Descubriendo la Inteligencia Artificial – 112

Grafos

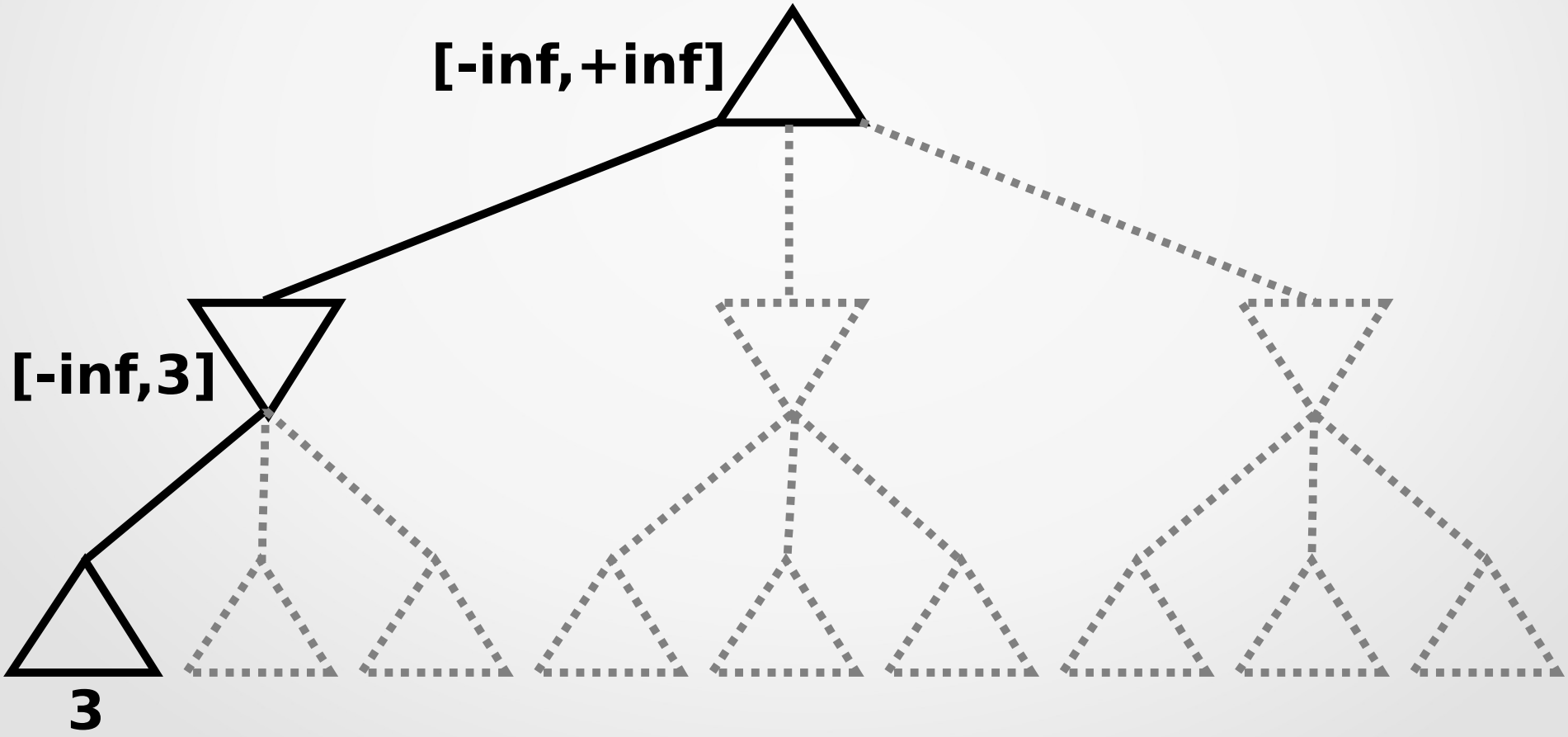
Poda Alfa-Beta

Poda Alfa-Beta

- Complejidad Minimax: exponencial. No se puede reducir a polinómica.
- Objetivo: calcular el valor correcto de la función MiniMax sin tener que mirar todos los nodos de un árbol.
- Técnica: poda de ramas.

