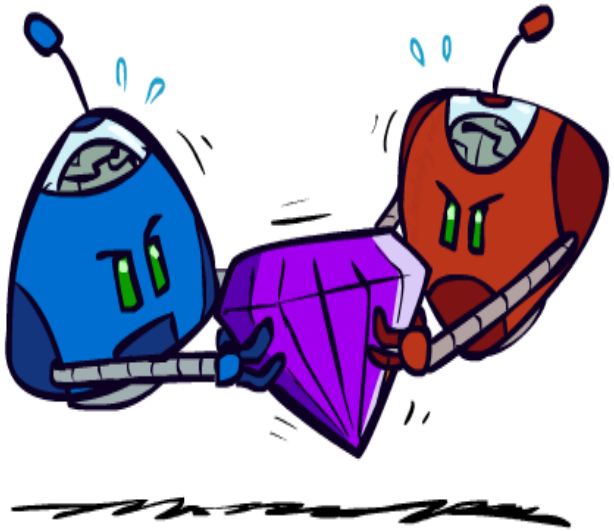
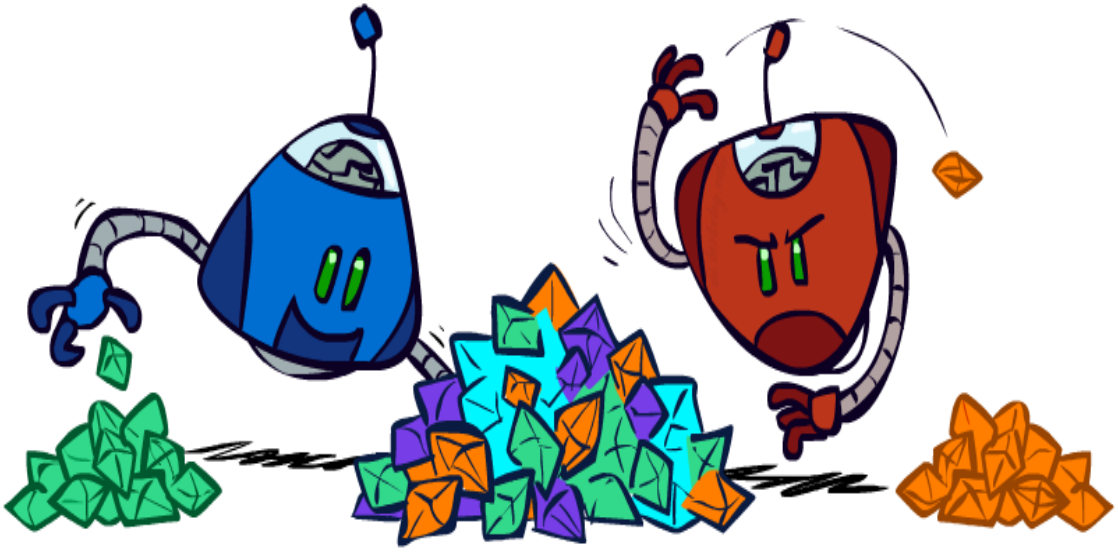
Juegos de suma cero

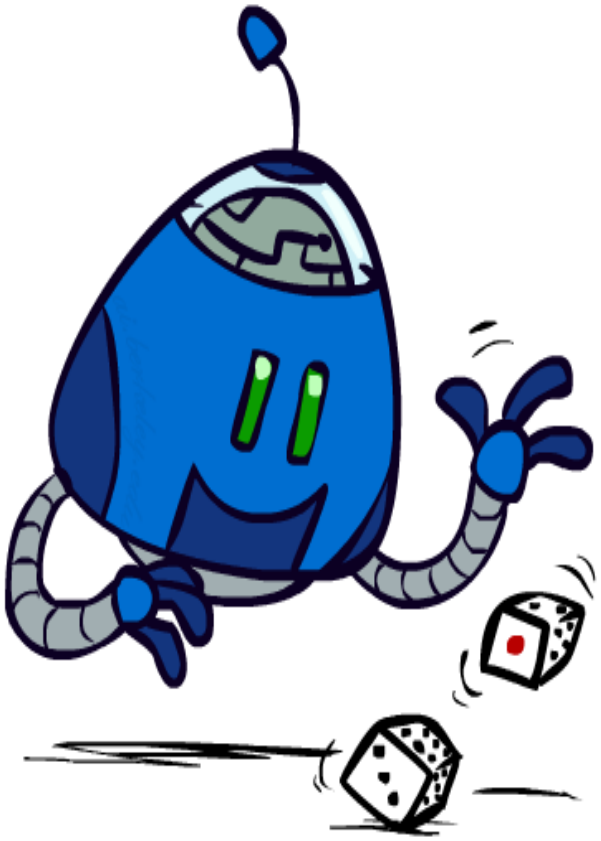




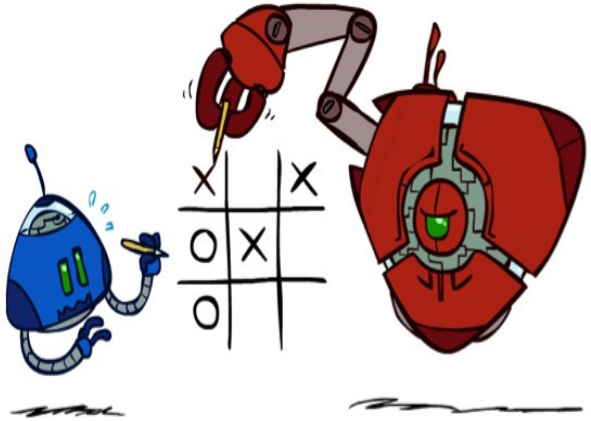
- Juegos de suma cero
  - Los agentes tienen utilidades opuestas (valores)
  - Podemos pensar en un único valor que uno maximiza y el otro minimiza
  - Adversarial, competición pura

