



Universidad de Oviedo

# Optimización de Algoritmos de Búsqueda en Grafos: Implementación y Comparación de Rendimiento en FPGA

*Autor:* Rodríguez López, Alejandro

*Tutor:* Palacios Alonso, Juan José

2024-07-24

# Introducción

Problemas de complejidad alta

Algoritmos de búsqueda (A\*)

Coste computacional alto

Problema clásico (Job Shop Scheduling)

Heurísticos, paralelismo y FPGA

# Contenidos

1. Job Shop Scheduling Problem (JSP)
2. Algoritmo A\*
3. Funciones heurísticas
4. Estrategias de paralelismo
5. FPGA
6. Conclusiones y cierre

# Job Shop Scheduling Problem (JSP)

¿Qué recibe el JSP?

RDY

# Job Shop Scheduling Problem (JSP)

```
type Task = {  
    duration: number,  
    qualifiedWorker: number  
};  
  
const jobs: Task[][] = [  
    [  
        ({duration: 2, qualifiedWorker: 0}),  
        ({duration: 5, qualifiedWorker: 1}),  
        ({duration: 1, qualifiedWorker: 2})  
    ], [  
        ({duration: 3, qualifiedWorker: 1}),  
        ({duration: 3, qualifiedWorker: 2}),  
        ({duration: 3, qualifiedWorker: 0})  
    ]  
];
```

RDY

# Job Shop Scheduling Problem (JSP): Tareas (Tasks)

¿Qué retorna el JSP?

RDY

# Job Shop Scheduling Problem (JSP): Soluciones

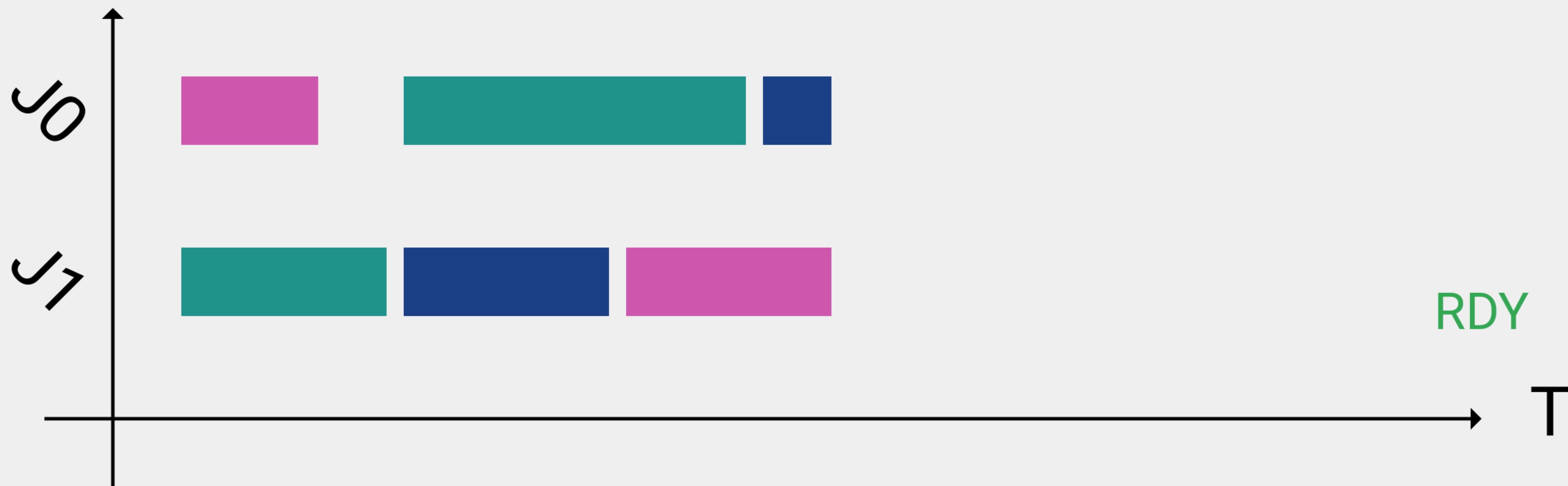
## Solución óptima

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(0), (3), (8)],  
  [(0), (3), (6)]  
]
```



# Job Shop Scheduling Problem (JSP): Soluciones

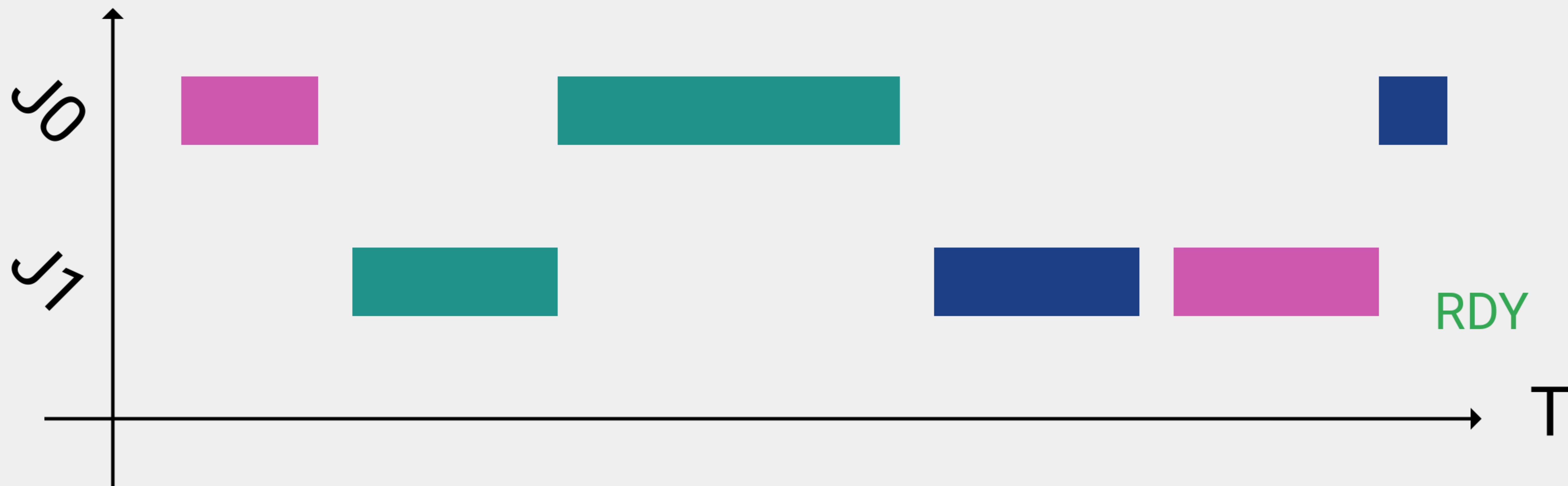
## Otra solución

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(0), (5), (16)],  
  [(2), (10), (13)]  
]
```



# Job Shop Scheduling Problem (JSP): Sucesores

```
// JOBS - TRABAJOS
[
  [(2, 0), (5, 1), (1, 2)],
  [(3, 1), (3, 2), (3, 0)]
]

// SCHEDULE - PLANIFICACIÓN
[
  [(-1), (-1), (-1)],
  [(-1), (-1), (-1)]
]
```

¿Cómo obtiene los estados sucesores?

RDY

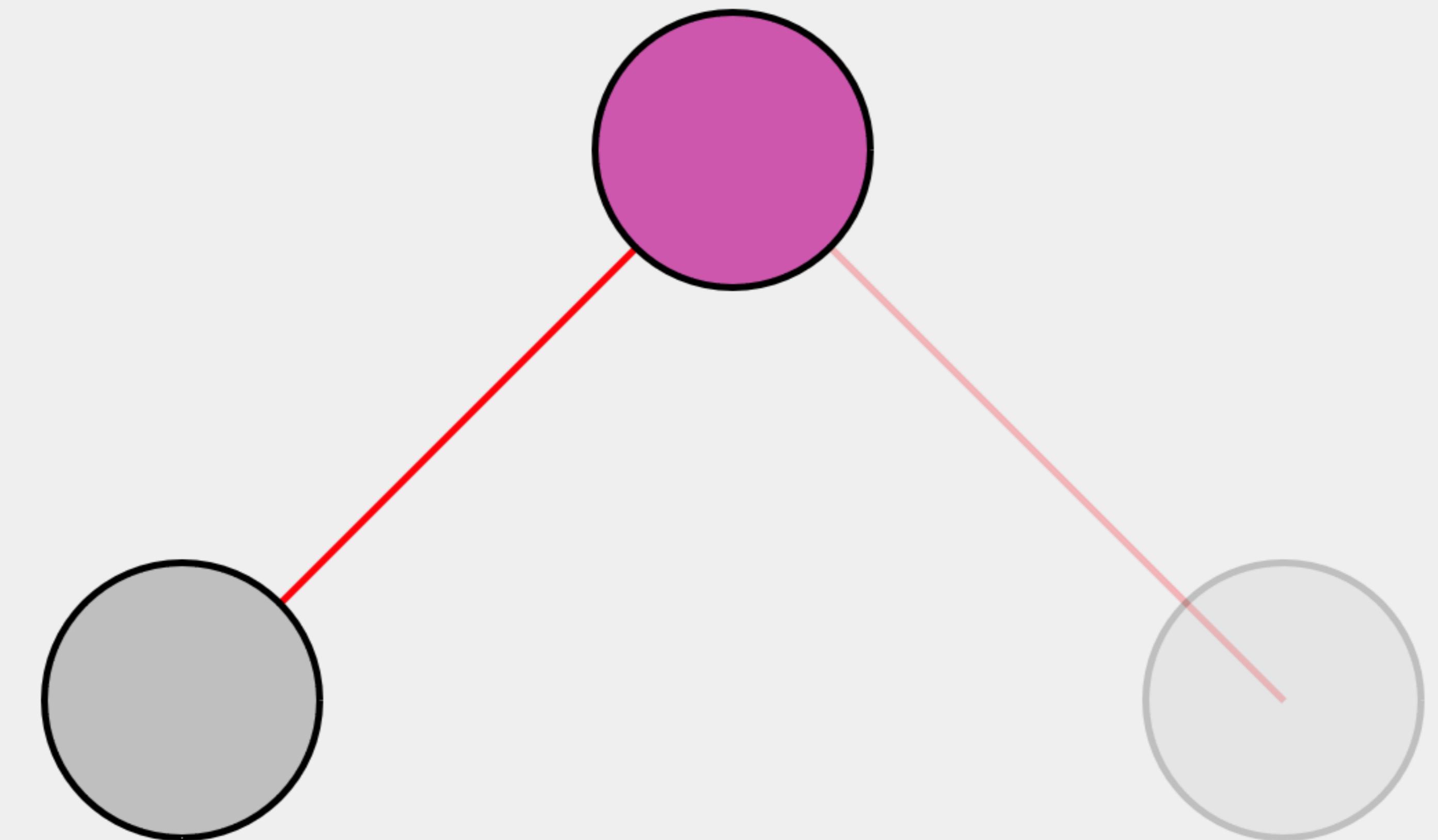
# Job Shop Scheduling Problem (JSP): Sucesores

