Published in Proc. 2012 IEEE Conf. on Mechatronics and Automation, Chengdu, China, August 5-8, 2012, pp. 1550-1556

Manufacturing algebra. Part II: aggregation, control and simulation

Enrico Canuto[1], Manuela De Maddis[2]
[1]*Dipartimento di Automatica e Informatica,*
[2]*Dipartimento di Ingegneria Gestionale e della Produzione*
Politecnico di Torino
Torino, 1 1029 Italy
{enrico.canuto, manuela.demaddis}@polito.it

***Abstract* – A few formal entities for modeling manufacturing processes and factories have been outlined in a companion paper of this conference. The link between a manufacturing model and its implementation in a factory plant has been formulated by ordering object drawing and delivery in a sequence of events, and by scheduling their starting time. Real-time simulation is then viable by providing the production units with a sequence of operations to be performed together with their starting times: simulation may be referred to as the direct problem. Nothing has been said about the synthesis of command sequences from higher-level production orders listing quantities of finished products to be delivered at certain times. The problem, which may be referred to as the inverse problem, is treated in the paper with the help of aggregation/disaggregation concepts and procedures. They allow to formulate the order sequence as a higher-level command sequence dispatched to the higher-level production unit in charge of the whole factory. The higher level command is then real-time disaggregated into lower-level commands (push). The status of the lower-level units is in turn transmitted to higher-level units (pull) for building-up their aggregated status. Simulation results of the factory model driven by the designed real-time control are provided.**

Index Terms - Manufacturing Algebra, aggregation, production control, control unit.

I. I. INTRODUCTION

A few mathematical entities for modelling manufacturing processes (objects, types, manufacturing operations (MO)) and factories (storage, production and resource units) have been outlined in a companion paper [1]. The link between a manufacturing process and its implementation in a factory plant has been formulated by ordering part drawing and delivery as an event sequence, and by scheduling their starting times. Real-time simulation is viable by providing production units with a sequence of operations to be performed together with starting times: simulation is a direct problem.

Nothing has been said about how to synthesize command sequences from higher-level production orders (the demand) listing quantities of finished product to be delivered at certain times (master production scheduling [2]). The problem that may be referred to as the inverse problem, is approached with the help of aggregation and disaggregation concepts, that allow to formulate the order sequence as a higher-level command sequence to the higher-level production unit in charge of controlling the whole factory. Real-time production control should blend push and pull strategies and data, according to the definition in [2]. Push strategy must disaggregate orders (top-down data flow) down to the factory-level operations. Pull strategy must aggregate the factory-level status up to data compatible with the demand itself (bottom-up data flow) for authorizing implementation.

The aggregation process (Section II) starts from manufacturing operations. Two ways of composing them are strictly related to precedence, series and parallel. The different compositions influence input and output quantities and the resulting cycle time. Real-time implementation of the composition is obtained by scheduling the composition events (disaggregation process). It is thus possible to view the composition as a new operation having its own events, input-output and balance models like elementary operations [1].

Among the possible compositions the 'balanced MO' are of interest, as they are composed in a way that does neither require nor deliver semifinished materials. Their model can be thus simplified, since semifinished quantities and events may be dropped, and the relevant operation is referred to as aggregate MO. Any (balanced) production order can be formulated as an aggregate MO having finished products as output objects and raw materials as input objects. The aggregation process continues by aggregating the set of production units (PU, station) in charge of the aggregated operations as a single aggregated PU. The aggregation process can be organized into several layers, which reduce to a pair in the case study, since all the factory stations are aggregated into a single PU, encompassing the whole factory. Once aggregated, a MO can be real-time disaggregated, since it contains a relocatable schedule of the elementary MO. In turn, the information lost by the aggregated MO (semifinished objects) is recovered by the model of the elementary MO.

The paper concentrates on a four-step design of the aggregated MO. The procedure is demonstrated with the help of the case study (Section III). The design deploys well known problems of production planning and control: product mix planning, scheduling, capacity utilization. All of them may be solved using literature techniques. Examples of solution are provided for the case study. The design result is the set of the feasible MO of an aggregate PU, which latter can be treated

and commanded as an elementary unit. The task of real-time scheduling and routing the elementary MO to the elementary PU is the task of the control unit (CU), taking into account the actual object quantities at the storage units and the actual event occurrence times (pull strategy), because of micro and macro irregularities with respect to the designed MO model. An aggregate PU is the set of elementary PU plus a CU.

A brief mention of the CU discrete-event model and of the control strategies is done in Section IV. Simulation results of the factory model driven by the designed real-time production control are provided and assessed with respect to standard performance indices in Section V.

II. THE AGGREGATION PROCESS

Optimization problems arise in the aggregation process. They will be neither formulated nor generically solved, but only mentioned and manually solved in the case study.

A. Composition of manufacturing operations

The elements of a basis $\mathcal{M}$ can be combined depending on their representation. Event-sequence representations are composed by fixing the starting times $t_w(m)$ of each MO m and adding the time-shifted sequences $\sigma_q\left(m,t_w(m)\right)$ and $\sigma_w\left(m,t_w(m)\right)$ into a single one

$$\underline{\sigma}_q=\sum_m\sigma_q\left(m,t_w(m)\right),\ \underline{\sigma}_w=\sum_m\sigma_w\left(m,t_w(m)\right), \quad (1)$$

where underline denotes aggregation. The procedure providing sequences in (1) is indicated as 're-locatable a priori schedule', or schedule. Up to now no constraint has been mentioned in fixing starting times: for instance, the constraint that the interval between successive starting times $t_w(m_2)>t_w(m_1)$ is larger than the cycle time $\tau(m_1)$ of the former operation.

