Seminar, NTNU, Trondheim, 3.1.2001

Methods for solving large-scale scheduling and combinatorial optimization problems

Iiro Harjunkoski
(in collaboration with Ignacio E. Grossmann)

Department of Chemical Engineering
Carnegie Mellon University
Pittsburgh, PA 15213

http://egon.cheme.cmu.edu/

Recent Work

- Vipul Jain and Ignacio E. Grossmann (2000). Algorithms for Hybrid ILP/CP Models for a Class of Optimization Problems. *Presented at INFORMS, Paper SD32.1, Salt Lake City, UT.*
- Iiro Harjunkoski, Vipul Jain and Ignacio E. Grossmann (2000). Hybrid Mixed-Integer/Constraint Logic Programming Strategies for Solving Scheduling and Combinatorial Optimization Problems. *Computers and Chemical Engineering, 24*, 337-343.
- Iiro Harjunkoski and Ignacio E. Grossmann (2000). A Decomposition Strategy for Optimizing Large-Scale Scheduling Problems in the Steel Making Industry. *Presented at AIChE Annual Meeting 2000, Los Angeles, CA.*

Outline

- Classification of scheduling problems
- Hybrid methods for handling the combinatorial complexity
- Another combinatorial problem
- Decomposition method for a large-scale problem
- Conclusions

Outline

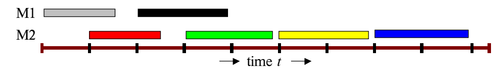
- **Classification of scheduling problems**
- Hybrid methods for handling the combinatorial complexity
- Another combinatorial problem
- Decomposition method for a large-scale problem
- Conclusions

Scheduling

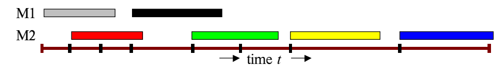
- Allocation of jobs into restricted resources or equipment
- Often critical release and duedates
- Both assignment and sequencing decisions are discrete: large number of binary variables
- Often non-convexities or poor relaxations
- Heuristics, AI-methods widely used for real problems
- Mathematical programming can bring significant improvements into decision making!

Discrete time representation

- Events can take place only at certain times
  - Uniform time discretization



  - Non-uniform time discretization

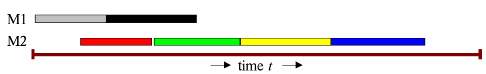


$y_{jmt} \in \{0,1\}$ = 1 if job j starts at equipment m at time t

- Flexible, easy to maintain linearity, sequencing done through constraints

Continuous time representation

- Events can take place at any time



  - One set of variables for assignment & sequencing

    $y_{jj'm} \in \{0,1\}$ = 1 if job j precedes job j' on equipment m

  - Separate variables for assignment & sequencing

    $y_{jm} \in \{0,1\}$ = 1 if job j is processed on equipment m

    $y_{jj'} \in \{0,1\}$ = 1 if job j precedes job j'

- Complete time domain considered, no problems in finding optimal time grid

How to solve?

"A mathematician (thoughtful) might decide to ask a different question: Can I find an algorithm that is guaranteed to find a solution "close to optimal" in polynomial time in all cases?"

"An engineer would start looking for a heuristic algorithm that produces practically usable solutions."

--Wolsey, Integer Programming (Wiley-Interscience, 1998)

- Extensive literature both in mathematical and engineering journals
- Large amount of heuristic strategies

Outline

- Classification of scheduling problems
- Hybrid methods for handling the combinatorial complexity
- Another combinatorial problem
- Decomposition method for a large-scale problem
- Conclusions

Overview

- Methods presented to overcome combinatorial complexity for large discrete optimization problems.
- Basic idea: combine mixed integer programming (MIP) and constraint programming (CP) and exploit complementary strengths.
- Illustration: parallel scheduling and trim-loss problems