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(0), (-1), (-1)],  
  [(-1), (-1), (-1)]  
]
```



RDY

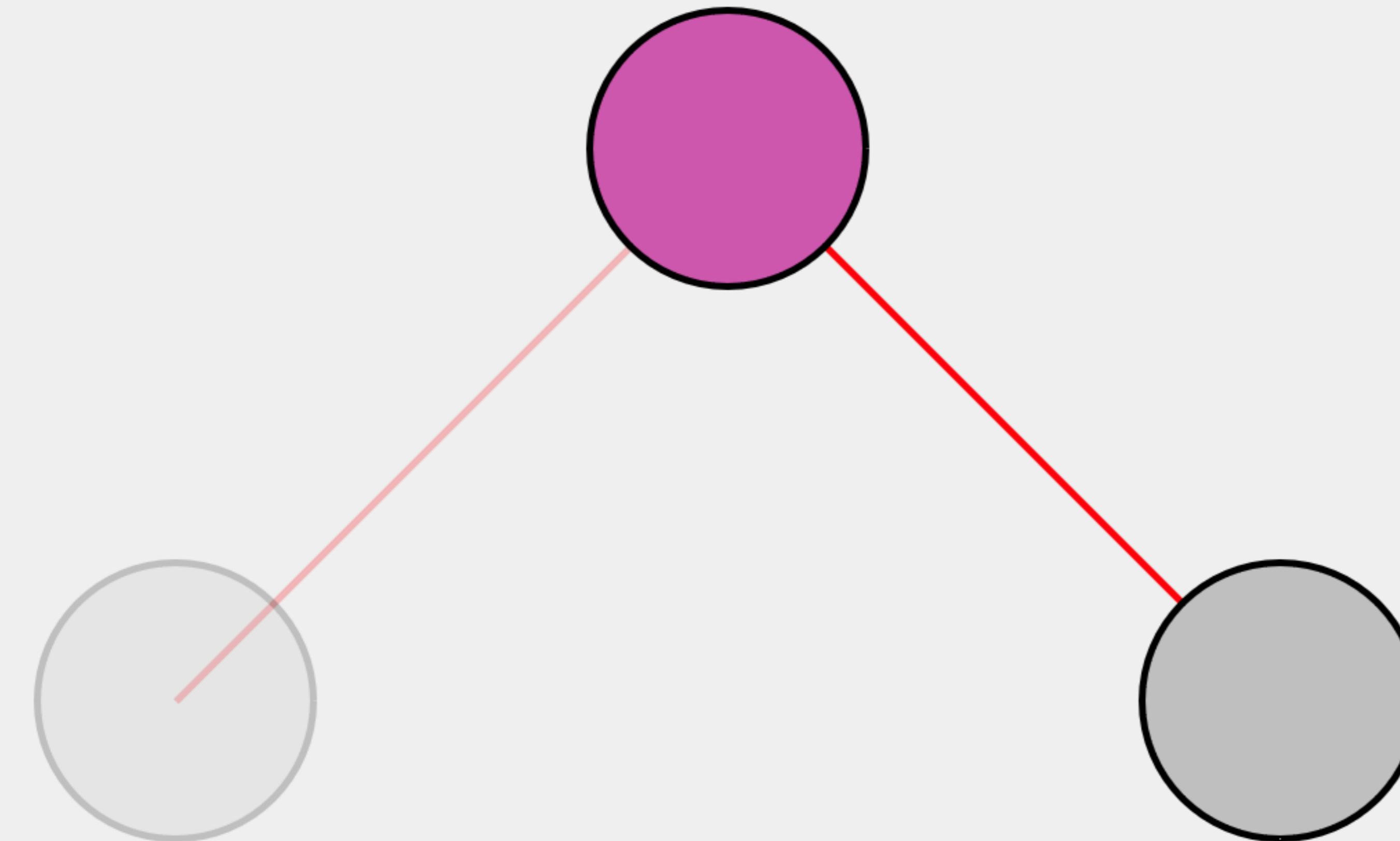
# Job Shop Scheduling Problem (JSP): Sucesores

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(-1), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```

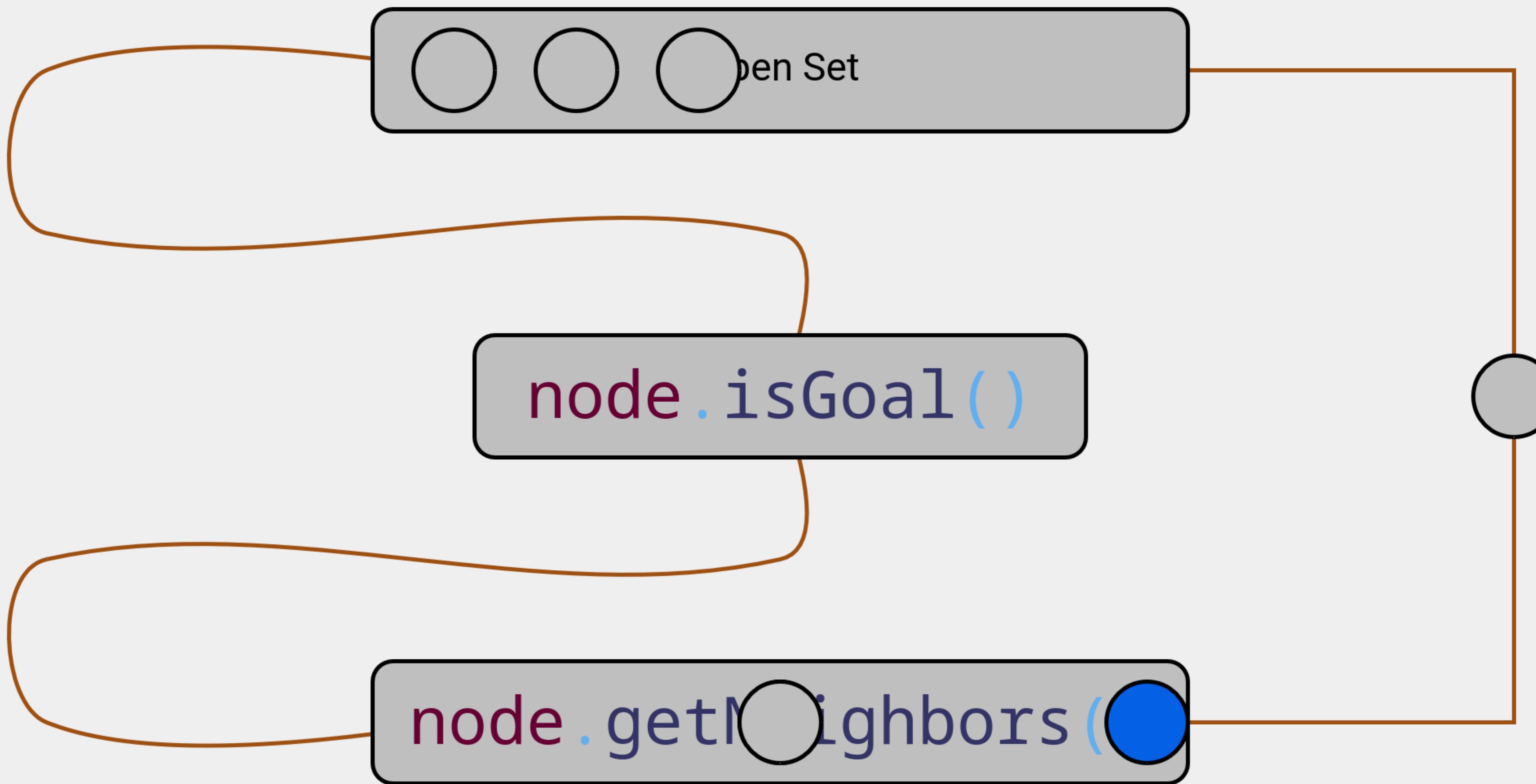


RDY

# Algoritmo A\*

¿Cómo obtiene A\* la solución?

# Algoritmo A\*: Repetir hasta objetivo



ANM

# Costes y Función Heurística

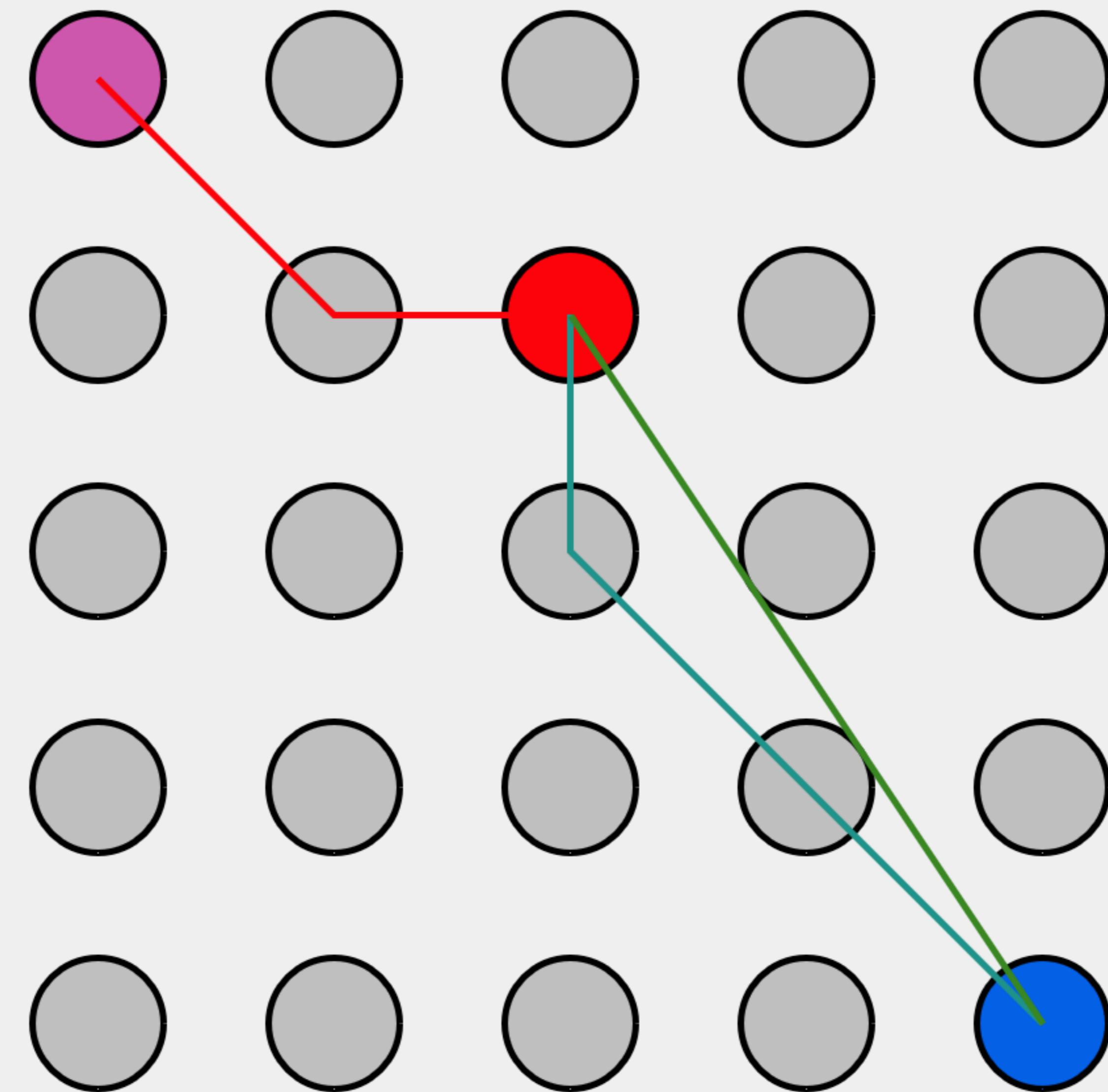
¿Cómo se ordena el open set?

# Costes y Función Heurística

Recorrido (G)

Solución

Estimación (H)



RDY

# Funciones Heurísticas

¿Cómo se implementa un heurístico?

RDY

# Funciones Heurísticas

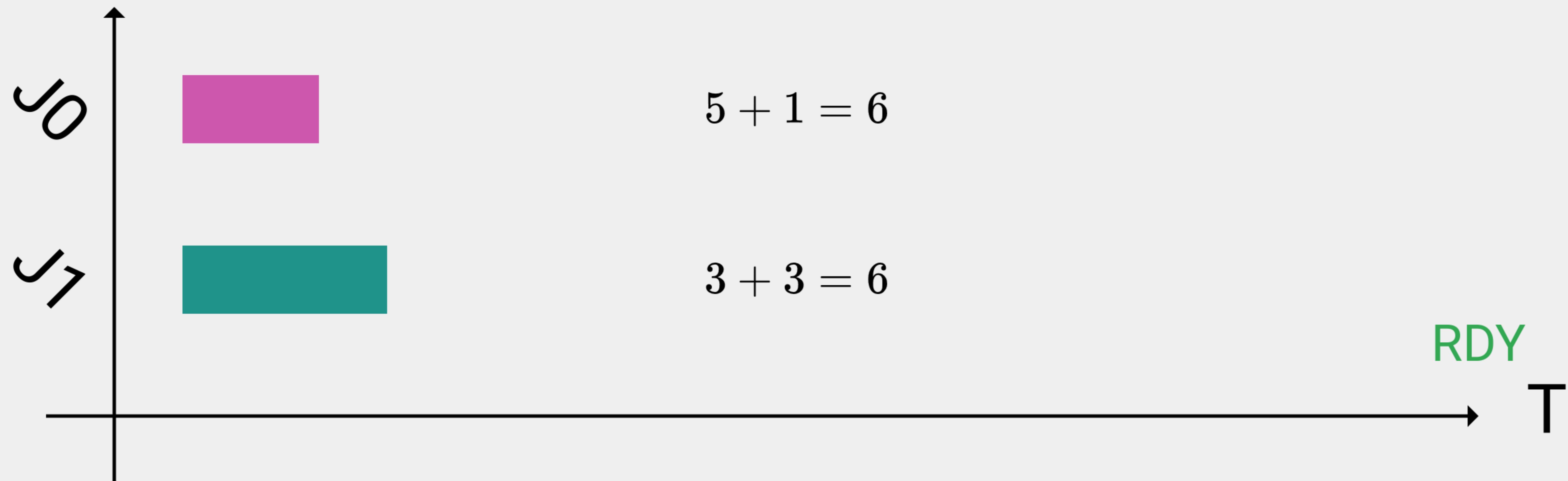
Heurístico Óptimo / Lento (cota inferior)

// JOBS - TRABAJOS

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

// SCHEDULE - PLANIFICACIÓN