Precedence among elementary operations entering the composition can be expressed without explicit mentioning time, just through algebraic rules over the input/output representations. Operations can be composed either in series or in parallel. The series imposes an order to operations so that the MO $m=b$, which must employ the object types produced by $m=a$, has to start after a has been completed. The series composition, expressing precedence of a, is denoted by $c=ba$, and applies to the ordered pair $a=\left(\mathbf{u}(a),\mathbf{y}(a)\right)$ and $b=\left(\mathbf{u}(b),\mathbf{y}(b)\right)$. It is defined by $c=\left(\mathbf{u}(c),\mathbf{y}(c)\right)$ where

$$\begin{aligned}\mathbf{u}(c)&=\mathbf{u}(a)+\mathbf{u}(b)-\min\left\{\mathbf{y}(a),\mathbf{u}(b)\right\}\\ \mathbf{y}(c)&=\mathbf{y}(a)+\mathbf{y}(b)-\min\left\{\mathbf{y}(a),\mathbf{u}(b)\right\}\end{aligned}. \quad (2)$$

Series imposes a time constraint to the operation schedule as their starting times must satisfy $t_w(b)\ge t_w(a)+\tau(a)$. As a consequence the cycle time [1] of c must satisfy

$$\tau(c)\ge\tau(a)+\tau(b). \quad (3)$$

When equality holds, (3) is the 'linearity condition'.

The parallel composition is denoted by $c=a+b$ and it is defined by $c=\left(\mathbf{u}(c),\mathbf{y}(c)\right)$ where

$$\mathbf{u}(c)=\mathbf{u}(a)+\mathbf{u}(b),\ \mathbf{y}(c)=\mathbf{y}(a)+\mathbf{y}(b). \quad (4)$$

A parallel composition does not entail any time precedence: start times can be arbitrary and cycle times satisfy

$$\tau(c)\ge\max\left(\tau(a),\tau(b)\right). \quad (5)$$

The latter inequality is referred to as the 'bottleneck principle', since the least parallel cycle time is imposed by the longer operation, i.e. the bottleneck as defined in [2].

The balance vectors $\mathbf{b}(c)$ of the series and parallel compositions are the same and hold

$$\mathbf{b}(c)=\mathbf{y}(c)-\mathbf{u}(c)=\mathbf{y}(a)+\mathbf{y}(b)-\mathbf{u}(a)+\mathbf{u}(b), \quad (6)$$

which is coherent with the loss of information of the balance vector representation. Series and parallel compositions are employed to build up a set of possible schedules.

As an example consider the composition of a,b,c:

$$d=c(a+b), \quad (7)$$

to be adopted later, and implying precedence of the parallel $a+b$ with respect to c. Meaning is that the operation c needs the semi-finished objects produced by a and b. Only the start time of c is constrained and must satisfy $t_w(c)\ge\max\left\{t_w(a)+\tau(a),t_w(b)+\tau(b)\right\}$. The cycle time satisfies

$$\tau(d)\ge\tau(c)+\max\left(\tau(a),\tau(b)\right), \quad (8)$$

and shows relation to max-plus algebra [3].

Of interest are those compositions void of semi-finished quantities in their balance vector, because they leave inviolate the possible stock of semifinished types (the 'crib inventory' in [2]). They are called balanced operations. The balance vectors $\mathbf{b}$ of a basis $\mathcal{M}$ can be composed through linear combinations, that are expressed in a matrix form as $\mathbf{b}=B\mathbf{a}$. The entries $a(m)$ of the vector $\mathbf{a}$ define the so-called Bill-of-Manufacturing-Operations (BOMO) of the complex operation having balance vector $\mathbf{b}$. The entry $a(m)$ denotes the number of times an elementary operation m has to be repeated in the composition. A composition $B\mathbf{a}$ is said to yield a feasible operation if $\mathbf{b}$ is integer and $\mathbf{a}$ is a non negative rational vector. A typical balance problem is to find $\mathbf{a}$ and the raw material quantity $\mathbf{b}_r$ given a desired quantity $\mathbf{b}_f$ of finished products and zero semifinished types. Using the partition of B in Part I [1], the problem can be split in two parts:

1) BOMO computation from

$$\begin{bmatrix}0\\ \mathbf{b}_f\end{bmatrix}=\begin{bmatrix}B_s\\ B_f\end{bmatrix}\mathbf{a}=B_{sf}\mathbf{a}, \quad (9)$$

2) raw material computation given $\mathbf{a}$ from $\mathbf{b}_r=B_r\mathbf{a}$.

At least a solution of (9) exists if $\operatorname{rank}B_{sf}=M$, where M is the dimension of $\mathbf{a}$ and $\mathcal{M}$. The previous rank condition is more restrictive than $\operatorname{rank}B=M$ [1], and requires that each MO $m\in\mathcal{M}$ has at least one output type which is different from the other ones (as in the case study, see Figure 1).

Event sequences and input/output representations of a composition may become rather complicated to be employed. For instance the event sequence of a balanced composition includes all delivery/drawing events of the semifinished types. A simpler model, aggregate MO (MO), is attached.

Given a balanced composition $\mathbf{b}$ of a basis $\mathcal{M}$, the aggregate $\underline{m}$ is defined by a pair of event sequences.

