonistas. Los periodistas de los distintos medios de prensa no se quedan atrás. Es por eso que ahora todos quieren ser parte y opinan qué jugadores deben formar parte del plantel.

Si bien Scaloni no se deja influenciar por los medios, por un partido amistoso va a hacer una excepción y elegirá al menos uno de los jugadores propuestos por los medios, pero quiere minimizar la cantidad de jugadores convocados por esta vía. Asumiendo que tenemos N medios/periodistas y que cada uno propone entre 1 y M jugadores, Scaloni convocará por lo menos a 1 jugador propuesto por cada uno de los N medios.

Por otro lado, Scaloni está bien acompañado por otros ídolos de la Selección Argentina. No es el único técnico que ocupa la silla de Campeón del Mundo, entre los que se encuentran Menotti y Bilardo. Justamente el doctor Carlos Salvador ya vivió situaciones similares y menciona:

Esto no es más que un caso particular del Hitting-Set Problem. Dado un conjunto A de n elementos y m subconjuntos $B_1, B_2, ..., B_m$ de A $(B_i \subseteq A\ \forall i), queremos el subconjunto C \subseteq A$ de menor tamaño tal que C tenga al menos 1 elemento de cada B_i (es decir $C \cap B_i \neq 0$). En nuestro caso A son los jugadores convocados, los B_i son los deseos de la prensa, y C es el conjunto de los jugadores que deberían jugar el partido amistoso si o si.

1.2. Entendiendo el problema

El problema en sí es sencillo de entender: tenemos distintos subconjuntos de jugadores propuestos y tenemos que elegir al menos un jugador de cada subconjunto de manera de minimizar la cantidad total de jugadores convocados.

Empecemos a analizar distintos casos que pueden presentarse. Si todos los medios mencionaran jugadores distintos, Scaloni tendría que elegir 1 jugador distinto por medio, de manera que extendido a los N medios, el DT convocaría N jugadores distintos porque es la única forma de contentarlos a todos. Este es un caso sencillo de entender. Por otro lado, si todos los jugadores eligieran un único jugador y todos eligieran al mismo (por ejemplo podrían querer convocar todos a un tal Lionel Messi), para que todos queden satisfechos solo existe una única posibilidad, convocar a Messi. Estos 2 casos mencionados representan las fronteras de nuestra solución: podemos convocar de 1 a N jugadores, siendo N la cantidad de periodistas que proponen jugadores.

También puede darse casos similares a los de la fronteras pero con pequeñas permutaciones. Por ejemplo si todos los medios opinan que debe jugar Messi, pero un periodista se confunde y en su lugar quiere convocar al Pulga Rodriguez, para contentar a N - 1 periodistas convocaremos a Messi y para contentar al restante tendremos que llamar a la Pulga equivocada, siendo 2 los jugadores. De esta forma y siguiendo la misma línea de razonamiento vemos como se puede dar cualquier rango de valores entre 1 y N de jugadores a convocar.

Habiendo explorado la cantidad de jugadores posibles que podemos convocar, también podemos ver que existe más de una combinación posible de jugadores para hacer feliz a la prensa. Supongamos que opinan 3 medios partidarios, dos de River y uno de Boca. Los de River piden por Enzo Fernández y por Julián Álvarez, mientras que el de Boca pide por Roncaglia. Para contentarlos a todos podríamos elegir tanto a Enzo Fernández como a Roncaglia, y es la cantidad mínima 2 del Ejemplo 1. Pero también podríamos elegir otra combinación de cantidad 2, cuyo conjunto estaría compuesto por Roncaglia y Julián Álvarez. Y no solo eso, sino que este mismo ejemplo nos permite visualizar un conjunto que no es el óptimo: Roncaglia, Enzo Fernández (o bien Julián Álvarez) y Jonatan Maidana. Hay 3 jugadores en este conjunto, y no es el óptimo porque vimos que pode-

mos contentar a todos los periodistas eligiendo sólo 2, dejando lugar a Scaloni a poder elegir más jugadores de acuerdo a su estilo de juego.

