

(19) **DANMARK**



Patent- og
Varemærkestyrelsen

(12)

Oversættelse af europæisk patentskrift

(10) **DK/EP 3816896 T3**

-
- (51) Int.Cl.: **G 06 Q 10/08 (2023.01)** **B 65 G 43/10 (2006.01)** **G 06 Q 10/063 (2023.01)**
- (45) Oversættelsen bekendtgjort den: **2023-11-13**
- (80) Dato for Den Europæiske Patentmyndigheds bekendtgørelse om meddelelse af patentet: **2023-09-06**
- (86) Europæisk ansøgning nr.: **20202699.3**
- (86) Europæisk indleveringsdag: **2020-10-20**
- (87) Den europæiske ansøgnings publiceringsdag: **2021-05-05**
- (30) Prioritet: **2019-10-30 FR 1912231** **2019-11-15 FR 1912777**
- (84) Designerede stater: **AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR**
- (73) Patenthaver: **Savoye, 18 Boulevard des Gorgets, 21000 Dijon, Frankrig**
Université de Technologie de Compiègne, Rue du Docteur Schweitzer , BP 60319, 60203 Compiègne Cedex, Frankrig
- (72) Opfinder: **VACHER, Blandine, La Loge des Bois, 03340 NEUILLY-LE-RE AL, Frankrig**
PIETROWICZ, Stéphane, 27 rue de la Croix Blanche, 21220 Fixin, Frankrig
JOUGLET, Antoine, 23, Avenue de la Marne, 60200 COMPIEGNE, Frankrig
NACE, Dritan, 6, Rue Guillaume Apollinaire, 60200 COMPIEGNE, Frankrig
- (74) Fuldmægtig i Danmark: **RWS Group, Europa House, Chiltern Park, Chiltern Hill, Chalfont St Peter, Bucks SL9 9FG, Storbritannien**
- (54) Benævnelse: **FREMGANGSMÅDE TIL SAMMENLÆGNING AF K INKOMMENDE STRØMME AF NYTTELASTER TIL EN UDGÅENDE STRØM I ET LOGISTIKLAGER**
- (56) Fremdragne publikationer:
WO-A1-2013/150080

Description

1. TECHNICAL FIELD

5 In recent years, intra-logistics activity has been significantly affected by a drastic change in the behaviour of consumer society and by the development of new technologies. The constant increase in demand forces logistics stakeholders to become increasingly responsive and competitive. It has become vital to
10 improve and optimise routing, storage and order-picking solutions. The SAVOYE company, which builds order-picking equipment, specialising in the automation and computerisation of logistics warehouses, is constantly improving and evaluating its solutions.

15 Management of a logistics warehouse includes procuring supplies, as well as receiving and managing flows and stocks. Improving the travel time of an order often means minimising its conveyance time. 50 % of the picking time for an order is devoted to this
20 conveying operation, according to the following article: De Koster, R., Le-Duc, T., and Roodbergen, K. J., "Design and control of warehouse order picking: a literature review" (2007), European Journal of Operational Research 182(2), 481-501. Therefore, in the first instance, the main problem encountered
25 by warehouses needs to be addressed, namely that of optimising the conveyed flows in order to ensure the highest possible production rate. The present description will, more specifically, focus on the intersections of flows of loads (packages, bins, containers, etc.). The aim is to maximise the throughput of the
30 final flow (also called outgoing flow) combining various incoming flows originating from different places.

2. PROBLEM

35 2.1 SUBJECT MATTER

In a logistics warehouse, several incoming flows of loads (also called injection flows) are actually grouped together in order

to form a single outgoing flow (also called outflow). A flow of loads corresponds to a list of sequenced loads, proceeding one after the other. These flows can be transported by a conveyor that is similar to a conveyor belt that moves all the loads located thereon in a given direction in single file. Throughout the remainder of the description, the conveyors transporting the incoming flows are called aisles and the conveyor transporting the outgoing flow is called collector.

The injection flows are distributed all along the collector. The positions of the loads forming these injection flows are known and identifiable. For each injection flow, it is possible to choose the time at which its first load is injected onto the collector. Sequential distribution of the injection flows allows an aisle located upstream of the collector flow to benefit from a greater number of injection possibilities than that of an aisle located downstream. Indeed, said aisle can only discharge its loads when space is left free by the aisles further upstream.

In order to prevent an imbalance in the injection of loads between the incoming flows, previously defining the order of the loads once they are all placed on the collector is preferred (this order is called final sequence or even exit sequence).

The guiding principle is to push the flow of the collector to the maximum mechanical capacity of the collector. To this end, the number of empty spaces in the collector needs to be limited. The present invention proposes an approach based on the correct dates for injecting loads, in order to form a final flow that has no empty spaces (or as few of them as possible). These injection dates are defined in order to comply with any achievable final sequence, i.e., complying with FIFO (First-In-First-Out) orders of the injection flows.

The solution proposed and described hereafter in order to address this problem is novel and inventive in light of known models of scheduling problems, notably:

- Runway Cheng, Mitsuo Gen and YasuhiroTsujimura, "A tutorial

survey of Job Shop scheduling problems using genetic algorithms”
--- Section I. Representation, Ashikaga Institute of Technology,
Ashikaga 326, Japan; and

- Imran Ali Chaudhry and Abid Ali Khan, “A research survey:
5 review of flexible job shop scheduling techniques, International
transactions in operation research, 2015.

2.2 EXAMPLE OF AN APPLICATION

10 To summarise, the studied system is made up of injection flows
(incoming flows) transported by aisles (set of conveyors) and a
collector (another conveyor) combining these injection flows
into a single final flow (outgoing flow). The control system
proposed and described hereafter notably, but not exclusively,
15 applies to a “continuous non-accumulation conveyor” type of
collector or to a “sorter” type of collector (“belt tray”, “tilt
tray”, etc.). This control system allows compliance with any
type of final achievable sequence. The present description
therefore provides the solution corresponding to the optimal
20 throughput relative to a final given sequence. If no final
sequence is desired, it is possible to obtain the maximum
throughput on a completely unoccupied collector in order to
combine these injection flows.

25 However, in some cases, obstructions can already occupy places
on the collector. The term “obstruction of a time step of the
collector” is notably, but not exclusively, understood to mean
a disruptive load originating from a disruptive flow different
from the injection flows. An obstruction is not necessarily a
30 disruptive load, but also can be a point on the collector that
is damaged, reserved, etc. In this case, management of the
injection flows must take into account such obstructions.

In the particular case whereby the disruptive loads originate
35 from one or more disruptive flows, a distinction is made between
two types of disruptive flows. A disruptive flow is referred to
as “non-controlled” if the injection of its constituent
disruptive loads onto the collector cannot be controlled by the

control system (these disruptive loads arrive without taking into account the environment). Conversely, a disruptive flow is referred to as **"controlled"** if the injection of its constituent disruptive loads onto the collector can be controlled by the control system. The proposed control system supports these different types of disruptive flows (non-controlled or controlled) and provides an optimal solution that includes the management of the disruptive flows to be controlled.

One (illustrative and non-limiting) example of a practical application is found in the GTP Intelis PTS (Goods To Person Intelis Picking Tray System) solution proposed by SAVOYE. This is an overall automated order-picking solution in which an automated shuttle-based storage and retrieval system (called Intelis PTS) 3 allows the picking stations (also called GTP stations) 4 to be fed at very high speeds. Therefore, the role of the automated system (Intelis PTS) is to store and enter and exit loads comprising articles needed for correctly filling orders on the GTP stations. The automated system is made up of several storage racks (also called "PTS aisles", which nevertheless should not be mistaken for aisles (within the meaning of conveyors) transporting incoming flows within the context of the present invention, see the explanations hereafter), each made up of shuttles and elevators allowing these loads to move to-and-fro on the incoming 5_1 , 5_2 and outgoing conveyors 2_1 , 2_2 , respectively. These different incoming and outgoing conveyors are all connected by a collector 1 that feeds the picking stations 4, as shown in **figure 50** (for example, with three picking stations 4_1 , 4_2 and 4_3).

In order to establish a link with the above contextual definition, the outgoing conveyors 2_1 , 2_2 of the automated system 3 form the conveyors (also called "aisles") transporting the incoming flows of payloads, with these incoming flows being merged on the collector 1 into an outgoing flow of payloads.

In other words, the to-and-fro movement of the loads that enter and exit the automated system 3 is managed by means of the

collector. There are therefore several injection flows of loads (also called incoming flows in the present problem section) on the outgoing conveyers 2₁, 2₂ of the automated system 3, all injected onto the collector.

5

In addition, when the flows of loads that return to the automated system 3 (via the incoming conveyers 5₁, 5₂) pass through the same collector 1, then this is an example of disruptive flows, called return flows in this particular case. These disruptive flows are highly interesting since they disappear following the passage of all the injection zones, thus creating empty spaces for some injection flows but not for others. As described in detail hereafter, one embodiment of the proposed solution is aimed at controlling, as far as is possible, these disruptive flows in order to fill these empty spaces rather than letting them remain.

10

15

The solutions proposed in this example of an application are described throughout the remainder of the document.

20

2.3 CURRENT SOLUTIONS

2.3.1 Existing solutions

25

- Known solution A, with desired final sequencing: in this case, the injection of the loads onto the collector follows a well-defined order corresponding to the expected final sequence after merging the loads. If this desired final sequence when exiting the collector is formed, for example, by the loads with sequence numbers 1, 2, 3, etc. (hereafter called "load 1", "load 2", etc.), then load 1 will be injected as a priority, followed by load 2 only if load 1 is downstream of load 2 on the collector, followed by load 3 only if load 2 is downstream of load 3 on the collector, and so on;

30

35

- Known solution B, without desired final sequencing: each first load of each injection flow is injected onto the collector as soon as there is space in its injection zone.

Document WO 2013/150080 describes techniques for implementing the aforementioned solutions A and B.

2.3.2 Disadvantages of these solutions

5

- Known solution A: one disadvantage of known solution A is that it leaves (empty) spaces between the loads, which slows down the outgoing flow relative to the mechanical capacity of the collector. An empty space appears between two successive loads in the final sequence whenever the previous load in the final sequence belongs to an injection flow further downstream than the following load. In addition, the length of these empty spaces is proportional to the distance of the injection aisles (conveyors) from these two successive loads.

15

Several steps of this known solution A are shown in figure 1. In this example, the desired final sequence 1, 2, 3, 4 is considered. In order to comply with this sequence, directly injecting loads 1, 2, 3 and 4 onto the collector at the same time is not possible. In step 0, the installation is activated. Instantly, loads 1 and 2 are injected onto the collector. This operation is made possible because the injection flow of load 1 is downstream of the injection flow of load 2. Loads 1 and 2 are on the collector, separated by the distance between the aisles (conveyors) from which they originate. The collector moves forward without changing until step 1, where load 2 passes in front of the injection zone for the flow containing load 3. From this time onwards, load 3 is injected onto the collector directly behind load 2. It should be noted that the empty space left between loads 1 and 2 has remained the same and will never be filled. In this same step 1, load 4 is also injected onto the collector with a space between itself and load 3. Step 2 shows the empty spaces left by this known injection method.

20

25

30

35

- Known solution B: one disadvantage of known solution B is that the aisle (the conveyor) furthest downstream will continuously inject its loads, while the others will have to settle for filling the remaining free spaces; another

disadvantage of known solution B is that it proposes a control system that in reality only works when the speed of the collector (transporting the outgoing flow) is much greater than that of the aisles (conveyors) transporting the incoming flows (as many times greater as there are incoming flows to be injected). In known solution B, the speeds of the aisles (conveyors) therefore must be capped. However, in the current context, the number of injection flows increases faster than the mechanical capacities of the collector. This known solution B is no longer sufficient, and does not control the optimisation of the throughput of the outgoing flow.

3. SUMMARY

In a first particular embodiment of the invention, a method is proposed for merging, in a logistics warehouse, k incoming flows of payloads, respectively transported by k conveyors, called aisles a_i , with $i \in \{1, \dots, k\}$, in one outgoing flow of payloads, transported by another conveyor, called collector, the logistics warehouse being such that:

- the k aisles are of the "first-in first-out" type, distributed along the collector and numbered from a_1 to a_k in a direction of displacement of the collector; and
- Δ_i is a time distance between aisles a_i and a_k expressed in units of time each corresponding to a time step of the collector, the method being executed at least once by a control system, a given execution being performed at a time T_b and comprising:
 - obtaining a set L comprising n payloads distributed at the time T_b over the k aisles and to be injected onto the collector to form an output sequence σ , each of the n payloads being identified by a unique sequence number within the output sequence σ , each of the k aisles containing an ordered set, according to an upward direction of the sequence number, of h_i payloads to be injected one by one onto the collector;
 - calculating a date t_0 on which the first payload σ_1 of the output sequence σ passes in front of aisle a_k ;
 - calculating an injection date onto the collector for each of the n payloads of the set L , according to the following

formula: $T(u) = t_0 + \sigma(u) - 1 - \Delta_i$, with:

- $u = a_i(j) \in L$, being a payload of the set L and originating from the j^{th} position in aisle a_i , $i \in \{1, \dots, k\}, j \in \{1, \dots, h_i\}$;

5 • $\sigma(u)$ being the sequence number of the payload u in the output sequence σ ;

- controlling the collector and the k aisles, for an injection of the n payloads onto the collector in accordance with the n injection dates $T(u)$, $\forall u \in L$.

10

According to a particular feature of the first embodiment, $\Delta_{i,i'}$, is a time distance between aisles a_i and $a_{i'}$ expressed in units of time each corresponding to a time step of the collector, and the method further comprises the following steps:

15 - for each payload $u = a_i(j) \in L$ originating from an aisle a_i , calculating the dates of passage of the payload in front of aisles a_i to a_k , knowing the injection date $T(u)$ of the payload onto the collector;

20 - for each aisle a_i , $i \in \{1, \dots, k\}$, obtaining a set V_i containing each date of passage of one of the payloads in front of aisle a_i ;

- for each controlled disruptive load p of a disruptive flow controlled by virtue of control means located along the collector and in front of an aisle $a_{g(p)}$ in the direction of displacement of the collector, with $g(p) \in \{1, \dots, k\}$, $a_{g(p)}$ also being the first aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes:

30 a) initialising t with the date $\delta_1 - \Delta_1$, with δ_1 representing the possible arrival date, in the case of non-queuing, of the first controlled disruptive load in front of aisle a_k ;

b) calculating the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, $t + \Delta_{1,l(p)}$ of passage of the controlled disruptive load in front of aisles $a_{g(p)}$ to $a_{l(p)}$, with $a_{l(p)}$ being the last aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes;

35 c) if none of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to the sets $V_{g(p)}$, $V_{g(p)+1}, \dots, V_{l(p)}$,

calculating a fictitious date of passage $start_p$ of the controlled disruptive load p in front of aisle a_1 , with the following formula: $start_p = t$;

- d) if one of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to one of the sets $V_{g(p)}$, $\Delta_{1,g(p)+1}, \dots, V_{l(p)}$, incrementing t by one unit and repeating steps b), c) and d) with the new value of t ;
- controlling the control means, for an injection on the date $start_p + \Delta_{1,g(p)}$ of each controlled disruptive load p in front of aisle $a_{g(p)}$.

In a second particular embodiment of the invention, a method is proposed for merging, in a logistics warehouse, k incoming flows of payloads, respectively transported by k conveyors, called aisles a_i , with $i \in \{1, \dots, k\}$, in one outgoing flow of payloads, transported by another conveyor, called collector, the logistics warehouse being such that:

- the k aisles are of the "first-in first-out" type, distributed along the collector and numbered from a_1 to a_k in a direction of displacement of the collector; and
- Δ_i is a time distance between aisles a_1 and a_k expressed in units of time each corresponding to a time step of the collector, the method being executed at least once by a control system, a given execution being performed at a time T_b and comprising:
 - obtaining a set L comprising n payloads distributed at the time T_b over the k aisles and to be injected onto the collector to form an output sequence σ , each of the n payloads being identified by a unique sequence number within the output sequence σ , each of the k aisles containing an ordered set, according to an upward direction of the sequence number, of h_i payloads to be injected one by one onto the collector;
 - for each aisle a_i , $i \in \{1, \dots, k\}$, obtaining a set U_i , of invalid dates for an injection of one of the payloads of aisle a_i onto the collector due to a non-controlled disruptive load or another obstruction of a time step of the collector;
 - by not taking into account the sets U_i , $i \in \{1, \dots, k\}$, calculating a date t_0 on which the first payload σ_1 of the output sequence σ passes in front of aisle a_k ;

- for the first payload σ_1 of the output sequence σ , assuming that the first payload σ_1 originates from aisle a_i :
 - a) initialising t with t_0 ;
 - b) if none of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, calculating an injection date of the first payload σ_1 onto the collector with the following formula:
$$T(u) = t - \Delta_i,$$
with $u = \sigma_1 = a_i(j)$;
 - 10 c) if one of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to one of the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, incrementing t by one unit and repeating steps b) and c) with the new value of t ;
- for each following payload σ_c , of the output
15 sequence $\sigma, c \in \{2, \dots, n\}$, assuming that the following payload σ_1 originates from aisle a_i :
 - a') incrementing t by one unit, with t used to calculate the injection date of the preceding payload σ_{c-1} ;
 - b') if none of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$,
20 calculating an injection date of the following payload σ_c onto the collector with the following formula:
$$T(u) = t - \Delta_i,$$
with $u = \sigma_c = a_i(j)$;
 - 25 c') if one of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to one of the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, incrementing t by one unit and repeating steps b') and c') with the new value of t ;
- controlling the collector and the k aisles, for an
30 injection of the n payloads onto the collector in accordance with the n injection dates $T(u), \forall u \in L$.

According to a particular feature of the second embodiment, the step of controlling the collector and the k aisles, for an
35 injection of the n payloads onto the collector in accordance with the n injection dates $T(u), \forall u \in L$, is preceded by the following steps:

- calculating t_{min} according to the following formula:

$t_{min} = \min \{T(u), \forall u \in L\};$

- if $t_{min} > 0$, modifying the n injection dates according to the following formula:

$T(u) = T(u) - t_{min}, \forall u \in L.$

5

According to a particular feature of the second embodiment, $\Delta_{i,i}$, is a time distance between aisles a_i and $a_{i'}$ expressed in units of time each corresponding to a time step of the collector, and the method further comprises the following steps:

10 - for each payload $u = a_i(j) \in L$ originating from an aisle a_i , calculating the dates of passage of the payload in front of aisles a_i to a_k , knowing the injection date $T(u)$ of the payload onto the collector;

- for each aisle $a_i, i \in \{1, \dots, k\}$, obtaining a set V_i ,
15 containing, on the one hand, each date of passage of one of the payloads in front of aisle a_i and, on the other hand, the set U_i of invalid dates for an injection of one of the payloads of aisle a_i onto the collector;

- for each controlled disruptive load p of a disruptive flow
20 controlled by virtue of a control means located along the collector and in front of an aisle $a_{g(p)}$ in the direction of displacement of the collector, with $g(p) \in \{1, \dots, k\}$, $a_{g(p)}$ also being the first aisle, in the direction of displacement of the collector, in front of which the controlled disruptive
25 load passes:

a) initialising t with the date $\delta_1 - \Delta_1$, with δ_1 representing the possible arrival date, in the case of non-queuing, of the first controlled disruptive load in front of aisle a_k ;

b) calculating the dates $t + \Delta_{1,g(p)}, t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ of
30 passage of the controlled disruptive load in front of aisles $a_{g(p)}$ to $a_{l(p)}$, with $a_{l(p)}$ being the last aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes;

c) if none of the dates $t + \Delta_{1,g(p)}, t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively
35 belongs to the sets $V_{g(p)}, \Delta_{1,g(p)+1}, \dots, V_{l(p)}$, calculating a fictitious date of passage $start_p$ of the controlled disruptive load p in front of aisle a_1 , with the following formula: $start_p = t$;

d) if one of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to one of the sets $V_{g(p)}$, $V_{g(p)+1}, \dots, V_{l(p)}$, incrementing t by one unit and repeating steps b), c) and d) with the new value of t ;

- 5 - controlling the control means, for an injection on the date $start_p + \Delta_{1,g(p)}$ of each controlled disruptive load p in front of aisle $a_{g(p)}$.

According to a particular feature of the first or second
10 embodiment of the invention, the date t_0 is calculated with the following formula:

$$t_0 = \max_{u = a_i(j) \in L'} \{ \Delta_i + 1 - \sigma(u) \} .$$

with:

- $u = a_i(j) \in L'$, being a payload of a set L' comprising
15 payloads placed in the first position of aisles located, following the direction of displacement of the collector, from aisle a_i to aisle a_{i1} containing the first payload σ_1 of the output sequence σ .

20 According to a particular feature of the first or second embodiment, the method is executed iteratively, each new execution being performed at a new time T_b calculated with the following formula: $T_b = T(u_{last}) - \Delta_1 + \Delta_{ilast}$, with $u_{last} = a_{ilast}(j^{last})$ being a last payload, originating from
25 the $(j^{last})^{th}$ position in aisle a_{ilast} , of the output sequence σ of a previous execution at a previous time T_b .

According to a particular feature of the first or second
30 embodiment, the method is executed iteratively, each new execution being performed at a new time T_b defined as a time at which no load of the output sequence σ of a previous execution at a previous time T_b is in a portion of the collector ranging from the first aisle a_1 to aisle a_{i1} containing the first load of the output sequence σ of the new execution at the new time
35 T_b .

According to a particular feature of the first or second embodiment, the method is executed iteratively, each new

execution being performed at a new time T_b calculated with the following formula:

$$T_b = \text{Max}(T_b + 1, T(U_{last}) - \Delta_1 + \Delta_{ilast}),$$

with:

- 5 • $U_{last} = a_{ilast}(j^{last})$ being a last payload, originating from the $(j^{last})^{th}$ position in aisle a_{ilast} , of the output sequence σ of a previous execution at a previous time T_b ; and
- $(T_b + 1)$ being obtained by incrementing the previous time T_b by one unit.

10

Another embodiment of the invention proposes a computer program product comprising program code instructions for implementing the above method (in any one of its various embodiments), when said program is executed on a computer.

15

Another embodiment of the invention proposes a computer-readable and non-transient storage medium storing a computer program comprising a set of computer-executable instructions for implementing the above method (in any one of its various

20 embodiments).

Another embodiment of the invention proposes a control system (device) comprising means for implementing steps that it performs in the method as described above, in any one of its

25 various embodiments.

4. LIST OF FIGURES

In the following description, which is provided by way of an indicative and non-limiting example, reference is made to the

30 appended drawings, in which:

Figure 1 shows an existing method for inserting loads in a given order;

Figure 2 is an image of two aisles (conveyors), a collector and a load (box);

35

Figure 3 is a drawing of two aisles (conveyors), a collector and a load (box);

Figure 4 is a drawing of the studied aisle system;

Figure 5 is a drawing of the system indicating the loads using their sequence number;

Figure 6 shows coinciding points without loss of generality;

Figure 7 shows a start point and an injection point;

5 Figure 8 shows the division of the collector into steps (positions or slots);

Figure 9 shows the desired exit sequence expected when dividing the collector;

10 Figure 10 shows the unit of time associated with the step of the collector relative to the duration for injecting a load;

Figure 11 shows integer distances between the injection flows;

Figure 12 shows distances between the injection flows that are not necessarily integer distances;

15 Figure 13 is a drawing of the collector with injection points consecutively following each step;

Figure 14 shows an example of a collector with 4 aisles consecutively placed according to the step of the collector;

Figure 15 is a Gantt chart related to the job shop and retrieving data to be calculated by a formula;

20 Figure 16 shows an example of a collector with 4 randomly placed aisles;

Figure 17 is a Gantt chart related to the job shop with any distances between the aisles;

25 Figure 18 shows an example of a collector and 2 aisles with the load identifiers;

Figure 19 shows an example of a collector and 2 aisles with the order number identifiers;

Figure 20 shows an example of a collector and 2 aisles with the sequence number identifiers;

30 Figure 21 shows an image of the collector at time 0;

Figure 22 shows an image of the collector at time 1;

Figure 23 shows an image of the collector at time 2;

Figure 24 shows an image of the collector at time 3;

35 Figure 25 shows an example of a FIFO list associated with three aisles;

Figure 26 shows an achievable sequence, expected on the collector;

Figure 27 shows the first load of the flow attached to the

injection point, forming part of the FIFO list;

Figure 28 shows the first load of the flow not attached to the injection point, not forming part of the FIFO list;

Figure 29 shows loads on continuous hold behind a first load
5 forming part of the FIFO list;

Figure 30 shows loads on continuous hold behind a first load not forming part of the FIFO list;

Figure 31 shows an example of a work batch;

Figure 32 shows a calculation of the injection dates and the
10 processing of batch 1;

Figure 33 shows the processing of an ongoing batch;

Figure 34 shows the end of processing of batch 1 and the formation of batch 2;

Figure 35 shows the end of processing of batch 2 and its
15 application;

Figure 36 is a diagram of the algorithm in the general case with a non-controlled disruptive flow;

Figure 37 shows an example of a disruptive flow: shaded (crossed) loads that return to the PT;

20 Figure 38 shows an example of a collector with 4 unspecified aisles with a non-controlled disruptive flow;

Figure 39 is a Gantt chart with a non-controlled disruptive flow forming spaces;

Figure 40 is a diagram of algorithm 4;

25 Figure 41 shows an example of a collector with 4 randomly placed aisles with a controlled disruptive flow;

Figure 42 is a Gantt chart of the injections showing the spaces to be used for the disruptive loads;

Figure 43 is a Gantt chart when choosing the placement of the
30 disruptive loads;

Figure 44 shows a structure of a control system according to a particular embodiment of the invention;

Figure 45 is a diagram of algorithm 5;

Figure 46 is a flowchart of a method for merging k incoming
35 flows in one outgoing flow, according to a first particular embodiment based on algorithm 2;

Figure 47 is a flowchart of a method for merging k incoming flows in one outgoing flow, according to a second particular

embodiment based on algorithm 3 (illustrated in figure 36);
 Figure 48 is a flowchart of a method for merging k incoming
 flows in one outgoing flow, according to a third particular
 embodiment based on algorithm 4 (illustrated in figure 40);
 5 Figure 49 is a flowchart of a method for merging k incoming
 flows in one outgoing flow, according to a fourth particular
 embodiment based on algorithm 5 (illustrated in figure 45);
 Figure 50 shows an example of an application.

