Busqueda Heuristica

Algo de historia

- George Polya, 1945
- “El estudio de los metodos y reglas del descubrimiento e invension”
- Eurisco: griego: “I discover”
- Archimedes: Eureka: griego : “I have found”

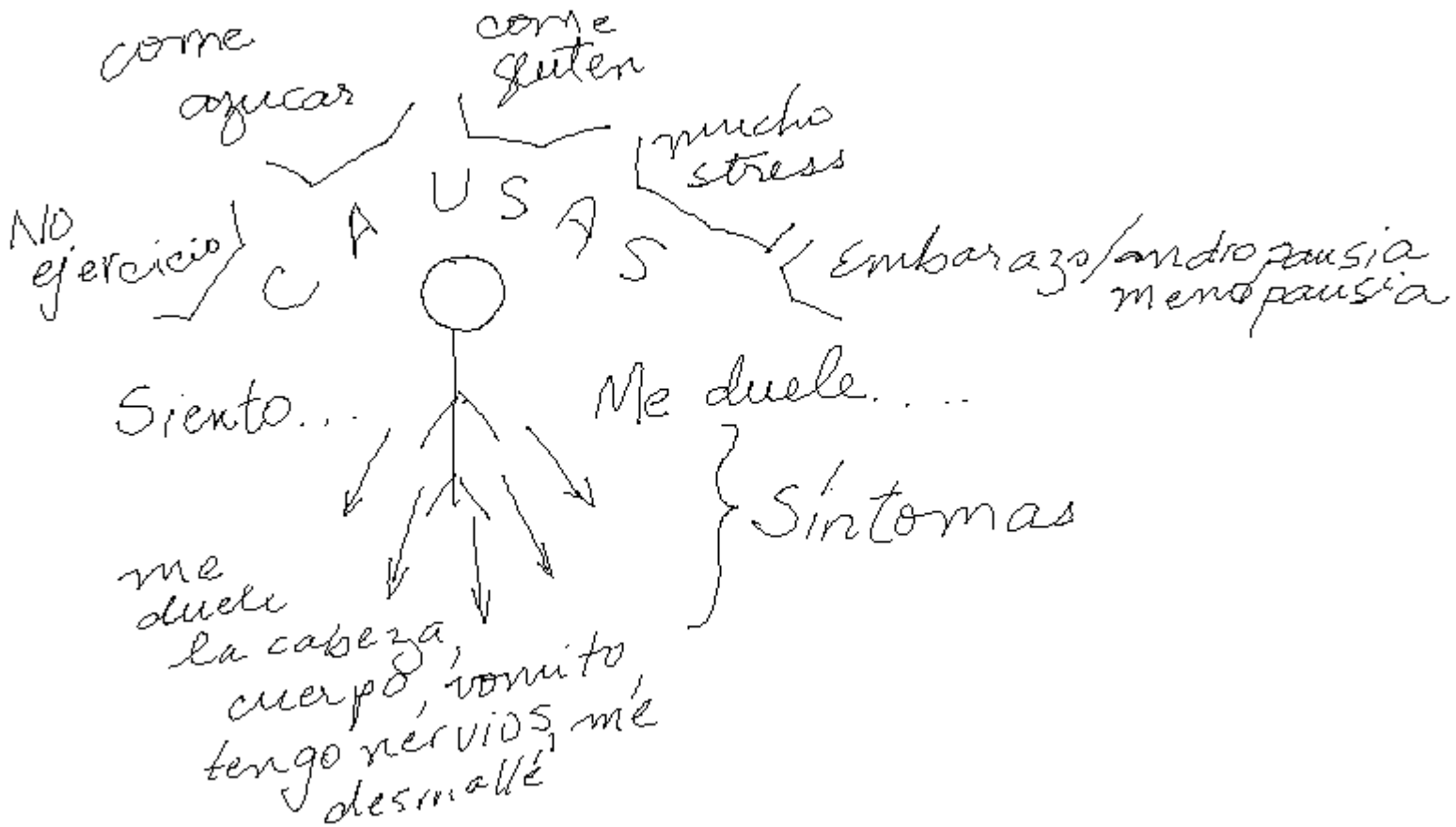
Que es heuristica?

- En la busqueda de espacios de estado, las heuristicas son reglas para escoger esas ramas en un espacio de estado que son las que mas aceptablemente me llevaran a la solucion aceptable de un problema.

Cuándo se usan las heuristicas?