```
[  
  [(0), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```



# Funciones Heurísticas

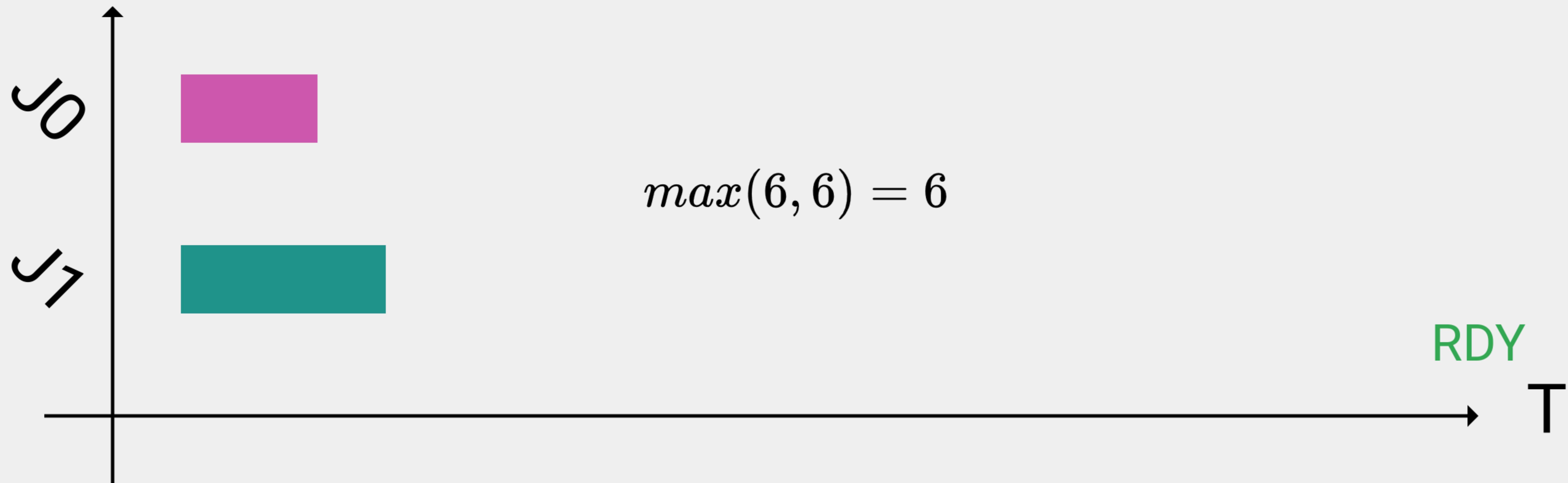
Heurístico Óptimo / Lento (cota inferior)

// JOBS - TRABAJOS

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

// SCHEDULE - PLANIFICACIÓN

```
[  
  [(0), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```



# Funciones Heurísticas

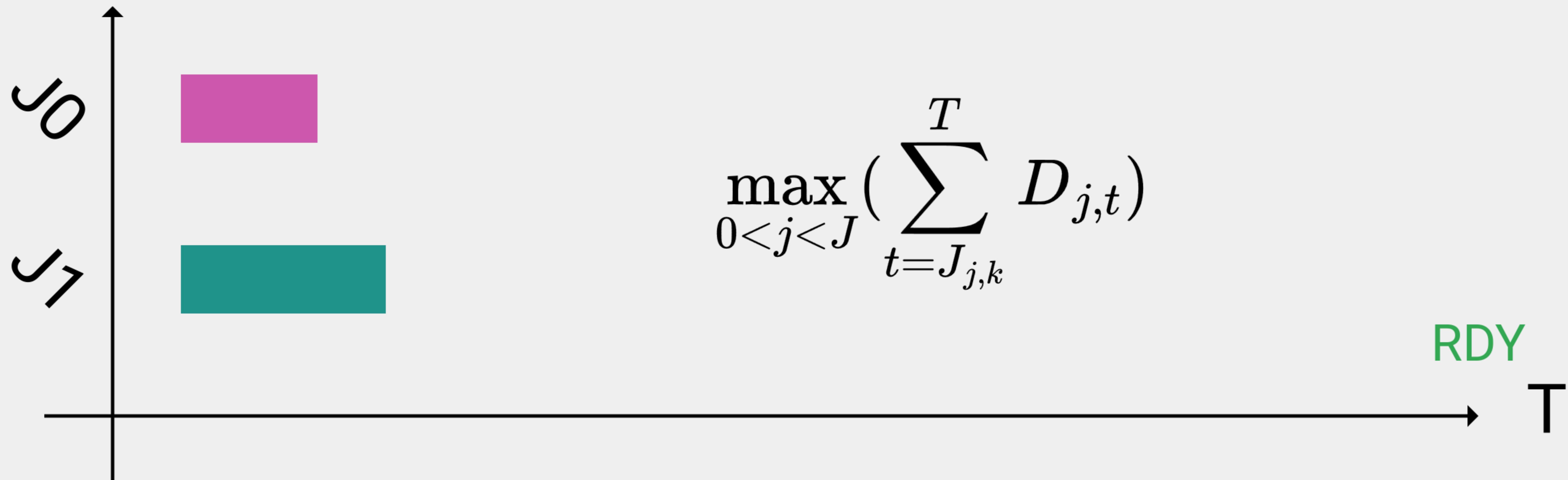
Heurístico Óptimo / Lento (cota inferior)

// JOBS - TRABAJOS

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

// SCHEDULE - PLANIFICACIÓN

```
[  
  [(0), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```



# Funciones Heurísticas

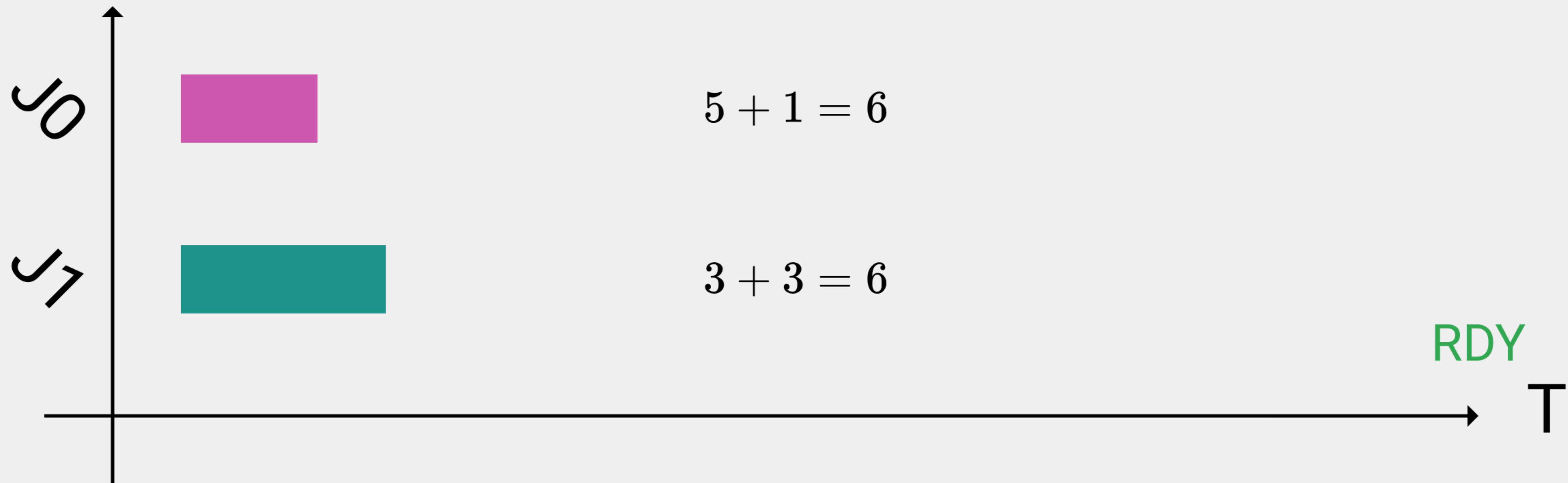
## Heurístico Rápido (cota superior)

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(0), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```



