PRINCETON UNIVERSITY

# Chemical Production Scheduling:

## Mixed-Integer Programming Models and Methods

**Christos Maravelias**

**Introduction and Basic Methods**

**Chapters 1, 3-5, 7**

**(Five 50-minute Lectures)**

**Cambridge University Press**

***Chemical Engineering Series***

# Outline

- Basics
- Single-unit problems
- Single-stage problems
- Multi-stage problems
- Network problems

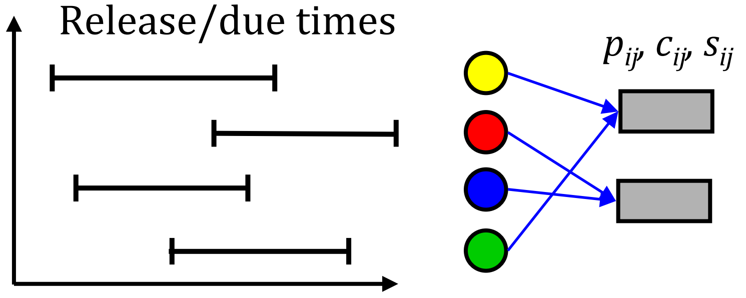
# What is Scheduling?

*It is the allocation of limited resources to tasks over time*

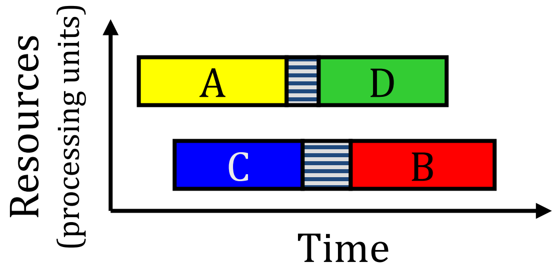
Michael Pinedo, 1998

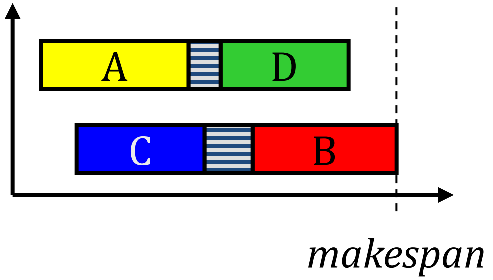
## Single-Stage Scheduling

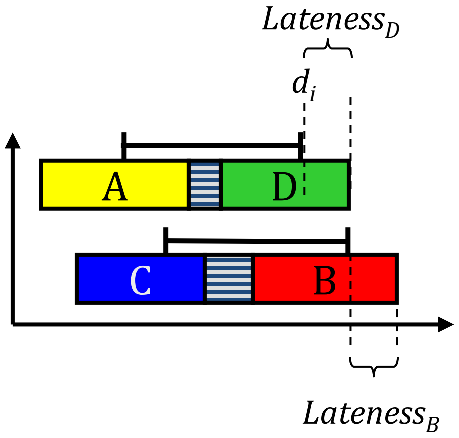
$N$ jobs to be processed in $M$ machines

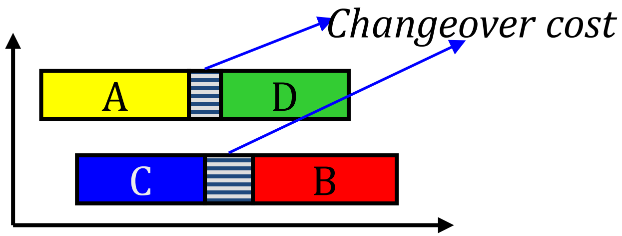



- Assignment of jobs to machines
- Sequencing of jobs in the same machine



# Preliminaries

PRINCETON UNIVERSITY

- Systematic scheduling practiced in manufacturing since early 20th century
- First scheduling publications in the early 1950s
  Salveson, M.E. On a quantitative method in production planning and scheduling. *Econometrica*, 20(9), 1952
  Johnson, S.M. Optimal two- and three-stage production schedules with setup times. *Naval Research Logistics Quarterly*, 1(1), 61-68, 1954.
- Extensive research in 1970s
  - Closely related to developments in computing and algorithms
  - Computational Complexity: *Job Sequencing* one of 21 NP-complete problems in (Karp, 1972)
- Widespread applications
  - Airlines industry (e.g., fleet, crew scheduling)
  - Transportation (e.g., vehicle routing)
  - Government, educational institutions (e.g., class scheduling)
  - Sports
  - Services (e.g., service center scheduling)
  - Manufacturing industries
- Chemical industries
  - Batch process scheduling (e.g., pharma, food industry, fine chemicals)
  - Continuous process scheduling (e.g., polymerization)
  - Transportation and unloading of crude oil
- Very challenging problem: Small problems can be very hard
  - Most *Open* problems in MIPLIB are scheduling related
    - Railway scheduling: 1,500 constraints, 1,083 variables, 794 binaries
    - Production planning: 1,307 constraints, 792 variables, 240 binaries
    - Crew scheduling: 1,803 constraints, 11,612 variables, 9,720 binaries

# Problem Statement



Given are:

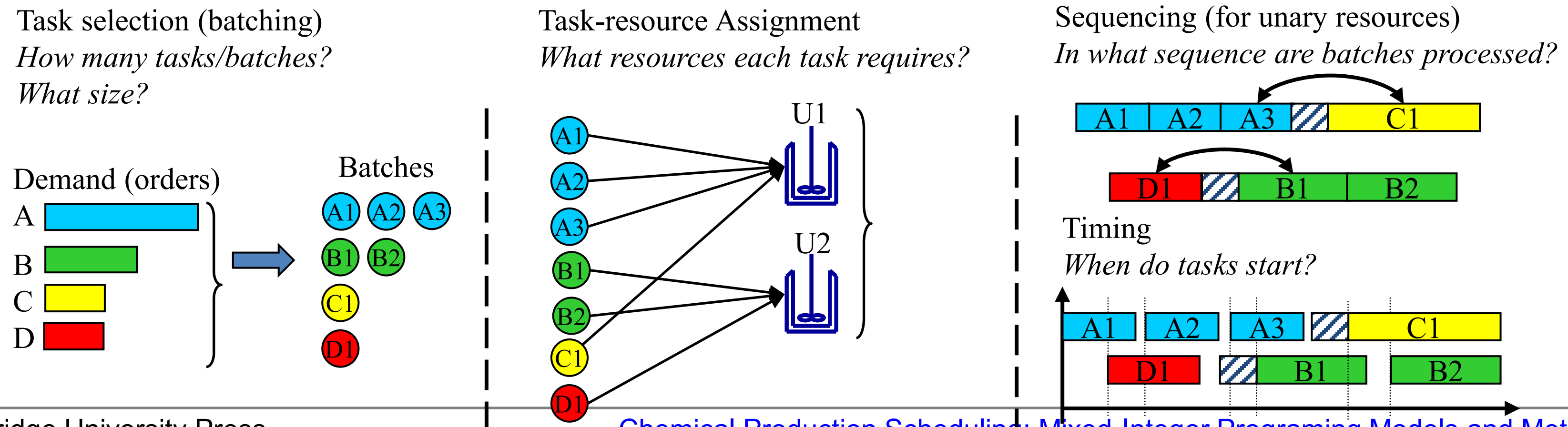
a) Production facility data; e.g., processing and storage unit capacities, unit connectivity, etc.
b) Production recipes; i.e., mixing rules, processing times/rates, utility requirements, etc.
c) Equipment unit – task compatibility.
d) Production costs; e.g., raw materials, utilities, changeover, etc.
e) Material availability; e.g., deliveries (amount and date) of raw materials.
f) Resource availability; e.g., maintenance schedule, resource allocation from planning, etc.
g) Production targets or orders with due dates.

Our goal is to find a least cost schedule that meets production targets subject to resource constraints.

Alternative objective functions are the minimization of tardiness or lateness (minimization of backlog cost) or the minimization of earliness (minimization of inventory cost) or the maximization of profit.

In the general problem, we seek to optimize our objective by making four types of decisions:

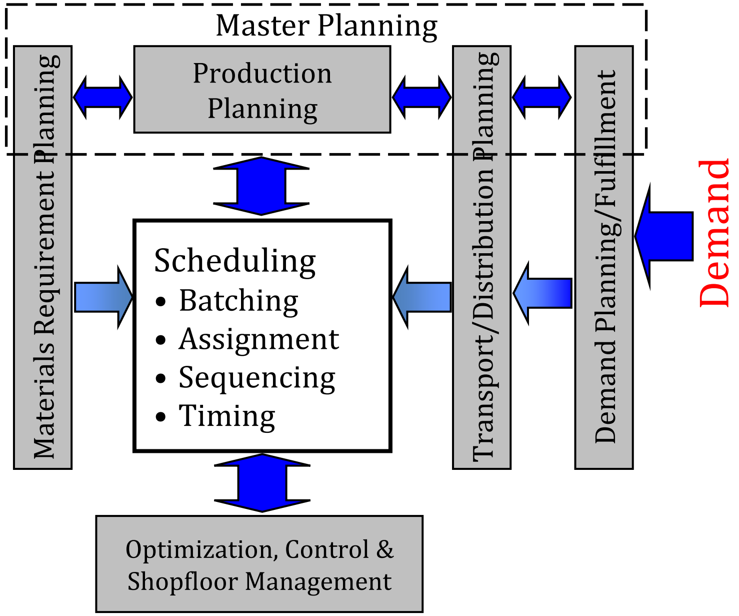
a) Selection and sizing of batches to be carried out (batching)
b) Assignment of batches to processing units or general resources.
c) Sequencing and timing of batches on processing units.



# Scheduling in the Supply Chain



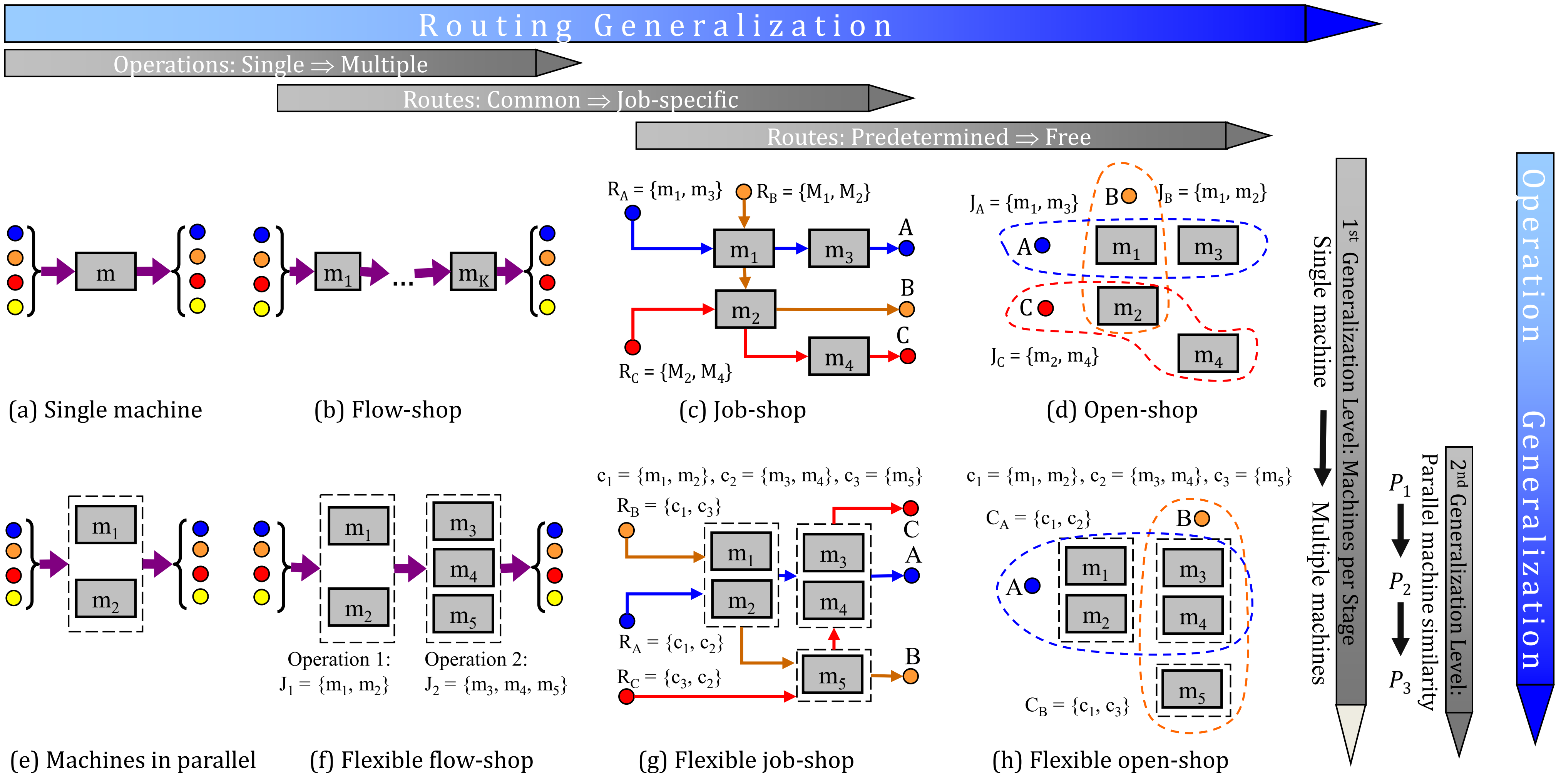
- Scheduling is only one planning function in SC optimization
- Interactions with other planning functions determine:
  - Type of scheduling problem: cyclic *vs*. short-term scheduling
  - Overarching production goal → scheduling objective
  - Optimization decisions for scheduling
  - Inputs: resource availability, production targets, etc.




- Wide range of problems due to market environments and company planning structure
- Variability increases due to multiple production environments and processing restrictions

# Discrete Manufacturing: Machine Environments

| | | | |
|---|---|---|---|
| $i \in \mathbf{I}$: | Jobs | $\mathbf{R}_i$: | Routing of job *i* (jobshop) |
| $j \in \mathbf{J}$: | Machines | $\mathbf{J}_i$: | Machines for job *i* (open-shop) |
| $k \in \mathbf{K}$: | Operations | $\mathbf{J}_k$: | Machines in stage *k* |
| $c \in \mathbf{C}$: | Work centers | $\mathbf{C}_i$: | Centers for job *i* |

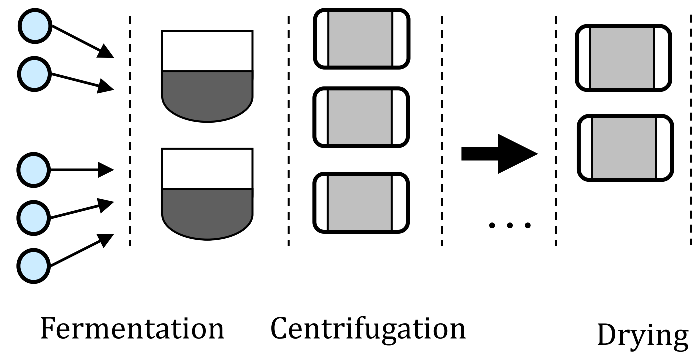


# Basic Insights

- **Discrete manufacturing**
  - A *job* (e.g., chip) moves through *operations* consisting of parallel *machines*
  - Each job is not split into multiple jobs; jobs are not *merged*
- **Chemical production: *tasks* involve fluids**
  - Fluids (from different batches) can be mixed into a vessel; output of a batch can be used in multiple downstream batches
  - No mixing/splitting restrictions may be *added* (e.g., quality control)

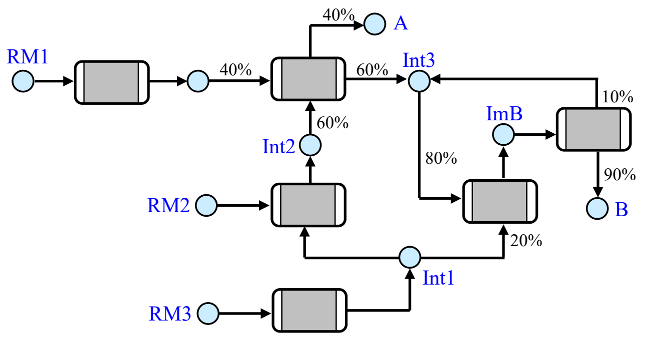
### Sequential processing

- Materials cannot be mixed/split/recycled
- Problem defined in terms of: *Batches*, stages, and units
- Usually, no utility & storage constraints