1. Un problema puede no tener una solucion exacta por ambiguedades.

Ejemplo 1:
Diagnostico medico:
Problema puede no tener una solucion exacta

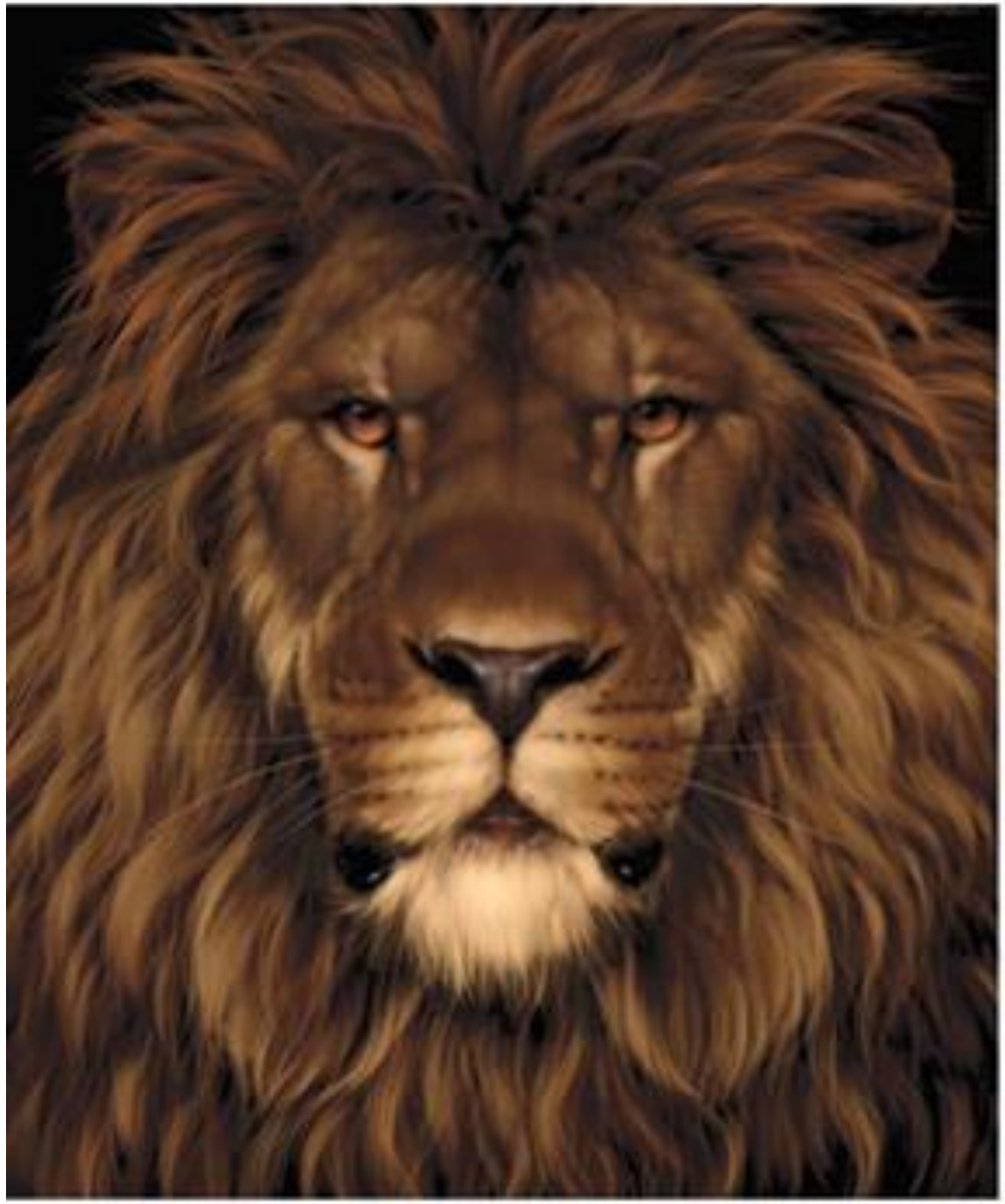


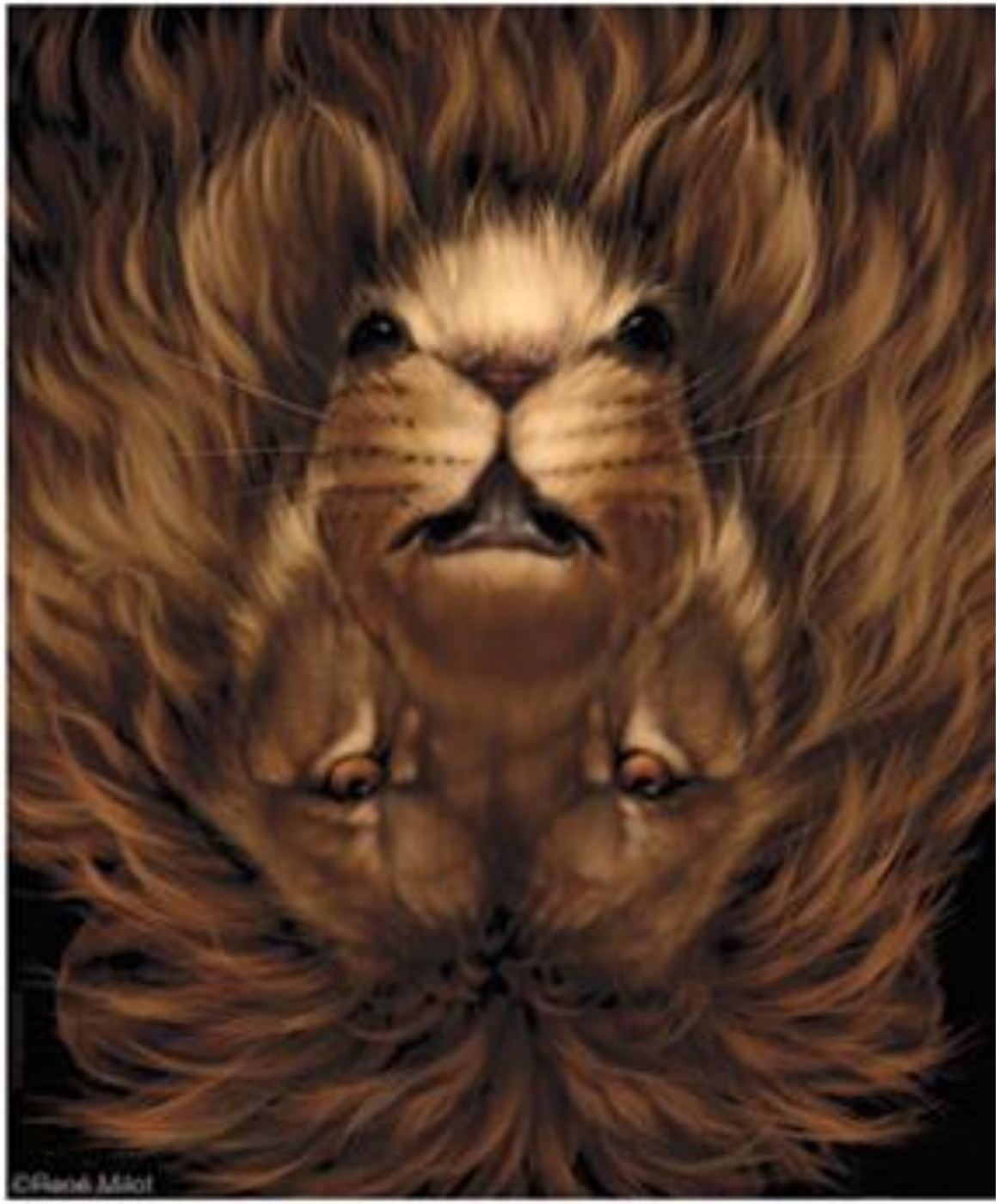
Qué usa el doctor para escoger el dianostico?

- Usa heuristicas (que le ayuden a hacer su espacio de busqueda mas pequeno)
- Estas heuristicas estan dadas por reglas generadas a traves de la experiencia
- Genera un primer diagnostico.
- Formula un plan de tratamiento
- Pero y si el tratamiento no funciona, entonces?

