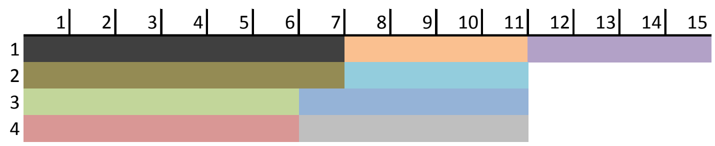
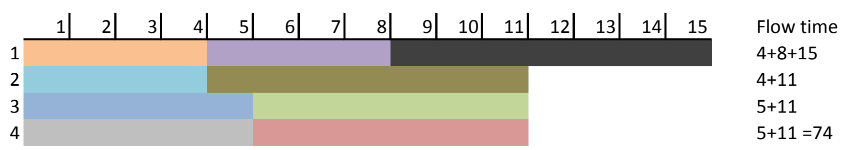
Parallel machines

- We assume, that the machines are similar to the extent that they can be used to process all jobs
- However, machines do not need to be equally fast for the MILP optimization, and the differences in speed may be job specific
- Unlike basic single machine case, makespan is also relevant criterion for parallel machines
- c_{max} (makespan) is minimized quite well by sorting in LPT order and allocating evenly (longest job first to the shortest schedule, but not always:

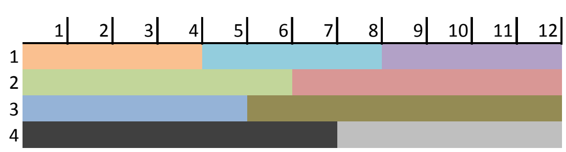


Parallel machines

- In a second step the jobs can be ordered for each machine with single machine rules
- Total flow time for the previous example is optimized using SPT rule:



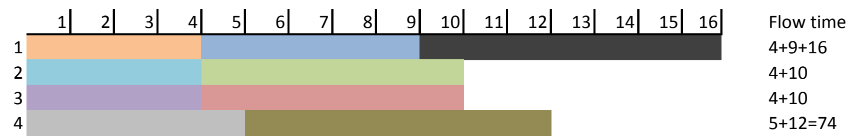
- In this case c_{max} can be optimized using MILP optimization:



Flow time: 76

Parallel machines

- Procedure, in which jobs are ordered with a simple rule and then allocated evenly to machines also usually works well
- In fact, total flow time is optimized using SPT rule. Previous example:



- Due date related criteria can not be optimized using simple rules, but a good result is usually obtained using the described two-phase procedures

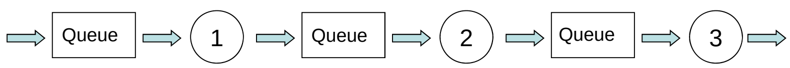
Parallel machines – MILP model

- MILP formulation allocates jobs *i* to machines *k* in order *j*. The result is realized in binary y_{ijk} values
- If machines are not similar, durations V_{ik} can be used for each job-machine combination (high value for V_{ik} for unfit machine-job combinations)
- Model minimizes total tardiness, but any objective is possible with minor modifications

$\text{Min } \sum_{\forall k} \sum_{\forall j} f_{jk}$		Total tardiness
$c_{0,k} = 0$	$\forall j, k$	
$c_{jk} = c_{j-1,k} + \sum_{\forall i} P_i y_{ijk}$	$\forall j, k$	Finishing time
$c_{jk} - \sum_{\forall i} D_i y_{ijk} \le f_{jk}$	$\forall j, k$	Tardiness determination
$\sum_i y_{ijk} \le 1,$	$\forall j, k$	Job order
$\sum_k \sum_j y_{ijk} = 1,$	$\forall i$	Each Job is executed once
$y_{ijk} \in \{0,1\},$	$\forall i, j, k$	
$f_{jk} \ge 0,$	$\forall j, k$	

Flow shops

- A flow shop is a serial production system where the process flow of all products (orders) is similar
- Queuing is allowed (unlike in lines) and jobs can pass each other, i.e. processing order of jobs does not need to be the same at all processing steps
- If jobs are processed in the same order at each step, we have a permutation schedule (there are N! permutation schedules for N orders)
- Optimal schedule is not necessarily a permutation schedule, but usually the best permutation schedule is not far from optimum.
- Flow shop is a common and efficient arrangement for production of product families with a lot of product variability



