06720: Project Report

Simultaneous Batching and Scheduling in Multistage Multiproduct Processes

---

Mihir Joshi

Sikandar Batra

Paper Introduction

Multistage Batch Scheduling is a class of problems, where the orders of customers are split into a multitude of batches. These batches are processed through a pre-determined number of stages in a sequential manner. Each one of these stages consist of one or more parallel units. The inherent nature of sequential processing of multiple batches enables models to treat batches as separate entities that continuously move through the process. A multitude of researchers have successfully leveraged the structure outlined in previous lines and developed mathematical models that may be less applicable to a diverse set of problems than networks based models but are computationally more effective and efficient. The approach of current MIP models that model batch sequencing can be classified in to the following categories:

- Slot Based approach:
    - The total available production time of each unit is partitioned into numerous ordered slots of unknown length
    - Each individual batch is assigned to a certain ordered slot. With these assignments we are imposing timing and sequencing constraints
    - The time value of the solution is dependent upon the number of ordered slots postulated for an given unit
- Sequence based approach:
    - The sequencing and timing of batches is determined on the basis of sequencing binary variables
    - The approach can further be classified in to the following subsets:
        - Uses global sequencing variables
        - Uses local sequencing variables that are immediate in nature

Lastly a common known network based approach is: Multiple time grid continuous time approach. The approach is based upon a resource task network representation. The approach is presented in papers written by: Castro and Grossmann; Castro et al.

Large Multistage Batch Scheduling problems are usually solved by using hybrid solution techniques. These hybrid solution techniques are presented in papers written by: Harjunkoksi and Grossmann; Maravelias. In both papers listed in the previous line the Multistage Batch Scheduling problem is decomposed into two sub-problems. The two subproblems are best described as a: sequencing sub problem; assignment sub-problem. The paper written by Harjunkoksi and Grossmann uses constraint programming to solve the sequencing sub-problem. The paper written by Maravelias uses specialized algorithms to solve the sequencing sub-problems

Motivation: The usual approach to model multistage batch processes is to view the batching and scheduling as two separate problems. In this ‘usual’ approach the orders are first split into batches on the basis of unit capacities, then these batches are used as input parameters for the model that solves the scheduling problem. This outlined two layer method is only efficient and efficacious if the pre-determined

demand of orders is static and the units working in parallel for each stage have similar capacity. Therefore the two layered method is not viable in the following instances: objective is not just to meet customer demand but also maximize production/sales during given time horizon ; post capacity expansion and retrofit projects pertaining to some units, resulting in dissimilar capacities between parallel units.

The two layered method is not viable for the listed instances as for each instance there is more than one way to split the orders into batches. To model the listed instances the decisions related to batching and scheduling have to be made simultaneously. Moreover in most existing multistage optimization models the base assumption is that the time taken to process batches is irrespective of batch size and is constant. This assumption is not valid for a simultaneous approach where batching is part of the optimization and the time to process is directly dependent on batch size. In order to resolve the limitation and instances stipulated above the paper posits a global sequence MIP model for simultaneous scheduling and batching in multistage multiproduct processes. The MIP model takes into account variable processing times.

The Problem Statement: The objectives of the model is to optimize the following: 1) The batch size and number of batches for every customer order; 2) Batch assignments for every unit; 3) An ordered sequence of batches for every unit. Furthermore, the following are assumptions made during development of the model: 1) Unlimited storage available for intermediates between every stage; 2) Every batch goes through every stage; 3) Preemption is forbidden.

Author's Approach:

- **Sets:** Overall, this model uses 4 sets: I denote the orders with index 'i'; J denote the set of units with index 'j'; K denote the set of stages with index 'k'; L denote the set of batches with index 'l'. Furthermore, subsets were also defined for this system, explained in the table below.

Table 1: Summary of Subsets

Subset	Definition	Subset	Definition
JA(i,j,k)	Units j in stage k which process order i	IL(i,l,ii,ll)	Allowable transitions from (i,l) to (ii,ll)
Li(i,l)	Possible batches l for order i		

- **Parameters:** This paper uses an exact approach, and therefore provides the necessary parameters to reproduce the results. The given parameters are summarized in the table below.