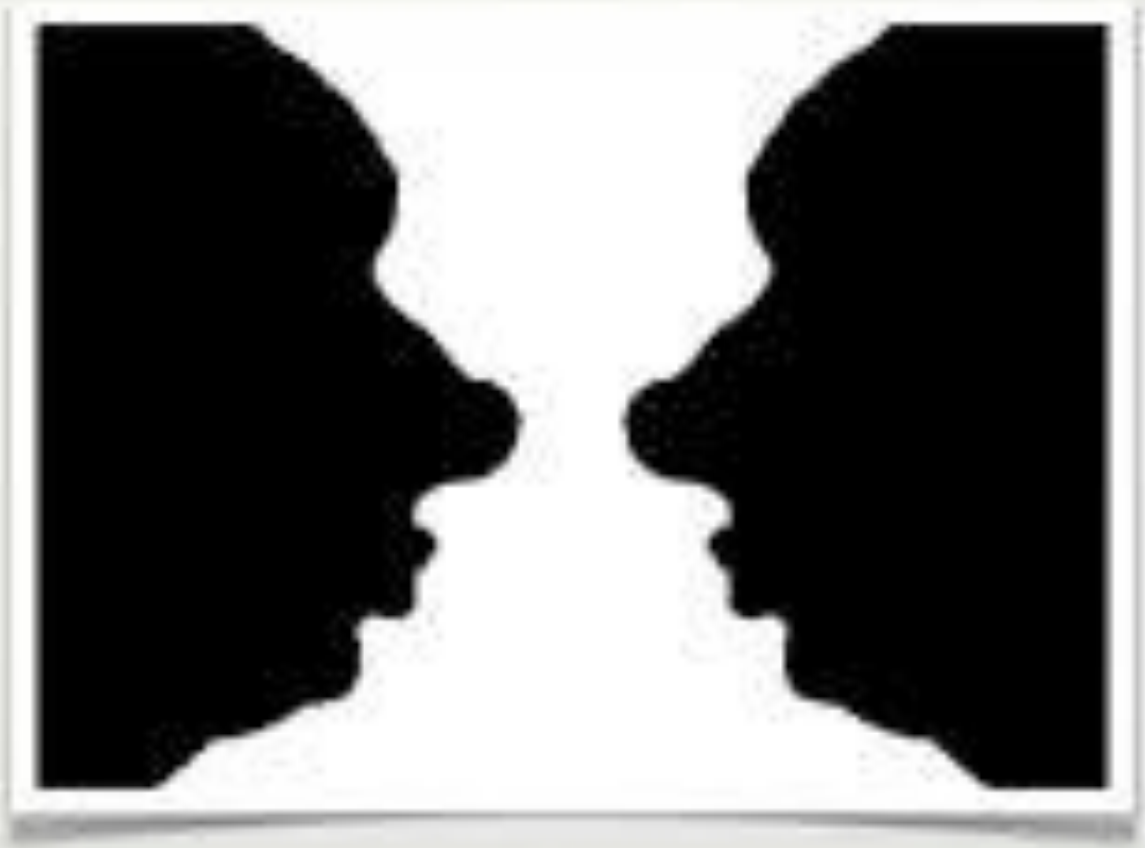
Ejemplo 2: Vision

- Escenas visuales pueden tener varias interpretaciones, a menudo ambiguas
- Ilusiones opticas ejemplifican estas ambiguedades
- Sistemas de vision utilizan heuristicas para seleccionar la mas possible de las soluciones de la escena.




©René Milot

FIGURAS AMBIGUAS



¿COPA O CARAS?

2. Problema con solucion exacta pero el costo computacional puede ser alto

- El costo de encontrar una solucion es alto
- En el **ajedres** el **espacio de busqueda es inmenso** y combinatorialmente explosive , con un numero de posibles estados creciendo exponencialmente o factorialmente con la profundidad de la busqueda
- **Tecnicas de busqueda** de fuerza bruta como **depth first or breadth first**, puden fallar en **encontrar una solucion** en un **tiempo descente.**
- Las **heuristicas** atacan esta complejidad **guiando la busqueda** por las **rutas mas prometedoras** a traves del espacio.
- **Eliminando los estados no prometedores** y sus descendientes un algoritmo heuristico evita esta explosion combinatoria y encuentra una solucion acceptable.

Ejercicio

- En una ciudad cualquiera, una noche cualquiera un taxi atropella a un peatón y se da a la fuga. La policía comienza con sus indagaciones.
- 1) En la ciudad existen dos clases de taxis, unos verdes y otros azules.
- 2) Hay un único testigo poco fiable -según la policía- que asegura que el taxi era de color azul.
- 3) La policía averigua que los taxis verdes en la ciudad representan el 80% del total, siendo los azules solo un 20%.
- ¿Era el taxi, de color verde o como asegura el testigo era azul?

Ejercicio

- Y ahora contemplemos el asunto visto de otra manera.
- Y añadamos una información más:
- *“La mayor parte de los taxistas verdes son unos imprudentes, extranjeros y en su mayor parte ilegales”*
- ¿Variaría en algo su predicción anterior?

Heuristicas

- **Heuristicas** como toda regla de descurimiento **pueden fallar**
- Una heuristica **es solo algo informado sobre el siguiente paso** para solucionar el problema, pero no va mas alla, no sabe si dos pasos despues va a fallar.
- Se basa en la **experiencia y en la intuicion**
- Usan informacion **limitada , como conocimiento de la situacion presente**
- O en el caso de los algoritmos estudiados, **los estados** que se encuentran en la **lista open []**
- Puede conseguir una solucion o fallar del todo en la busqueda de las soluciones

Heuristicas

- Es una preocupacion en inteligencia artificial
- **Juegos de videos, Prueba de teoremas** (2 aplicaciones Viejas)
- Requieren heuristicas para achicar espacios de posible solucion
- Heuristica es la unica manera practica
- Revisarias los 10 ^ 120 estados del ajedres?
- **Rules of thumb (reglas del dedo gordo)**
- Pensar en las heuristicas desde dos perspectivas: **la medida heuristica** y el **algoritmo que usa heuristicas para buscar el espacio de estado**

Más ejemplos de heuristicas

- Debo organizar un concierto, como cumplo con esta tarea?

Pensemos en el juego de tres en rayas

- Tienen alguna heuristica que les permita ganar el juego?

