

# **Disjunctive Scheduling**

**Peter Stuckey & Guido Tack**



**MONASH**  
University

# Resources

- Critical to most scheduling problems are **limited resources**
- There are two types of resources
  - **Unary resources**: at most one task at a time
  - **Cumulative resources**: a limit on the amount of resource used at any time

# Unary Resources

- The Project Scheduling problem with non-overlap involved a unary resource
  - number of tasks executing at one time
- Unary resources are common
  - machine
  - nurse, doctor, worker in a roster
  - track segment (one train at a time)

# Scheduling Concepts (so far)

- Tasks
  - start time, duration (and end time)
  - other attributes

```
array[TASK] of var int: start;  
array[TASK] of (var) int: duration;
```

- Precedences
  - one task can only start after another finishes
  - task t1 precedes t2

```
start[t1] + duration[t1] <= start[t2]
```

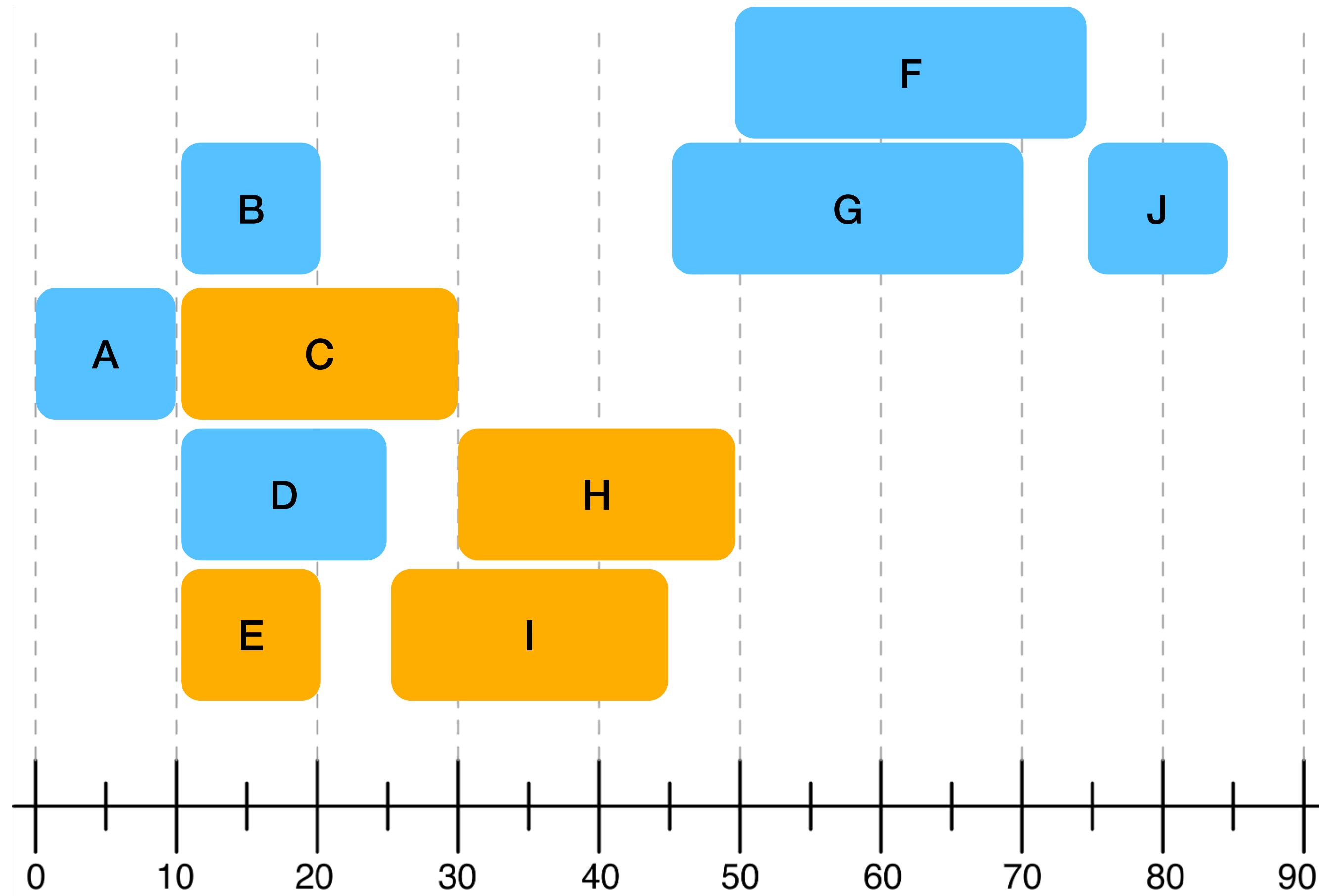
# Nonoverlap

- Tasks that use the same unary resource cannot overlap

```
predicate nonoverlap(var int:s1, var int:d1,  
                     var int:s2, var int:d2) =  
    s1 + d1 <= s2 \vee s2 + d2 <= s1;
```