1) The sequence $\underline{\sigma}_q(\underline{m})$ of delivery and drawing events

reports quantities of input and output types only – semi-finished being dropped. Addition of input and output quantities defines the vectors $\underline{\mathbf{u}}(\underline{m})$ and $\underline{\mathbf{y}}(\underline{m})$.

2) The sequence $\underline{\sigma}_w(\underline{m})$ of the start and end events

$$e_w(m,0)=\left[t_w(m),-1\right],\ e_w(m,1)=\left[t_w(m)+\tau(m),1\right], \quad (10)$$

associated to each elementary operation m.

The BOMO equation $\underline{\mathbf{y}}(\underline{m})-\underline{\mathbf{u}}(\underline{m})=B\underline{\mathbf{a}}(\underline{m})$ of $\underline{m}$ shows the quantity of each elementary operation entering the composition. The first sequence $\underline{\sigma}_q(\underline{m})$ allows treating aggregated operations as elementary ones. The second sequence $\underline{\sigma}_w(\underline{m})$ allows disaggregating a MO into elementary operations (push strategy).

B. Aggregated production units and their design

A basic concept in the Manufacturing Algebra is the aggregated production unit (PU for short). An aggregated unit $\underline{p}(c)$ is the PU-like model of a set of elementary units $\mathcal{P}(c)=\{p_0(c),...,p_\pi(c),...,p_{\Pi-1}\}$, that are real-time controlled by a single control unit c and are connected to a set $\mathcal{S}(\underline{p})$ of storage units. To respect the PU model outlined in the companion paper ([1], Section V.B), one must define

1) the set $\underline{\mathcal{M}}(\underline{p})$ of the admissible operations $\underline{m}$,
2) the production state variables $\underline{x}_p(\underline{p},t)$ and $\underline{z}_p(\underline{p},t)$.

The key element is the set $\underline{\mathcal{M}}(\underline{p})$, that is the set of the aggregated and balanced operations obtained by suitable compositions of the admissible operations of the elementary stations $p_\pi(c)$. Given a set $\mathcal{P}(c)$, the construction (or planning) of $\underline{\mathcal{M}}(\underline{p})$ is a fundamental issue in production control design, and is dealt with through four steps, that are further detailed in Section III.

1) Balancing. First is the computation of the BOMO $\underline{\mathbf{a}}(\underline{m})$ of an aggregate (balanced) operation $\underline{m}$ satisfying (9). The solution can be made dependent on some free parameters $\boldsymbol{\lambda}$ of the balance matrix $B(\boldsymbol{\lambda})$, as in the matrix of the case study in [1], Eq. (3). Finding $\boldsymbol{\lambda}$ may be interpreted as a 'product mix planning' problem [2], but unlike [2]the solution must involve scheduling steps.
2) Scheduling. Second is the a-priori (nominal) scheduling of the elementary operations to the production units of the set $\mathcal{P}(c)$, with the constraint that each unit can only process a series of operations. The result is the sequence $\underline{\sigma}_w(\underline{m})$ of the start and end events and an estimation of the cycle time $\underline{\tau}(\underline{m})$ when performed on the aggregated PU. To this end, consider a station $p\in\mathcal{P}(c)$, and denote a series of operations $\{m_1,...,m_\mu,...,m_{M(p)}\}\in\mathcal{M}(p)$ of size $M(p)$ to be scheduled on the unit p with

$$s(p)=\prod_{\mu=1}^{M(p)} m_\mu^{a(\mu)}, \quad (11)$$

where $a(\mu)$ is the number of repetitions of the operation m_μ (an entry of the BOMO $\mathbf{a}(s)$). Assuming linearity conditions (Section II.A), the series working time holds

$$\tau\left(s(p),\boldsymbol{\lambda}\right)=\sum_\mu a(\mu,\boldsymbol{\lambda})\tau\left(m_\mu\right), \quad (12)$$

where $\boldsymbol{\lambda}$ is the vector of free parameters to be optimized. Moreover, since the series operations $s(p)$ that have been scheduled on the various units $p\in\mathcal{P}(c)$ can be performed in parallel, the aggregated cycle time of the paralleled series operation (aggregated operation) denoted with $\underline{m}$ satisfies the bottleneck principle (Section II.A) and holds

$$\underline{\tau}(\underline{m})=\max_{p\in\mathcal{P}(c)}\left\{\tau\left(s(p),\boldsymbol{\lambda}\right)\right\},\ \underline{m}=\sum_{p=1}^{P(c)}s(p). \quad (13)$$

Equation (13) says that a bottleneck unit exists and holds

$$p_{\max}=\arg\max_{p\in\mathcal{P}(c)}\left\{\tau\left(s(p)\right)\right\}. \quad (14)$$

Bottleneck severity can be measured by the PU (capacity) utilization defined as the ratio of the average to the largest cycle time (bottleneck)

$$\underline{\upsilon}(\underline{m})=\left(P(c)\underline{\tau}(\underline{m})\right)^{-1}\sum_{p=1}^{P(c)}\tau\left(s(p),\boldsymbol{\lambda}\right)\le 1. \quad (15)$$

Equation (15) extends the definition of 'utilization' in [2], where the numerator is taken as the job 'arrival rate', to an aggregate PU. In [2] the term 'balanced' is reserved to machines in series with maximum utilization, $\underline{\upsilon}(\underline{m})=1$.