Motivation

- Scheduling problems are often huge and solution is required quickly

  Jobshop: 10 prod., 1 mach. ⇒ 100 binary variables

- Heuristic methods: solutions may be far from optimal
- Branch-and-bound is not the only rigorous approach for handling integer variables

Logic based Methods

Alternative logic based methods are:

- Generalized disjunctive programming (GDP)
  (Raman and Grossmann, 1994)
- Constraint logic programming (CLP)
  (Hentenryck, 1989)

CLP & MILP

Properties

MILP

- Search is based on variables (B&B)
- Relaxation important
- All constraints evaluated simultaneously in a tree search
- Software: OSL, CPLEX, XPRESS-MP

CLP

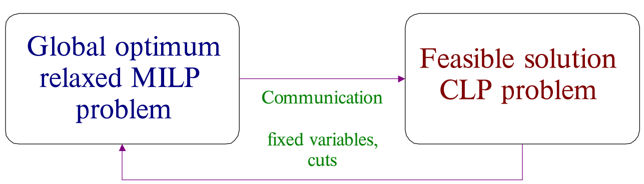
- Focus on constraints
- Requires well constrained problems
- Constraint propagation at each node, domain reduction through implicit enumeration
- Software: ILOG, ECLiPSe, CHIP

Comparison

- Both MILP and CLP: continuous and integer variables
- MILP: only linear constraints
- CLP: linear and nonlinear constraints plus logic constructs e.g. implications
- MILP: excludes large search areas ⇒ good for proving optimality and directing the search
- CLP: finds a feasible solution fast ⇒ more efficient for feasibility check through constraint propagation

Integration

- Optimize a relaxed MILP problem to its global optimal solution
- Solve a feasibility problem with CLP to check the feasibility
- Communication between steps important (cuts)



Requirements

- Objective function separable: only a subset of the coefficients non-zero
- Clearly defined relation between MILP and CLP models
- MILP relaxation needs to be efficient
- CLP part well-constrained
- Efficient cuts for eliminating more than a single solution

Scheduling Problem

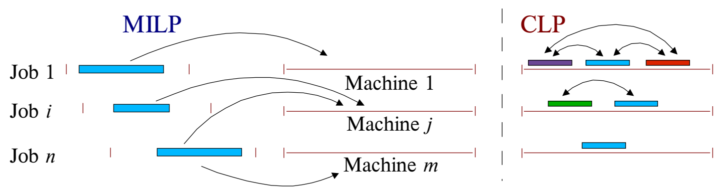
(Jain and Grossmann, 2000)

Jain and Grossmann (2000): two subproblems

1) **MILP**: assignment problem, tight LP-relaxation, contains the objective function variables

2) **CLP**: sequencing problem, no objective function variables, constraints with poor relaxations

These subproblems are joined by equivalence relations



Scheduling MILP Formulation

$$\min \sum_{i\in I}\sum_{m\in M} C_{i,m}x_{i,m}$$

s.t.

$$ts_i \ge r_i \quad \forall i\in I$$

$$ts_i \le d_i = \sum_{m\in M} p_{i,m}x_{i,m} \quad \forall i\in I$$

$$\sum_{m\in M} x_{i,m}=1 \quad \forall i\in I$$

$$\sum_{i\in I} x_{i,m}p_{i,m} \le \max_i\{d_i\}-\min_i\{r_i\} \quad \forall m\in M$$

$$y_{ii'}+y_{i'i} \ge x_{i,m}+x_{i',m}-1 \quad \forall i,i'\in I, i'>i, m\in M$$

$$ts_{i'} \ge ts_i+\sum_{m\in M} p_{i,m}x_{i,m}-U(1-y_{ii'}) \quad \forall i,i'\in I, i'\ne i$$

$$ts_i \ge 0$$

$$x_{i,m}\in\{0,1\} \quad \forall i\in I, m\in M$$

$$y_{ii'}\in\{0,1\} \quad \forall i,i'\in I, i\ne i'$$

...Scheduling Problem

$$\min \; c_x^T x \quad (1)$$

s.t.