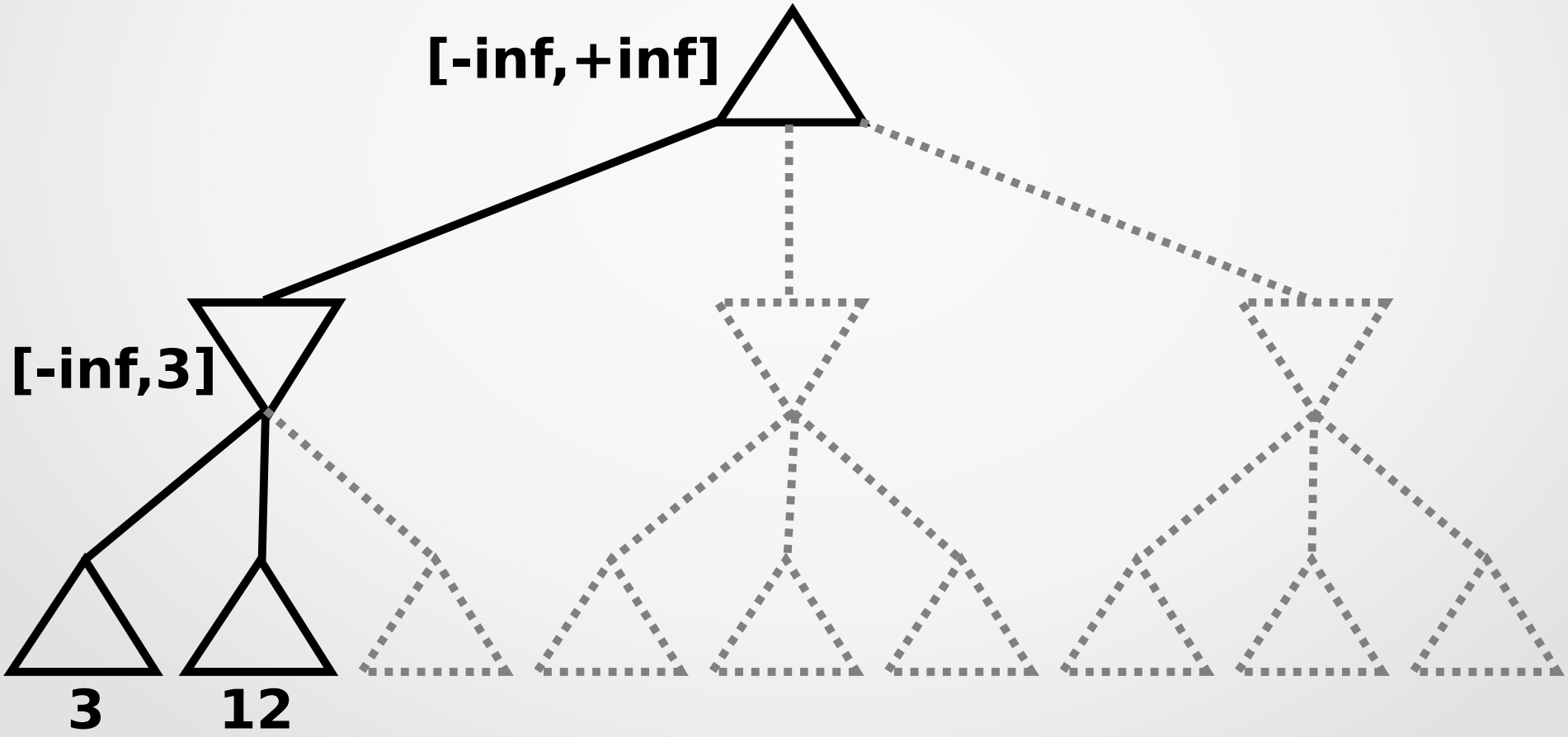
Descubriendo la Inteligencia Artificial – 112