# Funciones Heurísticas

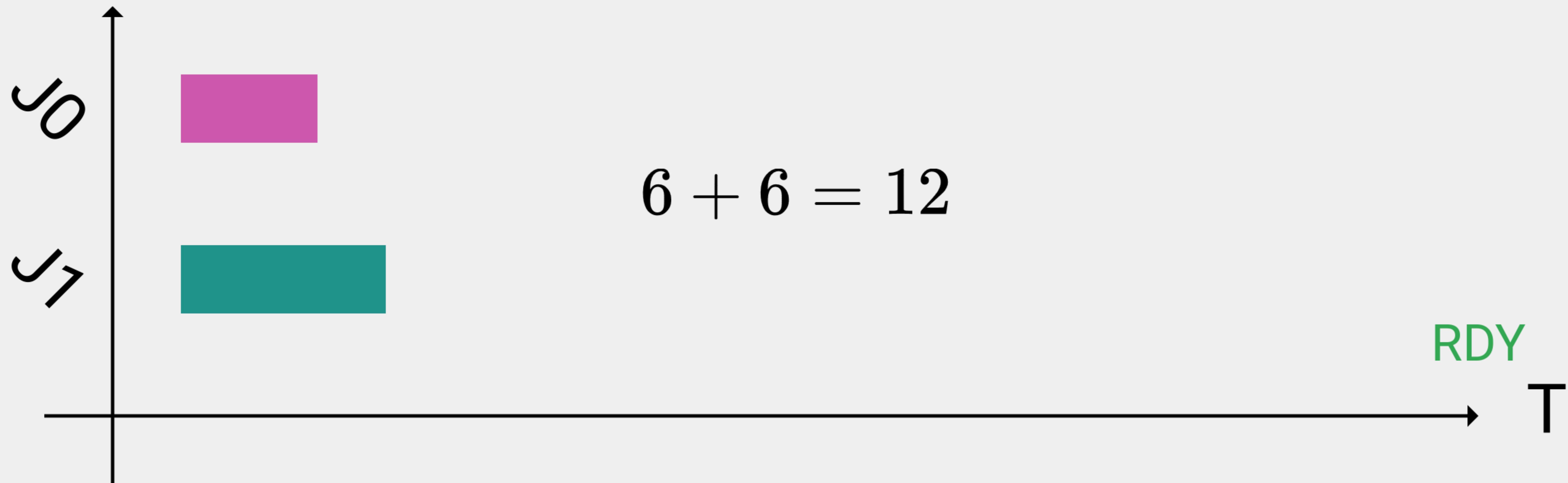
## Heurístico Rápido (cota superior)

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(0), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```



# Funciones Heurísticas

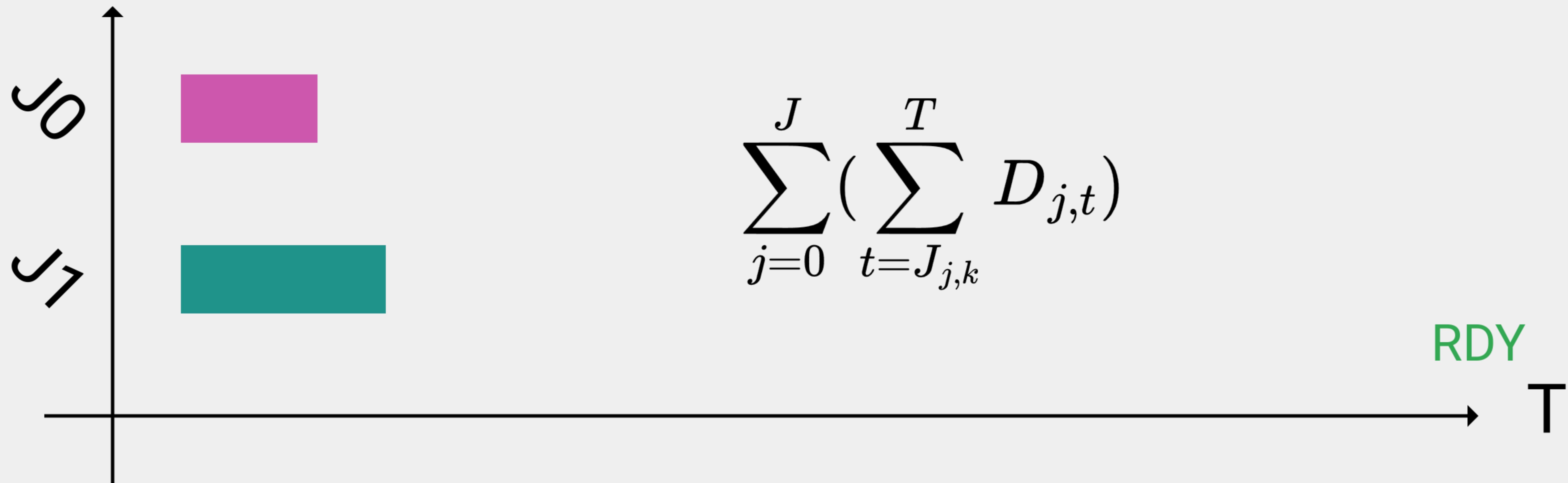
## Heurístico Rápido (cota superior)

```
// JOBS - TRABAJOS
```

```
[  
  [(2, 0), (5, 1), (1, 2)],  
  [(3, 1), (3, 2), (3, 0)]  
]
```

```
// SCHEDULE - PLANIFICACIÓN
```

```
[  
  [(0), (-1), (-1)],  
  [(0), (-1), (-1)]  
]
```



# Comparativa de Heurísticos

¿Cómo se comparan los dos heurísticos?

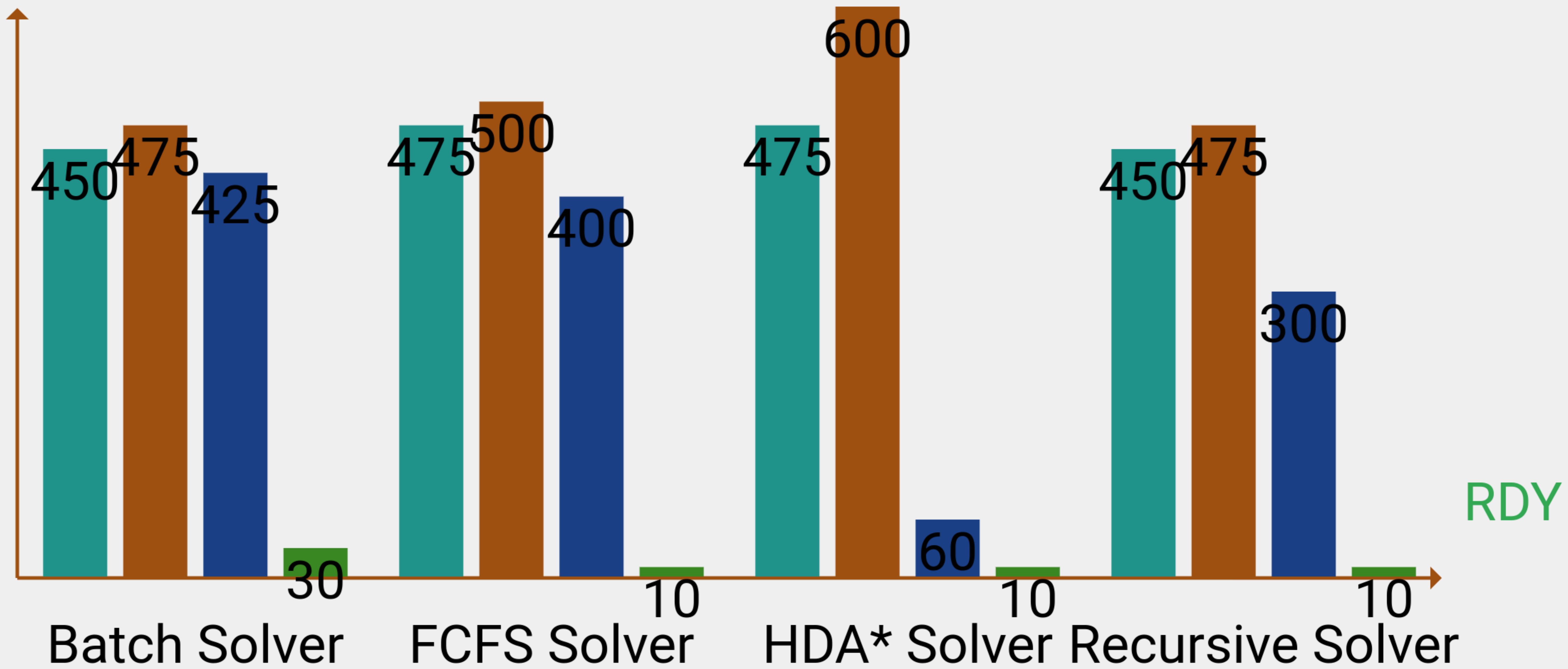
RDY

# Comparativa de Heurísticos

(Tiempo de ejecución x100)

8 HILOS

- (M) Lento
- (M) Rápido
- (R) Lento
- (R) Rápido

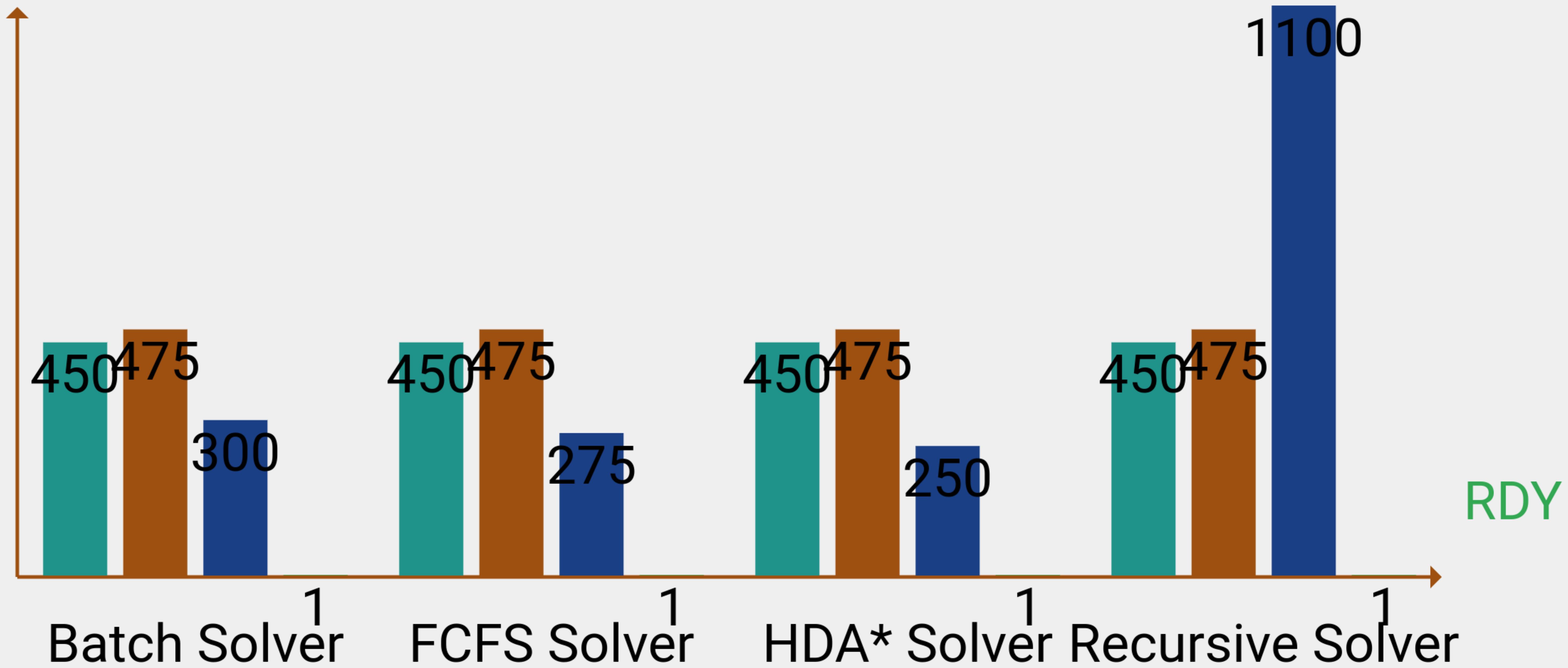


# Comparativa de Heurísticos

(Tiempo de ejecución x100)