Reducción por Simetria

Reduccion por simetria

- Reduce la busqueda espacios
- No hay 9 hay solo 3
  - Esquina
  - Centro
  - Lado

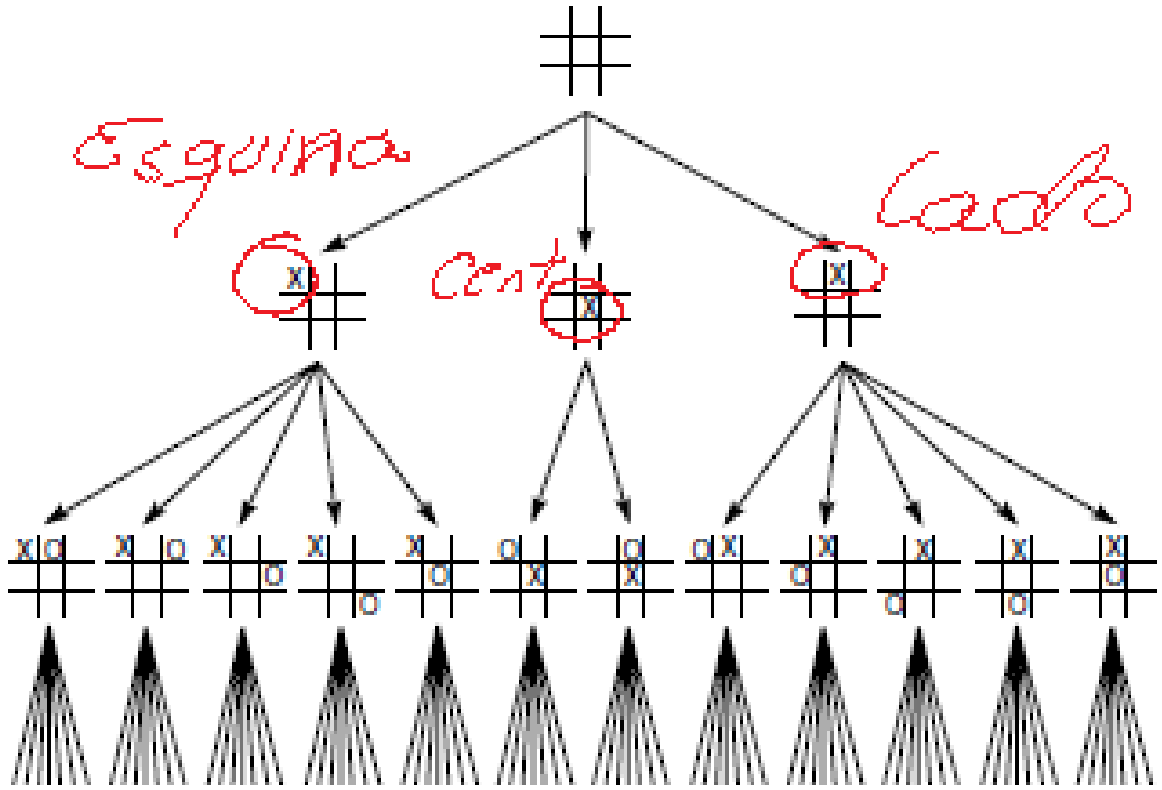
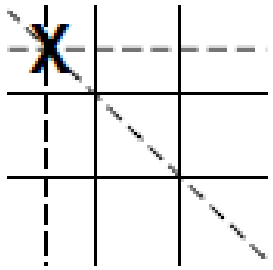


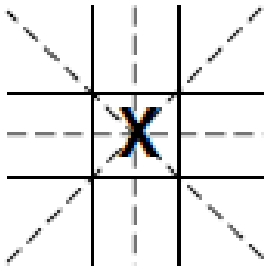
Figure 4.1 First three levels of the tic-tac-toe state space reduced by symmetry.

Reduccion por simetria

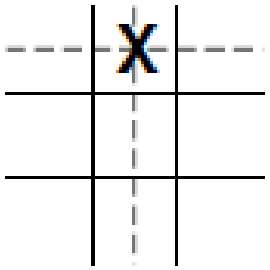
- Cual de estas tres opciones presentan mas oportunidades de ganar?
- Cual deberia ser la que yo escoja?