### Network processing

- Materials can be mixed/split/recycled
- Problem defined in terms of: *Materials*, tasks, resources
- Utility and storage constraints modeled




### Discrete Manufacturing ⇨ Chemical Production

*Absence of* material handling restrictions makes chemical production scheduling problems different

- The notion of a batch (job) moving through stages cannot be defined
- Consumed/produced materials must be monitored
- Model material balances

### Notation

- Jobs ⇒ Orders or batches
- Operations ⇒ Tasks
- Machines ⇒ Units

### Key Features

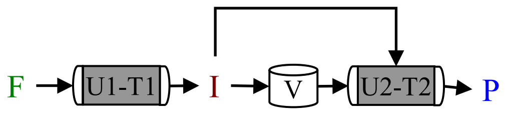
- Modeling of storage is important (solid, liquid and gas phases) ⇒ Storage vessels & states
- Utilities are also important (steam, water, electricity) ⇒ Utilities (or resources)
- No Preemption
- [Variable Processing Times]
- Long changeover times

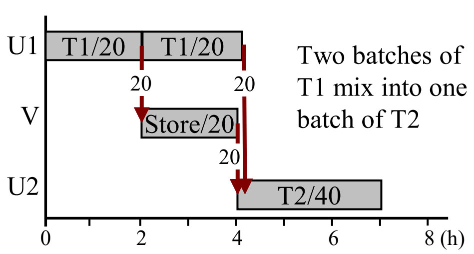
# Basic Insights

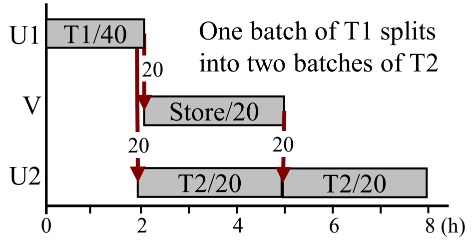
## *Sequential-looking* process

<u>Tasks:</u>
T1 in U1: F → I (2 h);
T2 in U2: I → P (3 h)
<u>Capacities: $\beta^{min}/\beta^{max}$ (kg):</u>
U1: 20/40, U2: 20/40, V: 0/40v

No special restrictions for intermediate I



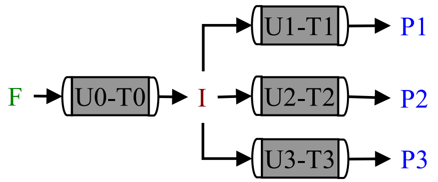



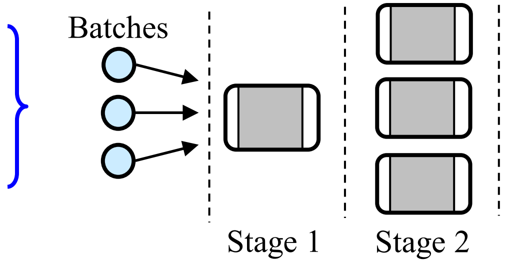



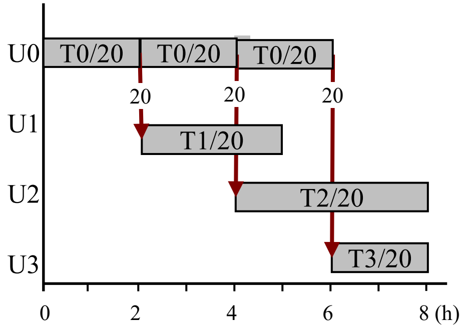

## *Network-looking* process

<u>Tasks:</u>
T0/U0: F → I (2 h);
T1/U1: I → P1 (3 h)
T2/U2: I → P2 (4 h);
T3/U3: I → P3 (2 h)
<u>Capacities: $\beta^{min}/\beta^{max}$ (kg):</u>
All units should run 20 kg batches

Each batch of intermediate I should be consumed by a single downstream batch



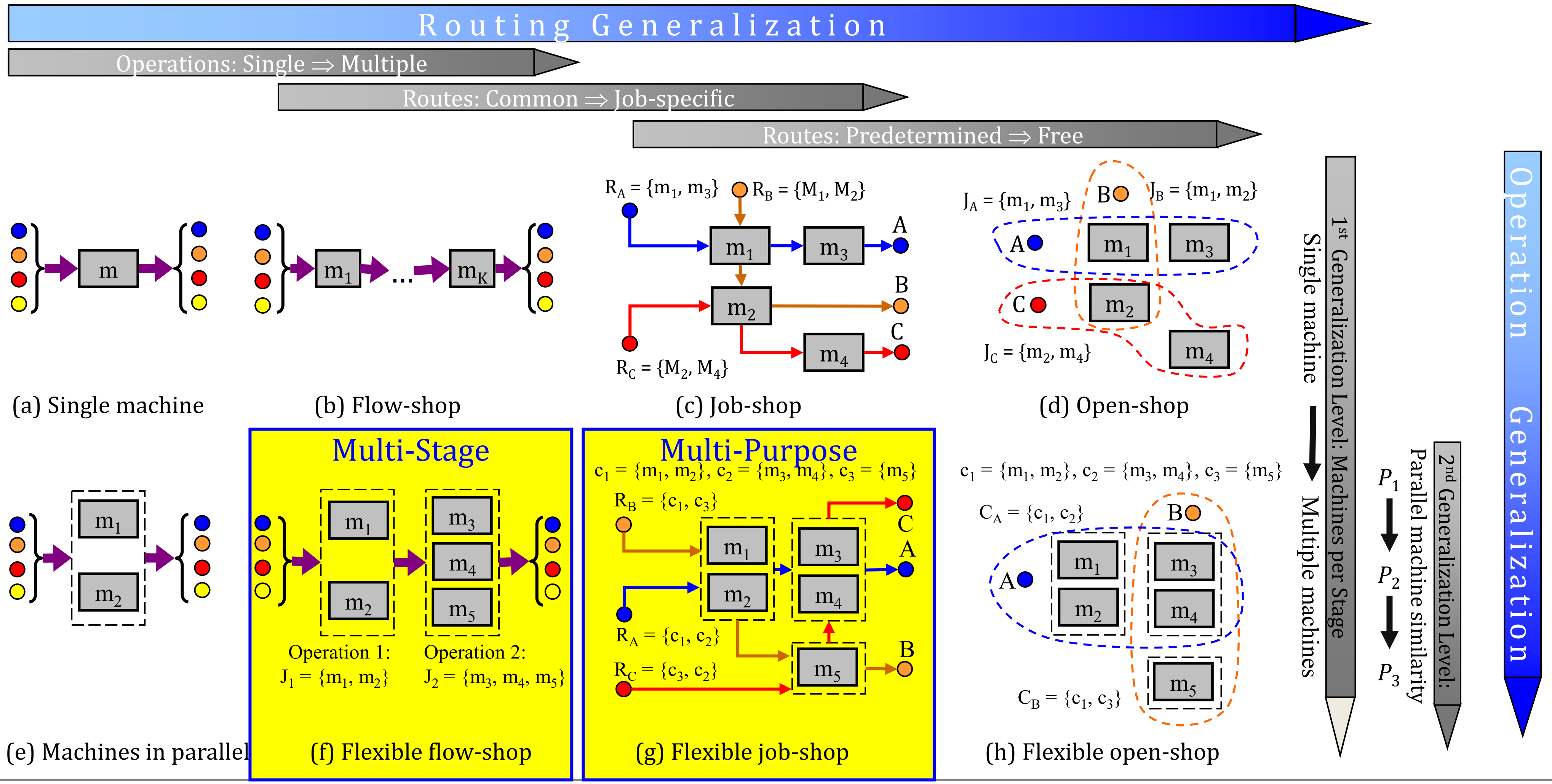




- Sequential *plant structure* does NOT imply *sequential processing*
- Materials handling restrictions determine type of processing

# Discrete Manufacturing: Machine Environments

| | | | |
|---|---|---|---|
| $i \in \mathbf{I}$: | Jobs | $\mathbf{R}_i$: | Routing of job $i$ (jobshop) |
| $j \in \mathbf{J}$: | Machines | $\mathbf{J}_i$: | Machines for job $i$ (open-shop) |
| $k \in \mathbf{K}$: | Operations | $\mathbf{J}_k$: | Machines in stage $k$ |
| $c \in \mathbf{C}$: | Work centers | $\mathbf{C}_i$: | Centers for job $i$ |



# Problem Classes



$$\alpha / \beta / \gamma$$

$\alpha$ Production environment

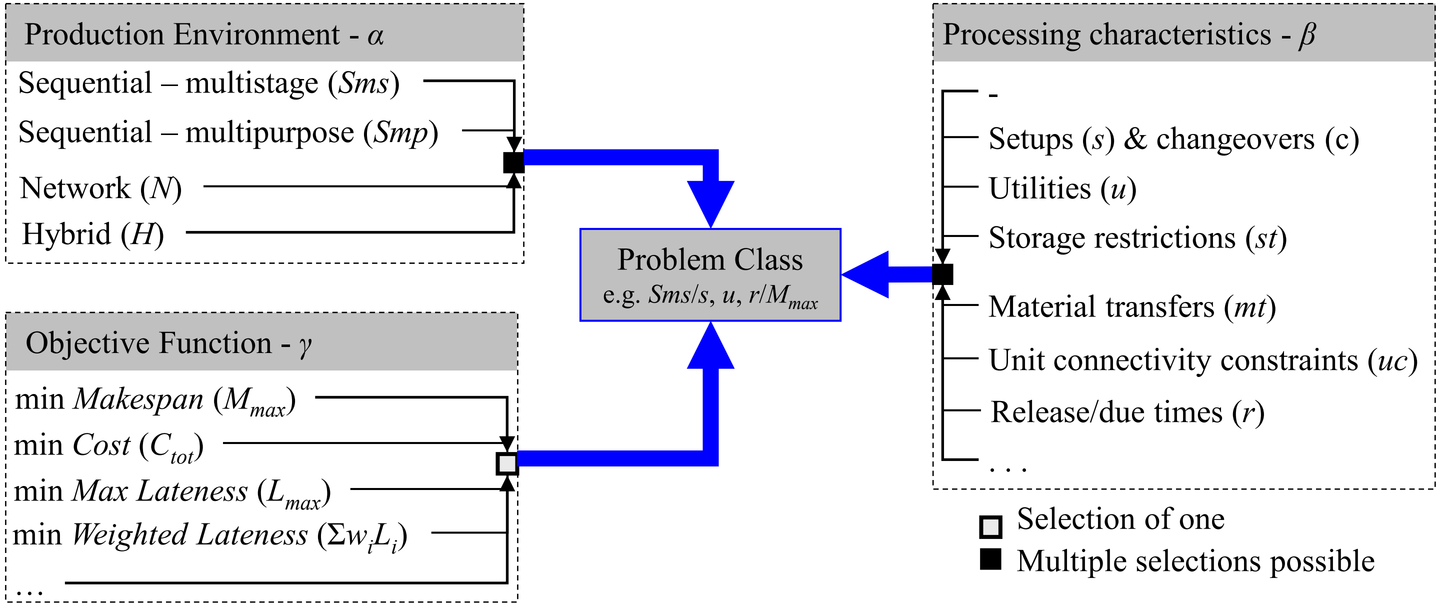
- Material handling constraints are key (not facility structure): sequential, network, hybrid

$\beta$ Processing characteristics

- Typical characteristics: setups, changeovers, release/due times, etc.
- *Chemical production* characteristics: storage constraints, material transfer constraints, utilities, etc.

$\gamma$ Objective functions

- min *Cost*, min *Lateness*, min *Tardiness*, max *Throughput*, etc.



# Notation

PRINCETON UNIVERSITY

We use:

- lower case Latin characters for indices,
- uppercase Latin bold letters for sets,
- uppercase Latin characters for variables,
- Greek letters for parameters, and
- regular uppercase Latin letters for set elements.

All subsets of a set will be denoted by the letter used for the set and a subscript and/or superscript. Specifically,

- indices are used as subscripts to denote subsets that are index-specific
  (e.g., the subset of units **J** in stage $i$ is denoted by $\mathbf{J}_k$);
- uppercase letters are used as superscripts to further differentiate subsets
  (e.g., if **J** is the set of units, the subset of processing units and storage vessels are denoted by $\mathbf{J}^P$ and $\mathbf{J}^S$, respectively).

Parameters and variables may also have superscripts for differentiation;
e.g., , the processing cost of task $i$ is denoted by $\gamma_i^P$ while the changeover cost between tasks $i$ and $i'$ is denoted by $\gamma_i^{CH}$.

# Outline

- ❑ Basics
- ❑ Single-unit problems
- ❑ Single-stage problems
- ❑ Multi-stage problems
- ❑ Network problems

# Single Unit Problem

- There is only one unit available with known capacity
- The minimum number of batches to meet orders can be pre-calculated,
- We have predefined set of batches (some of which may be identical)
- We are given:
  - A set of batches, $i \in \mathbf{I}$, to be carried out on a single unit, U.
  - Processing time of batch $i$ is $\tau_i$
- Is this an optimization problem?
- Since all batches will be carried out on the same unit, we observe the following:
  - The minimum makespan will be equal to $\sum_i \tau_i$ regardless of the sequencing of batches.
  - If there is a processing cost, $\gamma_i^P$ the total processing cost will be $\sum_i \gamma_i^P$.
  - Since there are no due times, lateness, tardiness, and earliness cannot be defined.
  - Since the number and size of batches is fixed, the total production and thus profit are fixed.
- There are two features that make this problem more interesting:
  - Each batch is subject to release and due times, $\rho_i$ and $\varepsilon_i$, respectively.
  - There are sequence dependent changeover times, $\sigma_{ii'}$, and costs, $\gamma_{ii'}^{CH}$.
- Note: setup (i.e., sequence independent) times and costs are not relevant for single-unit problems because:
  - all feasible solutions have exactly the same setup cost (since all batches are assigned to the same unit) and
  - setup times can be simply added to the processing times, so the problem statement remains the same.

# Single Unit Problem: Global Sequence-Based Models

PRINCETON UNIVERSITY

## Basic idea

- Use binary variable to represent the relative order (sequence) and then employ big-M constraints to enforce a *no-overlap condition*
- $Y_{ii'} \in \{0,1\} \coloneqq 1$ if batch $i$ is processed before batch $i'$
- $S_i \in \mathbb{R}_+ \coloneqq$ start time of batch $i$

## Equations

- How do we ensure that batch $i$ is finished before batch $i'$ starts if $Y_{ii'} = 1$ (disjunctive constraint)?

  $$S_i + \tau_i \le S_{i'} + M(1 - Y_{ii'}), \quad i, i' \ne i \qquad (1)$$

  where $M$ is a sufficiently large number.
  (Note: we could have defined $E_i = S_i + \tau_i$ but we used $S_i + \tau_i$ instead).

- Eq (1) enforces correct batch timing provided that binary variables $Y_{ii'}$ assume *correct* values.
  How do we ensure that a sequencing relationship is established for every pair of tasks (i.e., no *overlap*)?

  $$Y_{ii'} + Y_{i'i} = 1, \quad i, i' > i \qquad (2)$$

- How do we enforce release and due dates?

  $$S_i \ge \rho_i, \quad i \qquad (3); \qquad S_i + \tau_i \le \varepsilon_i, \quad i \qquad (4)$$

- If due times can be violated, at a cost, then (4) can be replaced with:

  $$S_i + \tau_i \le \varepsilon_i + L_i, \quad i \qquad (5)$$

  where $L_i$ is the lateness/tardiness of batch $i$.
  - If the objective function is to minimize lateness then (5) will be satisfied as equality; i.e, $L_i$ will assume the smallest possible value, which can be negative.
  - If the objective tardiness minimization, then we should enforce $L_i \ge 0$ and eq (3.5) will be satisfied as equality only if $S_i + \tau_i \le \varepsilon_i$.

# Single Unit Problem: Global Sequence-Based Models



- How do we enforce changeover times?

  If the triangle inequality, $\sigma_{ii''} < \sigma_{ii'} + \tau_{i'} + \sigma_{i'i''}$, holds for all $i', i', ii''$ then changeover times can be addressed through:

$$S_i + \tau_i + \sigma_{ii'} \leq S_{i'} + M(1 - Y_{ii'}), \qquad i, i' \neq i$$

- Can we model changeover costs using global sequencing variables?

