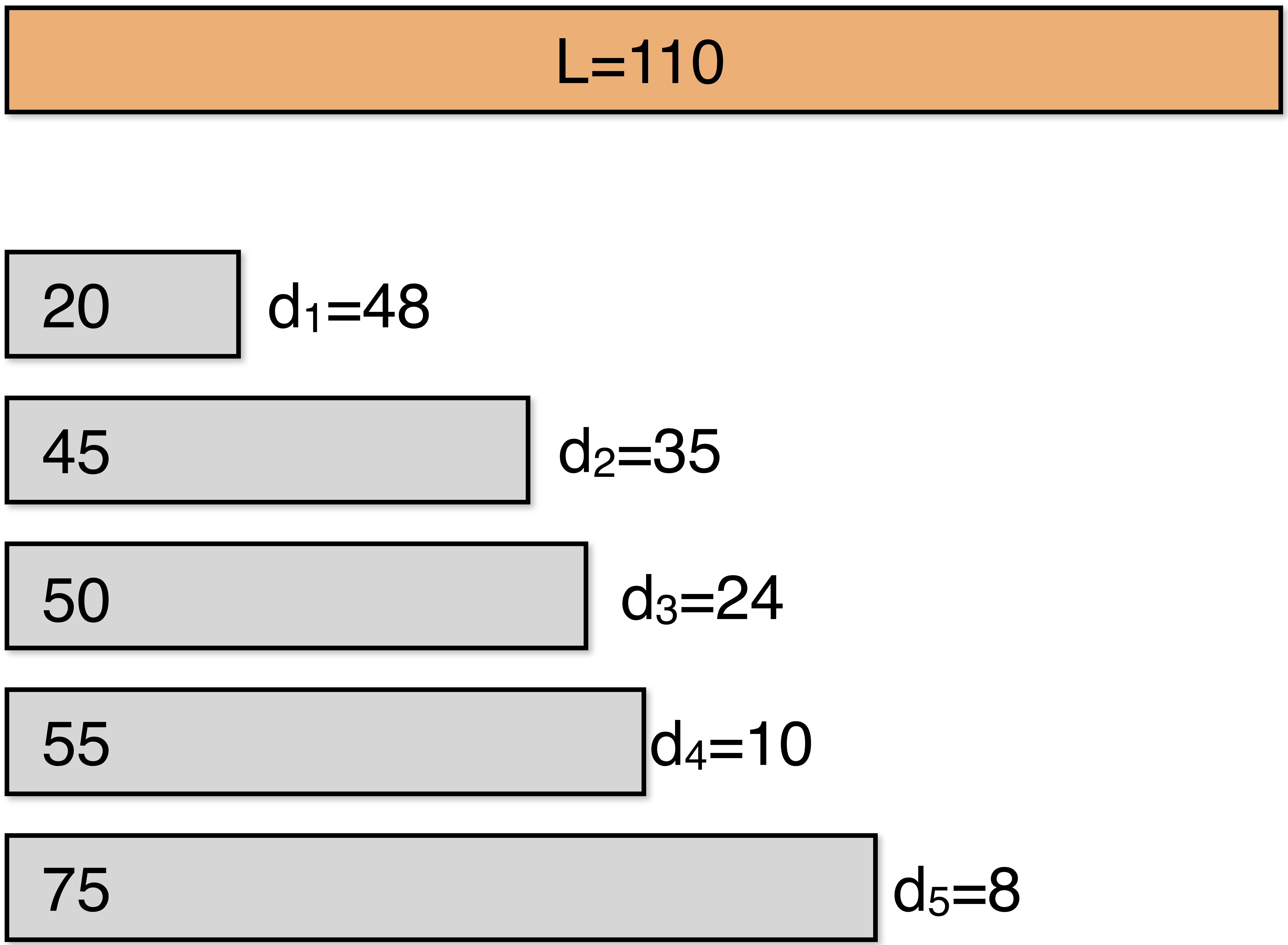
Discrete Optimization

Column Generation

Cutting Stock (Gilmore and Gomory)

- Given
  - a number of large wood boards of a length L
  - a number of shelves of various sizes that need to be cut from the boards
  - the demand for each shelf size
    - how many to cut.
- Find
  - the smallest number of boards to cut in order to meet the demand for shelves

Cutting Stock (Gilmore and Gomory)