Three wins through a corner square

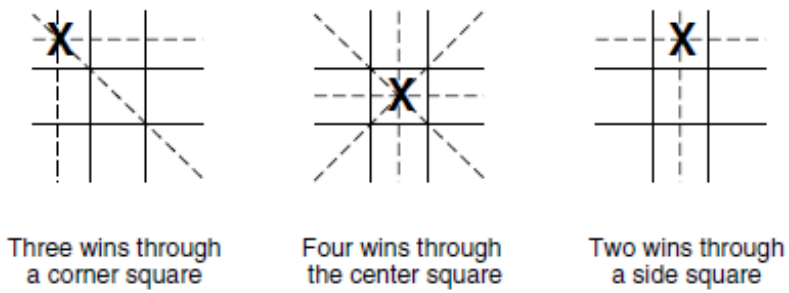


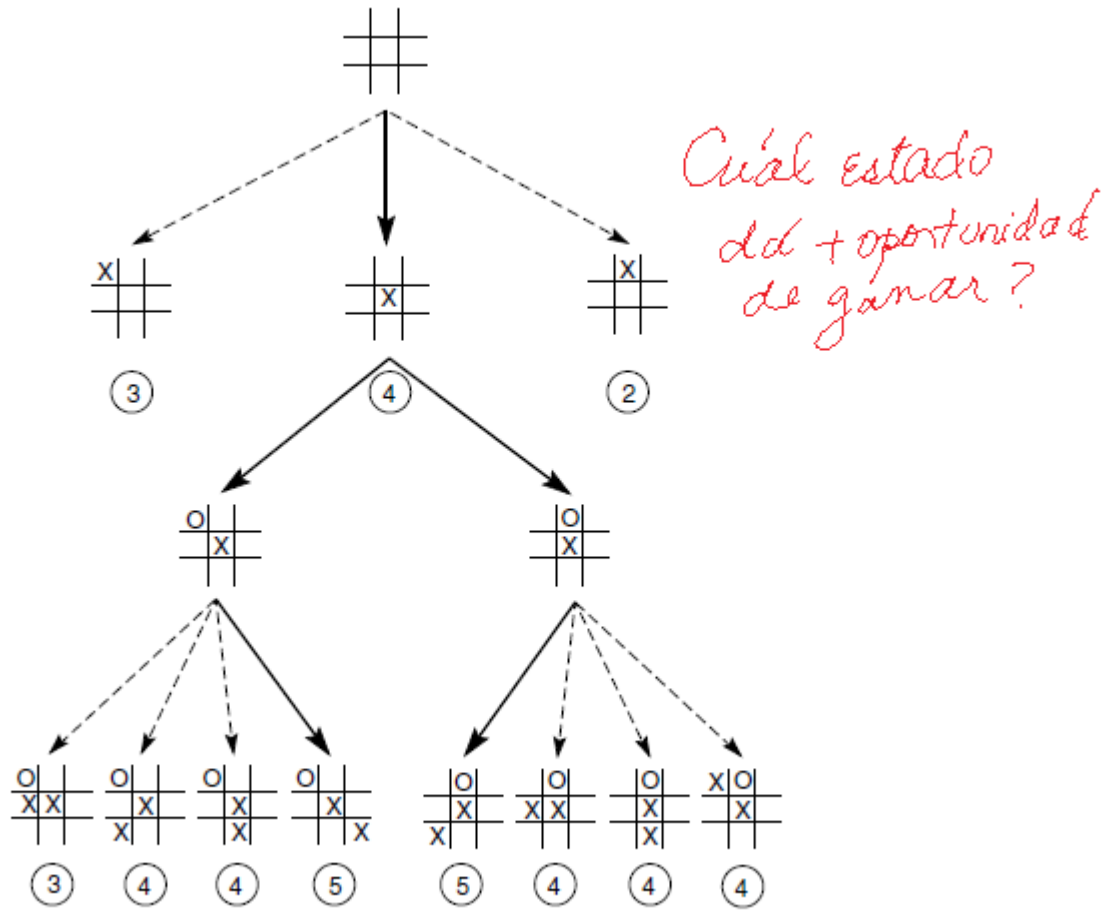
Four wins through the center square



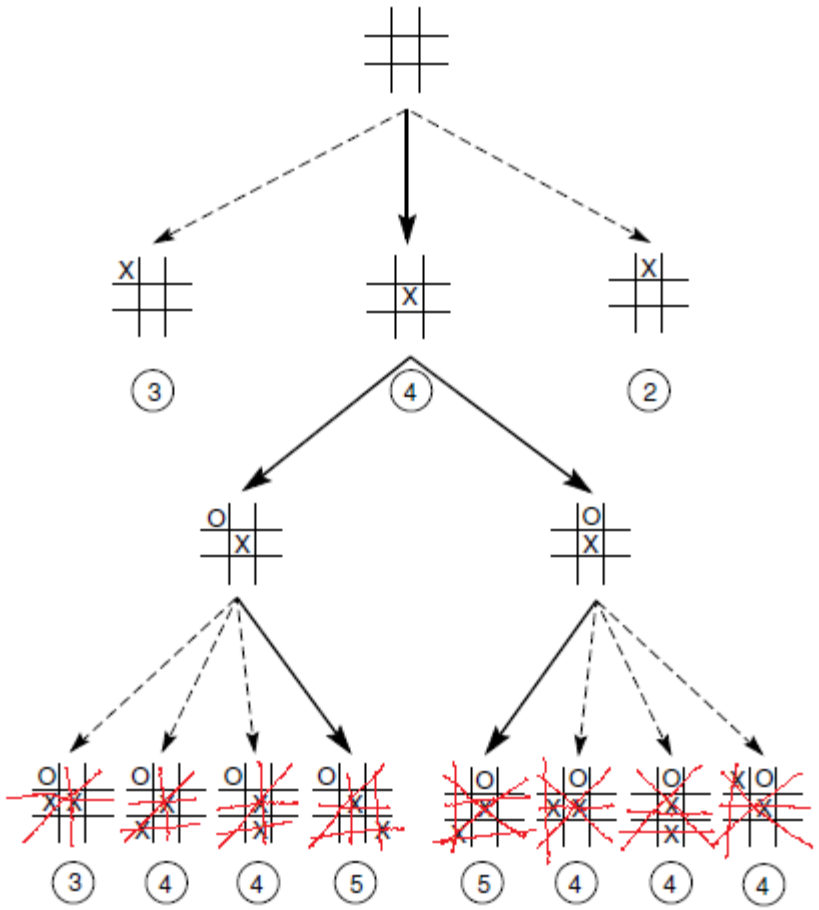
Two wins through a side square

Reduccion por simetria





Reduccion por simetria



Algoritmos para implementar heuristicas

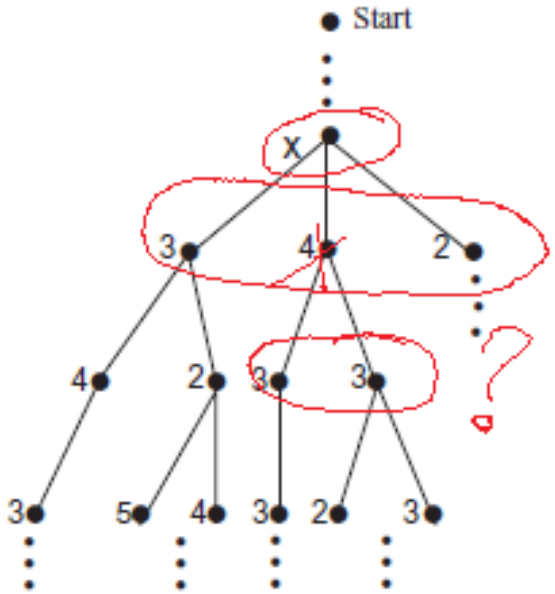
Hill climbing and Dynamic programming, Best first search,

Algoritmos para implementar heuristicas

- High Climbing
- Dinamic programming
- Best first search

Hill Climbing

- Evalua sus hijos
- El mejor hijo es seleccionado para extender la expansion
- No retiene en memoria ni a sus hermanos ni a sus padres
- Hill Climbing : subir vendado hasta lo mas alto de un risco
- Como no guarda historia no puede recuperarse de la falla
- Ej. Tomar el estado con mas probabilidad de ganar
- Problema de detenerse en una local maxima
- Otros algoritmos



Hill Climbing

- Otros algoritmos: Trabajan con medidas heuristicas: $\sum_i a_i x_i$

- Representacion de caracteristicas del juego: **piezas de ventaja, locacion de la pieza, control del centro del juego, oportunidades para sacrificar piezas por ventaja y hasta calculos de momentos de inercia de las piezas de un jugador**
- La constante corresponde a un peso especialmente definido que modela la importancia de los factores

