

# Modeling Complex Systems to Ensure Implementable Solutions: Imbedding Complexity in the Variable Definition

Amy Cohn

University of Michigan

December 9, 2005



Michigan**Engineering**

Industrial and Operations Engineering



# Overview

- Focus is on finding *implementable* solutions to large-scale network design/flow problems in complex systems
  - ◆ Timeliness as well as quality
  - ◆ Importance of real-world detail
  - ◆ “Regularity”
- Two main applications to discuss
  - ◆ Load matching/routing and empty balancing
    - ◆ Sarah Root, Alex Wang, UPS
  - ◆ Scheduling of automotive stamping
    - ◆ Ada Barlatt, Christy Griffen, Ford Motor Company

# Challenges of Complex Systems

- Large numbers of decisions
- Tightly inter-coupled because of network structures
- Very constrained resources (time, money, raw materials, labor...)
- Timeliness

# Math Programming Approach

## ■ Pros:

- ◆ Global viewpoint
- ◆ General solution techniques
- ◆ Established commercial solvers

## ■ Cons:

- ◆ Large numbers of constraints, variables
- ◆ Non-linearities
- ◆ Weak LP relaxations
- ◆ No exploitation of problem-specific structures

# Customized Algorithms

## ■ Pros:

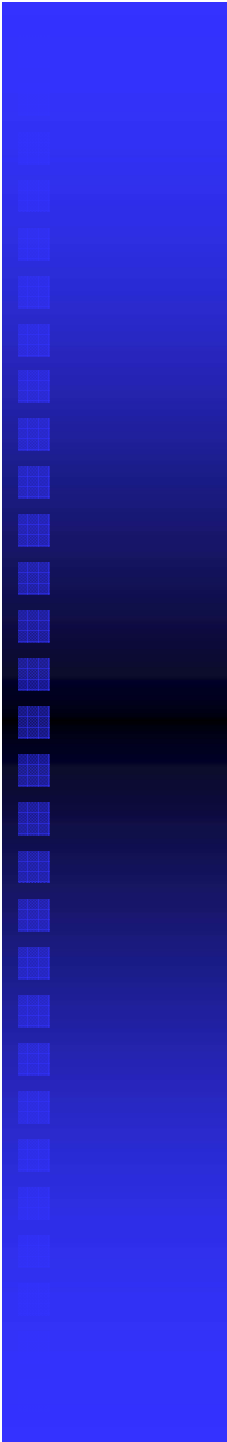
- ◆ Exploit problem structure
- ◆ More realistic
- ◆ Fast run times, convergence

## ■ Cons:

- ◆ Feasibility?
- ◆ Solution quality?
- ◆ Adaptability?

# Our Hybrid Approach

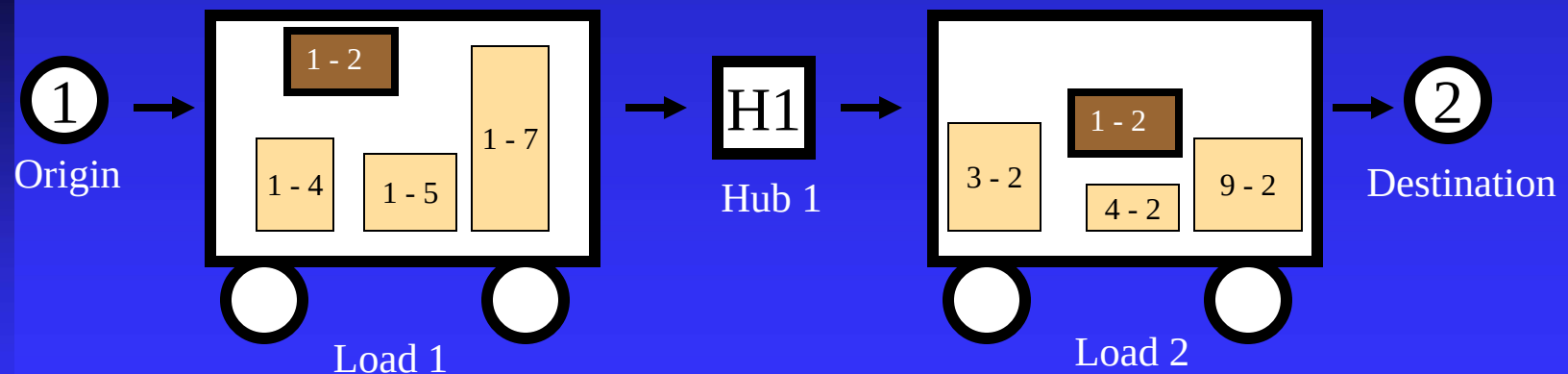
- Seek to combine the strengths of both approaches
- MP-based *master problem*
- *Complex variable definitions*
  - ◆ Multiple decisions in a single variable
  - ◆ Capture problem complexity within variable definition
  - ◆ Eg. crew pairing
- Application-specific algorithms to generate columns
  - ◆ Wealth of resources for generating candidate columns
  - ◆ Leveraging user expertise
  - ◆ Trading solution time vs quality



