DETERMINAR RUTA EN UN MAPA A PARTIR DE AUTÓMATAS DETERMINISTAS

Universidad Industrial de Santander

Presentado por:

Roger Fuentes 2221890
Camilo Jaimes 2221882
Camilo Carvajal 2190187

INTRODUCCIÓN

- Optimización de rutas urbanas mediante el modelamiento de autómatas finitos deterministas (DFA).
- Se representan las intersecciones de un barrio en Chicago como estados de un DFA, con transiciones que corresponden a las direcciones de movimiento (izquierda, derecha, arriba, abajo). Este enfoque innovador facilita el análisis y la resolución del problema sin necesidad de algoritmos complejos para hallar caminos cortos (usualmente con grafos).

OBJETIVOS

- El conocimiento actual sobre optimización de rutas está bien desarrollado, pero la aplicación de DFA en este contexto es limitada.

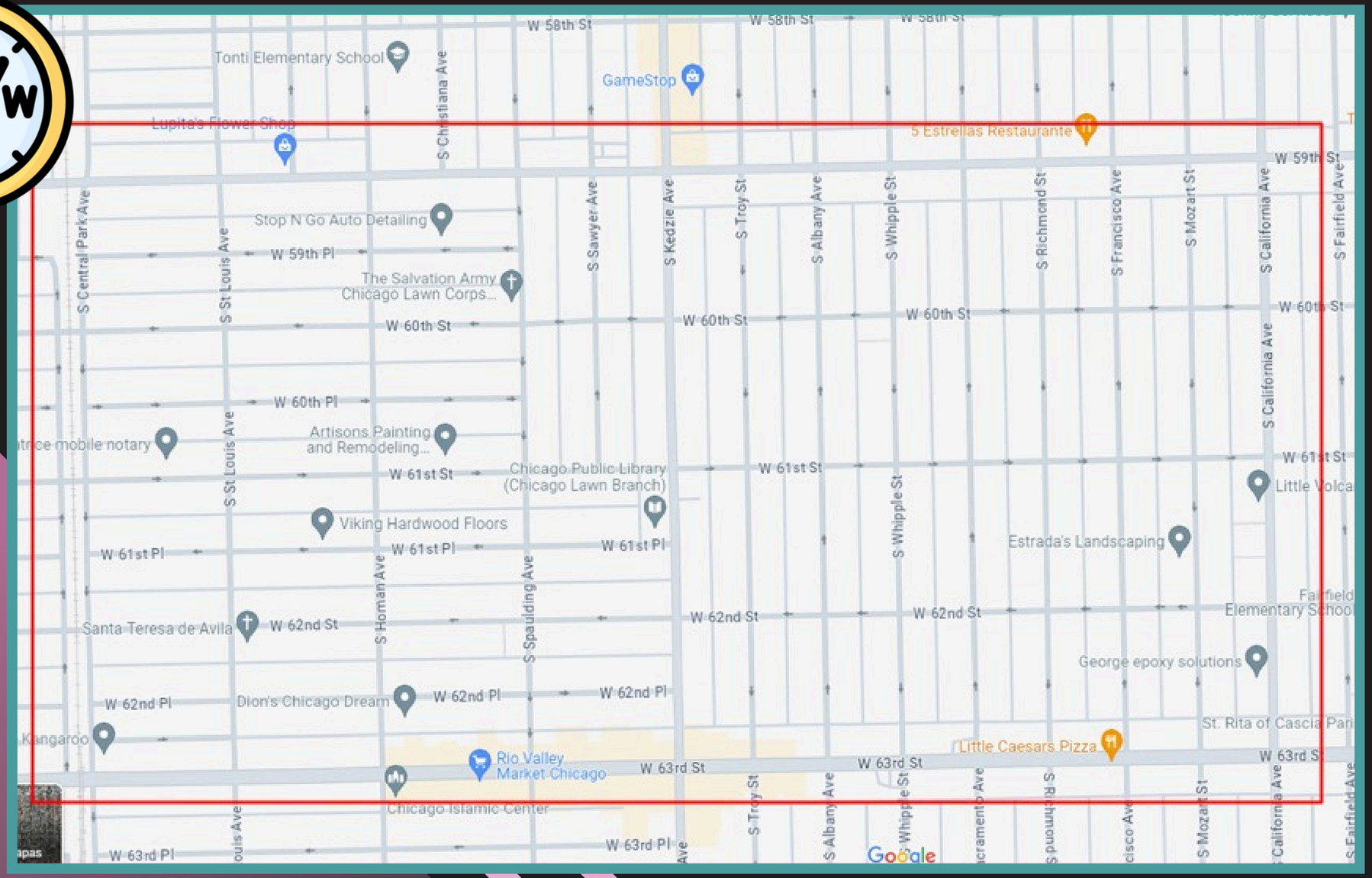
- El objetivo es desarrollar un modelo DFA que determine la ruta más corta y las posibles rutas entre dos puntos especificados por el usuario, demostrando su eficacia en comparación con métodos tradicionales.