Table 2: Summary of Parameters			
Parameter	**Definition**	**Parameter**	**Data Source**
q(i)	Demand	bmin(j)	Max. batch size
r(i)	Release Time	bmax(j)	Min. batch size
tf(i,j)	Fixed constant of processing time	bhatmax	Max. feasible batch size
tp(i,j)	Proportional constant of processing time	bhatmin	Min. feasible batch size
H	A large number		

- **Variables:** This model makes heavy use of variables. The following table summarizes both the binary variables and non-binary variables used in the model with respect to the aforementioned sets.

Table 2: Summary of Variables			
Binary Variables		**Continuous Variables**	
Z_{il}	Selection of batch l of order i	B_{il}	Size of batch l for order i
		$Bbar_{ilj}$	Size of disaggregated batch l for order i in unit j
X_{ilj}	Assignment of batch l for order i to unit j	T_{ilk}	Finish time of batch l for order i in stage k
		LT_{il}	Slack variable for late completion of batch l
$Y_{i,l,i',l',k}$	Sequencing (i,l) before (i',l')	MS	Makespan

- **Formulation:** Putting the previous information together, the following formulation was made, where the equation number refers to the equation number in the original paper:

Table 3: Summary of Model Formulation

Equation #	Formulation	Significance
3	$\sum_{l \in L_i} B_{il} \geq q_i$	The sum of all batches should be greater than or equal to the demand
4	$bhatmin_i \cdot Z_{il} \leq B_{il} \leq bhatmax_i \cdot Z_{il}$	If a batch is selected, it has upper and lower bounds. Else, forced to 0.
5	$Z_{il} = \sum_{j \in JA_{ik}} X_{ilj}$	If a batch is selected, then it must be assigned to only one unit
6	$B_{il} = \sum_{j \in JA_{ik}} Bbar_{ilj}$	The sum of the disaggregated batches must equal the total batch size
7	$bmin_j \cdot X_{ilj} \leq Bbar_{ilj} \leq bmax_j \cdot X_{ilj}$	The disaggregated batch size must fall within limit of the unit
8	$Z_{i(l+1)} \leq Z_{il}$	If there is no first batch, there will be no second batch
9	$X_{ilj} + X_{i'l'j} - 1 \leq Y_{ili'l'k} + Y_{i'l'ilk}$	Sequencing constraint
10	$T_{i'l'k} \geq T_{ilk} + \sum_{j \in JA_{ik}} (tf_{ij} \cdot X_{ilj} + tp_{ij} \cdot Bbar_{ilj}) - H(1 - Y_{il}$	Subsequent batch time is greater than or equal to previous batch time
11	$T_{il(k+1)} \geq T_{ilk} + \sum_{j \in JA_{ik}} (tf_{ij} \cdot X_{ilj} + tp_{ij} \cdot Bbar_{ilj})$	The time of the same batch at the next stage has to be greater than/equal to time in the previous batch
12	$T_{ilk} \geq r_i Z_{il} + \sum_{k' > k} \sum_{j \in JA_{ik'}} (tf_{ij} \cdot X_{ilj} + tp_{ij} \cdot Bbar_{ilj})$	Enforcing release time after batch is finished in all previous stages
16	$B_{i(l+1)} \leq B_{il}$	Symmetry breaking constraints
17	$Z_{il} = 1$	Need at least 1 batch per order, so first batch is enforced
23	$MS \geq T_{il\|K\|}$	Makespan is greater than time of last unit

24	Min MS	Minimize makespan

- **Overall Concept:**

Example 1: To effectively apply the previously described model, five examples are used. This example considers a plant which needs to fulfill three orders, named ‘A’, ‘B’ and ‘C’. The orders are 30 kg, 40kg and 40 kg, respectively, in size. In order to fulfill these orders, there are two stages with two units in each stage. There are four units which can be used (J1, J2, J3, J4); J1 and J2 are in stage 1, while J3 and J4 are in stage 2. The objective is to minimize makespan.

- **Example 1A:** Given that the order quantities are within capacity, the batches can be assigned as only one batch per order. This example does not consider any disaggregated batches, and minimizes makespan accordingly. This was enforced through subset Li(i,l).
- **Example 1B:** This example considers disaggregated batches as well - in order words, the orders can be split up into multiple batches. Based on the order quantities and size restrictions on the units, there can be, at most, two batches. This example is also solved to minimize the makespan.

The batch size restrictions and processing time restrictions are detailed in the following tables.