# Integrated Planning for Small Package Carriers

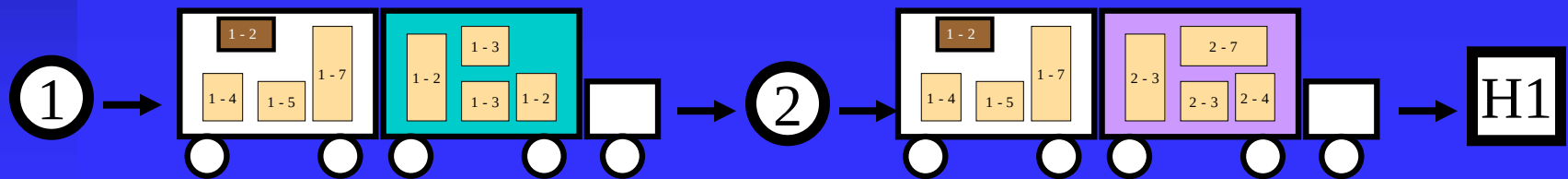
# Express Package Networks

- Packages move from origin to destination via one or more hubs at which freight is unloaded, sorted, and reloaded
- Each trailer movement is called a *load*



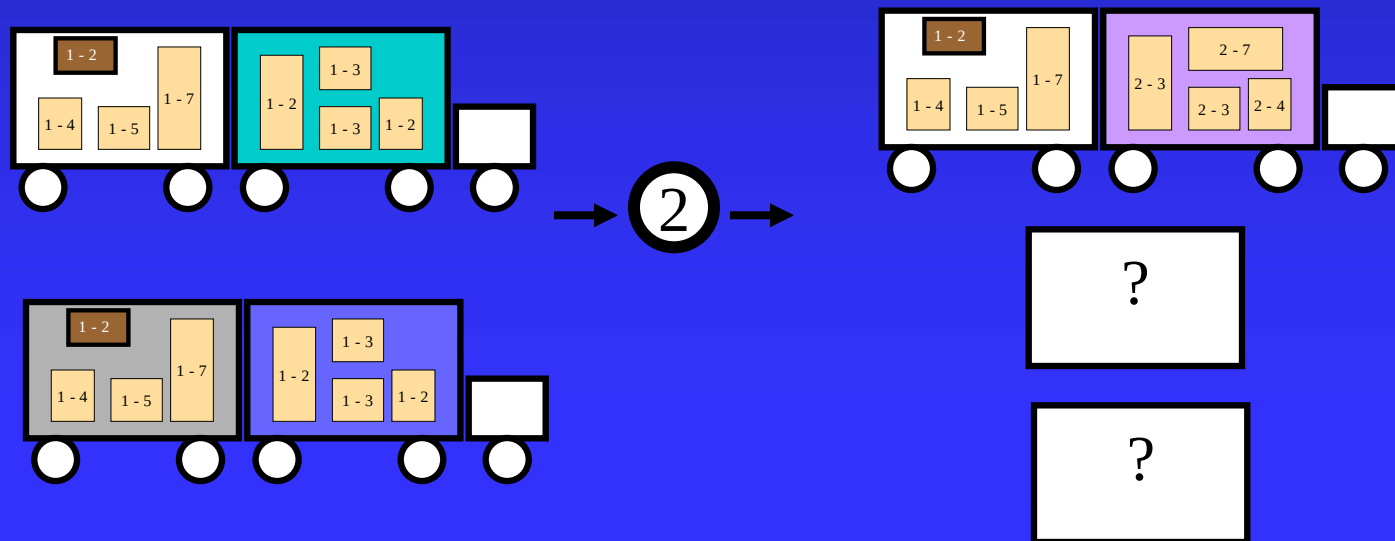
# Load Matching/Routing

- Loads (trailers) are often coupled with other trailers for reduced transportation cost
  - ◆ The cost of pulling two trailers behind the same tractor is less than twice the cost of pulling a single trailer
- Loads may stop at intermediate points along the way to re-configure the tractor
- For example, loads from Boston to Chicago and Detroit might travel together as far as Detroit, then the Chicago load would continue as a single
- What is the best way to combine and route the loads?



# Equipment Balancing

- Loaded trailer movements can result in system imbalances – some locations have more outbound loads than inbound, others more inbound than outbound
- We need to move empty trailers throughout the system to regain balance
- Which surplus nodes should supply which deficit nodes?



# Integrated Load Matching and Equipment Balancing Problem

- What loads should be matched?
  - ◆ When is it worth driving extra miles to combine two trailers?
- How should empty trailers be redistributed throughout the system?
  - ◆ At each location, total trailer flow out must equal total flow in
- Can we make these two decisions together for even greater savings?
  - ◆ Loaded and empty trailers can be paired together

# Problem Description: Inputs

- Set of loads (each load has an origin, destination, earliest available time, and latest delivery time)
- Cost per mile
- Mileage and travel time between each pair of locations

