HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Big Data Platforms for Machine Learning Applications

Keijo Heljanko ‹keijo.heljanko@helsinki.fi›

April 17, 2019

University of Helsinki
Department of Computer Science
Helsinki Center for Data Science (HiDATA)

Big Data

- Seagate sponsored IDC study estimates the Global Datasphere to be 33 Zettabytes (33 000 000 000 TB) in 2018 and to grow to 175 Zettabytes by 2025
- Data comes from: Video, sensor data, Internet sites, social media, AI Applications, healthcare, ...
- For example Netflix is collecting 1 PB of data per month from its video service user behaviour, total data warehouse 60+ PB
- According to Cisco, Internet traffic volume is growing 30% per year, while Big Data storage is growing 51% per year

Some Example Big Data Applications

- Web Search - Google, Bing
- Video Streaming and Recommendation - Netflix, Youtube
- Social Media & Advertisement - Facebook, Twitter, Instagram, Google
- Healthcare - Measurement data, e.g., next generation sequencing (genomics) data
- Computer Vision & Speech recognition - Autonomous cars, Virtual Assistants
- Industry - Preventive maintenance
- Business & Government - Data driven decision making

Massive Datacenters for Big Data Applications

- We need massive amounts of data storage
- We need massive amounts of computing power
- We need to be able to do this very cheaply by using economics of scale

Distributed Warehouse-scale Computing (WSC)

- The smallest unit of computation in Google scale is: Warehouse full of computers
- [WSC]: Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle: *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second edition* Morgan & Claypool Publishers 2013

  `http://dx.doi.org/10.2200/S00516ED2V01Y201306CAC024`
- The WSC book says:
  "...we must treat the datacenter itself as one massive warehouse-scale computer (WSC)."

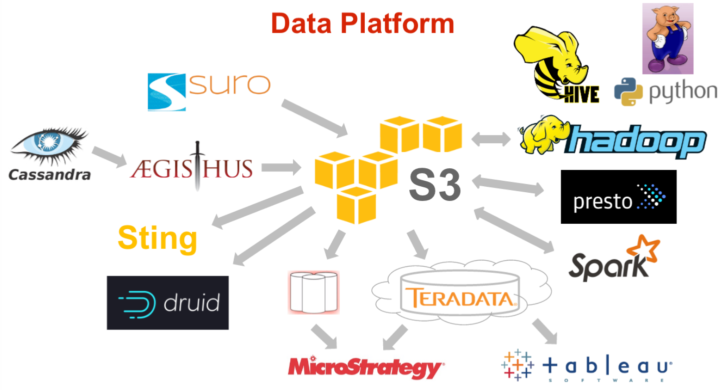
Hadoop - Linux of Big Data

- Hadoop = Open Source Distributed Operating System Distribution for Big Data
  - Based on “cloning” the Google architecture design
  - Fault tolerant distributed filesystem: HDFS
  - Batch processing for unstructured data: Hadoop MapReduce (HDD), Apache Spark (RAM)
  - Distributed SQL database queries for analytics: Apache Hive, Spark SQL, Cloudera Impala, Presto
  - Fault tolerant real-time distributed databases: HBase, Kudu
  - Machine learning libraries, text indexing & search, and much more
- Hadoop and Spark operate warehouse scale computers of Facebook, Netflix, Twitter, LinkedIn, ...

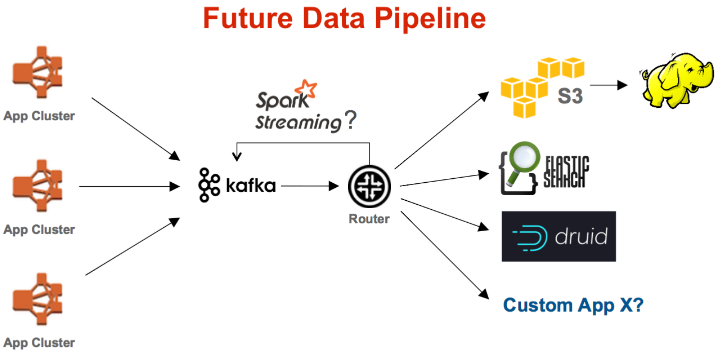
Example: Netflix Big Data Architecture

- Netflix is the largest Web based video service with more that 1/3 peak Internet traffic in USA
- Netflix has a 60+ PB of data collected from all its operations
- Company policy is to only do data driven business decisions
- Data is used to:
  - Recommend films
  - Choose which content to purchase
  - Improve user interface through A/B testing
  - Do ad-hoc customer analytics, etc.

Netflix Big Data Architecture



Netflix Data Collection



Big Data approach to Business Intelligence

- Storing data is becoming cheaper every year
  - The cost of storage of raw data is quite small for most applications
  - Just store all the raw data for future use
  - Do “schema on read” - Make the data usable by structuring it when needed
  - Because the raw data has already been collected, there is a history of data to analyze when an opportunity arises
  - Big Data platforms are needed to handle the vast masses of (potentially unstructured) data
  - The Gartner 3 Vs of Big Data: Volume, Variety, Velocity

Conclusions

- Artificial Intelligence applications require a Big Data backend for data collection and analytics
- Hadoop is becoming the “Linux distribution for Big Data”, including also other components such as Apache Spark for main memory computing
- Hadoop consists of a number of interoperable open source components
- Commercial support is available from commercial vendors, e.g., Cloudera and MapR
- There is a move to hosted Big Data Applications: Billing is done on data volume being processed instead of number of computers used