3) Optimization of the utilization. It is a balancing task [2] to make $\underline{\upsilon}(\underline{m})$ to approach unity, by equalizing the cycle times of the series operations $s(p)$. This may be obtained by maximizing $\underline{\upsilon}(\underline{m})$ with respect to free parameter $\boldsymbol{\lambda}$ in the matrix $B(\boldsymbol{\lambda})$ as it will be done in the case study.
4) WIP reduction. The ratio of the manufacturing time over the cycle time may be referred to as the work-in-process (WIP) $w(\underline{m})$ as it marks the number of unfinished operations of an aggregated production unit:

$$\underline{w}(\underline{m})=\underline{T}(\underline{m})/\underline{\tau}(\underline{m})\ge 1. \quad (16)$$

Notice that $\underline{\tau}(\underline{m})$ may be much shorter than the aggregated time $\underline{T}(\underline{m})$, the latter depending on how the series operations $s(p)$ to be performed in parallel are actually sequenced. Thus a sequencing goal is to minimize $w(\underline{m})$. In [2] WIP is defined in terms of the inventory between start and end points of a product routing, but excluding end products. The definition derives from the concept of job. Here the definition just refers to the job logical information, i.e. to aggregated operations.

III. THE CASE STUDY

The case study allows clarifying meaning and construction rules of the aggregate operations, as they play a key role in control design and actuation. Aggregate operations have been defined as the simplified model of MO compositions built over the basis $\mathcal{M}$ of the manufacturing process.

Several criteria may lead to aggregate operations:

1) to provide the requested mix of finished products,
2) to guarantee high utilization of factory capacity.

Here we are simple:

1) four aggregate operations $\underline{m}_j$, $j=0,\ldots,3$ are constructed, one for each finished product $n=7+j$, as in Figure 1,
2) each MO yields an integer quantity p_j of finished products,
3) the input composition of the different finished products $n=7+j$ is made to correspond to different capacity utilizations by selecting the input quantity ratio β_j/γ_j,
4) the design procedure outlined in Section II.A is pursued.

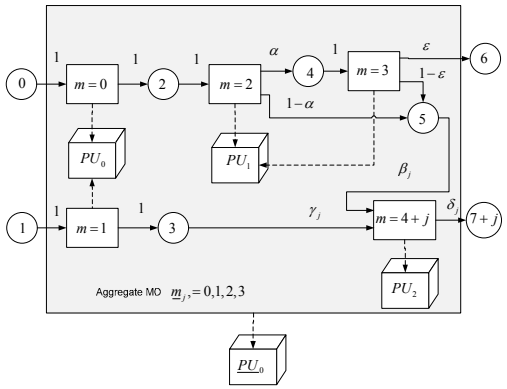

Figure 1 Manufacturing model and aggregated units.

A. Balancing: the BOMO computation

First, the BOMO computation is made. In general, given a desired finished product quantity p_j, the balance equation $\mathbf{b} = B(\boldsymbol{\lambda})\mathbf{a}$ with the balance matrix derived in Part I [1], must be solved for $\mathbf{a}$ with the constraint of zero semi-finished quantities. The balance matrix can be found in [1], Eq. (3). The vector of free parameters for each j is

$$\boldsymbol{\lambda}_j^T = \begin{bmatrix} \beta_j & \gamma_j & \delta_j \end{bmatrix}. \tag{17}$$

Then the raw material quantities $-q(0)$ and $-q(1)$, and the discarded quantity $q(6)$ are solved for. To this end, consider a single finished product $n = 7 + j$, which implies different balance equations $\mathbf{b}_j = B_j(\boldsymbol{\lambda}_j)\mathbf{a}_j$ of reduced size as follows:

$$\begin{bmatrix} -q_j(0) \\ -q_j(1) \\ \hline 0 \\ 0 \\ 0 \\ 0 \\ p_j \\ \hline q_j(6) \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ \hline 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -\gamma_j \\ 0 & 0 & \alpha & -1 & 0 \\ 0 & 0 & 1-\alpha & 1-\varepsilon & -\beta_j \\ 0 & 0 & 0 & 0 & \delta_j \\ \hline 0 & 0 & 0 & \varepsilon & 0 \end{bmatrix} \begin{bmatrix} a_j(0) \\ a_j(1) \\ a_j(2) \\ a_j(3) \\ a(4+j) \end{bmatrix}. \tag{18}$$

Full rank of B_j guarantees the unique solution

$$\mathbf{b}_j(\boldsymbol{\lambda}_j, \mu_j) = \begin{vmatrix} -q(0) \\ -q(1) \\ 0 \\ 0 \\ 0 \\ 0 \\ q_j(6) \\ p_j \end{vmatrix} = \begin{vmatrix} -\frac{\beta_j}{1-\varepsilon\alpha} \\ -\gamma_j \\ 0 \\ 0 \\ 0 \\ 0 \\ \varepsilon\alpha\frac{\beta_j}{1-\varepsilon\alpha} \\ \delta_j \end{vmatrix} \mu_j \cong \begin{vmatrix} -\beta_j \\ -\gamma_j \\ 0 \\ 0 \\ 0 \\ 0 \\ \varepsilon\alpha\beta_j \\ \delta_j \end{vmatrix} \mu_j \tag{19}$$