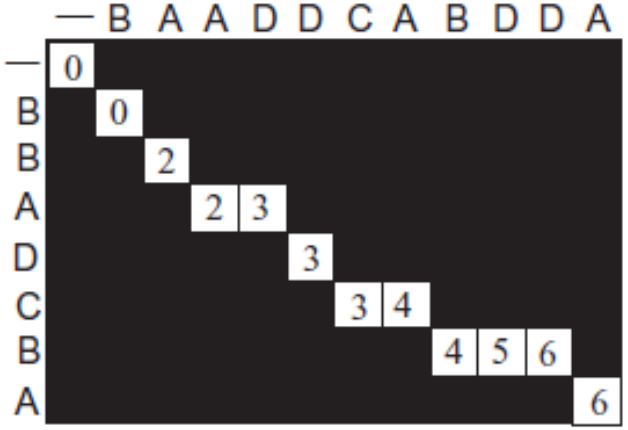
Dynamic Programming

- Tambien llamada el
  - forward – backward
  - Viterbi algorithm
- Richard Bellman (1956)
- Cuida el problema de la busqueda con problemas de memoria
- Usa lo que ya analizo para conseguir una solucion
- Por ejemplo el **reuso de soluciones** subseries para resolver la serie de Fibonacci.
- Reuso de tecnicas se denomina: **Memoizing partial subgoal solutions.**
- Resultado: algoritmo que se utiliza para string matching, spell checking, PNL

Dynamic Programing

- Para llenar la estructura utiliza el forward stage
  - Si hay que mover el character el costo es 1
  - Si se mueve y se inserta el costo es 2
  - Si se inserta un nuevo carácter el costo es 1
  - Si los caracteres son idénticos el costo es 0
- Una vez lleno empezamos el paso backward para producer la solucion

	—	B	A	A	D	D	C	A	B	D	D	A
—	0	1	2	3	4	5	6	7	8	9	10	11
B	1	0	1	2	3	4	5	6	7	8	9	10
B	2	1	2	3	4	5	6	7	6	7	8	9
A	3	2	1	2	3	4	5	6	7	8	9	8
D	4	3	2	3	2	3	4	5	6	7	8	9
C	5	4	3	4	3	4	3	4	5	6	7	8
B	6	5	6	5	4	5	4	5	4	5	6	7
A	7	6	5	4	5	6	5	4	5	6	7	6



Tarea:

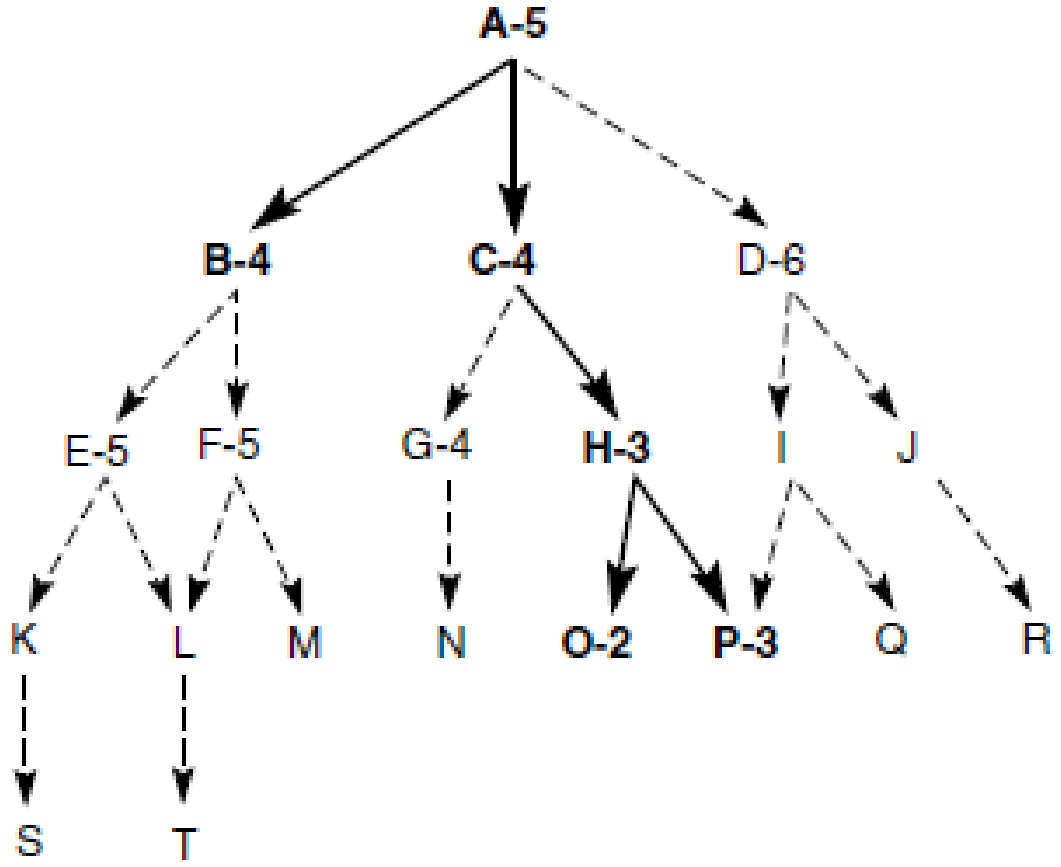
- Usando la siguiente comparacion de palabras y utilizando la function de dynamic programing compare las siguientes dos palabras.
- Magical , dramatical
- Arme la estructura, y llenela utilizando los costos mencionados en las paginas 129-130-131-132
- Muestre la estructura paso a paso, como voy llenandola al revisar y comparer caracteres.
- Muestre la estructura final
- Explique el resultado, de cual es el menor costo.

Best first search algorithm

- El objetivo es encontrar el estado meta mirando a la menor cantidad de estados posibles, **Mientras mas informada la heuristica** , menos estados son procesados para encontrar la meta.
- Con una **cola de prioridad**
- Recuperacion de estas situaciones (fin muerto, **maxima local**, anomalias relaciondas) es possible
- Usa listas para mantener los estados
- **Open[], Close[]**
- Paso adicional en la lista open, ordena los estados de acuerdo a una heuristica X estimada sobre su cercania a un objeto.
- Siempre se considera el estado en la lista open el mas prometedor.