$$Ax+Dv\le a \quad (2)$$

$$x \Leftrightarrow \bar{x} \quad (3)$$

$$\bar{G}(\bar{x},\bar{v},\bar{y})\le 0 \quad (4)$$

$$v\in R^v, x\in\{0,1\} \quad (5)$$

$$\bar{x},\bar{v},\bar{y}\in D \quad (6)$$

MILP: (1), (2), (5)

CLP: (3), (4), (6)

Scheduling Formulation

MILP

$$\min \sum_{i\in I}\sum_{m\in M} C_{i,m}x_{i,m}$$

s.t.

$$ts_i\ge r_i \quad \forall i\in I$$

$$ts_i \le d_i - \sum_{m\in M} p_{i,m}x_{i,m} \quad \forall i\in I$$

$$\sum_{m\in M} x_{i,m}=1 \quad \forall i\in I$$

$$\sum_{i\in I} x_{i,m}p_{i,m} \le \max_i\{d_i\}-\min_i\{r_i\} \quad \forall m\in M$$

CLP

$$i.start \ge r_i \quad \forall i\in I$$

$$i.start \le d_i - p_{z_i} \quad \forall i\in I$$

$$i.duration = p_{z_i} \quad \forall i\in I$$

$$i\ requires\ t_{z_i} \quad \forall i\in I$$

$$if\ (x_{i,m}=1)\ then\ (z_i=m) \quad \forall i\in I,\ \forall m\in M$$

Linking constraint

$$ts_i\ge 0$$

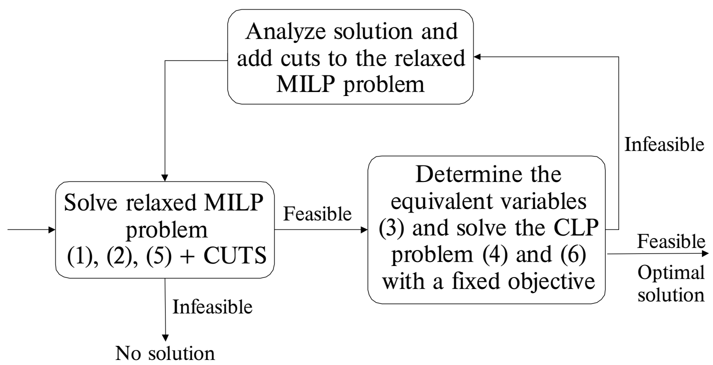
$$x_{i,m}\in\{0,1\} \quad \forall i\in I,\ \forall m\in M$$

$$z_i\in M \quad \forall i\in I$$

$$i.start\in Z \quad \forall i\in I$$

$$i.duration\in Z \quad \forall i\in I$$

Hybrid Procedure



Scheduling - Results

CPU-s from Jain and Grossmann (2000)

	2 machines 3 jobs		3 machines 7 jobs		3 machines 12 jobs		5 machines 15 jobs		5 machines 20 jobs	
Problem	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2
MILP	0.04	0.04	0.31	0.27	926.3	199.9	1784.7	73.3	18142.7	102672.3
CLP	0	0.02	0.04	0.14	3.84	0.38	553.5	9.28	68853.5	2673.9
Hybrid	0.02	0.01	0.52	0.02	4.18	0.02	2.25	0.04	14.13	0.41

In the following rerun, release dates, due dates and durations have arbitrary rational numbers (Harjunkoski et al., 2000)

	2M, 3J		3M, 7J		3M, 12J		5M, 15J		5M, 20J	
Problem	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2	Set 1	Set 2
Hybrid	0.00	0.01	0.51	0.02	5.36	0.03	0.64	0.92	36.63	4.79

Improvement

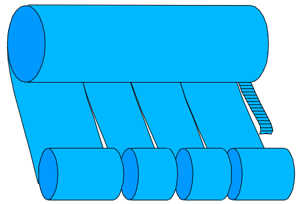
- The solution times dropped dramatically, especially for the largest problems
- The slowest hybrid example: 14.3 CPU-s
- CLP and MILP: combinatorial explosion
- Proposed strategy 1300 and 19000 times faster than the better one!!!

