Clase Virtual 2021-03-15

- Video 1
- Video 2

Recapitulación Busquedas convensionales

- Search problem:
  - States (configurations of the world)
  - Actions and costs
  - Successor function (world dynamics)
  - Start state and goal test
- Search tree:
  - Nodes: represent plans for reaching states
  - Plans have costs (sum of action costs)
- Search algorithm:
  - Systematically builds a search tree
  - Chooses an ordering of the fringe (unexplored nodes)
  - Optimal: finds least-cost plans

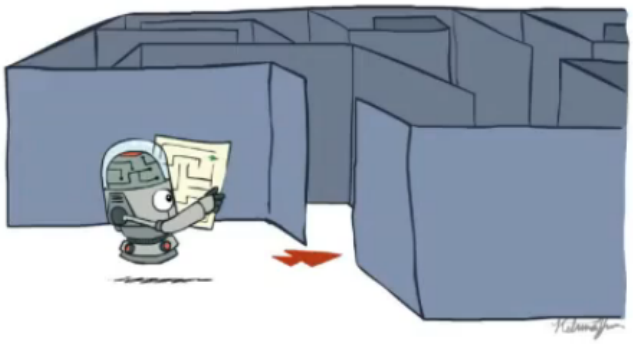


Figure 1: Recapitulación

- El árbol es una estructura que se crea a medida que avanza el código
- Cada nodo sabe cuál es su padre
- Cada nodo representa un plan, una secuencia de pasos con costos

