Cutting Stock with Bounded Open Orders

Jens Leoff

01.10.2014

Fraunhofer ITWM,
Department Optimization,
Kaiserslautern, Germany

Structure of the talk

- The real world problem
- Modelling
- Solution approach

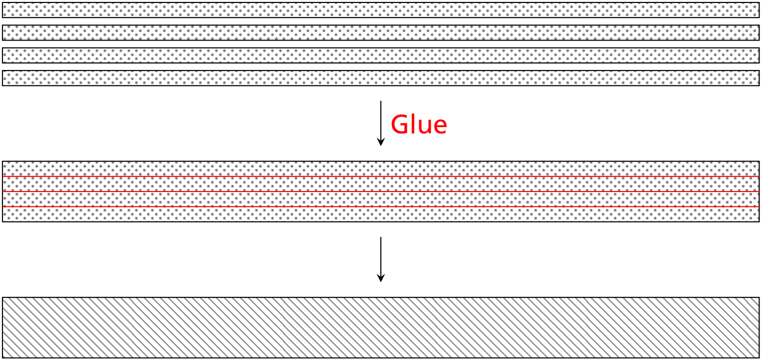
Structure of the talk

- The real world problem
- Modelling
- Solution approach

Structure of the talk

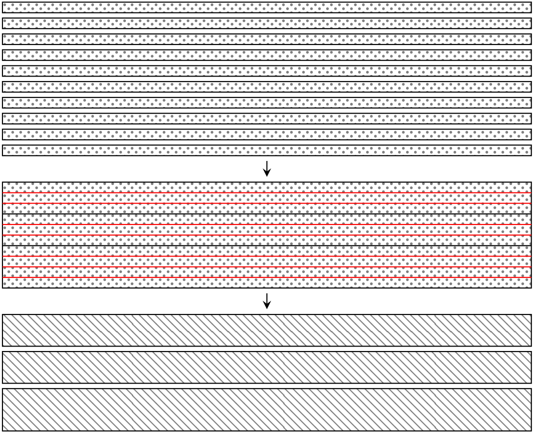
- The real world problem
- Modelling
- Solution approach

Glueing of slats in the woodcutting industries



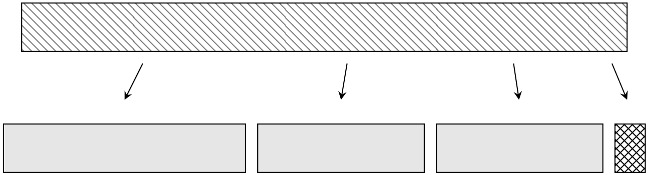
Wood slats are glued and pressed to beams.

Glueing of slats in the woodcutting industries



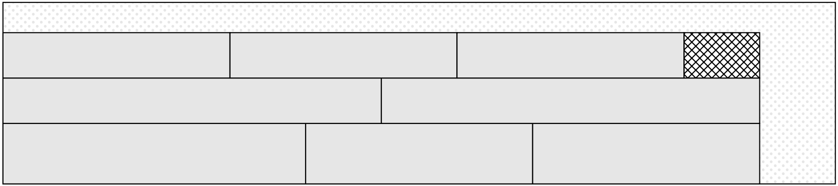
To produce multiple beams at once: leave away the glue

Glueing of slats in the woodcutting industries



A long beam is cut to obtain the final products.

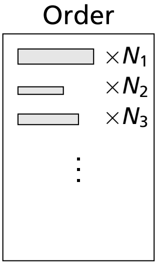
Glueing of slats in the woodcutting industries



Thus we want to cover all demands with 2-stage guillotine rectangular knapsack solutions.

Open Orders

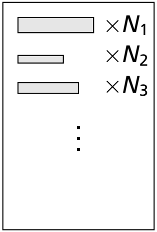
- Orders are formed by multiple product types with demand quantities



- Open Order:
  starting as the first part is produced,
  until completion of Order
- Constraint:
  there may be at most *k* Orders Open at any time

Open Orders

Order



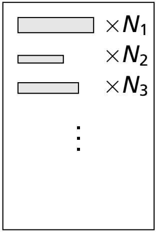
- Orders are formed by multiple product types with demand quantities

- Open Order:
  starting as the first part is produced,
  until completion of Order

- Constraint:
  there may be at most *k* Orders Open at any time

Open Orders

Order



- Orders are formed by multiple product types with demand quantities

- Open Order:
  starting as the first part is produced,
  until completion of Order
- Constraint:
  there may be at most *k* Orders Open at any time

Modelling

- Covering demands with patterns:
  this is the classical Cutting Stock Problem
  - The partition of input introduces symmetry
  - Using Column Generation, we can use a 1D Knapsack solver first, and switch to a 2D rectangular one later
- Pattern Sequencing:
  - leads to matrices with the Consecutive Ones Property (C1P)
  - prevents full aggregation of the knapsack sub-problems