Ejemplo



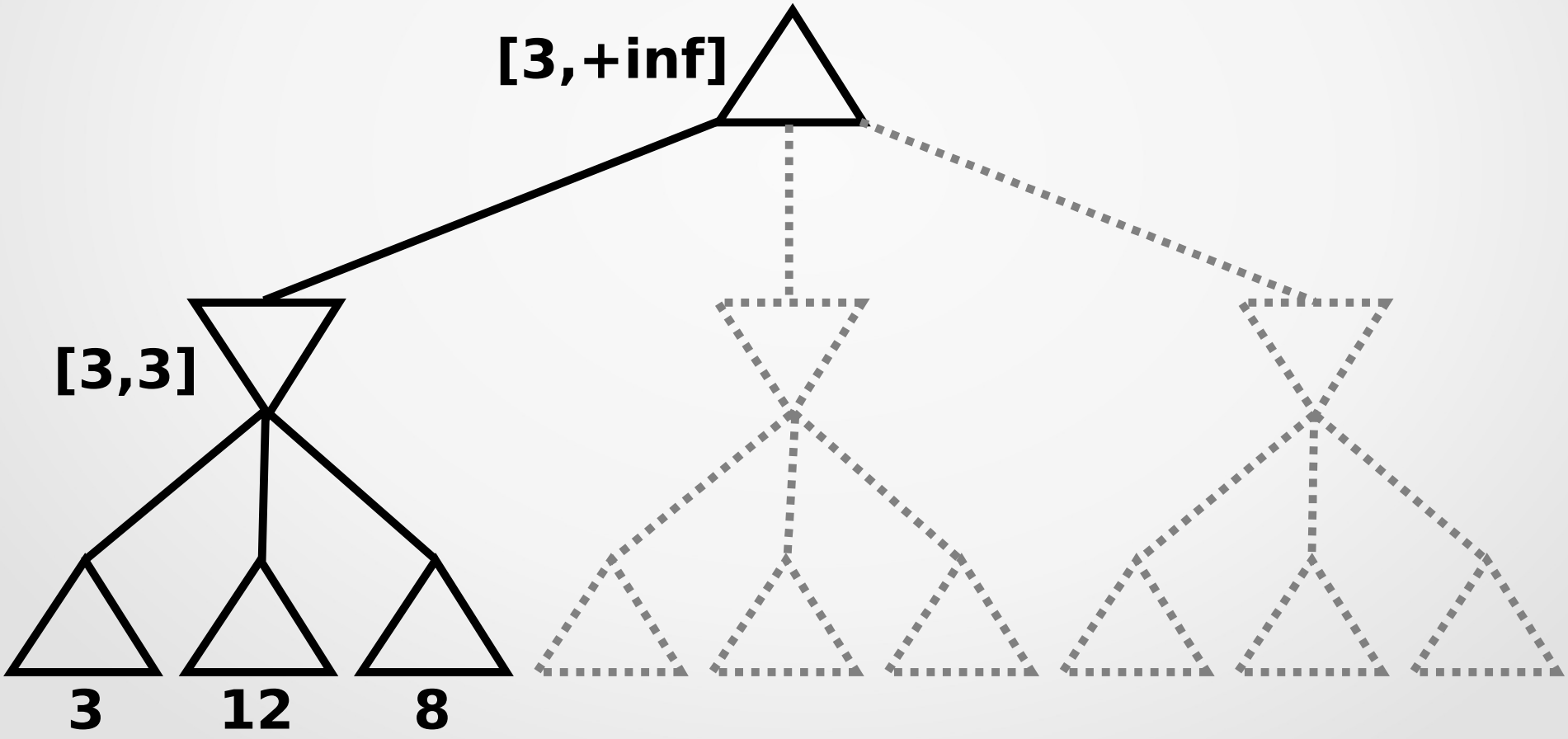
Descubriendo la Inteligencia Artificial – 112