  No, because for a given batch $i'$ we potentially have many variables with $Y_{ii'} = 1$

- Makespan, $MS$, minimization, we have:

$$\min MS$$

  and require:

$$MS \geq S_i + \tau_i, \quad i$$

- For earliness minimization,

$$\min \sum_i \omega_i(\varepsilon_i - (S_i + \tau_i))$$

  where $\omega_i$ is a weight factor

- Do you see a problem with the above equation?

  Yes, we can benefit from increasing the lateness of batch $i$ in order to decrease earliness of batch $i'$
  We should assume that due times are met.

- The objective function for total weighted lateness, $L^{TOT}$, and tardiness, $T^{TOT}$, minimization is,

$$\min \sum_i \omega_i L_i$$

  where

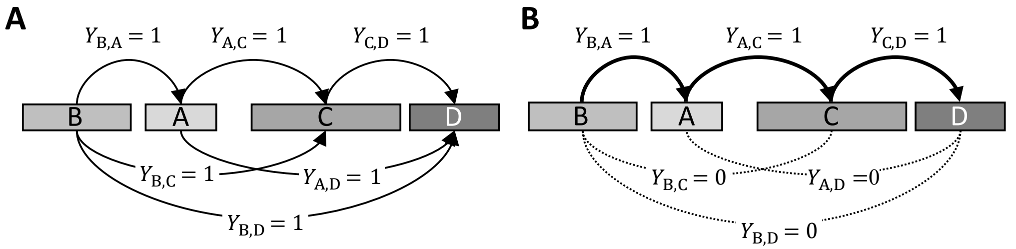
$$L_i \geq (S_i + \tau_i) - \varepsilon_i, \quad i$$

  and for tardiness we also require $L_i \geq 0$.

# Single Unit Problem: Immediate Sequence-Based Models

## Preliminaries

- In immediate-sequencing, binary $Y_{ii'}$ if batch $i$ is immediately followed by $i'$, which means that the number of variables being equal to 1 is different:
  - In global sequencing, $|\mathbf{I}|(|\mathbf{I}|-1)/2\ \ Y_{ii'}$ variables are equal to 1 (why?)
  - In immediate sequencing, there are only $|\mathbf{I}|-1$ variables being equal to 1




- Main difference: equations that activate $Y_{ii'}$; specifically, (2) should be replaced with alternative equations.

## Basic Ideas and Equations

- We exploit that all batches have one immediate predecessor/successor, except the first/last which has no predecessor/successor.
  - $Y_i^F \in \{0,1\} := 1$ if batch $i$ is processed first in unit U.
  - $Y_i^L \in \{0,1\} := 1$ if batch $i$ is processed last in unit U.
- The two new variables should satisfy:

  $\sum_i Y_i^F = \sum_i Y_i^L = 1$ (1)

- The condition on the number of immediate predecessors and successors is then enforced via,

  $\sum_i Y_{ii'} = 1 - Y_{i'}^F, \quad i'$ (2) $\qquad \sum_{i'} Y_{ii'} = 1 - Y_i^L, \quad i$ (3)

- Eqs (1) - (3) can be used to *activate* immediate sequencing binary variables $Y_{ii'}$.
- Once $Y_{ii'}$ assume *feasible* values, the equations presented in the previous slide can be used to enforce the remaining constraints.
- An alternative approach exploits the fact that exactly **I**-*1* immediate sequencing variables will be activated in any feasible solution:

  $\sum_{i,i'} Y_{ii'} = |\mathbf{I}| - 1$ (4)

- Eq (4) coupled with (2)-(3) and timing constraints (e.g., (1) in previous slide) enforce that only one batch is processed at a time

# Single Unit Problem: Continuous Time Grid-Based Models

PRINCETON UNIVERSITY

## Preliminaries

- Time-grid-based approaches require the definition of a time grid onto which the execution of tasks/batches are mapped.
- When a continuous grid is adopted, the horizon, $\eta$, is divided into $|\mathbf{I}|$ periods of unknown length.
- Each period $t \in \mathbf{T}$ starts at time point $t-1$ and ends at time point $t$; the timing of point $t$ is $T_t$.
- The horizon starts at $\rho^0 = \min_i\{\rho_i\}$; if deadlines are given, then it ends at $\varepsilon^F = \max_i\{\varepsilon_i\}$ (i.e., $\eta = \varepsilon^F - \rho^0$)
- The set of necessary periods is $\mathbf{T} = \{1, 2, \ldots, |\mathbf{I}|\}$, and the set of points is $\mathbf{T}' = \mathbf{T} \cup \{0\} = \{0, 1, 2, \ldots, |\mathbf{I}|\}$.
- The timing of time points should satisfy the following constraints:

$$T_0 = 0,\ T_{|\mathbf{T}|} = \eta;\ T_t \geq T_{t-1},\ \ t \qquad (1)$$

## Equations

- Basic idea: match a batch and a time period through the introduction of $X_{it} \in \{0,1\}$: is equal to 1 if batch $i$ is allocated to time period $t$,
- Enforce that each batch is allocated to exactly one period and each period is used for exactly one batch:

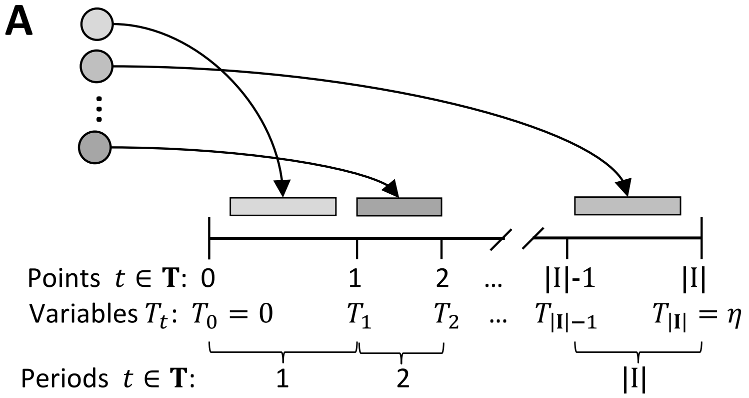
$$\textstyle\sum_t X_{it} = 1,\ \ i \qquad (2) \qquad\qquad \sum_i X_{it} = 1,\ \ t \qquad (3)$$

- Variables $X_{it}$ are used to enforce no-overlap between batches; if $S_i$ are the batch start time variables, then:

$$S_i \geq T_{t-1} - M(1 - X_{it}),\ \ i,t\ (4); \qquad S_i + \tau_i \leq T_t + M(1 - X_{it}),\ \ i,t\ \ (5)$$

- The first model based on a continuous time grid consists of eqs (1) – (5)
- Same objective functions can be used as in global-sequencing models

A



# Single Unit Problem: Continuous Time Grid-Based Models

PRINCETON UNIVERSITY

## Alternative Models

- A slightly different model is obtained when $S_i$ is required to be equal to the start of the period to which batch $i$ is allocated:

  $T_{t-1} - M(1 - X_{it}) \leq S_i \leq T_{t-1} + M(1 - X_{it}), \quad i, t \qquad (6)$

- Alternatively, the batch end time, $S_i + \tau_i$, can be enforced to be equal to the end time of the period to which is allocated

  $T_t - M(1 - X_{it}) \leq S_i + \tau_i \leq T_t + M(1 - X_{it}), \quad i, t \qquad (7)$

- A different model is obtained if batch start time, $S_i$, is removed; no overlap condition enforced using $T_t$:

  $T_t \geq T_{t-1} + \sum_i \tau_i X_{it}, \quad t \qquad (8)$

  where $\sum_i \tau_i X_{it}$ represents the processing time of the batch allocated to $t$

- Since variable $S_i$ is removed, $T_t$ is used to enforce release and due times:

  $T_{t-1} \geq \rho_i X_{it}, \quad i, t \qquad (9); \; T_{t-1} + \tau_i X_{it} \leq \varepsilon_i X_{it} + \eta(1 - X_{it}), \quad i, t \; (10)$

  assuming that batch processing starts at the start of the allocated period

- In general, models based on a continuous time grid can employ different variables and sets of constraints to enforce:
  (1) that a batch starts and ends within the period it is allocated to, and
  (2) release and due time constraints.

- Consequently, the same schedule can be represented by different variables and, when the same variables are employed, by a different solution vector.

# Single Unit Problem: Continuous Time Grid-Based Models

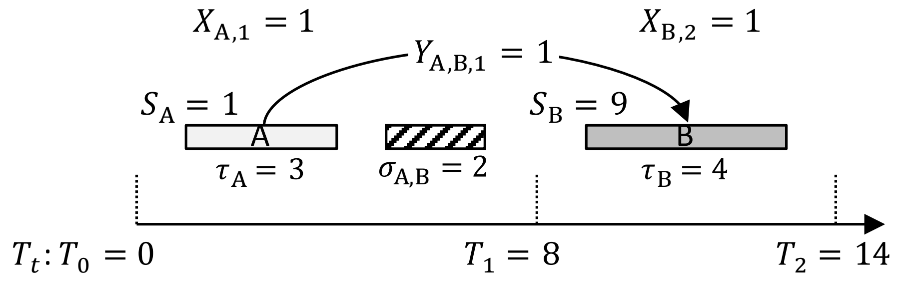
## Changeovers

- Accounting for changeovers is, in general, more challenging with time-grid-based models.
- One approach is to introduce period-specific immediate sequencing variables:
  $Y_{ii't} \in \{0,1\}$: are equal to 1 if the $i \rightarrow i'$ changeover occurs before point $t$ prior to the execution of batch $i'$
- Activation of $Y_{ii't}$ is accomplished using the batch-period allocation binaries,

$$Y_{ii't} \geq X_{it} + X_{i',t+1} - 1, \quad i, i' \neq i, t \qquad (11)$$

$$\sum_{i'} Y_{ii't} \leq X_{it}, \quad i, t \qquad (12)$$

$$\sum_{i} Y_{ii',t-1} \leq X_{i't}, \quad i', t \qquad (13)$$




- Then use a new timing constraint, whose most intuitive (but not tightest) form is,

$$S_i + \tau_i + \sigma_{ii'} \leq T_t + M(1 - Y_{ii't}), \quad i, i' \neq i, t \qquad (14)$$

- Note that sequencing variables $Y_{ii't}$ can be defined in multiple ways.
  E.g., $Y_{ii't} = 1$ if the $i \rightarrow i'$ changeover occurs within $t$ prior to the execution of $i'$ which also occurs in period $t$
  (Can you see the difference? How should (11) - (13) be modified in this case?).
- The minimization of changeover cost can then be expressed as follows:

$$\min \sum_{i,i'} \sum_t \gamma_{ii'}^{CH} Y_{ii't}$$

- The modeling of changeover times and costs using models based on a continuous time grid requires the introduction of new binary variables $Y_{ii't}$ which leads to large models.

# Single Unit Problem: Discrete Time Grid-Based Models

PRINCETON UNIVERSITY

## Preliminaries

- We use $n \in \mathbf{N} = \{0,1,2, \dots |\mathbf{N}|\}$ to denote time points[1].
- The horizon is divided into periods $n \in \mathbf{N}' = \mathbf{N} \backslash \{0\} = \{1,2, \dots, |\mathbf{N}|\}$ of equal length $\delta$; period $n$ runs between points $n-1$ and $n$.
- The natural choice of $\delta$ is the greatest common factor of time-related data (processing, $\tau_i$, changeover, $\sigma_{ii'}$, release/due, $\rho_i/\varepsilon_i$, times).
- The data for the model are then obtained by dividing the original parameters by $\delta$, and, if a coarse discretization is used, rounding:

  $\bar{\tau}_i = {}^{\tau_i}/_{\delta}, \bar{\sigma}_{ii'} = {}^{\sigma_{ii'}}/_{\delta}, \bar{\rho}_i = {}^{\rho_i}/_{\delta}, \bar{\varepsilon}_i = {}^{\varepsilon_i}/_{\delta}$

  $\bar{\tau}_i = \lceil {}^{\tau_i}/_{\delta} \rceil, \bar{\sigma}_{ii'} = \lceil {}^{\sigma_{ii'}}/_{\delta} \rceil, \bar{\rho}_i = \lceil {}^{\rho_i}/_{\delta} \rceil, \bar{\varepsilon}_i = \lfloor {}^{\varepsilon_i}/_{\delta} \rfloor$

  (Can you see why some parameters are rounded up and some are rounded down?)

## Equations

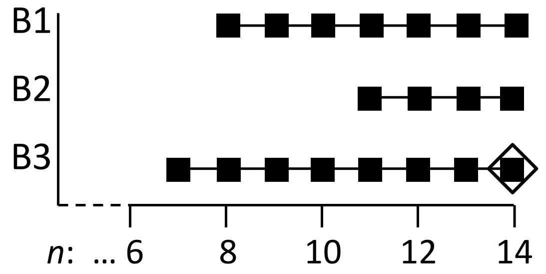
- We introduce binary variable $X_{in}$ which is equal to 1 if batch $i$ starts at time point $n$.
- The first constraint that should be enforced is that each batch is executed once,

  $\sum_n X_{in} = 1, \quad i$

- Next, we should enforce the *no-overlap* restriction. How can we do it?
- What do you think about this: $\sum_i X_{in} \leq 1, \quad n$ ?
- Can you think of another one?

  $\sum_i \sum_{n'=n-\bar{\tau}_i+1}^{n'=n} X_{in'} \leq 1, \quad n$ (1) [Clique constraint]

| $i$ | $\tau_{ij}$ |
|---|---|
| B1 | 7 |
| B2 | 4 |
| B3 | 8 |




[1] Note that we use index $t \in \mathbf{T}$ to denote points/periods in continuous time models but index $n \in \mathbf{N}$ in discrete time models.

## Equations (continued)

- Release times and deadlines are enforced via fixing the binaries outside the allowable window to zero,

  $X_{in} = 0, \quad i, n < \bar{\rho}_i, n > \bar{\varepsilon}_i - \bar{\tau}_i$

- In general, the timing of an event can be modeled through the multiplication of time points by the corresponding binary variable.

  $S_i = \sum_n nX_{in}\,, \quad E_i = \sum_n (n + \bar{\tau}_i)X_{in}$

- Using this idea, we can enforce constraints for makespan,

  $MS \geq \sum_n (n + \bar{\tau}_i)X_{in}\,, \quad i$

- We can also calculate batch earliness, $\bar{\varepsilon}_i - \sum_n (n + \bar{\tau}_i)X_{in}$ ; and lateness, $\sum_n (n + \bar{\tau}_i)X_{in} - \bar{\varepsilon}_i$.
- The corresponding objective functions become,

  $\min MS$

  $\min \sum_i \omega_i(\bar{\varepsilon}_i - \sum_n (n + \bar{\tau}_i)X_{in})$

  $\min \sum_i \omega_i(\sum_n (n + \bar{\tau}_i)X_{in} - \bar{\varepsilon}_i)$

- For tardiness minimization, we first define and constrain a non-negative batch-specific tardiness variable ($L_i \geq 0$),

  $L_i \geq \sum_n (n + \bar{\tau}_i)X_{in} - \bar{\varepsilon}_i\,, \quad i$

  and then express:

  $\min \sum_i \omega_i L_i$

# Single Unit Problem: Product Families

PRINCETON UNIVERSITY

- Products are often grouped into *product families* $f \in \mathbf{F}$ to generate computationally tractable optimization models.
- The grouping can be based on various criteria; e.g., product similarities, processing similarities, or changeover considerations.
- Since the number of batches of each product is known, batches are assigned to families: the subset of batches of family $f$ is $\mathbf{I}_f$.
- Transitions between batches in different families (from $i \in \mathbf{I}_f$ to $i' \in \mathbf{I}_{f'}$, $f \neq f'$) incur sequence-dependent changeover time, $\sigma_{ff'}$ and cost, $\gamma_{ff'}^{CH}$; transitions between batches within a family have no changeover or sequence-independent changeovers (setup).
- One approach is to use a model that accounts for changeover times/costs and simply update changeover parameters as follows.
  1) Products within a family (i.e., $i \in \mathbf{I}_f$ and $i' \in \mathbf{I}_f$ for some $f$):
     - If there is no changeover time/cost, then $\sigma_{ii'} = \gamma_{ii'}^{CH} = 0$.
     - If there is setup time, we can still use a model for changeover times but use: $\sigma_{i'i} = \sigma_i, \forall i' \in \mathbf{I}_f$, where $\sigma_i$ is setup time of $i \in \mathbf{I}_f$. If $\sigma_i < \sigma_{f'f}, \forall f, f', i \in \mathbf{I}_f$, then an alternative approach is to use adjusted processing times $\tau_i^S = \tau_i + \sigma_i$, zero changeovers between batches of the same family, and adjusted changeovers between batches of different families, $\sigma_{ii'}^S = \sigma_{ii'} - \sigma_{i'}$. (Do you see any other adjustments that have to be made? How would you make them?)
  2) Products in different families (i.e., $i \in \mathbf{I}_f$ and $i' \in \mathbf{I}_{f'}$, with $f \neq f'$):
     - Sequence-dependent time/cost: $\sigma_{ii'} = \sigma_{ff'}\, \gamma_{ii'}^{CH} = \gamma_{ff'}^{CH}$.
- The treatment of families with adjusted parameters can be effective when immediate-sequence models are employed (why?).
- Other models can also be simplified in additional ways;
  - In sequence-based models, if changeovers between products within a family are zero, then simpler equations can be used (why?).
  - In continuous time grid-based models, variables $Y_{ii't}$ removed for batches $i, i'$ in the same family; and the summations in the equations used for activating $Y_{ii't}$ can be modified to include batches outside the families.