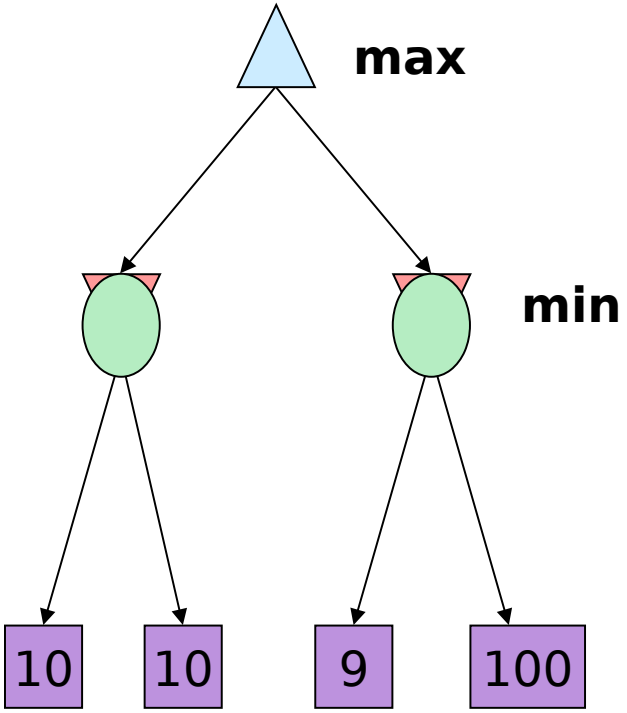
- Juegos Generales
  - Los agentes tienen utilidades independientes (valores)
  - Cooperación, indiferencia, competición, y más, todo es posible
  - Más después

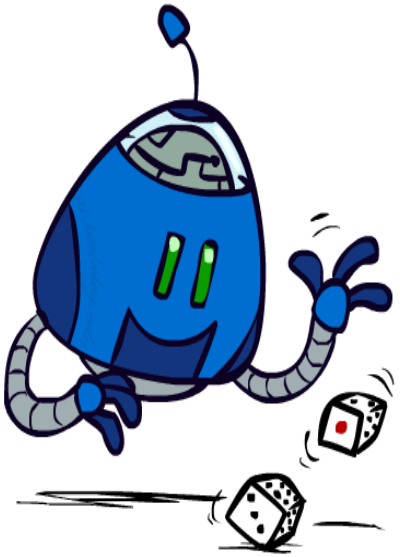
Resultados inciertos



Caso peor vs. caso medio



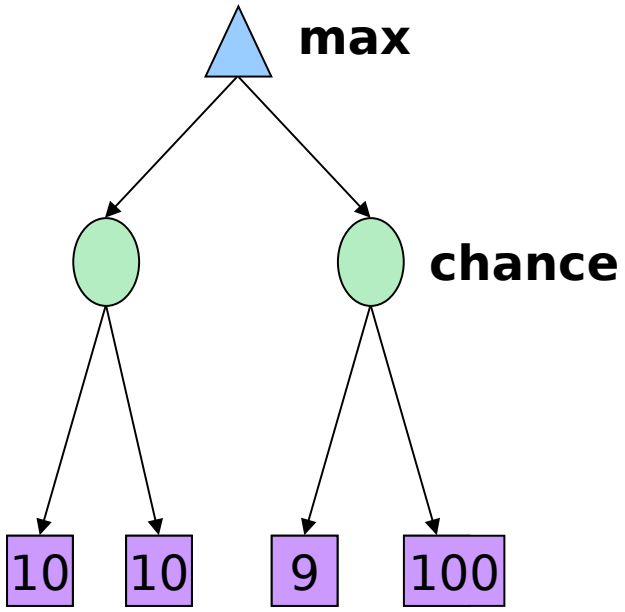




Idea: los resultados inciertos están controlados por el azar, no por el adversario!

Búsqueda Expectimax

- ¿Por qué podemos no conocer el resultado de una acción?
  - Aleatoriedad explícita: echar los dados
  - Oponentes impredecibles: los fantasmas responden aleatoriamente
  - Las acciones pueden fallar: al mover un robot, las ruedas pueden patinar
- Los valores deberían reflejar resultados medios (expectimax), no resultados en el caso peor (minimax)
- Búsqueda Expectimax: calcular la puntuación media con un juego óptimo
  - Nodos Max como en búsqueda minimax
  - Los nodos aleatorios son como los nodos min pero el resultado es incierto
  - Se calcularán las utilidades esperadas
  - P. ej. Tomar la media ponderada (expectativa) de los hijos



Pseudocódigo de Expectimax

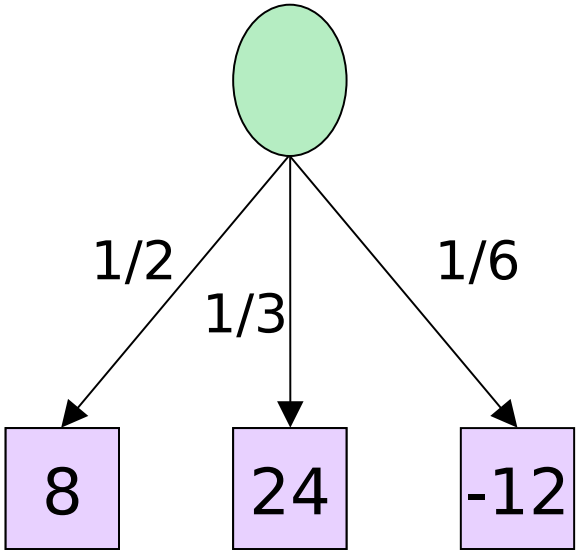
```
def value(state):
    if the state is a terminal state: return the state's utility
    if the next agent is MAX: return max-value(state)
    if the next agent is EXP: return exp-value(state)
```

```
def max-value(state):
    initialize v = -∞
    for each successor of state:
        v = max(v,
                value(successor))
    return v
```

```
def exp-value(state):
    initialize v = 0
    for each successor of state:
        p = probability(successor)
        v += p * value(successor)
    return v
```