10 5. CONTROL PARAMETERS

As already mentioned above, the studied system is made up of a
 discharge system (a conveyor called collector), several other
 conveyors (called aisles) and loads. This system is dynamic, the
 15 loads are transported by the aisles, injected onto the collector
 and then transported by the collector. Initially, the state of
 this system will be considered by setting the position of each
 load present therein at a given time. The set sub-systems of
 this system will be defined in order to be able to calculate the
 20 future dates for injecting certain loads. This corresponds to
 the resolution of the static problem, Section 7 will examine how
 to dynamically control this system.

Figure 2 is an image illustrating two aisles (conveyors) 2_1 , 2_2 ,
 25 a collector 1 and a load (box or carton) 6. **Figure 3** is a model
 view of the elements appearing in the image of figure 2 (with
 the same numerical references).

5.1 NOTATIONS

30

The following table is a summary of the notations.

[TABLE 1]

k	Number of injection flows (incoming flows) of the system
$a_i, i \in \{1, \dots, k\}$	k aisles (conveyors) each

	transporting an ordered set of loads at a given time
$h_i, i \in \{1, \dots, k\}$	Number of loads present on aisle (conveyor) a_i
$n = \sum_{i=1 \dots k} h_i$	Total number of loads to be injected onto the collector
L	Set of loads to be injected onto the collector
$u = a_i(j), i \in \{1, \dots, k\}, j \in \{1, \dots, h_i\}$	Load $u \in L$, coming from the j^{nth} position in aisle a_i
σ	Desired exit sequence, ordering all the loads of L
$\sigma(u), u \in L$	Position of the load u in the exit sequence
σ_c	c^{nth} load of the sequence σ , belonging to L set $u = a_i(j)$ to σ_c signifies the calculation of u, i, j for the load σ_c
L'	Set of loads placed in the first position of the aisles located between aisle a_1 and the aisle containing the first load of the sequence σ (that is σ_1)
$T_b(u), u \in L$	Start date of the load u , coinciding with the start point, i.e., the time at which the load must leave its aisle (conveyor)
$T(u), u \in L$	Injection date of the load u onto the collector, coinciding with the injection point at which the load physically arrives on the collector
$Pt_i, i \in \{1, \dots, k\}$	Time needed to travel the distance between the start

	point and the injection point for a load of aisle a_i
The "step"	Unit of time corresponding to the division of the collector into time steps (also called time positions or time slots)
The date 0	Date on which it is possible to start injecting the loads onto the collector (date expressed by the first unoccupied location thereon)
The date t_0	Date on which the first load of the desired exit sequence passes in front of aisle a_k
$\Delta_i, i \in \{1, \dots, k\}$	Distance in "steps" between aisle a_i and aisle a_k
$\Delta_{i,j}, i \in \{1, \dots, k-2\}, j \in \{1, \dots, k-1\}$	Distance in "steps" between aisle a_i and aisle a_j

5.1.1 Notations on aisles of the conveying system

For a configuration of the system at a given time, each aisle
 5 comprises an ordered set of loads. Let k be the number of aisles
 in the present system, each aisle is numbered a_1 to a_k in the
 direction of displacement of the collector. These aisles
 respectively have a number h_i of loads, represented by a First-
 In-First-Out (FIFO) list that has to be injected onto the
 10 collector one by one.

An example is provided in **figure 4**. The direction of displacement
 of the collector 1 is from right-to-left, this direction is
 indicated by the arrow. Each aisle is numbered and comprises
 15 ordered loads, which must be injected onto the collector one by
 one, from bottom to top. In this example, therefore, $k = 3$, with
 $h_1 = 2$, content of $a_1 = \{id11, id12\}$, $h_2 = 3$, content of $a_2 =$
 $\{id21, id22, id23\}$ and $h_3 = 4$, content of $a_3 = \{id31, id32, id33,$
 $id34\}$.

5.1.2 Notations of the loads to be injected

There are n loads (boxes, bins, containers, etc.) in total to be injected onto the collector, originating from the various injection flows. Let L denote all these loads and $u = a_i(j)$ denote a load u belonging to L , originating from the j^{th} position in aisle a_i .

In addition, let σ be the exit sequence in which these loads must be ordered once they are all injected onto the collector. The function $\sigma(u)$ provides the position of the load u in this sequence. Each load thus can be identified by a single "sequence" number, between 1 and n , corresponding to its position in the desired exit sequence (σ). This is how the loads will be identified hereafter. It should be noted that, in this case, it is easy to check if the sequence is achievable: each aisle simply needs to have an ordered list of loads identified by increasing sequence numbers that are not necessarily consecutive.

The set L' will now be specified, specifically combining the first expected loads in the final exit sequence, placed at the first position of aisles $a_1, a_2, \dots, a_{i_1}$, with i_1 being the aisle number containing the first load of this sequence σ . This set combines all the loads that can be the first load of the sequence to be injected onto the collector. Indeed, a load in the first position in an aisle placed upstream of the aisle comprising the first load of the sequence is likely to be injected first, while at the same time being located behind the first load in the final flow of the collector. However, a load in the first position in the aisles downstream of the aisle comprising the first load will never be injected onto the collector before this first load.

Figure 5 shows a diagram equivalent to that of figure 4 assuming the following achievable sequence:

$$\sigma = \{id21, id31, id22, id11, id23, id32, id33, id34, id12\}.$$

In this example, $n = 9$, therefore each load has a unique sequence number allowing it to be identified while at the same time quickly knowing its desired position in the sequence σ . Figure 5 immediately shows that the selected sequence is achievable, indeed, the content of $a_1 = \{4, 9\}$, the content of $a_2 = \{1, 3, 5\}$ and the content of $a_3 = \{2, 6, 7, 8\}$. In addition, the particular set $L_{i1} = \{id11, id21\} = \{4, 1\}$ is available.

5.1.3 Start dates and injection dates

By way of a reminder, each aisle corresponds to an injection flow at a given time, shown as a FIFO column attached to the collector. However, in the warehouse, it is possible that the load awaiting injection is at a distance from the collector. The start point denotes the point at which this load awaits the order to depart from its aisle (conveyor), illustrated by a point attached to the FIFO column, and injection point denotes the space of the collector that is the first to be touched by the load that is actually being injected onto this collector, illustrated by a point attached to the collector in figure 6.

In the diagrams of the aisle system, these two points coincide and are shown in black in figure 7. In reality, these points are often different, and the duration needed for a load of the flow i to transition from one point to the other is denoted Pt_i (see figure 6). The start date T_d of the load when it must leave its aisle (conveyor) is linked to its injection date T onto the collector (coinciding with the injection point) using the following simple formula: $T_d = T - Pt_i$. Consequently, without any loss of generality hereafter, only the calculation of the injection dates will be described.

5.2 DATA STRUCTURE

5.2.1 Division of the discharge system (collector): unit of time (time step)

The discharge system (or collector) will be considered to be a

system with time *steps* (like a sorter), also called *slots* or *positions*.

Figure 8 illustrates the division of the collector into time steps (positions or slots).

Thus, the aim of obtaining a maximum throughput with an ordered outgoing flow amounts to filling the steps of the collector, as shown in **figure 9**.

A *unit of time* will be considered that corresponds to this time step that sub-divides the collector, according to its speed of displacement. A unit of time, called "time step", corresponds to the duration needed for a point of the collector to move by exactly the physical distance of a position. The distance defined for this position corresponds to the size of a load, plus a safety distance. This safety distance must be adjusted according to the work requirements and in order to comply with the following condition (illustrated in **figure 10**).

The unit of time necessarily must be greater than the time taken for a load to be injected onto the collector from the time when part of the load touches the collector up until the time when the entire load is correctly placed on the collector (i.e., in the flow thereof).

5.2.2 Distance from the injection system

As illustrated in **figure 11**, Δ_i is defined as corresponding to the number of time steps of the collector (discharge system) between aisle a_i and the last aisle a_k , as can be seen in the following figures, with $k = 3$. As illustrated in **figure 12**, this number may not be an integer. $\Delta_k = 0$.

The time distance between two aisles a_i and a_j is denoted $\Delta_{i, j}$, with the second index being the aisle representing the location.

5.2.3 Associated Job Shop model

The following table is a summary of the notations of the Job Shop model.

5 [Table 2]

$\{J_u, u \in \{1, \dots, n\}\}$	Set of n jobs (tasks) associated with a model in Job Shop.
	The u^e load of the sequence σ is denoted $\sigma_u = a_i(j)$, occupies the j^e place of aisle a_i and is associated with job J_u
$\{M_1, \dots, M_k\}$	Set of k machines of the model in Job Shop. Aisle a_i is associated with the machine M_i .
$\{o_{u,i}, o_{u,i+1}, \dots, o_{u,k}\} \mid u \in \{1, \dots, n\}, \sigma_u = a_i(j)$	List of $k - i$ consecutive operations associated with job J_u , given that the operation $o_{u,i}$ must be performed on the machine M_i .

For each exit sequence σ , the system is modelled as a single task Job Shop model, with n jobs and k machines. Each load of the sequence σ_u (u^e load of this sequence) is associated with a job J_u and each aisle a_i is associated with a machine M_i . Each job J_u associated with a load contained in aisle a_i must follow an ordered list of $k - i + 1$ single operations $\{o_{u,i}, o_{u,i+1}, \dots, o_{u,k}\}$. An operation denoted $o_{u,i}$ is a single task of the job J_u to be specifically performed on the machine M_i . A job can be assigned to only one machine at a time and a machine can perform only one operation at a time.

The intention is to schedule these jobs on the machines in accordance with the order provided by their respective list of operations. The aim is to schedule them in order to minimise the total duration for completing all these jobs.

6. DETERMINING THE INJECTION DATE OF EACH LOAD, FOR A COLLECTOR ONLY PROCESSING THE INJECTION FLOWS

In this section, a collector will be considered that only processes the injection flows. The collector is completely empty and ready to retrieve the injection flows. Only the merging of these injection flows has to be controlled in order to correctly inject their loads in order to have a final flow that is fluid, continuous and corresponds to the desired exit sequence.

6.1 SOLUTION FOR CONSECUTIVELY PROPERLY DISTRIBUTED AISLES

For the sake of simplicity, it is assumed that the injection flows are consecutively distributed on the collector, in such a way that a time interval is needed in order for a load on the collector to transition from one injection zone to the next, as shown in figure 13. This simplified case corresponds to $\Delta_{i, i+1} = 1$, for all $i \in \{1, \dots, k-1\}$.

6.1.1 Modelling the system of job shop scheduling

Modelling the system as a problem of Job Shop scheduling will now be explained in this particular case.

Each of the n loads is associated with a job J_u , $u \in \{1, \dots, n\}$, with the numbering being in accordance with the sequence number of the load. Thus, if σ_u corresponds to the u^e load in the sequence σ , then its associated job is J_u . The operations of these jobs must be processed by a set of k machines $\{M_1, \dots, M_k\}$. An operation can be assigned to only one machine at a time. Each machine is associated with an aisle (i.e., an injection flow) that can perform only one operation at a time. By way of a reminder, these aisles are numbered $\{a_1, \dots, a_k\}$, from upstream to downstream (i.e., following the conveying direction of the collector).

Thus, a load σ_u that is being injected onto the collector from aisle a_i will pass in front of each aisle a_j , with j in $\{i, i+1, \dots, k\}$. This mechanism is represented by the fact that each job associated with a load of aisle a_i is made up of $k - i + 1$

single operations $\{o_{u,i}, o_{u, \{i+1\}}, \dots, o_{u,k}\}$ to be consecutively processed without waiting. Each operation must be scheduled on a specific machine. Thus, if the operation $o_{u,i}$ (processed by the machine M_i) starts at the time t , the operation $o_{u, \{i+1\}}$ (processed by the machine $M_{\{i+1\}}$) starts at the time $t+1$, etc., and, finally, the operation $o_{u,k}$ (processed by the machine M_k) starts at the time $t+k-i$.

If the start time of the first operation $o_{u,i}$ of this job corresponds to the injection date in the collector, the start times of the following operations (i.e., $\{o_{u, \{i+1\}}, \dots, o_{u,k}\}$) represent the times at which the load is in front of the following aisles.

It should be noted that each task is made up of at least one operation on the last machine M_k (since each load passes in front of at least the last aisle a_k). The sequence of the operations scheduled on the machine M_k exactly corresponds to the order in which the loads will pass in front of the last aisle and their start date corresponds to the time when these loads will pass in front of this last injection aisle. For this reason, if load u is in front of load v in the given exit sequence, the last operation of u must begin before the last operation of v on M_k .

The aim of not having space on the collector amounts to not having any idle time between the operations of the machine M_k .

6.1.2 Principle of the algorithm for achieving the injection dates

The scheduling problem proposed above can be resolved by the following algorithm.

To begin, assuming that the first load of the sequence originates from aisle a_i , the operations $\{o_{1,i}, o_{1, \{i+1\}}, \dots, o_{1,k}\}$ associated with this job start at the times $t, t+1, \dots, t+k-i$.

Then, the operations of the job corresponding to the following load of the sequence will be initially scheduled by executing the operation to be processed on the machine M_k . This operation $o_{u, k}$ will be scheduled immediately after (without idle time) the
 5 previous operation programmed on M_k . By assuming that this operation begins at the time t , the operation on the machine $M_{(k-1)}$ (if it exists) is scheduled at the time $t-1$, and so on, until all the operations are programmed.

10 This procedure is iteratively applied to each of the following jobs of the given sequence.

The actual injection dates onto the collector are directly deduced from the start of the first operation of each job. The
 15 unit of time that is used is that defined above ("the time step" of the collector).

6.1.3 Calculating injection dates using a formula

20 Since a description has been provided as to how the algorithm for resolving the scheduling problem works, it is possible to deduce the formula used to calculate the injection dates.

The notations used in the following formula are those from the
 25 aforementioned summary table.

The origin of the time will be set to the earliest injection date of the one or more very first loads injected onto the collector. In the present system, the unoccupied location that
 30 will receive this load is therefore placed immediately in front of the aisle of this load at the time 0. Thus, the date will be calculated:

$$t_0 = \max_{\text{coru} = a_i(j) \in L, 1 \leq i \leq k, 1 \leq j \leq h_i} \{k+1-i-\sigma(u)\}$$

 35 earliest date on which the first load of the sequence passes in front of aisle a_k .

Proof: Let 0 be the date on which the injection of the loads

starts. Let t_0 be the date on which the first load of the sequence (i.e., σ_1) passes in front of aisle a_k . It should be noted that the first load injected onto the collector is necessarily a load at the head of an aisle, in an aisle located from aisle a_1 and that of σ_1 inclusive. Let $L' \subset L$ be this sub-set of loads.

Therefore, for any load $u = a_i(j) \in L'$, it is known that if it was injected on date x , it would pass in front of aisle a_k on the date $x + k - i$. $x \geq 0$, since a load cannot be injected before the date 0. However, by defining t_0 , the load u will pass in front of aisle a_k on the date $t_0 + \sigma(u) - 1 = x + k - i$. That is, $t_0 + \sigma(u) - 1 - k + i = x \geq 0$.

It is then possible to deduce therefrom that for all $u = a_i(j) \in L'$, $t_0 \geq k + 1 - i - \sigma(u)$. The earliest date is sought for checking all these constraints, therefore:

$$t_0 = \max_{u = a_i(j) \in L'} \{k + 1 - i - \sigma(u)\}$$

Algorithm 1 hereafter provides the injection dates $T(u)$, i.e., the dates of entry of each load u onto the collector. The unit of time used is the "time step" defined above. Algorithm 1 provides the formula for calculating these injection dates in the event that the flows are injected onto the collector at each consecutive location (see figure 14).

25

Algorithm 1: Calculating the injection dates without disruptive flow in a particular case

Require: $\sigma, k, L, a_i(j), h_i, \forall i \in \{1, \dots, k\}, \forall j \in \{1, \dots, h_i\}$

$$t_0 = \max_{u = a_i(j) \in L} \{k + 1 - i - \sigma(u)\}$$

for all $i \in \{1, \dots, k\}$ **do**

for all $j \in \{1, \dots, h_i\}$ **do**

 set $u = a_i(j)$

$$T(u) = t_0 + \sigma(u) - 1 - (k - i)$$

end for

end for

return $T(u), \forall u \in L$

30 **6.1.4 Example**

For scheduling, the Gantt chart is a very useful tool for viewing the operations forming a job over time. This tool allows the progress of the work of each machine to be graphically shown and visually shows the solution provided for the present problem.

5 In addition, the link can be seen between the results obtained by the algorithm allowing the injection dates to be achieved and the associated formula.

Let the configuration of the system be as follows (see figure 14): $k = 4$ aisles comprising two or three loads. Each load is identified by a unique sequence number by virtue of the exit sequence $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9)$. This is an achievable sequence.

15 In this simplified case, the Job Shop solution can be shown in a Gantt chart such that each row corresponds to a machine. The fact that each load consecutively passes by the machines is represented by the fact that each load sequence number appears in consecutive boxes from left-to-right in the table (see figure 20 15). In addition, the first appearance of each load indicates its injection date provided by the formula by setting the time scale to the time 0 defined above. By way of a reminder, the date 0 corresponds to the time of the first possible injection of a load. The date t_0 even can be placed according to its definition and can be found by a calculation. By way of a 25 reminder, t_0 is the date on which the first load of the sequence (σ_1) passes in front of aisle $k = 4$, namely number 1 when the operation of job 1 is carried out by the machine M_4 .

30 The desired sequence is found, without an empty space, on machine M_4 and the injection date of each load is provided by reading the box of the first appearance (from left-to-right) of the sequence number identifying this load.

35 6.2 SOLUTION FOR AISLES WITH ANY DISTRIBUTION

Now that the simplified case has been seen, the general case of the collector that processes only the injection flows will now

be considered. The flows are therefore no longer consecutively injected for each time step of the collector, but are randomly distributed. The distances between aisles are known. It is the travel time between the point of injection of aisle a_i and the last aisle a_k that is of particular interest, it will be denoted $\Delta_i = \Delta_{i, k}$ (as already defined above).

6.2.1 Calculating injection dates using a formula

Algorithm 2 hereafter takes into account this notion of any time distance between aisles and thus addresses the general case of the collector dedicated to injection flows.

The date 0 will again be considered to be the date of the first possible injection. This time, a load u of aisle a_k injected on the date x arrives in front of aisle a_k on the date $x + \Delta_i$. Thus, the earliest date is obtained at which the first load of the sequence passes in front of aisle a_k using the following formula:

$$t_0 = \max_{u = a_i(j) \in L} \{ \Delta_i + 1 - \sigma(u) \}.$$

Algorithm 2: Calculating injection dates without disruptive flow in a general case

Require: $\sigma, k, L, a_i(j), \Delta_i, h_i, \forall i \in \{1, \dots, k\}, \forall j \in \{1, \dots, h_i\}$

$$t_0 = \max_{u = a_i(j) \in L} \{ \Delta_i + 1 - \sigma(u) \}$$

for all $i \in \{1, \dots, k\}$ **do**

for all $j \in \{1, \dots, h_i\}$ **do**

 set $u = a_i(j)$

$$T(u) = t_0 + \sigma(u) - 1 - \Delta_i$$

end for

end for

return $T(u), \forall u \in L$

Figure 46 shows a flowchart of a method for merging k incoming flows in one outgoing flow, according to a first particular embodiment based on algorithm 2. The method is executed at least once by the control system. A given execution is performed at a time T_b and comprises the following steps:

- obtaining (step 461) a set L comprising n payloads

distributed at the time T_b over the k aisles and to be injected onto the collector to form an output sequence σ , each of the n payloads being identified by a unique sequence number within the output sequence σ , each of the k aisles containing an ordered set, according to an upward direction of the sequence number, of h_i payloads to be injected one by one onto the collector;

- calculating (step 462) a date t_0 on which the first payload σ_1 of the output sequence σ passes in front of aisle a_k ;
- calculating (step 463) a date of injection onto the collector for each of the n payloads of the set L , according to the following formula:

$$T(u) = t_0 + \sigma(u) - 1 - \Delta_i,$$

with:

- $u = a_i(j) \in L$, being a payload of the set L and originating from the j^{th} position in aisle a_i , $i \in \{1, \dots, k\}, j \in \{1, \dots, h_i\}$;
- $\sigma(u)$ being the sequence number of the payload u in the output sequence σ ;
- controlling (step 464) the collector and the k aisles, for an injection of the n payloads onto the collector in accordance with the n injection dates $T(u)$, $\forall u \in L$.

6.2.2 Example

Using an example similar to that of paragraph 6.1.4, yet this time spacing apart the aisles.

Let the configuration of the system be as follows (see **figure 16**): $k = 4$ aisles comprising two or three loads. Each load is identified by a unique sequence number as a function of the exit sequence $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9)$. This is an achievable sequence. The distance is known from each aisle to the last one: $\Delta_1 = 12$, $\Delta_2 = 7$, $\Delta_3 = 4$ and $\Delta_4 = 0$.

Figure 17 provides the representation of the solution in the form of a Gantt chart. This time, two consecutive machines must be spaced apart by as many rows as there are "time steps" between the two corresponding consecutive aisles. For this reason, the

solution is provided in a table with 13 ($= \Delta_1 + 1$) rows. By means of a calculation, then
 $t_0 = \max_{u \in \{1,2\}} \{\Delta_i + 1 - \sigma(u)\} = \max(7, 11) = 11$, which is actually found by definition at time 11 on the graph.

5

In this case, for ease of comprehension, an example is provided with inter-aisle distances that are integer distances, but it is possible to take non-integer distances in numbers of "time steps".

10

6.3 DESCRIPTION OF A PARTICULAR EMBODIMENT

6.3.1 List of elements to be combined

15

Each injection flow (incoming flow) is represented by an aisle containing a list of loads in a FIFO order. The number of aisles can vary from 2 to k , and the list of aisle i contains h_i loads (h_i can be zero). There are, in all, n loads to be merged into the collector. Taking, for example (see figure 18):

$k = 2$, $h_1 = h_2 = 5$ and $n = 10$.

20

In addition, in order to remain with the general case, the resolutions of the general case will be taken, with $\Delta_1 = 1$ and $\Delta_2 = 0$.

25

6.3.2 Construction of a default sequence

When the final sequence of the loads on the collector is of no importance, an achievable sequence ordering the loads simply needs to be defined in any manner.

30

Various ways can be considered for defining a "default" exit sequence:

35

- alternately exiting a load from each aisle,
 $\sigma = (id_5, id_{10}, id_4, id_9, id_3, id_8, \dots, id_6)$;
- exiting the loads as a function of their arrival in the injection flow;
- taking the first load from each randomly drawn aisle;
- etc.