Table 4: Summary of Batch Size Restriction for Example 1

K	Unit	bmin(j), kg	bmax(j), kg
1	J1	10	30
	J2	20	40
2	J3	20	35
	J4	25	50

Table 5: Summary of Processing Time Restriction for Example 1

Orders	r(i)	Fixed Term tf(i,j), hours				Proportional Term tp(i,j), hours/kg			
		J1	J2	J3	J4	J1	J2	J3	J4
A	0	2.5	2.0	0.889	2.0	0.083	0.1	0.089	0.08
B	0	2.5	2.0	0.889	2.0	0.083	0.1	0.089	0.08
C	0	2.5	2.0	0.889	2.0	0.083	0.1	0.089	0.08

Lastly, there were three more parameters, defined as follows: bhatmax = 40, bhatmin = 20 and H = 20.

Paper Reproduction

Implementation Details

Implementation Focus: The implementation focus was one reproducing both results from example 1.

Coding Environment: All of the required coding was carried out in GAMS. Example 1A and 1B were split up into two different files.

Implementation Steps: All of the parameters were individually coded into the GAMS document from the paper. Thereafter, each of the continuous and binary variables were defined, followed by the definition of the constraints and objective function. For example 1A, subset Li(i,l) was only defined in terms of l = 1 for all orders. In example 1B, subset Li(i,l) was only defined in terms of l = 1,2 for all orders. Constraints structured as $a \leq x \leq b$ were broken into two individual constraints as such: $a \leq x$ and $x \leq b$.

Device Capabilities: It must be noted that while the authors of "*Simultaneous Batching and Scheduling in Multistage Multiproduct Processes*" coded the model into GAMS, both models used CPLEX as the solver. The original author used CPLEX 10.1 on a Pentium 4 2.80 GHz, 512 MB GB RAM device, while the reproduced model used CPLEX 12.10 on a 16 GB RAM device.

Challenges Faced

Challenge #1 - Determining the Appropriate H value: In the context of this model, the value of H, which is supposed to be a sufficiently large number much like a big-M formulation, played a vital role in terms of the objective value. While this value did not impact the batch sizes or the corresponding assignment to a unit in each stage, it drastically impacted the time variable T(i,l,k). Based on the provided information, along with some trial and error, it was found that the best value to use was 20. This value sufficiently relaxed the time constraints, but did not over-relax the constraints either.

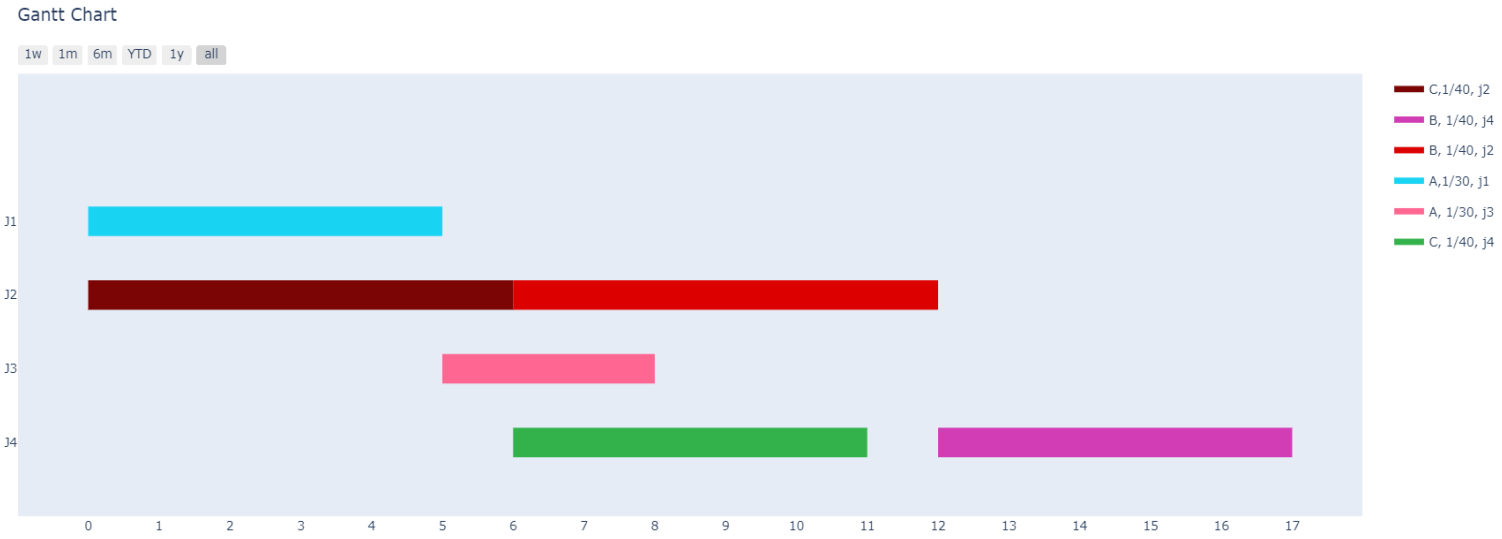
Challenge #2 - Enforcing logic constraints: The subset IL(i,l,ii,ll,k) was defined with respect to logic constraints provided in the paper; however, initially these logic constraints were not being enforced by GAMS. Thus, in order to enforce the logic constraints, IL(i,l,ii,ll,k) was removed as a condition and then solved. Once the error was found, IL(i,l,ii,ll,k) was added back into the equation.