# Single Unit Problem: Prize Collection

- No minimum demand; we maximize profit; we should consider *batch selection decisions*: $Z_i \in \{0,1\} = 1$ if batch $i$ is carried out
- The objective function, $PC$, then becomes (where $\pi_i$ is the *prize* for carrying out batch $i$): $\max \sum_i \pi_i Z_i$

## Global sequence models

- Variables $Y_{ii'}$ can be equal to 1 only if both batches $i$ and $i'$ are selected:

$$Y_{ii'} \leq Z_i, \quad i, i' \neq i \; (1); \quad Y_{i'i} \leq Z_i, \quad i, i' \neq i \qquad (2)$$

- If two batches are selected, then a sequence should be established

$$Y_{ii'} + Y_{i'i} \geq Z_i + Z_{i'} - 1, \quad i, i' > i$$

- If a batch is not selected, its start and end date are set to zero and the release and deadline constraints are relaxed,

$$S_i \leq M Z_i, \quad i \qquad (3); \quad S_i \geq \rho_i Z_i, \quad i \qquad (4) \quad S_i + \tau_i Z_i \leq \varepsilon_i Z_i, \quad i \qquad (5)$$

## Continuous time grid models

- Introduce $\hat{Z}_t \in \{0,1\}$, which is 1 if a batch is allocated to period $t$.
- Since the number of the selected batches is an optimization decision, $|\mathbf{I}|$ time periods should be postulated; we add (why?)

$$\hat{Z}_t \leq \hat{Z}_{t-1}, \quad t > 1$$

- Batch $i$ is allocated to a period only if it is selected and only active periods can be allocated a batch,

$$\sum_t X_{it} = Z_i, \quad i \qquad (6); \quad \sum_i X_{it} = \hat{Z}_t, \quad t \qquad (7)$$

## Discrete-time grid models

- What changes are needed? Check model in previous slide.

# Single Unit Problem: Exercises

PRINCETON UNIVERSITY

Consider a single unit environment consisting of four batches $\mathbf{I} = \{B1, B2, B3, B4\}$ with processing times, release and due times, and changeover times and costs given in the following table.

Table. Processing ($\tau_i$), release ($\rho_i$) and due ($\varepsilon_i$) times; and changeover ($\sigma_{ii'}$) times and costs ($\gamma_{ii'}$).

| Batch | $\tau_i$ | $\rho_i/\varepsilon_i$ | $\sigma_{ii'}/\gamma_{ii'}$ (=left, =top) | | | |
|---|---|---|---|---|---|---|
| | | | B1 | B2 | B3 | B4 |
| B1 | 2 | 0/15 | - | 1/1 | 2/1 | 1/1 |
| B2 | 4 | 6/15 | 1/4 | - | 1/2 | 1/2 |
| B3 | 3 | 5/20 | 1/1 | 2/8 | - | 1/1 |
| B4 | 5 | 2/15 | 1/1 | 3/1 | 1/1 | - |

Using the above data, formulate and solve the following problems using a sequencing model:

a) Makespan minimization without changeovers.
b) Makespan minimization accounting for changeover times; compare with (a) above.
c) Earliness minimization with weight factor $\omega_i$= {4,5,1,10}.
d) Makespan using the immediate sequencing model and compare with (i).
e) Changeover cost minimization.

# Outline

PRINCETON UNIVERSITY

- Basics
- Single-unit problems
- Single-stage problems
- Multi-stage problems
- Network problems

# Single Stage (Multi-Unit) Problem: Sequence-Based Models

PRINCETON UNIVERSITY

## Problem Statement

- The facility has a set of units $\mathbf{J}$
- Demand is converted into a set of batches $\mathbf{I}$; each batch $i \in \mathbf{I}$ has a release, $\rho_i$, and due, $\varepsilon_i$, time, and has to be carried out in exactly one *compatible* unit $j \in \mathbf{J}_i \subseteq \mathbf{J}$; the set of batches that can be processed on unit $j$ is denoted by $\mathbf{I}_j$.
- The processing time of batch $i$ on unit $j$ is $\tau_{ij}$ and the processing cost is $\gamma_{ij}^{P}$; the changeover cost/time is denoted by $\gamma_{ii'j}^{CH}/\sigma_{ii'j}$.

## Equations

- The new type of decision is the assignment of batches to units. Regardless of the modeling approach, we introduce: $X_{ij} \in \{0,1\}$: is equal to 1 if batch $i$ is assigned to unit $j \in \mathbf{J}_i$.
- Batch-unit assignment:

$$\sum_{j \in \mathbf{J}_i} X_{ij} = 1, \quad i$$

- If two batches are assigned to the same unit, then the relative order in which they are processed should be determined: $Y_{ii'j} \in \{0,1\}$: is equal to 1 if both batches $i$ and $i'$ are assigned to unit $j$, and batch $i$ is processed before batch $i'$.
- In global sequence models, the *activation* of the sequencing binary variables can be achieved via,

$$Y_{ii'j} + Y_{i'ij} \geq X_{ij} + X_{i'j} - 1, \quad i, i' > i, j \in \mathbf{J}_i \cap \mathbf{J}_{i'}$$

- Exercise: Develop equations to activate $Y_{ii'j}$ in immediate sequence models
- If feasible assignments ($X_{ij}$) and sequencing relationships ($Y_{ii'j}$) are available, then all remaining problem restrictions and features can be modeled using the same equations for all sequence-based models

# Single Stage Problem: Sequence-Based Models

PRINCETON UNIVERSITY

## Equations (continued)

The main remaining constraints are:

- Start time disaggregation

  $S_i = \sum_j S_{ij}\ ,\ i \qquad S_{ij} \leq MX_{ij},\ i,j$

- Enforcement of no-overlap condition;

  $S_{ij} + \tau_{ij} \leq S_{i'j} + M(1 - Y_{ii'j}),\ \ i,i',j \in \mathbf{J}_i \cap \mathbf{J}_{i'}$

- Release, due time constraints;

  $S_i \geq \rho_i,\ \ i;\ S_i + \sum_j \tau_{ij} X_{ij} \leq \varepsilon_i + L_i,\ \ i$

  where $L_i$ is batch lateness/tardiness ($L_i = 0$ if hard deadlines).

- Objective functions:
  - Makespan minimization

    $\min MS \quad$ with $\quad MS \geq S_i + \sum_j \tau_{ij} X_{ij}\ ,\ i$

  - Weighted earliness and lateness/tardiness minimization

    $\min \sum_i \omega_i (\varepsilon_i - (S_i + \sum_j \tau_{ij} X_{ij})), \qquad \min \sum_i \omega_i L_i$

  - Cost minimization

    $\min \sum_{i,j} \gamma_{ij}^{P} X_{ij}$

## Changeovers

Changeover times & costs can be handled using techniques similar to the ones presented for single unit problems

- Assuming that the triangular inequality holds, timing constraint:

  $S_{ij} + \tau_{ij} + \sigma_{ii'j} \leq S_{i'j} + M(1 - Y_{ii'j}), \qquad i,i',j \in \mathbf{J}_i \cap \mathbf{J}_{i'}$

- Changeover costs can be addressed using immediate-sequence-based models.

  $\min \sum_{i,i'} \sum_j \gamma_{ii'j}^{CH} Y_{ii'j}$

# Single Stage Problem: Continuous Time Grid-Based Models

PRINCETON UNIVERSITY

## Time Grids

- Time horizon of each unit divided into $T^j$ unit-specific periods
- If unit $j$ becomes available at $\bar{\varrho}_j$ and deadlines are given, then the start, $\varrho_j$, and end, $\varepsilon_j$, of the *unit* horizons are given by:

$$\varrho_j = \max\{\bar{\varrho}_j, \min_{i \in \mathbf{I}_j}\{\rho_i\}\}, \quad \varepsilon_j = \max_{i \in \mathbf{I}_j}\{\varepsilon_i\}$$

- $T^{MAX}$: max number of periods, $T^{MAX} = \max_j\{T^j\}$; we define $\mathbf{T} = \{1, \dots, T^{MAX}\}$ and unit-specific subsets $\mathbf{T}_j \subseteq \mathbf{T}$ of time periods $\mathbf{T}_j = \{1, \dots, T^j\}$.
- $T_{jt} \geq 0$: timing of point $t$ in the grid of unit $j$; each unit period starts at $T_{j,t-1}$ and ends at $T_{jt}$.

$$T_{j,0} = \varrho_j, \; T_{j,T^j} = \varepsilon_j$$

$$T_{jt} \geq T_{j,t-1}, \quad j, t \in \mathbf{T}_j$$

## Variables

- $X_{ijt} \in \{0,1\}$: equal to 1 if batch $i$ allocated to period $t$ of unit $j$
- $S_{ij} \geq 0$: (disaggregated) start time of batch $i$ on unit $j$
- $S_i \geq 0$: start time of batch $i$

## Equations

- Assignment of batches to units and time periods:

$$\sum_t X_{ijt} = X_{ij}, \quad i, j$$

$$\sum_{j,t} X_{ijt} = 1, \quad i$$

$$\sum_i X_{ijt} \leq 1, \quad j, t \in \mathbf{T}_j$$

- Start times:

$$S_i = \sum_j S_{ij}, \; i; \qquad S_{ij} \leq M \sum_t X_{ijt}, \; i, j$$

- Timing constraints:

$$S_{ij} \geq T_{j,t-1} - \eta\left(1 - X_{ijt}\right), \quad i, j, t \in \mathbf{T}_j$$

$$S_{ij} + \tau_{ij} \leq T_{jt} + \eta\left(1 - X_{ijt}\right), \quad i, j, t \in \mathbf{T}_j$$

- Release and due times:

$$S_i + \sum_{j,t} \tau_{ij} X_{ijt} \leq \varepsilon_i, \quad i; \; S_i + \sum_{j,t} \tau_{ij} X_{ijt} \leq \varepsilon_i + L_i, \quad i$$

- Objective functions:
  - Same as sequence-based for makespan, earliness, lateness
  - Cost minimization: $\min \sum_{i,j,t} \gamma_{ij}^P X_{ijt}$

# Single Stage Problem: Discrete Time Grid-Based Models

PRINCETON UNIVERSITY

## Time Grid

- Time points $n \in \mathbf{N} = \{0,1, \dots |\mathbf{N}|\}$; periods $n \in \mathbf{N}' = \mathbf{N} \backslash \{0\}$ of length $\delta$; period $n$ runs between points $n-1$ and $n$.
- Rounding time-related data as in single unit problem
- Unit grid: $\mathbf{N}_j = \{n | \max\{\varrho_j, \min_{i \in \mathbf{I}_j}\{\bar{\rho}_i\}\} \leq n \leq \max_{i \in \mathbf{I}_j}\{\bar{\varepsilon}_i\}\}$

## Equations

- All decisions encompassed in single assignment variable: $X_{ijn} \in \{0,1\}$: equal to 1 if batch $i$ starts on unit $j$ at time point $n$
- Each batch is executed once:

  $\sum_{j,n} X_{ijn} = 1, \quad i$

- Processing of at most one batch at a time:

  $\sum_i \sum_{n'=n-\bar{\tau}_{ij}+1}^{n'=n} X_{ijn'} \leq 1, \quad j,n$

- Release and due times:

  $X_{ijn} = 0, \quad i,j,n < \bar{\rho}_i, n > \bar{\varepsilon}_i - \bar{\tau}_{ij}$

- Do we need another equation?
- How many equations do we have?

## Objective Functions

- Makespan minimization

  min $MS$ with $MS \geq \sum_{j,n}(n + \bar{\tau}_{ij})X_{ijn}, \quad i$

- Weighted earliness:

  $\min \sum_i \omega_i(\bar{\varepsilon}_i - \sum_{j,n}(n + \bar{\tau}_{ij})X_{ijn})$

- Weighted lateness

  $\min \sum_i \omega_i(\sum_{j,n}(n + \bar{\tau}_{ij})X_{ijn} - \bar{\varepsilon}_i)$

- Weighted tardiness $(L_i \in \mathbb{R}_+)$:

  $\min \sum_i \omega_i L_i$ subject to $L_i \geq \sum_{j,n}(n + \bar{\tau}_{ij})X_{ijn} - \bar{\varepsilon}_i, \quad i$

- Cost minimization:

  $\min \sum_{i,j}(\gamma_{ij}^P \sum_n X_{ijn})$

# Single Stage Problem: Batching Decisions



## Problem Statement

- If units have different capacities, $\beta_j$, then the number of required batches to meet given demand (orders) is not known a priori
- Given are a set of orders $\mathbf{I}$ and a set of units $\mathbf{J}$; each order has amount $\xi_i$ due, release/due time $\rho_i/\varepsilon_i$; the batches towards it can be executed in $j \in \mathbf{J}_i \subseteq \mathbf{J}$; the processing time for a batch of order $i$ in unit $j$ is $\tau_{ij}$ and the processing cost is $\gamma_{ij}^P$.

