USING THE STOCHASTIC APPROACH FRAMEWORK TO MODEL LARGE SCALE MANUFACTURING PROCESSES

Benayadi Nabil, Le Goc Marc
LSIS, Laboratory for Information and System Sciences,University of Marseilles, Marseilles, France

Bouché Philippe
LGECO, INSA de Strasbourg, Strasbourg, France

Keywords: Information-Theory, Temporal Knowledge Discovering, Chronicles Models, Markov Processes.

Abstract: Modelling manufacturing process of complex products like electronic ships is crucial to maximize the quality of the production. The Process Mining methods developed since a decade aims at modelling such manufacturing process from the timed messages contained in the database of the supervision system of this process. Such process can complex making difficult to apply the usual Process Mining algorithms. This paper proposes to apply the Stochastic Approach framework to model large scale manufacturing processes. A series of timed messages is considered as a sequence of class occurrences and is represented with a Markov chain from which models are deduced with an abductive reasoning. Because sequences can be very long, a notion of process phase based on a concept of class of equivalence is defined to cut up the sequences so that a model of a phase can be locally produced. The model of the whole manufacturing process is then obtained with the concatenation of the model of the different phases. The paper presents the application of this method to model the electronics chips manufacturing process of the STMicroelectronics Company (France).

1 INTRODUCTION

[1]Modeling manufacturing process of complex objects like electronic ships is crucial to minimize the scrapped objects and so to maximize the production of objects. The methods and the algorithms developed in the ”Process Mining” domain aims at modeling such manufacturing process with a graph of manufacturing steps (treatments, operations or tasks) from the timed messages contained in the database. (Cook and Wolf, 1998). The proposed algorithms generally generate complex model so that they are difficult to use when the manufacturing process contains hundreds of steps. This paper proposes a modeling method and the corresponding algorithms to model manufacturing processes having hundreds of steps. The proposed method is based on the cutting up of the sequences in sub-sequences called *process phases* where there is no cycle. The corresponding hypothesis is that when a subsequence contains two occurrences of the same step *S* made on the same machine but at two different times, the product is not in the same state: there is then *a priori* no relations between the series of steps following the first step *S* and the series of steps following the second steps *S*. So the two series must be represented with two different models.

The section 2 recalls briefly the main approaches of the Process Mining area and the main characteristics of the Stochastic Approach framework. Section 3 defines the notions of *class of equivalence* and *process phase* that we propose and describes the algorithms that are required to this aim. Section 4 presents the application of the algorithms to model the manufacturing process of wafers (i.e. silicon plates where electronic ships are engraved) of the Rousset (France) plant of the STMicroelectronics Company. The paper concludes in section 5 with a summary of the proposed method and introduces of our current works.

2 RELATED WORKS

In the Process Mining framework, a series of messages is considered as an ordered set of events from

[1]This study has been made possible thanks to the financial support of STMicroelectronics Company (France).

Nabil B., Goc Marc L. and Philippe B. (2008).
USING THE STOCHASTIC APPROACH FRAMEWORK TO MODEL LARGE SCALE MANUFACTURING PROCESSES.
In *Proceedings of the Third International Conference on Software and Data Technologies - ISDM/ABF*, pages 186-191
DOI: 10.5220/0001888801860191