Ejemplo



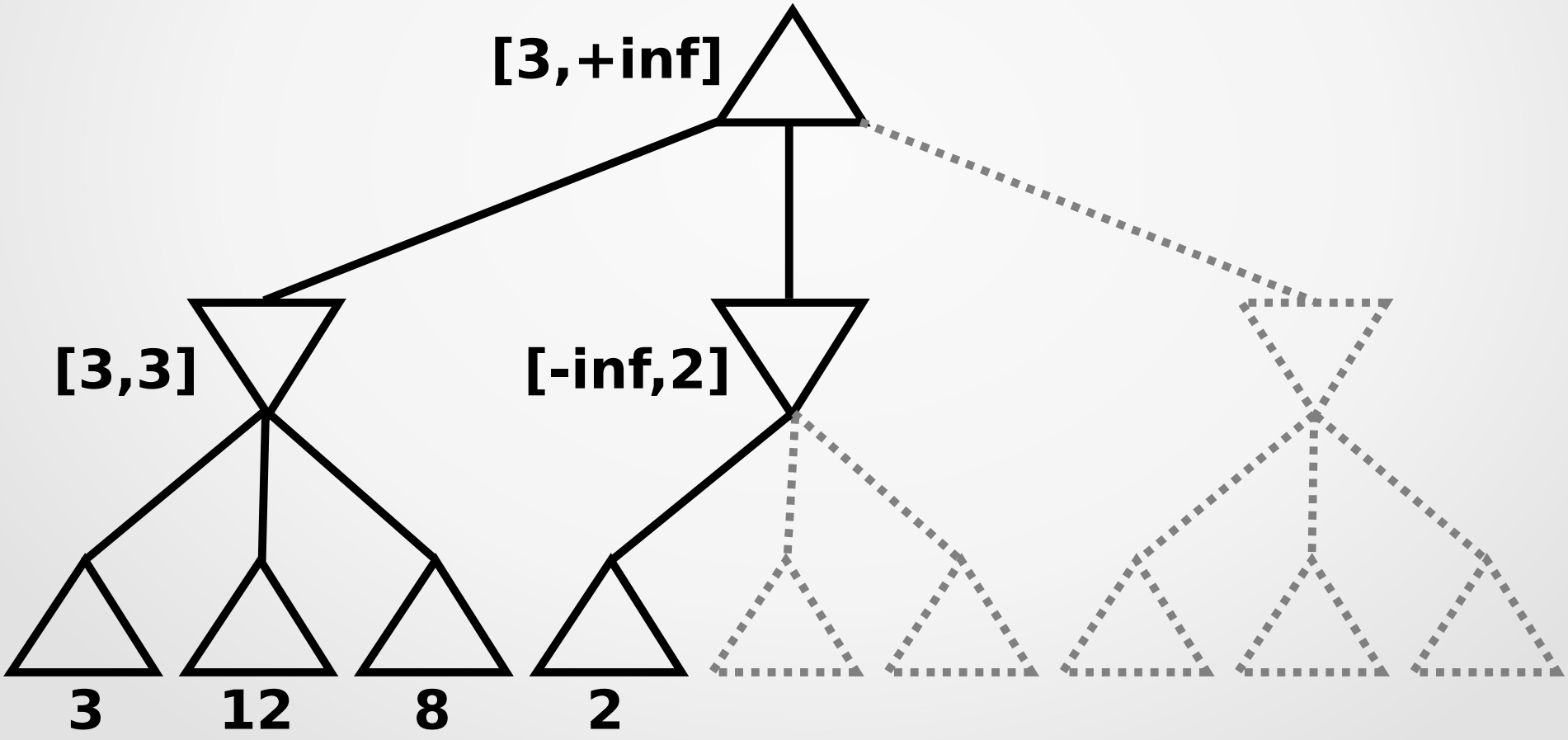
Descubriendo la Inteligencia Artificial – 112

Ejemplo



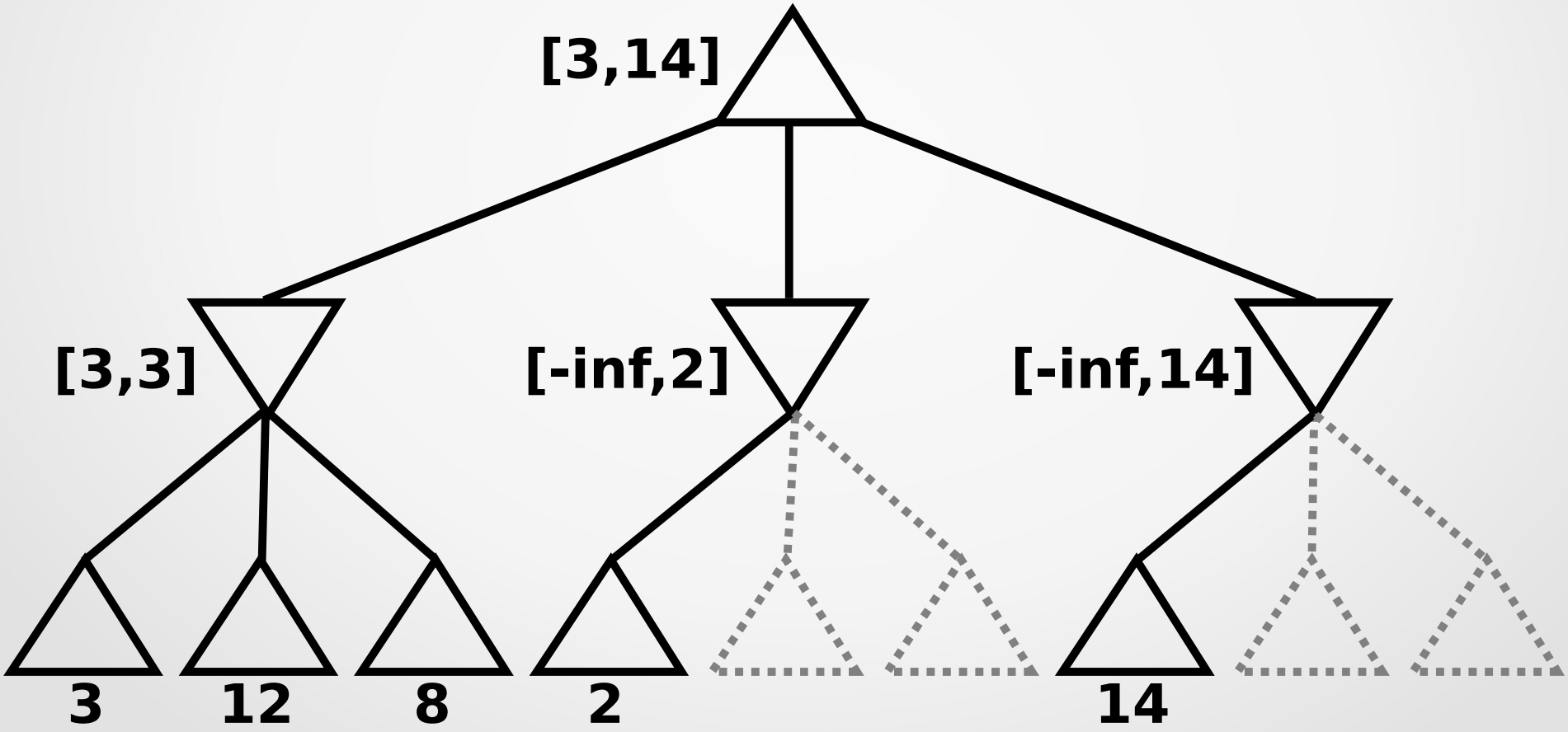
Descubriendo la Inteligencia Artificial – 112