Roncaglia		
Enzo Fernández	Julián Álvarez	
Enzo Fernández	Julián Álvarez	Jonatan Maidana

Cuadro 1: Ejemplo 1

Este ejemplo no solo nos permite ver que existen distintos conjuntos que pueden resolver el problema de forma óptima y no óptima, sino que también nos ayuda a ver algo que estaba implícito en el problema. Los medios que menos jugadores proponen son los que nos "limitan"la solución final. En el ejemplo, el periodista de Boca que propone a Roncaglia, al proponer únicamente a un jugador, nos obliga a elegirlo. De esta forma únicamente si otro medio propone al mismo jugador disminuiría el tamaño del conjunto óptimo. Este "descubrimiento.es importante para que luego podamos definir una estrategia Greedy de resolución al problema, aunque veremos que no siempre resulta en la solución óptima.

1.3. Hitting-Set Problem

1.3.1. Probando que es NP

Para probar que el problema de Hitting-Set (C) está en NP, necesitamos demostrar que existe una verificación en tiempo polinomial. La verificación se realiza al comprobar si el conjunto C que se propone como solución realmente cubre todos los vértices del grafo.

Utilizaremos un grafo G para nuestra demostración. A representará el conjunto de vértices del grafo. B_i serán las aristas del grafo G, donde cada B_i esta compuesto por los dos vértices que conecta. Por último C sera un subconjunto de A.

Para verificar si C cubre todos los vértices del grafo, podemos seguir estos pasos: Recorrer todos los vértices del grafo G y verificar si cada uno de ellos está en el conjunto C. Si todos los vértices están en el conjunto C, entonces el conjunto C cubre todos los vértices del grafo G. En este caso, podemos devolver ẅerdaderoc̈omo respuesta de la verificación.

Cada uno de estos pasos se puede realizar en tiempo $\mathcal{O}(n+m)$, donde n es el número de vértices del grafo G y m es el número de aristas.

Por lo tanto, la verificación en tiempo polinomial existe, lo que significa que el problema de Hitting-Set está en NP.

1.3.2. Probando que es NP-completo

Para demostrar que Hitting Set es NP-completo realizaremos una reducción a partir del problema de Vertex Cover.

En el problema de Vertex Cover tenemos un grafo $G = (V, E)$ donde V es el conjunto de vértices del grafo y E es el conjunto de las aristas del grafo y VC es la solución. Definimos:

- A = V(G), es el conjunto de todos los vértices de G
- Bi = {u, v}, donde U y V son vértices de G.

Si VC es la solución Vertex cover del grafo G de tamaño k, esto implica que para cada arista B_i tanto el vértice u como el vértice v pertenece a VC. Por lo tanto, VC forma el hitting set porque todos los subconjuntos de vértices formarán una intersección con los vértices de VC.

Si C es un Hitting Set de A de tamaño k, dado que C interseca cada subconjunto de A, al menos uno de los vértices de cada arista debe pertenecer a la solución. Por lo tanto, abarca al menos un vértice para cada arista, formando así VC.

Entonces tenemos que G tiene un Vertex Cover de tamaño k si y solo si C tiene un Hitting Set de tamaño k y por ende $C \cap B_i \neq 0$

2. Algoritmos

2.1. Solución Greedy

Partiendo de la base que los periodistas que menos jugadores proponen son los que nos van a terminar limitando el conjunto final (porque tenemos que elegir si o si uno de ellos para satisfacer los requisitos del problema), primero deberíamos elegir los jugadores de estos periodistas y luego apuntar a elegir esos mismos jugadores en los siguientes subconjuntos, que deberían ser de mayor tamaño y por lo tanto tener mayor cantidad de jugadores para elegir. Si ordenamos los distintos subconjuntos por su tamaño de menor a mayor, entonces por cada medio de prensa vamos a ir eligiendo un jugador que satisfaga la condición de minimizar los jugadores a convocar por Scaloni. Veamos esto con un ejemplo

	Lionel Messi	
Lautaro Martinez	Julián Álvarez	
Colo Barco	Lucas Alario	Lionel Messi

Cuadro 2: Ejemplo 2 - Greedy

Para el primer periodista solo tenemos una opción, que es elegir a Lionel Messi. Esta opción es la que nos minimiza la cantidad de jugadores, ya que tenemos que elegir a un jugador y es el único que podemos elegir. Para el siguiente medio, tenemos dos posibilidades, o elegimos a Lautaro y nuestro subconjunto óptimo es de tamaño 2, o bien elegimos a Julián pero nuestro subconjunto óptimo tendría el mismo tamaño. En este paso nuestra estrategia para el algoritmo será seleccionar al primero que satisfaga la condición óptima, así que elegimos a Lautaro Martinez. El último medio nos propone convocar a 3 jugadores: dos de ellos son jugadores nuevos, que no están en nuestro subconjunto óptimo, por lo que si los elegimos nuestro subconjunto óptimo ya sería de tamaño 3, pero tenemos la posibilidad de convocar a Messi y así lograr que nuestro subconjunto sea Lionel Messi; Lautaro Martinez y sea de tamaño 2, es decir el óptimo para este ejemplo.