Challenge #3 - Necessary definition of subset: As there are no constraints on any of the orders or batches, initially the subsets were not defined. However, this proved to be unfruitful as the incorrect objective value was obtained. By defining and enforcing the subsets, the objective value approached the example 1 value.

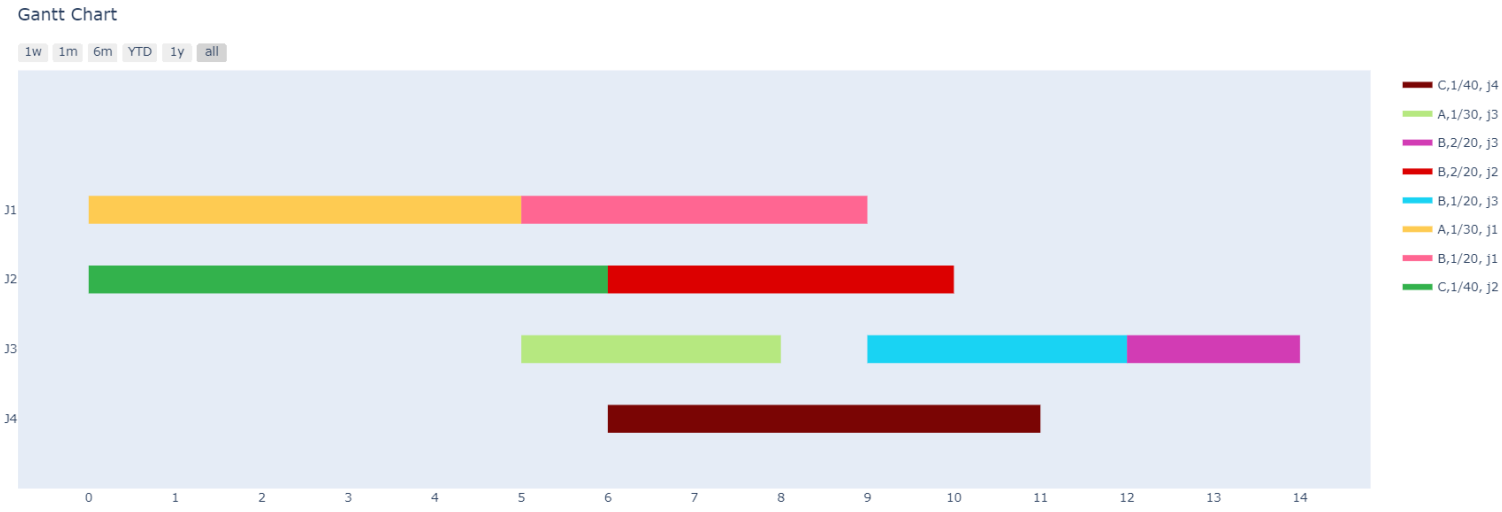
Results

- **Optimal Makespan:** The first set of results to reproduce are the makespan for example 1A and 1B, as done in the original paper:
    - **Example 1A:** The makespan was calculated to be 17.20 hours, agreeing with the paper.
    - **Example 1B:** The makespan was calculated to be 14.49 hours, agreeing with the paper.