# Problem Description: Objective

Minimize total transportation cost

# Problem Description: Constraints

- Loads must move contiguously in space and time from origin to destination
- Loads must satisfy time windows
- Each location must be balanced (i.e. total number of trailers in must equal total number out, including both loaded and empty trailers)

# MCF Model with Side Constraints

- This can be thought of as a multi-commodity flow problem with side constraints
- Each load is a commodity that must be moved from origin to destination
- Side constraints are needed to ensure timing restrictions, linearize the objective function, and balance the network

# Multi-commodity flow based model

- Use time-space network to capture time constraints
- Decision variables
  - ◆  $x_{ijk}$  – flow of commodity  $k$  on arc  $(i, j)$
  - ◆  $y_{ij}$  – flow of empty trailers on arc  $(i, j)$
  - ◆  $s_{ij}$  – flow of single trailer combinations on arc  $(i, j)$
  - ◆  $d_{ij}$  – flow of double trailer combinations on arc  $(i, j)$

# Multi-commodity flow based model

$$\min \sum_{(i,j) \in A} c_{ij}^s s_{ij} + \sum_{(i,j) \in A} c_{ij}^d d_{ij}$$

$$\text{s.t.} \quad \sum_{i:(j,i) \in A} x_{jik} - \sum_{i:(i,j) \in A} x_{ijk} = b_{jk} \quad \forall j \text{ in } V, k \text{ in } K$$

$$s_{ij} + 2d_{ij} = \sum_{k \in K} x_{ijk} + y_{ij} \quad \forall (i,j) \text{ in } A$$

$$\sum_{j \in V_f} \left( \sum_{k \in K} b_{jk} + \sum_{i:(j,i) \in A} y_{ji} - \sum_{i:(i,j) \in A} y_{ji} \right) = 0 \quad \forall f \text{ in } F$$

$$x_{ijk}, y_{ij}, s_{ij}, d_{ij} \text{ in } \mathbb{Z}^+$$

# Challenges of MCF

- Very large number of nodes, constraints
  - ◆ Time/space network
  - ◆ If non-facility meet points are permitted, grows further
- Weakness of LP relaxation – half a double is always better than a whole single