Two machine flow shop – Johnson's algorithm

- Johnson's algorithm minimizes makespan of a two machine flow shop as follows:
- Job i precedes j in schedule if

  min $\{P_{i1}, P_{j2}\}$ <= min $\{P_{i2}, P_{j1}\}$
- Procedure:

1. Find min i $\{P_{i1}, P_{i2}\}$
2. If smallest value is on machine 1, schedule is filled from beginning
3. If smallest value is on machine 2, schedule is filled from end
4. Return to step 1 until all jobs are scheduled

Idea: Because first machine is running all the time anyway, the schedule is extended only if the second machine waits. By scheduling shorter job first on the first machine, this waiting is minimized.

Two machine flow shop – Johnson's algorithm

Example:

Job *i*	**1**	**2**	**3**	**4**	**5**
P_{i1}	3	5	1	6	7
P_{i2}	6	2	2	6	5

Step	**Jobs left**	**Position**	**Schedule**
1	1,2,3,4,5	3 = [1]	3 X X X X
2	1,2,4,5	2 = [5]	3 X X X 2
3	1,4,5	1 = [2]	3 1 X X 2
4	4,5	5 = [4]	3 1 X 5 2
5	4	4 = [3]	3 1 4 5 2

Flow shop optimization

Flow shops with more than 2 machines can not be optimized with simple rules

Optimization model is obtained from Manne's single machine model by:

- Adding indexes for the machines
- Determining order of machines in the process

In this model tardiness is only counted for the last step.

IP model, $k,..,K$ is machine index

$$
\begin{array}{ll}
\text{Min } \sum_{\forall i} f_i & \\
t_{iK} + P_{iK} - D_i \le f_i, & \forall i \\
t_{ik} \ge 0,\ f_i \ge 0, & \forall i, k \\
t_{ik} + P_{ik} \le t_{ik+1}, & \forall i, k \in \{1,..,K-1\} \\
My_{ijk} + (t_{ik} - t_{jk}) \ge P_{jk}, & \forall i \in \{1,..,I-1\}, j \in \{i+1,..,I\}, k \\
M(1 - y_{ijk}) + (t_{jk} - t_{ik}) \ge P_{ik}, & \forall i \in \{1,..,I-1\}, j \in \{i+1,..,I\}, k \\
y_{ijk} \in \{0,1\}, & \forall i, j, k
\end{array}
$$

Flow shop – permutation model

Optimization model is obtained easily by adding a constraint to the previous model that forces values of y_{ijk} to be equal $\forall k$

or

From Wagner's single machine variation as follows:

- Indexes are added to machines
- Job order on machines and between machines so that there is no overlap

$$\text{Min } \sum_{\forall j}\left(t_{jK} + \sum_{\forall i} P_{iK} y_{ij}\right)$$

Sum of flow times (all jobs available at $t = 0$)

$$t_{jk} \ge 0, y_{ij} \in \{0,1\}, \qquad \forall ijk$$

$$\sum_i y_{ij} = 1, \qquad \forall j$$

$$\sum_j y_{ij} = 1, \qquad \forall i$$

$$t_{jk} + \sum_{\forall i} P_{ik} y_{ij} \le t_{j+1,k}, \qquad \forall j \in \{1,..,I-1\}, k$$

Job order on machines

$$t_{jk} + \sum_{\forall i} P_{ik} y_{ij} \le t_{j,k+1}, \qquad \forall j,k \in \{1,..,K-1\}$$

Job order between machines

Flow shop example

We have five jobs and a four machine flow shop

Example data:

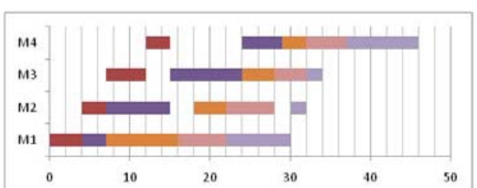
Job i	P_{i1}	P_{i2}	P_{i3}	P_{i4}	D_i
1	4	3	5	3	17
2	6	6	4	5	35
3	8	2	2	9	46
4	3	8	9	5	25
5	9	4	4	3	23