Pero esta estrategia no es la mejor, porque no siempre encuentra el óptimo para todos los casos. Si en el caso anterior, el último medio no hubiese propuesto a Lionel Messi y en su lugar prefiere a Julián Álvarez, como nuestro algoritmo eligió antes a Lautaro Martinez en lugar de Julián Álvarez (porque recordemos que el óptimo en ese segundo paso podía ser cualquiera de las dos opciones, y elegimos a Lautaro porque era la primera opción del subconjunto), no podríamos llegar por esta vía al óptimo que son 2 jugadores.

Una alternativa al algoritmo que solucionaría el caso anterior es ordenar cada subconjunto alfabéticamente. De esa manera, Julián Álvarez quedaría primero y lo elegiríamos por sobre Lautaro, y luego volveríamos a elegirlo para el tercer medio de prensa. Pero no es un algoritmo óptimo porque estaríamos en la misma situación: si el tercer periodista en lugar de proponer a Julián quisiera convocar a Lautaro, volveríamos a caer en un caso no óptimo (Ejemplo 3).

	Lionel Messi	
Julián Álvarez	Lautaro Martinez	
Lucas Alario	Colo Barco	Lautaro Martinez

Cuadro 3: Ejemplo 2 - Greedy

El dificultad de esta estrategia o cualquier estrategia Greedy para este tipo de problema radica en que no conocemos de antemano todos los valores posibles que pueden venir después la decisión actual de elegir a un jugador por sobre otro. Justamente un algoritmo Greedy en cada paso elige la solución óptima para ese momento, con el objetivo de llegar a la solución óptima general, que como vimos no siempre sucede. En cada paso del algoritmo tratamos de elegir al jugador que mayor restricciones nos plantea de cara a la solución final, y luego vamos eligiendo el óptimo en base a esas restricciones iniciales, asumiendo que al ser conjuntos progresivamente mayores, mayor probabilidad de encontrar a los jugadores previamente visitados.

```python
def greedy_jugadores(jugadores_matrix):
    jugadores = jugadores_matrix.copy()
    jugadores.sort(key=lambda x: len(x))

    solucion = set()

    for propuestas in jugadores:
        optimo_local = propuestas[0]
        for jugador in propuestas:
            if jugador in solucion:
                optimo_local = jugador
                break
        solucion.add(optimo_local)

    return solucion
```

2.2. Solución Backtracking

TODO

2.3. Solución Programación Lineal

El objetivo es seleccionar un conjunto mínimo de jugadores que aseguren que, dentro de una preferencia, todos los jugadores tengan al menos otro jugador que ya fue seleccionado para la solución.

Para esto creamos las variables binarias jugadoresVars donde cada variable representa a un jugador y toma valores entre 0 y 1, donde 1 indica que el jugador está en el conjunto óptimo.

Para cada preferencia de jugadores se añade una restricción al problema. Por cada jugador agregamos la variable indicadora y luego restringimos a que esta suma sea al menos 1 y así nos aseguramos que, para cada preferencia, haya al menos un jugador en el conjunto óptimo.

La función objetivo entonces resulta en minimizar la suma de todas las variables, lo que significa que queremos minimizar el número de jugadores en el conjunto óptimo.

```python
def jugadores_optimos_programacion_lineal(jugadores_segun_preferencias, todos_los_jugadores):
    problema = LpProblem("jugadores_optimos", LpMinimize)
    jugadores_vars = {jugador: LpVariable(f"x_{jugador}", 0, 1, LpBinary) for jugador in todos_los_jugadores}
    problema += lpSum(jugadores_vars[jugador] for jugador in todos_los_jugadores)
    for preferencia in jugadores_segun_preferencias:
        problema += lpSum(jugadores_vars[jugador] * (jugador in preferencia) for jugador in todos_los_jugadores) >= 1
    problema.solve()
```