CAUSALITY DISCOVERY APPROACH FOR NONLINEAR TIMESERIES

OUTLINE

Overview & Research Problems

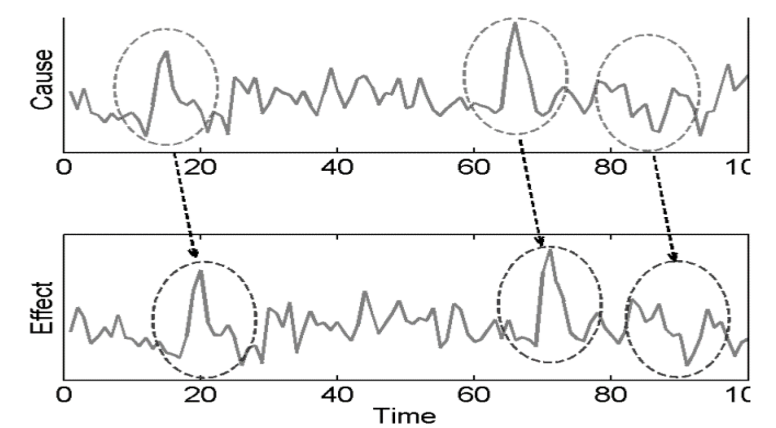
Background & Problem Formulation

Simulation Experiments

Concurrent and Future Work

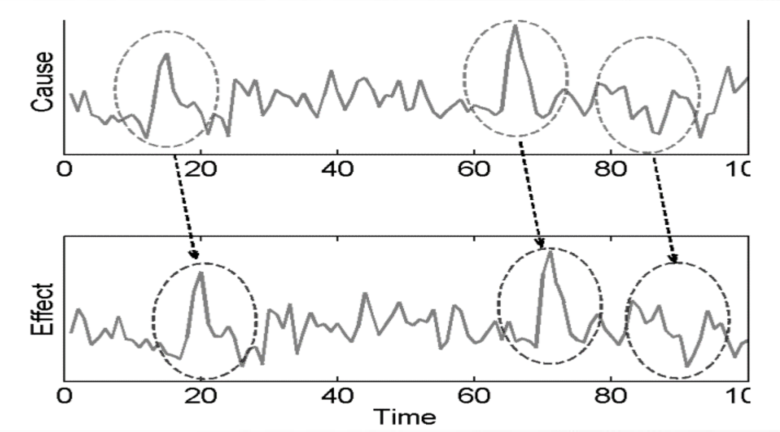
OVERVIEW & RESEARCH PROBLEMS

- (Granger) causality quantifies the extent to which one time series is predictive of another.
- Most classical methods of estimating Granger causality assume linear timeseries.
- More recent approaches of estimating Granger causality for nonlinear timeseries are not generic.
- Applicable for increasing predictive model interpretability, feature selection and root cause analysis (e.g. what caused a paper machine break?)



BACKGROUND & PROBLEM FORMULATION

- Granger causality
  - Cause is prior to effect.
  - The cause makes unique changes in the effect (i.e. contains unique info. about it).
  - If a signal ***X*** "Granger-causes" (or "G-causes") a signal ***Y***, then past values of ***X*** should contain information that helps predict ***Y*** above and beyond the information contained in past values of ***Y*** alone.



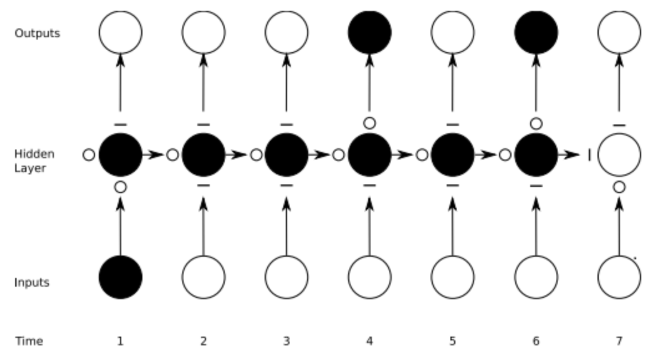
$$Y(t) = \sum_{l=1}^{L} a_l Y(t-l) + \epsilon_1$$

$$Y(t) = \sum_{l=1}^{L} a_l' Y(t-l) + \sum_{l=1}^{L} b_l' X(t-l) + \epsilon_2,$$

BACKGROUND & PROBLEM FORMULATION

- Nonlinear prediction models for Granger causality estimation
  - We model nonlinear dynamics with Random Forest and LSTM.
  - Lagged versions of predictor variables are used in Random Forest.
  - Lagged versions of predictor variables are inherently considered in LSTM as lookback window.
  - How to quantify Granger causality?

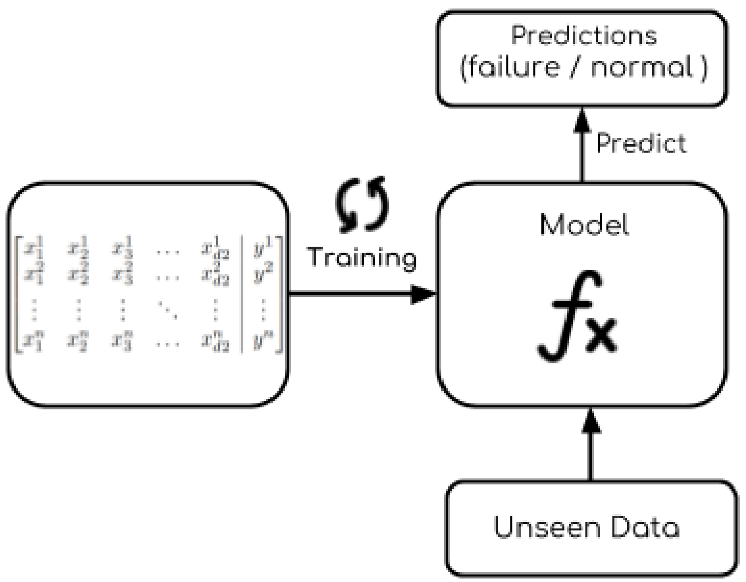
$$
\begin{aligned}
Y(t) &= \sum_{l=1}^{L} a_l Y(t-l) + \epsilon_1 \\
Y(t) &= \sum_{l=1}^{L} a_l' Y(t-l) + \sum_{l=1}^{L} b_l' X(t-l) + \epsilon_2,
\end{aligned}
$$



BACKGROUND & PROBLEM FORMULATION

- Quantifying Granger causality estimation
  - Accuracy difference between a model with or without a lagged version of a predictor variable.
  - But that means retraining our model ***lag * predictors*** times. Any better solution?
  - What if we permute or apply noise on each predictor during prediction (evaluation) instead of fully dropping it?

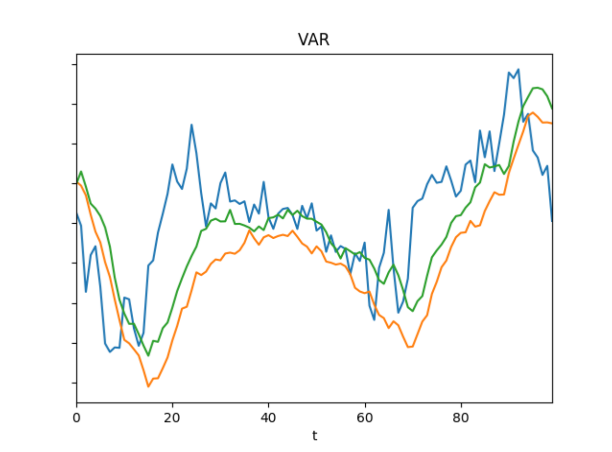
$$
\begin{aligned}
Y(t) &= \sum_{l=1}^{L} a_l Y(t-l) + \epsilon_1 \\
Y(t) &= \sum_{l=1}^{L} a_l' Y(t-l) + \sum_{l=1}^{L} b_l' X(t-l) + \epsilon_2,
\end{aligned}
$$