Busquedas Informadas

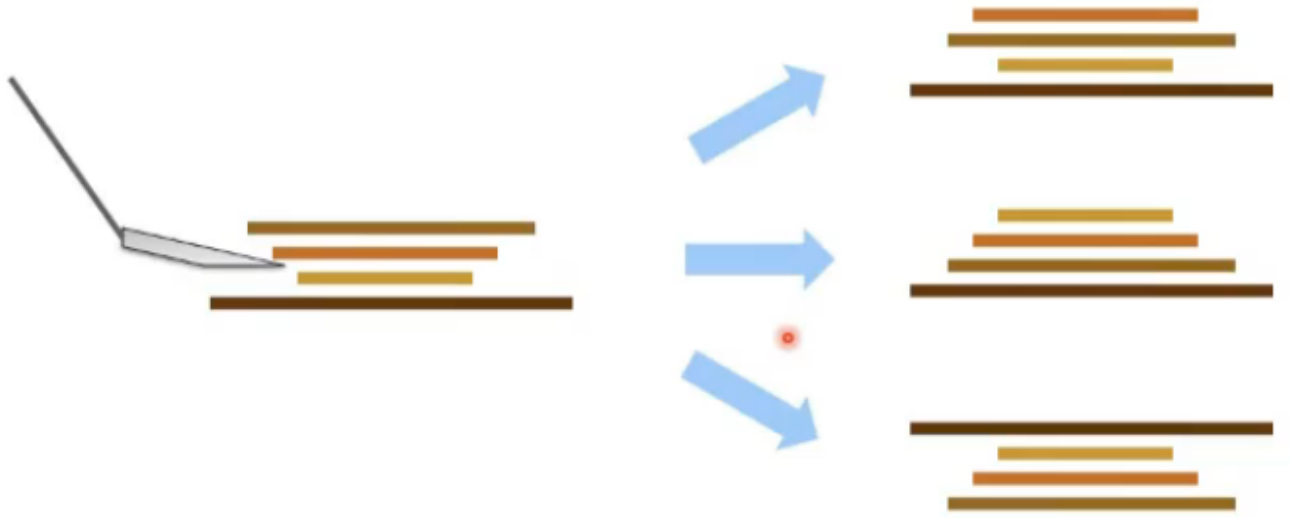
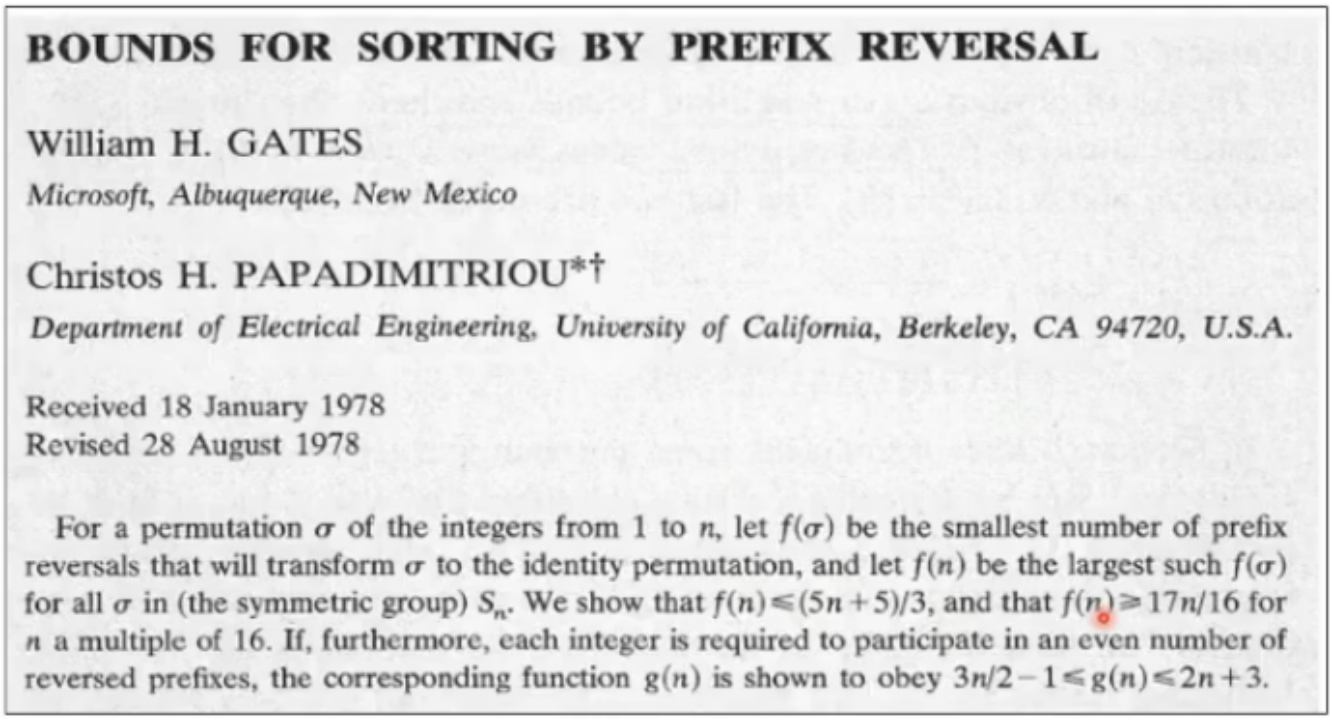


Figure 2: pancakes

- n pancakes, n-1 acciones posibles.

- n pancakes, se pueden organizar de n!
- Costo: número de pancakes bolteados
- objetivo: secuencia de acciones de costo mínimo



BOUNDS FOR SORTING BY PREFIX REVERSAL

William H. GATES
Microsoft, Albuquerque, New Mexico

Christos H. PAPADIMITRIOU*†
Department of Electrical Engineering, University of California, Berkeley, CA 94720, U.S.A.

Received 18 January 1978
Revised 28 August 1978

For a permutation σ of the integers from 1 to n, let $f(\sigma)$ be the smallest number of prefix reversals that will transform σ to the identity permutation, and let $f(n)$ be the largest such $f(\sigma)$ for all σ in (the symmetric group) S_n. We show that $f(n) \leqslant (5n+5)/3$, and that $f(n) \geqslant 17n/16$ for n a multiple of 16. If, furthermore, each integer is required to participate in an even number of reversed prefixes, the corresponding function $g(n)$ is shown to obey $3n/2 - 1 \leqslant g(n) \leqslant 2n+3$.

Figure 3: Gates

- cota superior: (5n + 5)/3
- Cota inferior: 17n/16
- La búsueda de costo uniforme, busca en todas las direcciones

Heurística - h(x)

- Es una función que estima que tan cerca estamos de un estado a un estado objetivo
- **Distancia euclidiana - L2:** hipotenusa, no es exacta para el pacman porque no corre en diagonal.
- **Distancia de manhatan - L1:** suma del tamaño de aristas
- Es fácil de calcular
- El número del pancake más largo, fuera de lugar
- la heurística cambia a partir del estado inicial.