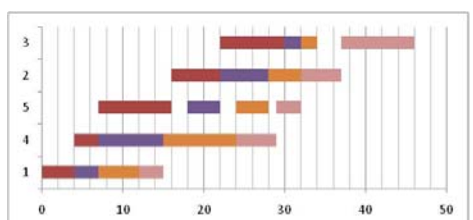
Various optimization criteria are relevant:

- Total taridness
- Max tardiness
- Makespan
- Total throughput time
- Utilization rate

Below permutation schedule for minimization of total tardiness

Note! Colors are not equal in the two views

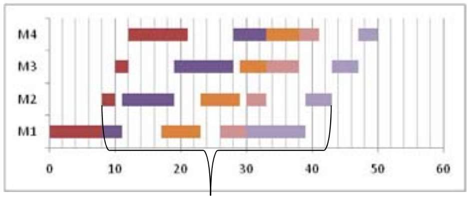


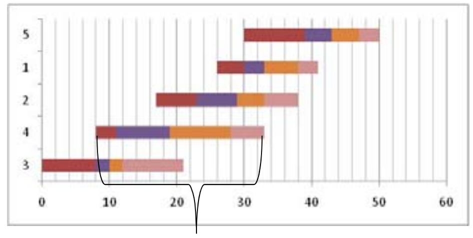


Flow shop example (permutation schedules)

Combined Utilization and throughput time minimization

Weights: Throughput time 10; Utilization 0

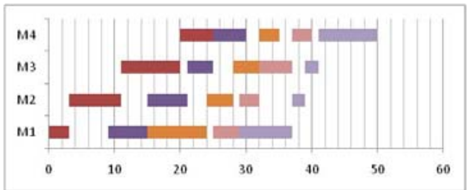


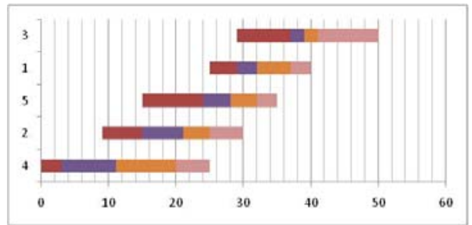


Utilization

Throughput time

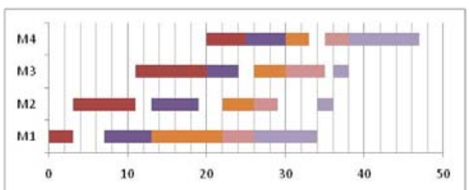
Weights: Throughput time 9; Utilization 1

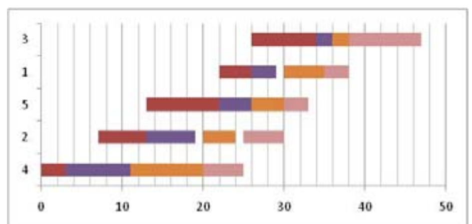




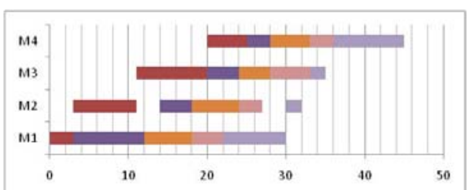
Flow shop - example

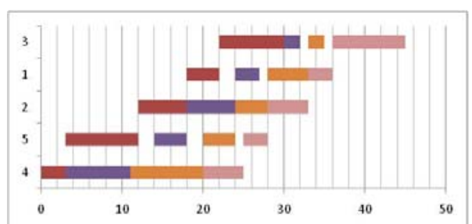
Weights: Throughput time 8; Utilization 2





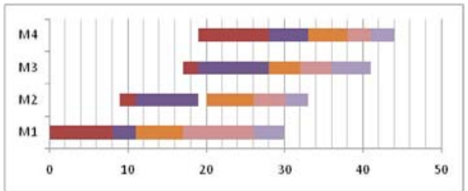
Weights: Throughput time 5; Utilization 5