which a process model is to be inferred and represented with a formalism (workflows, state charts or Petri nets for examples) (van der Aalst and Weijters, 2004). One of the first algorithm was proposed in (Agrawal et al., 1998). The algorithm aims at finding workflow graphs from a set of series of events contained in a workflow log. An event represents the start time of a task. To avoid the problem of potential cycles (i.e. repeated events in a series), the algorithm first renames the repeated labels of task before enumerating the binary dependency relations between the tasks. This set of relations is then reduced with the use of the transitivity property of the binary relations. Labels are again renamed to merge the tasks, making possible the introduction of cycles in the model. Difficulties arise with this approach when (i) the tasks are statistically independent and (ii) the number of tasks is large (Agrawal et al., 1998). Nevertheless, Pinter (Pinter and Golani, 2004) extends this algorithm notably with the introduction of events marking the end of the tasks. Similar issues in the context of software engineering processes are investigated in (Cook and Wolf, 1998) where the aim is to build a finite state machine from the set of the most frequent event patterns mined in a given log. In particular, the *Markov* algorithm is based on a two order Markov chain that is converted in states and state transitions. Cook and Wolf (Cook and Wolf, 2004) extend this method to concurrent processes and uses a first order Markov chain to this aim. The difficulties come from the pruning of the finite state machine to obtain a minimal model and the sensibility of pruning metrics to the ”noise” (van der Aalst and Weijters, 2004). Aalst (van der Aalst et al., 2004) defines the class of process that can be modeled with the α-algorithm but this algorithm requires the series of events in the log to be noise-free and complete.
There is a consensus to consider that finite state machines are difficult to understand and to validate. And most of the proposed methods have difficulties when (i) the process contains a lot of steps, (ii) the series in the log induce potential cycles in the models and (iii) the sequences are not noise-free and complete. The *Stochastic Approach framework* (Le Goc et al., 2005) for discovering temporal knowledge from timed observations provides a general framework for modeling dynamic processes that is based on a markovian representation but uses abstract chronicle models (Ghallab, 1996) instead of finite state machines. This framework considers that the timed messages of a series are written in a database by a program, called a monitoring cognitive agent *MCA*, that monitors a production process *Pr*. A timed message is represented with an occurrence of a discrete event class $C^i = \{e_i\}$ that is an arbitrary set of discrete event $e_i = (x_i, \delta_i)$, where δ_i is one of the discrete value of the variable x_i. When the variable x_i is not known, an abstract variable ϕ_i is used to define the discrete event $e_i = (\phi_i, \delta_i)$ corresponding to the constant δ_i. A discrete event class is often a singleton because in that case, two discrete event classes $C^i = \{(x_i, \delta_i)\}$ and $C^j = \{(x_j, \delta_j)\}$ are only linked with the variables x_i and x_j when the constants δ_i and δ_j are independent (Le Goc, 2006). This condition is only concerned with the programs the *MCA* is made with. A sequence of discrete event class occurrences is then considered as the observable manifestation of a series of state transitions in a timed stochastic automaton representing the couple (Pr, MCA). The *BJT4G* algorithm represents a set of sequences of discrete event class occurrences with a one order Markov chain and uses an abductive reasoning to identify the set of the most probable timed sequential binary relations between discrete event classes leading to a given class. A timed sequential binary relation $R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}])$ is an oriented relation between two discrete event classes C^i and C^j that is timed constrained with the interval $[\tau^-_{ij}, \tau^+_{ij}]$. $[\tau^-_{i,j}, \tau^+_{i,j}]$ is the time interval for observing an occurrence of the C^j class after an occurrence of the C^i class. The set of timed sequential binary relation is an abstract chronicles model that is graphically represented with the *ELP* language (Event Language for Process) where the nodes are discrete event classes and the links are timed sequential binary relations. In this paper, we propose to tackle the two main problems of the Process Mining approaches with the extension of the Stochastic Approach framework.

3 EXTENSION OF THE STOCHASTIC APPROACH FOR PROCESS MINING

3.1 Motivation

Let us take an example to illustrate the proposed extensions with a manufacturing process having a set $S = \{A, B, C, D, E\}$ of 5 manufacturing steps. Suppose the supervision system records the execution of a step with a message $X(t_k)$ denoting the time t_k of the beginning of the execution of the step X. The three series of messages of table 1 is represented with the abstract chronicle model of figure 1. In this model, if the two nodes labeled with A denote the same manufacturing step, then the two nodes must be confused, introducing a cycle in the model. The same reasoning must be done with the other nodes, making the model difficult to read and to understand.

Table 1: Three series of event.