Modelling

- Covering demands with patterns:
  this is the classical Cutting Stock Problem
  - The partition of input introduces symmetry
  - Using Column Generation, we can use a 1D Knapsack solver first, and switch to a 2D rectangular one later
- Pattern Sequencing:
  - leads to matrices with the Consecutive Ones Property (C1P)
  - prevents full aggregation of the knapsack sub-problems

Special case from the literature

Cutting Stock and the Minimization of Open Stacks Problem (MOSP)

- The sequential approach is well covered in the literature: Orders and product types are in 1:1 correspondence, first CS is solved and its solution is sequenced to minimize the number of Open Orders.
- Inversely, fixing the number of Open Orders to 1 can be solved optimally with a usual Cutting Stock solver
- We consider the integrated Cutting Stock and Sequencing problem

Fraunhofer ITWM

Special case from the literature

Cutting Stock and the Minimization of Open Stacks Problem (MOSP)

- The sequential approach is well covered in the literature: Orders and product types are in 1:1 correspondence, first CS is solved and its solution is sequenced to minimize the number of Open Orders.
- Inversely, fixing the number of Open Orders to 1 can be solved optimally with a usual Cutting Stock solver
- We consider the integrated Cutting Stock and Sequencing problem

Special case from the literature

Cutting Stock and the Minimization of Open Stacks Problem (MOSP)

- The sequential approach is well covered in the literature: Orders and product types are in 1:1 correspondence, first CS is solved and its solution is sequenced to minimize the number of Open Orders.
- Inversely, fixing the number of Open Orders to 1 can be solved optimally with a usual Cutting Stock solver
- We consider the integrated Cutting Stock and Sequencing problem

Special case from the literature

Cutting Stock and the Minimization of Open Stacks Problem (MOSP)

- The sequential approach is well covered in the literature: Orders and product types are in 1:1 correspondence, first CS is solved and its solution is sequenced to minimize the number of Open Orders.
- Inversely, fixing the number of Open Orders to 1 can be solved optimally with a usual Cutting Stock solver
- We consider the integrated Cutting Stock and Sequencing problem

The sequential approach

How is the trade off between the CS and MOSP objectives?

Scheithauer 03

Sequentially solving CS and then MOSP with 150 item types results in LBs $\geq$ 100 and solutions with 130 open stacks.

We can observe similar result with Open Orders

Numeric Teaser

How is the trade off between the CS objective and the number of Open Stacks?

The Cutting Stock Problem

Input: $\{(l_i, d_i)\}, C$

Original Formulation:

$$\begin{aligned} & \min \sum_t y_t \\ s.t. \quad & \sum_t x_{jt} \geq d_j && \forall j \\ & \sum_j l_j x_{jt} \leq C\, y_t && \forall t \\ & x_{jt} \in \mathbb{Z}_{\geq 0} && \forall j, t \\ & y_t \in \{0, 1\} && \forall t \end{aligned}$$

Dantzig-Wolfe Reformulation:

$$\begin{aligned} & \min \sum_t \sum_{g \in G^t} a_y^{gt} \lambda^{gt} \\ s.t. \quad & \sum_t \sum_{g \in G^t} a_j^{gt} \lambda^{gt} \geq d_j && \forall j \\ & \sum_{g \in G^t} \lambda^{gt} = 1 && \forall t \\ & \lambda^{gt} \in \{0, 1\} && \forall t, g \end{aligned}$$

The Cutting Stock Problem

Input: $\{(l_i, d_i)\}, C$

Dantzig-Wolfe Reformulation:

$$
\begin{aligned}
&\min \sum_{t} \sum_{g \in G^t} \cancel{a_y^{gt}} \lambda^{gt} && \\
s.t.\quad & \sum_{t} \sum_{g \in G^t} a_j^{gt} \lambda^{gt} \geq d_j && \forall j \\
& \sum_{g \in G^t} \lambda^{gt} \cancel{=} \leq 1 && \forall t \\
& \lambda^{gt} \in \{0, 1\} && \forall t, g
\end{aligned}
$$

Aggregation:

$$
\begin{aligned}
&\min \sum_{g} \lambda^{g} && \\
s.t.\quad & \sum_{g} a_j^{g} \lambda^{g} \geq d_j && \forall j \\
& \cancel{\sum_{g \in G} \lambda^{g} \leq T} && \\
& \lambda^{g} \in \mathbb{Z}_{\geq 0} && \forall g
\end{aligned}
$$

Aggregation removes the symmetry in t!

The Cutting Stock Problem

Input: $\{(l_i, d_i)\}, C$

Dantzig-Wolfe Reformulation:

$$
\begin{array}{llr}
 & \min \sum_{t} \sum_{g \in G^t} \cancel{a_y^{gt}} \lambda^{gt} & \\
s.t. & \sum_{t} \sum_{g \in G^t} a_j^{gt} \lambda^{gt} \geq d_j & \forall j \\
 & \sum_{g \in G^t} \lambda^{gt} \cancel{=} \leq 1 & \forall t \\
 & \lambda^{gt} \in \{0, 1\} & \forall t, g
\end{array}
$$

Aggregation:

$$
\begin{array}{llr}
 & \min \sum_{g} \lambda^{g} & \\
s.t. & \sum_{g} a_j^{g} \lambda^{g} \geq d_j & \forall j \\
 & \cancel{\sum_{g \in G} \lambda^{g} \leq T} & \\
 & \lambda^{g} \in \mathbb{Z}_{\geq 0} & \forall g
\end{array}
$$

Aggregation removes the symmetry in t!