## Sequence-Based Model

- Minimum and maximum number of batches for order $i$:

  $\lambda_i^{MIN} = \left\lceil \xi_i / \max_{j \in \mathbf{J}_i} \{\beta_j\} \right\rceil, \lambda_i^{MAX} = \left\lceil \xi_i / \min_{j \in \mathbf{J}_i} \{\beta_j\} \right\rceil, i$ **(Why?)**
- Set of potential batches for order $i$, $l \in \mathbf{L}_i = \{1, 2, \dots, \lambda_i^{MAX}\}$
- Batch selection: $Z_{il} \in \{0,1\}$
- If batch is selected, it should be assigned:

  $\sum_{j \in \mathbf{J}_i} X_{ilj} = Z_{il}, \quad i, l \in \mathbf{L}_i$
- Demand satisfaction

  $\sum_{l \in \mathbf{L}_i} \sum_j \beta_j X_{ilj} \geq \xi_i, \quad i$
- Remaining constraints similar to the ones for the fixed-batch problem: (1) variables defined for $i$ and $l \in \mathbf{L}_i$ instead of $i$ only; and (2) constraints are expressed for $i$ and $l \in \mathbf{L}_i$ instead of $i$ only; for example

  $Y_{ili'l'j} + Y_{i'l'ilj} \geq X_{ilj} + X_{i'l'j} - 1, \quad i, i' > i, l \in \mathbf{L}_i, l' \in \mathbf{L}_{i'}, j \in \mathbf{J}_i \cap \mathbf{J}_{i'}$

## Discrete-time Model

- Since sequencing is achieved via the mapping of the starting times onto the grid, there is no need to differentiate between batches of the same product
- No need to calculate $\lambda_i^{MIN}$ and $\lambda_i^{MAX}$
- Batch selection/activation:

  $\sum_{j \in \mathbf{J}_i, n} \beta_j X_{ijn} \geq \xi_i, \quad i$

  (instead of $\sum_{j,n} X_{ijn} = 1, \quad i$)
- Everything else remains the same, that is,

  $\sum_i \sum_{n'=n-\bar{\tau}_{ij}+1}^{n'=n} X_{ijn'} \leq 1, \quad j, n$

# Shared Resources

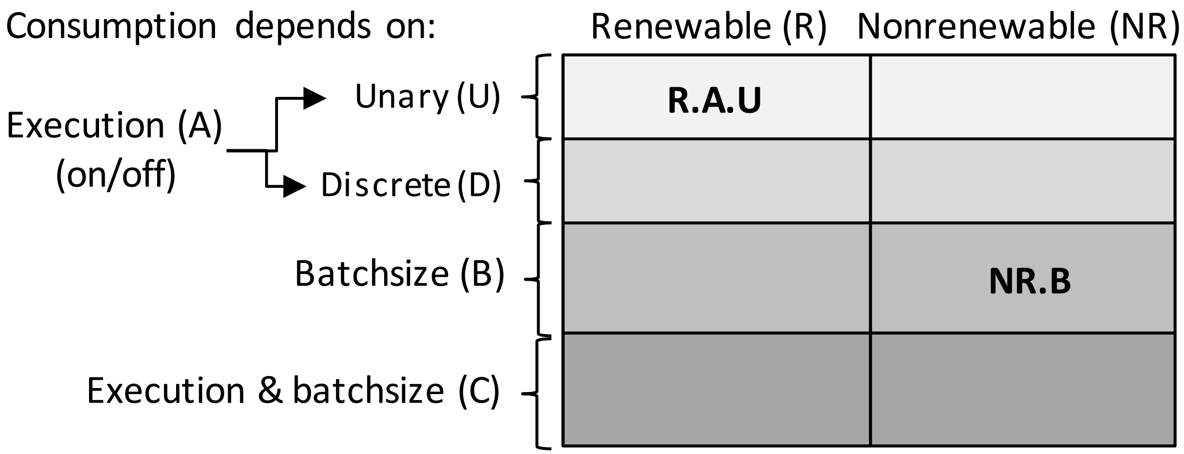
## Preliminaries

- So far, we have assumed that the only shared resources type is units and that each batch requires only one type of shared resource.
- This is not always true: a batch may require, simultaneously, multiple types of resources; e.g., labor, electricity, and cooling water.
- In general, shared resources can be classified as renewable and non-renewable
- Renewable resources are *used* during the execution of a batch and *freed* after a batch is finished; e.g, in the case of cooling (power), a certain load (which is subtracted from the available cooling capacity of the facility) is necessary during the execution of a batch.
- Non-renewable resources are *consumed* by a batch, that is, they do not become available after the batch is finished; e.g., a promoter, necessary for the initiation of a polymerization batch, can be viewed as a non-renewable resource.
- For now, we will focus on renewable resources (and, without loss of generality, problems with no changeovers)
- At a second level, resources can be classified as discrete (e.g., labor) and continuous (e.g., cooling load).
- A special discrete resource is a *unary* resource; i.e., a resource for which the demand is 1 unit and its capacity is also 1; an equipment unit can be viewed as a unary resource.

## Classification

We classify resources based on the nature of the demand for them:

- Type A: based on the execution (on/off) of a batch
  - Unary (e.g., unit, labor)
  - Discrete (e.g., pumps needed to load/withdraw material)
- Type B: based on the size (continuous variable) of a batch
- Type C: based on both the execution and batch size.



# Shared Resources

PRINCETON UNIVERSITY

## Formulation Basics

- If $\varphi_{im}$/ $\psi_{im}$ is the fixed/variable requirement of resource $m$ by batch $i$, then the resource use, $R_{im}$, during the execution of a batch is

  $R_{im} = \varphi_{im} + \psi_{im} B_i$

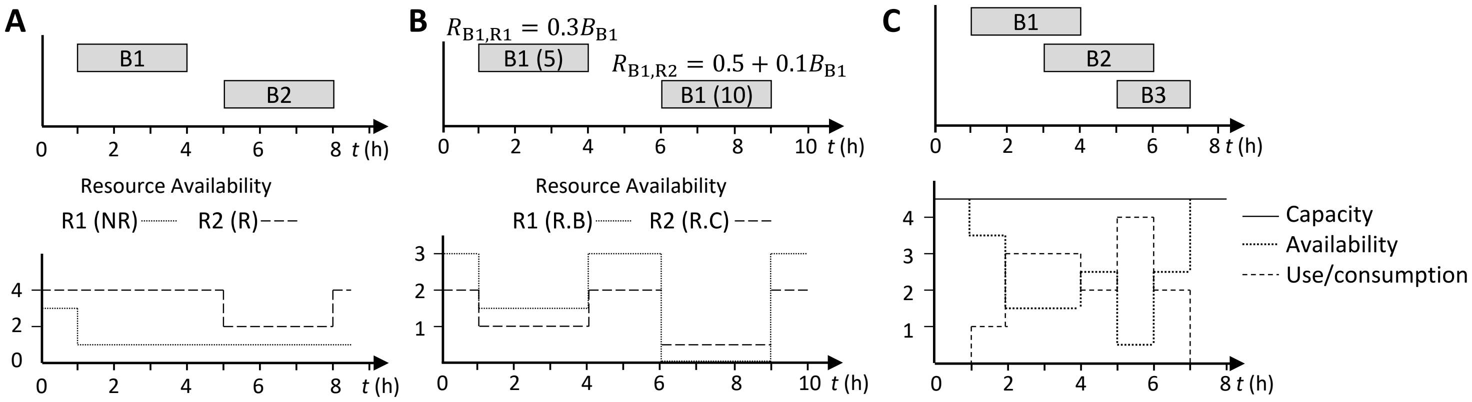
- Note that for type A resources, $\psi_{im} = 0$; for type B, $\varphi_{im} = 0$; and both parameters are positive for type C resources
- A resource can be classified differently based on the batch that requires it.
  - A batch processed in small units may always require 1 pump for loading/unloading ⇒ a pump can be viewed as a unary resource
  - A different batch, produced in larger units with variable batchsizes, may require many pumps (integer function of its batchsize) ⇒ pumps should be treated collectively as a discrete resource.
- Thus, it is more accurate to say that a resource-batch pair is classified according to the proposed scheme

## Example

We illustrate: (A) (non)renewable resources, (B) resource types A-C; and (C) the relationship between resource consumption and availability.

| | Instance 1 | | Instance 2 | | Instance 3 |
|---|---|---|---|---|---|
| $m$ | R1 (NR.A)[1] | R2 (R.A) | R1 (R.B) | R2 (R.C) | R1 (R.A) |
| $\chi_m$ | 3 | 4 | 3 | 2 | 4.5 |
| | $\varphi^S_{im}$ | $\varphi_{im}$ | $\psi_{im}$ | $\varphi_{im}/\psi_{im}$ | $\varphi_{im}$ |
| B1 | 2 | | 0.3 | 0.5/0.1 | 1 |
| B2 | | 2 | | | 2 |
| B3 | | | | | 2 |

[1] Parameter $\varphi^S_{im}$ is used, instead of $\varphi_{im}$, because consumption of renewable resource, at the start of the batch, is permanent.



# Shared Utilities: Sequence and Continuous Time Grid Models

## Sequence Based

- A key attribute of single-stage problems is that batches assigned to different units are not interacting with each other, i.e., batch sequencing ($Y_{ii'j}$) is unit-specific
- If batches assigned to different units require the same resource, then monitoring resource consumption across units is needed; i.e., the parallel units are *coupled*.
- Sequence-based models are not well suited to address problems under shared resources, unless they are unary (e.g., labor)
  - Each unary resource $m \in \mathbf{M}$ is modelled as a unit.
  - Resource-specific grids are defined
  - Variables $Y_{ii'm} \in \{0,1\}$ and $S_{im} \in \mathbb{R}_+$ are introduced
  - All sets of constraints are expressed for all unary resources

## Continuous Time Grid

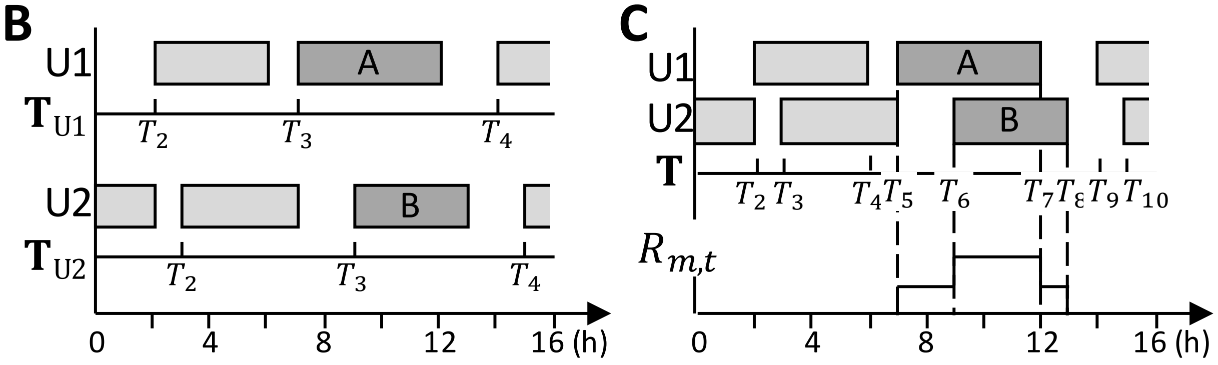
- The special case of unary resources can be treated similarly:
  - Define resource-unit-specific grids;
  - Assign batches to resource unit slots;
  - Disaggregate start time variables with respect to all resources; (note: all disaggregated $S_{im}$ must be equal).
  - Enforce that batches are carried out within resource unit slots.
- The general case cannot be readily handled by models based on unit-specific grids (see Figure)
- For the general case, a single common time grid is necessary
  - A batch spans multiple time periods;
  - The ending time point cannot be inferred from the starting one
  - Advantage of continuous time not present

**A**

Batch A starts on U1 at $t = 3$, $T_3 = 7$.
Batch B starts on U2 at $t = 3$, but $T_3 = 9$.

Batch A requires 2 units of resource m
Batch B requires 2 units of resource m

What is the resource consumption at $T_3$?
$T_3$ is not uniquely defined.




A. Description of instance.
B. Solution representation using unit-specific grids; $T_3$ corresponds to $t = 7$ in U1 but $t = 9$ in U2.
C. Representation of the same solution using a common time grid; the start of A and B now correspond to different time points

# Shared Utilities: Discrete Time Models

## Basics

- All types of resources can be readily addressed; even with $B_{ijn}$
- Same common time grid is used; no new assignment variables

## Resource Variables

- Resource utilization $R_{im} = \varphi_{im} + \psi_{im} B_i$
- Resource *engagement* and *release*:

$$R^{I}_{imn} = \varphi_{im} \sum_j X_{ijn} + \psi_{im} \sum_j B_{ijn}, \quad i, m, n \qquad (1)$$

$$R^{O}_{imn} = \varphi_{im} \sum_j X_{ij,n-\bar{\tau}_{ij}} + \psi_{im} \sum_j B_{ij,n-\bar{\tau}_{ij}}, \quad i, m, n \qquad (2)$$

- Plugged into (i.e., eqs (1)-(3) can be written as one constraint):

$$R_{mn} = R_{m,n-1} + \sum_i R^{I}_{im,n-1} - \sum_i R^{O}_{im,n-1} \leq \chi_m, \quad m, n \qquad (3)$$

- Objective function

$$\min \sum_{i,j} (\gamma^{P}_{ij} \sum_n X_{ijn}) + \sum_{i,i',j,n} \sigma_{ii'j} Z_{ii'jn} + \sum_{m,n} \gamma^{RES}_{m} \delta R_{mn}$$

## Alternative Approach

- Bound resource consumption:

$$\sum_{i,j} \sum_{n'=n-\bar{\tau}_{ij}}^{n'=n-1} (\varphi_{im} X_{ijn'} + \psi_{im} B_{ijn'}) \leq \chi_m, \quad m, n$$

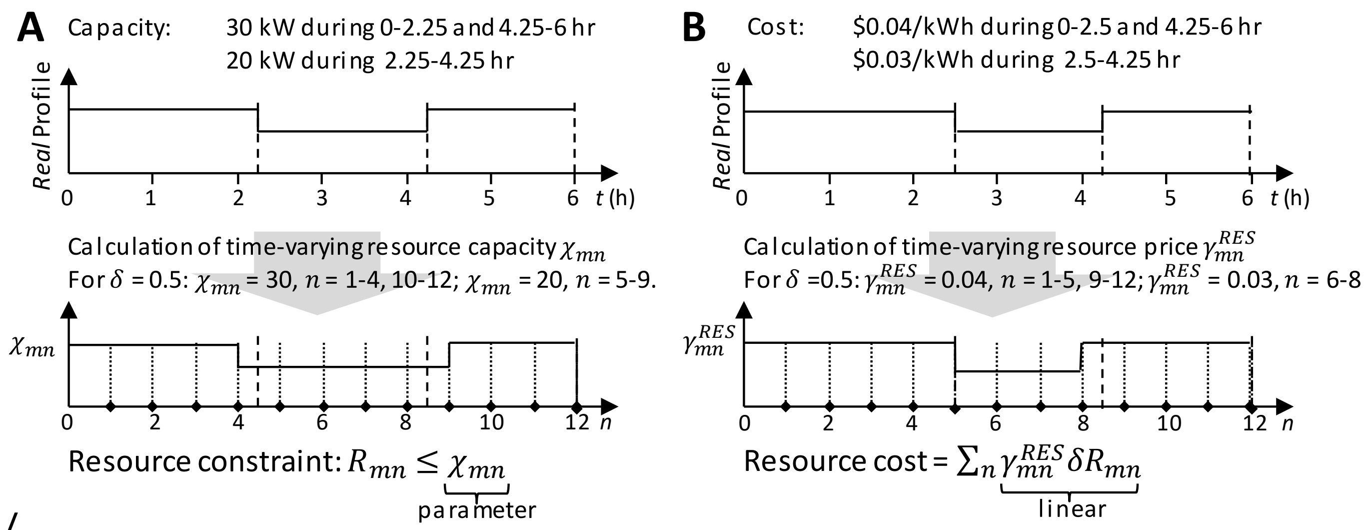
- Have you seen a similar constraint before?
- What happens if $\varphi_{im} = 1, \psi_{im} = 0$, and $\chi_m = 1$

# Shared Utilities: Extensions Using Discrete Time Models

PRINCETON UNIVERSITY

## Time-Varying Resource Capacity and Cost

- All variables and constraints remain **exactly** the same
- Need to only calculate time-varying parameters: capacity, $\chi_{mn}$, and cost $\gamma_m^{RES}$



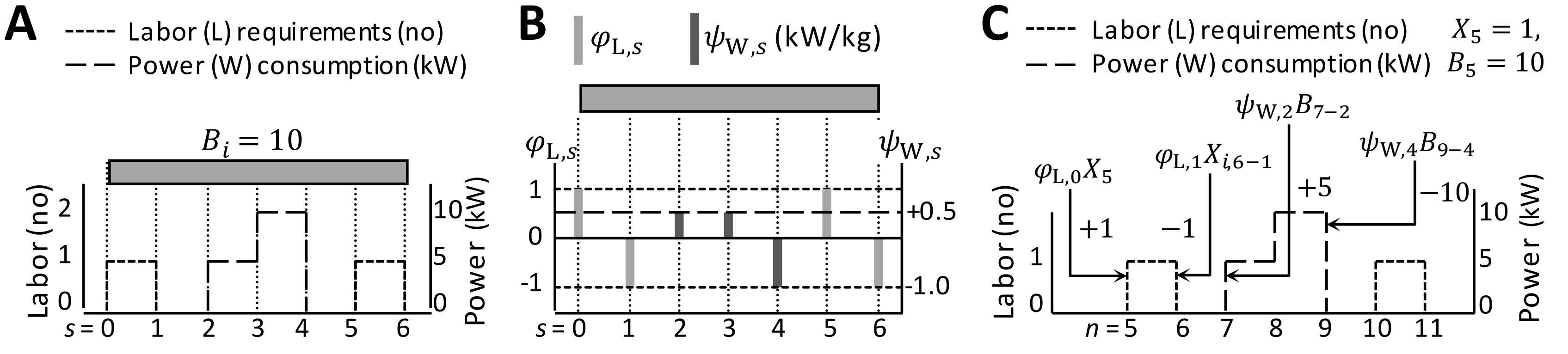

Modeling of time-varying resource capacity and cost using a discrete time model with δ = 0.5 h. Calculation of parameters $\chi_{mn}$ (A) and $\gamma_{mn}^{RES}$ (B) from real data.