	5 machines 20 jobs	
Problem	Set 1	Set 2
MILP	18142.7	102672.3
CLP	68853.5	2673.9
Hybrid	14.13	0.41

Outline

- Classification of scheduling problems
- Hybrid methods for handling the combinatorial complexity
- Another combinatorial problem
- Decomposition method for a large-scale problem
- Conclusions

Trim-Loss Problem



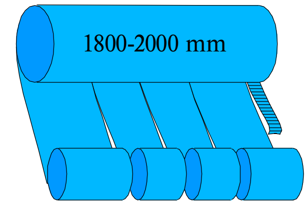
Width (mm)	Reels
330	12
360	6
370	15
415	6
435	9

Trim-Loss Problem

Manual test 1:
330+360+370+415+435 = 1910 x 6
(330 - 6, 370 - 9, 435 - 3)
330+330+370+370+435 = 1835 x 3
(370 - 3 = 1110 mm < 1800 mm)
370+370+370+370+370 = 1850 x 1

Manual test 2:
330+360+370+370+415 = 1845 x 6
(330 - 6, 370 - 3, 435 - 9)
330+330+370+435+435 = 1900 x 3
(435 - 3 = 1305 mm < 1800 mm)
435+435+435+330+330 = 1965 x 1

Optimization:
330+330+360+415+435 = 1835 x 6
370+370+370+370+435 = 1915 x 4



Width (mm)	Reels
330	12
360	6
370	15
415	6
435	9

Trim-Loss Problem: MINLP

$$\min \sum_{j \in J} (c_j \cdot m_j + C_j \cdot y_j)$$

s.t.

$$B_{max} - \Delta_{max} \le \sum_{i \in I} b_i \cdot n_{ij} \le B_{max}$$

$$\sum_{i \in I} n_{ij} \le N_{max}$$

$$y_j \le m_j \le M_j \cdot y_j$$

$$\forall j \in J$$

$$n_{i,order} \le \sum_{j \in J} m_j \cdot n_{ij} \le n_{i,max}$$

$$\forall i \in I$$

$$m_j, n_{ij}, \in Z^+ \quad y_j \in \{0,1\}$$

- Objective to minimize the waste / raw-paper usage
- Spill has to be withing specified limits
- Limited number of cuts per pattern
- Bilinear demand constraint with integer variables: transformations needed

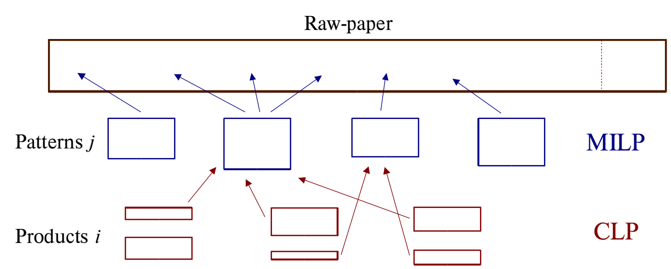
Trim-Loss Problem

(Harjunkoski et al., 1998)

Bilinearity: two-level problem

1) MILP: solve relaxed problem for the number of patterns needed

2) CLP: solve the exact pattern outlook - fixed objective



...Trim-Loss Problem

min $c^T x$ (7)

s.t. $\sum_k b_k y_k \le d$ (8)

$g^j(x,y) \le a$ (9)

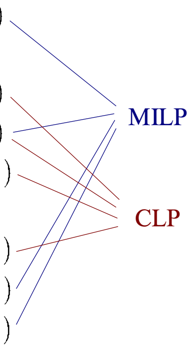
$x \in Z^x, y \in Z^y$ (10)

where

$g^j(x,y) = \sum_i a_i x_i + \sum_i \sum_k c_{ik} x_i y_k$ (11)