1 HILO

(M) Lento  
(M) Rápido  
(R) Lento  
(R) Rápido

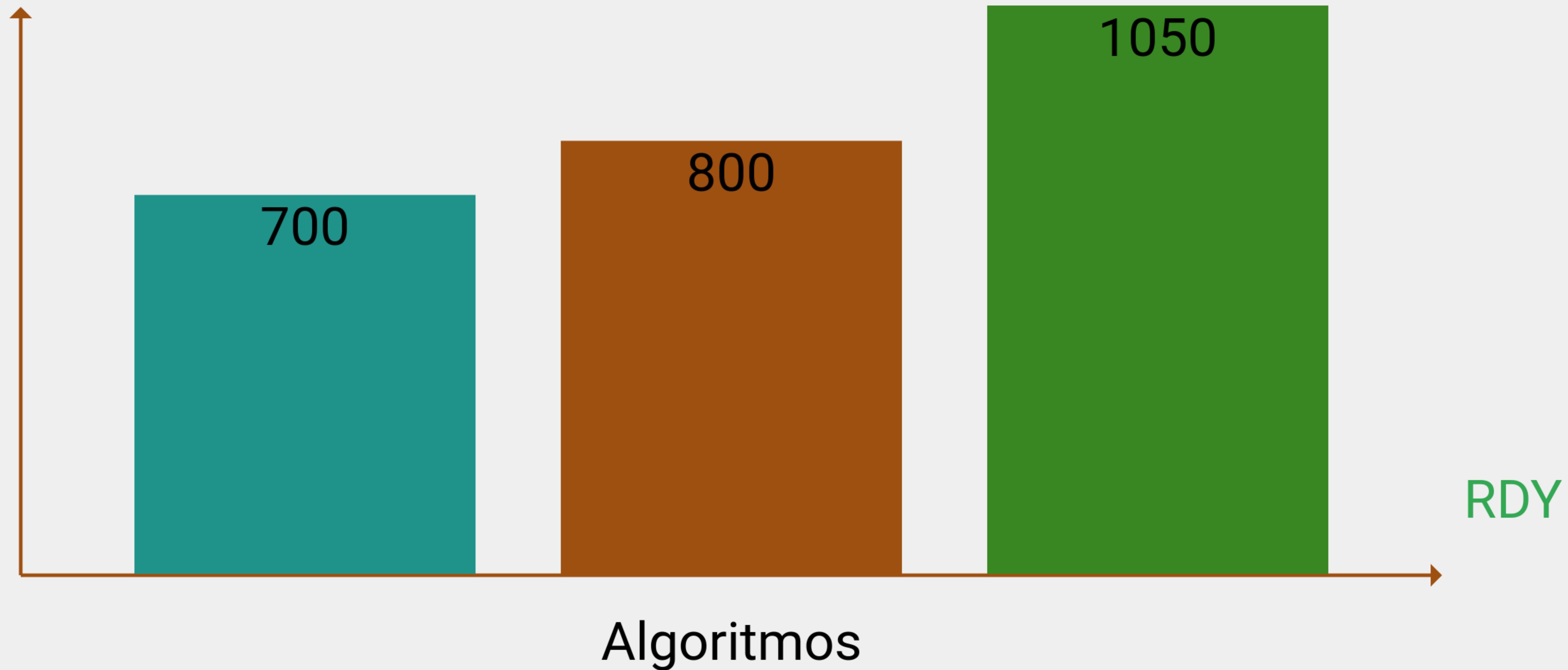


# Comparativa de Heurísticos

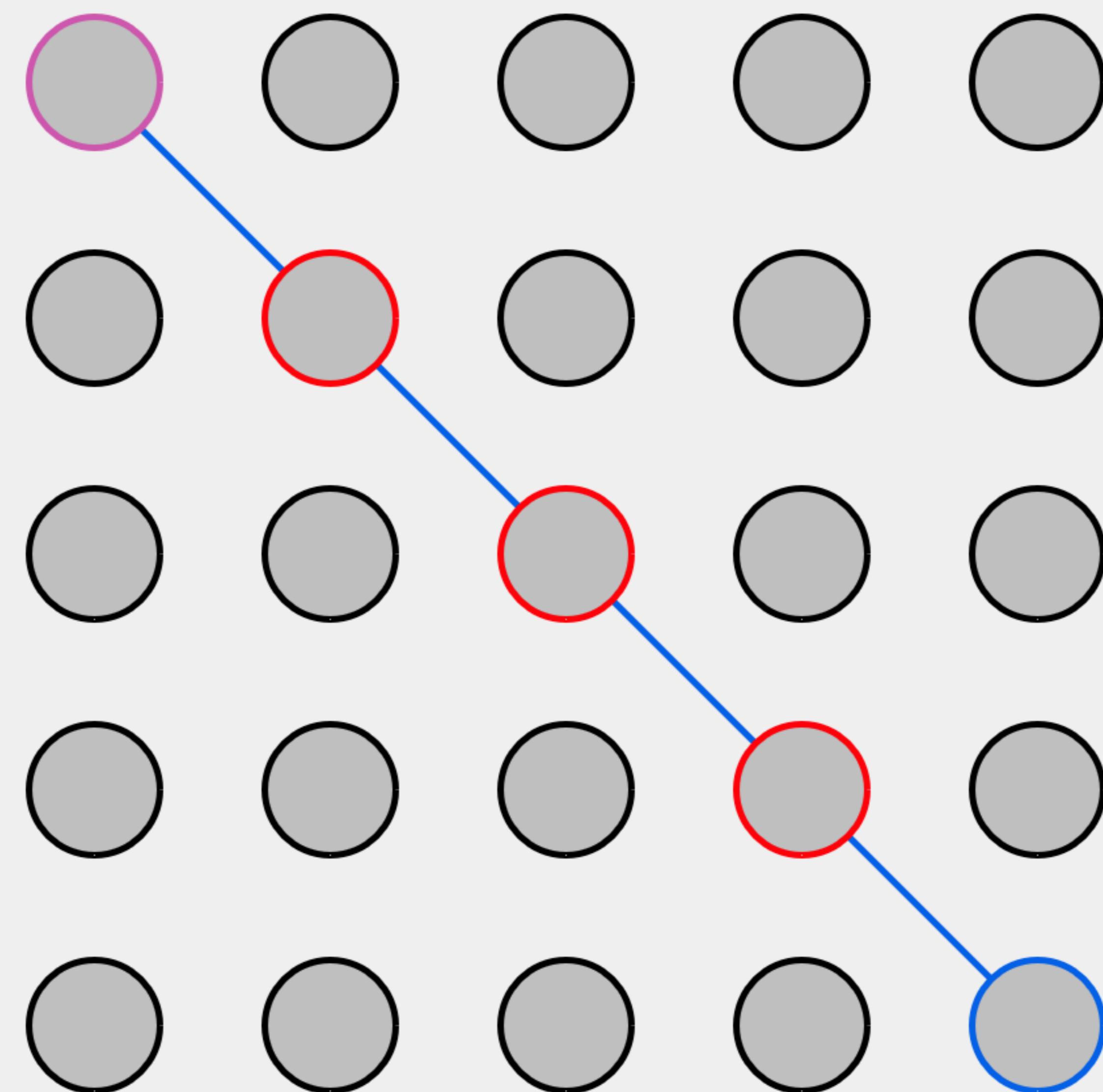
(Tiempo de ejecución x100)