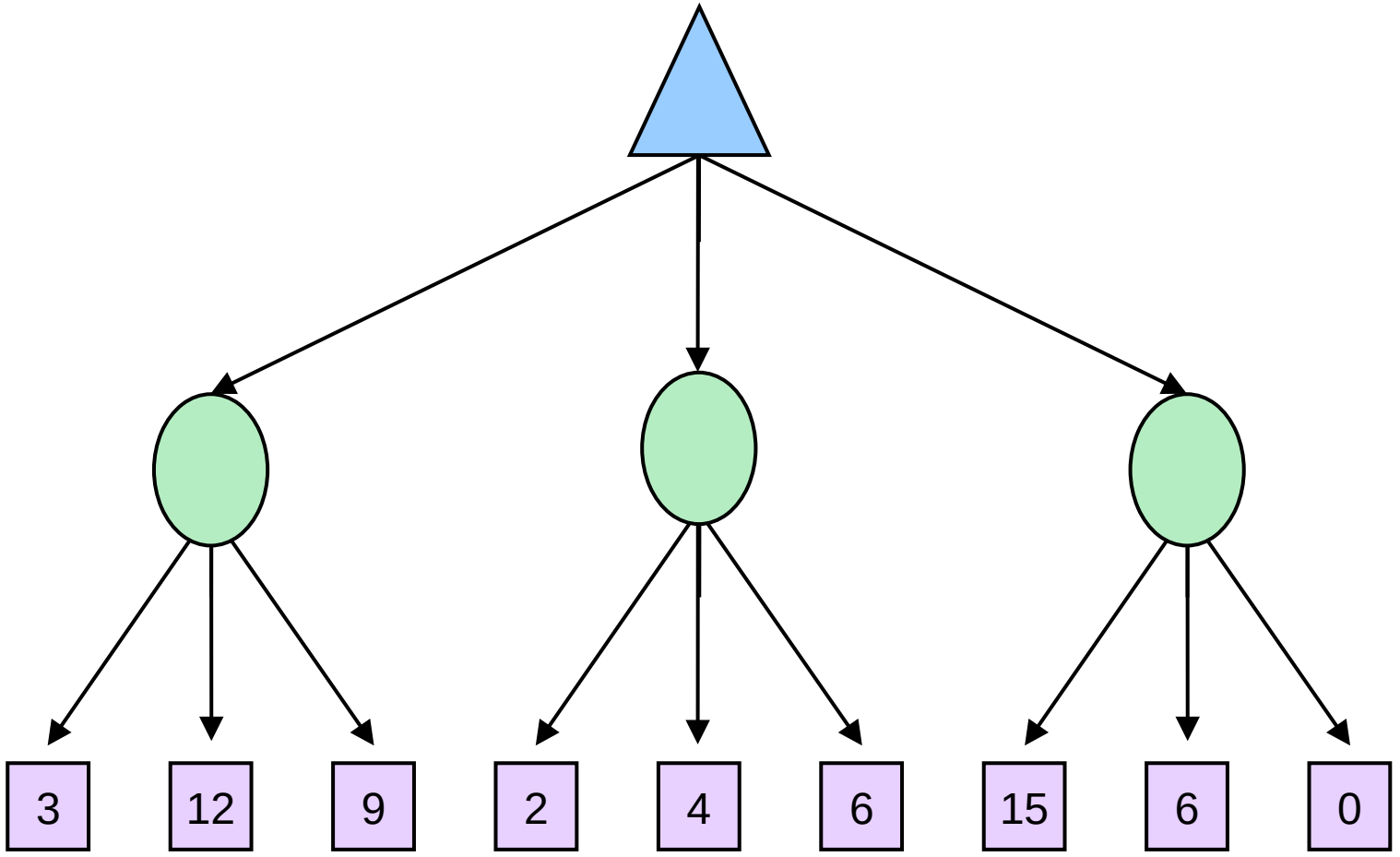
Pseudocódigo de Expectimax

```
def exp-value(state):
    initialize v = 0
    for each successor of state:
        p = probability(successor)
        v += p * value(successor)
    return v
```