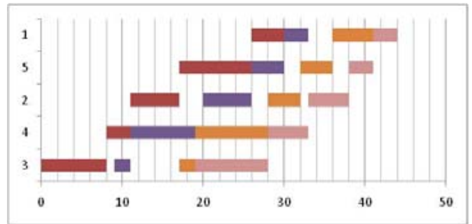




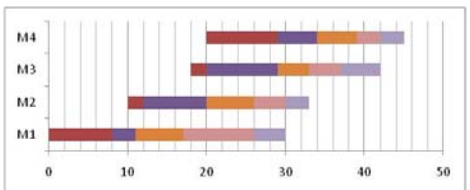
Flow shop - example

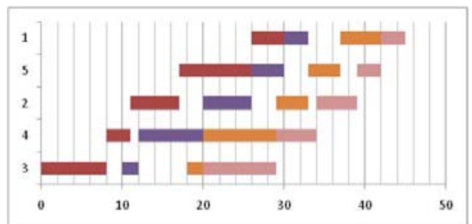
Weights: Throughput time 2; Utilization 8





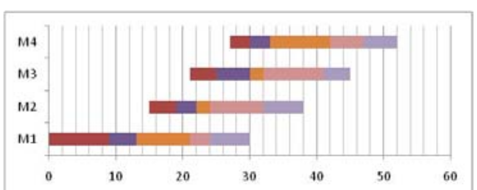
Weights: Throughput time 1; Utilization 9

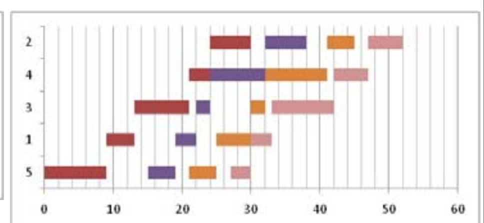




Flow shop - example

Weights: Throughput time 0; Utilization 10



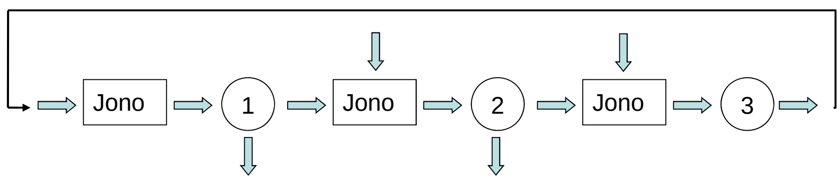


Open Flow shop and re-entrant Flow shop

- In an open flow shop processing may start from any step and stop at any step – flowing order stays the same
- In a re-entrant Flow shop orders may return to the same machines

Optimization model is the same as for other flow shops, but:

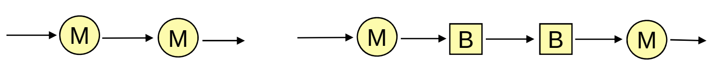
- Processing times are set to zero before the first processing step and after the last processing step
- Re-entering jobs are handled like new orders and a precedence constraint is defined between orders. This of course works for a single machine, too.



Production line

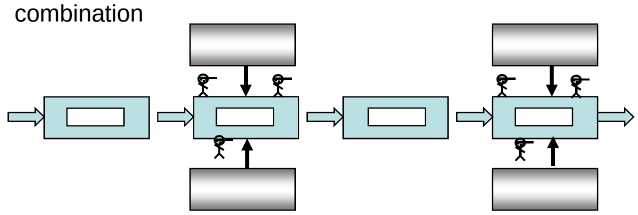
A production line consists of serial work stations between which there is no queuing or only a limited buffer

- Work is allocated to the stations
- Work at the stations is standardized, specialized, and efficient
- Learning is fast
- Resource utilization is high – only one set of equipment of a type is required
- Materials and tools can be located within reach of the worker
- Material delivery to one point only
- Job transfer between stations is required
- Vulnerable to disturbances