## Varying Consumption During Batch Execution

- Define $\varphi_{ims}/\psi_{ims}$ to denote fixed/proportional engagement/ release of resource $m$, $s$ periods after the start of batch $i$
- Replace (3) with:

$$R_{imn}^{NET} = \sum_j \sum_{s=0}^{\bar{\tau}_i} (\varphi_{ims} X_{ij,n-s} + \psi_{ims} X_{ij,n-s}), \quad i, m, n$$

$$R_{mn} = R_{m,n-1} + \sum_i R_{im,n-1}^{NET} \leq \chi_{mn}, \quad m, n$$




Varying resource consumption during batch

A. Resource consumption during batch with batchsize of 10 kg.
B. Resource consumption parameters.
C. Resource consumption calculations for batch starting at $n = 5$. (index $i$ omited from B and C).

# Outline

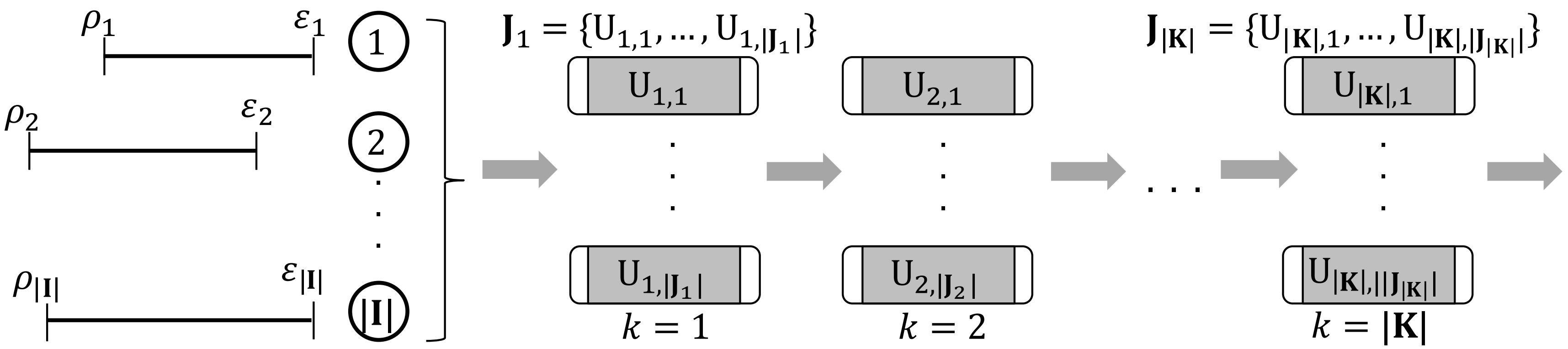
- ❑ Basics
- ❑ Single-unit problems
- ❑ Single-stage problems
- ❑ Multi-stage problems
- ❑ Network problems

# Multi-stage Problems

PRINCETON UNIVERSITY

## Problem Statement

- We consider the problem with fixed batching decisions.
- The facility has $k \in \mathbf{K}$ stages; each stage has units $j \in \mathbf{J}_k$ with $\cup_k \mathbf{J}_k = \mathbf{J}$ and $\mathbf{J}_k \cap \mathbf{J}_{k'} = \emptyset$ for all $k, k'$; i.e., each unit belongs to one stage
- We are given a set, $\mathbf{I}$, of batches that have to be processed on exactly one unit per stage
  - Each batch $i$ has a release, $\rho_i$, and due, $\varepsilon_i$, time
  - The processing time of batch $i$ on unit $j$ is denoted by $\tau_{ij}$ and the processing cost by $\gamma_{ij}^P$.
- The processing of a batch in stage $k+1$ can start only after its processing in $k$ is completed (*precedence relation* or *constraint*)
- $\mathbf{J}_{ik}$ is the subset of units in stage $k$ suitable for processing batch $i$; $\mathbf{I}_j$ is the set of batches that can be carried out in unit $j$.




## Remarks

- The problem can be viewed as many single-stage problems ***coupled*** together:
- The environment leads naturally to (intermediate ) storage considerations → ***Storage Policies***

# Multi-Stage Problems: *Coupled* Models

PRINCETON UNIVERSITY

## Sequence-Based Model

Each batch has to go through consecutive one stage problems

- Assigned to a unit:

  $\sum_{j \in \mathbf{J}_{i,k}} X_{ij} = 1, \quad i, k$

- Sequenced:

  $Y_{ii'j} + Y_{i'ij} \geq X_{ij} + X_{i'j} - 1, \quad i, i' > i, k, j \in \mathbf{J}_{ik} \cap \mathbf{J}_{i'k}$

  $S_{ij} + \tau_{ij} \leq S_{i'j} + M\left(1 - Y_{ii'j}\right), \quad i, i', k, j \in \mathbf{J}_{ik} \cap \mathbf{J}_{i'k}$

- Start on unit:

  $S_{ik} = \sum_{j \in \mathbf{J}_{ik}} S_{ij}, \quad i, k \quad \text{st.}\ S_{ij} \leq M X_{ij}, \quad i, k, j \in \mathbf{J}_{ik}$

- Meet release and due times:

  $S_{i,k=1} = S_{i,1} \geq \rho_i, \quad i; \quad S_{i,|\mathbf{K}|} + \sum_{j \in \mathbf{J}_{i,|\mathbf{K}|}} \tau_{ij} X_{ij} \leq \varepsilon_i, \quad i$

- **Satisfy precedence relationship**

  $S_{ik} + \sum_{j \in \mathbf{J}_{ik}} \tau_{ij} X_{ij} \leq S_{i,k+1}, \quad i, k < |\mathbf{K}|$

## Discrete Time Grid Model

Each batch has to go through consecutive one stage problems

- Assigned to a unit:

  $\sum_{j \in \mathbf{J}_{ik}} \sum_n X_{ijn} = 1, \quad i, k$

- Sequenced and timed:

  $\sum_i \sum_{n'=n-\bar{\tau}_{ij}+1}^{n'=n} X_{ijn'} \leq 1, \quad j, n$

- Meet release and due times:

  $X_{ijn} = 0, \qquad i, j, n < \bar{\rho}_i; \quad X_{ijn} = 0, \qquad i, j, n > \bar{\varepsilon}_i - \bar{\tau}_{ij}$

- **Satisfy precedence relationship**

  $\sum_{j \in \mathbf{J}_{ik}} \sum_n n X_{ijn} \geq \sum_{j \in \mathbf{J}_{i,k-1}} \sum_n \left(n + \bar{\tau}_{ij}\right) X_{ijn}, \quad i, k > 1$

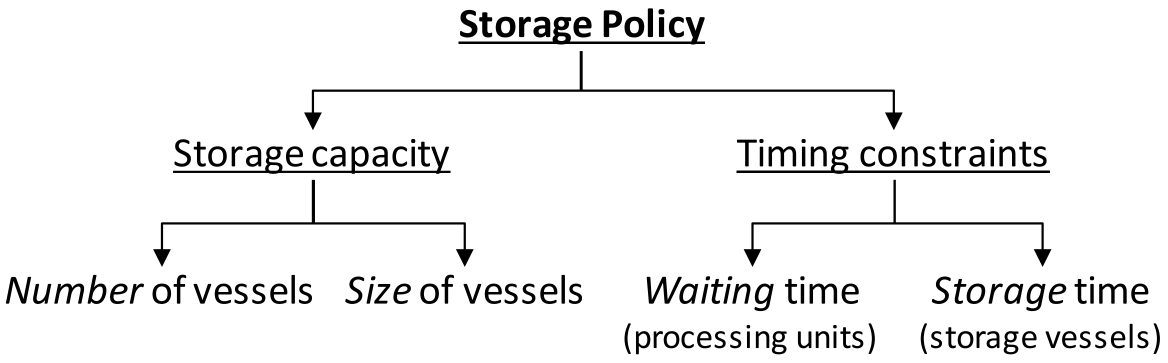
# Multi-Stage Environments: Storage

## Preliminaries

- Chemical manufacturing involves handling of fluids which leads to different types of *storage policies*.
- Most research has focused on storage in network environments, but the no splitting/mixing restrictions lead to complex constraints.
- In sequential facilities, batches of the same product cannot be mixed so the number and size of available vessels are relevant
- If enough vessels of sufficient size are available, then we operating under *unlimited* storage; otherwise, we are under *limited* storage
- The time (the material from) a batch can be stored in a vessel (*storage time*) is also important (e.g., food manufacturing)
- The time a batch can wait in a processing unit after its completion (*waiting time*) is also relevant; waiting in a processing unit can be used instead of actual storage in a vessel if the next processing stage is the bottleneck of the system.

## Main Idea

- A storage *policy* is based on both *capacity* and *timing* constraints.
  - The former depend on both the *number* and *size* of storage vessels
  - The latter depend on the *waiting* and *storage* times
- Note 1: We use the term *size* to describe the capacity of an individual storage vessel; and the term *capacity* to describe a feature of the entire process (depending on both the number and size of all vessels.



# Multi-Stage Environments: Storage Policy Classification

PRINCETON UNIVERSITY

- The number and size of vessels determine whether we have unlimited (US), limited (LS) or no storage (NS).
- The waiting and storage time constraints determine whether we have unlimited (UT) or limited (LT) storage/waiting time policies.
- Each box in the Figure corresponds to a different *case*, represented by a pair $\beta^C/\beta^T$, where $\beta^C$ and $\beta^T$ refer to storage capacity and timing, respectively, constraints.

1) US/UT: it can be assumed that a batch can always be stored as soon as its processing in stage is completed, and that it can remain stored for unlimited time. This is the problem we have considered so far.
2) US/LT: it is necessary to model waiting and storage times and introduce constraints that bound them.
3) Under limited storage, storage vessels become scarce resources, and hence the assignment and sequencing of competing batches to storage vessels should be considered. We should also account for batch transfer from a processing unit to a storage vessel or a processing unit in the next stage and the waiting/storage times.
4) When no storage is available, there is no need to model batch-vessel assignments; but, unlike the US/UT case, a batch should be directly transferred from one stage to the next with no *idle* time; no-storage automatically implies zero-wait *storage* time, so zero-wait is relevant only for the waiting time in a processing unit.
   We classify zero-wait as a special case of the limited waiting time.

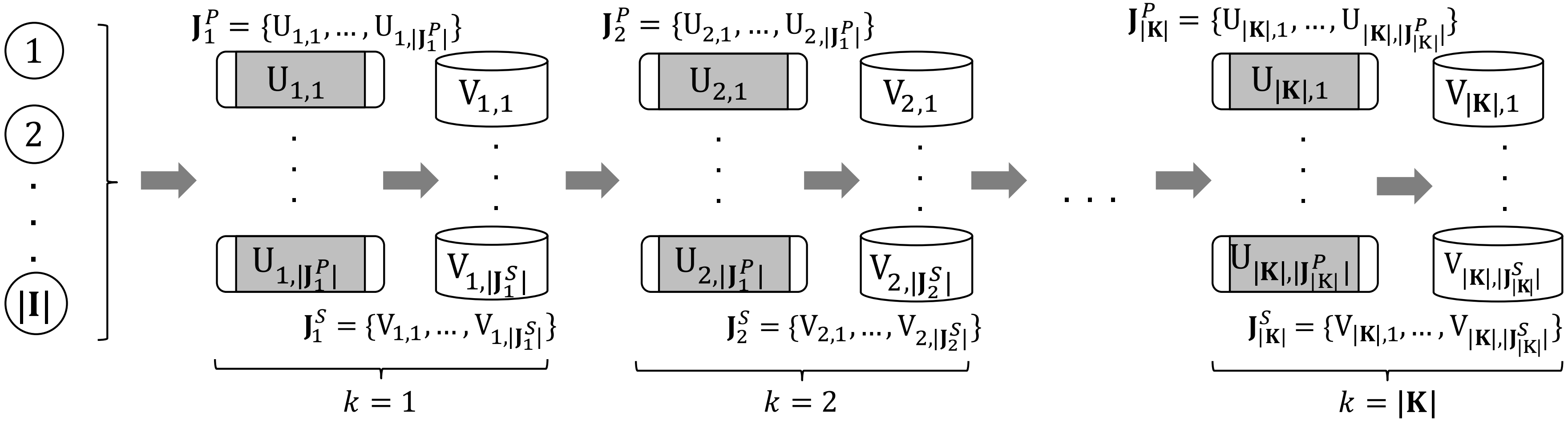
| Timing constraints: waiting and storage time | Storage capacity: number and size of storage vessels | | |
|---|---|---|---|
| | Unlimited (US) | Limited (LS) | No storage (NS) |
| Unlimited (UT) | US/UT<br>*Traditional approaches* | LS/UT<br>• Vessel resource constraints<br>• Modeling of transfers<br>• Modeling of waiting & storage times | NS/UT*<br>• Account for transfers (i.e., enforce storage bypass)<br>• Modeling of waiting times |
| Limited (LT) | US/LT<br>• Modeling & bounding of waiting & storage times | LS/LT<br>• Vessel resource constraints<br>• Account for transfers<br>• Modeling & bounding of waiting & storage times | NS/LT*<br>• Account for transfers (i.e., enforce storage bypass)<br>• Modeling & bounding of waiting times |

* UT and LT refer to waiting time only; since there are no storage vessels, all storage times are zero

# Multi-Stage Environments: Problem Statement



- The facility consists of processing stages $k \in \mathbf{K}$. Each stage has:
  - Processing units $j \in \mathbf{J}_k^P$ with $\cup_k \mathbf{J}_k^P = \mathbf{J}^P$ and $\mathbf{J}_k^P \cap \mathbf{J}_{k'}^P = \emptyset$ for all $k, k'$; and
  - Storage vessels $j \in \mathbf{J}_k^S$, where the output of stage $k$ can be stored, with $\cup_k \mathbf{J}_k^S = \mathbf{J}^S$ and $\mathbf{J}_k^S \cap \mathbf{J}_{k'}^S = \emptyset$ for all $k, k'$
- Each batch $i \in \mathbf{I}$ has to be processed on exactly one compatible unit, $j \in \mathbf{J}_{ik}^P$, in each stage, and can be stored in a compatible vessel, $j \in \mathbf{J}_{ik}^S$, before its processing in stage $k+1$ starts; $\mathbf{I}_j$ is the set of batches that can be carried out in unit $j$.
- Each batch has a release, $\rho_i$, and due, $\varepsilon_i$, time and the processing time of batch $i$ on unit $j$ is denoted by $\tau_{ij}$.




## Remarks

- To express the constraints, we have to account for the batch transfer time from a processing unit to a storage vessel and vice versa.
- Waiting time in processing units can be used in optimal solutions ⇨ waiting and storage times should be modeled and bounded.
- If batching is optimized, the number & size of batches are variables ⇨ the number & size of storage vessels should be considered.
- If batching decisions are known, batchsizes are given, so we can predetermine vessel-batch suitability in each stage; ⇨ it is not necessary to consider batchsizes and vessel sizes.
- We assume unlimited storage for raw materials; if raw material storage is limited, we introduce a dummy $k = 0$ stage with vessels.

# Outline

- ❑ Basics
- ❑ Single-unit problems
- ❑ Single-stage problems
- ❑ Multi-stage problems
- ❑ Network problems

# Network Environments

## Preliminaries

- An environment is classified as *network* when there are no restrictions in the way all materials are handled; i.e., multiple batches of the same task can be mixed or material produced by a single batch can be consumed by multiple batches.
- The notion of a batch going through different stages is irrelevant in network environments (no requirement for batch integrity)
- Processing stages not used because products can be produced in different ways; batching decisions have to be made
- New problem representation is necessary
- We use *tasks*, which are defined in terms of input (consumed) and output (produced) materials (*batch* denotes different executions of the same task, i.e. a schedule may include multiple batches of the same task).
- We use the concept of *materials*, consumed and produced by tasks.
- In the absence of other restrictions (e.g., unit connectivity), materials can flow freely from storage vessels to processing units.