Best first search algorithm

- Si el primer element en open no es el objetivo el algoritmo aplica reglas para generar descendientes
- Si un hijo no alcanza el objetivo, el algoritmo aqui aplica una evaluacion heuristica a ese estado , y la lista open se la Vuelve a sortear.
- Si la lista open se la mantiene sorteada constantemente, se la refiere como “una lista de prioridad”



Best first search algorithm

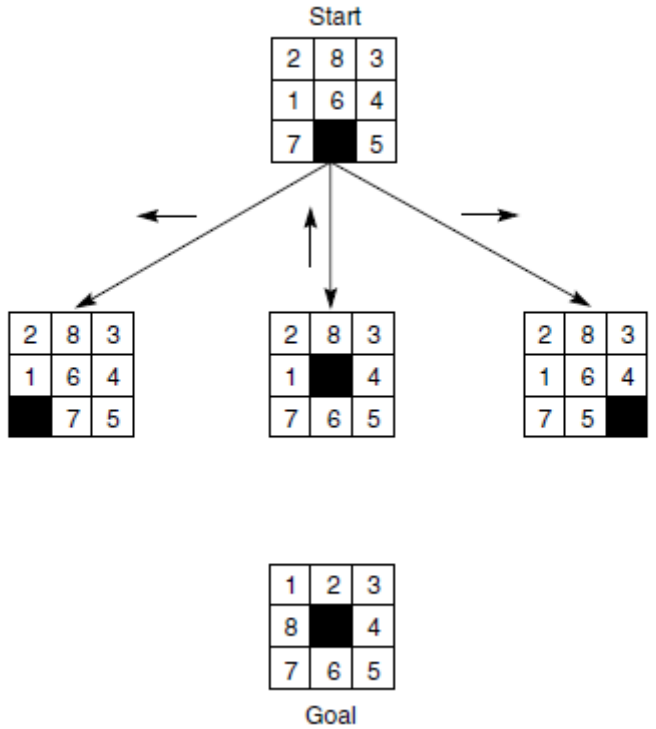
- El algoritmo best first search selecciona el estado mas prometedor salido de open para luego buscarle sus hjos ,
- Si la **heuristica se equivoca** , **no abandona** todos los otros estados y **los mantiene en open** . En el evento de que la heuristica los lleve hacia una ruta incorrecta , el algoritmo podra retomar algunos “next best” de open y enfocarse en otra parte de la busqueda.

Best first search algorithm

- Si el hijo es un estado que se encuentra en cualquiera de las dos listas open o closed , el algoritmo asegura que el estado registre la mas corta de dos rutas de soluciones parciales

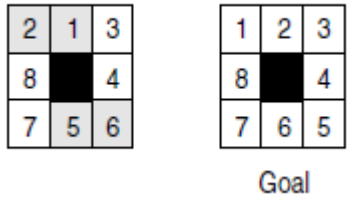
Implementacion de funciones de evaluacion heuristica

- Esta heuristica simple cuenta el numero de cuadritos que estan en un lugar diferente al marcado en el estado objetivo
- Esta heuristica no presenta toda la informacion disponible en la tabla. No toma en cuenta las distancias que se deben de mover los cuadritos
- Una mejor heuristica contaria las distancias o el numero de cuadritos que se deben de mover para llegar al objetivo



Implementacion de funciones de evaluacion heuristica

- Pero ambas heuristics tienen dificultad en voltear los cuadritos que estan juntos . Necesitan mas de 2 movimientos para ponerlos en su lugar
- Podemos ver como seria si se aplicaran las tres heuristicas



	Tiles out of place	Sum of distances out of place	2 x the number of direct tile reversals
2 8 3 1 6 4 7 5	5	6	0
2 8 3 1 4 7 6 5	3	4	0
2 8 3 1 6 4 7 5	5	6	0

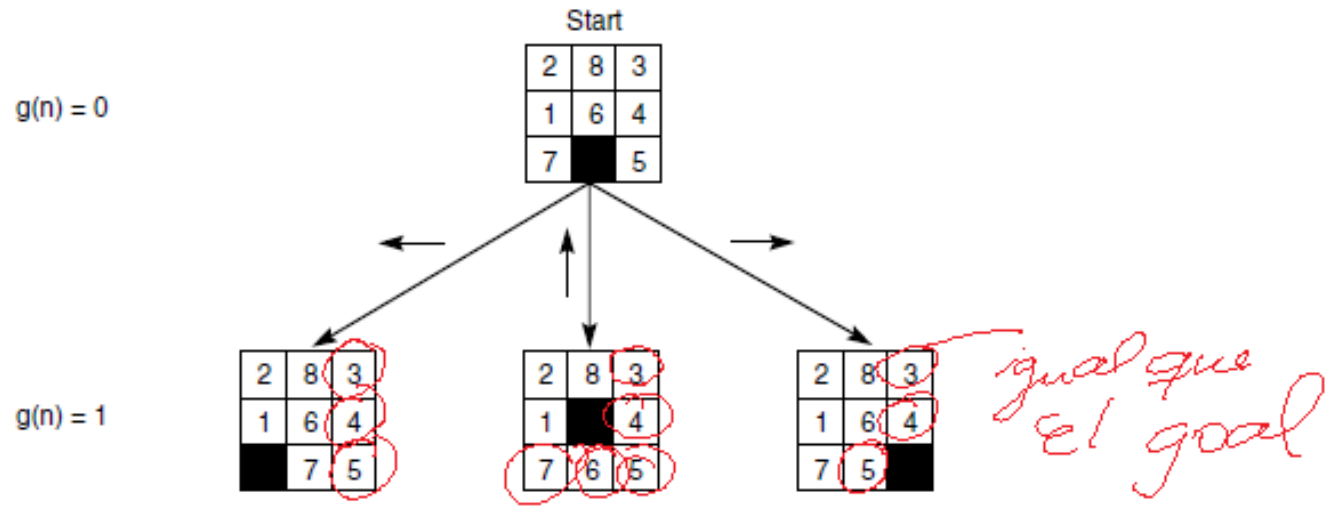
Implementacion de funciones de evaluacion heuristica

- La ruta desde su **estado inicial a sus hijos** se puede medir un **conteo de profundidad** por cada estado
- Este conteo es 0 para el estado inicial y **se incrementa en 1 por cada nivel de busqueda**
- Esta **medida de profundidad** se la puede anadir a la evaluacion heuristica de cada estado para sesgar la busqueda a favor de los estados que se encuentran menos profundos en el grafo
- La funcion de evaluacion queda:

Implementacion de funciones de evaluacion heuristica

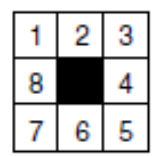
$$f(n) = g(n) + h(n)$$

- $g(n)$: mide la magnitud de la ruta de cualquier estado n al estado inicial
- $h(n)$: es la heurística estimada del estado n al estado objetivo