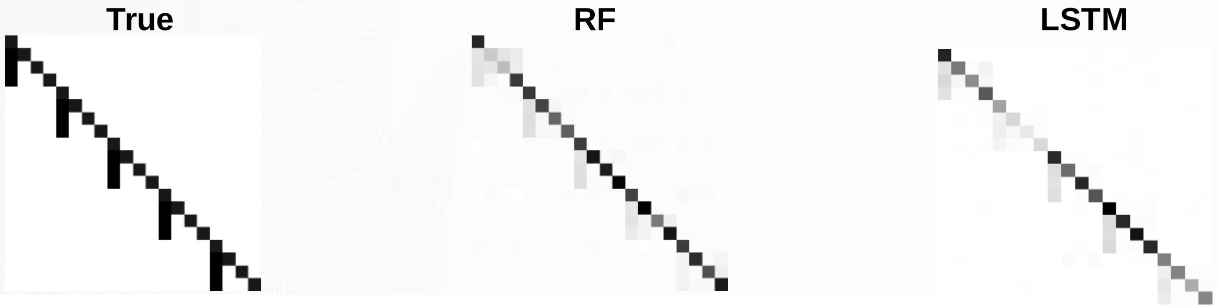


SIMULATION EXPERIMENTS

- Linear Vector Autoregressive Model
  - Simulated VAR model with known sparse ***A*** matrix.
  - Make prediction of each variable using all others as predictors.
  - Permute each predictor variable and make prediction again.
  - Calculate variable dependency based on reduction in accuracy (R2).

$$x_t = \sum_{k=1}^{K} A^{(k)} x_{t-k} + e_t,$$



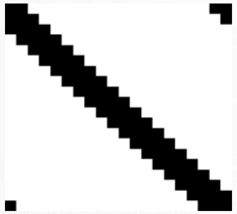


SIMULATION EXPERIMENTS

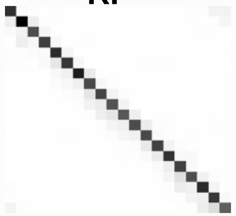
- Nonlinear Lorenz96 Model
  - Simulated data using Lorenz96 model with a given forcing value. Such model is often used to model climate dynamics.
  - Make prediction of each variable using all others as predictors.
  - Permute each predictor variable and make prediction again.
  - Calculate variable dependency based on reduction in accuracy (R2).

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F$$

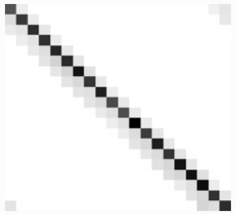
True

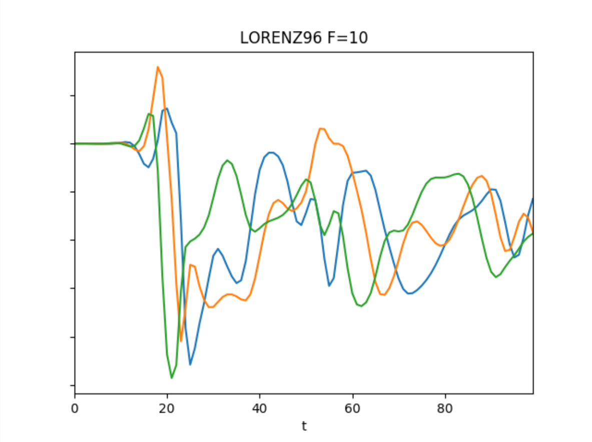


RF



LSTM





CONCURRENT & FUTURE WORK

- Experiment on real world
  - Gene expression and regulation dynamics
  - Human motion capture data
  - Manufacturing process data (root cause analysis)
- Adding numerical measures (AUROC and AUPR)
- Comparison with regularization based methods
- Other conditional permutation (noise addition) schemes

Thank You!

Questions?

Aalto University
School of Chemical
Engineering

Efficient and Systematic Production Scheduling Formulations

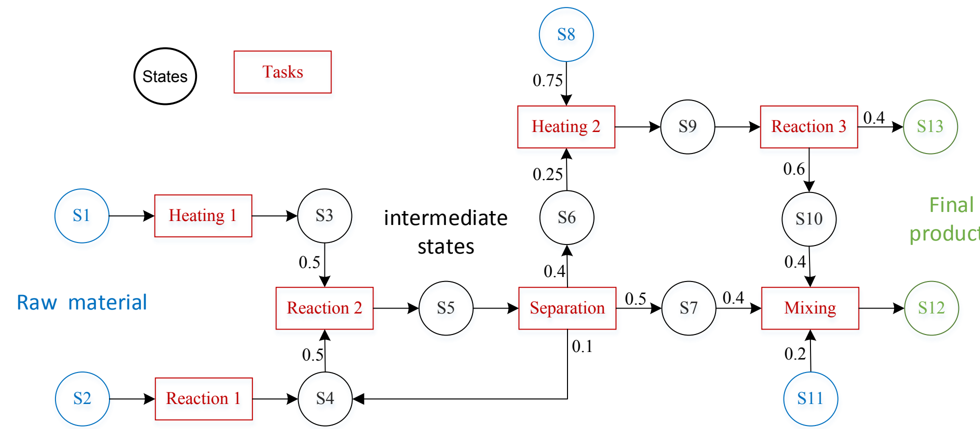
Hossein Mostafaei (hossein.mostafaei@aalto.fi)
Iiro Harjunkoski (iiro.harjunkoski@aalto.fi)

Seminar, 17.4.2019

Introduction

- **Batch Plant Scheduling**

- Optimal allocation of a set of limited resources to tasks over time
- Generic representations of batch processes: State Task Network, STN (Kondili et al., 1993) or Resource Task Network, RTN (Pantelides, 1994)



State task Network (STN)

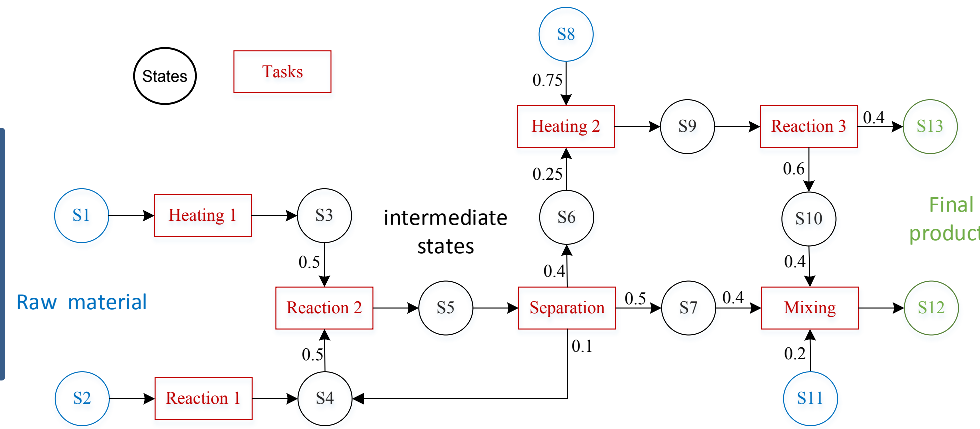
Introduction

- **Batch Plant Scheduling**

- Optimal allocation of a set of limited resources to tasks over time
- Generic representations of batch processes: State Task Network, STN (Kondili et al., 1993) or Resource Task Network, RTN (Pantelides, 1994)

To be determined

Number, size, assignment, sequencing, and timing of a set of tasks consuming/producing a set of materials during a given time horizon



State task Network (STN)