and

$$\mathbf{a}_j(\boldsymbol{\lambda}_j, \mu_j) = \begin{vmatrix} \frac{\beta_j}{1-\varepsilon\alpha} \\ \gamma_j \\ \frac{\beta_j}{1-\varepsilon\alpha} \\ \alpha\frac{\beta_j}{1-\varepsilon\alpha} \\ 1 \end{vmatrix} \mu_j \cong \begin{vmatrix} \beta_j \\ \gamma_j \\ \beta_j \\ \alpha\beta_j \\ 1 \end{vmatrix} \mu_j, \tag{20}$$

where $\mu_j = p_j / \delta_j$, and the approximations in (20) and in (21) hold if the product $\alpha\varepsilon$ can be kept as negligible, being the product of defective part probability. Alternatively $\alpha\varepsilon = 0$ can be kept as a nominal condition. The entries of the BOMO $\mathbf{a}_j$ must be selected to be nonnegative integers. In practice, the problem arises of finding a symbolic solution as in (21), in the case of complex manufacturing and factory models: a possible solution is to pass through Monte Carlo realizations in the expected range of the free parameters.

B. Scheduling

Second is the definition of an a-priori operation sequence to be achieved in two steps:

1) the series composition of the operations performed on the same production unit p provides intermediate aggregate operations $s_j(p)$ as in (11),
2) composition of $s_j(p)$ provides the assembly plant sequence.

Using the approximation $\varepsilon\alpha \simeq 0$, the series compositions $s_j(p)$ of the elementary operations in Figure 1, that must be performed on each p in agreement with the BOMO solution (20), are the following:

$$\begin{array}{ll} \text{PU } p & \text{series composition} \\ 0 & s_j(0) = m_1^{\gamma_j\mu_j} m_0^{\beta_j\mu_j} \\ 1 & s_j(1) = m_3^{\alpha\beta_j\mu_j} m_2^{\beta_j\mu_j} \\ 2 & s_j(2) = m_{4+j}^{\mu_j} \end{array}. \tag{21}$$

The exponents in (22) indicate repetition of the same operation, where operations have to be performed from right to left. Equation (22) imposes an a-priori schedule to each station. This implies for instance that $\beta_j\mu_j$ repetitions of m_0 must be performed before the $\gamma_j\mu_j$ series of m_1. Notice that sequences in (22) are not unique, as, for instance, the order of m_0 and m_1 may be exchanged since they do not share semifinished objects. However series in (22) is preferable as m_0 must produce the object 2 to be consumed by m_2 as shown in Figure 1. A generic solution to the problem is out of the paper [4]. Their cycle times, under the linearity condition that has been defined in (3), hold

$$\begin{aligned} \underline{\tau}_j(0) &= \mu_j\left(\beta_j\tau(0) + \gamma_j\tau(1)\right) \\ \underline{\tau}_j(1) &= \mu_j\beta_j\left(\tau(2) + \alpha\tau(3)\right) \\ \underline{\tau}_j(2) &= \mu_j\tau(4+j) \end{aligned}. \tag{22}$$

C. Utilization optimization

Third is the design of the entries of the BOMO $\mathbf{a}_j$ in (21), where $\alpha << 1$ is given, but $\beta_j\mu_j$ and $\gamma_j\mu_j$ are still unknown. Approaching, but not reaching unitary utilization is the goal, as mentioned in Section II.B, upon definition of the utilization index $\underline{\upsilon}_j$ from (15) and $P = 3$, as follows

$$\underline{\upsilon}_j = \left(3\max_{p=0,1,2}\left\{\underline{\tau}_j(p)\right\}\right)^{-1}\sum_{p=0}^{2}\underline{\tau}_j(p). \tag{23}$$

The largest utilization, i.e. $\underline{\upsilon}_j = 1$, implies equal cycles in (23), namely

$$\underline{\tau}_j(0) = \underline{\tau}_j(1),\ \underline{\tau}_j(1) = \underline{\tau}_j(2). \tag{24}$$

The latter equation can be rewritten as

$$\begin{bmatrix} \alpha\tau(3)+\tau(2)-\tau(0) & -\tau(1) \\ \alpha\tau(3)+\tau(2) & 0 \end{bmatrix} \begin{bmatrix} \beta_j\mu_j \\ \gamma_j\mu_j \end{bmatrix} = \begin{bmatrix} 0 \\ \tau(4+j)\mu_j \end{bmatrix}, \quad (25)$$

and must be solved in the integral domain, The latter solution can be approximated by a suitable selection of the free integer scale factor $\mu_j \geq 1$. Here μ_j is minimized to dispose of aggregate operations with the shortest cycle time. More generically, when switching between operations requires setup times, μ_j should be selected to reduce the impact of the setup time on the overall cycle time. The problem is not treated here. The solution of (26) in the real number domain is

$$\frac{\gamma_j}{\beta_j} = \frac{\alpha\tau(3)+\tau(2)-\tau(0)}{\tau(1)} = \frac{\alpha 80+26}{20} = 1.3(1+2\eta)$$

$$\beta_j = \frac{\tau(4+j)}{\alpha\tau(3)+\tau(2)} = \frac{\{75,60,90,90\}}{\alpha 80+36} \cong \quad . \quad (26)$$

$$\cong \{2.1,1.7,2.5,2.5\}(1-\eta), \ \eta \cong 2\alpha$$

Integral solution is provided by rounding to the nearest integer, under assumption of $\eta << 1$:

$$\begin{aligned} \beta_j &= \text{round}\left(\{2.1,1.7,2.5,2.5\}(1-\eta)\right) = 2 \\ \gamma_j &= \text{round}\{2\cdot 1.3(1+2\eta)\} = 3, \ \mu_j = 1 \end{aligned}, \quad (27)$$

which provides a solution independent of j. Any deviation from (28) would correspond to a lower capacity utilization and to the existence of a bottleneck unit, as defined in (13) and shown in Table I under $\alpha = 0$. Lower utilization has been on the purpose assigned to $\underline{m}_j, j \geq 2$ by modifying either the value of β_j or that of γ_j in (28). Notice that the modified value of $\gamma_j, j = 1$, in Table I provides a slighly better utilization, which shows that the optimal solution should be searched for each j. Utilization coming out of (28) is reported in bracket for $\underline{m}_j, j = 1,2,3$.