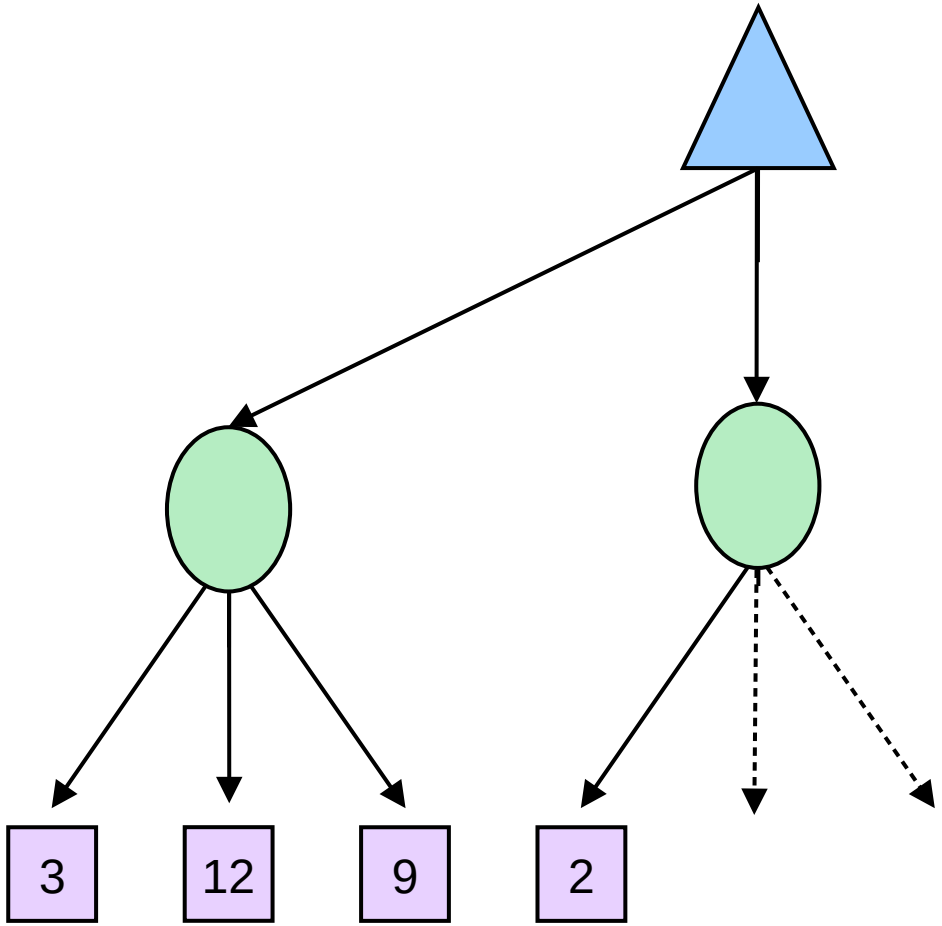


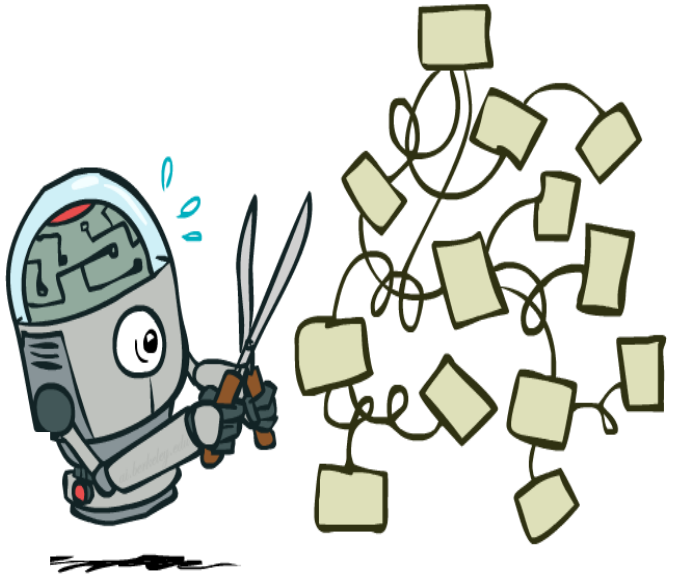
$$v = (1/2)\ (8) + (1/3)\ (24) + (1/6)\ (-12) = 10$$

Ejemplo de Expectimax

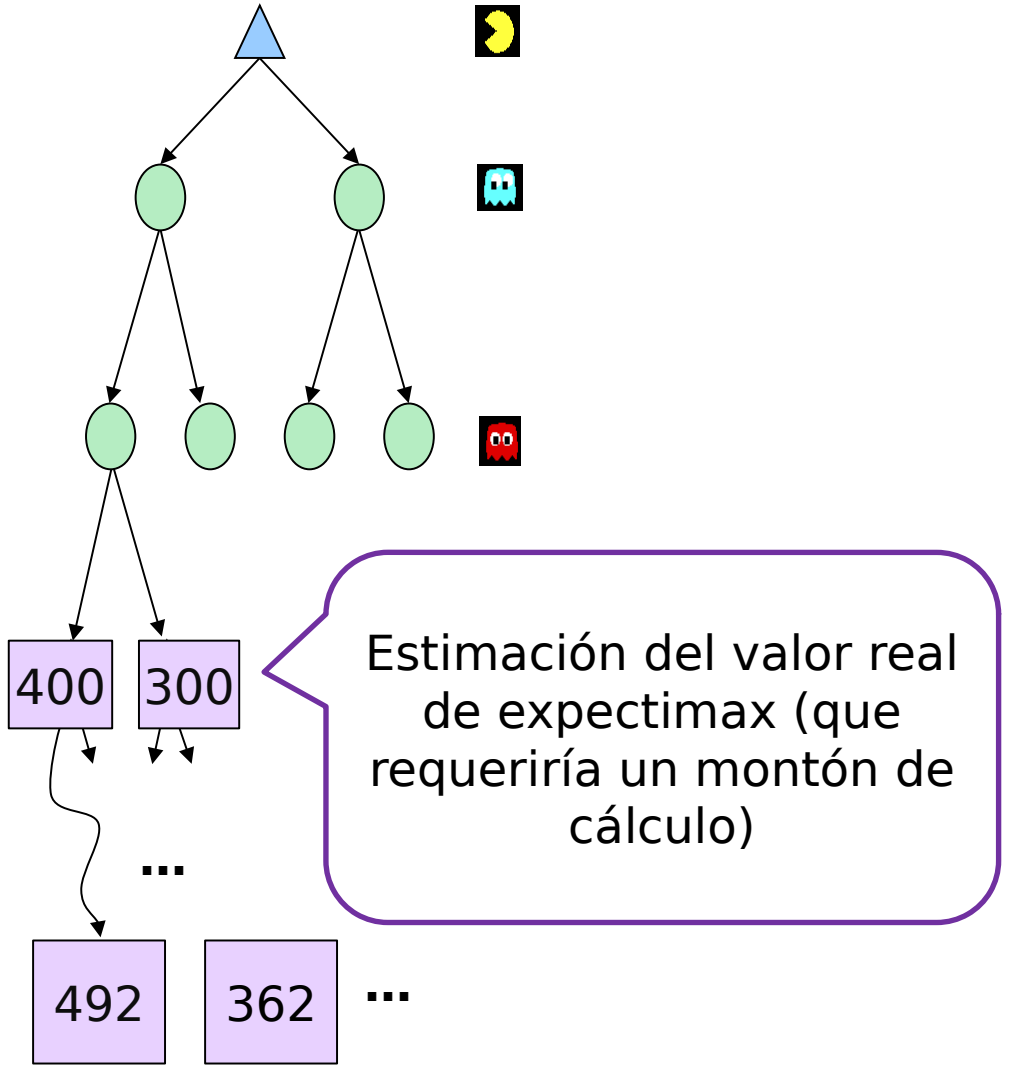


¿Poda de Expectimax?