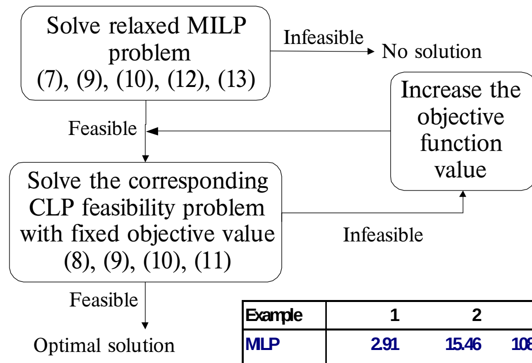
$g^j(x,y)^{lin} = \sum_i a_i x_i + \sum_u \sum_k c_{ik} u_{ik}$ (12)

$u \in Z^u$ (13)



Equation (9) is replaced by bilinear (11) or linear (12) when solving the MILP resp. CLP problems

Hybrid Procedure



Example	1	2	3	4
MLP	2.91	15.46	10842.63	9884.15
CLP	265.31	7465.31	8508.77	>50000
Hybrid	0.09	0.64	65.45	52.34

Improvement

- Even a loose integration fruitful
- CLP could not solve problem 4
- MILP around 10,000 CPU-s
- Improvement factor: 130-190
- Potential of further improvements by generating cuts for MILP relaxation

Example	3	4
MLP	10842.63	9884.15
CLP	8508.77	>50000
Hybrid	65.45	52.34

Trim-Loss examples

Example 1

Width	Order	Max
330	12	14
360	6	7
370	15	17
415	6	7
435	9	10

Width: 1800-2000

Example 2

Width	Order	Max
330	8	10
360	16	18
380	12	14
430	7	9
490	14	16
530	16	18

Width: 2100-2200

Example 3

Width	Order	Max
290	15	18
315	28	32
350	21	24
455	30	34
615	24	27

Width: 1900-2100

Example 4

Width	Order	Max
280	14	16
325	26	30
360	25	29
395	27	31
405	35	39
455	30	34
515	32	37

Width: 2000-2200

Other approaches

- MILP approaches suffer from combinatorial explosion
- Only special class of problems suitable for CP/MILP integration
- Manual planning methods slow and inflexible
- Pure heuristics fast but solution quality may be bad
- Clear potential benefits in using/integrating mathematical programming into solution of large number of industrial problems
- Example: A decomposition strategy for steel industry

Outline

- Classification of scheduling problems
- Hybrid methods for handling the combinatorial complexity
- Another combinatorial problem
- Decomposition method for a large-scale problem
- Conclusions

Decomposition Method

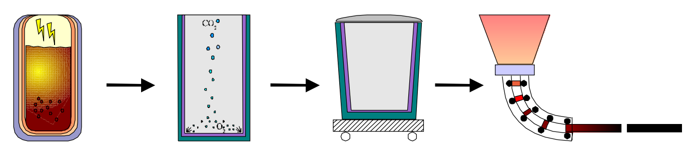
Scheduling of a steel plant is among the most difficult industrial problems. Complexities arise from:

- Temperature requirements
- Chemistry constraints
- Material properties
- Equipment availability

Most strategies include heuristics or expert systems. Simulations are often used to verify decisions. Relatively few mathematical programming approaches.