# Alternative Model: Load Clusters

- As an alternative, we define the notion of a *load cluster* – a set of interconnected loads, the path each load takes, and the tractor configurations used to combine these loads
  - ◆ All loads must move fully from origin to destination
  - ◆ All loads must satisfy their time windows
  - ◆ No outside loads can be partially included in the configurations

# Example Cluster

A load from A to B moves directly, as a single trailer:



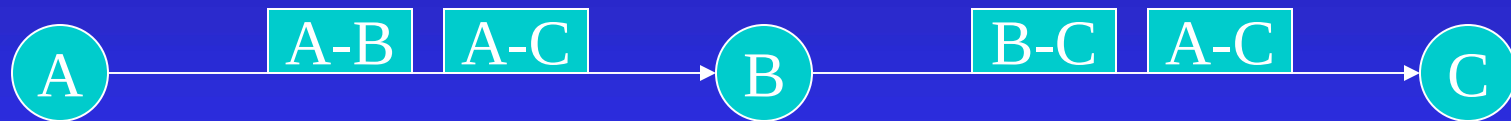
# Example Cluster

A load from A to B moves directly, in combination with an empty trailer:



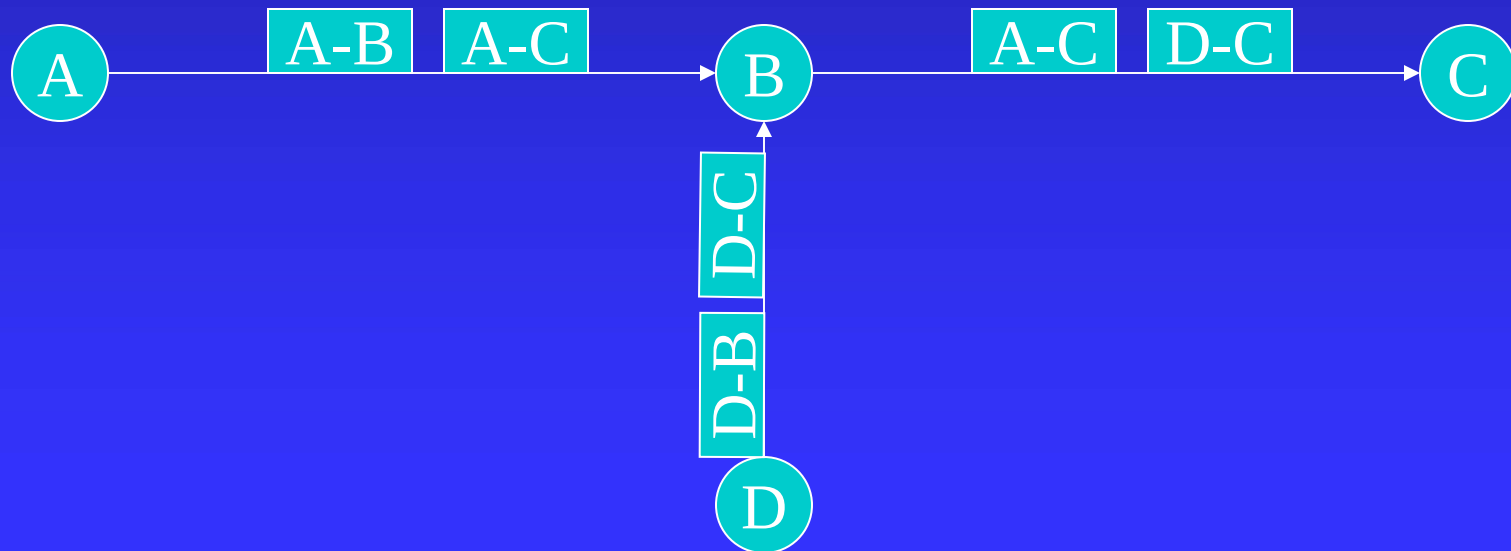
# Example Cluster

Loads from A to B and A to C move via B, where the A to B load is dropped off and a load from B to C is picked up:



# Example Cluster

Loads from A to B and A to C move via B,  
as do loads from D to B and D to C:



# The Cluster-Based Model

- $L$  is the set of loads
- $C$  is the set of load clusters
- $c_i$  is the cost of load cluster  $i$
- $F$  is the set of facilities
- $\delta_{il}$  is defined to be 1 if cluster  $i$  contains load  $l$  and 0 otherwise
- $\lambda_{if}$  is the net surplus of trailers at facility  $f$  relative to cluster  $i$
- $x_i$  is a binary variable that takes value 1 if cluster  $i$  is chosen and 0 otherwise

# The Cluster-Based Model

$$\text{Min.} \quad \sum_{i \in C} c_i x_i$$

$$\text{St} \quad \sum_{i \in C} \delta_{il} x_i = 1 \quad \forall l \in L$$

$$\sum_{i \in C} \lambda_{if} x_i = 0 \quad \forall f \in F$$

$$x_i \in \{0, 1\} \quad \forall i \in C$$

# Benefits

- The objective function is linear
- The LP relaxation is strengthened
- Timing issues are imbedded within the variable definition
- The number of constraints is greatly reduced
- Expanded solution space – the set of meet locations does not have to be limited to carrier facilities
- Multiple transportation modes, trailer configurations can easily be incorporated

# Original Solution Plan

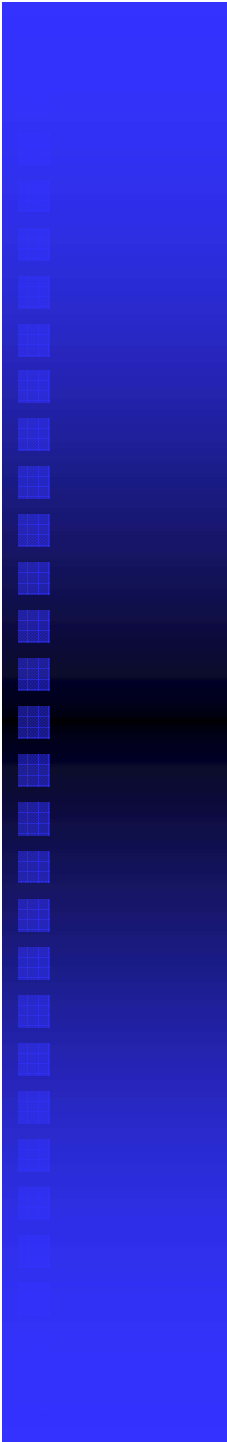
- Too many variables to explicitly enumerate in the model
- Use *branch-and-price*
  - ◆ Branch-and-bound to solve the IP
  - ◆ *Column generation* to solve the LP nodes
- What is the pricing problem???

# Computational Experience

- Started off with “initial” set of cluster *templates* to find seed duals
- Found significant improvement in solution quality just from using the templates!
  - ◆ Approximately 10% on a small network, with expectations of better savings from larger networks
- Feasibility
- Regularity/robustness
  - ◆ Simplifies cluster-based model
  - ◆ Makes MCF much more complex

# Future Research

- Symmetry
- Time window analysis
- Assigning volume to trailer types



# Scheduling for Automotive Stamping Facilities

# Project Goals

- Develop scheduling tools for Ford's stamping facility
- Expand beyond this to broader manufacturing system enterprise
- Do complex variable approaches work differently in manufacturing/supply chain than in transportation applications?