A Second MIP Model

- Key idea
  - reasoning about cutting configurations, i.e., a specific way to cut a board
- How is a configuration c specified?
  - by the number of shelves of different types that it consists of.
  - e.g., $[n_{c,1}, \ldots, n_{c,|S|}]$
  - we can find all these configurations.
- Decision variables
  - x_c: the number of configurations of type c

A Second MIP Model

$$\min \sum_{c \in C} x_c$$

minimize the number of boards

s.t.

$$\sum_{c \in C} n_{c,s}\ x_c \geq d_s \qquad (s \in S)$$

demand constraint

$$x_c \in \mathbb{N} \qquad (c \in C)$$

- Strong relaxation
- No capacity constraint
  - it is built in the configurations
- No symmetries
  - reasoning about the numbers of configurations

Configurations

- Key idea
  - reasoning about cutting configuration, i.e., a specific way to cut a board
- How do we find these configurations?
  - a configuration must satisfy the constraint

$$\sum_{s \in S} l_s \; n_s \leq L$$

- What about enumerate them all?
  - in practical applications, there may be billions and billions
- Can we generate them on demand?

The MIP Program

	X_1	X_1	...	X_i	X_c	Demand
Obj	1	1	...	1	1	
$Self_1$	$n_{1,1}$	$n_{2,1}$	...	$n_{i,1}$	$n_{c,1}$	d_1
$Self_2$	$n_{1,2}$	$n_{2,2}$	...	$n_{i,2}$	$n_{c,2}$	d_2
...	...	...	...	...	...	...
$Self_{\|s\|}$	$n_{1,\|s\|}$	$n_{2,\|s\|}$	...	$n_{i,\|s\|}$	$n_{c,\|s\|}$	$d_{\|s\|}$

Configurations and Linear Programming

- Which configuration to generate?
  - a configuration is a column in the LP relaxation
- What is an interesting configuration then?
  - a configuration with a negative reduced cost
  - if it is positive, it will not enter the basis
- How did we compute these reduced costs?

$$cx \quad = \quad c_B A_B^{-1} b + (c - c_B A_B^{-1} A) x$$

Configurations and Linear Programming

- How did we compute these reduced costs?

$$cx = c_B A_B^{-1} b + (c - c_B A_B^{-1} A) x$$

- For a specific configuration i in this problem

$$1 - c_B A_B^{-1} (n_{i,1}, \ldots, n_{i,k})^T$$

$$1 - \Pi (n_{i,1}, \ldots, n_{i,k})^T$$

dual variables

The MIP Program

	X_1	X_1	...	X_i	X_c	Demand	Dual
Obj	1	1	...	1	1		
$Self_1$	$n_{1,1}$	$n_{2,1}$	...	$n_{i,1}$	$n_{c,1}$	d_1	π_1
$Self_2$	$n_{1,2}$	$n_{2,2}$	...	$n_{i,2}$	$n_{c,2}$	d_2	π_2
...	...	...	...	...	...	...	...
$Self_{\|S\|}$	$n_{1,\|s\|}$	$n_{2,\|s\|}$	...	$n_{i,\|s\|}$	$n_{c,\|s\|}$	$d_{\|s\|}$	$\pi_{\|s\|}$

Configurations and Linear Programming

- A new configuration must satisfy two conditions
  - feasibility
  - quality: i.e., entering the basis

- Feasibility

$$\sum_{s \in S} l_s \; n_s \leq L$$

- Quality

$$1 - \Pi(n_{i,1}, \ldots, n_{i,|S|})^T < 0$$

Configurations and Linear Programming

- A new configuration must satisfy two conditions
  - feasibility
  - quality: i.e., entering the basis
- Solve the following linear program

$$
\begin{aligned}
\min \quad & 1 - \sum_{s \in S} \Pi_s^* \; n_s \\
s.t \\
& \sum_{s \in S} l_s \; n_s \leq L
\end{aligned}
$$

- If the objective is negative, we have a new configuration for the LP relaxation; otherwise, no such configuration exists

Column Generation for Cutting Stock

1. Generate an initial set of configurations
   - one configuration for each shelf type
2. Solve the linear program
   - with the existing configurations
3. Generate a new configuration based on the optimal solution to the relaxation
   - solve the knapsack problem
4. If a new column was found, repeat from step 2
5. Otherwise, round the solution upwards to find an integer solution.