Makespan

Óptimo / Lento  
Rápido  
Random

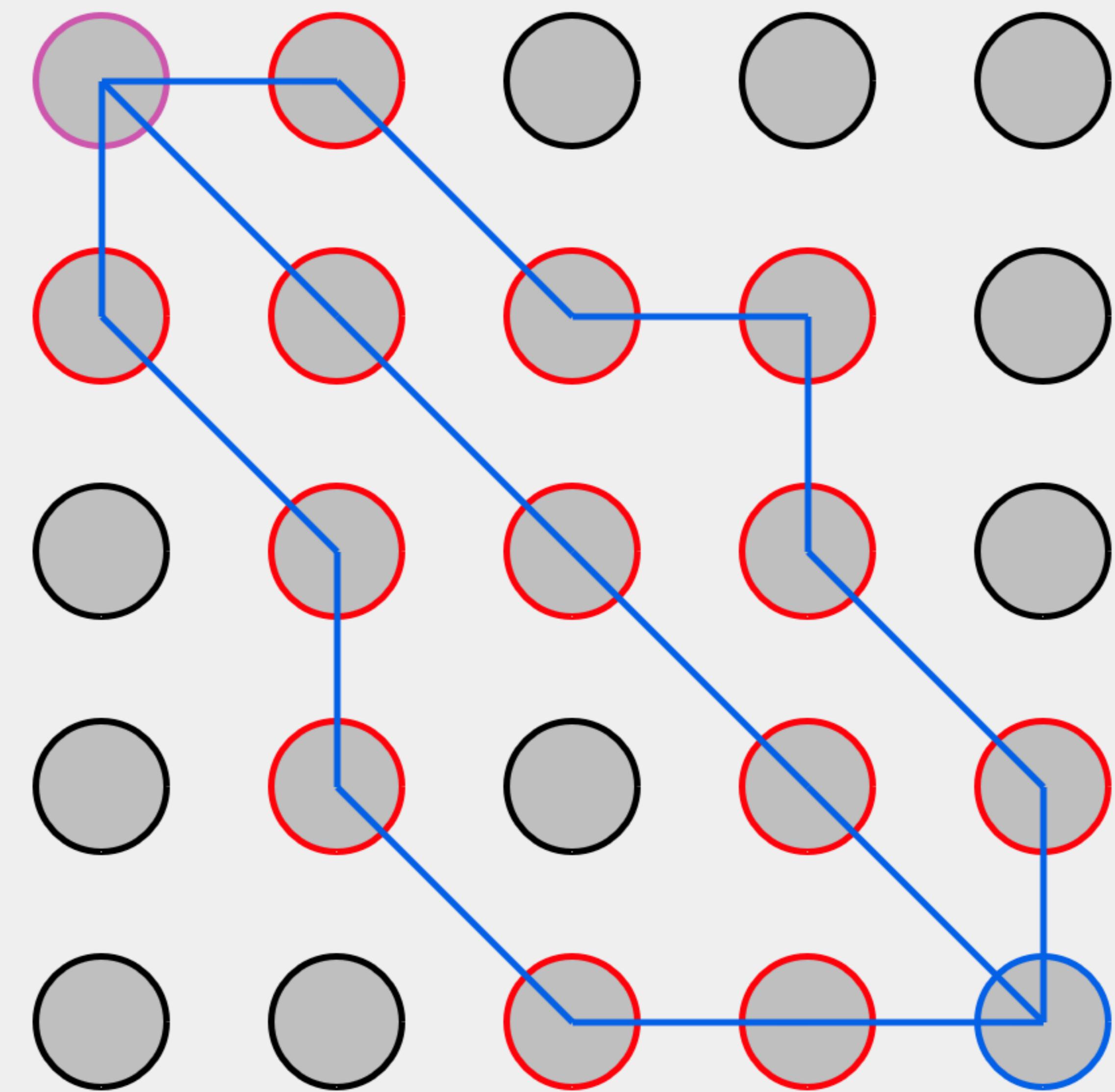


# A\* SINGLETHREAD SIMULATION



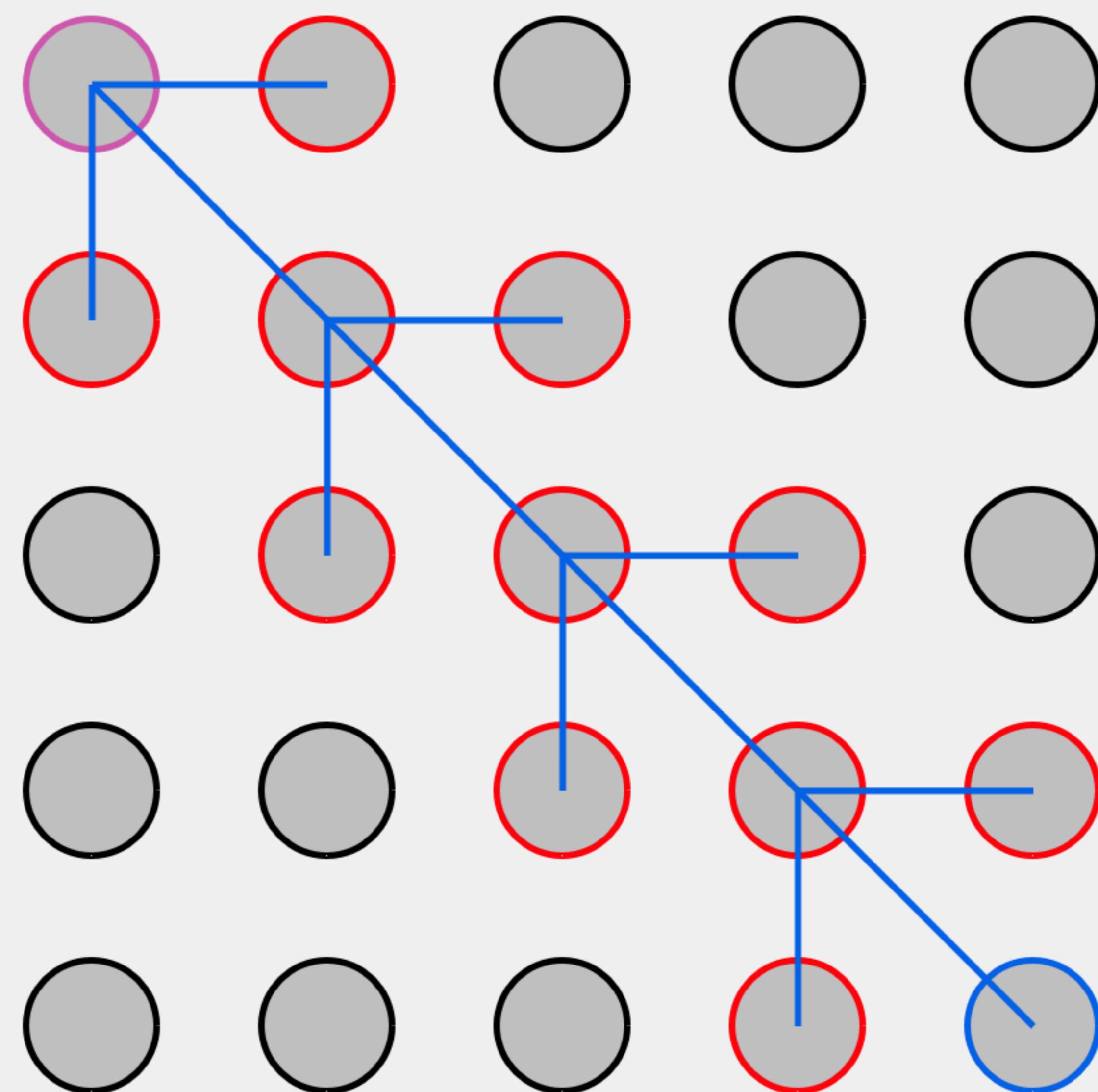
RDY

# RECURSIVE SIMULATION



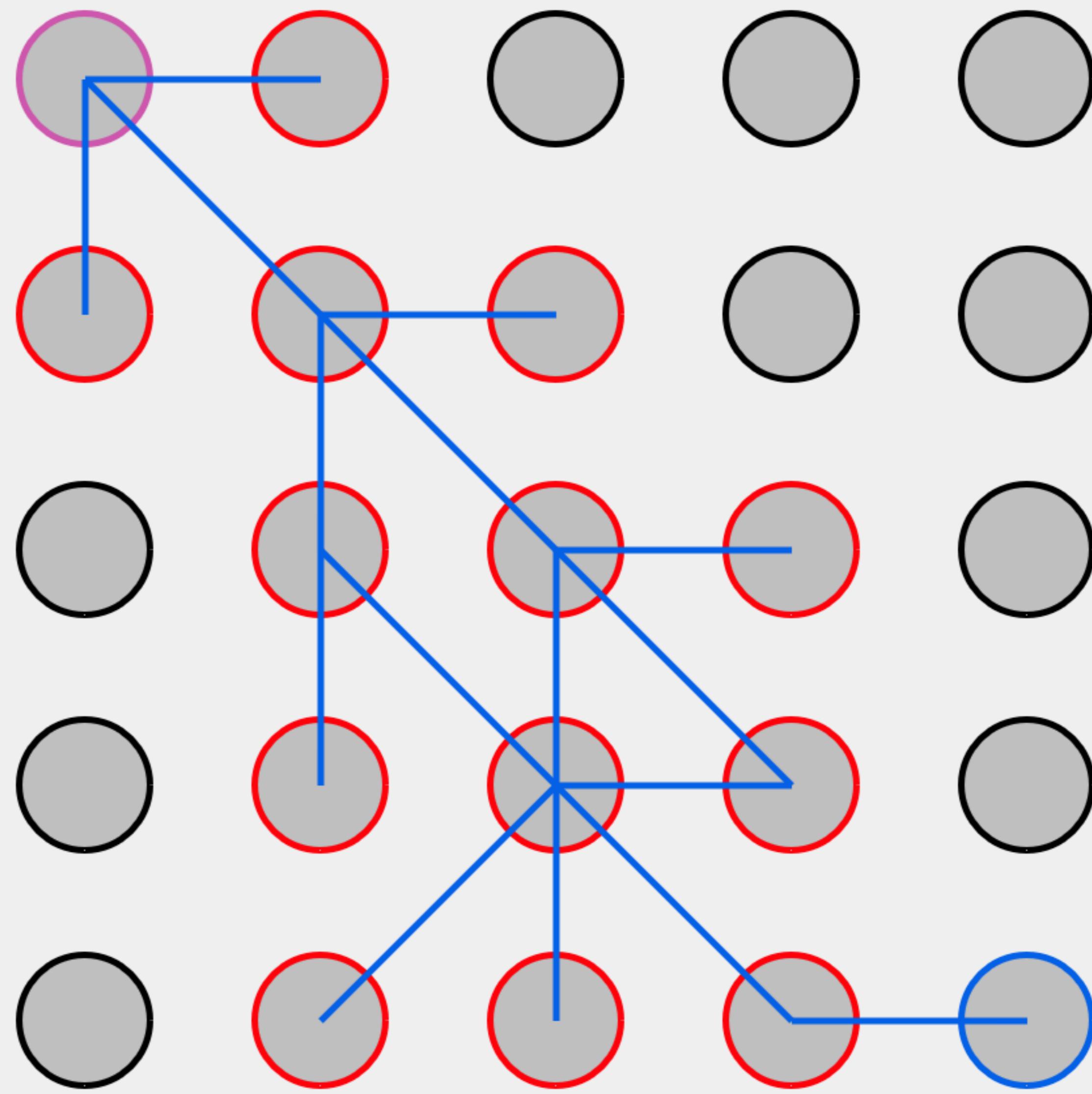
RDY

# BATCH SIMULATION



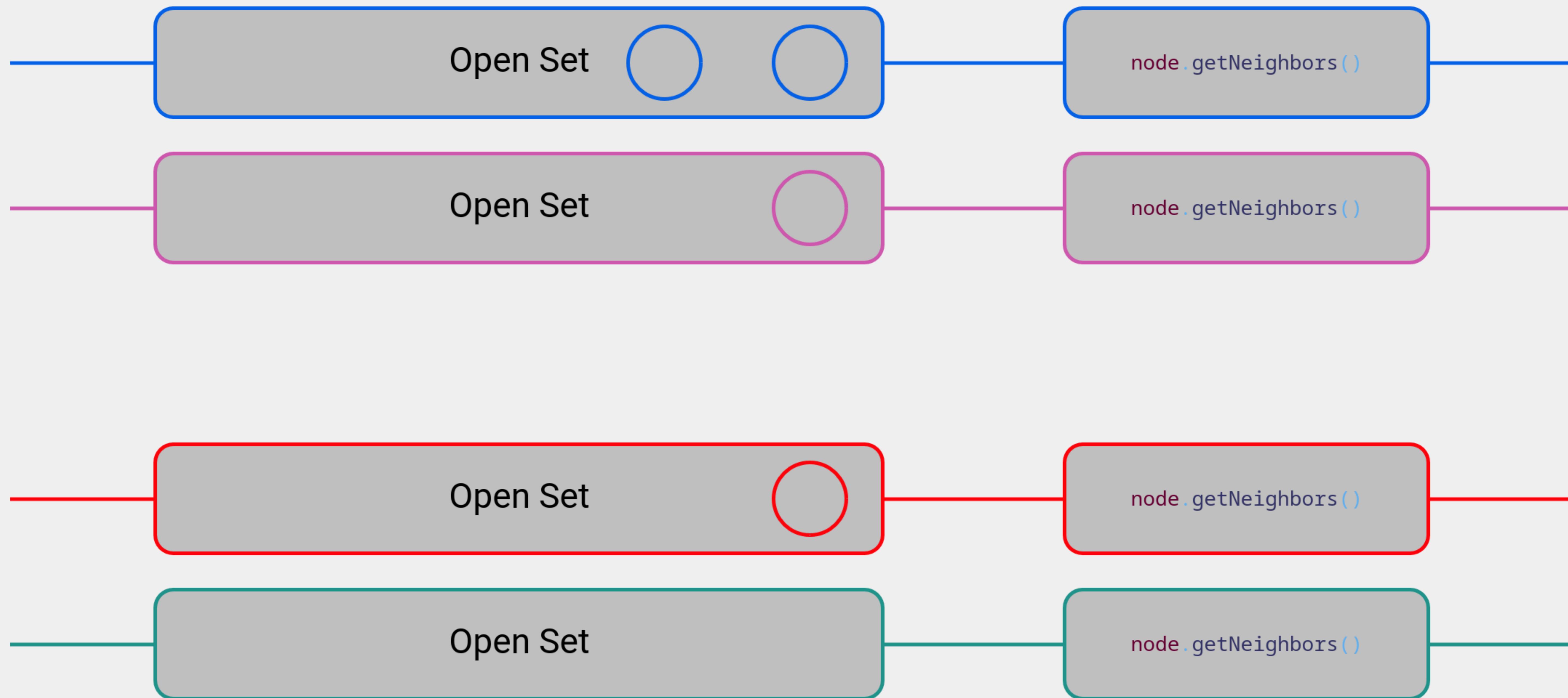