# Stamping Plant Application

## Automotive Stamping Plant

### Step 1

#### Sheet Metal Blanking

Forms coils of sheet metal into blank



### Step 3

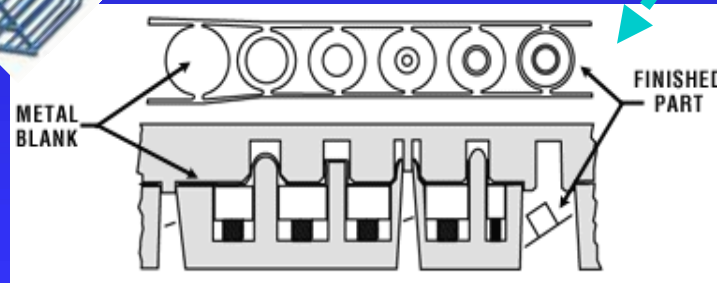
#### Sub-Assembly/Assembly

Pressed parts undergo additional operations  
(e.g. adding nuts and bolts)

### Step 2

#### Sheet Metal Stamping

Forms sheet metal into 3D parts



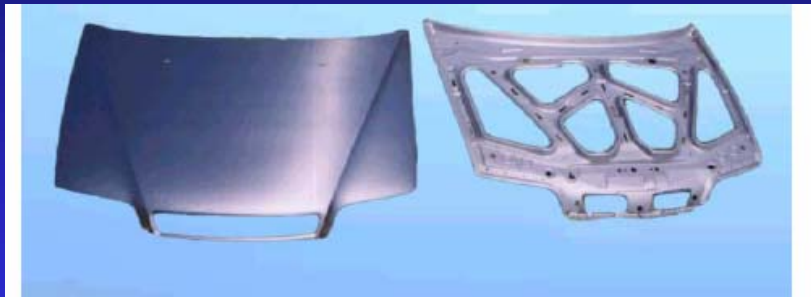
### Racks

Move parts between  
workstations

Diagrams from <http://www.advantagefabricatedmetals.com> and <http://www.maybury.com/>

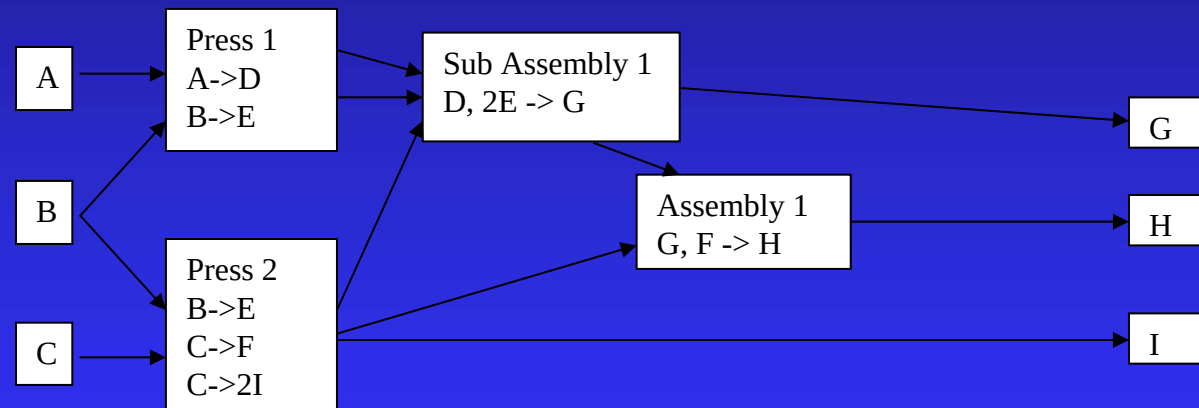
# Stamping Plant Application

## ■ Automotive Stamping Plant – Example Parts



Diagrams from <http://www.advantagefabricatedmetals.com> and <http://www.maybury.com/>