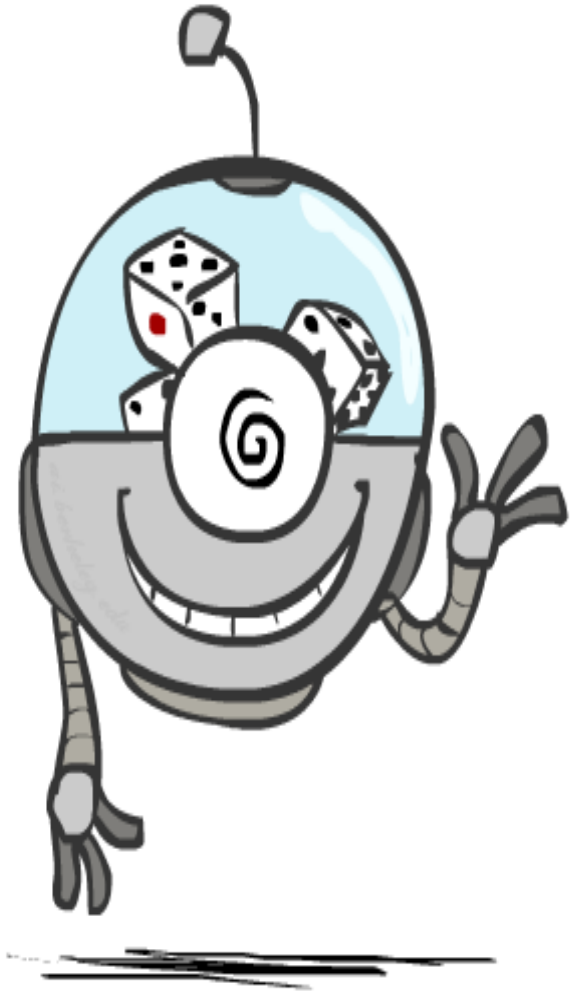




Expectimax con profundida limitada

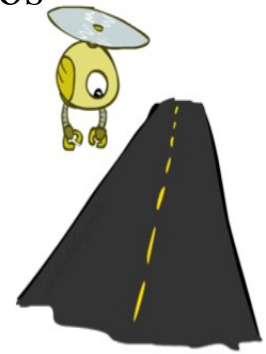


Probabilidades



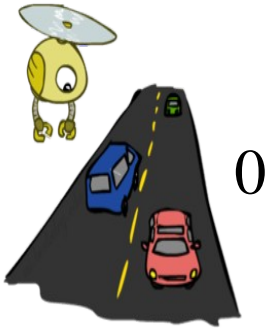
Recuerdo: Probabilidades

- Una **variable aleatoria (random)** representa un evento cuyo resultado es desconocido
- Una **distribución de probabilidad** es una asignación de pesos a resultados



0.25

- Ejemplo: Tráfico en la autovía
  - Variable aleatoria: T = hay tráfico o no
  - Valores: T en {nada, ligero, mucho}
  - Distribución: P(T=nada) = 0.25, P(T=ligero) = 0.50, P(T=mucho) = 0.25



0.50

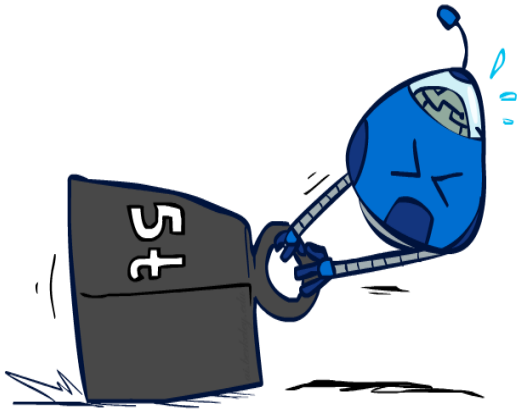
- Algunas leyes de probabilidad:
  - Las Probabilidades son siempre no negativas
  - La suma de Probabilidades sobre todos los valores posibles suma uno

0.25

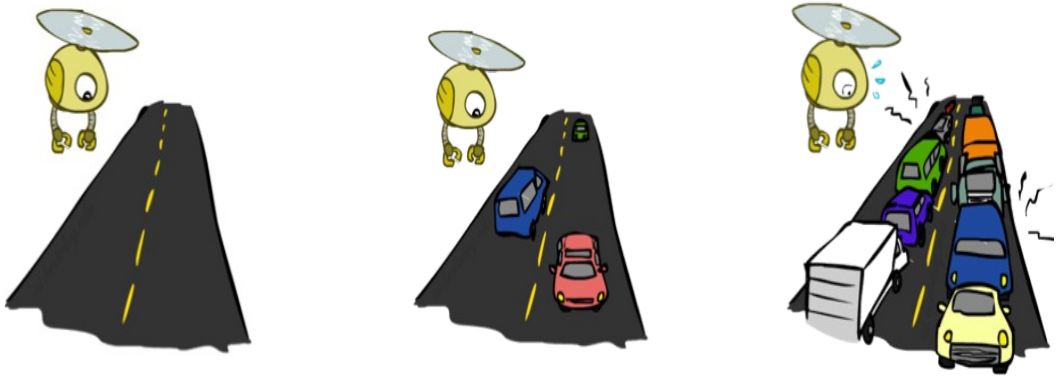
Recuerdo: Expectativas

- El valor esperado de una función de una variable aleatoria es la media, ponderada por la distribución de probabilidad de los resultados

- Ejemplo: ¿Cuánto tardaré en llegar al aeropuerto?