- **Production Schedule/Gantt Chart:** The second set of results to be reproduced are the schedules for each of the cases. Below are the optimal schedules.
    - **Example 1A** (Sequential Approach):



    - **Example 1B** (Simultaneous Approach):



Discussion & Conclusion

The results displayed in the previous selection are an exact reproduction of example 1 (Maravelias Paper), proving the reproducibility of the model. The model strived to reproduce the values for each of the binary variables and continuous variables - these are summarized in the Gantt Chart. Below, the model statistics are discussed:

Table 6: Model Statistics

Characteristics	Example 1A	Example 1B
No. of equations	214	265
No. of variables	128	224
Computational time (s)	0.085	0.203
Makespan (hours)	17.20	14.488

From Table 6, the model used to replicate the results was very efficient as it had a low computational time. As Example 1A only allows one batch per order, it makes sense that it has fewer equations and variables, thereby resulting in a lower computation time. On the flip side, it was analytically calculated that there could be at most two batches per order in Example 1B, resulting in Example 1B being a larger problem, and therefore requiring a long computational time.

However it must be noted that the problem solved in example 1 was small in size and the adapted model was a simplified version of the original model. A significant challenge faced in the execution of the model was determining the H parameter as it is highly dependent on the specific characteristic of the problem being solved. The attached files use an H value of 20, as this value is larger than the makespan. However, if an incredibly large H value were to be used, say 100000000, then the model returns a makespan of 6.669 hours. Upon further investigation, it is found that each order is broken down into two batches of 20 kg. A lapse in the model allows for all the batches to be scheduled on units J2 and J3, with the production happening concurrently. Obviously, this does not make sense from a physical standpoint, stressing the importance and accuracy of using 20 as the H value.

The original model is highly versatile and can cater to multiple case scenarios and objectives like: minimizing lateness/earliness cost; maximizing profit; taking into account changeover costs; generating production schedules that are balanced in terms of variety of products being produced. A major limitation of the original/adapted models is that they assume the storage space of intermediaries between stages is infinite. This assumption results in solutions that are unrealistic in an industry setting. Similarly, this model assumes a zero-wait policy, resulting in a smaller objective value. In order to improve, this model can make sure of inventory constraints and forced idle times between batches to make the model more realistic in nature.

It is important to note that the core formulation, while effective for determining batch sizes, disaggregated batch size and time, it could be improved. Specifically, the time variable values can be further improved. The reason this is discussed is because even though order A and order C only make use of one batch, the time values are still calculated for a second batch in each of these orders. This indicated that the time values were not forced to zero after the batch size was determined to be zero. In order to further improve this, inequality constraints can be added relating the batch size to the time variable. In doing so, if the batch size is found to be zero, then the time value can be forced to zero as well. This concept can be extended to the Y variable as well, where it takes a value of 1 even though the current or following batch has a size of 0 kg.

That being said, the model used to replicate the results was an adaptation of the original model presented in the Maravelias Paper, where only constraints which apply to example 1 were implemented. Specifically, this adaptation only contained the constraints listed in section 3 of the paper (the core formulation), and did not implement any of the extensions. Therefore the generalizability, versatility and efficiency of the adapted model is profoundly lower than the original model. To see the list of unused constraints, please refer to Table 7 in the Appendix; some constraints that were not part of the adapted model but are critical to the efficacy of the original model are discussed.

That being said, there are some ways in which the model can be improved and extended. In terms of improvement, the model can make use of Big-M constraints, which will allow the model to maintain its MIP status but still for the time variable values to zero. Else, binary Z(i,l) can be multiplied by the time constraints and the Y variables, but this will result in a MINLP model, which carries its own intricacies.

In terms of extensions, a big part of scheduling is to also account for backlogs and inventory. Backlogs refers to the production facility being behind schedule in terms of fulfilling orders. Thus this model can account for backlogs as well, making it a more robust formulation. Similarly, the model can set upper limits on interstage storage, which will have an impact on the optimal objective value, but will make the model more realistic. Furthermore, while this was not implemented, the model made use of due times. This aspect of the model can be further exploited - with known due times, the model can be expanded to accommodate longer term scheduling of the facility.

Overall, this model is very unique in nature, and has incredibly diverse opportunities in terms of applications. It can be extended through methodologies explained previously, but is very strong as it stands. Finally, this project is considered a success, as each of the results were exactly recreated accurately. The code associated with the presented results are also provided - note, two GAMS files are provided; one for each example.

References

Sundaramoorthy, A.; Maravelias, C. T. Simultaneous batching and scheduling in multistage multiproduct processes. Ind. Eng. Chem. Res. 2008, 47, 1546−1555.