## Example

Facility producing two products from two raw materials through four tasks requiring utilities. Figure B is not a process flow diagram (PFD):

- Batch process: connections represent one-time (instantaneous) transfers [kg], not flows [kg/h]
- Different connections are active at different times; some connections may be inactive
- *Units* are not to *unit operations*: a unit may carry out different tasks at different times

**A**

Basic Elements
Tasks: T1, T2, T3, T4
Processing Units: R1, R2, R3
Storage Units: V1, V2, V3, V4, V5, V6
Materials: RM1, RM2, IN1, IN2, IN3, P1, P2
Utilities: hot steam (HS), cooling water (CW)

Task Conversions

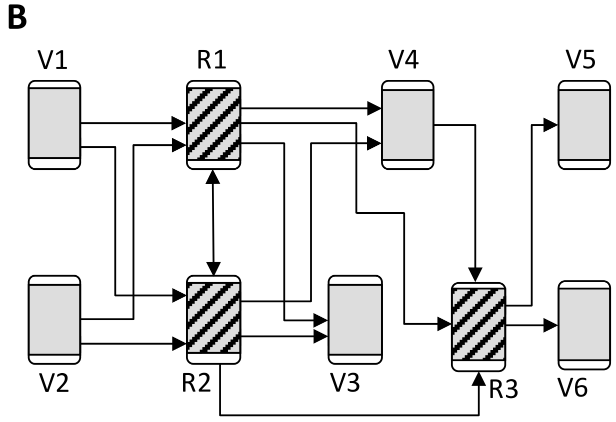
| | |
|---|---|
| T1: | 0.8 RM1 + 0.2 IN1 → IN3 |
| T2: | RM2 → 0.3 IN1 + 0.7 IN2 |
| T3: | IN3 → P1 |
| T4: | 0.6 IN2 + 0.4 IN3 → P2 |

*Task – Unit Suitability*
T1 and T2 carried out on R1 or R2
T3 and T4 carried out on R3

*Material – Vessel Compatibility*

| | |
|---|---|
| RM1 in V1; | RM2 in V2; |
| IN2 in V3; | IN2 and IN3 in V4; |
| P1 in V5; | P2 in V6 |

*Utility Requirements*
T1 and T3 require HS
T2 and T4 require CW




Representation of example network environment. **A**. Basic elements (tasks, processing units, storage vessels, materials, and utilities). **B.** Facility structure.

# Network Environments: Representations

## State Task Network (STN)

The STN representation is based on the following concepts:

- States (materials[1]): represented by circles.
- Tasks: activities consuming/producing materials (rectangles)
- Units: unary resources needed for task execution
- Utilities

The facility is defined in terms of the following:

*Indices/Sets*

$i \in \mathbf{I}$ tasks

$j \in \mathbf{J}$ processing units

$k \in \mathbf{K}$ materials

*Subsets*

$\mathbf{I}_k^+/\mathbf{I}_k^-$ tasks producing/consuming material $k$

$\mathbf{I}_j$ tasks that can be carried out on unit $j$

$\mathbf{J}_i$ processing units that can process task $i$

$\mathbf{K}_i^+/\mathbf{K}_i^-$ materials produced/consumed by task $i$

Parameters:

$\beta_j^{MIN}/\beta_j^{MAX}$ minimum/maximum capacity of unit $j$

$\gamma_{ij}^F/\gamma_{ij}^V$ fixed/variable cost for task $i$ in unit $j$

$\pi_k$ price of material $k$

$\rho_{ik}$ conversion coefficient of material $k$ produced (>0) or consumed (<0) by task $i$

$\mathbf{I} = \{T1, T2, T3, T4\}$

$\mathbf{J} = \{R1, R2, R3\}$

$\mathbf{K} = \{RM1, RM2, IN1, IN2, IN3, P1, P2\}$

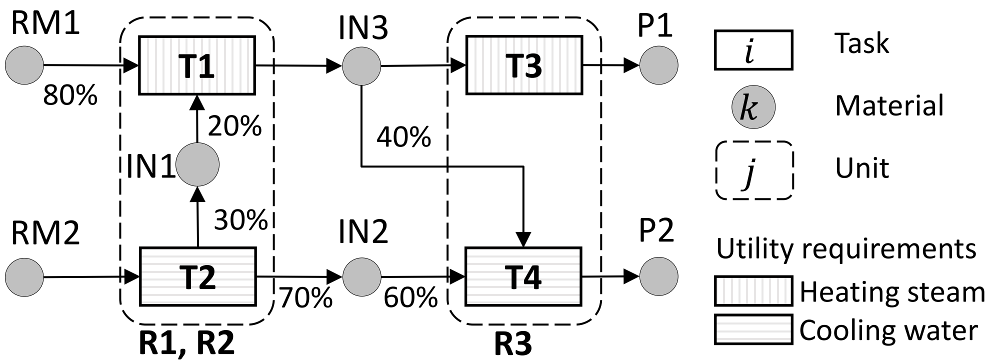
$\mathbf{L} = \{HS, CW\}$

$\mathbf{J}_{T1} = \mathbf{J}_{T2} = \{R1, R2\}$ $\mathbf{J}_{T3} = \mathbf{J}_{T4} = \{R3\}$

$\mathbf{I}_{HS} = \{T1, T3\}$ $\mathbf{I}_{CW} = \{T2, T4\}$

$\mathbf{I}_{IN3}^+ = \{T1\}$, $\mathbf{I}_{IN3}^- = \{T3, T4\}$

$\mathbf{K}_{T1}^- = \{RM1, IN1\}$, $\mathbf{K}_{T1}^+ = \{IN3\}$




[1] We use the term *material*, rather than *state*, because the latter is also used to describe the *system state*, a concept used in real-time scheduling

# Network Environments: Representations

## Resource Task Network (RTN)

- The major difference between STN and RTN representations is that in RTN materials and units are both treated as resources
- Utilities are also, naturally, treated as resources
- The only modeling entities in RTN are tasks and resources

The facility is defined in terms of the following:

*Indices/Sets*

| | |
|---|---|
| $i \in \mathbf{I}$ | tasks |
| $r \in \mathbf{R}$ | resources |

*Subsets*

| | |
|---|---|
| $\mathbf{I}_r^+/\mathbf{I}_r^-$ | tasks producing/consuming resource $r$; |
| $\mathbf{I}_r$ | tasks interacting with resource $r$; $\mathbf{I}_r = \mathbf{I}_r^+ \cup \mathbf{I}_r^-$ |
| $\mathbf{R}^U/\mathbf{R}^M/\mathbf{R}^{UT}$ | unit/material/utility resources |
| $\mathbf{R}_i^+/\mathbf{R}_i^-$ | resources produced/consumed by task $i$ |
| $\mathbf{R}_i$ | resources interacting task $i$, $\mathbf{R}_i = \mathbf{R}_i^+ \cup \mathbf{R}_i^-$ |

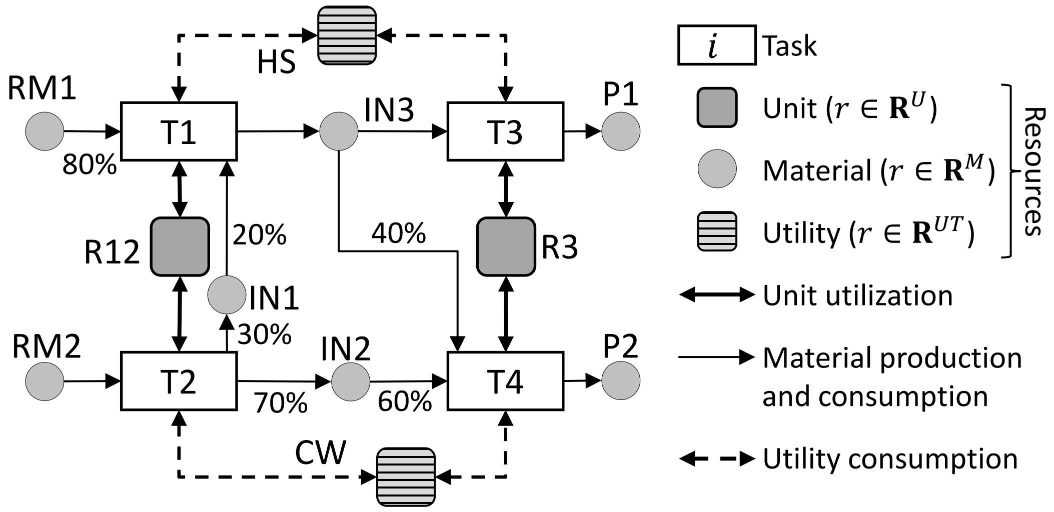
Parameters:

| | |
|---|---|
| $\beta_{ir}^{MIN}/\beta_{ir}^{MAX}$ | minimum/maximum extent of task $i$ executed on $r \in \mathbf{R}^U$ |
| $\gamma_i^F/\gamma_i^V$ | fixed/variable cost for carrying out task $i$ |
| $\tau_i$ | processing time for task $i$ |
| $\varphi_{ir}^S/\varphi_{ir}^E$ | fixed net consumption of $r$ by $i$ at the start/end of task $i$ |
| $\psi_{ir}^S/\psi_{ir}^E$ | variable consumption of $r$ by $i$ at the start/end of task $i$ |
| $\chi_r$ | capacity of resource $r$ |

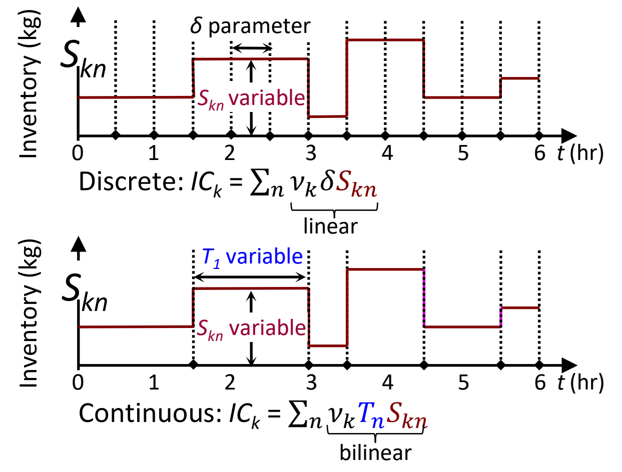
$\mathbf{I} = \{\mathrm{T1}, \mathrm{T2}, \mathrm{T3}, \mathrm{T4}\}$
$\mathbf{R}^U = \{\mathrm{R1}, \mathrm{R2}, \mathrm{R3}\}$
$\mathbf{R}^M = \{\mathrm{RM1}, \mathrm{RM2}, \mathrm{IN1}, \mathrm{IN2}, \mathrm{IN3}, \mathrm{P1}, \mathrm{P2}\}$
$\mathbf{R}^{UT} = \{\mathrm{HS}, \mathrm{CW}\}$

$\mathbf{I}_{\mathrm{R1}}^+ = \mathbf{I}_{\mathrm{R1}}^- = \mathbf{I}_{\mathrm{R2}}^+ = \mathbf{I}_{\mathrm{R2}}^- = \{\mathrm{T1}, \mathrm{T2}\}$,
$\mathbf{I}_{\mathrm{R3}}^+ = \mathbf{I}_{\mathrm{R3}}^- = \{\mathrm{T3}, \mathrm{T4}\}$;
$\mathbf{I}_{\mathrm{HS}}^+ = \mathbf{I}_{\mathrm{HS}}^- = \{\mathrm{T1}, \mathrm{T3}\}$, $\mathbf{I}_{\mathrm{CW}}^+ = \mathbf{I}_{\mathrm{CW}}^- = \{\mathrm{T2}, \mathrm{T4}\}$;
$\mathbf{I}_{\mathrm{IN1}}^+ = \{\mathrm{T2}\}, \mathbf{I}_{\mathrm{IN1}}^- = \{\mathrm{T1}\}$,
$\mathbf{I}_{\mathrm{IN3}}^+ = \{\mathrm{T1}\}, \mathbf{I}_{\mathrm{IN3}}^- = \{\mathrm{T3}, \mathrm{T4}\}$

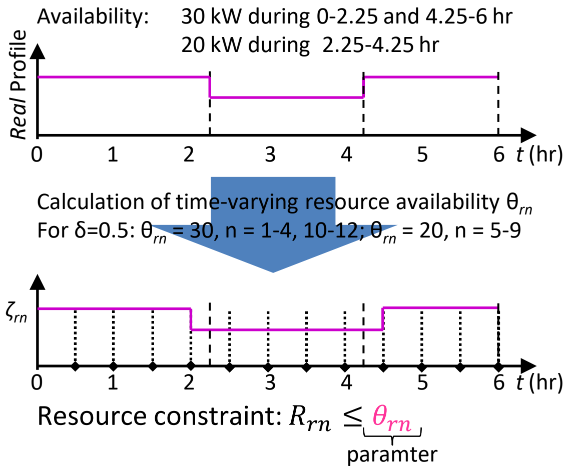
$\mathbf{R}_{\mathrm{T1}}^- = \{\mathrm{R1}, \mathrm{R2}, \mathrm{HS}, \mathrm{RM1}, \mathrm{IN1}\}$,
$\mathbf{R}_{\mathrm{T1}}^+ = \{\mathrm{R1}, \mathrm{R2}, \mathrm{HS}, \mathrm{IN3}\}$,
$\mathbf{R}_{\mathrm{T3}}^- = \{\mathrm{R3}, \mathrm{HS}, \mathrm{P1}\}$, $\mathbf{R}_{\mathrm{T1}}^+ = \{\mathrm{R1}, \mathrm{HS}, \mathrm{RM1}, \mathrm{IN1}\}$