A(t_1)B(t_2)D(t_3)C(t_4)E(t_5)A(t_6)E(t_7)B(t_8)
A(t_9)B(t_{10})C(t_{11})D(t_{12})E(t_{13})A(t_{14})E(t_{15})B(t_{16})
A(t_{17})B(t_{18})C(t_{19})D(t_{20})E(t_{21})A(t_{22})E(t_{23})B(t_{24})

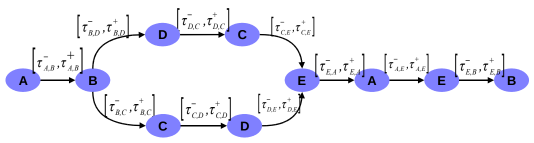
A
B
D
C
E
A
E
B
C
D
$[\tau^-_{A,B},\tau^+_{A,B}]$
$[\tau^-_{B,D},\tau^+_{B,D}]$
$[\tau^-_{D,C},\tau^+_{D,C}]$
$[\tau^-_{C,E},\tau^+_{C,E}]$
$[\tau^-_{B,C},\tau^+_{B,C}]$
$[\tau^-_{C,D},\tau^+_{C,D}]$
$[\tau^-_{D,E},\tau^+_{D,E}]$
$[\tau^-_{E,A},\tau^+_{E,A}]$
$[\tau^-_{A,E},\tau^+_{A,E}]$
$[\tau^-_{E,B},\tau^+_{E,B}]$

Figure 1: Model for the three Sequences.

Suppose now that each of the three sequences is cut up when a label appears two times. The model of the first part of the sequences will be similar to the one of figure 1 but without the path $A-E-B$ at the end of the model: the steps A, B and E introduce no more cycle. This fact motivates the notion of *process phase* proposed in this paper. But this notion is not sufficient to solve the cycles that are introduced with the steps C and D. We consider that this problem is due to the fact that the three series of events provide no information about the order of the steps C and D. Consequently, any solution of this problem must take into account some *a priori* knowledge about the process which we want to avoid it. The notion of *potential cycle* is then defined to detect this kind of situation to be able to make further investigations (i.e. finding new series or discussing with experts for examples).

To illustrate the notion of class of equivalence, let us take the *Edit* activity of the ”writing a scientific paper” process that can be made by different students and professors. The *Edit* activities can then be labeled differently according to the performer with a set of classes of the form: $C=\{C^{E1}=\{(s_1,\delta_1)\},C^{E2}=\{(s_2,\delta_2)\},\ldots\}$, where the variables s_i and p_i denotes respectively students and professors. In this case, the resulting model of the process will be complex without necessity. One of the interesting features of the Stochastic Approach framework is the notion of discrete event class. This notion can be used to define abstract classes of the form $C^{\phi_i}=\{(\phi_i,\delta_1),\ldots,(\phi_i,\delta_n)\}$ where ϕ_i denotes an abstract variable and the set $\{\delta_j\}, j=1\ldots n$, is an arbitrary set of constants. This property allows defining classes of equivalence that simplifies a process model. For example, an abstract class C^{ϕ_i} be defined as an equivalent class of the set of classes C of the ”writing a scientific paper” process. Doing this way allows constituting a set of sequences coming from different students and professors.

3.2 Class of Equivalence

By definition, a large scale process is made with a lot of steps. Some of these steps differs only with some characteristics but realizes similar treatments.

Definition 1. *Given a model* $M=\{R(C^i,C^j,[\tau^-_{ij},\tau^+_{ij}])\}$ *build with a set* $\Omega=\{\omega_i\}$ *of discrete event class occurrences* ω_i*, a class* $C^{\phi_i}=\{(\phi_i,\delta_1),(\phi_i,\delta_2),\ldots,(\phi_i,\delta_n)\}$ *is an equivalence class of a sub set of classes* $C=\{C^j\}$*,* $j=1\ldots n$*,* $C^j=\{(x_j,\delta_j)\}$*, of the set of classes* C_M *of a model* M *iff:*

$$\forall C^j\in C,\exists R(C^i,C^j,[\tau^-_{ij},\tau^+_{ij}])\in M$$
$$\wedge\exists R(C^j,C^k,[\tau^-_{jk},\tau^+_{jk}])\in M \qquad (1)$$