TABLE I PROPERTIES OF THE AGGREGATED OPERATIONS

MO	Finished product	β_j	γ_j / β_j	Capacity utilization	Cycle time	Bottleneck unit
$\underline{m}_0$	7	2	3/2	0.95	80	PU_0
$\underline{m}_1$	8	2	1	0.89 (0.88)	72	PU_1
$\underline{m}_2$	9	2	2	0.87 (0.90)	100	PU_0
$\underline{m}_3$	10	1	3	0.73 (0.90)	90	PU_2

D. *WIP reduction*

Fourth, the series operations $s_j(p)$ have to be each other scheduled, aiming to shorten the manufacturing time $\underline{T}_j$ of $\underline{m}_j$ and thus the WIP defined in (16). The schedule is constrained by the input/output representation

$$\underline{m}_j = s_j(2)\left(s_j(0)+s_j(1)\right), \quad (28)$$

indicating that $s_j(2)$ must wait for the output types of $s_j(0)$ and $s_j(1)$. Actually the parallel of $s_j(0)$ and $s_j(1)$ cannot start without pre-production of the semifinished type 2, that is the output of $m = 0$ in Figure 1. Pre-production can be eliminated by re-scheduling the parallel (29) as in Figure 2. Manufacturing times and a-priori WIP are shown in Table II. A-priori WIP $\underline{w}_j$ has been defined in (16).

TABLE II MANUFACTURING TIMES AND WIP OF THE AGGREGATE OPERATIONS

MO	$\underline{m}_0$	$\underline{m}_1$	$\underline{m}_2$	$\underline{m}_3$
Manufacturing time	210	193	236	206
Nominal WIP	2.65	2.40	2.35	2.30

IV. CONTROL UNIT AND CONTROL STRATEGIES

A production control system is defined as a set of hierarchically interconnected units, in charge of real-time scheduling the operations of production units [5], [6]. A control unit c is a dynamic operator receiving as input the sequence $\underline{\sigma}_c = \left\{\underline{e}_c(c,j) = \left[\underline{t}_c(c,j), \underline{m}(c,j)\right]\right\}$, where $\underline{m}(c,j)$ is an admissible aggregate operation of the corresponding PU $\underline{p}(c)$. The output is the set of the command sequences $\sigma_c(p) = \left\{e_c(p,j) = \left[t_c(p,j), m(p,j)\right]\right\}$, defined in [1], toward the set $\mathcal{P}(c) = \{..., p(c), ...\}$ of the stations belonging to the aggregate production unit $\underline{p}(c)$. Such input and output sequences define to the top-down event flow (push) of a control unit (see Figure 3).

Further there exists a bottom-up event flow allowing control unit to estimate the state of the controlled units and to real-time schedule the relevant operations (pull). The bottom-up input sequence is the collection of all the output sequences of the elementary units, namely $\sigma_q(p)$ (drawing/delivery) and $\sigma_w(p)$ (start/end), that have been defined in [1], Section IVB. The bottom-up output sequence contains the output events of the corresponding PU, and includes the subset of drawing/delivery events in $\underline{\sigma}_q(\underline{m})$ restricted to input and output objects of the aggregated operations $\underline{m} \in \mathcal{M}\left(\underline{p}(c)\right)$, and the start/end sequence $\underline{\sigma}_w\left(\underline{p}(c)\right)$ of the aggregated unit.

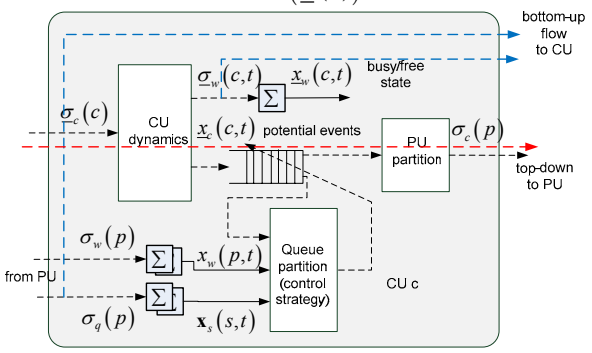

Figure 2 Block-diagram of the control unit dynamics.

The state equations are defined by the following three state variables: (i) the finite event sequence $\underline{x}_c(c,t)$ of the operations yet to be scheduled (potential events), (ii) the activation state $\underline{x}_w(c,t)$ of the aggregate PU c, (iii) the estimated states $\hat{\mathbf{x}}_s(s,t)$ and $\hat{x}_w(p,t)$ of the storage units s and of the production units p under control. The event queue $\underline{x}_c(c,t)$ is updated as soon as a new command event $\underline{e}_c(c,j)$ is received and as soon as the real-time control strategy dispatches a pending operation to an elementary unit. A simplified block-diagram of the control unit dynamics is shown in Figure 3.