6.3.3 A sequence provided by a disorder minimisation calculation

Figure 19 illustrates an example of a sequence ($\sigma' = (8, 4, 5, 2, 7, 6, 1, 9, 3, 10)$) resulting from a disorder minimisation calculation. The indicated numbers are not sequence numbers within the final sequence, but order numbers relative to customer orders to be picked.

This sequence σ' can be seen as a given exit sequence for the algorithm that calculates the injection dates. This sequence is necessarily achievable and minimises the disorder. A maximum throughput is obtained with minimum disorder for the flow exiting the collector.

Since it is always possible to assign a single sequence number from 1 to n to all the loads, corresponding to their position in the final sequence σ , the loads will be displayed with these sequence numbers in order to identify them and fully characterise them (see **figure 20**). Figures 19 and 20 are equivalent. The exit sequence $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$ allows the increasing order of the loads to be clearly seen on each injection flow (proof of the feasibility of the sequence).

This configuration will be kept in order to continue with the example.

6.3.4 Construction of t_0

In the present example, L' consolidates the loads numbered 1 and 5.

The maximum between $\{\Delta_2 + 1 - \sigma(1), \Delta_1 + 1 - \sigma(5)\}$ is sought, and $t_0 = 0$ is obtained.

6.3.5 Calculating injection dates (using a formula)

$$T(1) = t_0 + \sigma(1) - 1 - \Delta_2 = 0$$

$$\begin{aligned}
T(2) &= t_0 + \sigma(2) - 1 - \Delta_2 = 1 \\
T(3) &= t_0 + \sigma(3) - 1 - \Delta_2 = 2 \\
T(4) &= t_0 + \sigma(4) - 1 - \Delta_2 = 3 \\
T(5) &= t_0 + \sigma(5) - 1 - \Delta_1 = 3 \\
T(6) &= t_0 + \sigma(6) - 1 - \Delta_1 = 4 \\
T(7) &= t_0 + \sigma(7) - 1 - \Delta_1 = 5 \\
T(8) &= t_0 + \sigma(8) - 1 - \Delta_1 = 6 \\
T(9) &= t_0 + \sigma(9) - 1 - \Delta_1 = 7 \\
T(10) &= t_0 + \sigma(10) - 1 - \Delta_2 = 9
\end{aligned}$$

6.3.6 Explanation of the result in images

5 With the injection dates $T(u)$ being provided in the unit of time (i.e., the time step of the collector), this visually amounts to seeing the collector move by one step for each unit of time. When viewing an image of the system for each unit of time, each load u will appear for the first time on the collector on the
10 date $T(u)$ (its injection date).

The configuration of the collector at the time 0 (see figure 21) corresponds to the first real injection of a load, in this case load 1 (since $t_1 = 0$), even if this will not always be the case.
15 Indeed, it should be noted that the first load of the sequence is not necessarily the first load to be injected onto the collector.

Load 2 must be injected at the time 1, which means that it is
20 on the collector after the collector moves by one step, as shown in figure 22.

The load 3 is injected at the time 2, as shown in figure 23.

25 The loads 4 and 5 are injected on the date 3, as shown in figure 24.

And so on.

30 7. BATCH SEQUENCE

The injection dates are calculated based on a given state of the system. With the system developing over time, a first possibility involves repeating this calculation each time a new load reaches the system as it does not yet have its definite injection date. However, instead of also regularly launching the calculation algorithm, which is not advantageous for various reasons (notably the consumption of calculation resources), it is preferable to proceed in batches. This type of dynamic batchwise operation is explained in this section.

7.1 WORK BATCH

The intention is to have a continuous and optimal flow (outgoing flow) on the collector throughout a workday. It has been shown above how to define an optimal injection date (in order to achieve the stated aim) on the loads present in the aisles at a given, set time. This aim now needs to be achieved for all the loads arriving via the injection flows at any given time. The present invention proposes polling the injection date calculation algorithm at the time necessary for obtaining fluid and continuous filling of the collector.

To this end, this section provides a definition of how to fill the FIFO lists of each aisle for the configuration of the system at a set time. Then, a definition of a batch is provided and then, finally, an explanation is provided of the sequence for processing two batches without leaving any empty space on the collector.

7.1.1 Filling FIFO lists of the aisles

Each aisle contains a list of loads with a unique sequence number for each load. These sequence numbers of the loads necessarily increase per aisle, but are not necessarily consecutive (due to the definition of the achievable sequence to be provided).

Figure 25 illustrates an example of FIFO lists associated with

three aisles.

Figure 26 illustrates the final achievable sequence expected by the collector.

5

By way of a reminder, each aisle corresponds to an injection flow, shown as a FIFO column attached to the collector, with its injection point being defined by a black dot on the diagrams.

10 The FIFO list of the loads taken into account in an aisle for a set configuration of the system orders the loads on hold at the injection point of this aisle, in accordance with the following conditions:

- the loads taken into account during the analysis are on
15 hold at the injection point (shown by a black dot in the following figures) (**figure 27**: first load of the flow attached to the injection point and therefore forming part of the FIFO list; **figure 28**: first load of the flow not attached to the injection point and therefore not forming part of the FIFO list);
- 20 • the loads taken into account during the analysis are on "continuous" hold behind a first load forming part of the FIFO list (**figure 29**: second load on continuous hold behind a first load forming part of the FIFO list; **figure 30**: load on continuous hold behind a first load not forming part of the FIFO list).

25

7.1.2 Definition of batches

A work batch corresponds to the set of loads of the FIFO lists of all the aisles (it is the set L of n loads). However, these
30 loads correspond to the static picture of the present system at a given time, in accordance with the rules of the preceding paragraph. The loads present in the system at this time, which do not comply with the filling rules of the FIFO lists, will be subsequently assigned to another batch.

35

Once all the loads of the batch are assigned, the injection dates for this batch are calculated. These loads are injected over time, in accordance with the previously calculated

injection dates. The system develops over time, yet without polling the injection date calculation algorithm again, although new loads are eligible under the filling rules of the FIFO lists or even if new loads appear in the system.

5

All these injection dates are calculated in order to successfully place each load of the batch in its reserved place on the conveyor, in accordance with the order of the desired final sequence, while minimising the empty spaces. Let p designate the last place reserved for the last load of this batch provided by the order of the sequence. After this place p , the collector is empty. It is when this last place p (which still can be empty) passes in front of the first injection flow (first aisle) that a new batch will be defined.

15

The following batch will be built in the same way, but on the new static picture of the system at this set time (with, as its first unoccupied location, the step of the collector attached to this last place p).

20

It should be noted that, in a particular implementation, if a new batch is defined before the preceding batch has been completely executed, the last loads of the exit sequence of the preceding batch can be considered for the new batch (following batch). In this case, if a load at position i in the exit sequence is placed back into the following batch, then so too are all the loads with positions greater than i . However, it is worthwhile re-using loads only if the desired final sequence has changed. In this case, the last considered place p will be the place of the last load not re-used in the following batch. For the loads that are re-used, the injection dates already defined in the preceding batch will be obsolete and replaced by the injection dates calculated for the following batch.

30

35

In the example illustrated in **figure 31**, the first batch will be made up of the following loads: 1, 2, 3, 4, 5, 6, 7, 8, 9. The following batch could be: xx, yy or 9, xx, yy, . . . , or even 7, 8, 9, xx, yy, . . . (but not 8, xx, yy, nor 7, 9, xx,

yy).

7.2 DYNAMIC PROCESSING OF THE BATCHES

5 Each batch is processed continuously and in series (batch1, batch2, etc.). Processing a batch is a static calculation of all the injection dates of the loads forming this batch. The linking and the building of these batches allows an overall solution to be provided for the problem of merging an incoming flow in an
10 outgoing flow in warehouses, which is a dynamic problem.

7.2.1 Calculating injection dates of a first batch

Figure 32 illustrates an example of a first batch for an exit
15 sequence $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$, with loads 2, 6, 7 and 8 (for which the injection dates are denoted Time 4, Time 8, Time 9 and Time 10, respectively) on aisle a_3 , loads 1, 3 and 5 (for which the injection dates are denoted Time 0, Time 2 and Time 4, respectively) on aisle a_2 and loads 4 and 9 (for which
20 the injection dates are denoted Time 0 and Time 5, respectively) on aisle a_1 .

7.2.2 Injection of the batch over time

25 As long as the time step of the collector awaiting the highest sequence number (in this case No. 9, surrounded by a circle in figures 32 and 33) has not arrived in front of the injection aisle that is furthest upstream (i.e., aisle a_1), the following batch is not processed. Processing the loads of the first batch
30 continues over time, with these loads being injected on the previously calculated injection date, as shown in the upper part and the lower part of figure 33.

7.2.3 Formation of the following batch and static calculation 35 of the future injection dates thereof

When the last marked step of the last load of the sequence of the first batch arrives in front of aisle a_1 , this signals the

construction of the following batch. This batch is formed by all the valid loads of each FIFO list of each aisle according to the rules explained above.

5 **Figure 34** illustrates the formation of the following batch, made up of six new loads that have appeared on the aisles and are numbered 1 to 6 (sequence numbers of the new exit sequence desired for the new batch).

10 The algorithm for calculating static injection dates is applied to this new batch by considering the unoccupied part of the collector (to the right of the place reserved for load number 9 of the preceding batch). The injection dates are calculated as a function of a time scale, the origin of which corresponds to
15 the first injection of a load of the new batch.

The configuration of the static system, taken at "Time 5" as shown in figure 34, is entered (as input data) in the algorithm. This configuration is defined as follows: $k = 3$ aisles comprising
20 six new loads that have appeared (belonging to the new batch). Each new load is identified by a unique sequence number as a function of the new output **sequence of this new batch**: $\sigma = (1, 2, 3, 4, 5, 6)$. This is an achievable sequence. The time distances relative to the last aisle are known:

25 $\Delta_1 = 6, \Delta_2 = 3$ and $\Delta_3 = 0$.

The date 0 is set by simultaneously injecting two new loads numbered 1 and 4 (into aisles a_2 and a_1 , respectively). From this time origin, the algorithm provides an injection date $T(u)$ for
30 each of the six new loads u of this new batch.

$L' = \{1, 4\}$ and $t_0 = \max\{\Delta_1 + 1 - \sigma(4), \Delta_2 + 1 - \sigma(1)\} = \max\{3, 3\} = 3$

$$T(1) = t_0 + \sigma(1) - 1 - \Delta_2 = 0$$

$$T(2) = t_0 + \sigma(2) - 1 - \Delta_2 = 1$$

$$T(3) = t_0 + \sigma(3) - 1 - \Delta_3 = 5$$

$$T(4) = t_0 + \sigma(4) - 1 - \Delta_1 = 0$$

$$T(5) = t_0 + \sigma(5) - 1 - \Delta_2 = 4$$

$$T(6) = t_0 + \sigma(6) - 1 - \Delta_1 = 2$$

7.2.4 Link between the injection dates of a batch and the real time that elapses

5 Each load of each batch will be given an injection date as a function of a very precise time origin relative to the static system. These injection dates must be placed in the dynamic configuration at the right time in order to be consistent with the dynamic solution.

10

A time scale common to all the batches allows the injections of all the loads to be linked in the dynamic system. Its unit of time is the "time step" of the collector, as described above. Let "Time" be the time that elapses from this common time scale, with Time 0 representing the start date of operations in the system.

15

The injection dates provided by the algorithm must be correctly repositioned in this common time scale. To this end, the time match needs to be found, in the dynamic system, of the situation described at date 0 of the batch. In other words, a match needs to be found between the date 0 of the batch and the Time of the common scale that actually allows the first injection of the load of this batch. All the injection dates of this batch then need to be shifted by exactly this time.

25

To this end, the Time (also denoted "Tb") needs to be known at which polling of the algorithm was launched and the (or at least one of the) first load or loads of the batch to be injected onto the collector as soon as possible, called σ^* . The date 0 of the batch corresponds to the action of injecting this load $\sigma^* = a_i(j)$, and it is actually carried out by the dynamic system exactly $\sigma(\sigma^*) + \Delta_{1,*}$ time steps after polling.

30

Returning to the present example, the following needs to be added to each injection date calculated by the algorithm for the new batch (for all $u \in L, T(u)$): the actual date (algorithm polling date, in this case Time 5 (i.e., $T_b = 5$), plus the

35

position of the first injected load of the new batch in the new sequence (in this case, it is load 1, therefore $\sigma(1) = 1$ needs to be added), plus the distance between aisle a_1 and that containing this first injected load (in this case $\Delta_{1,2} = 3$). By noting that load 4 of the new batch is also firstly injected onto the collector (at the same time as load 1 of the new batch), it would have been possible to select it and add $\sigma(4) + \Delta_{1,1} = 4$; this really amounts to being the same as adding $\sigma(1) + \Delta_{1,2} = 1 + 3 = 4$.

10

For the dynamic application, a linear transformation is performed of the injection time, as follows:

$$\begin{aligned} T(1) + 5 + 4 &= 0 + 5 + 4 = 9 \\ T(2) + 5 + 4 &= 1 + 5 + 4 = 10 \\ T(3) + 5 + 4 &= 5 + 5 + 4 = 14 \\ T(4) + 5 + 4 &= 0 + 5 + 4 = 9 \\ T(5) + 5 + 4 &= 4 + 5 + 4 = 13 \\ T(6) + 5 + 4 &= 2 + 5 + 4 = 11 \end{aligned}$$

15 In this case, the earliest date of the first load of the new sequence passing in front of a_k will be exactly $Tb + \sigma(u^*) + \Delta_1 - \Delta_{i^*} + t_0$, with $u^* = a_{i^*}(1)$ being the first load of the new sequence to be injected onto the collector (not necessarily being the first load of the new sequence). However, it is known
20 that t_0 is a maximum that is achieved exactly by this load u^* , namely

$$t_0 = \max_{u = a_i(j) \in L} \{ \Delta_i + 1 - \sigma(u) \} = \Delta_{i^*} + 1 - \sigma(u^*)$$

Thus, it is known that the first load of the new sequence will
25 pass in front of a_k at $Tb + \sigma(u^*) + \Delta_1 - \Delta_{i^*} + t_0 = Tb + \sigma(u^*) + \Delta_1 - \Delta_{i^*} + \Delta_{i^*} + 1 - \sigma(u^*) = Tb + 1 + \Delta_1$, in the common time scale.

An equivalent way of providing the injection times in the common time scale is, by definition, to provide the algorithm with
30 $t_0 = Tb + 1 + \Delta_1$, when knowledge of the system is obtained exactly at a time $Tb = T(u_{last}) - \Delta_1 + \Delta_{i^{last}}$, with $u_{last} = a_{i^{last}}(j^{last})$ being the last load of the sequence of the preceding time.

In the above example, the time $Tb = 5$ exactly corresponds to the condition and therefore the algorithm could have been polled with $t_0 = Tb + 1 + \Delta_1 = 5 + 1 + 6 = 12$ (rather than 3) and the injection times expressed in the common time scale would have been available directly.

Ultimately, the present system can be represented as shown in figure 35. In the common time scale:

- the injection times of the loads 6, 7 and 8 of the preceding batch are denoted Time 8, Time 9 and Time 10, respectively; and
- the injection times of loads 1, 2, 3, 4, 5 and 6 of the new batch are denoted Time 9, Time 10, Time 14, Time 9, Time 13 and Time 11, respectively.

8. DETERMINING INJECTION DATES FOR THE LOADS, FOR A COLLECTOR PROCESSING ONE OR MORE DISRUPTIVE FLOWS

It is assumed that there is now at least one disruptive flow (of disruptive loads) already present on the collector. These disruptive loads do not form part of any of the injection flows (incoming flows); consequently, they are not ordered in the desired exit sequence of the injection flows. These disruptive loads disrupt the introduction of the payloads contained in the injection flows since they occupy places on the collector. These obstructions that block the injection of the payloads from time to time need to be taken into account. In addition, if these disruptive loads disappear from the collector between the injection aisles, this can create empty spaces in the outgoing flow of the collector. The aim is to always comply with an exit sequence while maximising the throughput of the collector and thus minimising these empty spaces.

In order to achieve this aim, lists of invalid dates will be used that are to be used with the Job Shop modelling described above.

The following table provides new notations needed for the remainder of the description (summary of notations for the

disruptive flows).

The following table provides new notations for the disruptive flows.

5

[TABLE 3]

F	Disruptive flow
n'	Total number of disruptive loads
$l(p)$ or $l_p, p \in F$	Number of the last aisle in front of which the disruptive load p passes
$g(p)$ or $g_p, p \in F$	Number of the first aisle in front of which the disruptive load p passes
$U_i, i \in \{1, \dots, k\}$	List of prohibited dates for a payload injection by aisle a_i
σ'	Ordered list of disruptive loads
σ'_p	The p^{th} disruptive load of the disruptive flow
$\sigma'(u, v)$	Duration that separates the disruptive load u and the disruptive load v in the disruptive flow, in number of "steps" at a given time
t_r	Date of arrival of the first disruptive load of the disruptive flow in front of aisle a_1 , in the particular case where this load effectively passes in front of this aisle (link with δ_p)
δ_p	Distance between the disruptive load p and the last aisle a_k on date 0
$start_p$	Date on which the disruptive load p is "fictitiously" injected in front of aisle a_1 . (Fictitiously if $g_p > 1$ since in this case the load p will not pass in front of aisle a_1)

8.1 MODELLING THE SYSTEM WITH A NON-CONTROLLED DISRUPTIVE FLOW

10

The disruptive loads of the disruptive flow block the injection

of a payload onto the collector when they pass in front of the injection aisles. Therefore, the invalid dates for an injection of a payload need to be calculated for each aisle. Now the aim is to calculate the new injection dates of the payloads by taking into account the non-controlled disruptive flow (over which, by definition, it is assumed that there is no control whatsoever). It should be noted that there nevertheless can be unoccupied spaces in the outgoing flow, and optimal minimisation of these unoccupied spaces nevertheless will be sought.

8.1.1 Creation of the list of invalid dates

It is possible to take into account a non-controlled disruptive flow, comprising disruptive loads that remain on the collector, thus passing in front of all the aisles and forming part of the final flow of the collector (mixed with the payloads of the injection flows). It is also possible to take into account a non-controlled disruptive flow, comprising disruptive loads present on the collector but disappearing between the injection aisles, thus creating unoccupied locations in front of some aisles but not all of them. It is also possible to take into account the case of a mixture of these types of non-controlled disruptive flows, as well as disruptive loads arriving and leaving between the aisles. Anything can be contemplated, and it is even possible to take into account an unspecified obstruction on one or more steps of the collector (with an obstruction not necessarily being a disruptive load but also being able to be a point on the collector that is damaged, reserved, etc.).

In any case, the idea is to take into account the disruptive loads and/or other obstructions in order to calculate the invalid dates for an injection of a payload of an aisle. The invalid dates are obtained by calculating the passage times of each obstruction (i.e., each disruptive load of the disruptive flow or any other obstruction) in front of this injection aisle. Let U_i denote the set of invalid dates where aisle a_i cannot inject a payload because of disruptive loads and/or any

unspecified obstructions of the time steps of the collector.

8.1.2 Idea behind the solution for consecutively properly distributed aisles

5

This Job Shop scheduling problem, with a list of invalid dates for each machine, can be resolved by following almost the same steps as those of paragraph 6.2.1.

10 Hereafter is the approach in the particular case of aisles that are properly distributed by conveyor steps:

To begin, assuming that the first load of the sequence originates from aisle a_i , the operations $\{o_{1,i}, o_{1,i+1}, \dots, o_{1,k}\}$ associated with this job start on the dates $t, t + 1, \dots, t + k - i$ from a date t that allows the dates $t, t + 1, \dots, t + k - i$ to be provided, respectively on each empty machine $\{M_i, M_{i+1}, \dots, M_k\}$ (i.e., without obstruction).

20 *Then, the operations of the job corresponding to the following load of the sequence will be initially scheduled by executing the operation to be processed on the machine M_k . This operation $o_{u,k}$ will be scheduled immediately after (without idle time) the previous operation programmed on M_k . By assuming that this*
25 *operation begins at a new time t (different from the instant t defined above for the calculations related to the first load) the operation on the machine $M_{(k-1)}$ (if it exists) is scheduled at the time $t - 1$, and so on, until all the operations are programmed. If an operation cannot be carried out on these*
30 *specified dates because of an obstruction, the scheduling of the (entire) job will begin again by shifting the execution time t of the operation to be processed on the machine M_k by one unit of time (i.e., by a location, step or slot).*

35 *This procedure is iteratively applied to each of the following jobs of the given sequence.*

The actual dates for injecting payloads onto the collector are

directly deduced from the start of the first operation of each job. The unit of time that is used is that defined above.

PLEASE NOTE: this method does not always provide a flow without empty spaces. The disruptive flow and/or the obstructions are non-controlled; therefore, the throughput is maximised, without ensuring maximality.

8.1.3 Idea behind the solution with a non-controlled disruptive flow in the general case

Based on the above paragraph, this is the approach followed in the general case for calculating the real injection dates when merging several flows comprising incoming flows (injection flows) and (at least) one non-controlled disruptive flow.

It is considered that it is possible to start to schedule jobs from the time 0. Remember that a particular sequential order needs to be followed when executing the operations of the machine M_k , while minimising the idle times of this machine.

To begin, let $t_0 (\geq 0)$ denote the earliest date on which the job associated with the first load of the sequence can be assigned to the machine M_k , if it did not have a disruptive flow. The loads are scheduled in the order provided by the desired sequence without taking into account the disruptive flow, and then the injection date is modified following this order. Indeed, it should be noted that scheduling a load σ_c only affects the loads $\sigma_{c'}$, such that $c' > c$.

For the job associated with a load u , its operation to be processed on the machine M_k will be scheduled first. This operation $O_{u,k}$ will be scheduled at t_0 if it is the load σ_1 , otherwise it will be scheduled immediately after (without idle time) the preceding operation programmed on M_k . Assuming that this operation on M_k starts at the time t , the operation on the machine $M_{(k-1)}$ (if it exists) is scheduled at the time $t - \Delta_{k-1}$. Then, the operation on the machine $M_{(k-2)}$ (if it exists) is

scheduled at the time $t - \Delta_k - 2$, and so on, until all the operations are programmed. If an operation cannot be carried out on these indicated dates due to an obstruction, the scheduling of the (entire) job will start again by shifting the time t of execution of the operation to be processed on the machine M_k by a unit of time (i.e., by a location, step or slot).

This procedure is iteratively applied on each of the following jobs of the given sequence.

10

It is necessary, if applicable, to shift the date 0 and the date t_0 in order to make them match their definition. Since shifting the date 0 must correspond to the first possible injection of loads onto the collector, it has repercussions on all the schedules (injection dates), which will be shifted in turn.

15

The real injection dates of the payloads onto the collector are directly deduced from the start of the first operation of each job. The unit of time used is that defined above.

20

PLEASE NOTE: the presence of a non-controlled disruptive flow does not always allow a flow without an empty space on the collector. With this method, the throughput of the collector is maximised, without ensuring maximality.

25

8.1.4 Algorithm in the general case

Algorithm 3 hereafter calculates the injection dates of the payloads by taking into account the disruptive flow expressed by dates for preventing any injection per aisle.

30

The date 0 corresponds to the first load injected onto the collector, at the earliest. The date t_0 is always the date on which the first load of the sequence passes in front of aisle a_k . The general formula for $i = 1 \dots k$:

35

$t_0 = \max, \{\Delta_i + 1 - \sigma(u)\}$ remains valid if there is no $u = a_i(j) \in L$, $j \leq h_i$ conflict with the disruptive flow when injecting the first load of the sequence.

Algorithm 3 provides the injection date of each load by taking into account the disruptive flow and by correcting the date t_0 , if necessary. The disruptive flow is taken into account by the prior calculation of the sets U_i , $i = 1 \dots k$ providing the dates of passage of each disruptive load of this disruptive flow in front of aisle a_i . These dates are therefore prohibited injection dates (also called invalid dates for a payload injection).

10 In this algorithm, the simplified notation " $u = \sigma_c = a_i(j) \in L$ " is used to define a load u corresponding to the c^{nth} load of the sequence σ , also defined by its position j in aisle a_i .