Introduction

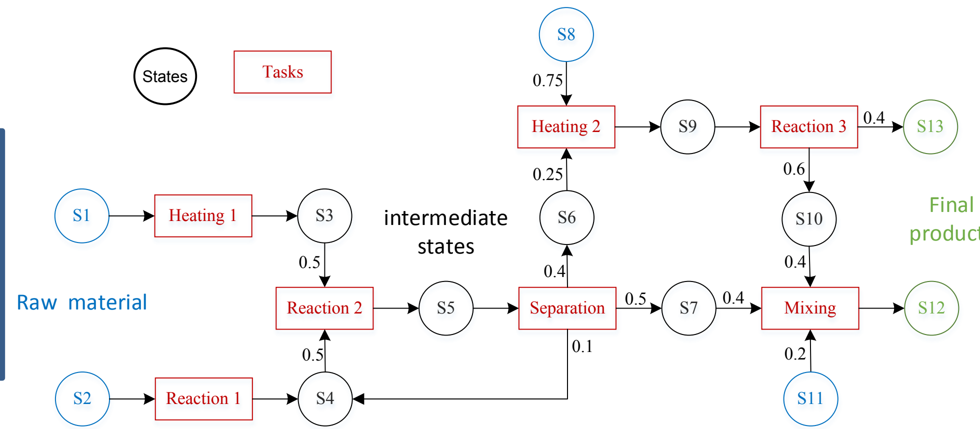
- ## Batch Plant Scheduling

- Optimal allocation of a set of limited resources to tasks over time
- Generic representations of batch processes: State Task Network, STN (Kondili et al., 1993) or Resource Task Network, RTN (Pantelides, 1994)

To be determined

Number, size, assignment, sequencing, and timing of a set of tasks consuming/producing a set of materials during a given time horizon

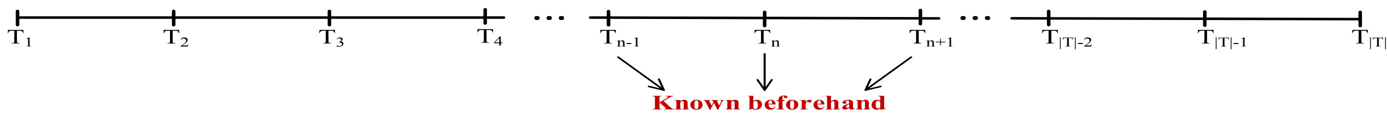
Maximum profit
Minimum cost / makespan



State task Network (STN)

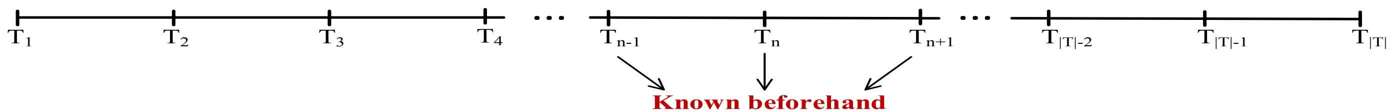
Time representation

- **Discrete-time**
  - **The length of time slots is known beforehand (from minute to hours)**

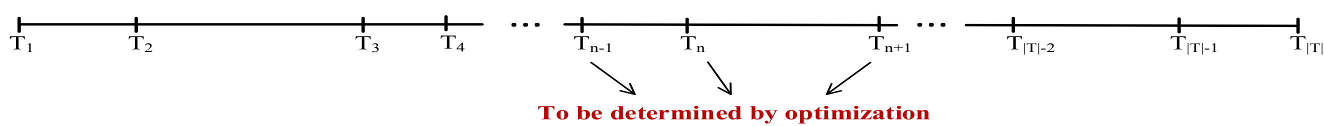


Time representation

- Discrete-time
  - The length of time slots is known beforehand (from minute to hours)



- Continuous-time
  - The length of time slots is a continuous variable
  - Task can be processed at any time of scheduling horizon



Scheduling formulation

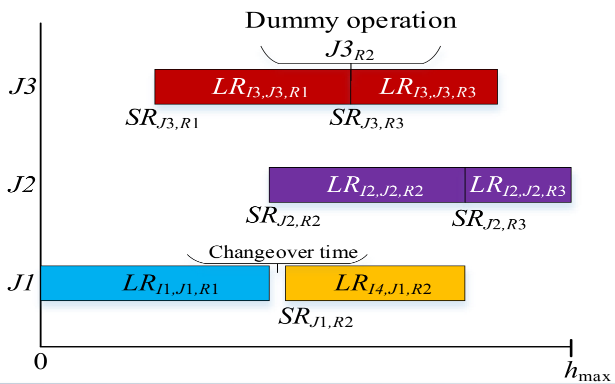
- **We use STN to represent the problem**
- **To arrange task in units we define the concept of runs**

- Run =Time slot
- Place holder for a task
- Which task will be processed in a run

$X_{i,j,r} = 1$ if task i is processed in unit j during run r

$V_{i,j,r} =$ batch size

$L_{i,j,r} =$ length / duration of a run



Novelty

- Formulation has fewer continuous variables and constraints
- Easier to solve

Scheduling formulation

Logical relation	Comments	Boolean Expression	Representation as Linear Inequalities
Logical OR		$Y_1 \vee Y_2 \vee ... \vee Y_n$	$y_1+y_2+...+y_n \geq 1$
Logical AND		$Y_1 \wedge Y_2 \wedge ... \wedge Y_n$	$\left\{ \begin{array}{l} y_1=1 \\ y_2=1 \\ ... \\ y_n=1 \end{array} \right.$
Implication	$Y_1 \Rightarrow Y_2$	$\neg Y_1 \vee Y_2$	$1-y_1+y_2 \geq 1$
Equivalence	Y_1 if and only if Y_2 $(Y_1 \Rightarrow Y_2 \wedge Y_2 \Rightarrow Y_1)$	$(\neg Y_1 \vee Y_2) \wedge (\neg Y_2 \vee Y_1)$	$y_1=y_2$
Exclusive OR	Exactly one of the variables is true	$Y_1 \veebar Y_1 \veebar ... \veebar Y_n$	$y_1+y_2+...+y_n=1$

Scheduling formulation

- **Allocation and processing time**

$$\bigvee_{i \in I_j} \begin{bmatrix} X_{i,j,r} \\ v_{i,j}^{\min} \leq V_{i,j,r} \leq v_{i,j}^{\max} \\ LR_{i,j,r} = cp_{i,j} + vp_{i,j} V_{i,j,r} \end{bmatrix} \bigvee \begin{bmatrix} X_{j,r}^{\text{no task}} \\ V_{i,j,r} = 0, \ \ \forall i \in I_j \\ LR_{i,j,r} = 0, \ \ \forall i \in I_j \end{bmatrix} \ \ \forall j \in J, r \in R$$