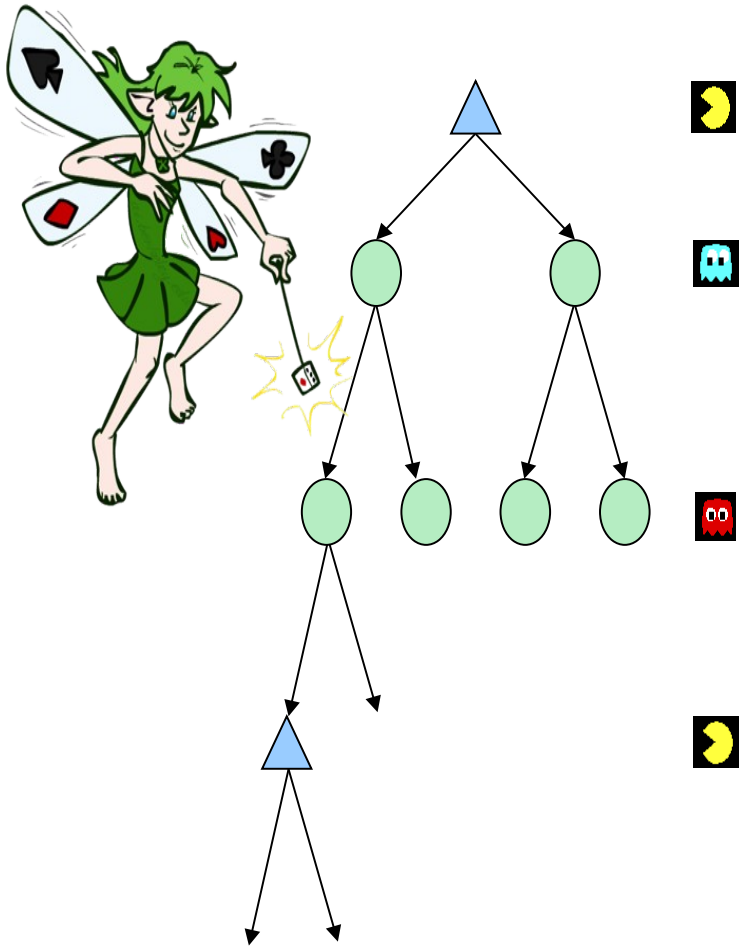
Tiempo:	20 min		30 min		60 min		
	x	+	x	+	x	→	35 min
Probabilidad:	0.25		0.50		0.25		



¿Qué probabilidades usamos?

- En la búsqueda expectimax, tenemos un modelo probabilístico de cómo el oponente (o entorno) se comportará en cualquier estado
  - El Modelo podría ser una distribución uniforme (echar los dados)
  - El Modelo podría ser sofisticado y requerir un montón de computación
  - Tendremos un nodo aleatorio por cada situación fuera de nuestro control: oponente o entorno
  - ¡El modelo podría decir qué acciones adversariales son probables!

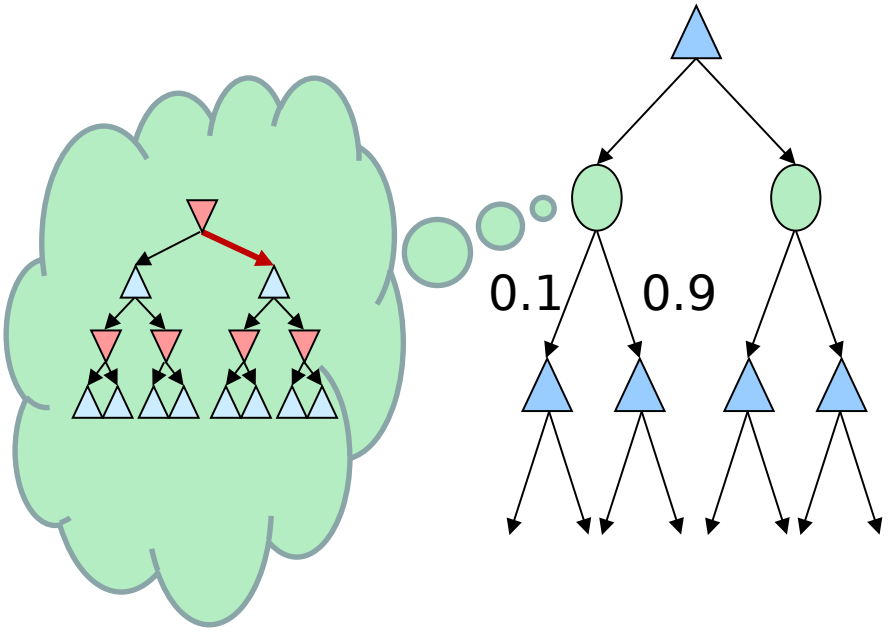
- Por ahora, asumiremos que cada nodo viene mágicamente con probabilidades que especifican la distribución respecto a sus valores



¡Tener una suposición probabilística sobre la acción de otro agente no quiere decir que ese agente esté echando una moneda!