A. Real-time control strategy

Real-time control strategies are event-driven closed-loop strategies tracking the demand and weakening the effects of micro and macro irregularities. They are strictly the same for any level of the hierarchical control system. Consider first a control strategy in absence of macro-irregularities. It is the sequence of three feedback laws arranging the potential event queue into tree sub-queues: pending, feasible and dispatchable.

1) *Feasible queue*: based on the estimated storage state $\hat{\mathbf{x}}_s(s,t)$, pending operations are declared feasible if their input objects are available and the output storages can receive output objects. Feasibility synchronizes operations with material flow. Kanban and CONWIP strategies [2] makes a MO feasible by re-circulating a finite number of reusable objects (cards, containers, ..) such to keep constant the WIP of production lines.
2) *Dispatchable queue*: based on the estimate of the production unit state $\hat{z}_p(p,t)$, feasible operations are declared dispatchable when the relevant unit is free. The law synchronizes operations with the factory plant capacity utilization. If applied to aggregated operations and production units, it forces the least amount of operations in production – work-in-process (WIP) - with the consequent reduction of the intermediate stock. The step is mandatory as the queued 'reusable' objects do not know whether downstairs PU are available or not.
3) *Dispatching rule*: several dispatching rules can be adopted for assigning priorities when more operations may be dispatched to a free unit. The criterion to be followed in the case study is of shortening the manufacturing time of the aggregated operation, thus reducing the WIP. A-priori and on-line scheduling serve to the purpose.

When macro-irregularities occur, i.e. the plant undergoes severe modifications, the model of the factory changes and the control strategy must be subsequently adapted. Model modifications require a self-diagnosis procedure, here not treated, to isolate and identify a specific macro-irregularity. Then the recovery strategy follows the same rules of the standard strategy. For instance, in the case of reworking,

1) the reworking operation is first declared feasible,
2) then is declared dispatchable as soon as the relevant unit becomes available,
3) finally it gains the highest priority to keep small the manufacturing time of the current MO.
4) In addition, the cycle time of the current MO is updated to slow down the command of a new MO. In this way the PU work-in-process is kept reasonably small.

Similar strategies can be adopted when production/storage units break down.

V. The Case-Study Simulated Results

Manufacturing and factory model, and the real-time production control system have been programmed in C language strictly following the outlined formulation [7]. No comparison with other strategies is presented.

A. Performance definition

Real-time control is assessed against the following a posteriori performance indices, strictly related to the criteria deployed in the design of the aggregated production unit. Performance prediction and dispersion because of irregularities is a further subject not treated here.

1) First is the SU capacity, to be kept small and defined for each pair $(\text{SU,type}) = (s,n)$ by
$$\bar{x}(s,n) = \max_t x_s(s,n,t). \qquad (29)$$
Small means sufficient not to starve downstream production units.
2) Second is the PU capacity utilization (15), to be kept close to the a priori values in Table I, and defined for each elementary unit p or aggregate unit $\underline{p}(c)$ by
$$\upsilon(p,T) = \max_t T^{-1}\int_t^{t+T}\left(1 - x_w(p,\tau)\right)d\tau \le 1, \qquad (30)$$
3) Third is the work-in-process $\underline{w}_j = \underline{w}(\underline{m}_j)$, to be kept close to a-priori values in Table II, and defined for each finished product $n = 7 + j$ as follows (each finished product corresponds to an aggregated MO). Denote with $\underline{x}_s(2,n,t)$ the state of a virtual storage where product quantities are added as soon as their MO is started by the aggregated $\underline{\text{PU}}_0$, and with $\underline{x}_s(2,n,t)$ the actual stock of products in the output SU $s = 2$ (see Figure 6 in [1]):
$$\underline{w}_j = \max_t\left(\underline{x}_s(2,7+j,t) - x_s(2,7+j,t)\right). \qquad (31)$$

B. Micro-irregularity control

An arbitrary series $\underline{s}$ of the four aggregate operations $\underline{m}_j$, $j = 0,\ldots,3$,
$$\underline{s} = \underline{m}_{j_0}^{\alpha_0}\underline{m}_{j_1}^{\alpha_1}\cdots\underline{m}_{j_h}^{\alpha_h}\cdots,\ j_h = 0,1,2,3, \qquad (32)$$
with exponents $\alpha_h = 25 \div 30$, has been assumed to be commanded to the CU_0 (the control unit of the whole factory) by the factory planner CU_1 (see Figure 6 in [1]). The resulting sequence is arbitrary except that the long-term average demand of any finished product is constant (close to 2.5 parts per 1000 time units). The factory planner may be formulated as the highest-level control unit. The same sequence has been applied in absence and under micro-irregularities, the latter ones affecting both cycle times and delivery delays to output storage units. Time irregularities have been modelled as normally distributed around with standard deviation around 10% of the mean cycle time [1].

TABLE III Simulated Performance under Arbitrary Production Plan

Performance	Micro/macro irregularities		
	Without	Micro	Micro and macro
Capacity utilization, PU_0	0.83÷0.84	0.83÷0.84	0.76÷0.78 (-8%)
Capacity utilization, PU_1	0.66÷0.74	0.66÷0.74	0.68÷0.75
Capacity utilization, PU_2	0.87÷0.89	0.87÷0.89	0.8÷0.82 (-8%)
Work-in-process (total and long-term)	2.42	2.7 (+11%)	2.4
SU capacity, any (s,k)	≤4	≤5	≤8
Average cycle time, $\underline{\text{PU}}_0$	87	87	94

Performance indices are reported in Table III, showing a consistent degradation with respect to a priori values in Table I and Table II. A deeper assessment is out of this paper.