15 *Algorithm 3: Calculating injection dates with non-controlled disruptive flows*

Require: σ , Δ_i , U_i , $i \in \{1, \dots, k\}$ // Provide the U_i calculations to be carried out on a case-by-case basis

$$t_0 = \max_{u = a_i(j) \in L} \{\Delta_i + 1 - \sigma(u)\} \quad // \text{Theoretical calculation of } t_0$$

```

last = t0 // Theoretical date of the passage of  $\sigma_i$  in front of aisle  $a_k$ 
for c = 1 to n do
    set u = ai(j) =  $\sigma_c$ 
do { // Scheduling of the load  $\sigma_c$  while avoiding any obstructions
    check = true
    for all z = k à z = i, z - - do
        if (last -  $\Delta_z \in U_z$ ) then check = false;
    end for
    if (check = false) then {
        last = last + 1 ;
        if (c = 1) then t0 = t0 + 1; // t0 is modified in the event of disruptions
    } end if
} while (check = false) ;
T(u) = last -  $\Delta_i$  // Final injection date for the load
 $\sigma_c$ 
last = last + 1 // Theoretical date of the passage of  $\sigma_{c+1}$  in front
of aisle  $a_k$ 
end for
// Possible shift of the date 0
set tmin to min{ T(u),  $\forall u \in L$  }
if (tmin ≠ 0) then
    t0 = t0 - tmin
for u ∈ L do
    T(u) = T(u) - tmin
end for
end
return t0, T(u) ,  $\forall u \in L$ 

```

Figure 36 is a diagram of algorithm 3 above.

- 5 **Figure 47** shows a flowchart of a method for merging k incoming flows in one outgoing flow, according to a second particular embodiment based on algorithm 3 (illustrated in figure 36). The method is executed at least once by the control system. A given execution is performed at a time Tb and comprises the following
- 10 steps:
- obtaining (step 471) a set L comprising n payloads distributed at the time Tb over the k aisles and to be injected

onto the collector to form an output sequence σ , each of the n payloads being identified by a unique sequence number within the output sequence σ , each of the k aisles containing an ordered set, according to an upward direction of the sequence number,

5 of h_i payloads to be injected one by one onto the collector;

- for each aisle a_i , $i \in \{1, \dots, k\}$, obtaining (step 472) a set U_i , of invalid dates for an injection of one of the payloads of aisle a_i onto the collector due to a non-controlled disruptive load or another obstruction of a time step of the collector;

10 - by not taking into account the sets U_i , $t \in \{1, \dots, k\}$, calculating (step 473) a date t_0 on which the first payload σ_1 of the output sequence σ passes in front of aisle a_k ;

- for the first payload σ_1 of the output sequence σ , assuming that the first payload σ_1 originates from aisle a_i (step 474):

15 a) initialising t with t_0 ;

b) if none of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, calculating an injection date of the first payload σ_1 onto the collector with the following formula:

20 $T(u) = t - \Delta_i$,
with $u = \sigma_1 = a_i(j)$;

c) if one of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to one of the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, incrementing t by one unit and repeating steps b) and c) with

25 the new value of t ;

- for each following payload σ_c , of the output sequence σ , $c \in \{2, \dots, n\}$, assuming that the following payload σ_1 originates from aisle a_i (step 475):

a') incrementing t by one unit, with t used to calculate the

30 injection date of the preceding payload σ_{c-1} ;

b') if none of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to the sets $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, calculating an injection date of the following payload σ_c onto the collector with the following formula:

35 $T(u) = t - \Delta_i$,
with $u = \sigma_c = a_i(j)$;

c') if one of the dates $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ respectively belongs to one of the sets $U_k, U_{k-1}, U_{k-2},$

..., U_i , incrementing t by one unit and repeating steps b') and c') with the new value of t ;

- calculating (step 476) t_{min} according to the following formula:

5 $t_{min} = \min\{T(u), \forall u \in L\};$

- if $t_{min} > 0$, modifying (step 477) the n injection dates according to the following formula: $T(u) = T(u) - t_{min}$, $\forall u \in L$;

- controlling (step 478) the collector and the k aisles, for an injection of the n payloads onto the collector in accordance with the n injection dates $T(u), \forall u \in L$.

8.1.5 Description of a particular embodiment

15 With reference to a non-controlled disruptive flow already present on the collector, not taken into account by the desired sequence in the end because the disruptive loads will all disappear before the last injection aisle. This is the case, for example, in the context of figure 50 (GTP Intelis PTS automated system) with management of payload injection onto the collector

20 originating from the automated system (PTS), but also for returning loads that arrive on this collector in order to return into the automated system (PTS) (see paragraph 2.2 for the context of this possible application).

25

Figure 37 illustrates this configuration with disruptive loads (marked by a cross) of the return flow (forming the non-controlled disruptive flow) that return to the automated system (PTS).

30

According to this configuration, two observations can be made:

- no disruptive load of the return flow (and therefore of the disruptive flow) will pass in front of the last aisle; and

- the disruptive loads leaving the collector (towards the automated system (PTS)) in front of aisle a_1 pose no problem and therefore will not be studied. All the remaining loads will necessarily pass in front of aisle a_1 .

8.1.5.1 Calculating lists of invalid dates induced by the flow of disruptive loads

Taking a flow T of n' disruptive loads already present in the collector and having to exit the collector between the injection aisles. This disruptive flow, called return flow, corresponds to the loads that have to return to stock in the automated system (PTS). These disruptive loads actually must pass in front of one or more injection aisles. For each disruptive load P of this return flow (disruptive flow), g_p (and l_p , respectively) denote the number of the first aisle (and of the last aisle, respectively) in front of which the disruptive load p passes. Two methods will be seen that allow calculation of the invalid injection dates (U_i) , induced by the return flow as a function of the known data.

In this particular embodiment, $g_p = 1$ and $l_p < k$, for any load p of the return flow specific to the automated system (PTS). However, the general solution will be provided using the notations g_p and l_p .

- Either, in the aforementioned specific case of a return flow, the date of arrival (denoted t_r) of the first disruptive load of the return flow (disruptive flow) in front of aisle number a_1 (or even the machine M_1) is known. The order in which the disruptive loads of this return flow are distributed on the collector is also known (this sequence is called σ'), as is the duration that separates each disruptive load and another disruptive load of this disruptive flow; or σ'_p yields the p^{th} disruptive load of the return flow and $\sigma'(u,v)$ yields the duration that separates the disruptive load u and the disruptive load v in the disruptive flow in terms of the number of "steps".

Then, the times t_r , $t_r + \sigma'(\sigma'_1, \sigma'_2)$, ..., $t_r + \sigma'(\sigma'_1, \sigma'_{n'})$ are congested for the machine M_1 and form U_1 .

- For the machine M_i , $i = 2, \dots, k$, the congested dates are:
 $U_i = \{t_r + \sigma'(\sigma'_1, \sigma'_j) + \Delta_{i,1} \text{ if } l(\sigma'_j) \geq i, \forall j = 1, \dots, n'\}.$
- Another perspective would be to copy the "congested" (or

invalid) dates U_i of the machine M_i for the disruptive loads u that have $l(u) \geq i + 1$ and add to all these times $\Delta_{i, i+1}$ in order to obtain the list of congested dates U_{i+1} for the machine M_{i+1} .

5

- Or, in a more general context, the distance of each disruptive load p relative to the last aisle a_k is known on the date 0, which will be denoted δ_p . With these data, the U_i are calculated as follows:

For all $i \in \{1, \dots, k\}$ do $U_i = \emptyset$;

For all $p \in F$ do

For all $i \in \{g_p, \dots, l_p\}$ do

$U_i = U_i \cup \{\delta_p - \Delta_i\}$;

end for

end for

10 *return $U_i, \forall i = 1 \dots k$*

8.1.5.2 Examples

15 Further reference will now be made to the configuration of the example of paragraph 6.2.2.

Let $k = 4$ be aisles comprising two or three payloads. Each payload is identified by a unique sequence number as a function of the exit sequence $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9)$. This is an achievable sequence. The distances from each aisle to the last aisle are known:

$\Delta_1 = 12, \Delta_2 = 7, \Delta_3 = 4$ and $\Delta_4 = 0$.

25 As illustrated in figure 38, it is assumed that this time there are three time steps of the collector obstructed by disruptive loads A, B and C. It is known that these disruptive loads pass in front of some aisles and disappear from the collector thereafter. Loads A and B pass in front of aisles a_1 and a_2 . Finally, load C passes in front of all the aisles a_1, a_2 and a_3 .
30 Thus, $l_A = 2, l_B = 2$ and $l_C = 3$, as well as $g_A = g_B = g_C = 1$.

A state of the system is shown on the date 0, when the first

payload of the batch will be injected onto the first unoccupied place of the collector. In the diagram, an unoccupied position of the collector is empty, and the crosses indicate the positions taken by the payloads of the preceding batch and the letters denote the disruptive loads of the disruptive flow. The position of the disruptive flow on the date 0 allows the following information to be deduced: $\delta_A = 13$, $\delta_B = 14$ and $\delta_C = 18$.

Disruptive load B forces the payload 4 to time shift by one position and will leave an empty space in the final flow of the collector. Then, disruptive flow C forces the payload 7 to shift by yet another position, leaving an empty space further along the final flow of the collector.

Figure 39 is a Gantt chart illustrating the solution proposed in this case. It can be seen that if the disruptive flow was controlled (and not non-controlled), the first empty space could be filled by aisle a_3 or a_4 and the second empty space could be filled solely by aisle a_4 . Thus, the empty spaces would be avoided, and the throughput of the final flow (outgoing flow) from the collector would be increased.

8.2 MODELLING THE SYSTEM WITH A CONTROLLED DISRUPTIVE FLOW

It has been seen that when the disruptive flow was non-controlled, it led to empty spaces in the final flow of the collector. It is therefore worthwhile controlling the disruptive flow in order to avoid these empty spaces. Thus, the loads of the injection flows could be prioritised over the loads of the disruptive flows and it would be possible to make the disruptive flows pass into the unoccupied or free spaces left by the injection of the different aisles.

8.2.1 Formulation of the problem

It will now be assumed that it is possible to control the disruptive flow. This involves managing the injection of the disruptive loads of this disruptive flow, and deciding upon the

date (denoted $start_p$) on which the disruptive load p is injected in front of aisle a_1 so as not to disrupt the outgoing flow (outgoing flow formed by merging the injection flows), i.e., not creating any empty space in the outgoing flow.

5

8.2.2 Idea behind the solution algorithm

Firstly, the injection dates of the payloads originating from the aisles will be calculated, as if there were no disruptive flows (see Section 6). It is these injection dates that will provide the unavailable dates (U_i) for a disruptive load passing in front of an aisle (a_i). Then, taking into account the list of invalid dates, the injection of the loads of the disruptive flow will be calculated. To this end, in the order of arrival of the disruptive loads, each of their passages in front of aisles will be scheduled as early as possible, i.e., as soon as all these dates of passage into the respective machines fall on free dates.

8.2.3 General algorithm

Algorithm 4, described in detail hereafter, provides the injection dates of the payloads, as well as the control of the disruptive flow. The loads to be injected form the final flow of the collector. Their injection is decided without taking into account the disruptive flow. Then, when to let through the loads of the disruptive flow will be determined, yet this time by considering that the injected payloads fulfil the same role as the non-controlled flow in algorithm 3 above. Thus, it is possible to use the same technique as before, with the calculation of the sets V_i , $i = 1 \dots k$ providing the dates of passage of each payload of the injected flow in front of aisle a_i . Therefore, these dates are injection dates that are prohibited for the disruptive loads.

35

By way of a reminder, $\sigma'_{p,p'}$ provides the distance that separates the disruptive loads p and p' on the date 0 and t_r provides the date of arrival, without waiting, of the first load of the

disruptive flow in front of aisle number a_i .

In this algorithm, the simplified notation " $u = \sigma_c = a_i(j) \in L$ " is used to define a load u corresponding to the C^{nth} load of the sequence σ , also defined by its position j in aisle a_i .

Algorithm 4: Calculating injection dates of the loads of the controlled disruptive flows

```

Require:  $\sigma, \Delta_i, \Delta_{1i}, i \in \{1, \dots, k\}$ 
 $t_0 = \max_{u = a_i(j) \in L} \{\Delta_i + 1 - \sigma(u)\}$  // Theoretical calculation of  $t_0$ 
for all  $i \in \{1, \dots, k\}$  do
    for all  $j \in \{1, \dots, h_i\}$  do
        set  $u = a_i(j)$ 
         $T(u) = t_0 + \sigma(u) - 1 - \Delta_i$ 
        for all  $z \in \{i, \dots, k\}$  do
             $V_z = V_z \cup \{T(u) + \Delta_i - \Delta_z\}$ 
        end for
    end for
end for
set  $t = \delta_1 - \Delta_1$  // Arrival of the first disruptive load in front of  $a_1$ 

```

Figure 40 is a diagram of the above algorithm 4.

Figure 48 shows a flowchart of a method for merging k incoming flows in one outgoing flow, according to a third particular embodiment based on algorithm 4 (illustrated in figure 40). The method is executed by the control system at least once. A given execution is performed at an instant T_b and comprises the following steps:

- for each payload $u = a_i(j) \in L$ originating from an aisle a_i , calculating (step 481) the dates of passage of the payload in front of aisles a_i to a_k , knowing the injection date $T(u)$ of the payload onto the collector;
- for each aisle $a_i, i \in \{1, \dots, k\}$, obtaining (step 482) a set V_i containing each date of passage of one of the payloads in front of aisle a_i ;

- for each controlled disruptive load p of a disruptive flow controlled by virtue of control means located along the collector and in front of an aisle $a_{g(p)}$ in the direction of displacement of the collector, with $g(p) \in \{1, \dots, k\}$,
 $a_{g(p)}$ also being the first aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes (step 483):
 - a) initialising t with the date $\delta_1 - \Delta_1$, with δ_1 representing the possible arrival date, in the case of non-queuing, of the first controlled disruptive load in front of aisle a_k ;
 - b) calculating the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, $t + \Delta_{1,l(p)}$ of passage of the controlled disruptive load in front of aisles $a_{g(p)}$ to $a_{l(p)}$, with $a_{l(p)}$ being the last aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes;
 - c) if none of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to the sets $V_{g(p)}$, $V_{g(p)+1}, \dots, V_{l(p)}$, calculating a fictitious date of passage $start_p$ of the controlled disruptive load p in front of aisle a_1 , with the following formula: $start_p = t$;
 - d) if one of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to one of the sets $V_{g(p)}$, $\Delta_{1,g(p)+1}, \dots, V_{l(p)}$, incrementing t by one unit and repeating steps b), c) and d) with the new value of t ;
- controlling (step 484) the control means, for an injection on the date $start_p + \Delta_{1,g(p)}$ of each controlled disruptive load p in front of aisle $a_{g(p)}$. SEP

8.2.4 Description of a particular embodiment

The example common to all the previous cases will be used in order to be able to compare the final solutions.

Let $k = 4$ be aisles comprising two or three payloads. Each payload is identified by a unique sequence number as a function of the exit sequence $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9)$. This is an achievable sequence. The distances from each aisle to the last aisle are known:

$\Delta_1 = 12$, $\Delta_2 = 7$, $\Delta_3 = 4$ and $\Delta_4 = 0$.

The disruptive flow is made up of loads A, B, C, such that $l_A = 2$, $l_B = 2$,

5 $l_C = 3$, $g_A = g_B = g_C = 1$, $\delta_A = 13$, $\delta_B = 14$ and $\delta_C = 18$.

This time, the disruptive flow is controllable. **Figure 41** shows the control point of this disruptive flow, represented by a vertical rectangle immediately in front of aisle a_1 .

10

Figure 42 is an illustration, in the form of a Gantt chart, of the solution without taking into account the disruptive flow ("pure injection solution").

15 The solution of the algorithm taking into account the disruptive flow amounts to sliding the disruptive loads into the shaded (crossed) boxes, which are left empty by the pure injection solution. Load A is allowed to pass as soon as it arrives at the control point of the disruptive flow, while load B is retained
20 for a unit of time. This wait allows the disruptive load B to pass into the free spaces of the injections, immediately before the load 5 is injected. This is also the case with load C and load 8 of the sequence.

25 **Figure 43** is an illustration, in the form of a Gantt chart, of the solution proposed for controlling the disruptive flow (i.e., by choosing the placement of the disruptive loads). Loads B and C no longer form an empty space in the final flow of the collector. They have been interposed at strategic points so that
30 the empty space that they initially leave is then filled by the injection of a time-compatible load. By virtue of the control of the disruptive flow, it has been shown that the empty spaces on the collector were avoidable.

35 8.3 MODELLING THE SYSTEM WITH A NON-CONTROLLED DISRUPTIVE FLOW AND A CONTROLLED DISRUPTIVE FLOW

In the event that there is at least one non-controlled disruptive

flow and at least one controlled disruptive flow at the same time, the method proceeds as follows.

5 A differentiation is made between the non-controlled disruptive loads and the controlled disruptive loads as follows:

Let S be the set of s non-controlled disruptive loads to necessarily be allowed to pass into the injection zone of the aisles of the collector without being controlled, with each load
10 being identified by a unique sequence number corresponding to its position in the order of arrival (denoted σ'_s).

Let δ'_p designate the distance between each disruptive load $p \in S$ and the last aisle a_k on the date T_b and let g_p and l_p
15 respectively designate the number of the first and last aisle in front of which the disruptive load p must pass.

Finally, the sets $V_i, i = 1 \dots k$, provide the dates when passage in front of aisle a_i is unavailable for the payloads.
20

- Let F designate the set of f disruptive loads to be allowed to pass into the injection zone of the aisles of the collector that are controlled, with each load being identified by a unique sequence number corresponding to its position in the order of
25 arrival (denoted σ_f).

Let δ_p designate the distance from each disruptive load $p \in F$ to the last aisle a_k on the date T_b and let g_p and l_p respectively denote the number of the first and last aisle in front of which
30 the disruptive load p must pass.

Finally, the sets $V_i, i = 1 \dots k$, provide the dates when passage in front of aisle a_i is unavailable for the payloads.

35 Algorithm 5, which is described hereafter, provides the injection dates of the payloads, as well as the control of the disruptive flow, taking into account the non-controlled disruptive flow.

Algorithm 5: Calculating the injection dates of the payloads and of the loads of the controlled disruptive flow, taking into account the non-controlled disruptive flow

Require: $F, S, \sigma, \Delta, \Delta_{11}, a_i(j), \forall i \in \{1, \dots, k\}, \forall j \in \{1, \dots, h_i\}$

$t_0 = \max_{u = a_i(j) \in L'} \{ \Delta_i + 1 - \sigma(u) \}$ // Theoretical calculation of t_0

// Calculation of the U_i :

for all $p = 1$ **to** s **do**

for all $i \in \{g'_p, \dots, l'_p\}$ **do**

5

```

         $U_i = U_i \cup \{ \delta'_p - \Delta_i \}$  // Load P in front of  $a_i$ 
    end for
end for
// Calculation of the  $T(u)$ 
last =  $t_0$  // Theoretical date of the passage of  $\sigma_i$  in front of aisle  $a_i$ 
for u = 1 to n do
    set  $u = a_i(j)$ 
do { // Scheduling of the uth load of  $\sigma$ 
    check = true
    for all  $z = k$  à  $z = i$ ,  $z = \dots$  do
        if  $(last - \Delta_z \in U_z)$  then check = false;
    end for
    if (check = false) then {
        last = last + 1;
        if (u = 1) then  $t_0 = t_0 + 1$ ;
    } end if
} while (check = false) ;
     $T(u) = last - \Delta_i$  // Final injection date for the load u

    last = last + 1 // Following place in front of  $a_i$ 
end for
// Possible shift of the date 0
set  $t_{min}$  to  $\min\{ T(u), \forall u \in L \}$ 
if ( $t_{min} \neq 0$ ) then
     $t_0 = t_0 - t_{min}$ 
for  $u \in L$  do
     $T(u) = T(u) - t_{min}$ 
end for
endif
// Calculation of the  $V_z$  :
for u = 1 to n do
    set  $u = a_i(j)$ 
    for all  $z \in \{i, \dots, k\}$  do
         $V_z = V_z \cup \{ T(u) + \Delta_i - \Delta_z \}$ 
    end for
end for

```

```

for all  $z \in \{1, \dots, k\}$  do
     $V_z = V_z \cup U_z$ 
end for
// Calculation of the  $start_p$  :
set  $t = \delta_1 - \Delta_1$  // Arrival of the first disruptive load in front of  $a_1$ 
for all  $p = 1$  to  $f$  do // Load identified by its place in  $\sigma_i$ 
do {
     $check = true$ 
    for  $i = g_p$  to  $l_p$  do
        if  $(t + \Delta_1 - \Delta_i \in V_i)$  then  $check = false; t++;$  break;
    end for
} while  $(check = false)$  ;
 $start_p = t$ 
 $t = \text{Max}(t + 1, \delta_{p+1} - \Delta_1)$  // Possible waiting for the arrival of  $p+1$ 
end for
return  $t_0, T(u), \forall u \in L, start_p, \forall p \in F$ 

```

Figure 45 is a diagram of the above algorithm 5.

- 5 **Figure 49** shows a flowchart of a method for merging k incoming flows in one outgoing flow, according to a fourth particular embodiment based on algorithm 5 (illustrated in figure 45). The method is executed by the control system at least once. A given execution is performed at an instant T_b and comprises the
- 10 following steps:
- for each payload $u = a_i(j) \in L$ originating from an aisle a_i , calculating (step 491) the dates of passage of the payload in front of aisles a_1 to a_k , knowing the injection date $T(u)$ of the payload onto the collector;
 - 15 - for each aisle $a_i, i \in \{1, \dots, k\}$, obtaining (step 492) a set V_i containing, on the one hand, each date of passage of one of the payloads in front of aisle a_i and, on the other hand, the set U_i of invalid dates for an injection of one of the payloads of aisle a_i onto the collector;
 - 20 - for each controlled disruptive load p of a disruptive flow controlled by virtue of control means located along the collector and in front of an aisle $a_{g(p)}$ in the direction of

displacement of the collector, with $g(p) \in \{1, \dots, k\}$, $a_{g(p)}$ also being the first aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes (step 493):

- 5 a) initialising t with the date $\delta_1 - \Delta_1$, with δ_1 representing the possible arrival date, in the case of non-queuing, of the first controlled disruptive load in front of aisle a_k ;
- b) calculating the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, $t + \Delta_{1,l(p)}$ of passage of the controlled disruptive load in front of aisles
- 10 $a_{g(p)}$ to $a_{l(p)}$, with $a_{l(p)}$ being the last aisle, in the direction of displacement of the collector, in front of which the controlled disruptive load passes;
- c) if none of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to the sets $V_{g(p)}$, $V_{g(p)+1}, \dots, V_{l(p)}$,
- 15 calculating a fictitious date of passage $start_p$ of the controlled disruptive load p in front of aisle a_1 , with the following formula: $start_p = t$;
- d) if one of the dates $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}, \dots, t + \Delta_{1,l(p)}$ respectively belongs to one of the sets $V_{g(p)}$,
- 20 $\Delta_{1,g(p)+1}, \dots, V_{l(p)}$, incrementing t by one unit and repeating steps b), c) and d) with the new value of t ;
- controlling (step 494) the control means, for an injection on the date $start_p + \Delta_{1,g(p)}$ of each controlled disruptive load p in front of aisle $a_{g(p)}$.

25

9. CONCLUSION

Since travel time is critical for obtaining the highest possible production rate, knowing how to combine several incoming flows

30 on a collector is very important in the field of logistics. Using all the mechanical capacities of each system and therefore maximising the throughput of such a collector is crucial for avoiding times losses when order-picking, for example.

35 Optimal throughput has been achieved by injecting the loads as early as possible in a synchronised manner. Notably, the maximum capacities of the collector have been achieved when it is dedicated to the injection flows. In addition to the modelling

and the mathematical resolution, a formula has been extracted that allows direct calculation of the injection dates of each load. These results are provided in order to comply with a final exit sequence of the loads once they are all on the collector, thus allowing the optimisation of speed and sorting to be combined.

The management of several types of flows on the collector has also been considered, using the term "disruptive flow" to denote a flow different from the injection flow. The injection dates can be calculated using an algorithm that takes into account this disruptive flow when it is non-controlled. However, the throughput is optimal without guaranteeing maximality. Indeed, such a non-controlled disruptive flow does not allow recovery of a space left by a disruptive flow that would have exited the collector between the injection aisles.

A solution has also been proposed in which the disruptive flow is controlled. Another algorithm has been provided in order to make the controlled disruptive flow pass in such a way as to prioritise the maximum throughput of the collector once all the injection aisles have passed. Controlling this controlled disruptive flow allows a maximum throughput to be provided for the injection flows once on the collector when all the loads of the controlled disruptive flow disappear before the last injection aisle. Otherwise, the remaining disruptive flows are included in the final flow of the collector and are inserted into the sequence without spaces.

