# Lab 8: Wood Cutting

## Problem Description

In this workshop you will model a problem of scheduling the tasks involved in a process of cutting different kinds of logs into wood chips, employing state constraints.

**Principles of the problem:**

- A wood factory machine cuts stands (processed portions of log) into chips.
- Each stand has these characteristics:
    - length
    - diameter
    - species of wood
- The machine can cut a limited number of stands at a time with some restriction on the sum of the diameters that it can accept.
- The truck fleet can handle a limited number of stands at a given time.
- Stands processed simultaneously must all be of the same species.
- Each stand has a fixed delivery date and a processing status of one of::
    - standard
    - rush

    Any delay on a rush stand will cost a penalty

The wood cutting company needs to minimize costs per unit time, and reduce penalty costs resulting from late deliveries of rush stands to a minimum.

## Problem data

Here is the list of stands to cut giving the characteristics of each stand, the date delivery is due, and whether the order is a rush order or not.

| Diameter | Species | Length | Due Date | Rush? |
|---|---|---|---|---|
| 10 | oak | 20 | 991 | Yes |
| 10 | oak | 10 | 998 | No |
| 20 | beech | 20 | 210 | No |
| 10 | pine | 30 | 210 | No |
| 30 | beech | 30 | 1013 | No |
| 30 | oak | 10 | 1005 | No |
| 10 | oak | 40 | 1120 | No |
| 10 | beech | 41 | 300 | Yes |
| 10 | beech | 42 | 600 | Yes |
| 20 | oak | 21 | 1010 | No |
| 20 | pine | 10 | 918 | No |

**The restrictions on the process are as follows:**

- Maximum diameter the machine can process: 60
- Number of trucks in fleet: 10
- Maximum number of stands that can be processed together: 10
- Maximum number of batch periods per day: 10
- Cutting cost per day: 100
- Penalty cost for rush orders delivered late: 100/linear foot

- Species to cut:
    - oak
    - beech
    - pine
- Cutting time for each species:
    - oak: 3
    - beech: 5
    - pine: 7

## Exercise folder

**`<trainingDir>\OPL63.labs\Scheduling\Wood\work\sched_woodWork`**

Almost all of this model is already done. You are going to examine it, then write a state function and a state constraint to complete it.

## Step 1: Examine the completed parts of the model

### Actions

- Import the project
- Modeling the processing of the stands
- Modeling the quantity constraint
- Modeling the diameter constraint
- Modeling the fleet constraint

### Import the project

1. Import the **`<trainingDir>\OPL63.labs\Scheduling\Wood\work\sched_woodWork`** project into the OPL Projects Navigator. Leave the **Copy projects into workspace** box unchecked.
2. Open the **`sched_wood.mod`** file for editing and examine it.

### Modeling the processing of the stands

An interval variable is associated with each of the stands. The size of an interval variable is the product of the length of the stand and the time it takes to cut one unit of the stand's species:

```
dvar interval a[s in stands] size (s.len * cutTime[s.species]);
```

### Modeling the quantity constraint

The number of stands being processed at a time can be modeled by a cumulative expression function. Between the start and end of the interval representing the processing of the stand, the cumul function is increased by 1 using the **`pulse`** function. A constraint that the cumul function never exceeds the stand capacity of the machine is added to the model:

```
cumulFunction standsBeingProcessed = sum (s in stands) pulse(a[s], 1);

  standsBeingProcessed   <= maxStandsTogether;
```

### Modeling the diameter constraint

The total diameter of the stands being processed at a time can be modeled by a cumulative function. Between the start and end of the interval representing the processing of the stand, the cumul function is increased by the diameter using the **`pulse`** function. A constraint that the cumul function never exceeds the diameter capacity of the machine is added to the model:

```
cumulFunction diameterBeingProcessed = sum (s in stands) pulse(a[s],
s.diameter);

  diameterBeingProcessed <= maxDiameter;
```

### Modeling the fleet constraint

The constraint on the number of trucks being used can be placed on the cumul function for the number of stands being processed:

```
cumulFunction trucksBeingUsed = standsBeingProcessed;

  trucksBeingUsed        <= nbTrucks;
```

## Step 2: Define the one species constraint

### Actions

- Declare the state function
- Write an **`alwaysEqual`** constraint

### Reference

***alwaysEqual***

### Declare the state function

In this model, the wood cutting company can profit from processing multiple stands at the same time in the same batch, provided that certain constraints are met. One of these is that the cutting machine can only process one species of wood at a time. To express this in the model, you are first going to declare a state function called **`species`**.

- Do this now in the model file.

### Write an alwaysEqual constraint

1. Write a constraint that says that the value **`species`** in each member of the tuple set **`stands`** is equal when being processed by the cutting machine.

   Use the **`ord`** keyword to order the species together, and the scheduling constraint **`alwaysEqual`** to constrain the state function **`species`**.

2. Check your work against the file
   **`<trainingDir>\OPL63.labs\Scheduling\Wood\solution\sched_woodSolution\sched_wood.mod`**

## Step 3: Examine the objective function

### Action

- Transform the business objective into the objective function

### Transform the business objective into the objective function

The objective requires the model to minimize the sum of two calculations.

The first is the product of the maximum cutting time per stand and the cost per time unit.

- In the model, the maximum cutting time per stand is defined by a decision expression using the **dexpr** OPL keyword:

```
dexpr int makespan =
  max (s in stands) endOf(a[s]);
```

- The first part of the objective function calculates the product of **makespan** and the cost per time unit:

```
minimize makespan * (costPerDay / nbPeriodsPerDay)
```

The second quantity to be minimized is the product of the length of rushed stands that are late and the cost per unit of length for being late.

- To calculate the length (in feet, in this case) of stands identified as “rush” orders that are to be delivered late, we use another decision expression, named **lateFeet**:

```
dexpr float lateFeet =
  sum (s in stands : s.rush == 1) s.len * (endOf(a[s])  >s.dueDate);
```

- We can now complete the objective function by adding the calculation of cost of late rushed footage. The entire objective function is:

```
minimize makespan * (costPerDay / nbPeriodsPerDay)
  + costPerLateFoot * lateFeet;
```

Spend some time examining the solution in
**<trainingDir>\OPL63.labs\Scheduling\Wood\solution\sched_woodSolution\sched_wood.mod**

Pay special attention to how the state constraint interacts with the objective.