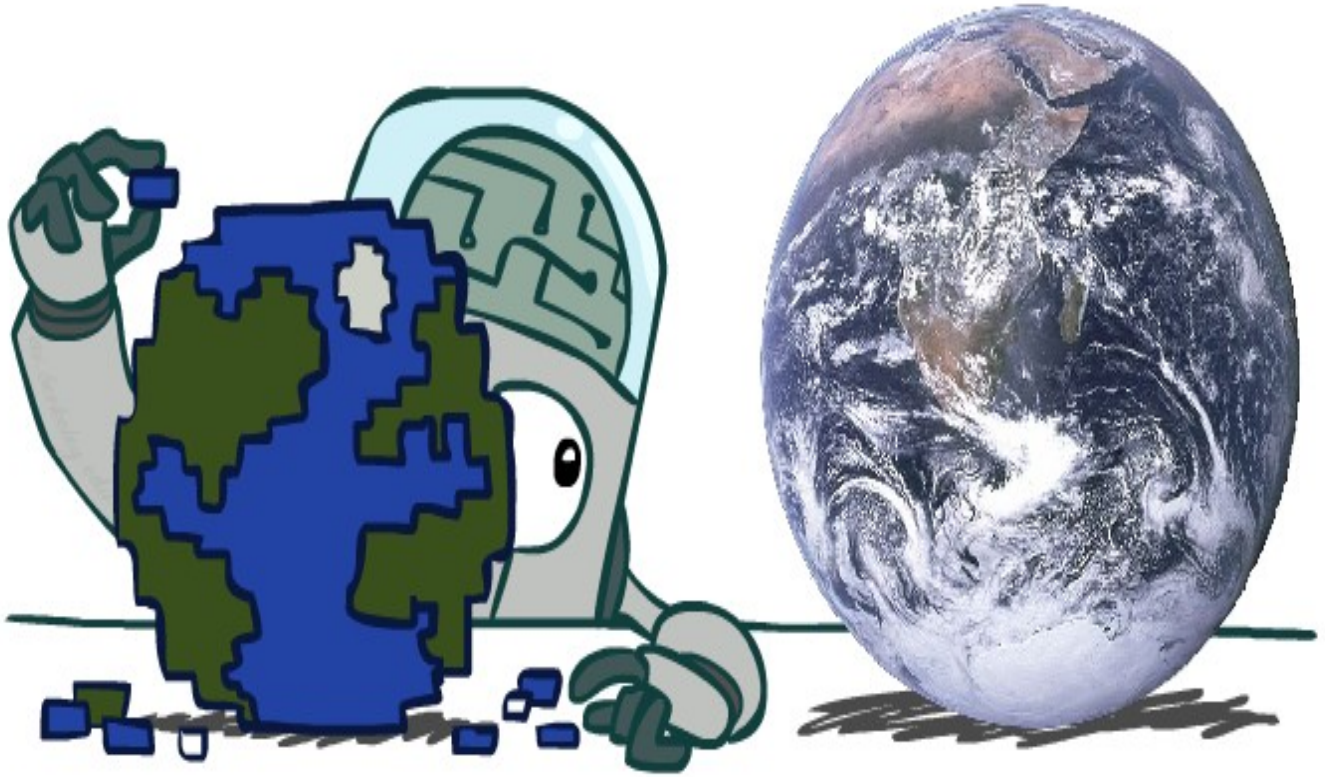
Quiz: Probabilidades informadas

- Supongamos que nuestro oponente está ejecutando un minimax de profundidad 2, usando ese resultado un 80% de las veces, y moviéndose aleatoriamente en otro caso
- Pregunta: ¿Qué tipo de búsqueda en árbol usaríamos?



- Respuesta: ¡Expectimax!
  - Para estimar las probabilidades de CADA nodo aleatorio, tendríamos que ejecutar una simulación de nuestro oponente
  - Esto nos puede llevar rápidamente a ineficiencia (tiempo)
  - Peor si tenemos que simular a nuestro oponente simulándonos a nosotros ...
  - … excepto para minimax, que tiene la (buena) propiedad de que todo se junta en un árbol de juegos