Greedy Search

- Algoritmo voraz, seguir la heurística óptima para cada paso.
- Se va a lo que tiene más cerca
- h(x): distancia lineal
- La cola de prioridad está ordenada con base en la función heurística
- La búsqueda voráz, encuentra la solución, pero no siempre es la más óptima

A*

- Combinanción:
  - **Búsqueda de costo uniforme UCS:** lento y garantiza el óptimo, tiene en cuenta costos reales. g(n)
  - **Voráz - Greedy:** No garantiza el óptmo, solo tiene en cuenta la heurística. h(n)
- f(n) = g(n) + h(n)

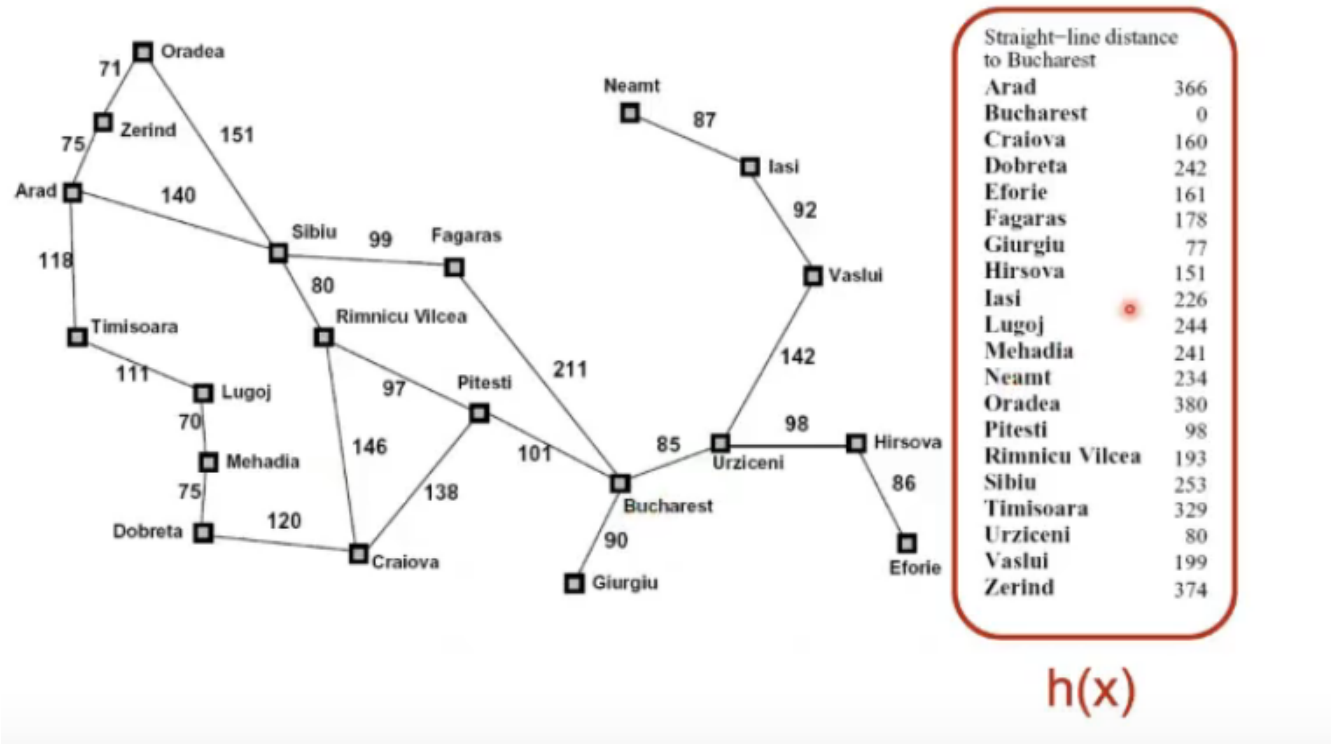


Figure 4: camino arad

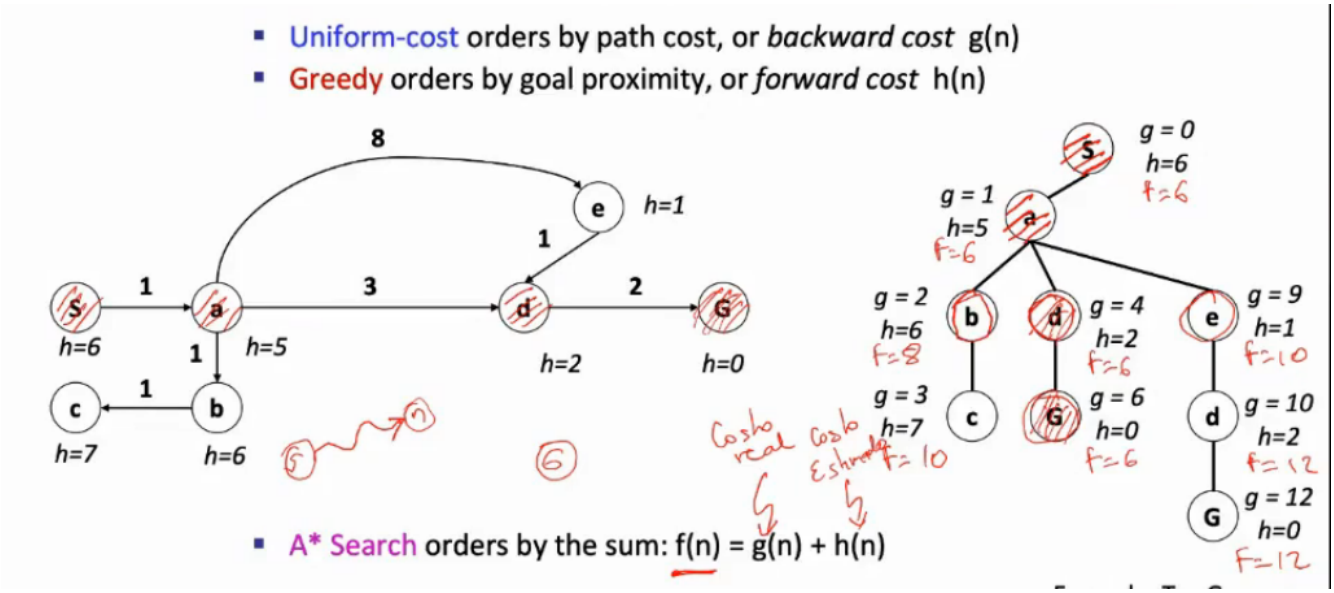


Figure 5: A* ejemplo

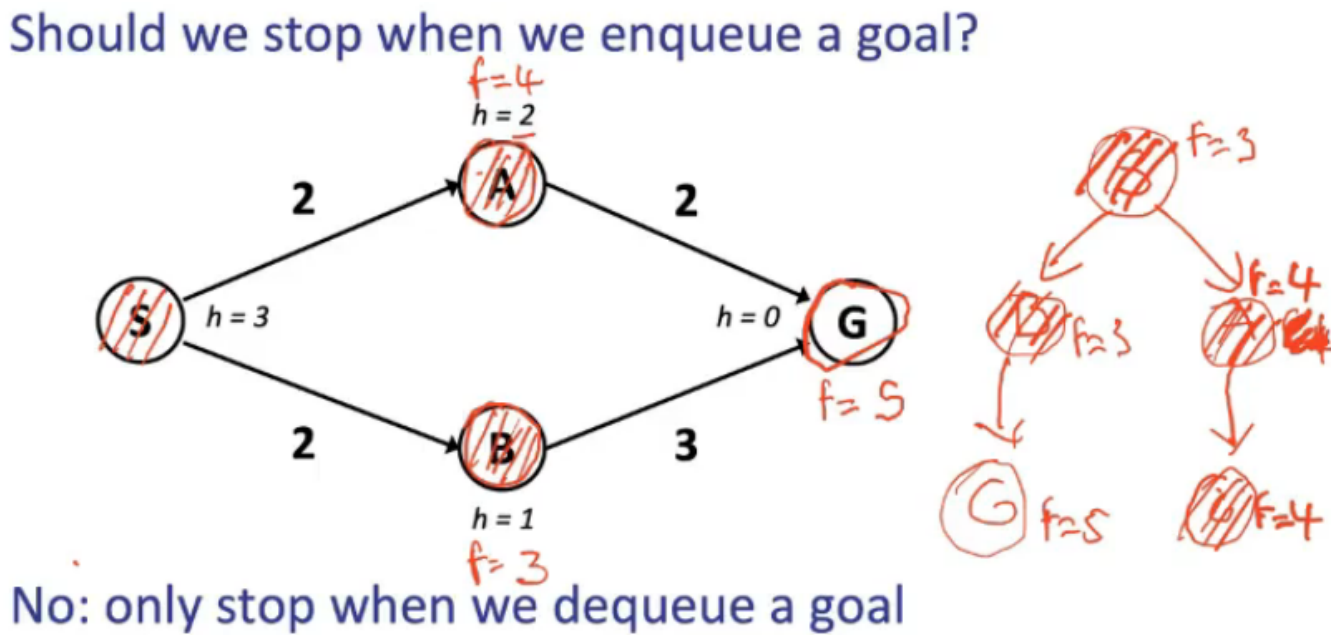


Figure 6: terminar al encontrar nodo objetivo?

- Termina cuando saca un nodo del **fringe** y ese nodo es **objetivo**