Ejemplo



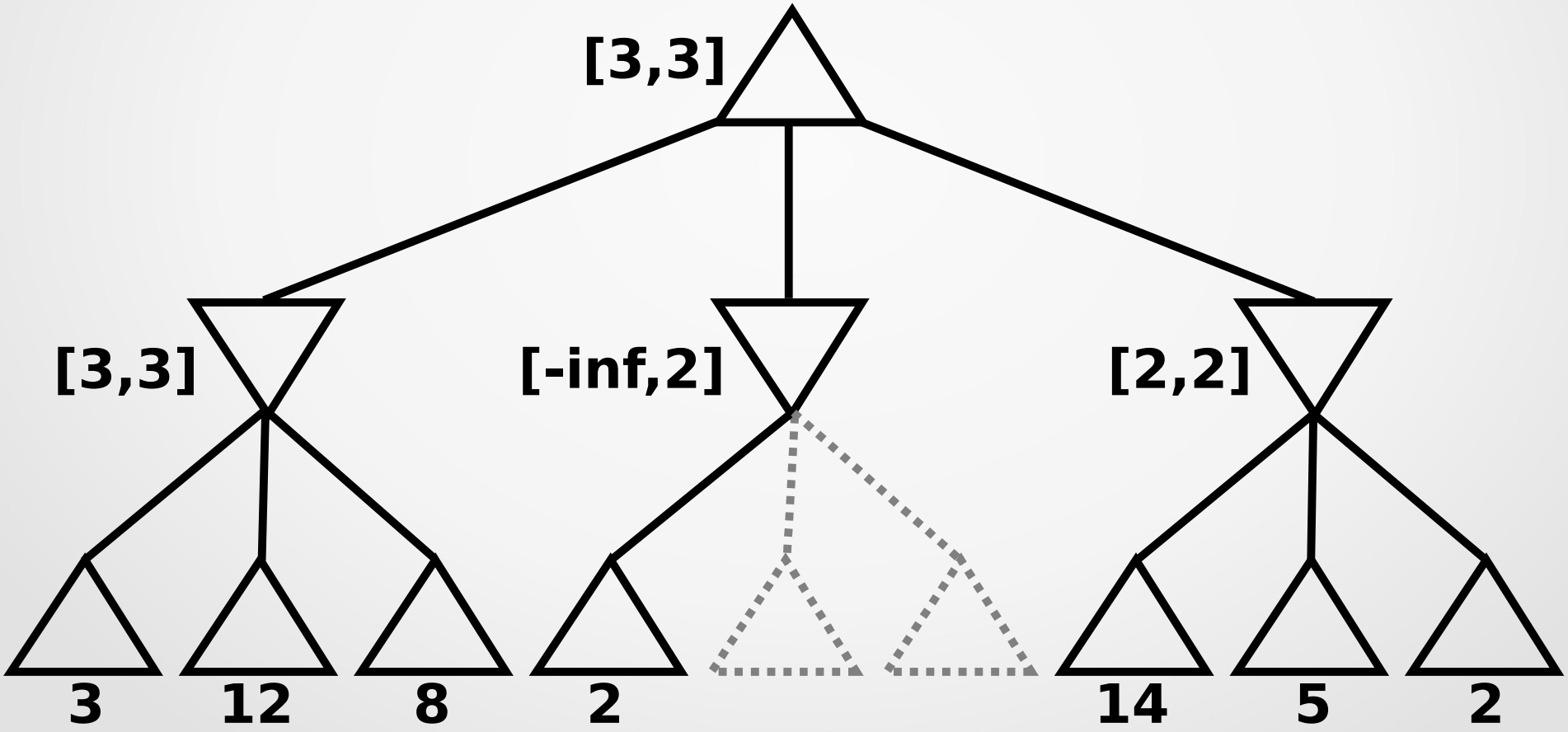
Descubriendo la Inteligencia Artificial – 112