An algorithm has also been provided for managing at least one non-controlled flow (obstructions that are not necessarily disruptive loads) and at least one controlled flow (disruptive loads) at the same time.

10. EXAMPLE OF A CONTROL SYSTEM

The proposed solution is a method for merging, within a logistics warehouse, k incoming flows of payloads, respectively

transported by k conveyors, called aisles a_i , with $i \in \{1, \dots, k\}$, in one outgoing flow of payloads transported by another conveyor, called collector.

- 5 The merging method is executed by a control system. This is, for example, a central computer management system (also called WCS (Warehouse Control System)).

Figure 44 shows the structure of the control system 90 according to a particular embodiment of the invention. This control system comprises a volatile memory 92 (for example, a RAM), a processing unit 91 equipped, for example, with a processor and controlled by a computer program 930 stored in a read-only memory 93 (for example, a ROM or a hard drive).

15 On initialisation, the code instructions of the computer program are loaded, for example, into the volatile memory 92 and then executed by the processor of the processing unit 91, in order to implement the merging method of the invention (according to any one of the various embodiments described above). The processing unit 91 receives information 94 as input that relates to the incoming flows. The processor of the processing unit 91 processes the information 94 and generates instructions or commands 95 as output that are used to control (command) various elements included in the system, notably the aisles, the collector, the control means, etc.

This figure 44 illustrates only one particular way, among several possible ways, of carrying out the technique of the invention, in any one of its embodiments. Indeed, the control system can be equally implemented on a reprogrammable calculating machine (for example, a PC computer, a DSP processor, a microcontroller, etc.) executing a program comprising a sequence of instructions, or on a dedicated calculating machine (for example, a set of logic gates, such as an FPGA or an ASIC, or any other hardware module).

In the event that the control system is implemented with a reprogrammable calculating machine, the corresponding program (i.e., the sequence of instructions) can be stored in a detachable or non-detachable storage medium (such as, for
5 example, a floppy disk, a CD-ROM or a DVD-ROM), with this storage medium being partially or fully readable by a computer or a processor.

Patentkrav

1. Fremgangsmåde til sammenlægning i et logistiklager af k indgående strømme af nyttelaster, som transporteres af
5 henholdsvis k transportører, som kaldes gange a_i , hvor $i \in \{1, \dots, k\}$, til en udgående strøm af nyttelaster, der transporteres af en anden transportør, som kaldes samler, idet logistiklageret er således, at:
- de k gange er af typen "først ind først ud" og er fordelt
10 langs samleren og nummereret fra a_1 til a_k i samlerens bevægelsesretning, og
 - Δ_i er en tidsafstand mellem gangene a_i og a_k , udtrykt i tidsenheder, der hver svarer til et tidstrin for samleren, idet fremgangsmåden udføres mindst én gang af et kontrolsystem,
15 idet en givet udførelse foretages på et tidspunkt T_b og omfatter:
 - opnåelse (461) af et sæt L , der omfatter n nyttelaster, som er fordelt på tidspunktet T_b over de k gange, og som skal føres ind på samleren for at danne en udgangssekvens σ , idet
20 hver af de n nyttelaster identificeres af et unikt sekvensnummer inden for udgangssekvensen σ , idet hver af de k gange indeholder et sæt, indordnet efter en tiltagende rækkefølge af sekvensnummeret, af h_i nyttelaster, som én efter én skal føres ind på samleren,
 - 25 idet fremgangsmåden er kendetegnet ved, at den omfatter de følgende trin:
 - beregning (462) af en dato t_0 , på hvilken den første nyttelast σ_1 i udgangssekvensen σ passerer foran gangen a_k
 - beregning (463) af en dato for indføring på samleren for
30 hver af de n nyttelaster i sættet L i henhold til følgende formel: $T(u) = t_0 + \sigma(u) - 1 - \Delta_i$, hvor:
 - * $u = a_i(j) \in L$ er en nyttelast i sættet L og kommer fra den j 'te position i gangen a_i , $i \in \{1, \dots, k\}$, $j \in \{1, \dots, h_i\}$
 - * $\sigma(u)$ er sekvensnummeret for nyttelasten u i udgangssekvensen
35 σ
 - betjening (464) af samleren og de k gange til indføring af de n nyttelaster på samleren i overensstemmelse med de n indføringsdatoer $T(u)$, $\forall u \in L$.

2. Fremgangsmåde ifølge krav 1, hvorved $\Delta_{i,i}$ er en tidsafstand mellem gangene a_i og $a_{i'}$, udtrykt i tidsenheder, der hver svarer til et tidstrin for samleren, og som i øvrigt

5 omfatter de følgende trin:

- for hver nyttelast $u = a_i(j) \in L$, der kommer fra en gang a_i , beregning (481) af datoerne for passage af nyttelasten foran gangene a_i til a_k , når datoen $T(u)$ for indføring af nyttelasten på samleren er kendt

10 - for hver gang a_i , $i \in \{1, \dots, k\}$, opnåelse (482) et sæt V_i , der indeholder hver dato for passage af en af nyttelasterne foran gangen a_i

- for hver kontrolleret forstyrrende last p i en forstyrrende strøm, der styres ved hjælp af et styremiddel placeret langs samleren og før en gang $a_{g(p)}$ i samlerens bevægelsesretning, hvor $g(p) \in \{1, \dots, k\}$, idet $a_{g(p)}$ ligeledes er den første gang i samlerens bevægelsesretning, foran hvilken den kontrollerede forstyrrende last passerer (483):

15 a) initialisering af t med datoen $\delta_1 - \Delta_1$, hvor δ_1 angiver den mulige ankomstdato, hvis ikke i venteposition, for den første kontrollerede forstyrrende last foran gangen a_k

20 b) beregning af datoerne $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, ..., $t + \Delta_{1,1(p)}$ for passage af den kontrollerede forstyrrende last foran gangene $a_{g(p)}$ til $a_{1(p)}$, hvor $a_{1(p)}$ er den sidste gang i samlerens bevægelsesretning, foran hvilken den kontrollerede forstyrrende last passerer

25 c) hvis ingen af datoerne $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, ..., $t + \Delta_{1,1(p)}$ henholdsvis tilhører sættene $V_{g(p)}$, $V_{g(p)+1}$, ..., $V_{1(p)}$, beregning af en fiktiv dato $start_p$ for passage af den kontrollerede forstyrrende last p foran gangen a_1 med følgende formel:

$start_p = t;$

30 d) hvis en af datoerne $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, ..., $t + \Delta_{1,1(p)}$ henholdsvis tilhører et af sættene $V_{g(p)}$, $V_{g(p)+1}$, ..., $V_{1(p)}$, forøgelse af t med én enhed og gentagelse af trin b), c) og d) med den nye værdi af t

- betjening (484) af styremidlet til en indføring på datoen $start_p + \Delta_{1,g(p)}$ af hver kontrolleret forstyrrende last p foran

gangen $a_{g(p)}$.

3. Fremgangsmåde til sammenlægning i et logistiklager af k indgående strømme af nyttelaster, som transporteres af henholdsvis k transportører, som kaldes gange a_i , hvor $i \in \{1, \dots, k\}$, til en udgående strøm af nyttelaster, der transporteres af en anden transportør, som kaldes samler, idet logistiklageret er således, at:

- de k gange er af typen "først ind først ud" og er fordelt langs samleren og nummereret fra a_1 til a_k i samlerens bevægelsesretning, og

- Δ_i er en tidsafstand mellem gangene a_i og a_k , udtrykt i tidsenheder, der hver svarer til et tidstrin for samleren, idet fremgangsmåden udføres mindst én gang af et kontrolsystem, idet en givet udførelse foretages på et tidspunkt T_b og omfatter:

- opnåelse (471) af et sæt L , der omfatter n nyttelaster, som er fordelt på tidspunktet T_b over de k gange, og som skal føres ind på samleren for at danne en udgangssekvens σ , idet hver af de n nyttelaster identificeres af et unikt sekvensnummer inden for udgangssekvensen σ , idet hver af de k gange indeholder et sæt, indordnet efter en tiltagende rækkefølge af sekvensnummeret, af h_i nyttelaster, som én efter én skal føres ind på samleren,

idet fremgangsmåden er kendetegnet ved, at den omfatter de følgende trin:

- for hver gang a_i , $i \in \{1, \dots, k\}$, opnåelse (472) af et sæt U_i af ugyldige datoer for indføring af en af nyttelasterne fra gangen a_i på samleren på grund af en ukontrolleret forstyrrende last eller anden hindring af et tidstrin for samleren

- uden at tage hensyn til mængderne U_i , $i \in \{1, \dots, k\}$, beregning (473) af en dato t_0 , på hvilken den første nyttelast σ_1 i udgangssekvensen σ passerer foran gangen a_k

- for den første nyttelast σ_1 i udgangssekvensen σ , idet det antages, at den første nyttelast σ_1 kommer fra gangen a_i (474):

a) initialisering af t med t_0

b) hvis ingen af datoerne t , $t - \Delta_{k-1}$, $t - \Delta_{k-2}$, ..., $t - \Delta_i$

henholdsvis tilhører sættene $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, beregning af en dato for indføring af den første nyttelast σ_1 på samleren med følgende formel: $T(u) = t - \Delta_i$, hvor $u = \sigma_1 = a_i(j)$

c) hvis en af datoerne $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ henholdsvis tilhører en af sættene $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, forøgelse af t med én enhed og gentagelse af trin b), c) og d) med den nye værdi af t

- for hver efterfølgende nyttelast σ_c i udgangssekvensen σ , $c \in \{2, \dots, n\}$, idet det antages, at den efterfølgende nyttelast σ_c kommer fra gangen a_i (475):

a') forøgelse af t med én enhed, hvor t anvendes til at beregne datoen for indføring af den foregående nyttelast σ_{c-1}

b') hvis ingen af datoerne $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ henholdsvis tilhører sættene $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, beregning af en dato for indføring af den første efterfølgende nyttelast σ_c på samleren med følgende formel: $T(u) = t - \Delta_i$, hvor $u = \sigma_c = a_i(j)$

c') hvis en af datoerne $t, t - \Delta_{k-1}, t - \Delta_{k-2}, \dots, t - \Delta_i$ henholdsvis tilhører en af sættene $U_k, U_{k-1}, U_{k-2}, \dots, U_i$, forøgelse af t med én enhed og gentagelse af trin b') og c') med den nye værdi af t

- betjening (478) af samleren og de k gange til indføring af de n nyttelaster på samleren i overensstemmelse med de n indføringsdatoer $T(u)$, $\forall u \in L$.

4. Fremgangsmåde ifølge krav 3, hvorved trinnet til betjening (478) af samleren og de k gange til en indføring af de n nyttelast på samleren i overensstemmelse med de n indføringsdatoer $T(u)$, $\forall u \in L$ forudgås af følgende trin:

- beregning (476) af $t_{\min}$ i overensstemmelse med følgende formel: $t_{\min} = \min \{T(u), \forall u \in L\}$

- hvis $t_{\min} > 0$, ændring (477) af de n indføringsdatoer i overensstemmelse med følgende formel:

$$T(u) = T(u) - t_{\min}, \forall u \in L.$$

5. Fremgangsmåde ifølge krav 3 eller 4, hvorved $\Delta_{i,i}$ er en tidsafstand mellem gangene a_i og $a_{i'}$, udtrykt i tidsenheder, der hver svarer til et tidstrin for samleren, og som i øvrigt

omfatter de følgende trin:

- for hver nyttelast $u = a_i(j) \in L$, der kommer fra en gang a_i , beregning (491) af datoerne for passage af nyttelasten foran gangene a_i til a_k , når datoen $T(u)$ for indføring af nyttelasten på samleren er kendt

- for hver gang a_i , $i \in \{1, \dots, k\}$, opnåelse (492) af et sæt V_i , der indeholder dels hver dato for passage af en af nyttelasterne foran gangen a_i , og dels sættet U_i af ugyldige datoer for en indføring af en af nyttelasterne fra gangen a_i

på samleren

- for hver kontrolleret forstyrrende last p i en forstyrrende strøm, der styres ved hjælp af et styremiddel placeret langs samleren og før en gang $a_{g(p)}$ i samlerens bevægelsesretning, hvor $g(p) \in \{1, \dots, k\}$, idet $a_{g(p)}$ ligeledes er den første gang i samlerens bevægelsesretning, foran hvilken den kontrollerede forstyrrende last passerer (493):

a) initialisering af t med datoen $\delta_1 - \Delta_1$, hvor δ_1 angiver den mulige ankomstdato, hvis ikke i venteposition, for den første kontrollerede forstyrrende last foran gangen a_k

b) beregning af datoerne $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, ..., $t + \Delta_{1,l(p)}$ for passage af den kontrollerede forstyrrende last foran gangene $a_{g(p)}$ til $a_{l(p)}$, hvor $a_{l(p)}$ er den sidste gang i samlerens bevægelsesretning, foran hvilken den kontrollerede forstyrrende last passerer

c) hvis ingen af datoerne $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, ..., $t + \Delta_{1,l(p)}$ henholdsvis tilhører sættene $V_{g(p)}$, $V_{g(p)+1}$, ..., $V_{l(p)}$, beregning af en fiktiv dato $start_p$ for passage af den kontrollerede forstyrrende last p foran gangen a_1 med følgende formel: $start_p = t$

d) hvis en af datoerne $t + \Delta_{1,g(p)}$, $t + \Delta_{1,g(p)+1}$, ..., $t + \Delta_{1,l(p)}$ henholdsvis tilhører et af sættene $V_{g(p)}$, $V_{g(p)+1}$, ..., $V_{l(p)}$, forøgelse af t med én enhed og gentagelse af trin b), c) og d) med den nye værdi af t

- betjening (494) af styremidlet til en indføring på datoen $start_p + \Delta_{1,g(p)}$ af hver kontrolleret forstyrrende last p foran gangen $a_{g(p)}$.

6. Fremgangsmåde ifølge et hvilket som helst af kravene 1

til 5, hvorved datoen t_0 beregnes med følgende formel:

$$t_0 = \max_{u=a_i(j) \in L'} \{\Delta_i + 1 - \sigma(u)\},$$

hvor:

- $u = a_i(j) \in L'$ er en nyttelast i et sæt L' , der omfatter
 5 nyttelaster placeret i den første position af de gange, som
 i samlerens bevægelsesretning befinder sig fra gangen a_1 til
 gangen a_{i1} , der indeholder den første nyttelast σ_1 i
 udgangssekvensen σ .

10 7. Fremgangsmåde ifølge et hvilket som helst af kravene 1
 til 6, kendetegnet ved, at den udføres iterativt, idet hver
 ny udførelse foretages på et nyt tidspunkt T_b , der beregnes
 med følgende formel: $T_b = T(u_{last}) - \Delta_1 + \Delta_{ilast}$,
 hvor $u_{last} = a_{ilast}(j^{last})$ er en sidste nyttelast, der kommer fra
 15 den (j^{last}) 'te position i gangen a_{ilast} , i udgangssekvensen σ
 fra en foregående udførelse ved et foregående tidspunkt T_b .

8. Fremgangsmåde ifølge et hvilket som helst af kravene 1
 til 6, kendetegnet ved, at den udføres iterativt, idet hver
 20 ny udførelse foretages på et nyt tidspunkt T_b , der defineres
 som et tidspunkt, på hvilket ingen last i udgangssekvensen σ
 fra en foregående udførelse ved et foregående tidspunkt T_b
 befinder sig i en del af samleren, som går fra den første
 gang a_1 til gangen a_{i1} , der indeholder den første last i
 25 udgangssekvensen σ for den nye udførelse på det nye tidspunkt
 T_b .

9. Fremgangsmåde ifølge et hvilket som helst af kravene 1
 til 6, kendetegnet ved, at den udføres iterativt, idet hver
 30 ny udførelse foretages på et nyt tidspunkt T_b , der beregnes
 med følgende formel: $T_b = \text{Max}(T_b + 1, T(u_{last}) - \Delta_1 + \Delta_{ilast})$, hvor:
 - $u_{last} = a_{ilast}(j^{last})$ er en sidste nyttelast, der kommer fra
 den (j^{last}) 'te position i gangen a_{ilast} , i
 35 udgangssekvensen σ fra en foregående udførelse ved et
 foregående tidspunkt T_b , og
 - $(T_b + 1)$ opnås ved forøgelse af det foregående tidspunkt T_b
 med én enhed.