- Es óptimo sólo si las heurísticas **NO sobrestima**
- **inadmisible:** buenos caminos quedan en fringe por la mala heurística
- **admisible:** los malos caminos son retrasados por la heurística
- h*(n) es el costo real que generalmente no se sabe
- Los antepasados de A, se expanden primero que B.
- A va a ir al fringe y va a ser sacado primero que B

Segunda parte

- Procesamiento de voz: modelos probabilisticos

¿Cómo crear las heurísticas adminisbles?

- pensar en versiones más simples (relajadas) del problema.
- mapa -> viajar en avión, pacman -> atraviesa paredes
- las heurísticas inadmisibles podrían llegar a ser úties, no la optima pero sí una solución
- Ejemplo Puzzle
  - Cantidad de estados: 9!
  - acciones: mover el espacio en blanco
    - Arriba, abajo, izquierda y derecha
  - sucesores del estado inicial: 4
  - Heuristicas
    - Número de fichas fuera de lugar
      - Admisible: sí
      - Version relajada: sacar las fichas y ponerlas en su posición
    - contar los pasos de manhatan para cada pieza del puzzle y sumar el total
    - ambas heurísticas subestiman, sin embargo la 2 es mejor ya que está más cerca del costo real que h1
- Si se usa como heurística el costo real
  - el número de nodos son exactamente los mínimos
  - Sería admisible

- A heuristic h is *admissible* (optimistic) if:

$$0 \leq h(n) \leq h^*(n)$$

where $h^*(n)$ is the true cost to a nearest goal

- Examples:

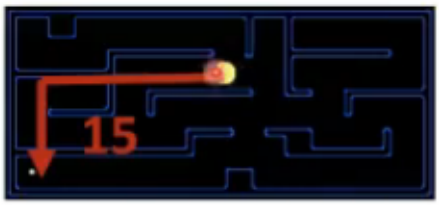
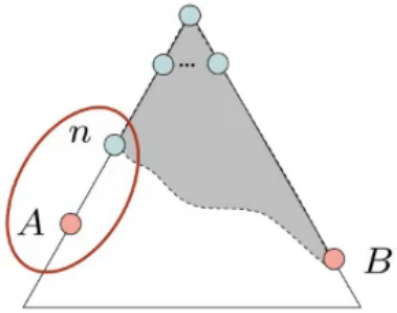
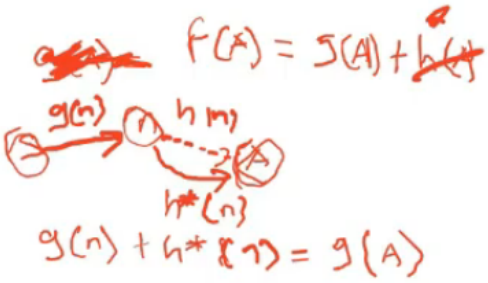