Figure 4 shows a simulated realization of the average work-in-process of all finished products, in presence of micro-irregularities. The time span is about 40000 time units. Micro irregularities increase the average WIP of about 11%, which is consistent with the cycle time variability.

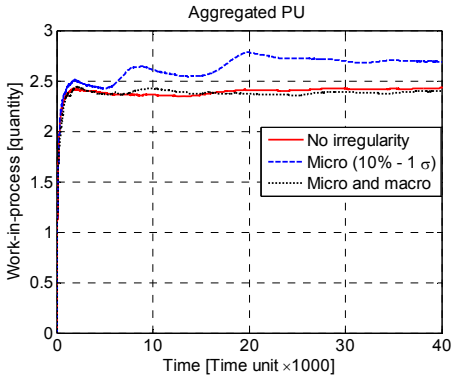

Figure 3 Simulated time evolution of the average work-in-process.

Figure 4 shows the throughput of the finished product 9 tending to 2.5 parts every 1000 time units, which is consistent with the production order rate. Micro irregularities abate the throughput of about 16%, larger than 10% because a single type is considered.

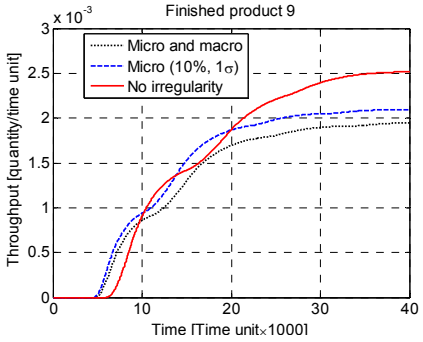

Figure 4 Throughput of the finished product 9.

C. *Macro-irregularity control*

Reworking macro-irregularities have been introduced, with a defective probability $\alpha = 0.05$ .Performance clearly degrades, especially the average cycle time of the aggregate production unit in Table III, due to extra working time and delivery delay imposed by reworking. However work-in-process in Figure 3 remains almost the same as the case of no irregularities, which is significant a control strategy should keep bounded and as small as possible WIP. Storage capacity is two times greater due to self-diagnosis delay. Of course throughput in Figure 4 is further abated. Figure 5 shows the utilization of the PU0 with and without macro irregularities. Utilization is reduced in the latter case of about 8%.

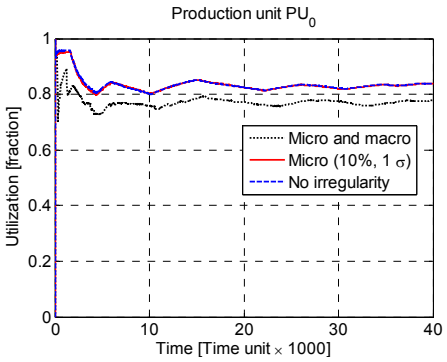

Figure 5 Utilization of PU 0.

Deeper and more formal assessments should be done by comparing predictions and performance under different demand scenarios, and using alternative control strategies. Thus the above results should be meant as introductory.

VI. CONCLUSIONS

Concepts and entities of Manufacturing Algebra have been employed to formulate aggregation of manufacturing operations and production units. Aggregation is strictly related to planning and scheduling problems. Specifically balanced operations (leaving intact the semifinished stock) are candidate to become aggregate operations. In turn a set of aggregate operations defines the capability of an aggregate production units, collecting all the stations where elementary MO are implemented. The inverse problem of disaggregating a finished product demand to the commands of the factory stations, requires a careful design of aggregate PU in terms of their feasible aggregate MO. A procedure of this sort has been outlined and demonstrated with a simple case study, without any claim of optimality and generalization. Aggregation is strictly related to the control hierarchy, which is two layer in the case study. After a brief outline of the control unit dynamics, data flow and real-time strategies, simulated results have been presented and briefly assessed.

REFERENCES

[1] E. Canuto and M. De Maddis "Manufacturing Algebra. Part I:modelling principles and case study", acceptede to *ICMA 2012*.

[2] W.J. Hopp and M.L. Spearman, *Factory Physics. Foundations of manufacturing management,*2nd ed., Irwin McGraw-Hill, Boston, 2000.

[3] G. Cohen, S. Gaubert and J.P. Quadrat "Max-plus algebra and System Theory: where we are and where to go now", *Annual Reviews in Control*, Vol. 23, pp 207-219, 1999.

[4] J.-M. Proth and C. Chu *L'ordonnancement et ses applications,* Masson, 1996.

[5] E. Canuto E. "Discrete-event modelling and control of manufacturing Systems". *1998 IEEE Int. Conf. on Control Applications,* Trieste (Italy), September 1998, pp. 781-785.

[6] E. Canuto "Discrete-event models of manufacturing systems", *6th IEEE Med. Control Conf.*, Alghero (Italy), 9-11 June 1998. pp. 747-752.

[7] E. Canuto, F. Donati and M. Vallauri, "Discrete-event simulator of manufacturing systems as a tool for designing and testing real-time hierarchical production control systems". In *12 Symposium Simulationstechnik*, Zurich, September 1998. pp. 161-168.