Steel Making Process

1) Hot iron and scrap mixed in EAF
2) Decarburization in AOD
3) Quality adjustments in LMF
4) Casting into steel slabs



1 2 3 4

http://www.steel.org (American Iron and Steel Institute)

Grades

The products of the steel making process are defined by their grades

- product quality description (chemical and physical)
- each grade has a given recipe that specifies temperature and chemistry at each stage
- grades are subdivided into subgrades
- subgrades have minor differences to actual grade (e.g. lower carbon content)
- Example: grade 301 can have a low-carbon subgrade 301L

Assumptions

- Grades and at their subgrades can be casted in the same sequence (in specific order)
- For simplicity, we assume here that grades are subdivided into subgrades a,b,c
- Besides their grades, orders (heats) are characterized by slab width and thickness
- Only one order can be assigned to one equipment at a time
- Most chemistry rules embedded in parameters i.e. not explicitly considered in the model

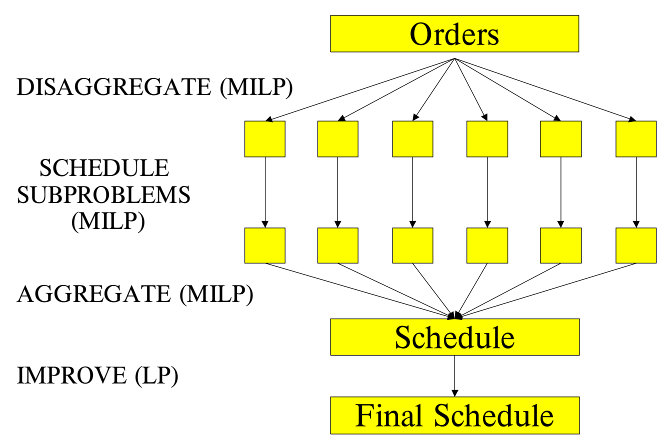
Problem Statement

- Given the grade constraints, production equipment and a number of customer orders (heats), find the schedule that minimizes the makespan following the recipe for each product grade.
- Producing a valid schedule for 10 heats is hard with MILP because of the problem size and complexity.

  470 0-1 vars, 83 cont. vars., 1187 constr., > 10000 CPU-s

- Need to schedule one week production (80-90 heats).

Strategy



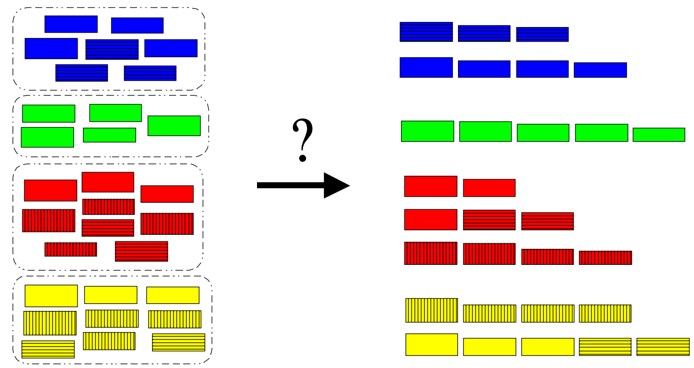
Disaggregation

First step: group the heats into sequences and define the correct casting order

- This step is done separately for each grade
- For sub-grades, the casting order is a→b→c
- Grades presorted by width
- Casting order: decreasing width
- Upper limit for width change between heats
- Number of heats/group restricted
- Minimize number of sequences (setup time)

Disaggregation

Grades are distinguished by color and sub-grades by pattern (a-solid, b-horizontal, c-vertical)



Disaggregation (MILP)

$$\min \sum_{l \in L} z_l$$

$$s.t.$$

$$\sum_{l \in L} x_{il} = 1 \quad \forall i \in I$$

$$\sum_{i \in I} x_{il} \le M \cdot z_l \quad \forall l \in L$$

$$\sum_{i \in I} q_{il} \le z_l \quad \forall l \in L$$

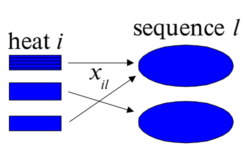
$$x_{il} - q_{il} \le \sum_{i' \in I | i < i'} F_{ii'} \cdot y_{ii'l} \quad \forall i \in I \quad \forall l \in L$$

$$(w_{i'} - w_i) y_{ii'l} \le 0 \quad \forall l \in L \quad \forall i, i' \in I, i < i'$$

$$z_l, x_{il} \in \{0,1\} \quad y_{ii'l}, q_l \in R$$

$M =$ upper limit of heats/seq.