Logical relation	Comments	Boolean Expression	Representation as Linear Inequalities
Logical OR		$Y_1 \vee Y_2 \vee \ldots \vee Y_n$	$y_1+y_2+\ldots+y_n \geq 1$
Logical AND		$Y_1 \wedge Y_2 \wedge \ldots \wedge Y_n$	$\left\{ \begin{array}{l} y_1=1 \\ y_2=1 \\ \ldots \\ y_n=1 \end{array} \right.$
Implication	$Y_1 \Rightarrow Y_2$	$\neg Y_1 \vee Y_2$	$1-y_1+y_2 \geq 1$
Equivalence	Y_1 if and only if Y_2 $(Y_1 \Rightarrow Y_2 \wedge Y_2 \Rightarrow Y_1)$	$(\neg Y_1 \vee Y_2) \wedge (\neg Y_2 \vee Y_1)$	$y_1=y_2$
Exclusive OR	Exactly one of the variables is true	$Y_1 \underline{\vee} Y_1 \underline{\vee} \ldots \underline{\vee} Y_n$	$y_1+y_2+\ldots+y_n=1$

Scheduling formulation

- **Allocation and processing time**

$$\bigvee_{i\in I_j}\begin{bmatrix} X_{i,j,r} \\ v_{i,j}^{\min} \le V_{i,j,r} \le v_{i,j}^{\max} \\ LR_{i,j,r} = cp_{i,j} + vp_{i,j}V_{i,j,r}\end{bmatrix} \bigvee \begin{bmatrix} X_{j,r}^{\text{no task}} \\ V_{i,j,r} = 0, \ \forall i \in I_j \\ LR_{i,j,r} = 0, \ \forall i \in I_j\end{bmatrix} \quad \forall j \in J, r \in R$$

Logical relation	Comments	Boolean Expression	Representation as Linear Inequalities
Logical OR		$Y_1 \vee Y_2 \vee \ldots \vee Y_n$	$y_1+y_2+\ldots+y_n \ge 1$
Logical AND		$Y_1 \wedge Y_2 \wedge \ldots \wedge Y_n$	$\begin{cases} y_1=1 \\ y_2=1 \\ \ldots \\ y_n=1 \end{cases}$
Implication	$Y_1 \Rightarrow Y_2$	$\neg Y_1 \vee Y_2$	$1-y_1+y_2 \ge 1$
Equivalence	Y_1 if and only if Y_2 $(Y_1 \Rightarrow Y_2 \wedge Y_2 \Rightarrow Y_1)$	$(\neg Y_1 \vee Y_2) \wedge (\neg Y_2 \vee Y_1)$	$y_1 = y_2$
Exclusive OR	Exactly one of the variables is true	$Y_1 \underline{\vee} Y_1 \underline{\vee} \ldots \underline{\vee} Y_n$	$y_1+y_2+\ldots+y_n=1$

- By replacing $X_{j,r}^{\text{no task}} 1 - \sum_i X_{i,j,r}$ in convex hull reformulation of disjunction:

$$\sum_{i\in I_j} X_{i,j,r} \le 1\,, \qquad \forall j \in J, r \in R$$

$$v_{i,j}^{\min} X_{i,j,r} \le V_{i,j,r} \le v_{i,j}^{\max} X_{i,j,r} \quad \forall i \in I_j, j \in J, r \in R$$

$$LR_{i,j,r} = cp_{i,j} X_{i,j,r} + vp_{i,j} V_{i,j,r} \quad \forall i \in I_j \ \ j \in J, r \in R$$

Scheduling formulation

- **Sequencing and changeover constraints**

- Run r in unit j always starts after run $r\text{-}1$ in the same unit

Start time of run r in unit j

$$SR_{j,r} \geq SR_{j,r-1} + \sum_{i \in I_j} LR_{i,j,r-1} \qquad \forall\ j \in J, r \in R|_{r>1}$$

$$SR_{j,r} \geq SR_{j',r-1} + \sum_{i \in I_{j'}} LR_{i,j',r-1}, \qquad \forall\, j \in J_s^c, j' \in J_s^p (j \neq j'), s \in S, r \in R|_{r>1}$$

Scheduling formulation

- **Mass balance and demand**

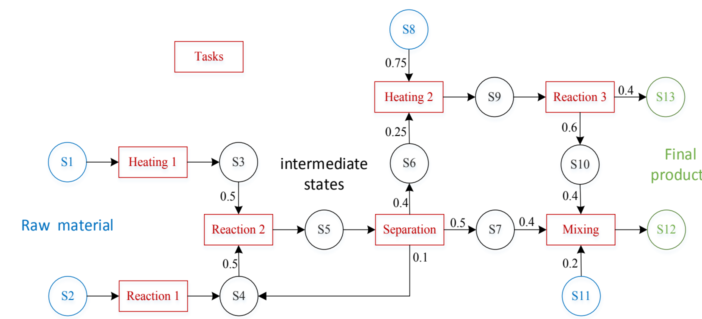
$$F_{s,r} = f_s^{\text{initial}} + \sum_{r'<r} \sum_{i \in I_s^p} \rho_{i,s}^p \sum_{j \in J_i} V_{i,j,r'} - \sum_{r' \leq r} \sum_{i \in I_s^c} \rho_{i,s}^c \sum_{j \in J_i} V_{i,j,r'} \quad \forall s \in S, r \in R$$

$$f_s^{\min} \leq F_{s,r} \leq f_s^{\max} \quad \forall s \in S, r \in R$$

J_i: Units that can process tasks i

I_s^p : Tasks i that produce state s

I_s^p : Tasks i that consume state s



Scheduling formulation

- ## Mass balance and demand

$$F_{s,r} = f_s^{\text{initial}} + \sum_{r'<r} \sum_{i \in I_s^p} \rho_{i,s}^p \sum_{j \in J_i} V_{i,j,r'} - \sum_{r' \le r} \sum_{i \in I_s^c} \rho_{i,s}^c \sum_{j \in J_i} V_{i,j,r'} \quad \forall s \in S, r \in R$$

$$f_s^{\min} \le F_{s,r} \le f_s^{\max} \quad \forall s \in S, r \in R$$

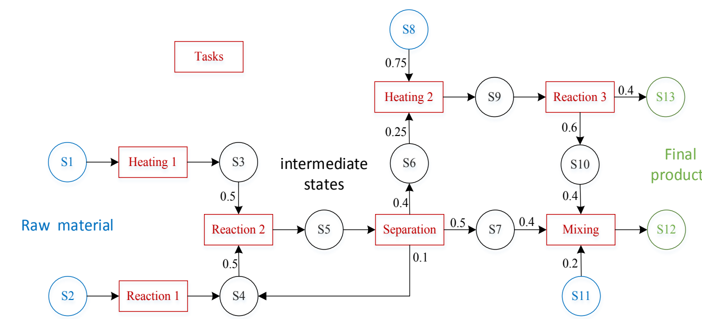
The amount of final states in last run should be as large as demand d_s