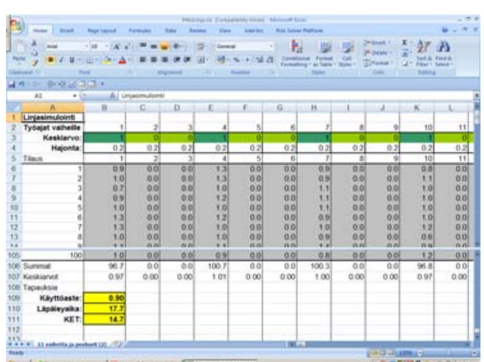
Synchronous and asynchronous lines

- In ***synchronous lines*** jobs move at the same time
  - The main objective is to balance work content between stages
- In ***asynchronous lines*** workpieces may move forward at different times
  - Allows processing time variation compensation, especially if buffers are used
  - Downside is longer throughput time and WIP and longer line if buffers are used
  - Line length limits use of buffers with large products
  - Line pressure is lower than with synchronous lines and conveyors
  - Undermanned asynchronous line and moving workers is a popular combination



Asynchronous line simulation using Excel

- Asynchronous line is mainly evaluated according to line utilization, in practice high output or low waiting are desirable
- Waiting occurs if line is blocked or job in previous stage is not finished (starving)
- Throughput time is not a very relevant criterion, as not much can be done about it after the line is built

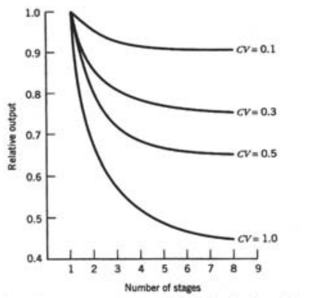


”LineSimulator.xls”

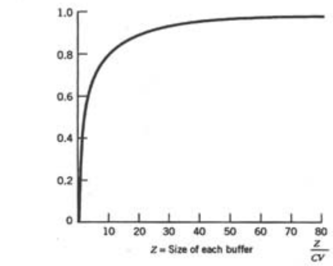
Esko Niemi

Asynchronous line simulation

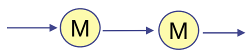
- If stages are equal, capacity of line depends on line length, processing time variation and buffer sizes
- Here cv = processing time standard deviation / mean processing time
- Stages are in balance on average
- Jobs are always available in the beginning of the line

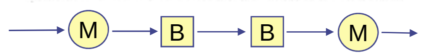


Effect of random processing time in balanced, unbuffered lines.



Proportion of lost output recovered by buffering in balanced lines.

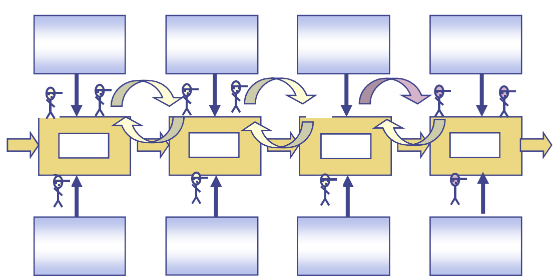




Esko Niemi

Asynchronous line – moving workers

- If products are large, buffers may be out of question
- Processing time variation can be balanced by moving workers
- This requires definition of some roles for workers or active management
- Causes losses like congestion due to non-optimal number of workers
- Differences in workers may be utilized, for example: only the best move

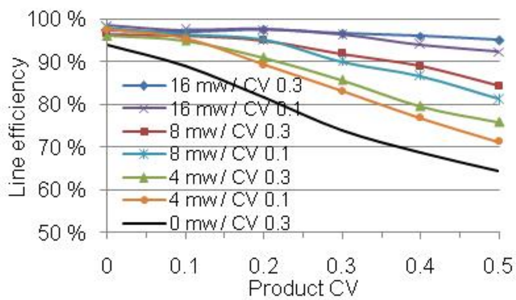


Esko Niemi

Asynchronous line – moving workers

Simulation results for 4 station line:

- About 80 % (90 %) efficiency can be reached when product variation (processing time) CV is 0.3 (0.2)
- In these results a significant congestion loss is involved
- The share of moving workers is 16 – 33 %
- One can assume, that the specialization, logistics and other benefits are greater than losses