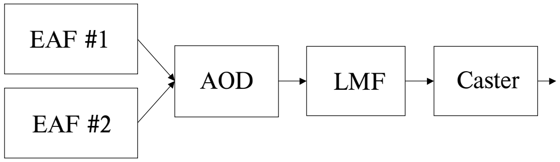
Compatibility matrix: $F_{ii'}=1$ if heat i' can be casted after i, else 0.



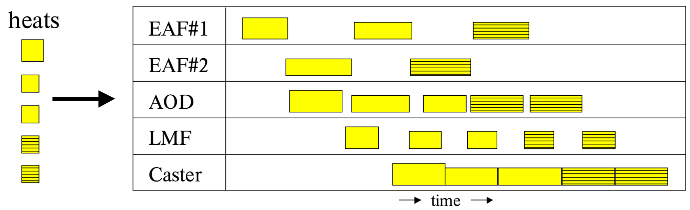
Scheduling of sub-problems

Second step: schedule each sequence, generated in the previous step, on equipment

- Minimize makespan, in-process times and hold-time violations
- Fixed casting order
- Formulated as jobshop scheduling problem



Scheduling of sub-problems



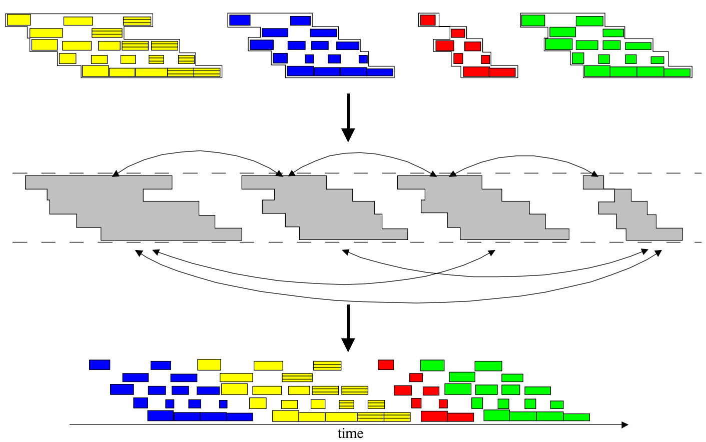
- All machine & product constraints considered
- Result: fully valid schedule for a small problem
- Simplified mathematical model presented
- Allocation and sequencing constraints

Aggregation

Third step: decide the order of scheduled sequences

- Results from step two ⇒ machine requirements
- Consider due dates, mold thickness changes etc.
- Special grade precedence constraints (e.g. wash grades)
- Minimize makespan, tardiness, earliness
- Formulate as flow-shop scheduling problem

Aggregation



Aggregation (MILP)

$$min \quad MS + \sum_{l \in L} (C_1 \cdot T_l^T + C_2 \cdot T_l^E)$$

$$s.t.$$

$$\sum_{g \in G} z_{g,l} = 1 \quad \forall l \in L$$

$$\sum_{l \in L} z_{g,l} = 1 \quad \forall g \in G$$

$$c_{m,l} = t_{m,l} + \sum_{g \in G} \tau_{p,m} \cdot z_{g,l} \quad \forall m \in M \quad \forall l \in L$$