This definition means that every classes C^j of the sub set $C\subseteq C_M$ are linked with the same classes in M. When this condition is verified, each occurrences $C^j(k)$ of the classes C^j in the sequences ω_i of Ω can be rewritten as occurrences $C^{\phi_i}(k)$ of the equivalence class C^{ϕ_i}. The abstract variable ϕ_i has no *a priori* meaning: ϕ_i can be substituted with the corresponding concrete variable x_i in any occurrences $C^{\phi_i}(k)$. Consequently, the set of uphill relations $\{R(C^i,C^j,[\tau^-_{ij},\tau^+_{ij}])\}$ of M will become $\{R(C^i,C^{\phi_i},[\tau^-_{i\phi_i},\tau^+_{i\phi_i}])\}$ and the set of downhill relations $\{R(C^j,C^k,[\tau^-_{jk},\tau^+_{jk}])\}$ of M will become $\{R(C^{\phi_i},C^k,[\tau^-_{\phi_i k},\tau^+_{\phi_i k}])\}$. In practice, an equivalence class can be used to represent discrete event classes having similar meanings. In the application presented in the next section, a discrete event class represents a treatment made on a product with a particular machine. Equivalence classes are then used to represent the same treatment made on different machines: in that case, the machines are equivalents because the same treatment can be done on each of the machines.

Given a set of sequences $\Omega=\{\omega_i\}$, the algorithm for defining equivalence classes find all the equivalence classes and rewrite the corresponding occurrences in each sequence ω_i (Algorithm 1):

1. Build a model M given a set Ω of sequences ω_i
2. Find all the subset of classes $C=\{C^j\}$ verifying the equation 1
3. For all the sub sets C
    - Create an equivalence class C^{ϕ_i}.
    - For all $C^j\in C$, rewrite all the occurrences $C^j(k)$ in all the sequences $\omega_i\subset\Omega$ with the rewriting rule: $C^j(k)\equiv C^{\phi_i}(k)$.
4. Build a new model M' with Ω.

3.3 Process Phase

The information contained in a series of manufacturing messages is concerned both with the state of the manufactured object and the manufacturing process that make evolving this state from an initial state up to a final state. But generally, the state of the manufactured object is not provided with the messages. So we propose to capture indirectly this dimension with the notion of *process phase*.

Definition 2. *A process phase is a sub model* $M' = \{R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}])\}$ *so that there is no path* $P = \{R(C^i, C^{i+1}, [\tau^-_{ii+1}, \tau^+_{ii+1}])\} \in M'$, $i = 1 \dots n$, *where:*

$$\forall i < j \leq n+1, C^i = C^j \tag{2}$$

A process phase is then a sub model that does not contain two times the same discrete event class. The algorithm 2 aims at cutting up a set Ω of sequences ω_i in sub sequences ω^i_k that respects the equation 2 (i.e. ω^i_k does not contain two occurrences of the same class):

1. $\forall \omega_i \in \Omega$ do
   - Remove ω_i from Ω.
   - Cut up ω_i in a set $\Omega_i = \{\omega^i_k\}$ of sub sequences ω^i_k verifying the equation 2.
2. $\forall \omega^i_k \in \Omega_i$ do
   - Add an occurrence of the C^0 and C^1 classes at the beginning and the end of ω^i_k
3. $\Omega = \bigcup \Omega_i$.

An occurrence of an abstract start class C^0 and an occurrence of an abstract final class C^1 are added at the beginning and the end of each sub sequences ω^i_k so that the $BJT4G$ algorithm automatically identifies the process phases. For example, when applying the algorithm 2 on the three sequences of Table 1, the $BJT4G$ algorithm will find two process phases: the first phase starts from the event class C^A and finishes at the first event class C^E, the algorithm add the start class C^0 and finish class C^1 at this phase. The second process phase being: $\{\ R(C^0, C^A, [\tau^-_{0A}, \tau^+_{0A}])\ ,\ R(C^A, C^E, [\tau^-_{AE}, \tau^+_{AE}]), R(C^E, C^B, [\tau^-_{EB}, \tau^+_{EB}]), R(C^B, C^1, [\tau^-_{B1}, \tau^+_{B1}])\ \}$.