Definición formal

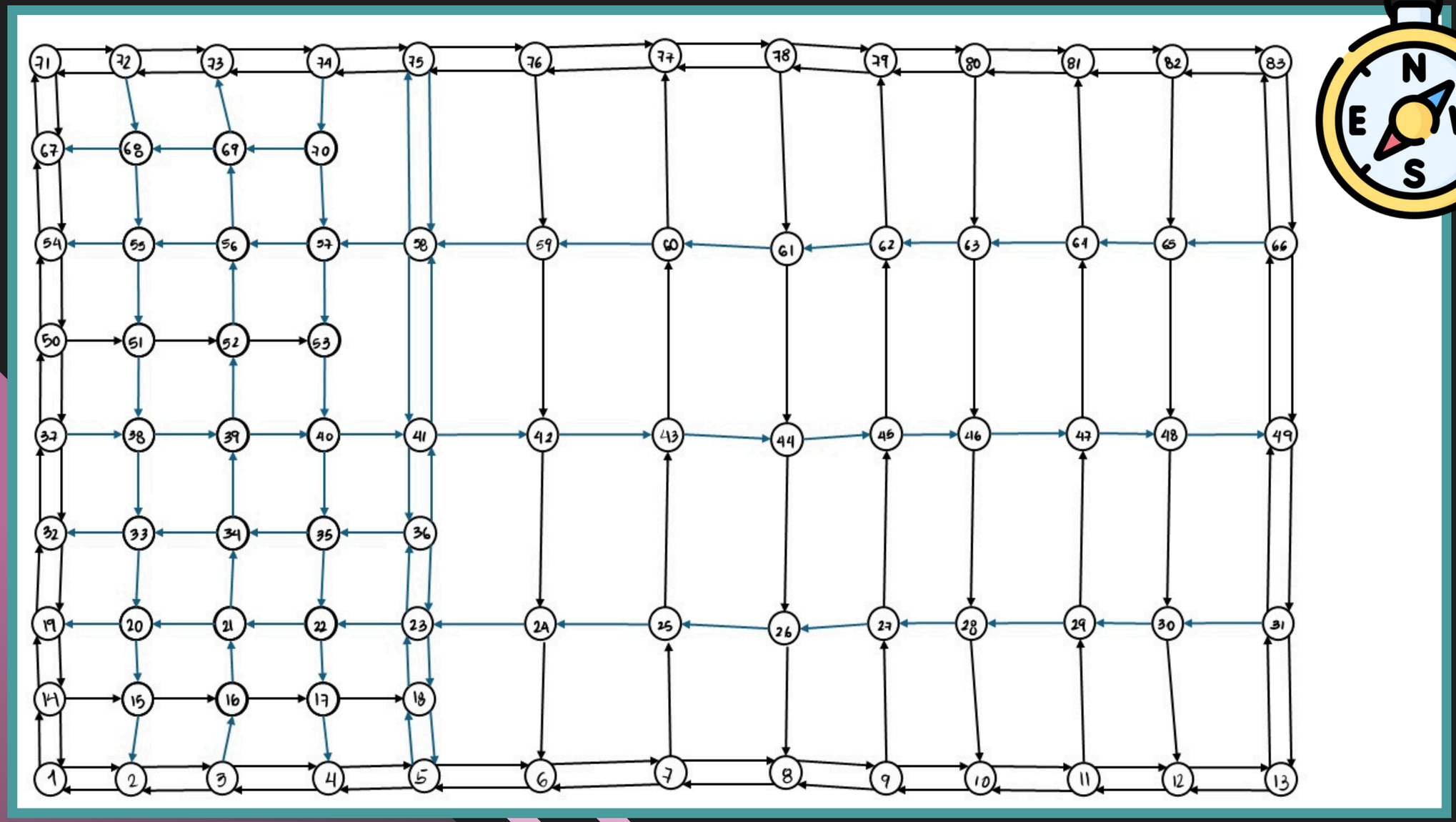
$$A = (Q, \Sigma, \delta, q0, F)$$

{left, up, down, right}	Definido por el usuario	Definido por el usuario	Son los puntos de ínteres del mapa	Definidas por las direcciones que conectan los estados o P.I
Input symsbols Σ	**Estados finales F = {}**	**Estado inicial q0**	**Estados del automata Q (P.I)**	**Transiciones δ**

Mapa del barrio de Chicago modelado



Modelo utilizando grafos



Implementación

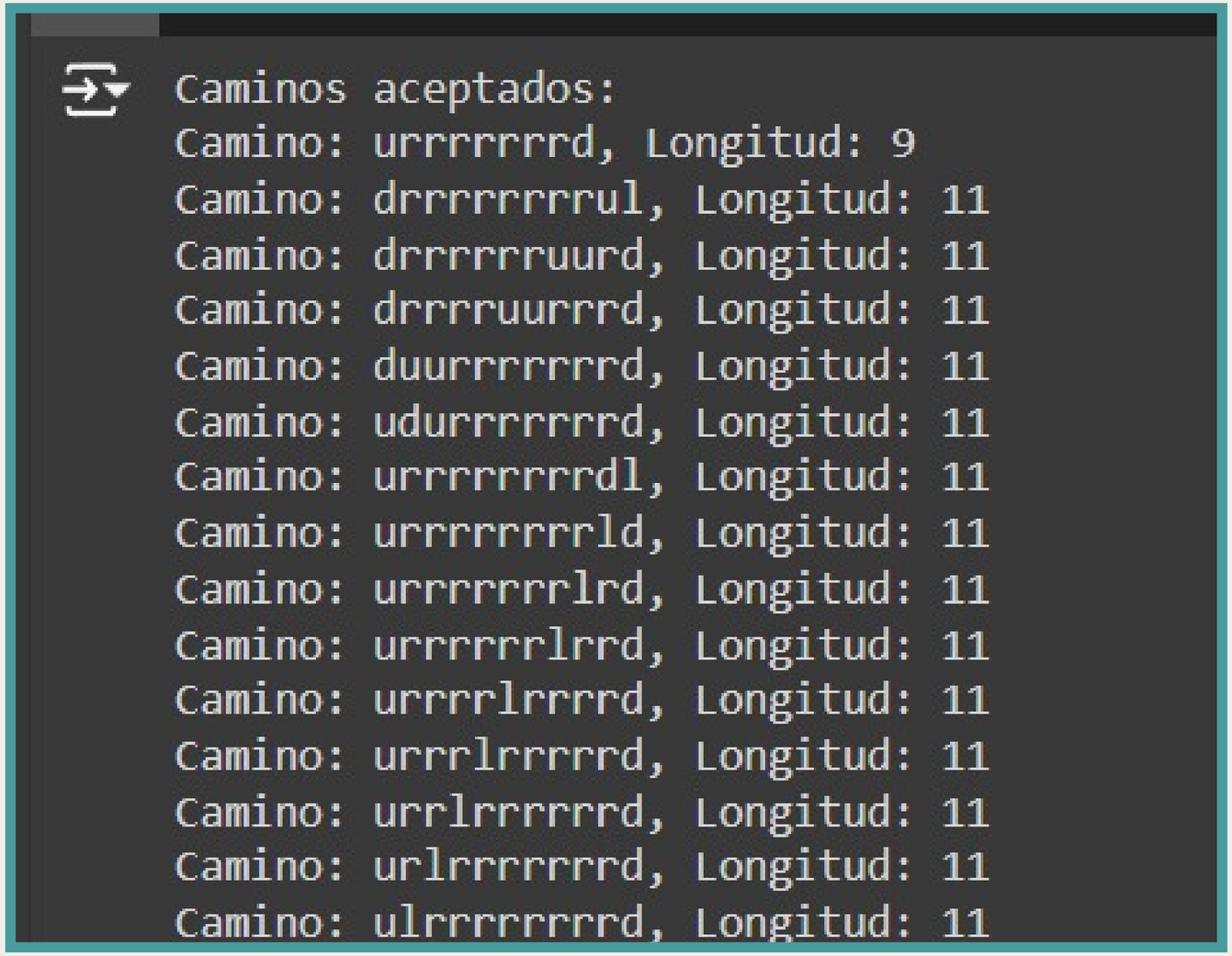
Implementación Caso de uso: Chicago

```
[ ]  puntos_de_interes = [str(i) for i in range(1, 84)]
     aristas = [
         ('1', '2', 'r'),  # 'r' para derecha
         ('1', '14', 'u'),  # 'u' para arriba
         ('2', '3', 'r'),  # 'd' para abajo
         ('2', '1', 'l'),  # 'l' para izq
         ('3', '2', 'l'),
         ('3', '4', 'r'),
         ('3', '16', 'u'),
         ('4', '5', 'r'),
         ('4', '3', 'l'),
         ('5', '6', 'r'),
         ('5', '4', 'l'),
         ('5', '18', 'u'),
```