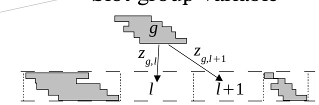
$$c_{m,l} + T_m^S \le t_{m,l+1} \quad \forall m \in M \quad \forall l \in L, l < |L|$$

$$c_{m,l} + T_l^E - T^T = \sum_{g \in G} T_g^{due} \cdot z_{g,l} \quad \forall l \in L \quad m = CST$$

$$c_{m,l} \le MS \quad \forall l \in L \quad m = CST$$

$$z_{g,l} \in \{0,1\} \quad t_{m,l}, c_{m,l}, T_l^T, T_l^E \in \mathbb{R}$$

Slot-group variable



Slot end-time

Subsequent slots

Tardiness and earliness

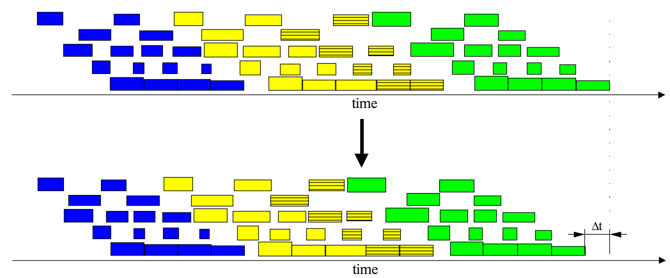
Improvement

Aggregation: no individual machines. To fill the time gaps and remove empty capacity between jobs, an LP problem is formulated

- Fixed assignments and sequences (no binary variables)
- Machine and capacity constraints
- Possible maintenance and service
- Exact evaluation of makespan
- Further improvement possible by making EAF into a variable

Improvement (LP)

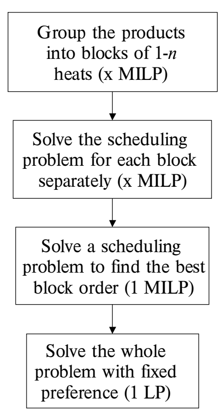
By solving the LP-model, a tighter schedule can be obtained. Problem model combines earlier steps with fixed discrete decisions.



Summary

Steps of decomposition strategy

- Disaggregation done by multiple MILP-problems
- Sub-problems solved by multiple MILP-problems
- Aggregation done by single MILP
- Result improvement with LP



Results

In the following, results from example problems are presented. The problems are weekly schedules. The problems were solved using GAMS/XPRESS-MP.

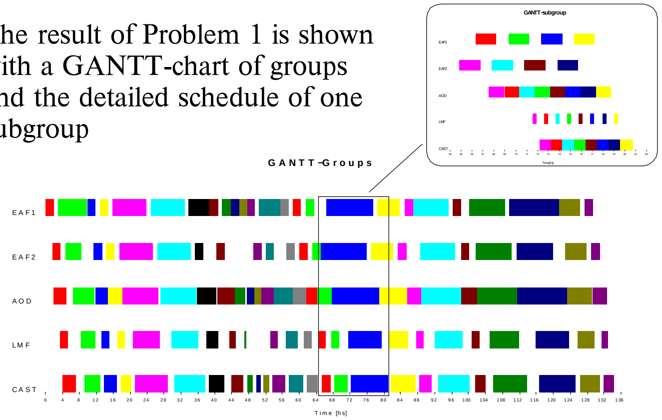
Problem	Grades/Subgr.	Heats/Seq.	CPU-mins*	Makespan
1	9/20	82/25	74	5d+14:14
2	10/17	80/20	334	5d+07:45
3	10/18	86/24	172	5d+21:02
4	9/17	84/21	173	5d+15:58
5	9/16	83/19	169	5d+12:53

MILP: Problem 1, single machine = 6724 0-1 vars.

*) PIII, 667MHz, Linux RedHat-6.2

Example

The result of Problem 1 is shown with a GANTT-chart of groups and the detailed schedule of one subgroup



Conclusions

- Future integration of CP/MILP methods (etc. ILOG)
- Role of optimization more important
- Only small part of problems identified yet
- More demanding problems, challenge for modeling
- Discrete optimization techniques needed for solving engineering problems: planning, design, synthesis
- General methods needed for combinatorial search

Acknowledgments

The authors would like to thank

- The Center for Advanced Process Decision-making (CAPD) at Carnegie Mellon University
- Kvaerner Metals