3.4 Potential Cycles

When looking the model of figure 1, it is clear that the classes C^C and C^D introduce a cycle. Cycles present a strong problem of interpretation, making hard to understand the resulting models. This explains why there is a lot of works aiming at avoiding cycles in process models (cf. (Cook and Wolf, 2004), (Schimm, 2004) (van der Aalst et al., 2004), (Pinter and Golani, 2004), (Weijters and van der Aalst, 2003) or (Agrawal et al., 1998) for examples). But these works make assumptions about the process or impose constraints about the constitution of the sequences. In all the case, this consists in having some *a priori* knowledge about the process to be modeled or the set of programs that write the messages in the process data base.

The aim of the Stochastic Approach is to provide models of sequences without any *a priori* knowledge about the process and the set of programs that have generated the occurrences. One difficulty is that cycles often appear when mining a process because of the transitivity property of the sequential binary relations.

Property 1. *The timed sequential binary relations* $R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}])$ *of a given abstract chronicle model* $M = \{R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}])\}$ *are transitives.*

$$\begin{aligned} \forall R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}]) \in M \wedge \forall R(C^j, C^k, [\tau^-_{jk}, \tau^+_{jk}]) \in M \\ R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}]) \wedge R(C^j, C^k, [\tau^-_{j,k}, \tau^+_{j,k}]) \\ \Rightarrow \exists R^T(C^i, C^k, [\tau^-_{ik}, \tau^+_{ik}]) \end{aligned} \tag{3}$$

Definition 3. *Given a process model M, two discrete event classes* C^i *and* C^j *are not ordered when:*

$$M \vdash R^T(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}]) \wedge M \vdash R^T(C^j, C^i, [\tau^-_{ji}, \tau^+_{ji}]) \tag{4}$$

Two classes C^i and C^j that can not be ordered in a model will be denoted $C^i \| C^j \equiv C^j \| C^i$.

For examples the three sequences of the Table 1 do not provide any order between the classes C^C and C^D (Figure 1). Consequently: $C^C \| C^D \equiv C^C \| C^D$.

Property 2. *The set of discrete event classes* $C^{\|} = \{C^1, C^2, \dots, C^n\}$ *can not be ordered when:*

$$\forall C^i \in C^{\|}, \forall C^j \in C^{\|}, C^i \| C^j. \tag{5}$$

In the theory of graphs, the classes of a set $C^{\|}$ are strongly connected components. The algorithm 3 aims at detecting a potential cycle (i.e. a set $C^{\|}$ is defined):

1. Build a process model M from Ω with the $BJT4G$ algorithm.
2. Build the set $C\| = \{C^{\|}_i\}$ of the sets $C^{\|}_i$ of classes without order with the equation 4
3. $\forall C^{\|}_i \in C\|$ do
   - Remove the relations $R(C^i, C^j, [\tau^-_{ij}, \tau^+_{ij}])$ of M where $C^i \in C^{\|}_i$ or $C^j \in C^{\|}_i$.
   - Generate all the pathes $P = \{P_k\}$ with $P_k = \{R(C^i, C^{i+1}, [\tau^-_{ii+1}, \tau^+_{ii+1}])\}$ where $C^i \in C^{\|}_i$ and $C^{i+1} \in C^{\|}_i$

- Insert the relations of the pathes P in M

To avoid the adding of *a priori* knowledge about the process or the programs, the algorithm 3 computes all the paths linking the classes in $C^{\|}$ (cf. model of Figure 1 with the classes C and D). It is clear that if $Card(C^{\|}) = n$, there is $n!$ possible paths. But it is a simple way to put the emphasis on potential cycles.

3.5 Modeling a Large Scale Process

The algorithm 4 aims at modeling a large scale manufacturing process. It simply uses the three algorithms provided in the preceding sub sections. Given a set of sequences $\Omega = \{\omega_i\}$, the algorithm 4 finds a process model M with the $BJT4G$ algorithm:

1. Rewrite the sequences of Ω with the algorithm 1.
2. Produce the sets Ω_k of sub sequences ω_k^i with the algorithm 2
3. $\forall \Omega_k$ do
   - Build a process model M_k of the phase k with the algorithm 3.
4. $M = \bigcup M_k$

Applied to the sequences of the Table 1, this algorithm provides the model of the Figure 1. This algorithm has also been used to model the wafer manufacturing process of the Rousset (France) plant of the STMicroelectronics company.