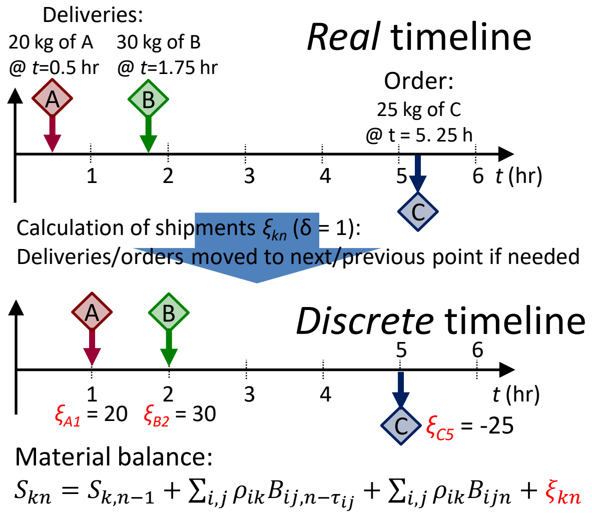
# Modeling Using Discrete Time Grid



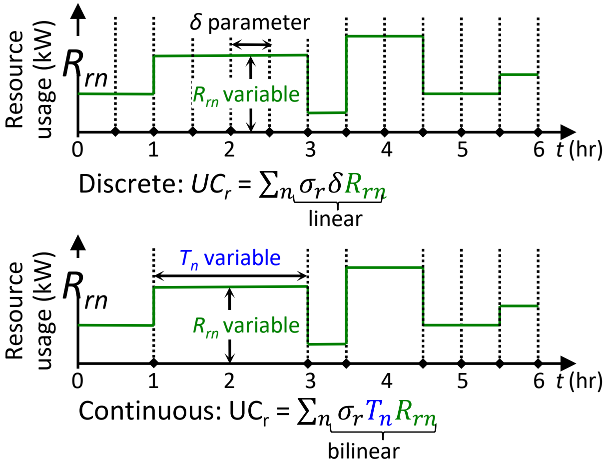

**A. Inventory cost, $IC_k$** ($v_k$: unit cost [\$/(kg·hr)])



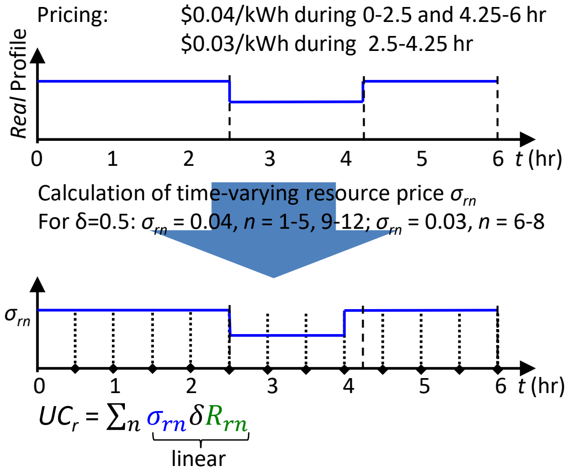

**C. Time-varying resource availability**



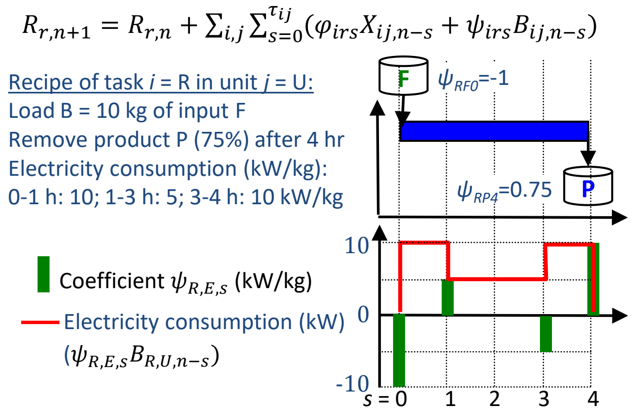

**E. Modeling of release and due times**




**B. Utility cost, $UC_r$** ($\sigma_r$: unit cost [\$/(kW·hr)])




**D. Time-varying resource pricing**

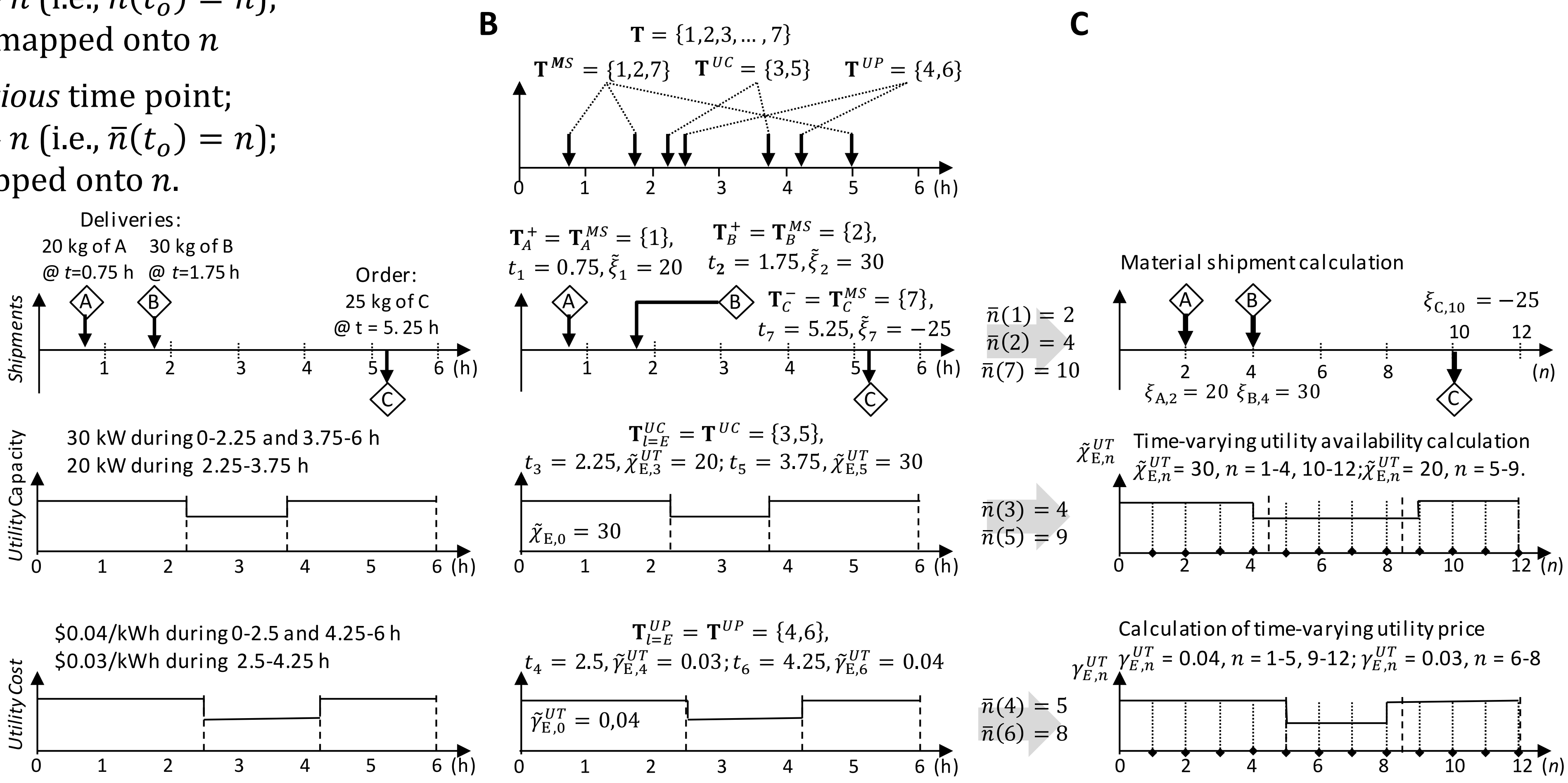



**F. Variable resource consumption during task**

# Data Generation

PRINCETON UNIVERSITY

- Intermediate shipments and time-varying utility capacity & cost are given in terms of an external time grid with points $t \in \mathbf{T}$,
- Shipments, $o \in \mathbf{O}$, are defined in terms of: (1) an associated material $\bar{k}(o)$; (2) timing $t_o$ (of a $t \in \mathbf{T}^{MS}$), and (3) amount $\tilde{\xi}_o$.
- $\mathbf{O}_k$: shipments of material $k$ (i.e., $\mathbf{O}_k = \{o|\bar{k}(o) = k\}$; $\mathbf{T}_k^+/\mathbf{T}_k^-$: time points where a delivery/order of material $k$ occurs; $\mathbf{T}_k^{MS} = \mathbf{T}_k^+ \cup \mathbf{T}_k^-$.
- Deliveries mapped onto the *next* time point;
  i.e., if $t_o \in (t_{n-1}, t_n]$ then $o \to n$ (i.e., $\bar{n}(t_o) = n$);
  $\mathbf{O}_{kn}^+$: deliveries of material $k$ mapped onto $n$
- Orders mapped onto the *previous* time point;
  i.e., if $t_o \in [t_n, t_{n+1})$ then $o \to n$ (i.e., $\bar{n}(t_o) = n$);
  $\mathbf{O}_{kn}^-$: orders of material $k$ mapped onto $n$.



# State-Task Network: Discrete Time Model

PRINCETON UNIVERSITY

## Variables

- $X_{ijn} \in \{0,1\}$: 1 if batch $i$ starts on $j$ at time point $n$
- $B_{ijn} \in \mathbb{R}_+$: batchsize of a batch of task $i$ starting on unit $j$ at $n$
- $I_{kn} \in \mathbb{R}_+$: inventory level of material $k$ during period $n$,

## Equations

- Task-unit assignment

  $$\sum_{i \in \mathbf{I}_j} \sum_{n' \in \mathbf{N}_{in}^U} X_{ijn'} \leq 1, \quad j, n$$

  where $\mathbf{N}_{in}^U = \{n - \bar{\tau}_i + 1, \ldots, n\}$
- Batchsize constraints

  $$\beta_j^{MIN} X_{ijn} \leq B_{ijn} \leq \beta_j^{MAX} X_{ijn}, \quad i, j, n$$
- Inventory balance

  $$I_{k,n+1} = I_{kn} + \sum_{i \in \mathbf{I}_k^+} \sum_j \rho_{ik} B_{ij,n-\bar{\tau}_{ij}} + \sum_{i \in \mathbf{I}_k^-} \sum_j \rho_{ik} B_{ijn} + \xi_{kn} - S_{kn} \leq \chi_k^M, \quad k, n$$
- Utility constraints

  $$\sum_{i,j \in \mathbf{J}_i} \sum_{n' \in \mathbf{N}_{i,n-1}^C} (\varphi_{il} X_{ijn'} + \psi_{il} B_{ijn'}) \leq \chi_{ln}^{UT}, \quad l, n$$
- Objective Function

  $$\max(C^{REV} - C^{PR} - C^{UT} - C^{CH} - C^{BCK} - C^{LS})$$

  where $C^{REV} = \sum_k \pi_k \sum_n S_{kn}$, $C^{PR} = \sum_{i,j} \{\gamma_{ij}^F \sum_n X_{ijn} + \gamma_{ij}^V \sum_n B_{ijn}\}$ …

# Resource-Task Network: Discrete Time Model

## Parameters

In addition to the already introduced parameters, we have:

- $\xi_{rn}^{M}$: net addition of (nonrenewable) material resource $r \in \mathbf{R}^{M}$ at time point $n$.
- $\chi_{rn}$: capacity of (renewable) resource $r \in \mathbf{R}^{UT}$ during period $n$.
- $\gamma_{rn}^{RES}$: cost of resource $r \in \mathbf{R}^{UT}$ during period $n$.

## Variables

- $X_{in} \in \{0,1\}$: 1 if batch $i$ starts at time point $n$
- $B_{in} \in \mathbb{R}_{+}$: batchsize (extent) of a batch of task $i$ starting at point $n$
- $R_{rn} \in \mathbb{R}_{+}$: availability of resource $r$ during period $n$

## Equations

- Unit resource availability

$$R_{r,n+1} = R_{rn} + \sum_{i\in\mathbf{I}_r} X_{i,n-\bar{\tau}_{ij}} - \sum_{i\in\mathbf{I}_r} X_{in}, \quad r \in \mathbf{R}^{U}, n$$

- Batchsize constraints

$$\beta_{ir}^{MIN} X_{in} \le B_{in} \le \beta_{ir}^{MAX} X_{in}, \quad i, r \in \mathbf{R}_i \cap \mathbf{R}^{U}, n$$

- Material resource availability

$$R_{r,n+1} = R_{rn} + \sum_{i\in\mathbf{I}_r^{+}} \psi_{ir}^{E} B_{i,n-\bar{\tau}_i} + \sum_{i\in\mathbf{I}_r^{-}} \psi_{ir}^{S} B_{in} + \xi_{rn}^{M} - S_{rn} \le \chi_r^{M}, \quad r \in \mathbf{R}^{M}, n$$

- Utility resource availability

$$R_{r,n+1} = R_{rn} + \sum_{i\in\mathbf{I}_r^{+}} (\varphi_{ir}^{E} X_{i,n-\bar{\tau}_i} + \psi_{ir}^{E} B_{i,n-\bar{\tau}_i}) - \sum_{i\in\mathbf{I}_r^{-}} (\varphi_{ir}^{S} X_{in} + \psi_{ir}^{S} B_{in}) + \xi_{rn}^{UT}, \quad r \in \mathbf{R}^{UT}, n$$

# Network Environments: Extensions

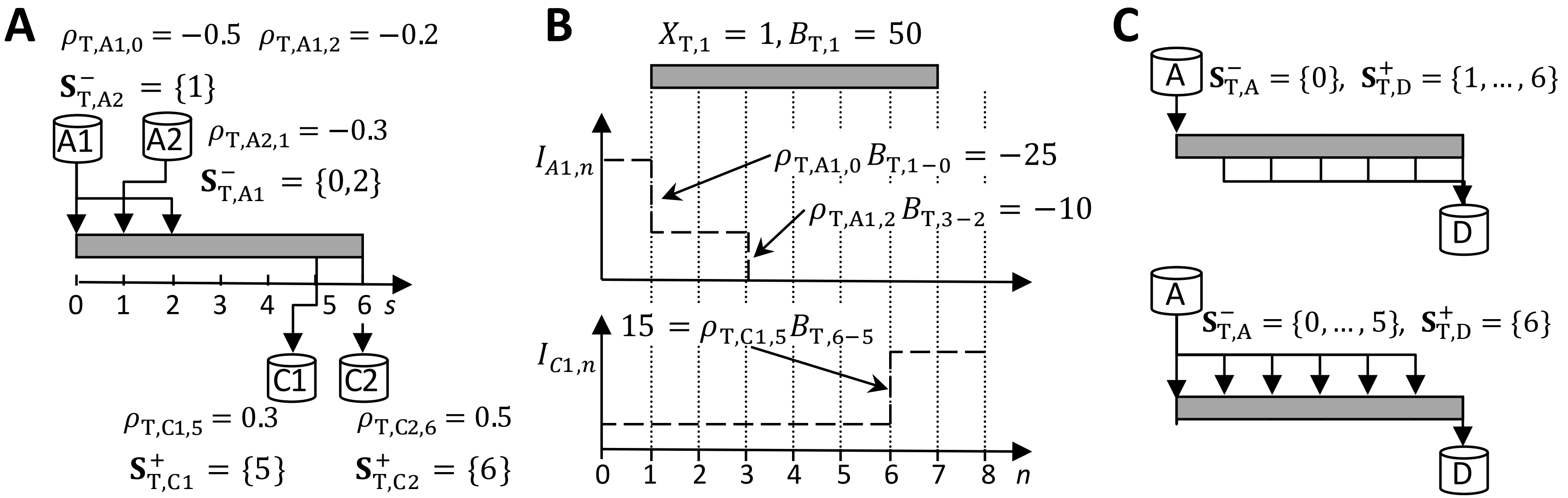
PRINCETON UNIVERSITY

## Material Consumption/Production During Task Execution

- A task may require consumption of some input materials after its start and may lead to output material production before its end.
- Introduce $\rho_{iks}$: conversion coefficient of task $i$ for material $k$, $s$ periods after the start of $i$.
- We assume that $\rho_{iks}$ are readily available, but we note that they should be calculated based on the chosen scheduling horizon step $\delta$;
- Previous coefficients, $\rho_{ik}$ (used assuming consumption/production occurs at the start ($s=0$)/end ($s=\tau_i$)) become:
  $\rho_{ik,s=0} = \rho_{ik} < 0$, for $k \in \mathbf{K}_i^-$ and $\rho_{ik,s=\tau_i} = \rho_{ik} > 0$, for $k \in \mathbf{K}_i^+$.
- Generalized STN material balance, assuming $|\mathbf{J}_i| = 1$, is

$$I_{k,n+1} = I_{kn} + \sum_{i \in \mathbf{I}_k^+} \sum_{s \in \mathbf{S}_{ik}^+} \rho_{iks} B_{i,n-s} + \sum_{i \in \mathbf{I}_k^-} \sum_{s \in \mathbf{S}_{ik}^-} \rho_{iks} B_{i,n-s} + \xi_{ik} - S_{kn} \le \chi_k^M, \quad k,n$$

where $\mathbf{S}_{ik}^+ = \{s: \rho_{iks} > 0\}$ and $\mathbf{S}_{ik}^- = \{s: \rho_{iks} < 0\}$ are the sets of points, with respect to the start of $i$, where $k$ is produced/consumed



# Network Environments: Extensions

PRINCETON UNIVERSITY

## Material Storage and Transfer

- If storage vessels are shared among materials, then, in the STN representation, they are treated explicitly as additional units.
- The set of units, **J**, in this case, has two subsets: the set of processing units, $\mathbf{J}^{PU}$, and the set of storage vessels, $\mathbf{J}^{SV}$.
- We define subsets: $\mathbf{K}_j$: materials that can be stored in $j \in \mathbf{J}^{SV}$;
  $\mathbf{K}^{SV}$: materials stored in shared vessels;
  $\mathbf{K}^{DV}$: materials stored in dedicated vessels;

## Storage in Shared Vessels

- $X_{jkn}^{ST} \in \{0,1\}$: 1 if material $k$ is stored in $j \in \mathbf{J}^{SV}$ during period $n$
- $I_{jkn} \in \mathbb{R}_+$: inventory of material $k$ in vessel $j$ during period $n$.
- Only one material can be stored at in a vessel at any time:

$$\sum_{k \in \mathbf{K}_j} X_{jkn}^{ST} \leq 1, \quad j \in \mathbf{J}^{SV}, n \quad (1); \qquad I_{jkn} \leq \chi_k^M X_{jkn}^{ST}, \quad j \in \mathbf{J}^{SV}, k \in \mathbf{K}_j, n \quad (2)$$

- In general, a material can be stored in many vessels
- If $\mathbf{J}_k^{SV}$ is the set of vessels material $k$ can be stored in and there is no dedicated storage vessel for $k$, then

$$\sum_{j \in \mathbf{J}_k^{SV}} I_{jk,n+1} = \sum_{j \in \mathbf{J}_k^{SV}} I_{jkn} + \sum_{i \in \mathbf{I}_k^+} \rho_{ik} B_{i,n-\tau_{ij}} + \sum_{i \in \mathbf{I}_k^-} \rho_{ik} B_{in} + \xi_{ik} - S_{kn}, \quad k, n$$

## Extensions

- Storage in processing units before and after the execution of a batch
- Resource-constrained material transfer tasks