RDY

# FCFS SIMULATION



RDY

# HDA\* SIMULATION

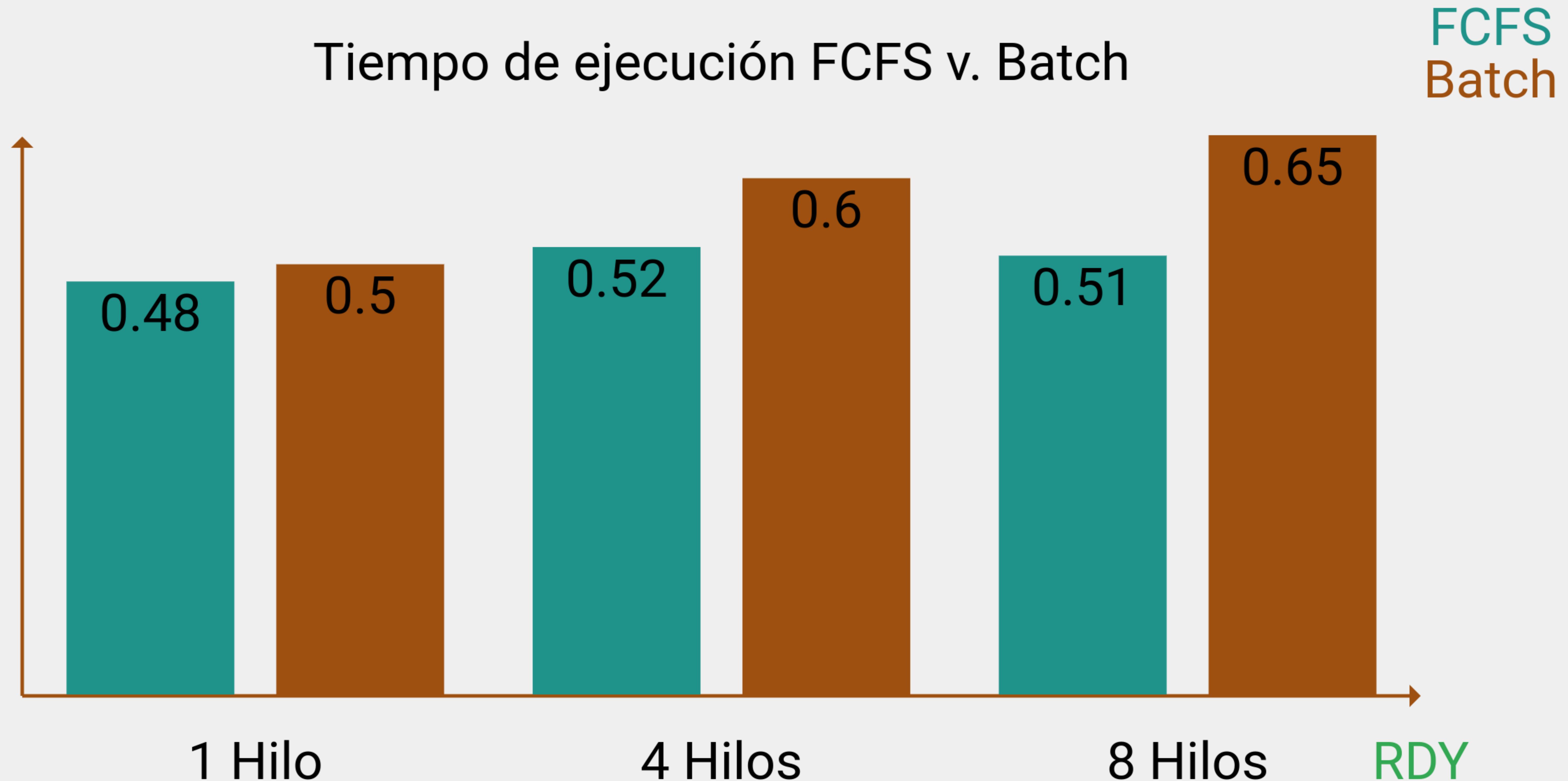


RDY

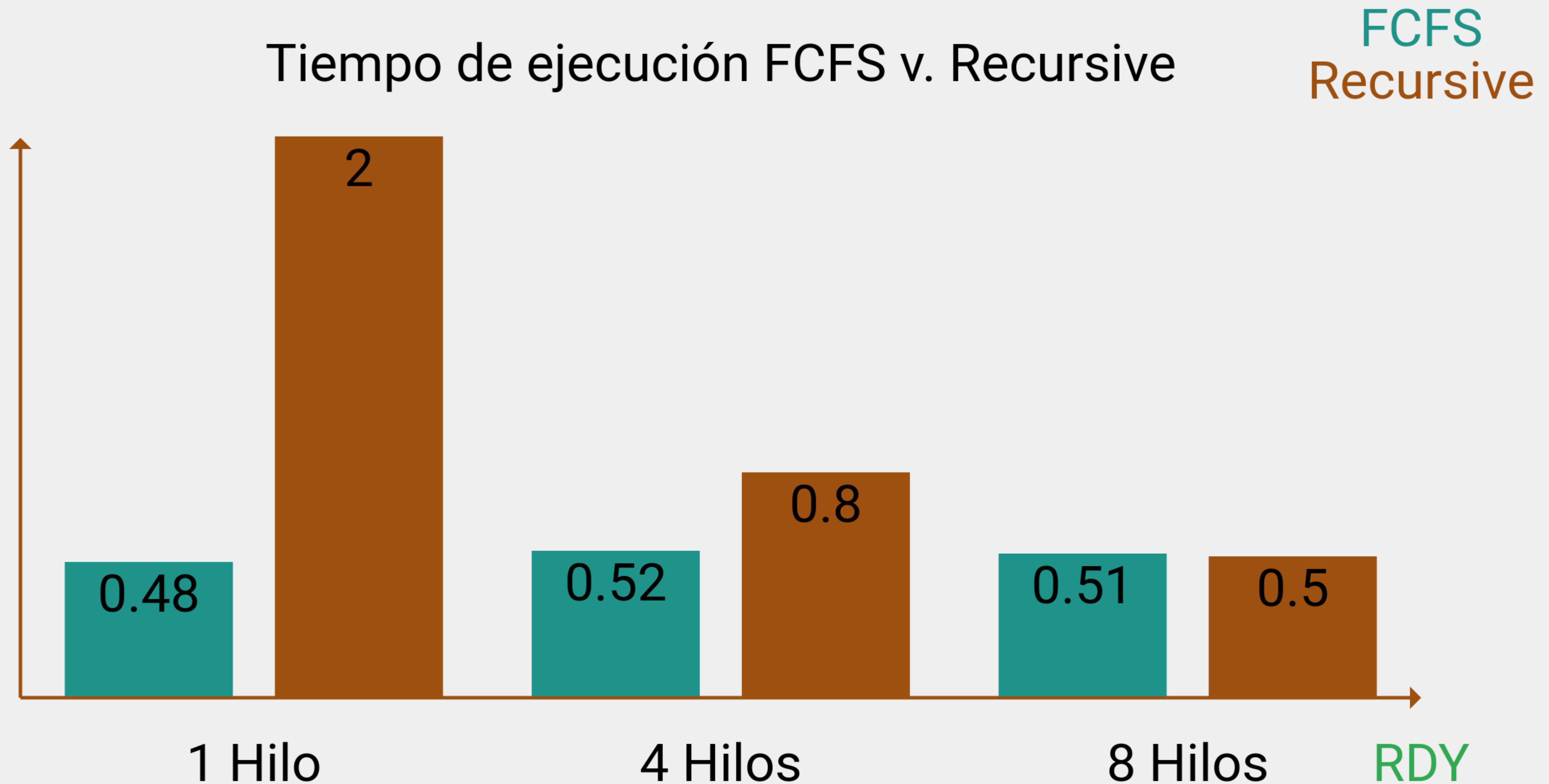
# Comparativa de Algoritmos

¿Cómo se comparan los algoritmos?

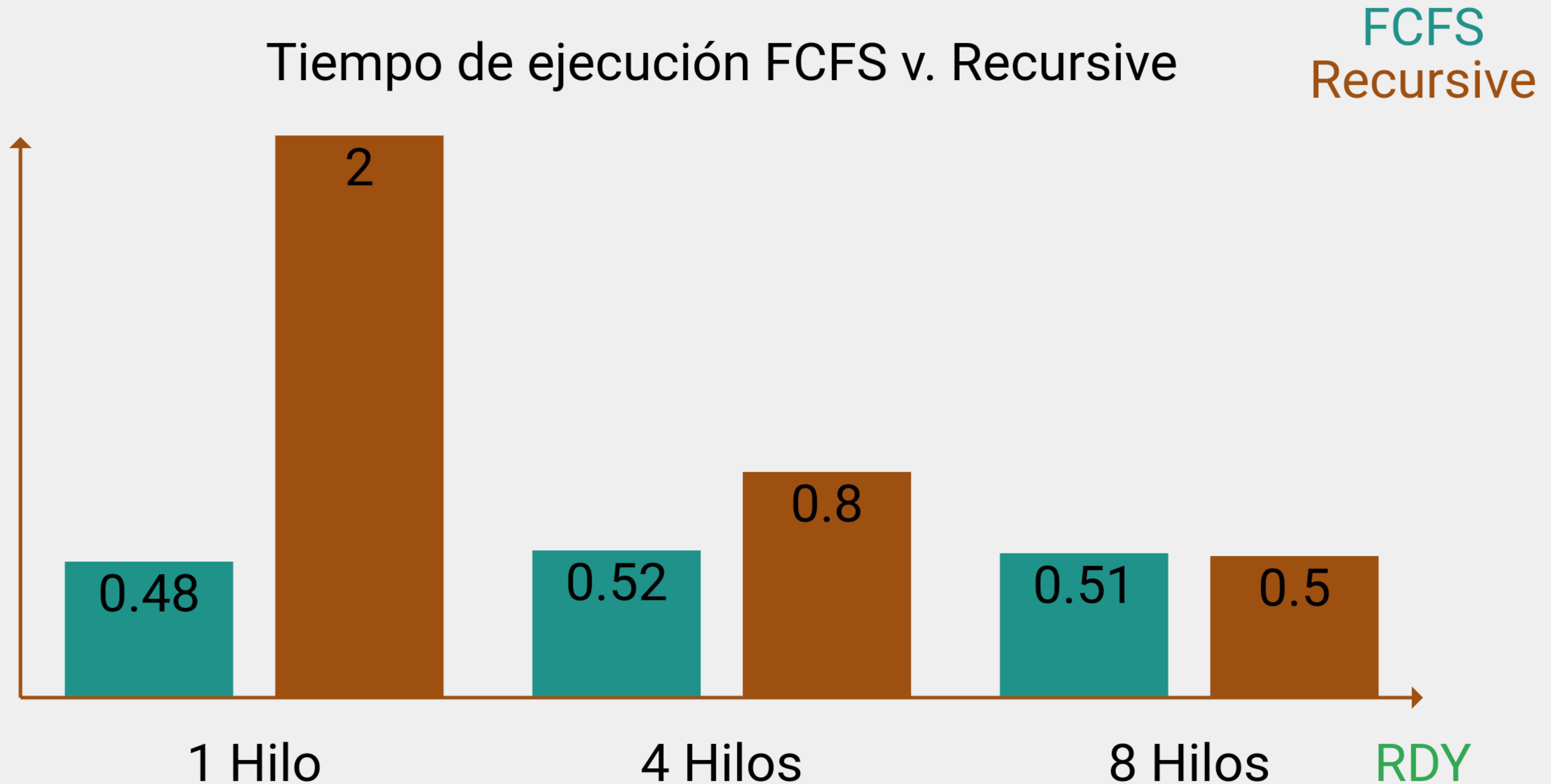
# Comparativa de Algoritmos: FCFS v. Batch