Appendix

Table 7: Summary of Unused Constraints

Equation #	Formulation	Significance
13	$T_{ilk} \leq d_i Z_{il} + \sum_{k' > k} \sum_{j \in JA_{ik'}} (tf_{ij} \cdot X_{ilj} + tp_{ij} \cdot Bbar_{ilj}) + LT_{il}$	Enforcing due time of each batch at each stage. d_i is the due time of each order. LT_{il} is a positive penalty variable if an order is satisfied after a predefined due date. Also serves as the upper bound for the finish times of a batch at each stage.
14 and 15	$X_{ilj} + X_{ilj'} \leq Z_{il} \; \forall \, i, l \in L_i, (j, j') \in FP$ $X_{ilj} = 0 \; \forall i, l \in L_i, \; j \in JF_i$	Used to enforce that forbidden assignments and paths are not violated. FP is a subset that consists of the forbidden paths for orders between j and j'. JF_i is a subset consisting of the forbidden units for order i.
20	$min \sum_{i, l \in L_i} LT_{il}$	The original model is very versatile. It can cater to a number of objective functions. This objective function is used to minimize the lateness cost
21	$min \sum_{i, l \in L_i} (d_i - T_{il\|k\|})$	This is an objective function used to minimize the earliness cost
22	$min \sum_{i, l \in L_i, j \in JA_i} (c^F_{ij} X_{ilj} + c^p_{ij} Bbar_{ilj})$	This is an objective function used to minimize the processing cost
25	$max \sum_{i, l \in L_i} \pi_i B_{il} - \sum_{i, l \in L_i, j \in JA_i} (c^F_{ij} X_{ilj} + c^p_{ij} Bbar_{ilj})$	An objective function used to maximize profit. π_i is the price of product associated with order i.
3a	$q_i \leq \sum_{l \in L_i} B_{il} \leq q_i^{max} \; \forall i$	When the objective is to maximize profit, the original model has a tendency to favor the production of a few products. This tendency is not realistic in scenarios, where balanced production schedules are imperative. Eq. 3a is used to resolve this issue. The equation integrates a upper bound -

		q_i^{max} on the production of order i. In this scenario, Eq. 3a replaces Eq. 3.
2b	$l_i^{max} = \left[\frac{q_i^{max}}{bhatmin_i}\right]$	Since the maximum number of batches is directly dependent on the upper bound of production, Eq. 2a replaces Eq. 2 when the objective is to maximize profit but retain a balanced production schedule.
9a and 9b	$\sum_{i',l' \in L_i} W_{i'l'ilj} + XF_{ilj} = X_{ilj} \; \forall i, l \in L_i, j$ $\sum_{i',l' \in L_i} W_{ili'l'j} + XL_{ilj} = X_{ilj} \; \forall i, l \in L_i, j$	Constraints used in scenarios where the model needs to enforce sequence dependent changeovers costs. Eq. 9a, 9b,9c, 9d replace Eq. 9 in this case scenario. $W_{ili'l'j}$ are local sequence binary variables that replaces the global sequence binary variables, $Y_{ili'l'jk}$. $W_{ili'l'j}$ are active only for adjacent batches that are assigned for a specific unit j. XF_{ilj} is a positive variable that assigns the first batch (*i,l*) of sequence in unit j. XL_{ilj} is a positive variable that assigns the last batch (*i,l*) of sequence in unit j.
9c and 9d	$\sum_{i,l \in L_i} XF_{ilj} \leq 1 \; \forall j$ $\sum_{i,l \in L_i} XL_{ilj} \leq 1 \; \forall j$	Constraints used in scenarios where the model needs to enforce sequence dependent changeovers costs. The constraints imply that at most there can only be one first/ last order in the sequence of each unit j.
10b	$T_{i'l'k} \geq T_{ilk} + \sum_{j \in JA_{ik}} (tf_{ij} \cdot X_{ilj} + tp_{ij} \cdot Bbar_{ilj} + tch_{ii'j} \cdot W_{ili'l'j}$ $- H(1 - \sum_{j \in JA_{ik}} W_{ili'l'j})$	Constraint used in scenarios where the model needs to enforce sequence dependent changeovers costs. $tch_{ii'j}$ is a parameter that represents change over time to transition from order i to i' in unit j. Replaces Eq. 10