4 APPLICATION

The aim of the STMicroelectronics Company is to improve the control of the wafer manufacturing process through the definition of human scale process models and a better knowledge of the timed constraints between the different steps of manufacturing. A ”wafer” is a silicon plate on which are engraved electronic chips.A wafer manufacturing process is a series of elementary treatments called ”receipts” that are made on a particular machine called ”equipment”. An ”operation” is a particular series of receipts associated with an equipment. A complete series of operations is called manufacturing ”road”. The Rousset (France) plant of the STMicroelectronics Company counts more than 5.200 receipts, 1.400 operations and more than 310 equipments. The supervision system of the wafer manufacturing process describes a manufacturing road with messages providing the name of a receipt, the machine on which the receipt is performed, the corresponding operation and the start and finish times of the receipt. The algorithm 4 is applied at the equipment level so that a process model M represents a manufacturing road with a series of equipments. For this application, the initial set of sequences Ω contains 45 sequences ω_i of occurrences of 235 discrete event classes. Each sequence has 6 to 220 event classes occurrences and is 1 to 75 days long. A class is defined with a singleton $C^i = \{(\phi_i, i)\}$ where the constant i is a natural number in the interval $[1000, \ldots, 1286]$ which denote a particular equipment

To illustrate the application of the extension of the Stochastic Approach proposed in this paper, the algorithms will be applied with the subsequences of two different sequences of the Figure 2. Naturally, the products (i.e. the wafers) follow the same series of state with these two subsequences. The $BJT4G$ algorithm produces the model of Figure 3.

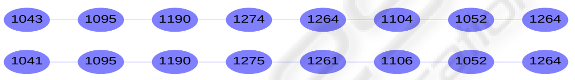

Figure 2: The Ω set of sequences ω_1 (up) and ω_2 (down).

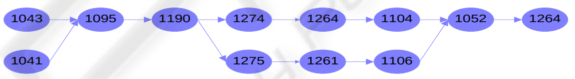

Figure 3: Model made with the $BJT4G$ algorithm with Ω.

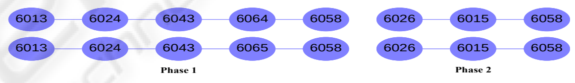

Figure 4: Process Phases (ω_k^i Subsequences).

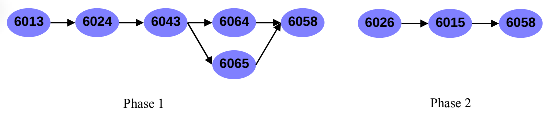

Figure 5: Model of each phase.

The equivalence classes created with the algorithm 1 are singletons of the form $C^i = \{(\phi_i, i)\}$ with $i \in [6012, \ldots, 6065]$. For example, the class of equivalence $C^{6013} = \{(\phi_{6013}, 6013)\}$ contains 15 classes: $C^{6013} = \{ C^{1031}, C^{1032}, \ldots, C^{1045} \}$. This algorithm rewrites the two subsequences of Ω (Figure 2) to produce the sub sequences of Figure 4. It is easy to see that the equivalence class of the classes C^{1043} and C^{1041} is C^{6013}, when the equivalence class of the C^{1095} is the C^{6024}. With the rewritten sequences, the algorithm 2 identifies the two process phases $\Omega_1 = \{ \omega_1^1, \omega_1^2 \}$ and $\Omega_2 = \{ \omega_2^1, \omega_2^2 \}$ of Ω (Figure 4). The algorithm 3 is then used with $\Omega = \{ \Omega_1, \Omega_2 \}$ to build the models M_1 and M_2 of Figure 5. The process model is simply provided with the union of the two models: $M = M_1 \bigcup M_2$.