$$F_{s,r} \ge d_s, \quad \forall s \in SM, r = |R|$$

J_i: Units that can process tasks i

I_s^p : Tasks i that produce state s

I_s^p : Tasks i that consume state s

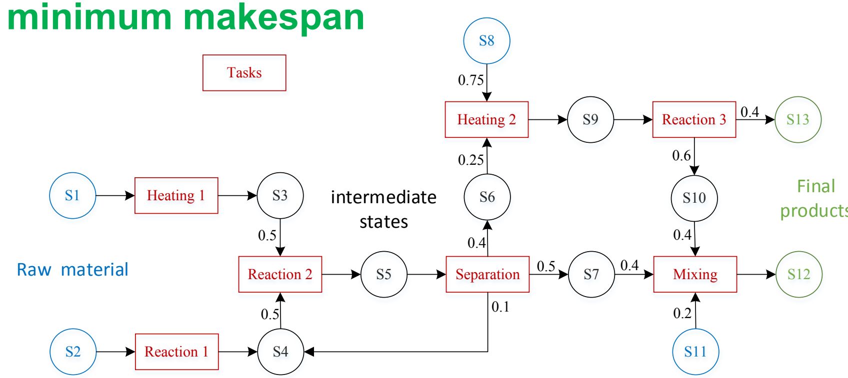


Results for a complex and comprehensive case study

Goal: Satisfy demand at final states S12 and S13 with a minimum makespan

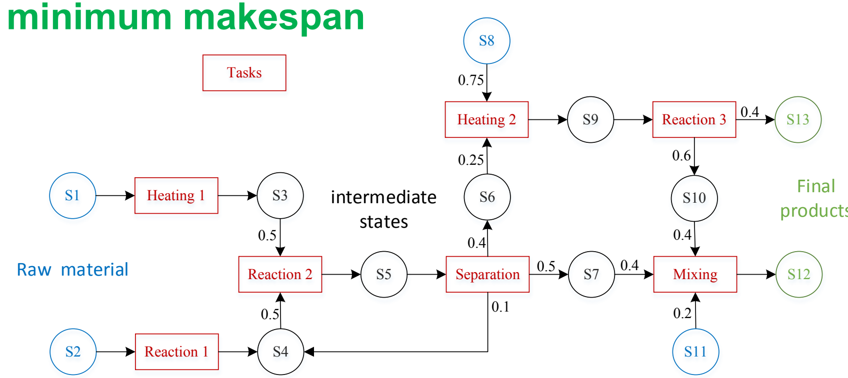
Demand:

- Example 1: $d_{S12} = 100$ mu, $d_{S13} = 200$ mu
- Example 2: $d_{S12} = 930$ mu, $d_{S13} = 840$ mu



Results for a complex and comprehensive case study

Goal: Satisfy demand at final states S12 and S13 with a minimum makespan



Demand:

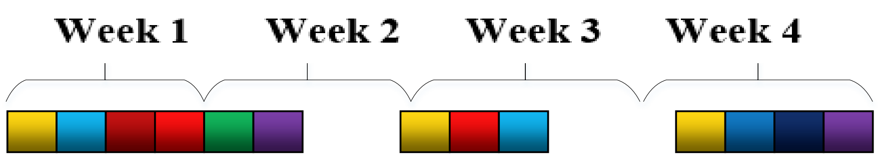
- Example 1: $d_{S12} = 100$ mu, $d_{S13} = 200$ mu
- Example 2: $d_{S12} = 930$ mu, $d_{S13} = 840$ mu

	#runs	CPU (s)	Binary variables	Total variables	Constraints	MILP (h)	RMILP (h)
		$d_{S12} = 100$ mu, $d_{S13} = 200$ mu					
Shaik and Floudas	8	0.81	88	458	730	13.36	11.25
Our	8	0.46	83	418	598	13.36	11.62
		$d_{S12} = 930$ mu, $d_{S13} = 840$ mu					
Shaik and Floudas	29	719.60	319	1655	2704	51.82	49.92
Our	29	5.61	319	1510	2178	51.82	50.26

- Smaller problem size
- Same number of time slots
- Lower CPU time

Continuous Processes (grade change optimization)

By slightly changing the generic model for batch plant scheduling, we can easily account for grade change optimization to minimize makespan or transition cost.



$$\bigvee_{i\in I}\begin{bmatrix} X_{i,r} \\ v_{\min,i} \le V_{i,r} \le v_{\max,i} \\ \frac{V_{i,r}}{v_{\max,i}} \le LR_r \le \frac{V_{i,r}}{v_{\min,i}} \end{bmatrix} \bigvee \begin{bmatrix} X_r^{no\ i} \\ V_{i,r} = 0 \ \forall i \in I \\ LR_r \le h \end{bmatrix}, \qquad \forall r \in R|_{r\ge 1}$$

$$SR_r = SR_{r-1} + LR_{r-1} \ \forall r \in R|_{r\ge 2}$$

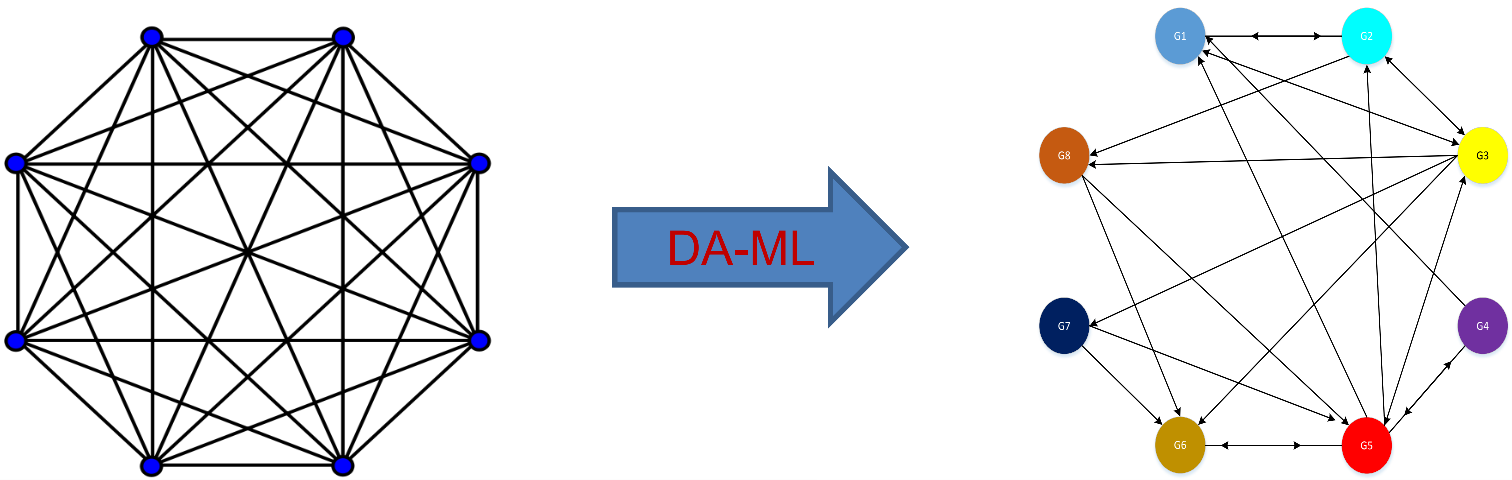
$$\sum_{r\in R} LR_r \le h$$

Continuous Processes (grade change optimization)

- Problem very hard to solve with more than 8 grades and multiple due dates
- Data analytics (DA) and machine learning (ML) methods are used to cut off the feasible region