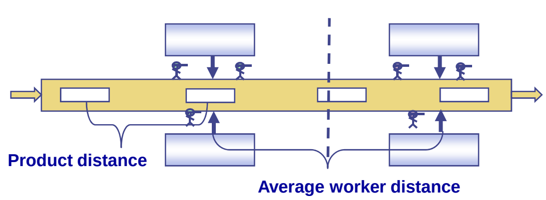
Example

- Totally 48 workers and no congestion loss
- mw = moving workers
- CV 0.1 and 0.3 indicate worker productivity differences

Esko Niemi

Conveyor line – large products

- Line is constantly moving
- Product transfer does not require any resources
- Either workers or floor must move with product
- A buffer effect is created by locating products on the line tighter than workers
- Product sequence rules can be used to limit required buffer size

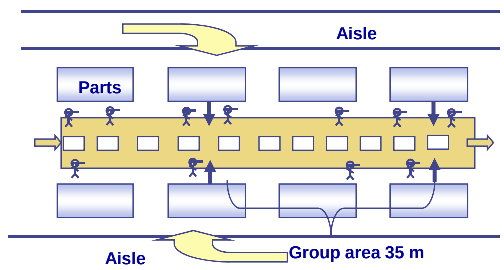


Esko Niemi

Truck cabin assembly conveyor line

- Conveyor speed ca. 1.5 m/min, cabin distance 5 m => cycle time ca. 3.3 min
- Group area 35 m => about 7 cabins in the area
- Large product variation
- Balancing with product mix, too
- Only some main parts are sequence controlled, others standard from stock

- Line speed always the same, capacity is adjusted with flexible working time:
- 2 shifts +/- 300 h/y
- Line length 450 m
- Throughput time 5 h



Esko Niemi

Asynchronous line optimization

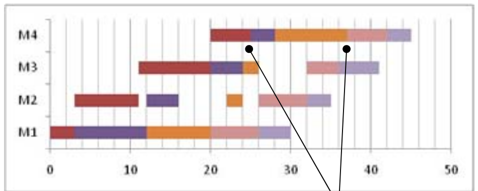
- Relevant decision variable is job order
- Previous flow shop models are appended with a constraint that prevents starting of processing of a job before the processing of the previous job on the next station has started – i.e. the next machine (or buffer position) must be empty before product can move
- Buffer positions are modeled as machines with 0 processing time

- **IP optimization model**

Min $t_{IK} + \sum_{\forall i} P_{iK} y_{iI}$		Completion of last job
$t_{jk} \geq 0, y_{ij} \in \{0,1\},$	$\forall ijk$	
$\sum_{i} y_{ij} = 1,$	$\forall j$	
$\sum_{j} y_{ij} = 1,$	$\forall i$	
$t_{jk} + \sum_{\forall i} P_{ik} y_{ij} \leq t_{j+1,k},$	$\forall j \in \{1,..,I-1\}, k$	Job order on machines
$t_{jk} + \sum_{\forall i} P_{ik} y_{ij} \leq t_{j,k+1},$	$\forall j,k \in \{1,..,K-1\}$	Job order between machines
$t_{j+1,k} \geq t_{j,k+1},$	$\forall j \in \{1,..,I-1\}, k \in \{1,..,K-1\}$	Line condition

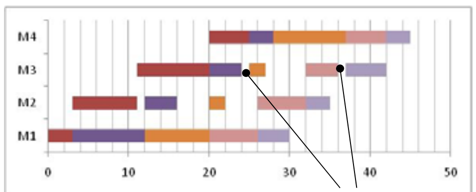
Line example

- Same case as a flow shop and (no-buffer) line
- Makespan is optimized
- Here job order is the same (permutation schedules)