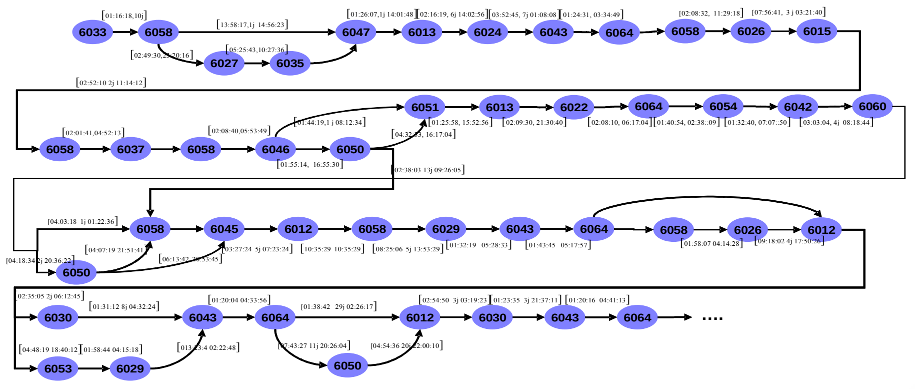

Figure 6: The first part of the process model.

Applied to the 45 sequences, the algorithm 4 builds a process model that contains 439 classes of equivalence (i.e. nodes): this process model represents a wafer manufacturing road under the form of a set of timed sequential binary relation between equipments. The Figure 6 shows the beginning of this model. Each of the 45 sequences is an instance of this model. The STMicroelectronics experts are currently analyzing this model to validate the meaning of a road represented at the equipment level of granularity. This validation task is difficult because of the size of process model.

5 CONCLUSIONS

This paper presents an extension of the Stochastic Approach framework to the modeling of manufacturing processes from the timed data contained in the supervision system database. One of the interesting features of the Stochastic Approach framework of modeling is the notion of discrete event class. This notion is used to define a process phase concept and discrete event classes of equivalence that are required large scale manufacturing processes. The definition of these concepts leads to a global algorithm that has been applied to the modeling of the electronics plates manufacturing process of the Rousset plant of the STMicroelectronics Company. This concrete application shows the operational flavor of the extensions of the Stochastic Approach Framework.

REFERENCES

Agrawal, R., Gunopulos, D., and Leymann, F. (1998). Mining process models from workflow logs. *In Sixth International Conference on Extending Database Technology*, pages 469–483.

Cook, E. J. and Wolf, A. L. (1998). Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology*, 7:215–249.

Cook, E. J. and Wolf, A. L. (2004). Event-based detection of concurrency. In *Proceedings of the 6th ACM SIGSOFT international symposium on Foundations of software engineering*, volume 53, pages 35–45.

Ghallab, M. (1996). On chronicles: Representation, on-line recognition and learning. *Proc. Principles of Knowledge Representation and Reasoning, Aiello, Doyle and Shapiro (Eds.) Morgan-Kauffman*, pages 597–606.

Le Goc, M. (2006). *Notion d'observation pour le diagnostic des processus dynamiques: Application à Sachem et à la découverte de connaissances temporelles*. Hdr, Faculté des Sciences et Techniques de Saint Jérôme.

Le Goc, M., Bouché, P., and Giambiasi, N. (2005). Stochastic modeling of continuous time discrete event sequence for diagnosis. *16th International Workshop on Principles of Diagnosis (DX'05) , California, USA.*

Pinter, S. and Golani, M. (2004). Discovering workflow models from activities' lifespans. In *Special issue: Process/workflow mining*, volume 53, pages 283–296.

Schimm, G. (2004). Mining exact models of concurrent workflows. In *Computers in Industry*, volume 53(3), pages 265–281.

van der Aalst, W., Weijters, T., and Maruster, L. (2004). Workflow mining: Discovering process models from event logs. In *IEEE Transactions on Knowledge and Data Engineering*, volume 16, pages 1128–1142.

van der Aalst, W. M. P. and Weijters, A. J. M. M. (2004). Process mining. *Special issue of Computers in Industry*, 53:231–244.

Weijters, A. and van der Aalst, W. (2003). Rediscovering workflow models from event-based data using little thumb. In *Integrated Computer-Aided Engineering*, volume 10(2), pages 151–162.