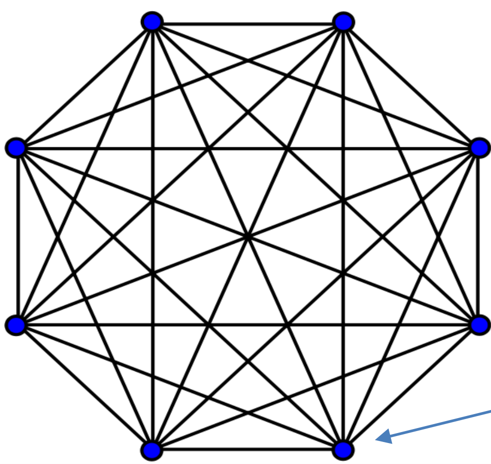
Continuous Processes (grade change optimization)

- Problem very hard to solve with more than 8 grades and multiple due dates
- Data analytics (DA) and machine learning (ML) methods are used to cut off the feasible region
- Using historical data, DA and ML, we are able to generate heuristic constraints and omit large number of connections between grades



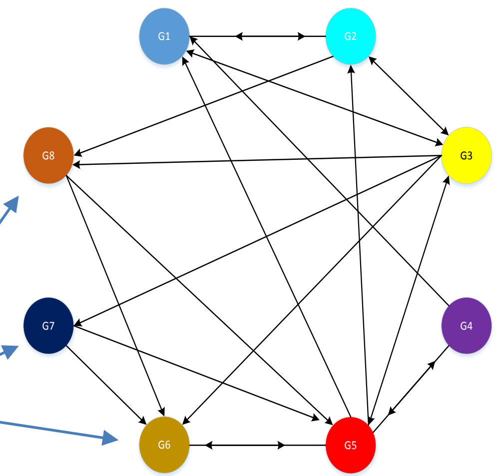
Continuous Processes (grade change optimization)

- Problem very hard to solve with more than 8 grades and multiple due dates
- Data analytics (DA) and machine learning (ML) methods are used to cut off the feasible region
- Using historical data, DA and ML, we are able to generate heuristic constraints and omit large number of connections between grades



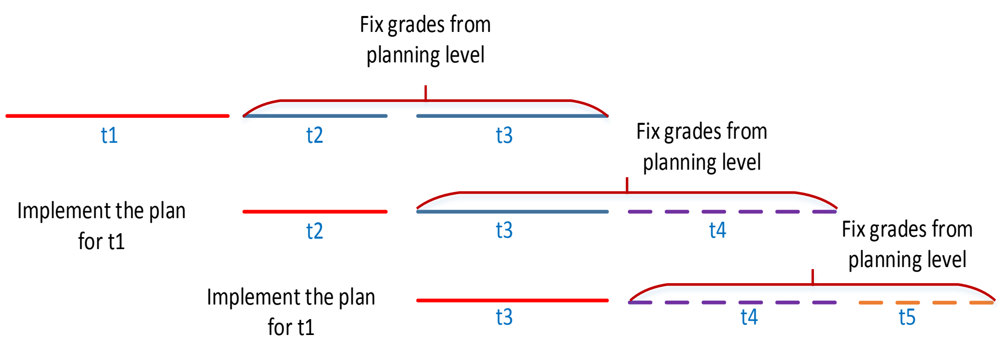
DA-ML

each vertex in graph represents a grade



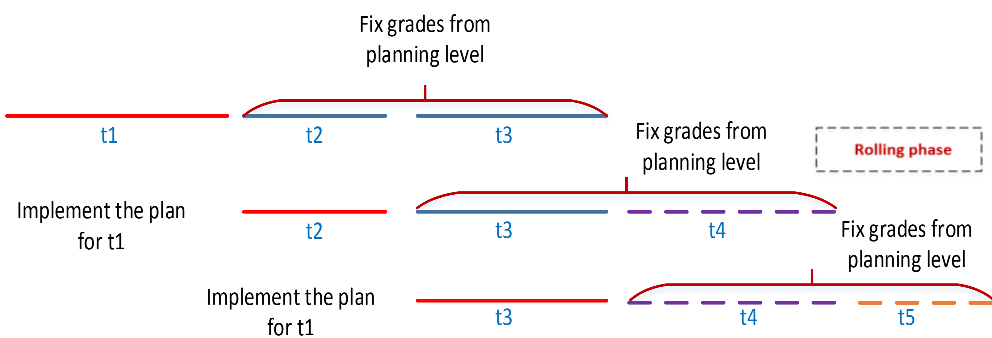
Continuous Processes (grade change optimization)

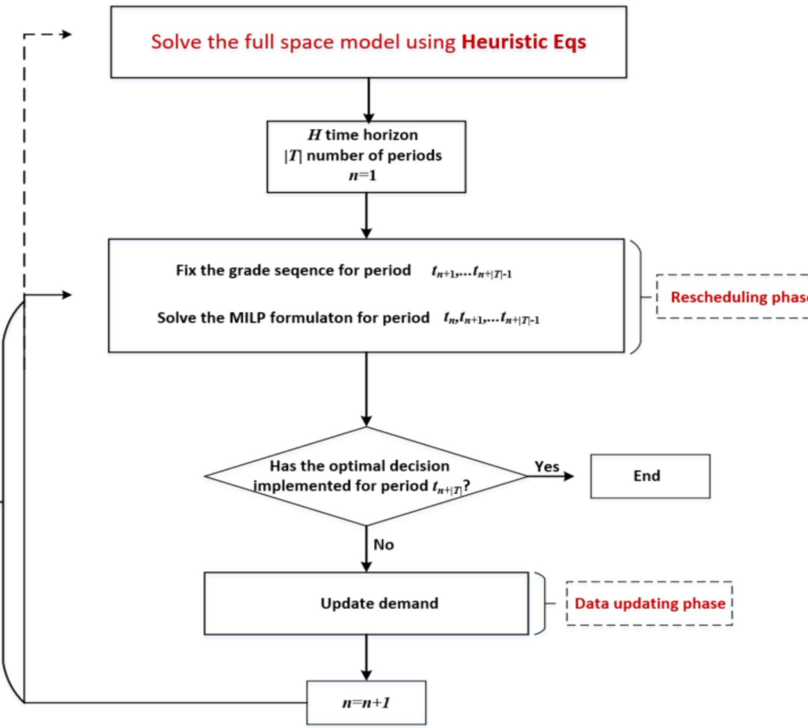
- Heuristic approach: Fast but non-optimal
- With a rolling horizon (RH) approach we can improve the solution quality obtained from the heuristic approach



Continuous Processes (grade change optimization)

- Heuristic approach: Fast but non-optimal
- With a rolling horizon (RH) approach we can improve the solution quality obtained from the heuristic approach





Grade change optimization (8 grades)

```
TABLE DEM(G,T) Demand in periods T1 (168 h) and T2 (336 h)
            T1         T2
G1          4000       2000
G2          3000       6000
G3          0          2000
G4          2000       2000
G5          4000       4000
G6          0          0
G7          5000       2000
G8          2000       3000;
```