Flow shop

Waiting – next job on machine 3 can start immediately

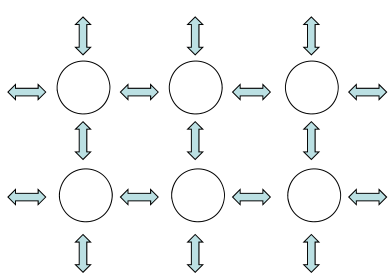


Line

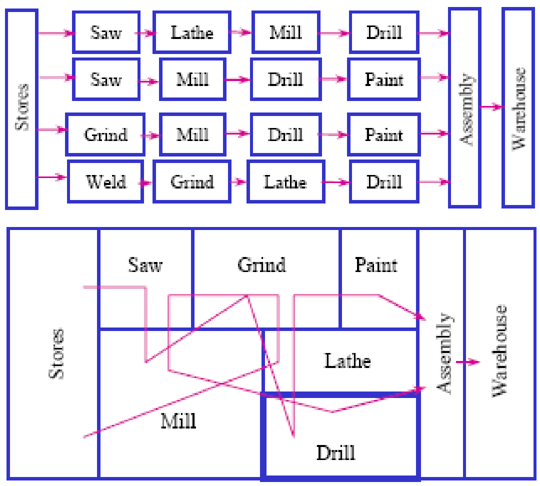
Blocked – job waits on machine 3 (gap in schedule), until machine 4 is freed and next job can start

Job shop

- In a job shop job routing is not costrained in any way and any recipe can be followed
- Typical properties:
  - Product variation is large
  - Process and processing time variation are large
  - Process stages number and order vary
  - Processes are flexible
  - Value adding time in relation to throughput time is small, because of the variation and set-up times (-> batches)



Flow shop vs. Job shop



Flow shop
- Similar processes for product families
- Short transportation distances
- Visual control
- Short throughput times

Job shop
- Control is based on queuing priorities
- Difficult to balance
- Long transportation times
- Long throughput times
- Utilization rate of skilled labor is high

Kuvat Hopp et al. / Factory physics

Job shop

Optimization model is obtained from the flow shop model with minor changes:

- Job's processing order is determined with recipe specific constraints
- Set O represents processing steps (machines) o and p of all jobs i that are consecutive in the chain
- If job i does not use machine k, set $t_{ik} = 0$
- In the example total tardiness is minimized and in order to keep notation simple, all steps are examined for tardiness and not just he last one

$$
\begin{array}{ll}
\text{Min } \sum_{\forall i} f_i & \\
t_{ik} + P_{ik} - D_i \le f_i, & \forall i, k \\
t_{ik} \ge 0,\ f_i \ge 0, & \forall i, k \\
t_{io} + P_{io} \le t_{ip}, & \forall io, ip \in O \\
My_{ijk} + \left(t_{ik} - t_{jk}\right) \ge P_{jk}, & \forall i \in \{1,..,I-1\}, j \in \{i+1,..,I\}, k \\
M\left(1 - y_{ijk}\right) + \left(t_{jk} - t_{ik}\right) \ge P_{ik}, & \forall i \in \{1,..,I-1\}, j \in \{i+1,..,I\}, k \\
y_{ijk} \in \{0,1\}, & \forall i, j, k
\end{array}
$$

Prioritizing

- Because production control is difficult, scheduling in practice usually resorts to queue prioritizing
- This means choosing the next job from those available with a ”rule of thumb”
- Typical rules:
  - SPT Shortest Processing Time
  - EDD Earlies Due Date
  - FIFO (FCFS) First In (Come) First Out (Served)
  - S/RO Slack per Remaining Operation
  - Covert Cost of delay over remaining processing time
  - CR Critical Ratio – Slack per Total Work Remaining or machine loading rate
  - LTWK Least Total Work
  - LWKR Least Work Remaining
  - RANDOM
  - WINQ Work in Next Queue

FM systems

- FM systems (FMS, Flexible Manufacturing System) are automatic centrally controlled machining systems that consist typically of NC machine tools and supporting machines like washing machines etc., central workpiece changing and storing system (AS/RS, Automatic Storage / Retrieval System) and loading and unloading stations
- The purpose is to increase utilization rate and weekly running time of machines with automation
- Throughput time is secondary objective in the sense that long daily running time guarantees a high turnover rate of orders anyway
- Use of flexible technology makes manufacturing a large variety of products possible
- Set-ups are mostly external and therefore small batch sizes or one-off manufacturing is possible

FM systems' properties

From the production control point of view FM systems are job shops with the following features