Figure 7: h admisible

Proof:

- Imagine B is on the fringe
- Some ancestor *n* of A is on the fringe, too (maybe A!)
- Claim: *n* will be expanded before B
  1. f(n) is less or equal to f(A)





$f(n) = g(n) + h(n)$ Definition of f-cost

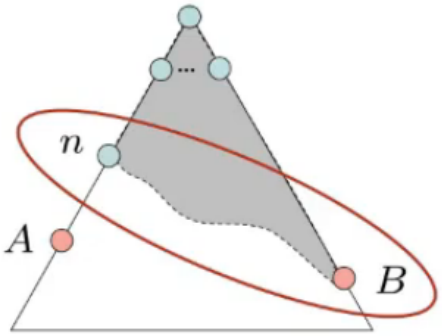
$f(n) \leq g(A)$ Admissibility of h

$g(A) = f(A)$ h = 0 at a goal

Figure 8: demostración optimalidad A*

Proof:

- Imagine B is on the fringe
- Some ancestor *n* of A is on the fringe, too (maybe A!)
- Claim: *n* will be expanded before B
  1. f(n) is less or equal to f(A)
  2. f(A) is less than f(B)
  3. *n* expands before B



$$f(n) \leq f(A) < f(B)$$

Figure 9: demostración optimalidad A*

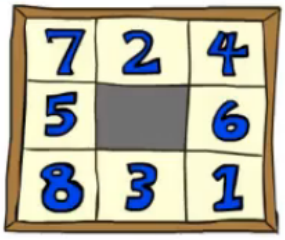
- Video games
- Pathing / routing problems
- Resource planning problems
- Robot motion planning
- Language analysis
- Machine translation
- Speech recognition
- ...

Figure 10: aplicaciones

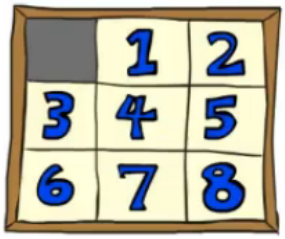
	Average nodes expanded when the optimal path has...		
	...4 steps	...8 steps	...12 steps
UCS	112	6,300	3.6×10^6
TILES	13	39	227

Figure 11: comparación UCS y esta heurística

- What if we had an easier 8-puzzle where any tile could slide any direction at any time, ignoring other tiles?
- Total *Manhattan* distance
- Why is it admissible?
- h(start) = 3 + 1 + 2 + ... = 18



Start State



Goal State

	Average nodes expanded when the optimal path has...		
	...4 steps	...8 steps	...12 steps
TILES	13	39	227
MANHATTAN	12	25	73

Figure 12: Comparación h1 y h2

  - Ahorra nodos expandidos
  - Es difícil de calcular

Heurísticas matemáticamente

- retículo: orden parcial de las cosas

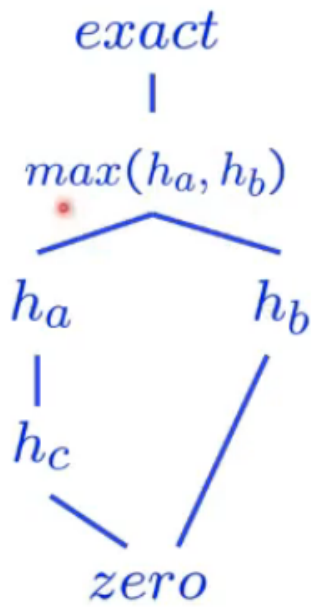
- Dominance: $h_a \geq h_c$ if

$$\forall n : h_a(n) \geq h_c(n)$$

- Heuristics form a semi-lattice:
  - Max of admissible heuristics is admissible

$$h(n) = max(h_a(n), h_b(n))$$

- Trivial heuristics
  - Bottom of lattice is the zero heuristic (what does this give us?)
  - Top of lattice is the exact heuristic



- comparar heurísticas
  - Si para todos los estados ha es siempre mayor que hc, ha > hc
  - ha y hb no son comparables, esto porque puede existir un nodos donde no se cumpla que una siempre es mayor que la otra
- la heurística exacta siempre va a estar por encima de las admisibles
- la heurística que está por debajo de todas es la cero