2.4. Solución Programación Lineal con Valores Reales

TODO

3. Ejemplos de ejecución

3.1. Ejemplos cátedra

```python
for f in all_files:
    jugadores = read_file(f"samples/{f}")

    print(f"{f}:\n")

    for algoritmo, name in ALL_ALGORITHMS:
```

```
7         print(f"Algoritmo {name}")
8         solucion = algoritmo(jugadores)
9         if not solucion: continue # Programacion Lineal
10        print(f"Jugadores minimos {len(solucion)}: {solucion}")
```

El bloque de código descripto ejecuta todos los ejemplos de la cátedra y devuelve los siguientes resultados:

3.1.1. Algoritmo Greedy

5.txt:

Jugadores mínimos 3: {'Colo Barco', 'Messi', "Barcon't"}

7.txt:

Jugadores mínimos 4: {'Colo Barco', 'Chiquito Romero', "Barcon't", 'Armani'}

10_pocos.txt:

Jugadores mínimos 4: {'Colo Barco', 'Chiquito Romero', 'Casco', 'Colidio'}

10_todos.txt:

Jugadores mínimos 10: {'Perrone', 'Paredes', 'Guido Rodriguez', 'Palacios', 'Cuti', 'Garnacho', 'Dibu', 'Lautaro', 'Messi', 'Molina'}

10_varios.txt:

Jugadores minimos 8: {"Barcon't", 'Di Maria', 'Riquelme', 'Beltran', 'Soule', 'Pezzella', 'Casco', 'Colidio'}

15.txt:

Jugadores minimos 5: {'Colo Barco', 'Palermo', 'Luka Romero', 'Dybala', 'Chiquito Romero'}

20.txt:

Jugadores minimos 7: {'Gallardo', "Barcon't", 'Colo Barco', 'Riquelme', 'Di Maria', 'Buendia', 'El fantasma de la B'}

50.txt:

Jugadores minimos 11: {'Gallardo', 'Colo Barco', 'Pity Martinez', 'Soule', 'Beltran', 'Palermo', 'Tucu Pereyra', 'Langoni', 'Casco', 'Buonanotte', 'Messi'}

75.txt:

Jugadores minimos 12: {'Simeone', "Barcon't", 'Colo Barco', 'Riquelme', 'Pezzella', 'Pity Martinez', 'Beltran', 'Dybala', 'Changuito Zeballos', 'Tucu Pereyra', 'Casco', 'Dibu'}

100.txt:

Jugadores minimos 12: {'Armani', "Barcon't", 'Ogro Fabianni', 'Colo Barco', 'Di Maria', 'Pezzella', 'Palermo', 'Dybala', 'Burrito Ortega', 'Changuito Zeballos', 'Colidio', 'El fantasma de la B'}

200.txt:

Jugadores minimos 13: {'Gallardo', 'Colo Barco', 'Riquelme', 'Pezzella', 'Di Maria', 'Palermo', 'Dybala', 'Burrito Ortega', 'Buendia', 'Casco', 'Dibu', 'Wachoffisde Abila', 'Messi'}

3.1.2. Algoritmo Backtracking

5.txt:

Jugadores mínimos 2: {'Casco', 'Cuti Romero'}

7.txt:

Jugadores mínimos 2: {'Mauro Zarate', 'Pezzella'}

10_pocos.txt:

Jugadores mínimos 3: {'Casco', 'Di Maria', 'Chiquito Romero'}

10_todos.txt:

Jugadores mínimos 10: {'Medina', 'Lo Celso', 'Armani', 'Tagliafico', 'Nico González', 'Almada', 'Correa', 'Otamendi', 'De Paul', 'Messi'}

10_varios.txt:

Jugadores mínimos 6: {'Beltran', 'Palermo', 'Changuito Zeballos', 'Wachoffisde Abila', 'Dibu', 'Dybala'}

15.txt:

Jugadores mínimos 4: {'Palermo', 'Luka Romero', 'Chiquito Romero', 'Dybala'}

20.txt:

Jugadores mínimos 5: {'Mauro Zarate', 'Riquelme', 'El fantasma de la B', 'Ogro Fabianni', "Barcon't"}