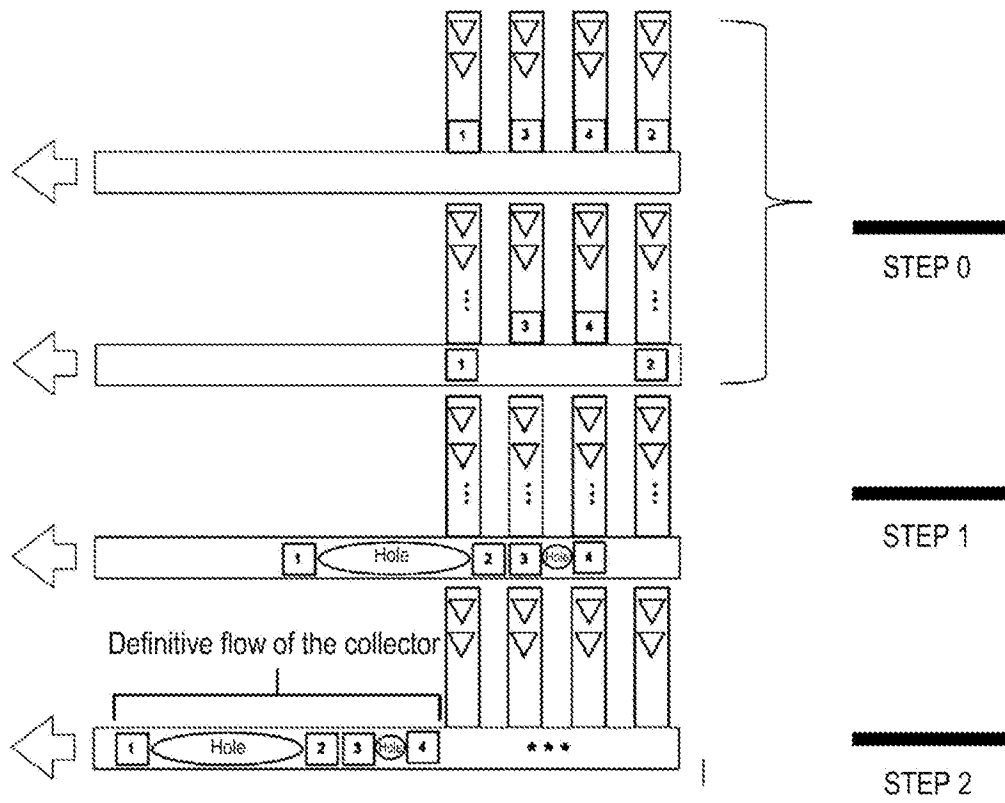


Fig. 1

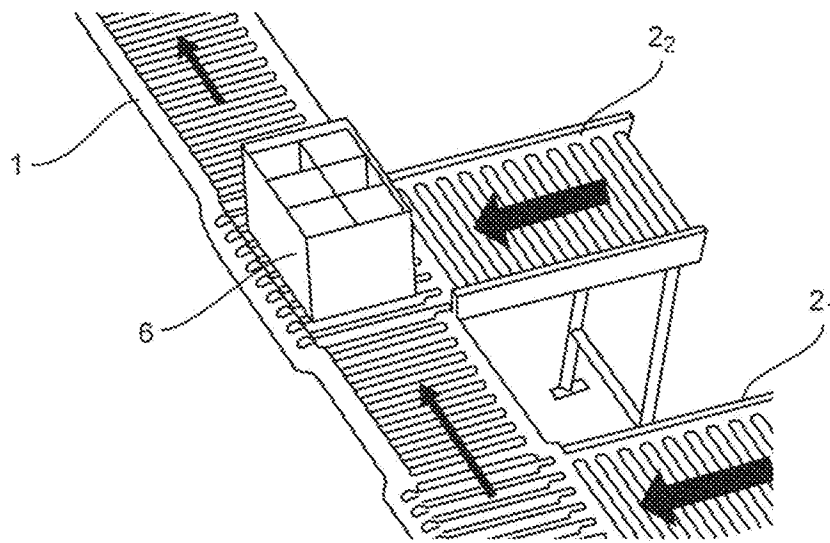


Fig. 2

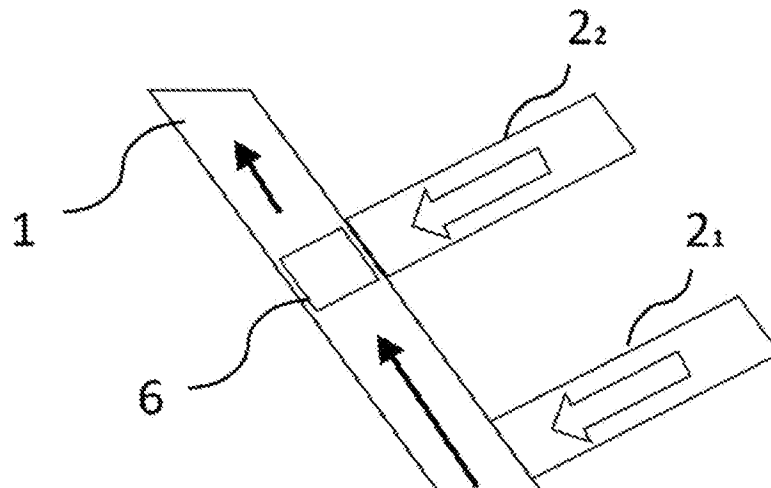


Fig. 3

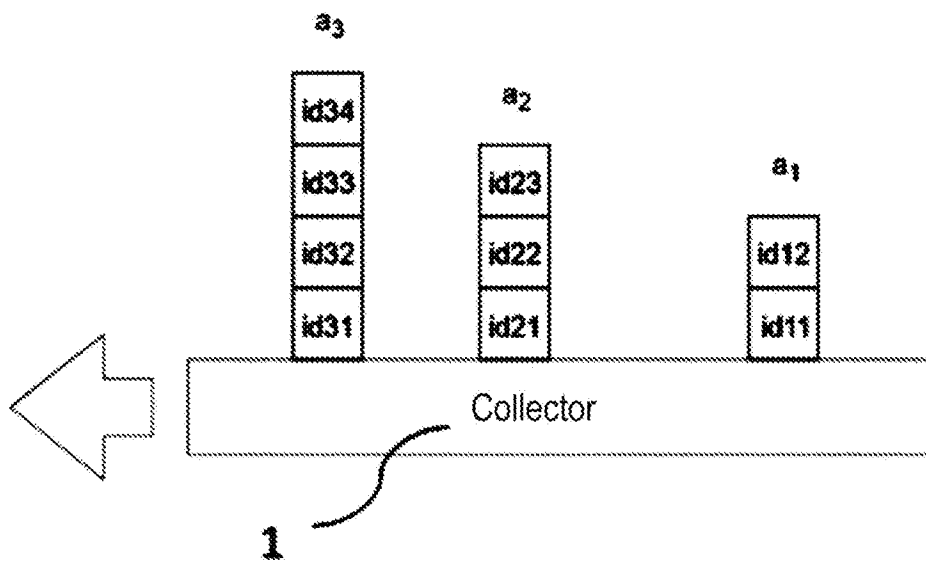


Fig. 4

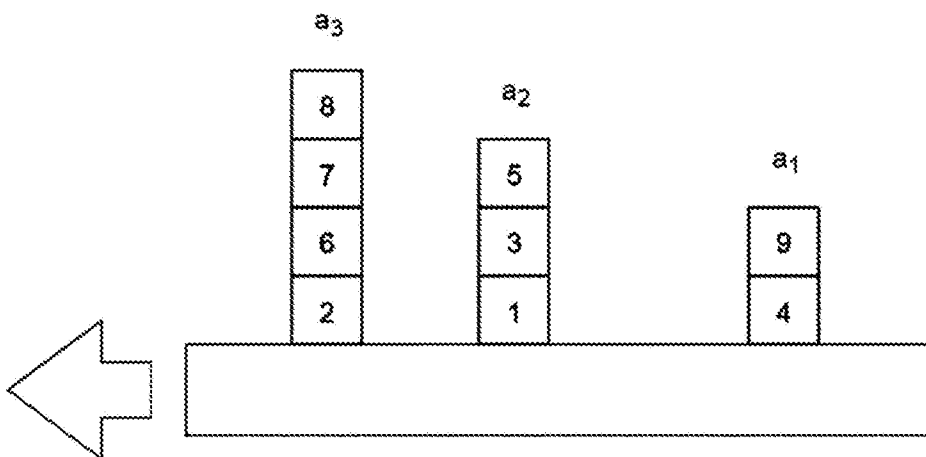


Fig. 5

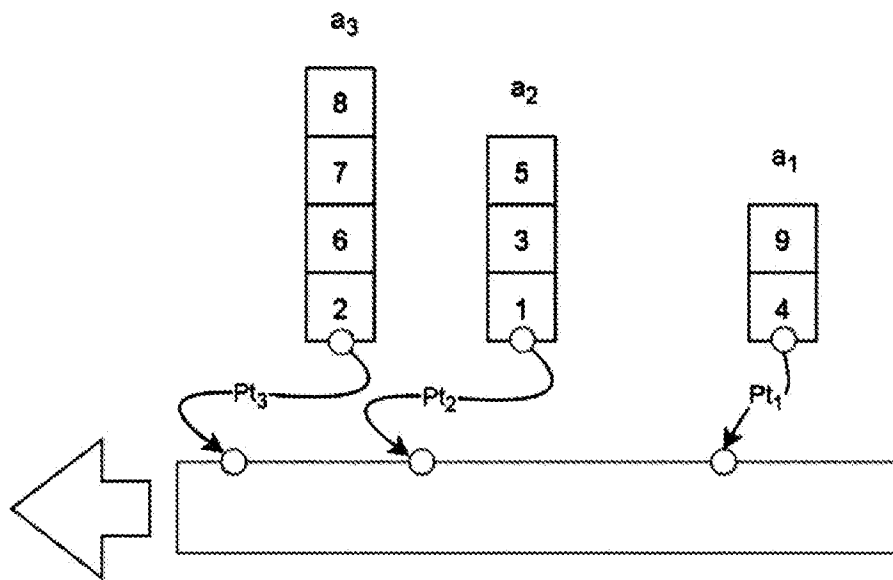


Fig. 6

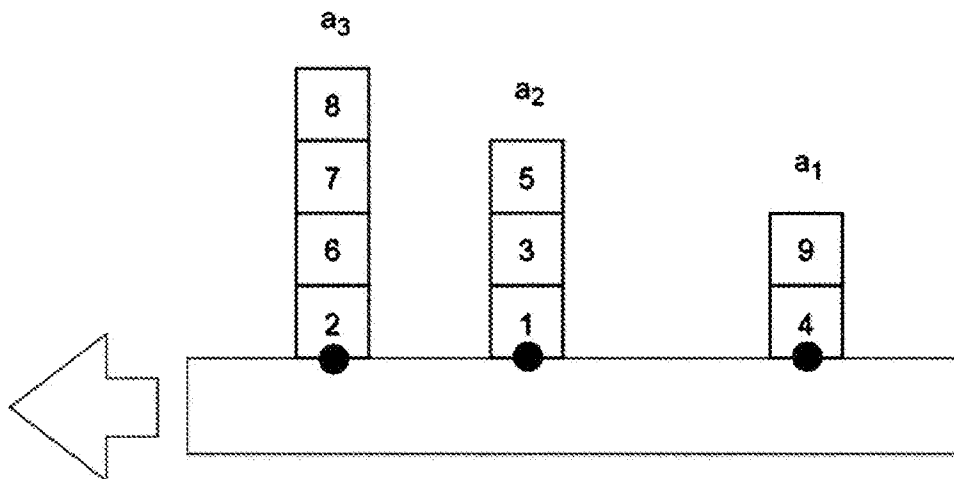


Fig. 7

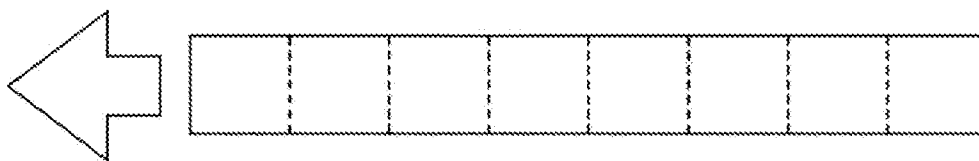


Fig. 8

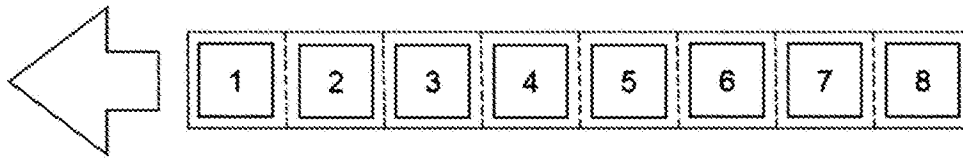


Fig. 9

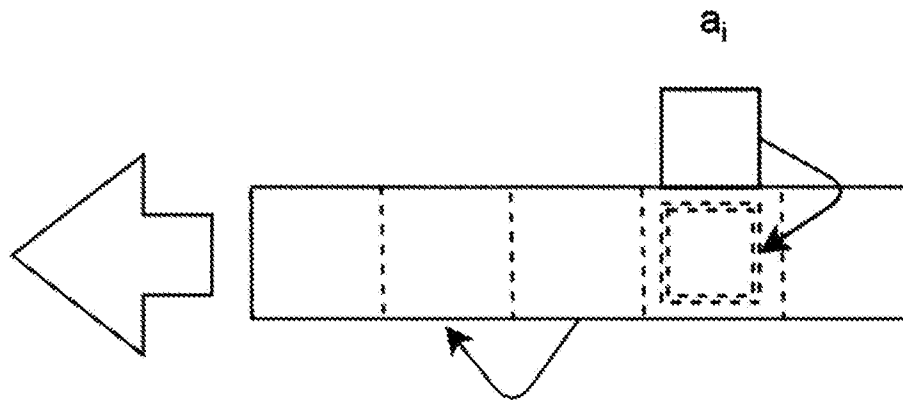


Fig. 10

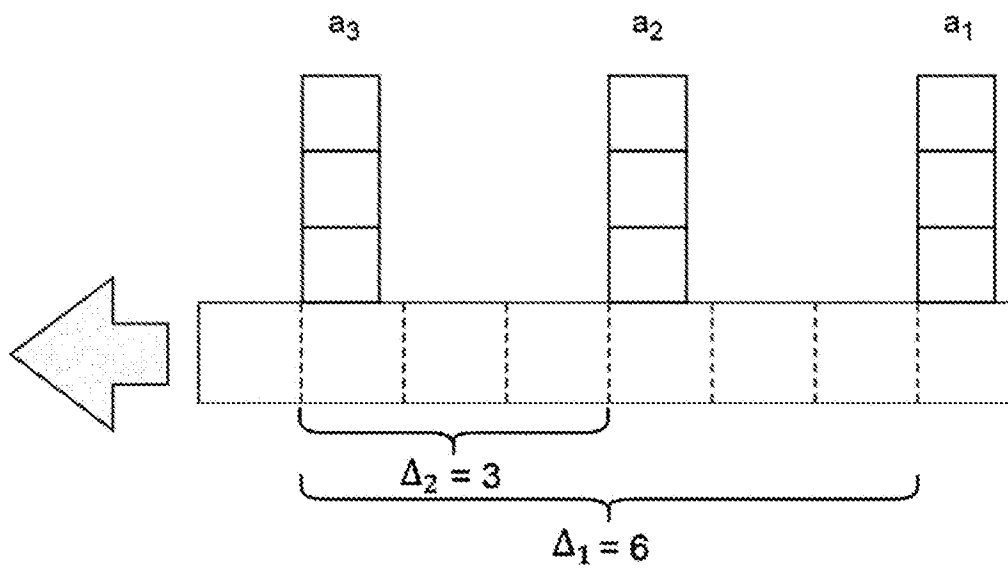


Fig. 11

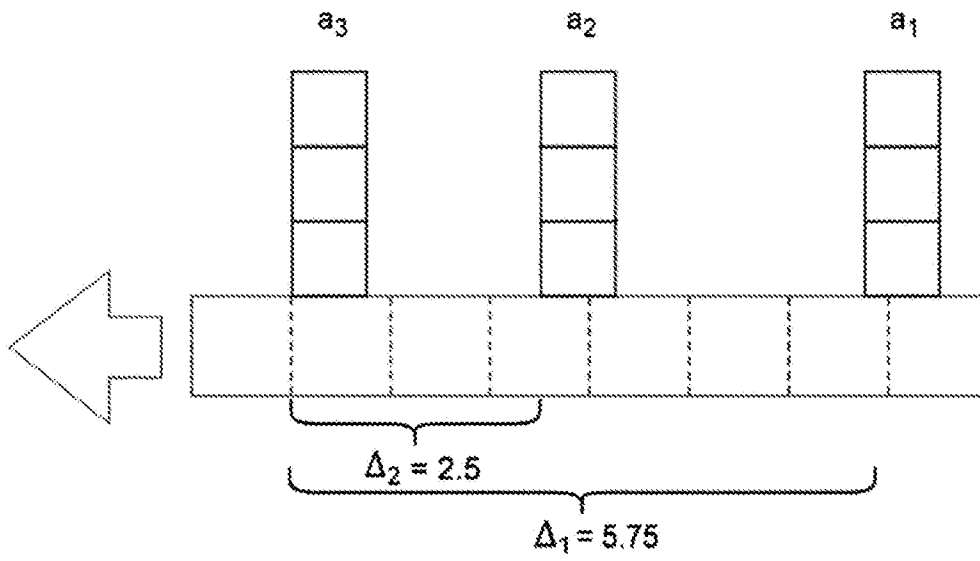


Fig. 12

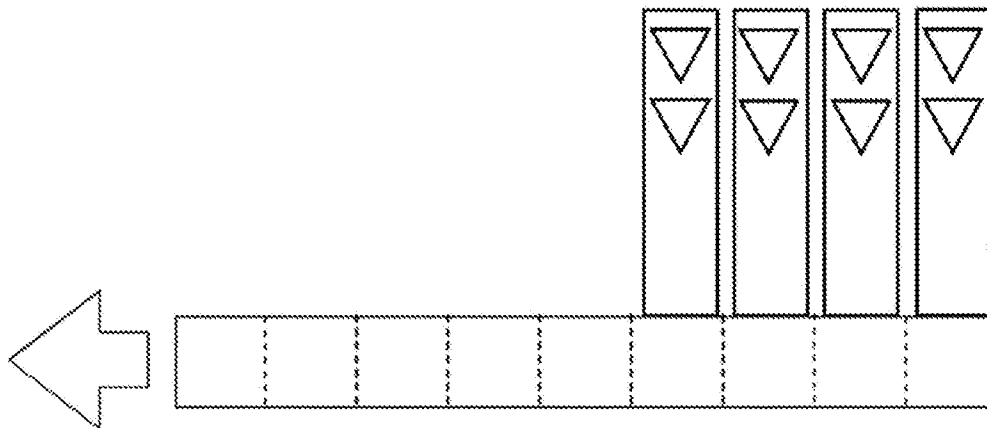


Fig. 13

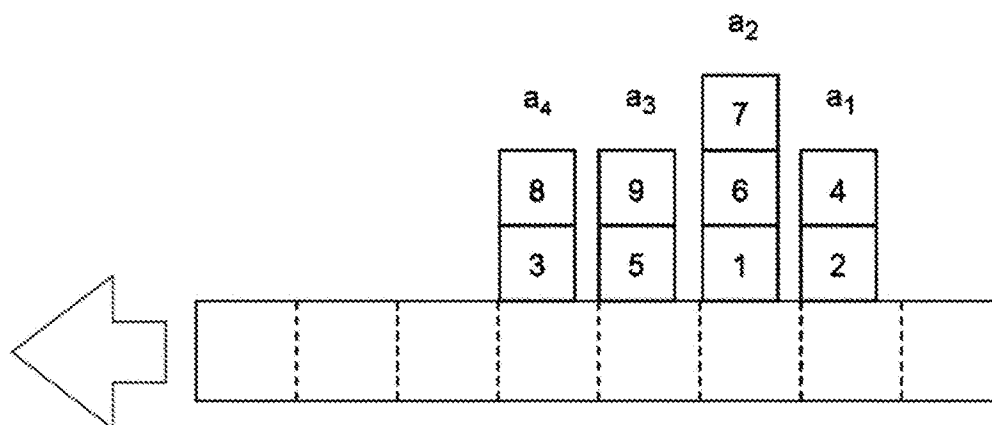


Fig. 14

Time	$\emptyset$	t_0					5						
M_1	2		4										
M_2	1	2		4			6	7					
M_3		1	2		4	5	6	7			9		
M_4				1	2	3	4	5	6	7	8	9	

Desired final sequence σ

Fig. 15

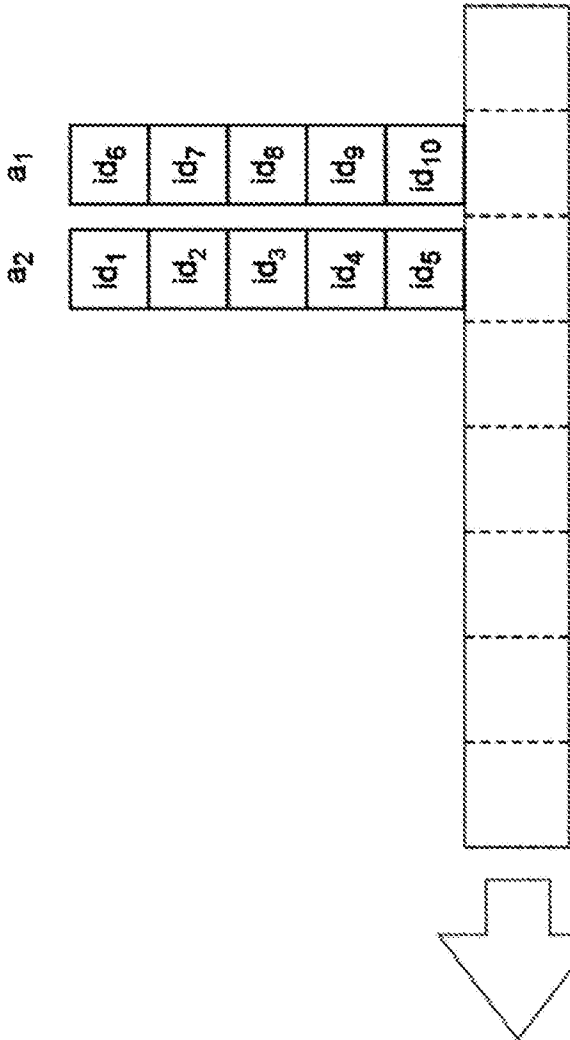


Fig. 18

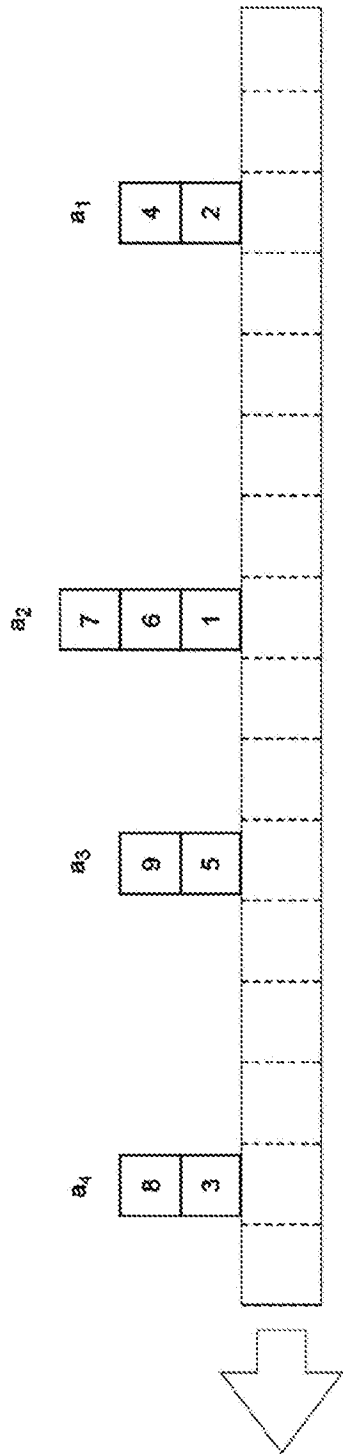


Fig. 16

Time		0		5		t_0	
M1		2	4				
			2	4			
M2			2	4			
				1	2	4	
M3				1	2	4	
					1	2	
M4							