# Sample Product Flow



# Problem Definition

- Sequence all product demands (final and up-stream) on all machines so as to minimize cost while satisfying:
  - ◆ Demand constraints
  - ◆ Production limitations
  - ◆ Rack capacity
  - ◆ Rack movements
  - ◆ Machine setup time (implicit and explicit)

# Traditional Modeling Approach

- How do we define decision variables such that we can capture all these interactions?
  - ◆  $x_{pmd}$  = the time at which product  $p$  begins production on machine  $m$  to satisfy demand  $d$
  - ◆  $y_{pmt}$  = 1 if product  $p$  begins production on machine  $m$  at time  $t$  to satisfy demand  $d$
  - ◆  $z_{nmp}$  = 1 if the  $n^{\text{th}}$  rack produced on machine  $m$  is of type  $p$

# Variable Definition

- Most challenging constraints are machine change-overs
  - ◆ Critical to feasibility
  - ◆ Rely on substantial previous information
- Define a variable to represent a *feasible schedule for an entire machine*
  - ◆ One (potentially very large) set of variables for each machine
  - ◆ Captures all machine constraints *implicitly*
- Master problem considers all inter-machine constraints *explicitly*

# Model

## Constraints:

1) A machine can only be assigned one feasible schedule

$$\sum_{s \in S[m]} x[s] = 1$$

$$\forall m \in M$$

2) Production can only occur if the rack is at the machine

$$\sum_{s \in S[o[p]]} b^w[n, p, s] * x[s] - \sum_{s' \in S[e[p]]} g[n - r[p], p, s'] * x[s'] \geq 0$$

$$\forall p \in P, n : r[p] + 1 \leq n \leq d[p]$$

3) The time the nth rack of product p is completed has to be earlier than the time that it is consumed on another machine.

$$\sum_{s' \in S[e[p]]} u[n, p, s'] * x[s'] - \sum_{s \in S[o[p]]} t[n, p, s] * x[s] \geq 0$$

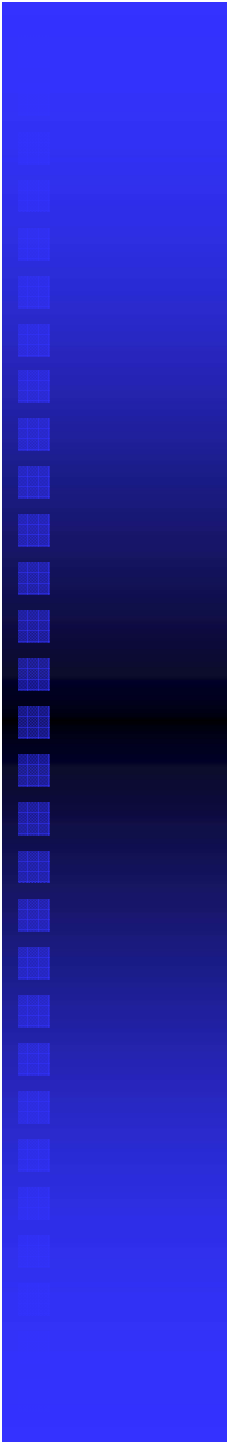
$$\forall p \in P, n : n = 1..d[p]$$

# Current Status

- Working on coding infrastructure
  - ◆ C++ calling CPLEX
- Developing algorithms to generate candidate machine schedules
- Validating model with users

# Future Research

- Expanding level of detail
  - ◆ Cranes
  - ◆ Labor
- Expanded supply chain scope
- Generalization
- Variability, robustness, and responsiveness



# Conclusions, Challenges, and Future Research

# Conclusions

- Many applied problems are hard!
- Simplification can lead to tractability but real-world infeasibility
- Optimality is not all it's cracked up to be...
- ...but the quality of our solution really matters!
- Sometimes, a hybrid approach with a complex variable model can help

# Challenges

- How to generate candidate columns
- Feasibility vs. optimality
- Lower bounds
- Degeneracy, convergence

# Future Research

- Expanding problem scope
- Generalizing results
- Robustness