TODO

3.1.3. Algoritmo Programación Lineal

	5	7	10_p	10_t	10_v	15	20	50	75	100	200
Objective value	2.00	2.00	3.00	10.00	6.00	4.00	5.00	6.00	8.00	9.00	9.00
Nodes	0	0	0	0	0	0	0	0	4	12	36
Iterations	0	0	0	0	0	0	0	0	3681	5014	10744
Time (CPU secs)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.61	0.98	0.55
Time (Wallclock secs)	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.63	1.02	0.56

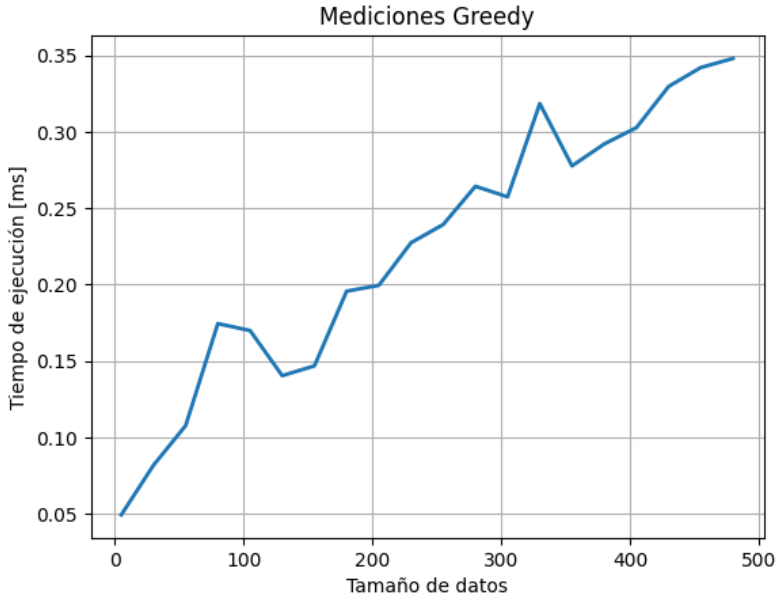
4. Mediciones

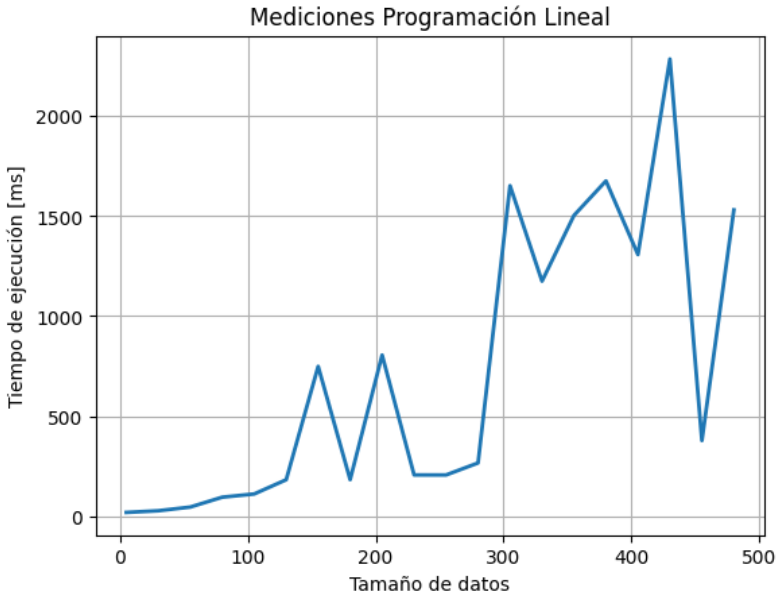
Para poder hacer las mediciones tuvimos que generar sets de datos aleatorios. La estrategia para poder hacer esto fue generar distintos números en un rango específico, donde cada número representa a un jugador distinto. En todas nuestras pruebas, el número máximo de jugadores distintos no cambió, ya que los jugadores iban desde 1 a 50.

Por otro lado, como cada periodista propone una cantidad de 1 a m jugadores, nuestro m fue de 8, teniendo la posibilidad de ajustarlo y probar con distintas combinaciones.

Por último, la cantidad de periodistas que proponen jugadores fue un número variable entre 5 y 500, con incrementos de a 25. Es decir se calculó con 5, 30, 55, 70.... 470, 495 listas de jugadores, y se midió únicamente el tiempo de ejecución del algoritmo y no el tiempo de generación de sets de datos aleatorios.

Para todos los algoritmos se utilizó el mismo set de datos para cada tamaño de N periodistas.





TODO

5. Conclusiones

TODO