Values of f(n) for each state, 6 4 6

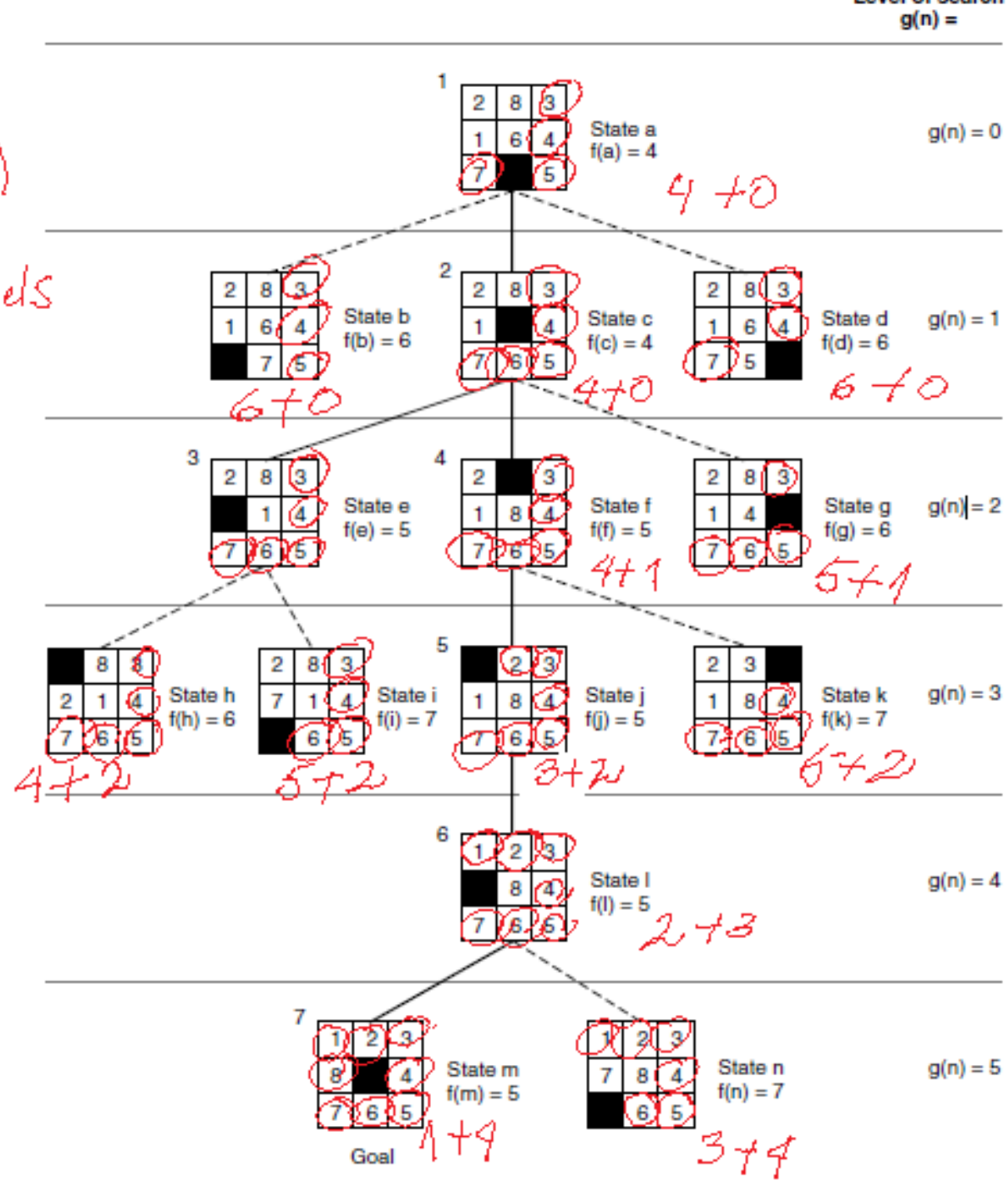
where:
f(n) = g(n) + h(n),
g(n) = actual distance from n to the start state, and
h(n) = number of tiles out of place.



$f(n) = h(n) + g(n)$

↓ tiles diferentes

↓ #levels



Busqueda heuristica y sistemas expertos

Dimensiones

Admisibilidad, informacion y monotonicidad

Dimensiones: Admisibilidad, monotonicidad e informedness

- Evaluar el comportamiento de la heuristica a traves de varias dimensiones
- Las heuristicas encuentran la ruta mas corta a un objetivo cuando este existe y se les denomina “**admisibles**”
- Como saber si una heuristica es mejor que otra, esto lo da el “**informedness”** de la heuristica o la cantidad de informacion.
- Si un estado es descubierto a travez de una busqueda heuristica, no hay garantia de que el mismo estado no se lo encuentre mas tarde en la busqueda a un costo mas barato(ruta mas corta del estado inicial) a esto se le denomina “**monotonicity”**

Medidas de admisibilidad

- Usando la function de evaluacion conocida f(n)= g(n)+h(n)
- Si conseguimos varias rutas, digamos dos.
- Calculamos la function de evaluacion en ambos casos.
- La ruta que tenga el menor costo y que sea mas efectiva es la que da el valor de admisibilidad de esa heuristica

- En este sentido f(n) estima el costo total de la ruta del estado de inicio a traves de n hacia el estado objetivo
- Para determiner las propiedades heuristicas admisibles , se define una function de evaluacion asterisco igual a la function f(n)
- Donde g(n) es el costo de la ruta mas corta del inicio al nodo n y h(n) devuelve el costo actual de la ruta mas corta de n al objetivo.
- Entonces la function f*(n) = al costo actual de la ruta optima , entonces del nodo inicial a un nodo objetivo que pase a trave del nodo n
- Si se emplea best first search con la function f* la estrategia resultante es admissible

- Por ejemplo la heuristica de contra el numero de cuadrito que no esta en la posicion objetivo es menoro igual al numero de moviemientos que se requiren para moverlos a su posicion objetivo. Asi esta heuristica es admissible y garantiza una optima o mas corta ruta solucion si existe la ruta
- La sumaroria de las distancias directas de cuadritos fuera del lugar es Tambien menor o igual a la minima ruta

Usando heuristicas en los juegos.

- Procedimiento MINIMAX
- Minimaxing para arreglar ply depth
- El procedimiento Alpha beta

Usando heuristicas en los juegos

- Consideremos juegos donde el espacio de estado sea suficiente pequeno para que sea buscado exhaustivamente
- Es decir buscando sistematicamente el espacio para psoibles movimientos y contramovimientos de parte del oponente
- Los oponentes se los refiere como MIN y MAX
- Max representa el jugador tratando de ganar o Maximizar su ventaja
- MIN es el oponente que trata de Minimizar el marcador de MAX

Aspectos de complejidad