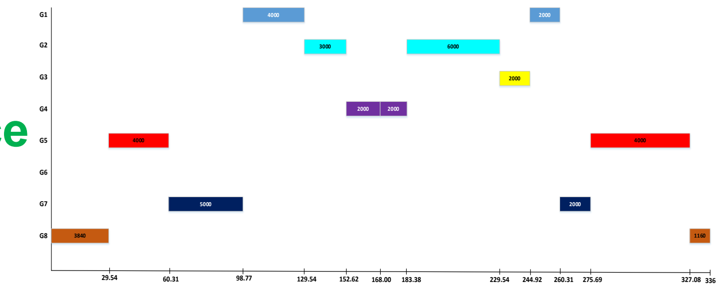
	Full Space	Heuristic	Rolling Horizon T2 Fix	Rolling Horizon T1 fix
Number of runs	13	13	13 (6+7)	13 (6+7)
CPU (s)	412.06	5.84	0.35	0.54
Binary variables	140	140	104	104
Total variables	499	499	352	352
Constraints	2954	3058	2502	2502
Objective Function	148.70	172.6	159.70	148.7

Grade change optimization (8 grades)

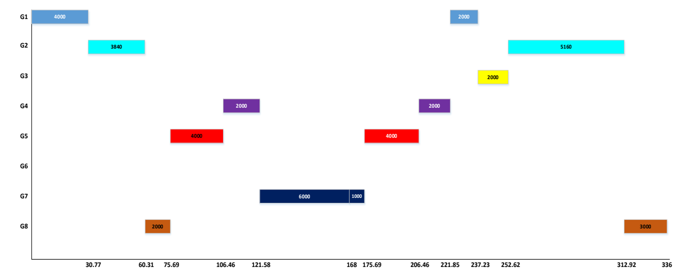
```
TABLE DEM(G,T) Demand in periods T1 (168 h) and T2 (336 h)
           T1        T2
G1         4000      2000
G2         3000      6000
G3         0         2000
G4         2000      2000
G5         4000      4000
G6         0         0
G7         5000      2000
G8         2000      3000;
```

	Full Space	Heuristic	Rolling Horizon T2 Fix	Rolling Horizon T1 fix
Number of runs	13	13	13 (6+7)	13 (6+7)
CPU (s)	412.06	5.84	0.35	0.54
Binary variables	140	140	104	104
Total variables	499	499	352	352
Constraints	2954	3058	2502	2502
Objective Function	148.70	172.6	159.70	148.7

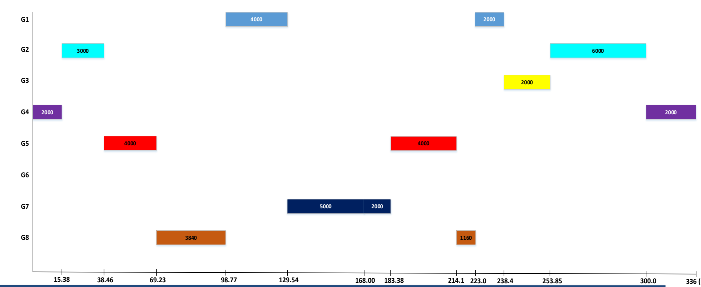
Full space



DL-ML

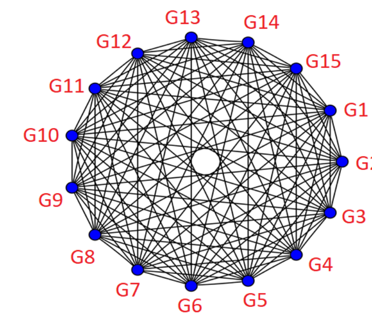


RH



Aalto University
School of Chemical Engineering

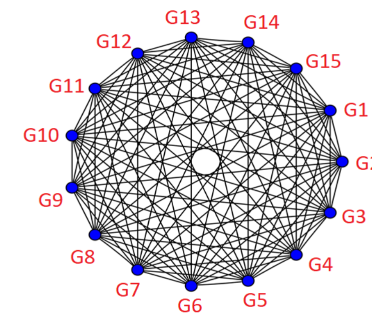
Grade change optimization (15 grades)



- Full space model not solved to optimality in 20000 CPU-s
- Number of possible sequences 15! (1,3*10^{12})

	Full Space	Heuristic	Rolling Horizon T2 Fix	Rolling Horizon T2 Fix	Rolling Horizon T1 fix	Rolling Horizon T1 fix
Number of runs	18	18 (10+8)	10+8	11+8	11+8	11+9
CPU (s)	20000	251.95	39.68	80	0.8	1.17
Binary variables	306	306	281	302	302	311
Total variables	645	645	606	645	645	672
Constraints	5191	3423	6659	7158	7116	7422
Objective Function	7300	7515.6	7300	7098.94	7098.94	6844.4

Grade change optimization (15 grades)

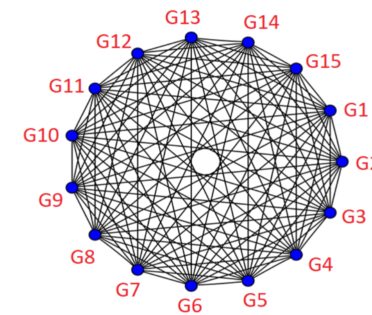


- Full space model not solved to optimality in 20000 CPU-s
- Number of possible sequences 15! $(1{,}3 * 10^{12})$

	Full Space	Heuristic	Rolling Horizon T2 Fix	Rolling Horizon T2 Fix	Rolling Horizon T1 fix	Rolling Horizon T1 fix
Number of runs	18	18 (10+8)	10+8	11+8	11+8	11+9
CPU (s)	20000	251.95	39.68	80	0.8	1.17
Binary variables	306	306	281	302	302	311
Total variables	645	645	606	645	645	672
Constraints	5191	3423	6659	7158	7116	7422
Objective Function	7300	7515.6	7300	7098.94	7098.94	6844.4

Total CPU time by the algorithm: 373 s

Grade change optimization (15 grades)



- Full space model not solved to optimality in 20000 CPU-s
- Number of possible sequences 15! $(1{,}3 \cdot 10^{12})$

	Full Space	Heuristic	Rolling Horizon T2 Fix	Rolling Horizon T2 Fix	Rolling Horizon T1 fix	Rolling Horizon T1 fix
Number of runs	18	18 (10+8)	10+8	11+8	11+8	11+9
CPU (s)	20000	251.95	39.68	80	0.8	1.17
Binary variables	306	306	281	302	302	311
Total variables	645	645	606	645	645	672
Constraints	5191	3423	6659	7158	7116	7422
Objective Function	7300	7515.6	7300	7098.94	7098.94	6844.4

Solution quality improved by 6%

Conclusions and future directions

- Mathematical model formulation has a big impact on the solution efficiency
  - New model for batch plant scheduling is faster than state-of-the-art models in the literature
- DA and ML methods can support the solution of scheduling problems even for short term horizons

Future work:

- Extending the model using the Resource-Task Network
- Applying DA and ML for other parameters (not just for transition time or -cost) and for more complex scheduling optimization problem

Thank you for your attention!