Ejemplo



Descubriendo la Inteligencia Artificial – 112

Ejemplo



Valores

- Alfa: el valor de la mejor acción que se ha encontrado en cualquier punto de la búsqueda para Max.
- Beta: el valor de la mejor acción que se ha encontrado en cualquier punto de la búsqueda para Min.

Algoritmo

```
función ALFA-BETA(problema)
devuelve acción
  inicial ← problema.ESTADO-INICIAL
  (accion, valor) ← VALOR-MAX(problema,
                              inicial,
                              –inifinito,
                              +infinito)
  devolver accion
```

Descubriendo la Inteligencia Artificial – 112

```
funcion VALOR-MAX(problema, estado, alfa, beta)
devuelve (accion, valor)
  si problema.ES-OBJETIVO(estado) entonces
    devolver problema.UTILIDAD(estado)
  mayor-valor  ← –infinito
  mejor-accion ← nulo
  por cada accion en problema.ACCIONES(estado) hacer
    resultado ← problema.RESULTADO(estado, accion)
    utilidad  ← VALOR-MIN(problema, resultado, alfa, beta)
    si utilidad > mayor-valor entonces
      mayor-valor  ← utilidad
      mejor-accion ← accion
    si mayor-valor >= beta entonces
      devolver (mejor-accion, mayor-valor)
    si mayor-valor > alfa entonces
      alfa ← mayor-valor
  devolver (mejor-accion, mayor-valor)
```

Descubriendo la Inteligencia Artificial – 112

```
funcion VALOR-MIN(problema, estado, alfa, beta)
devuelve (accion, valor)
  si problema.ES-OBJETIVO(estado) entonces
    devolver problema.UTILIDAD(estado)
  menor-valor  ← +infinito
  mejor-accion ← nulo
  por cada accion en problema.ACCIONES(estado) hacer
    resultado ← problema.RESULTADO(estado, accion)
    utilidad  ← VALOR-MAX(problema, resultado, alfa, beta)
    si utilidad < menor-valor entonces
      menor-valor  ← utilidad
      mejor-accion ← accion
    si menor-valor <= alfa entonces
      devolver (mejor-accion, mayor-valor)
    si menor-valor > beta entonces
      beta ← menor-valor
  devolver (mejor-accion, menor-valor)
```

Complejidad

Reducción: depende del orden.

- Ordenar Todo: $O(b^{m/2})$
  Ramificación Media: $b^{1/2}$
  Se alcanza doble de profundidad.
  (Problema: es casi imposible)
- Orden Aleatorio: $O(b^{3m/4})$ de media.

Mejoras

Tablas de Transposición:

- Se guarda algunas permutaciones de conjunto de acciones que llevan al mismo resultado.
- Similar al conjunto explorado en búsqueda no informada.
- Ahorra muchos cálculos.

Aprendizaje

Refuerzo: probar primero acciones que fueron bien en el pasado:

- Juegos anteriores.
- Búsqueda en profundidad iterativa:
  - Se guarda las mejores acciones de cada iteración.
  - Tiempo límite: siempre una acción.

Descubriendo la Inteligencia Artificial – 112

Despedida

- ¡Participa y Colabora!
- Dale a "Me Gusta"
- Suscríbete al Canal
- Deja tus Comentarios

José Luis Iglesias Feria