Implementación

```python
def crear_automata(puntos_de_interes, aristas, inicio, fin):
    estados = set(puntos_de_interes)
    input_symbols = {'l', 'u', 'r', 'd'}  # Alfabeto, son las direcciones posibles

    transitions = {state: {symbol: None for symbol in input_symbols} for state in estados}

    for (origen, destino, direccion) in aristas:
        transitions[origen][direccion] = destino

    dfa_transitions = {}
    for state, paths in transitions.items():
        dfa_transitions[state] = {}
        for symbol, next_state in paths.items():
            dfa_transitions[state][symbol] = next_state if next_state is not None else state

    #Se crea el autómata de la manera vista en clase
    dfa = DFA(
        states=estados,
        input_symbols=input_symbols,
        transitions=dfa_transitions,
        initial_state=inicio,
        final_states={fin}
    )

    return dfa
```

Salida


Caminos aceptados:
Camino: urrrrrrrd, Longitud: 9
Camino: drrrrrrrrrul, Longitud: 11
Camino: drrrrrrruurd, Longitud: 11
Camino: drrrruurrrd, Longitud: 11
Camino: duurrrrrrrrd, Longitud: 11
Camino: udurrrrrrrrd, Longitud: 11
Camino: urrrrrrrrrdl, Longitud: 11
Camino: urrrrrrrrrld, Longitud: 11
Camino: urrrrrrrrlrd, Longitud: 11
Camino: urrrrrrrlrrd, Longitud: 11
Camino: urrrrlrrrrd, Longitud: 11
Camino: urrrlrrrrrd, Longitud: 11
Camino: urrlrrrrrrd, Longitud: 11
Camino: urlrrrrrrrd, Longitud: 11
Camino: ulrrrrrrrrd, Longitud: 11

MUCHAS GRACIAS