# Comparativa de Algoritmos: FCFS v. HDA\*



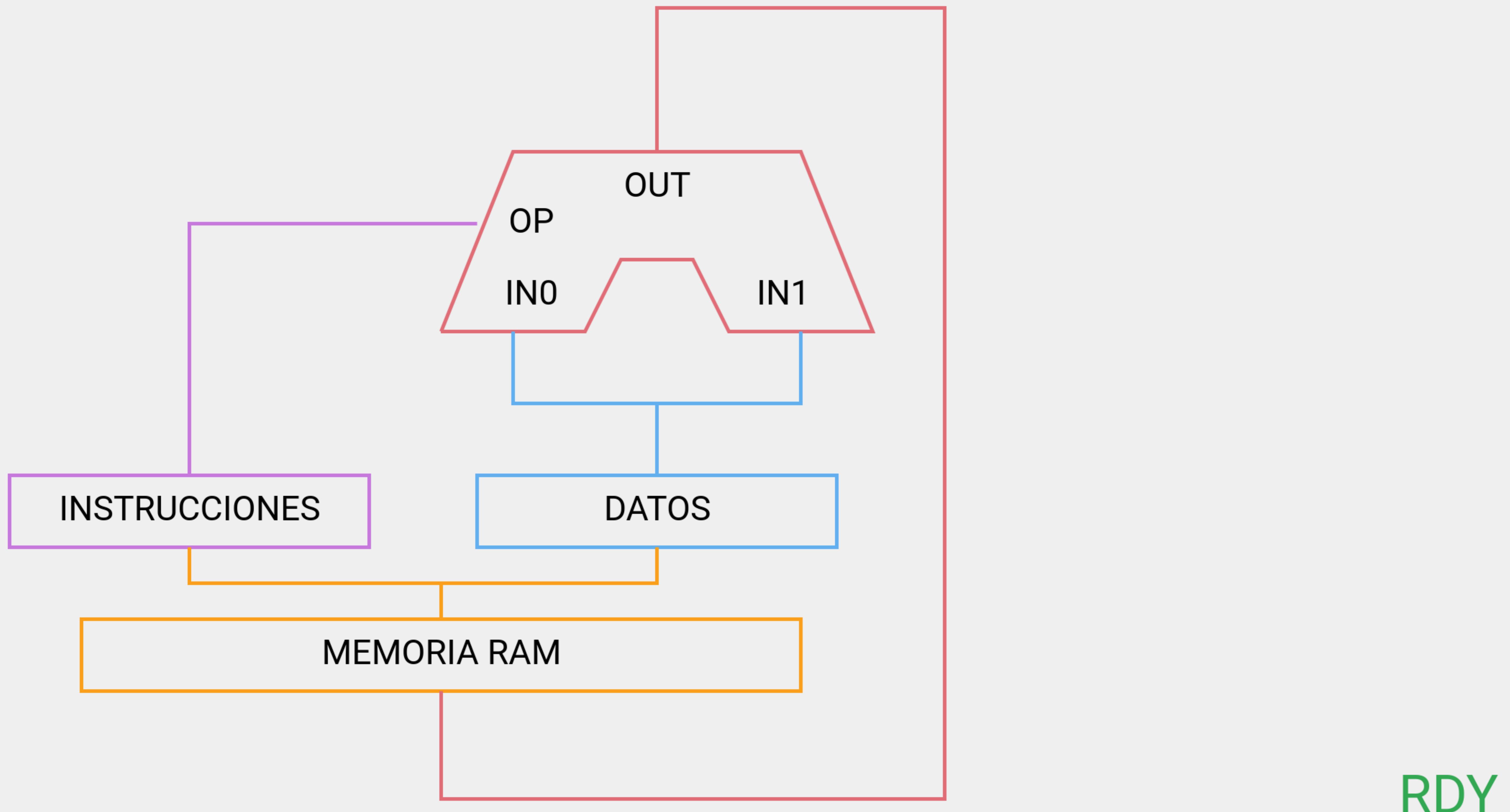
# Comparativa de Algoritmos: FCFS v. HDA\*



¿Cómo es la arquitectura de una CPU?

RDY

# Arquitectura: CPU

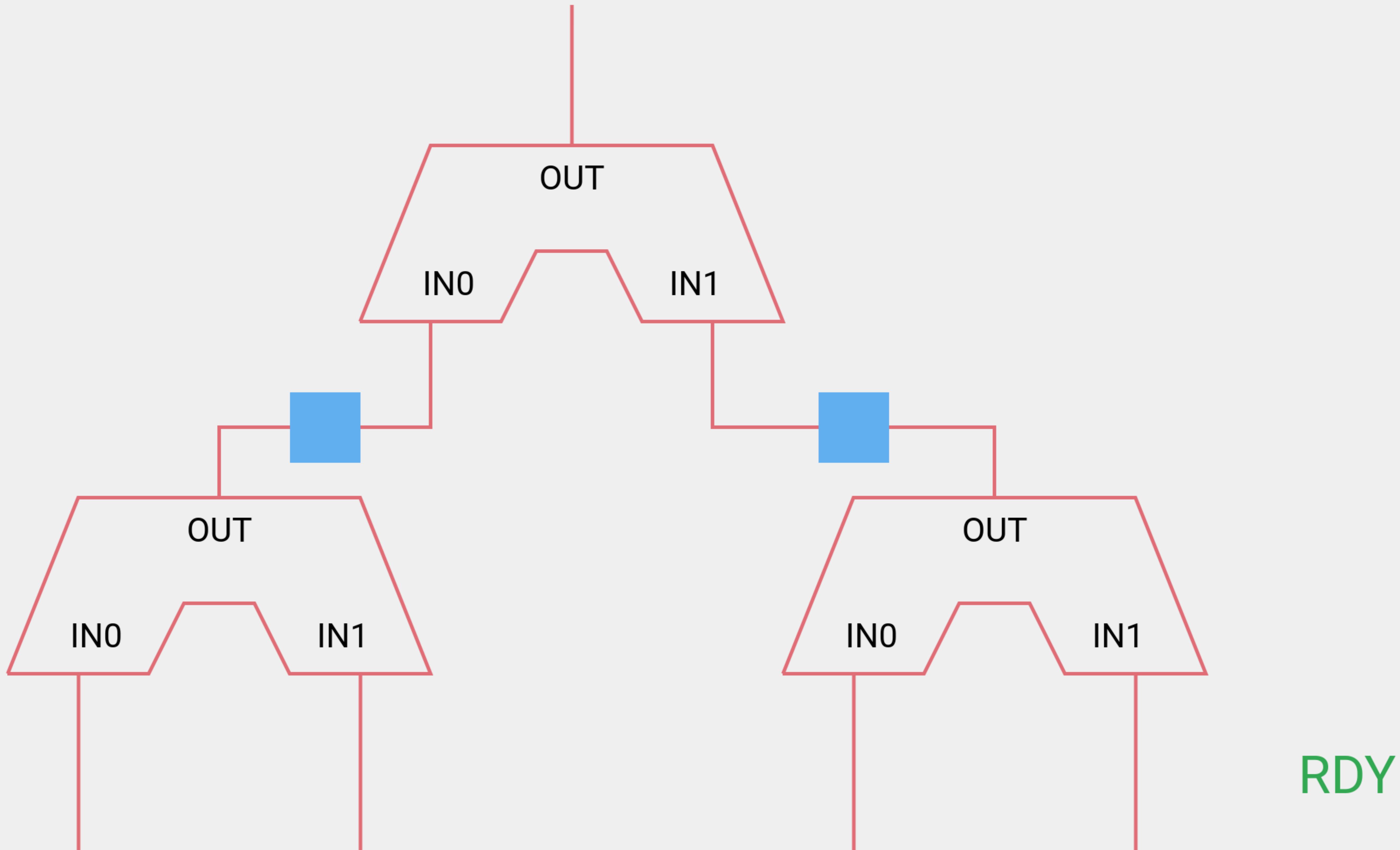


# Arquitectura: FPGA

¿Cómo es la arquitectura de una FPGA?

RDY

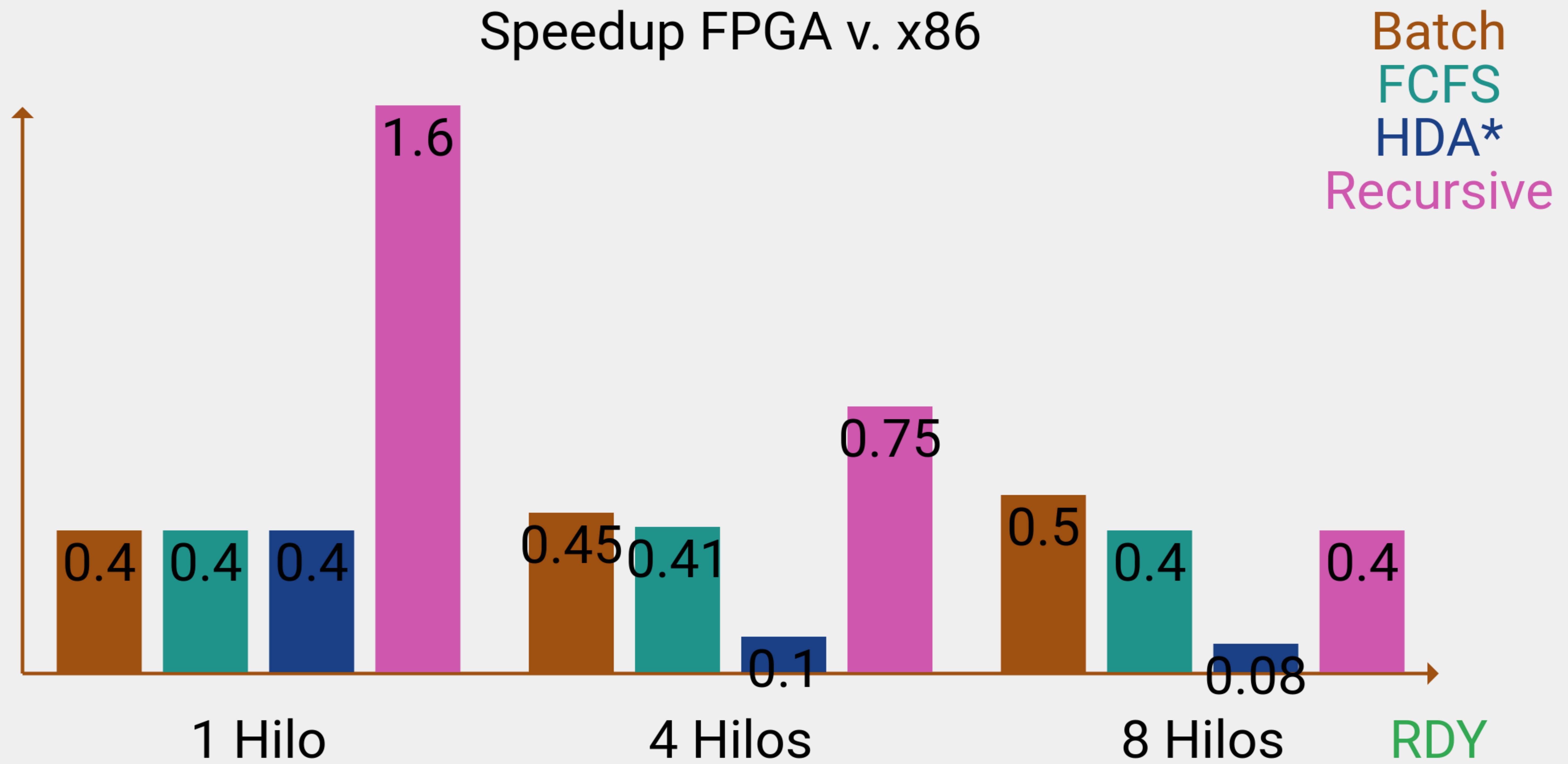
# Arquitectura: FPGA



# Comparativa de Arquitecturas

¿Cómo se comparan las arquitecturas?

# Comparativa de Arquitecturas: FPGA v. x86



# Conclusiones

1. El algoritmo A\* es adaptable a una variedad de problemas.
2. La selección del heurístico es el principal factor que determinará el rendimiento.
3. El paralelismo puede beneficiar al algoritmo A\* si está diseñado correctamente.
4. El manejo del open set es el principal problema del A\*.
5. El uso de HLS para transpilar una implementación existente de A\* no es práctica.



Universidad de Oviedo

# Optimización de Algoritmos de Búsqueda en Grafos: Implementación y Comparación de Rendimiento en FPGA



*Autor:* Rodríguez López, Alejandro

*Tutor:* Palacios Alonso, Juan José

2024-07-24