- Alternative parallel machines, but
- Similarity of machines is affected by tool set-up
- AS/RS is a scheduled or at least loaded resource in the system
- Number of pallets is limited
- Set-ups are external and they mainly consist of change of fixtures on the pallets
- Change of fixtures takes considerable time
- Workpiece change and fixture change are done manually during manned shifts

FM system control constraints

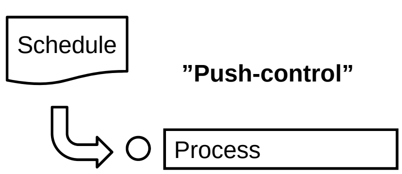
- Tool set-up must allow execution of manufacturing program
- Fixture set-up must allow execution of manufacturing program
- There should be enough work for the unmanned shifts
- Fixture and workpiece change must be executed during the manned time
- AS/RS system must not become too limiting a bottleneck
- In addition, normal manufacturing targets must be met: required parts must be manufactured within the given time

FM system planning and control

- The products to be manufactured are selected based on their relative fit and economy
- In the medium term the products are determined in the master production schedule
- In the short term the job selection can be formulated as an optimization problem with the following constraints
  - Tool set-up is such that the jobs can be executed
  - There is sufficient work (loaded pallets) for the unmanned shifts
  - Workpiece loading and fixture changes can be done during the manned shifts
- System control is based on prioritizing, where the AS/RS system serves machines based on priorities set to jobs and machines
- Operator selects jobs to be loaded on pallets from the job queue and calls for pallet and material to the loading station
- Unloading has its own job queue

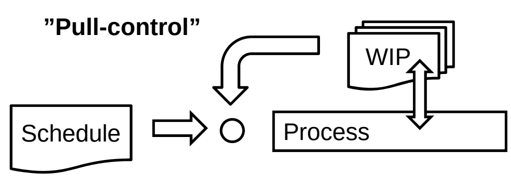
CONWIP (Constant WIP) control

- Classical aggregate planning pushes orders to the system without consideration of the system status



- Jobs are easily congested due to poor planning and disturbances

- In Conwip systems jobs are released only if system status (WIP) allows it



CONWIP systems

- Work in process is controlled with the number of Kanban cards or similar book keeping in the EDP system
- Same effect takes place in line production, where buffers limit the amount of WIP

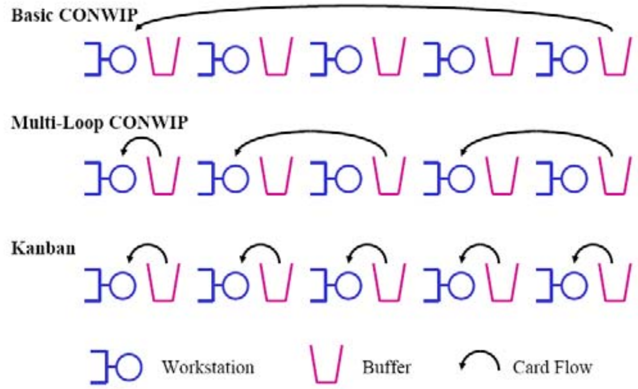


Figure: Hopp et al. / Factory physics

Bottleneck control

- Because production bottleneck constrains the production flow of the whole system, it requires special attention
- Bottleneck can be handled as a single machine, which simplifies control
- More waiting and large batch sizes are allowed in the bottleneck than elsewhere in the production process
- In practice the bottleneck tends to travel, and the situation is more complicated

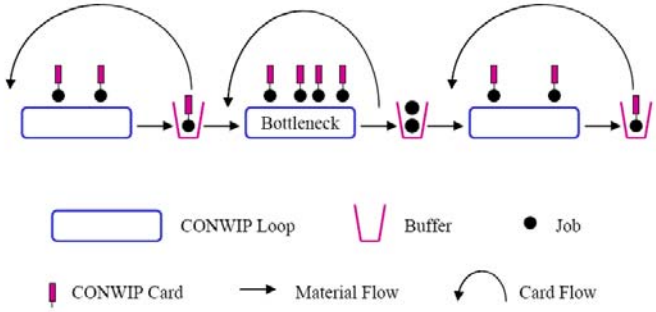


Figure Hopp et al. / Factory physics