Desired final sequence σ

Fig. 17

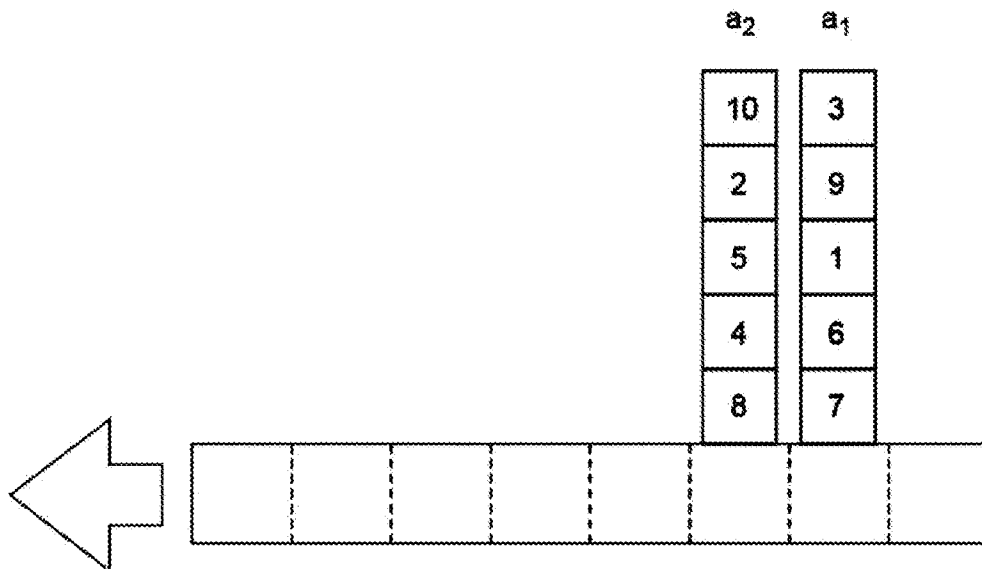


Fig. 19

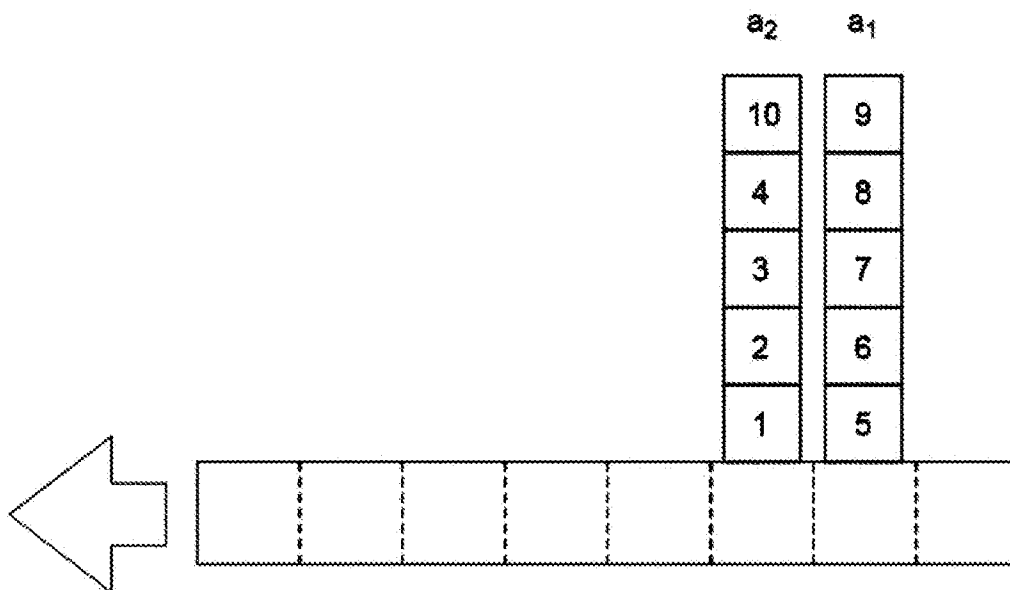


Fig. 20

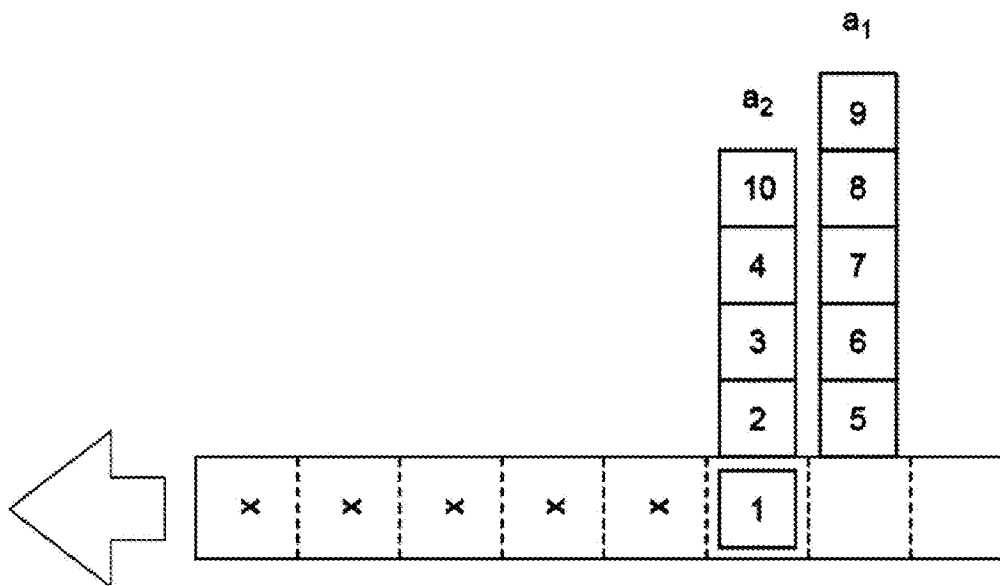


Fig. 21

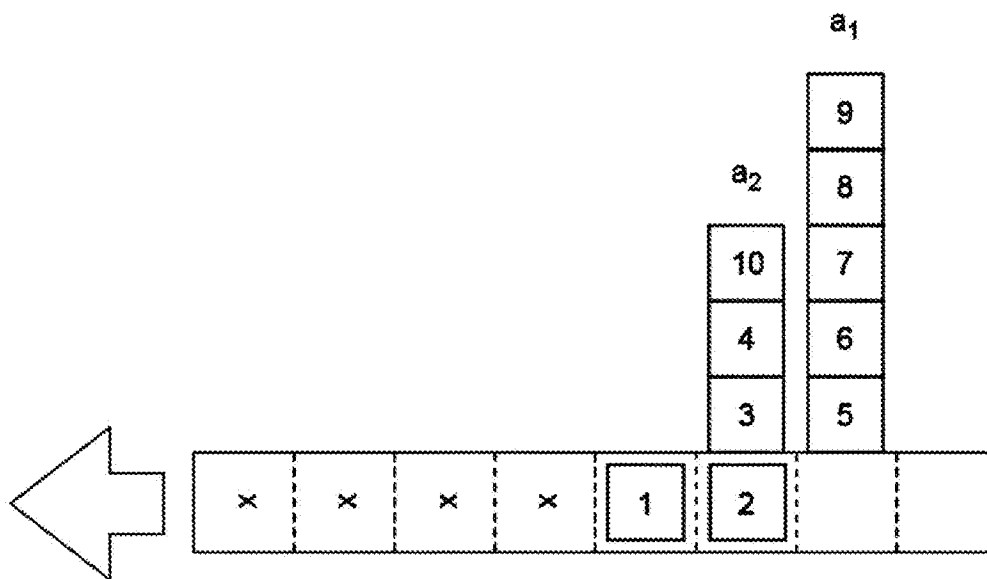
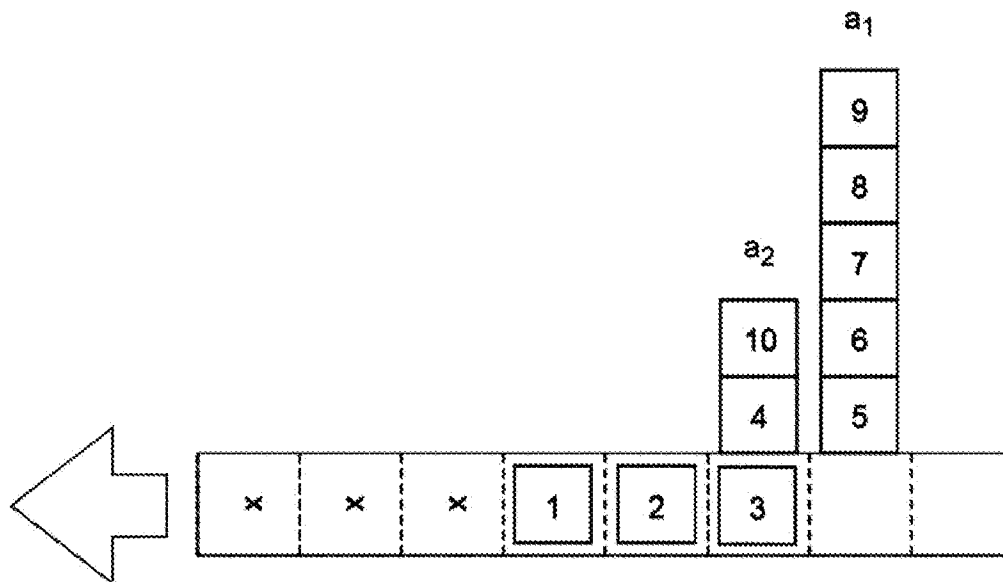
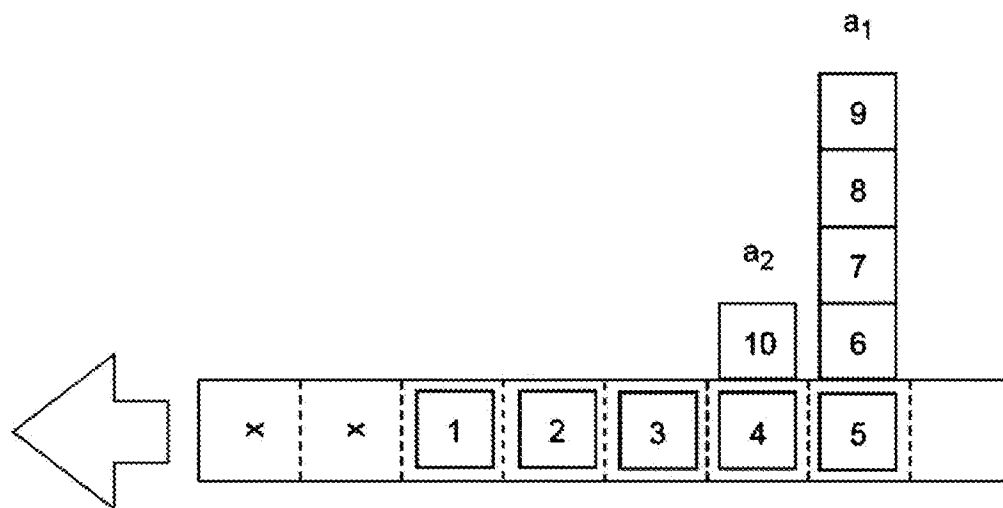
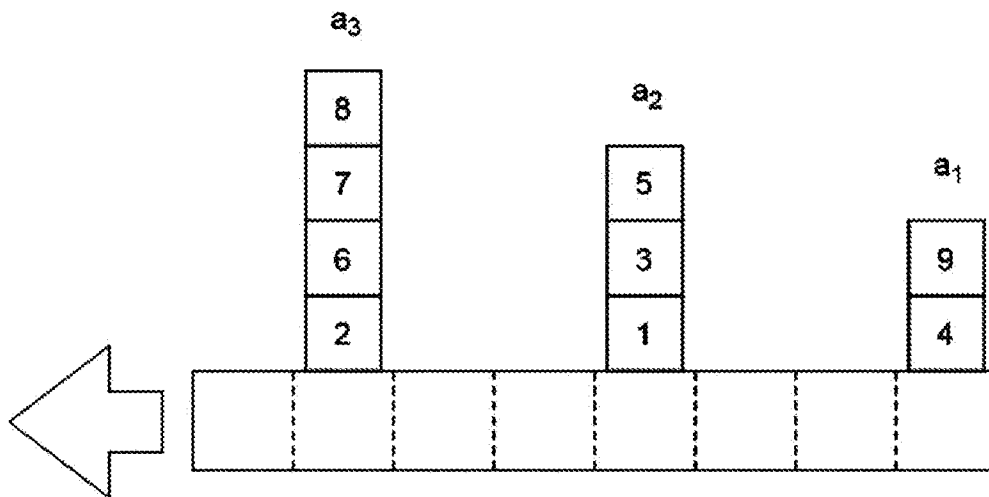
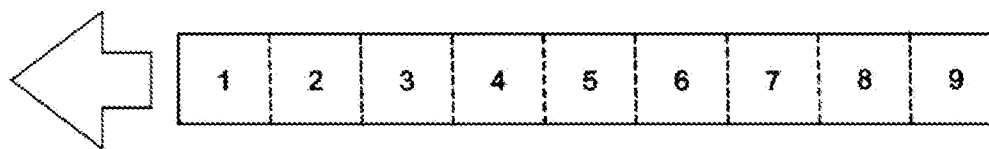
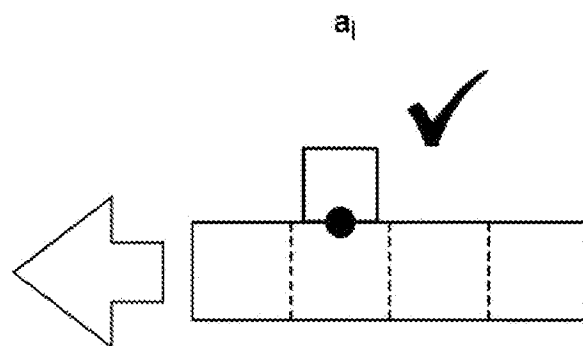
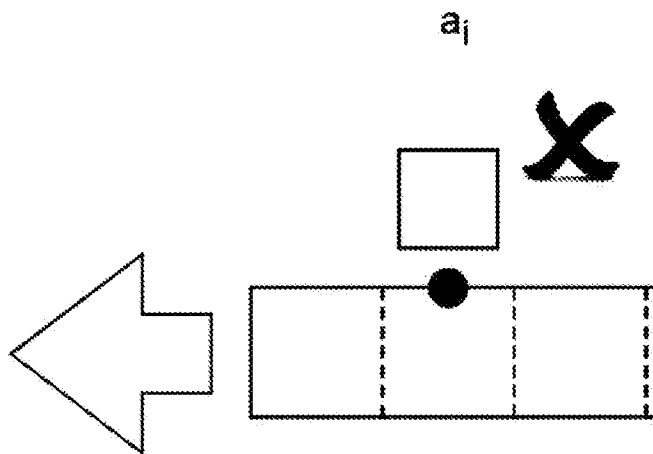
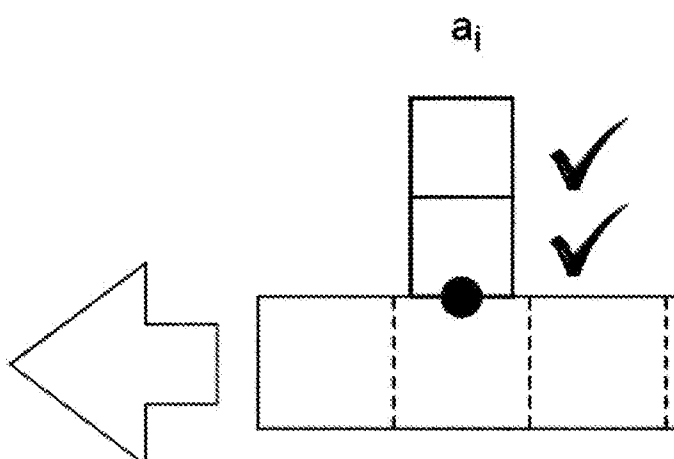
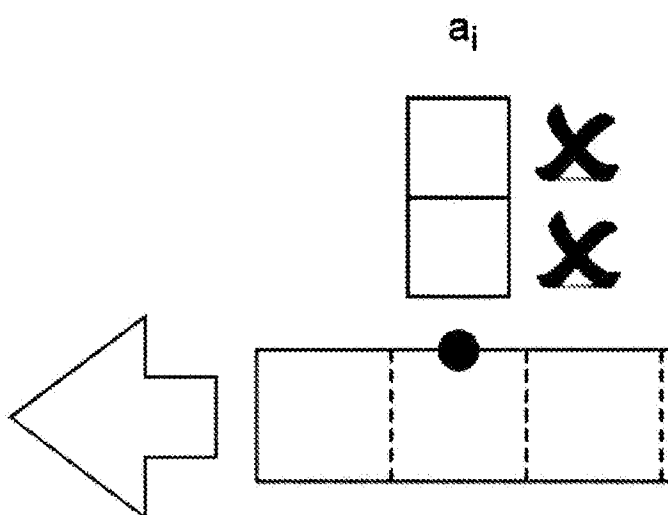
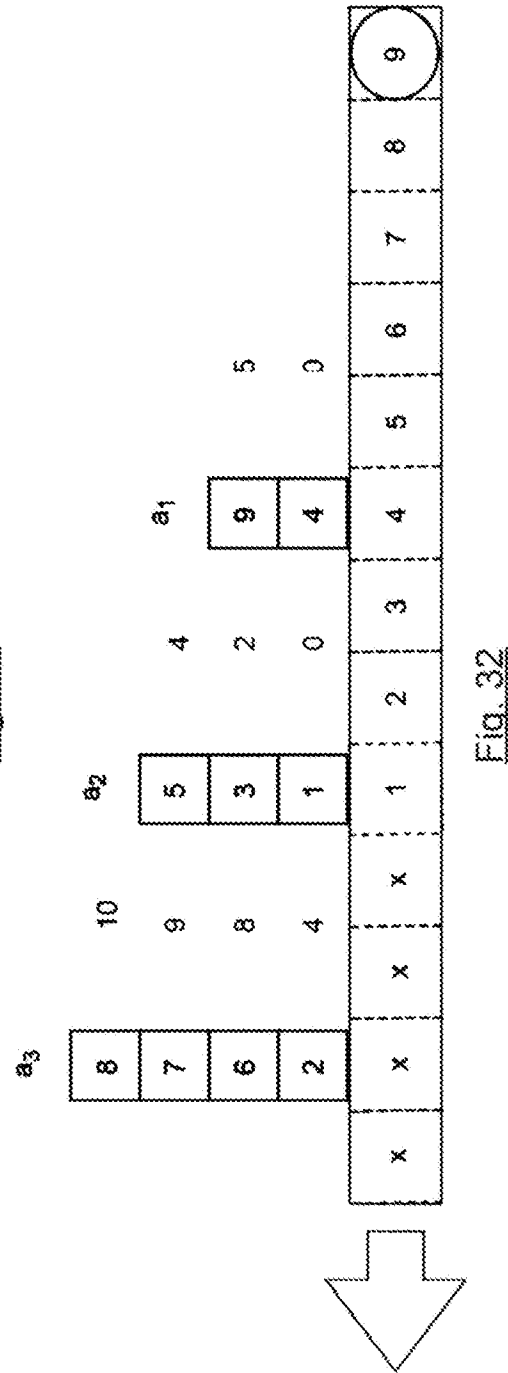
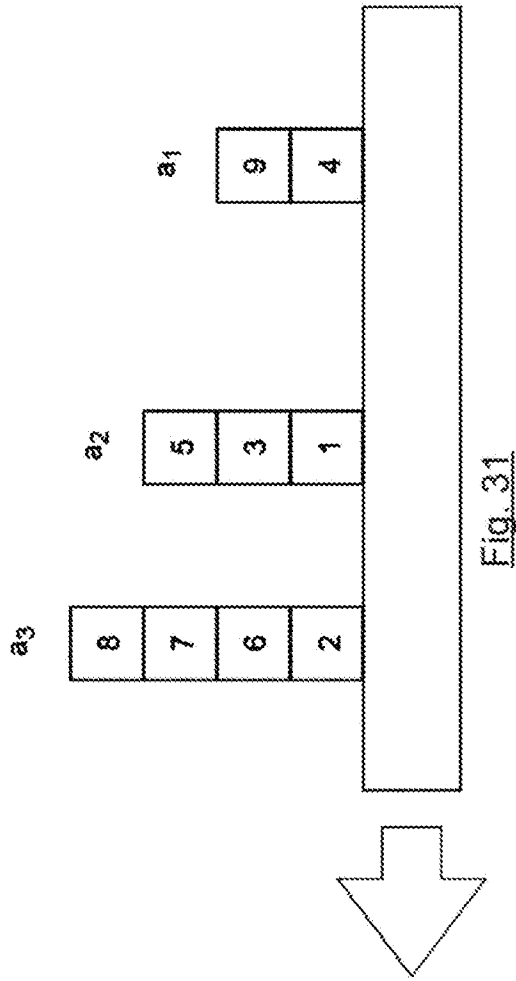


Fig. 22

Fig. 23Fig. 24

Fig. 25Fig. 26Fig. 27

Fig. 28Fig. 29Fig. 30



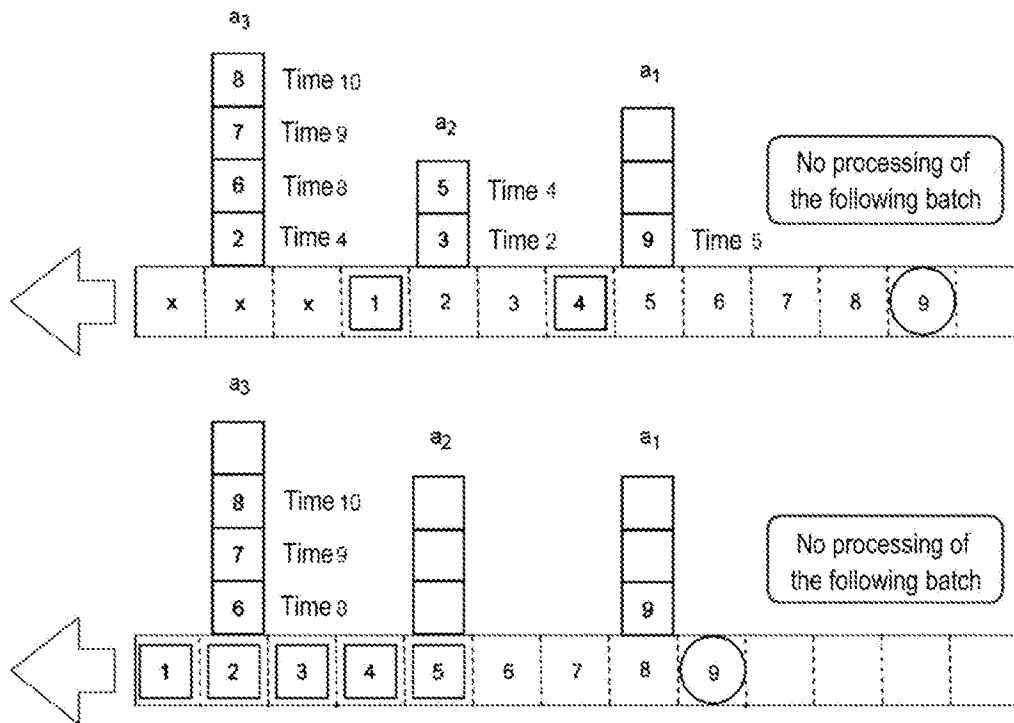


Fig. 33

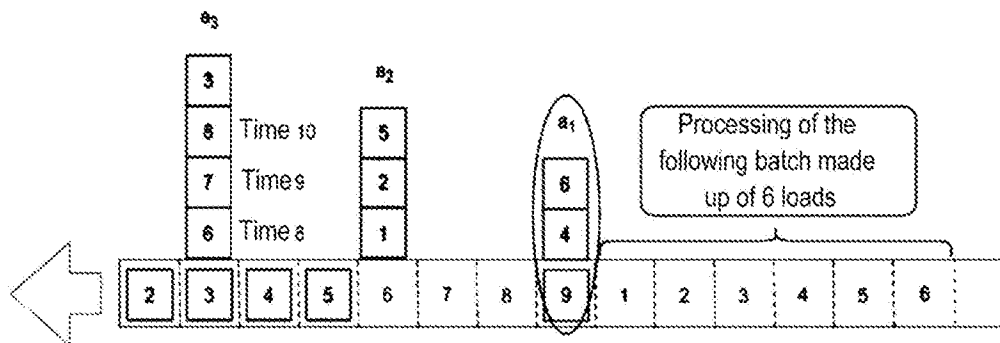


Fig. 34

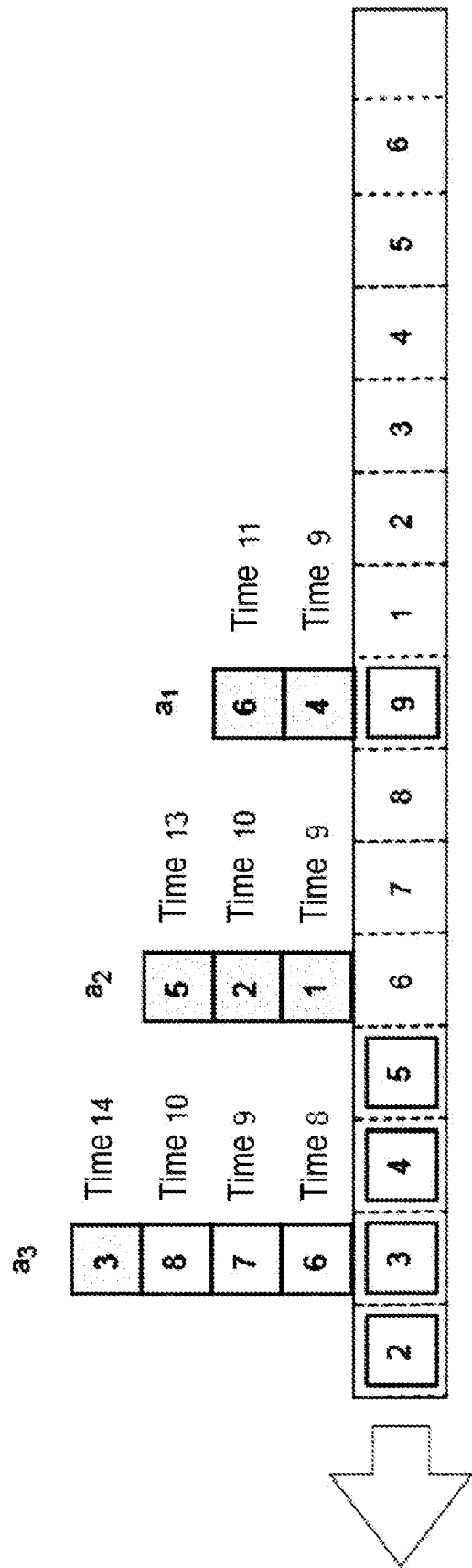


Fig. 35

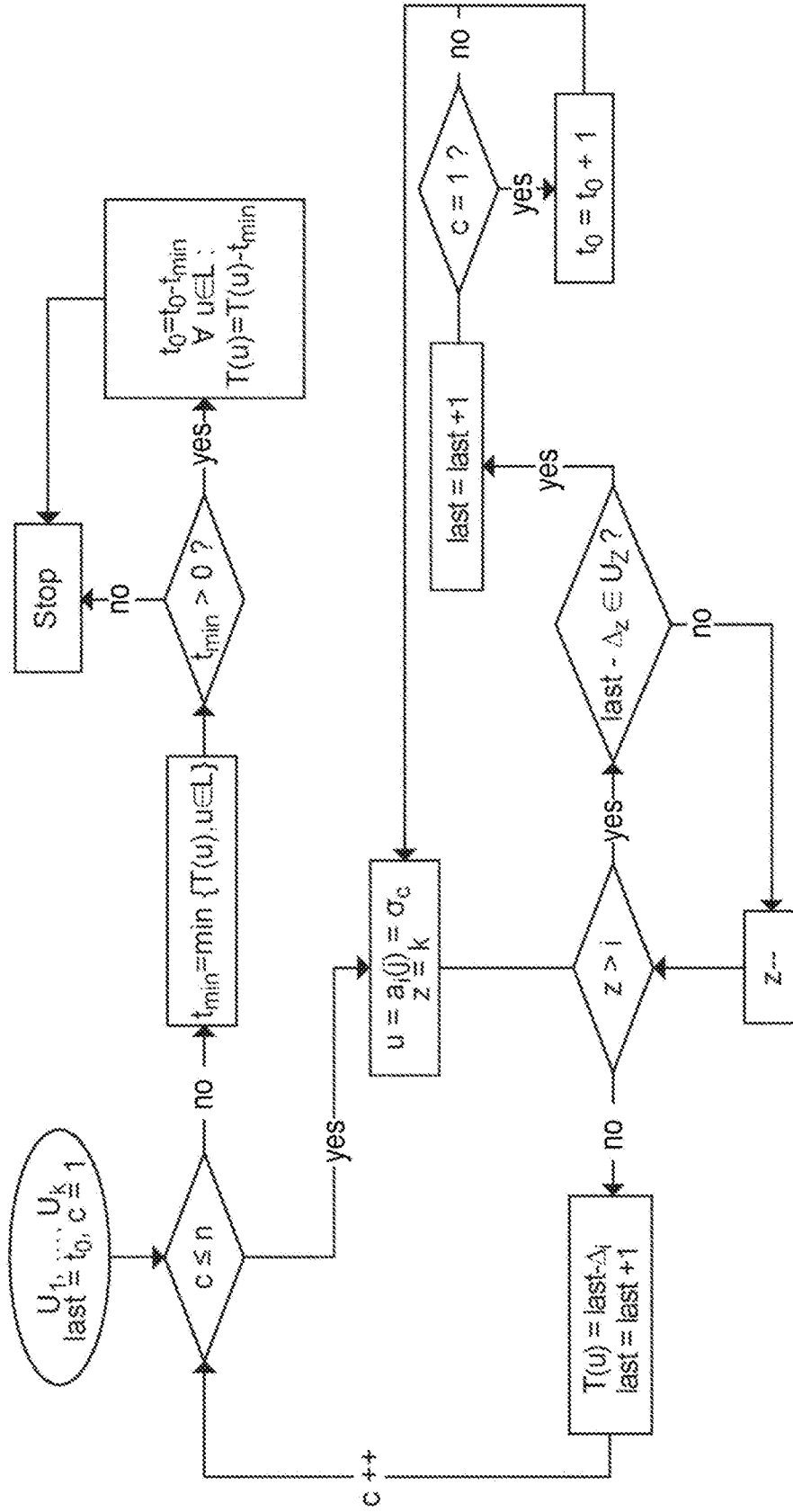


Fig. 36

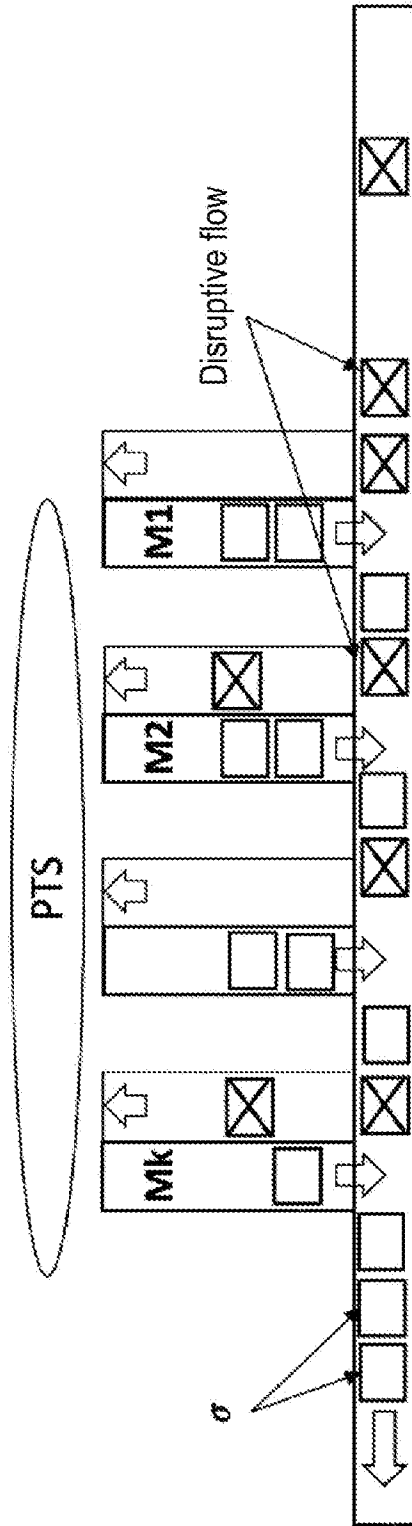


Fig. 37

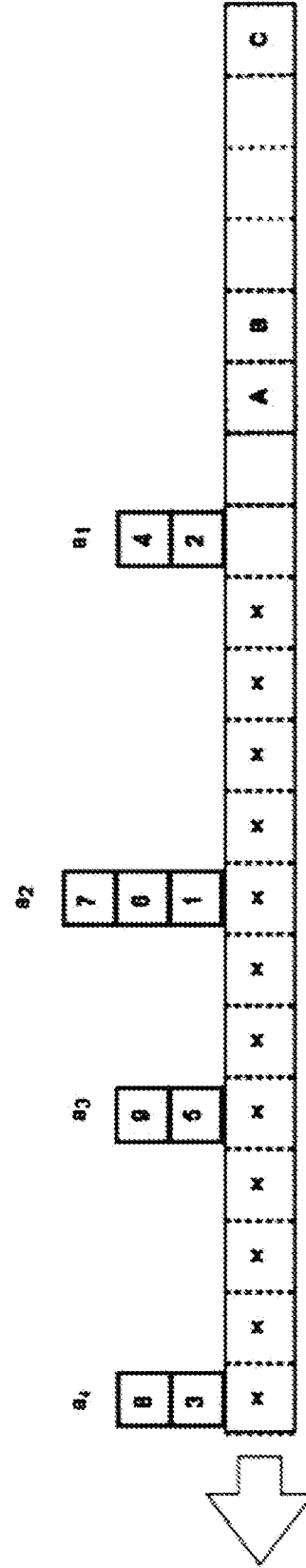


Fig. 38

Time	$\emptyset$		5		t_0																		
M1	2	A	B	4		C																	
		2	A	B	4		C																
			2	A	B	4		C															
				2	A	B	4		C														
				2	A	B	4		C														
M2				1	2	A	B	4		6	C	7											
				1	2				4		6	C	7										
					1	2				4		6	C	7									
						1	2			4		6	C	7									
M3							1	2			4	5	6	C	7		9						
								1	2			4	5	6		7		9					
									1	2			4	5	6		7		9				
										1	2			4	5	6		7		9			
											1	2			4	5	6		7	9			
												1	2			4	5	6		7	9		
M4													1	2	3	x	4	5	6	x	7	8	9