Cutting Stock with Bounded Open Orders

$$
\begin{array}{lll}
 & \min \sum_t y_t & \\
s.t. & \sum_j l_j x_{jt} \leq C y_t & \forall t \\
 & \sum_t x_{jt} \geq d_j & \forall j \\
 & s_{ot} \leq s_{o\ t+1} & \forall o, t = 1, \ldots, T-1 \\
 & c_{ot} \leq c_{o\ t+1} & \forall o, t = 1, \ldots, T-1 \\
 & \sum_o (s_{ot} - c_{ot}) \leq K & \forall t \\
 & x_{jt} \leq M_j (s_{ot} - c_{ot}) & \forall t, o, j \in o \\
 & x_{jt} \in \mathbb{N} & \\
 & y_t, s_{ot}, c_{ot} \in \{0, 1\} &
\end{array}
$$

Cutting Stock with Bounded Open Orders

- $s_{ot} - c_{ot} \in \{0, 1\}$ describes if an order is open at time t
- in a feasible solution theses values have the strict C1P:

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

- identical and dominated columns can be aggregated:

$$\rightarrow \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Cutting Stock with Bounded Open Orders

- $s_{ot} - c_{ot} \in \{0, 1\}$ describes if an order is open at time t
- in a feasible solution theses values have the strict C1P:

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

- identical and dominated columns can be aggregated:

$$\rightarrow \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Cutting Stock with Bounded Open Orders

- $s_{ot} - c_{ot} \in \{0, 1\}$ describes if an order is open at time t
- in a feasible solution theses values have the strict C1P:

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

- identical and dominated columns can be aggregated:

$$\rightarrow \quad \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad \rightarrow \quad \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Reformulated and Aggregated CSBOO

$$\min \sum_{t} \sum_{g \in G^t} \lambda^{gt}$$

$$s.t. \sum_{t} \sum_{g \in G^t} a_j^{gt} \lambda^{gt} \geq d_j \qquad \forall j$$

$$s_{ot} \leq s_{o\ t+1} \qquad \forall o, t = 1, \ldots, \tilde{T} - 1$$

$$c_{ot} \leq c_{o\ t+1} \qquad \forall o, t = 1, \ldots, \tilde{T} - 1$$

$$\sum_{o} (s_{ot} - c_{ot}) \leq K \qquad \forall t$$

$$\sum_{g \in G^t} a_j^{gt} \lambda^{gt} \leq d_j (s_{ot} - c_{ot}) \qquad \forall t, o, j \in o$$

$$\lambda^{gt} \in \mathbb{N}$$

$$s_{ot}, c_{ot} \in \{0, 1\}$$

Branching

- s_{ot}, c_{ot} variables first
- add constraint $s_{ot} - c_{ot} = 0/1$
- choose t such that a range of fractional $s_{ot} - c_{ot}$ is halved

Branching

- s_{ot}, c_{ot} variables first
- add constraint $s_{ot} - c_{ot} = 0/1$
- choose t such that a range of fractional $s_{ot} - c_{ot}$ is halved

Branching

- s_{ot}, c_{ot} variables first
- add constraint $s_{ot} - c_{ot} = 0/1$
- choose t such that a range of fractional $s_{ot} - c_{ot}$ is halved

Branching

- λ^{gt} integrality is enforced using Vanderbeck's branching scheme
- works well with Knapsack Problem
- does not force as much disaggregation as branching on original variables

Branching

- λ^{gt} integrality is enforced using Vanderbeck's branching scheme
- works well with Knapsack Problem
- does not force as much disaggregation as branching on original variables

Branching

- λ^{gt} integrality is enforced using Vanderbeck's branching scheme
- works well with Knapsack Problem
- does not force as much disaggregation as branching on original variables

Some Issues

- Instead of $\sum_o (s_{ot} - c_{ot}) \leq K \quad \forall t$:

$$\sum_{l=0}^{t} s_{ot} = K + t, \qquad \sum_{l=0}^{t} c_{ot} = t$$

- I get stuck at some point in the branching tree.
- How to choose what to branch on?

Some Issues

- Instead of $\sum_o (s_{ot} - c_{ot}) \leq K \quad \forall t$:

$$\sum_{l=0}^{t} s_{ot} = K + t, \qquad \sum_{l=0}^{t} c_{ot} = t$$

- I get stuck at some point in the branching tree.
- How to choose what to branch on?

Some Issues

- Instead of $\sum_o (s_{ot} - c_{ot}) \leq K \quad \forall t$:

$$\sum_{l=0}^{t} s_{ot} = K + t, \qquad \sum_{l=0}^{t} c_{ot} = t$$

- I get stuck at some point in the branching tree.
- How to choose what to branch on?