Modelando Asunciones



Los peligros del optimismo y el pesimismo

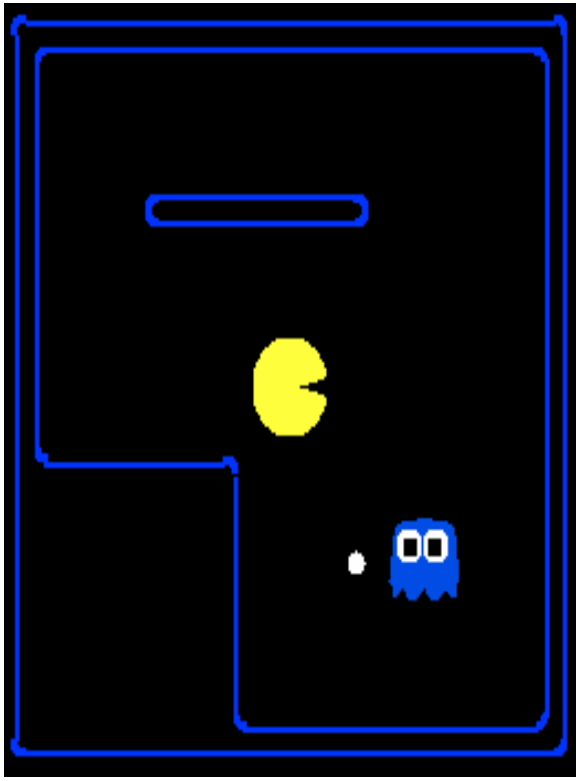
Optimismo peligroso

Asumiendo azar cuando el mundo es adversarial

Pesimismo peligroso

Asumiendo el caso peor cuando es poco probable

Asunciones vs. Realidad



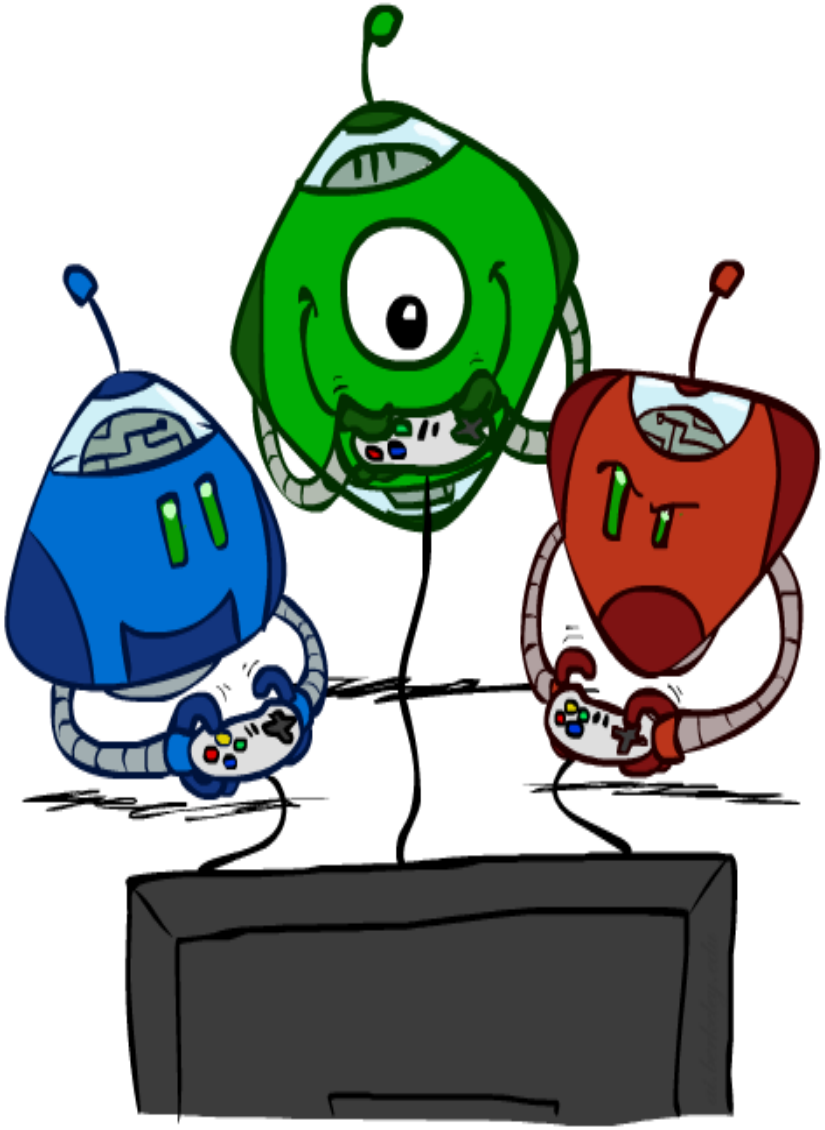
	Fantasma Adversarial	Fantasma Aleatorio
Minimax Pacman	Won 5/5 Avg. Score: 483	Won 5/5 Avg. Score: 493
Expecti max Pacman	Won 1/5 Avg. Score: -303	Won 5/5 Avg. Score: 503

Resultados jugando 5 juegos

- Pacman usó búsqueda de profundidad 4 con una función de evaluación que evita problemas
- El fantasma usó búsqueda de profundidad 2 con una función de evaluación que busca a Pacman

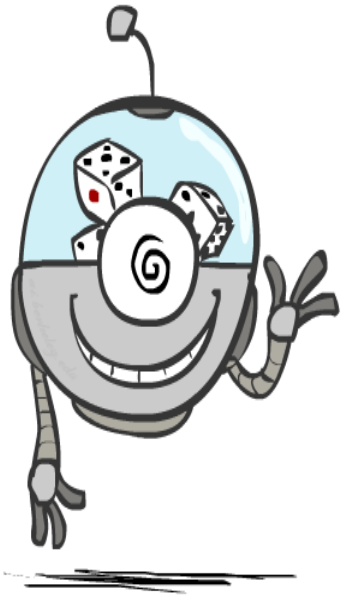
[Demos: world assumptions (L7D3,4,5,6)]

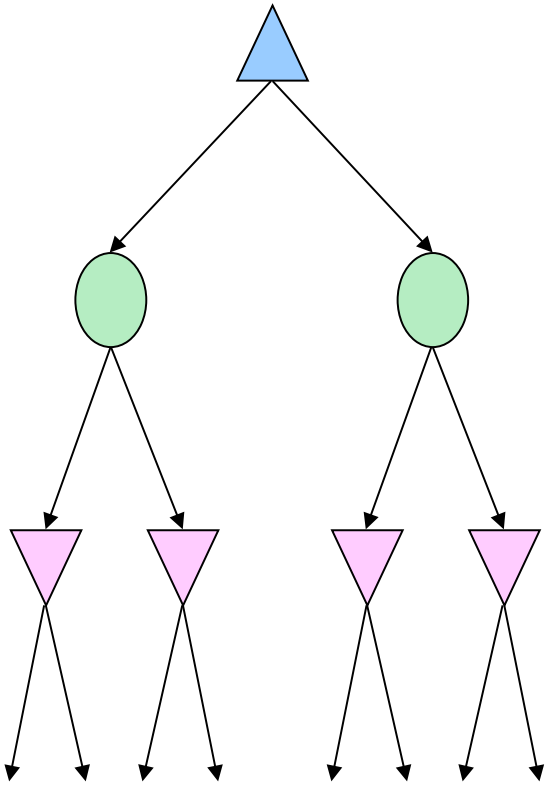
Otros tipos de juegos



Tipos de nivel mixtos

- P. ej. Backgammon
- Expectiminimax
  - El entorno es un "agente aleatorio" extra que mueve después de cada agente min/max
  - Cada nodo calcula la combinación apropiada para sus hijos





Utilidades Multi-agente

- ¿Qué pasa si el juego no es de suma cero, o hay varios jugadores?

- Generalización de minimax:
  - Los terminales tienen tuplas de utilidades
  - Los valores de los nodos son también
  - tuplas de utilidades
  - Cada jugador maximiza su componente
  - Puede dar lugar a competición y
  - cooperación dinámicamente