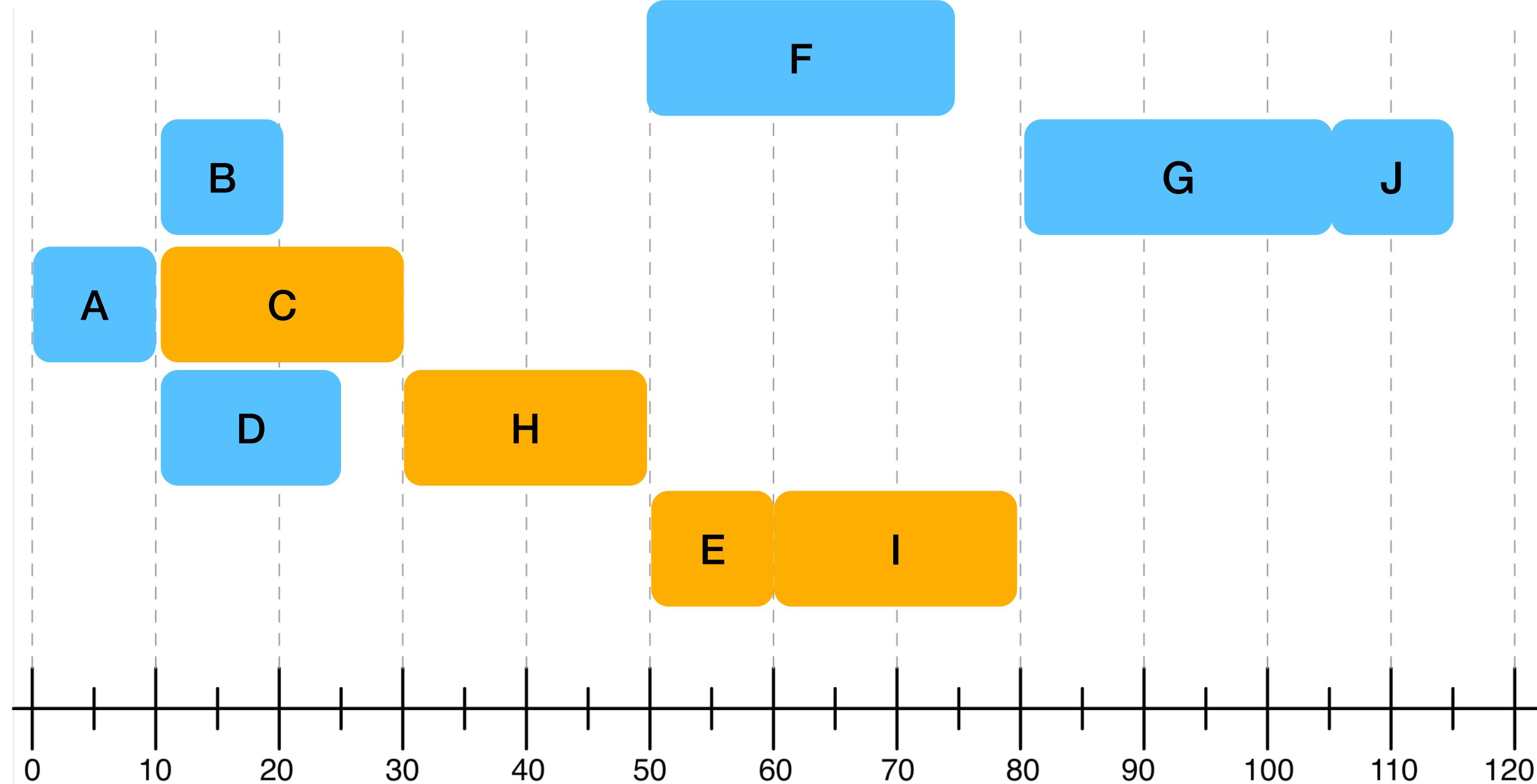
```
set of TASK: MUTEX = {C, E, H, I};  
constraint forall(t1, t2 in MUTEX where t1 < t2)  
    (nonoverlap(start[t1],duration[t1],  
               start[t2],duration[t2]));
```

# Basic Scheduling Solution



makespan = [A, B, C, D, E, F, G, H, I, J]  
85

# Scheduling with Unary Resource



makespan

**115**

= [0, 10, 10, 10, 50, 50, 80, 30, 60, 105]

# Disjunctive Global Constraint

- `nonoverlap` only considers two tasks at a time
  - unary resources require non overlap for all pairs of tasks that use it
  - The **disjunctive constraint** ensures that no two tasks in the array overlap in execution

```
predicate disjunctive(array[int] of var int:s,  
                      array[int] of var int:d)  
=  
  forall(i1,i2 in index_set(s) where i1 < i2)  
    (nonoverlap(s[i1],d[i1],s[i2],d[i2]));
```

# Replacing Nonoverlap with Disjunctive

- Tasks that use the same unary resource cannot overlap

```
include "disjunctive.mzn";  
  
constraint disjunctive(  
    [start[t] | t in MUTEX],  
    [duration[t] | t in MUTEX],  
);
```

# Summary

- Disjunctive scheduling
  - allows us to express that two tasks do not overlap in execution without specifying the relative order
- Disjunctive global constraint
  - capture a set of tasks on a unary resource

# EOF