Desired final sequence σ with two holes

Desired final sequence σ with two holes

Fig. 39

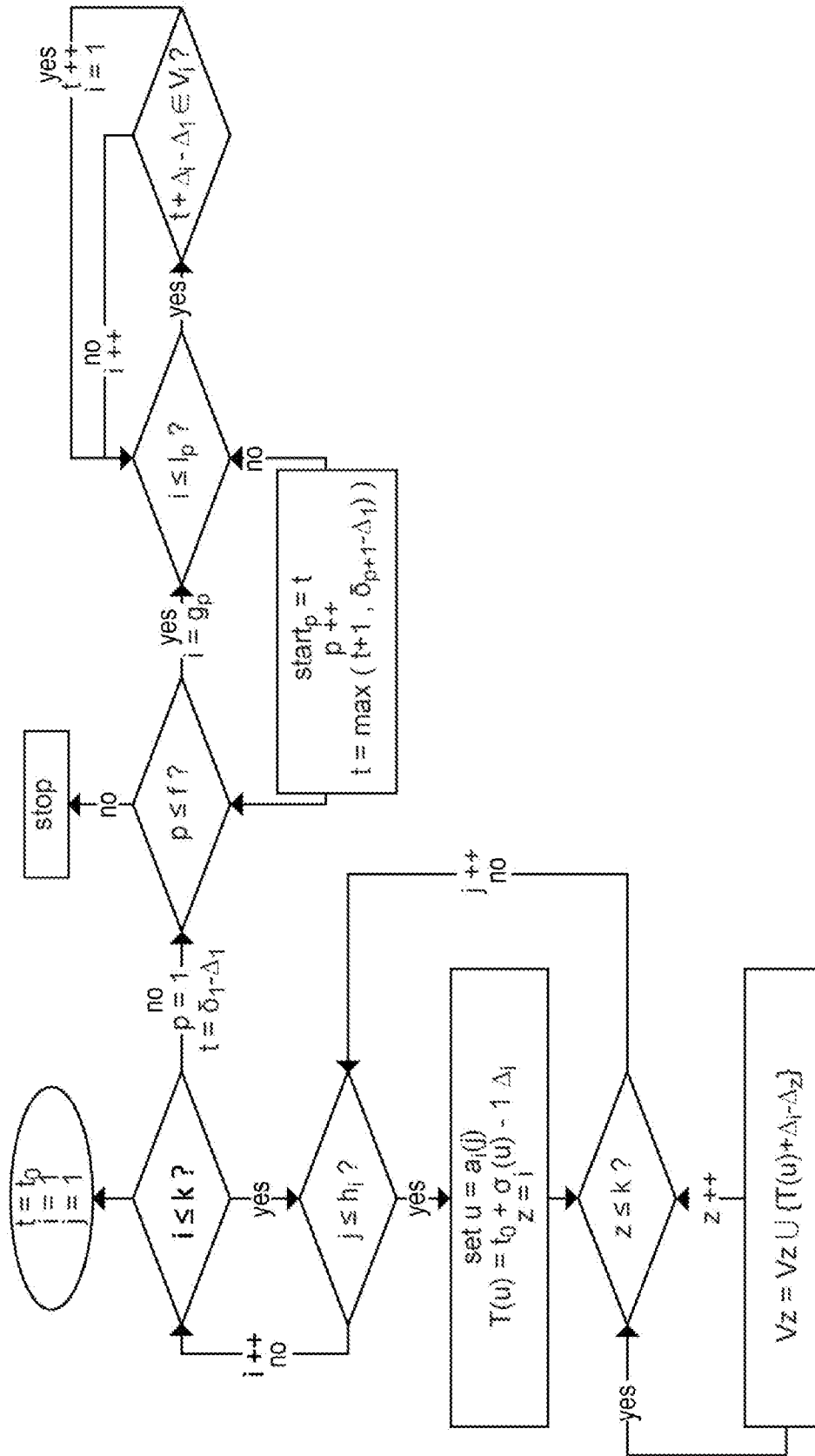
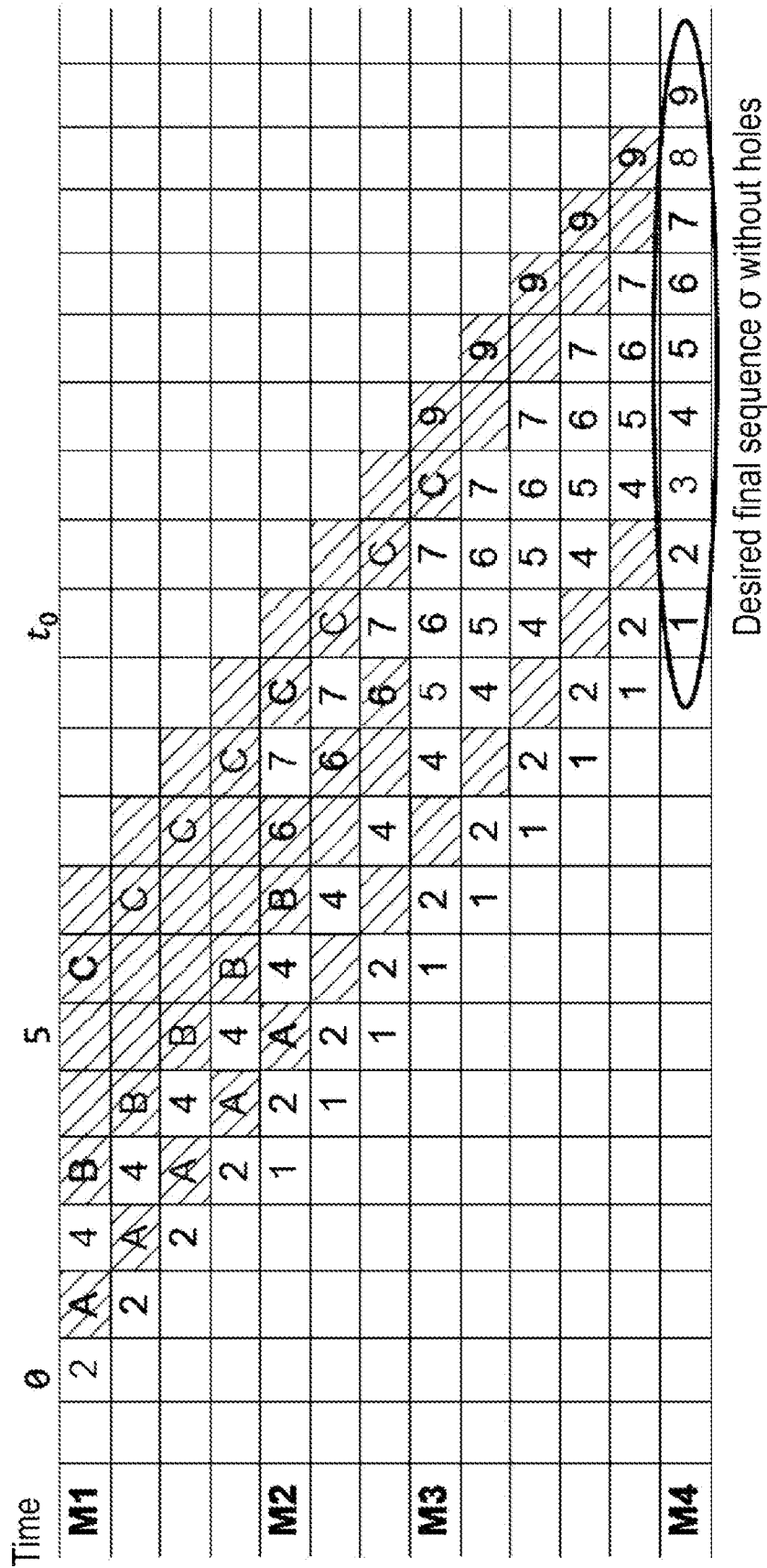
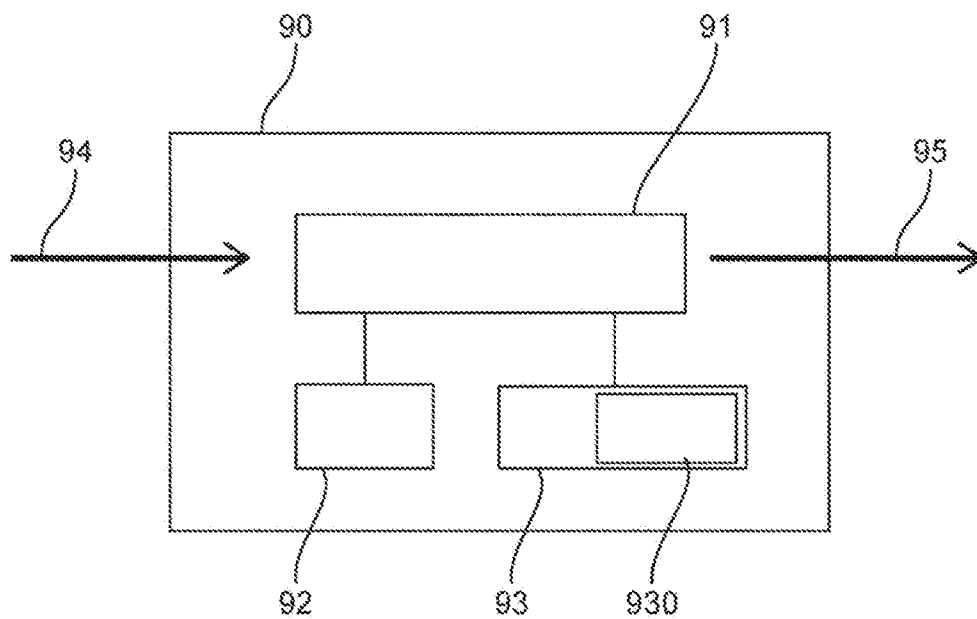
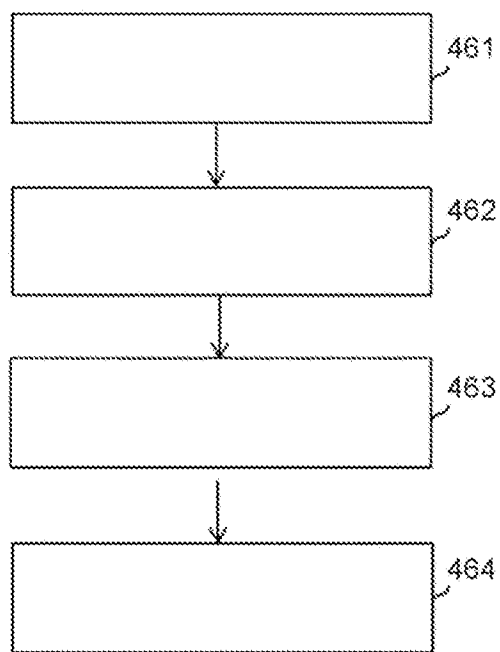


Fig. 40



Fig. 44Fig. 46

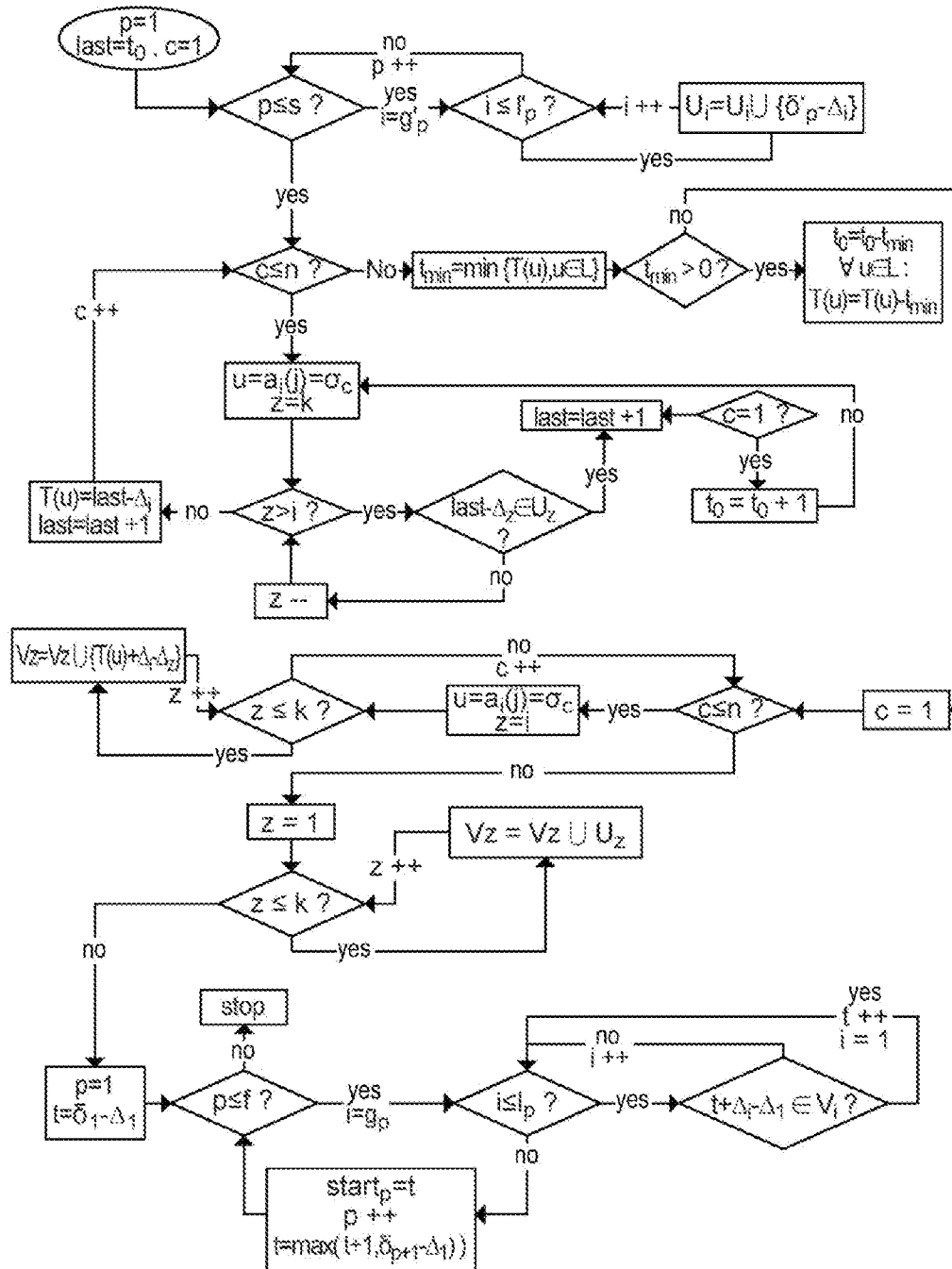
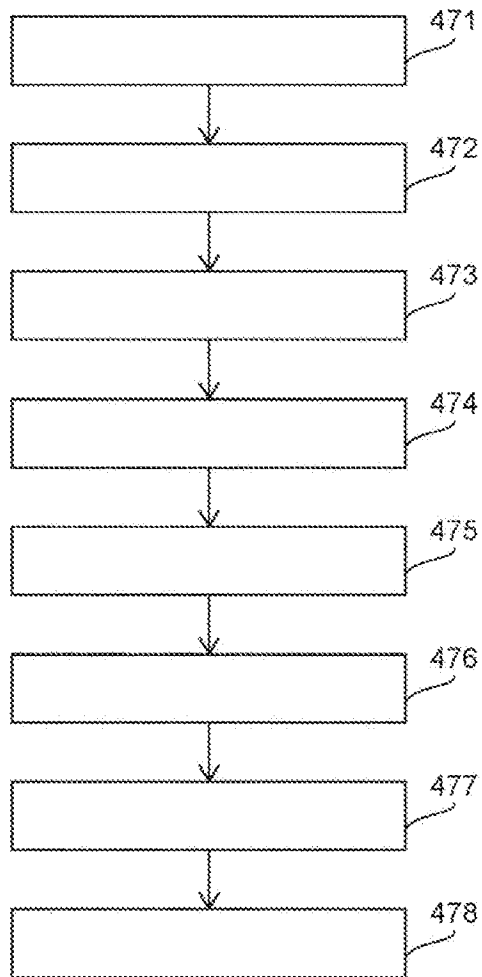
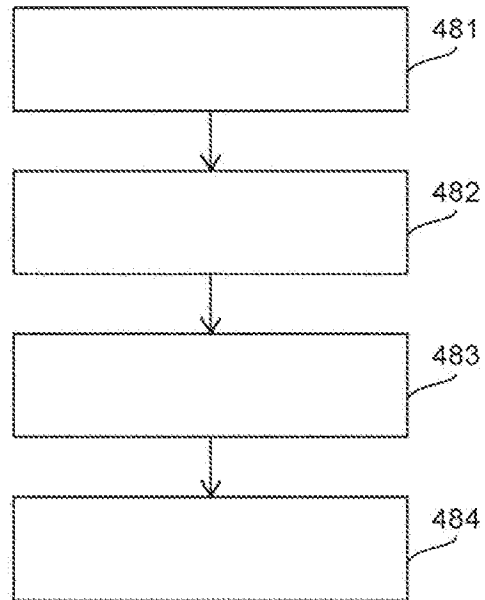
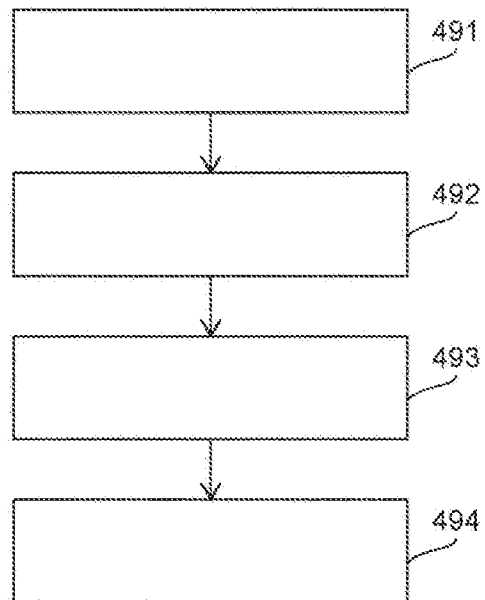


Fig. 45

Fig. 47Fig. 48Fig. 49

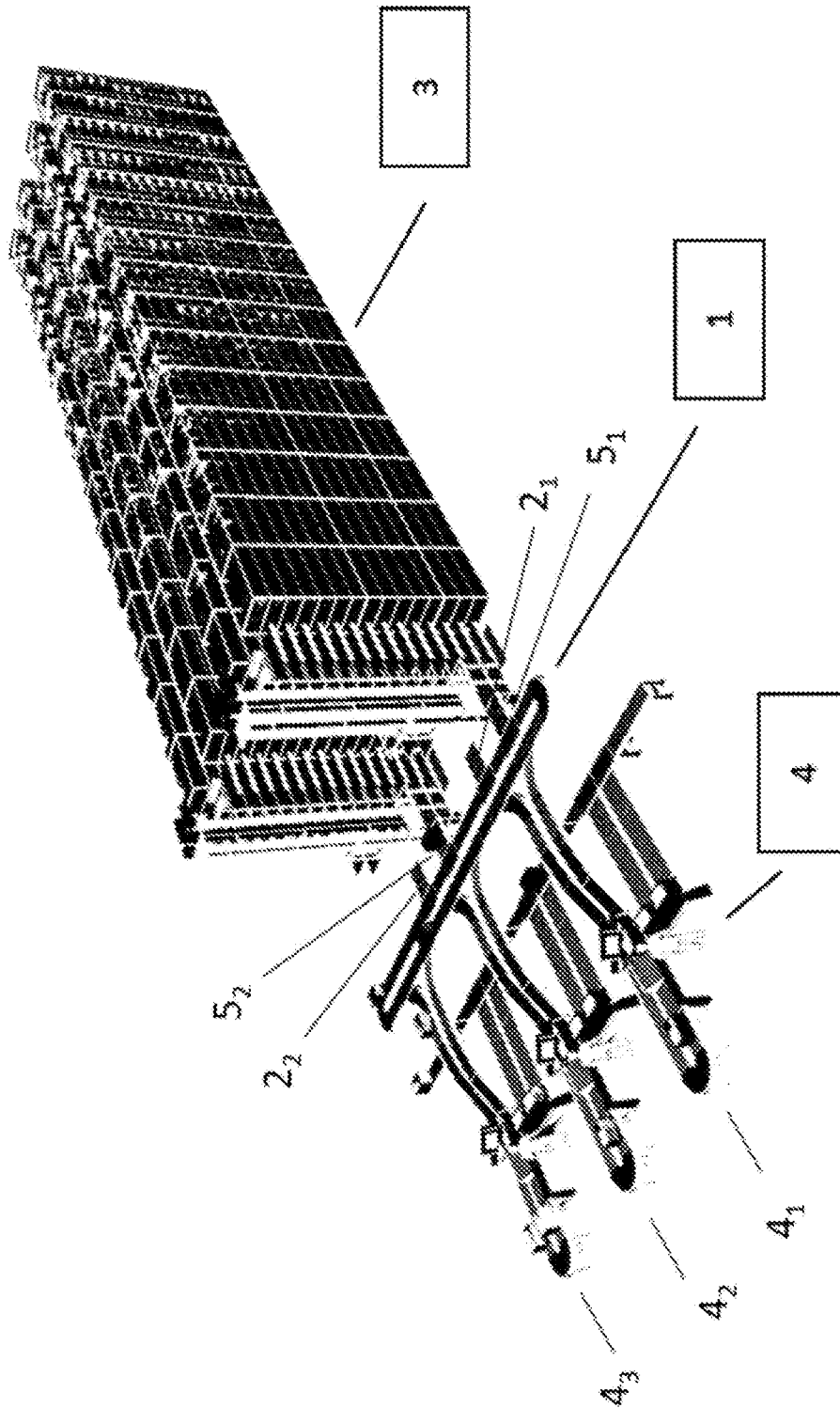


Fig. 50