UNIVERSITE SIDI MOHAMED BEN ABDELLAH
FACULTE DES SCIENCES ET TECHNIQUES - FES
Centre d'Etudes Doctorales : Sciences et Techniques de l'Ingénieur

Fès, le 19/01./2018

N° d'ordre 01/2018

THESE DE DOCTORAT

Présentée par

Mr : Mohamed FRI

Spécialité : Génie Electrique

Sujet de la thèse :

Elaboration d'un Diagnostiqueur pour les Systèmes à Evénements Discrets et Contribution à la Prise de Décision

Thèse présentée et soutenue le 19/01/2018 devant le jury composé de

Nom Prénom	Titre	Etablissement	
LEFEBVRE Dimitri	PES	Université Le Havre - France	Président
QJIDAA Hassan	PES	Faculté des Sciences Dhar-Elmahraz - Fès	Rapporteur
MOUHSEN Ahmed	PES	Faculté des sciences et Techniques - Settat	Rapporteur
SOULHI Aziz	PES	Ecole Nationale Supérieure des Mines -Rabat	Rapporteur
ABDALI Abdelmounaïm	PES	Faculté des Sciences et Techniques - Marrakech	Examinateur
LAGRAT Ismail	PH	Ecole Nationale des Sciences Appliquées - Khouribga	Examinateur
BELMAJDOUB Fouad	PES	Faculté des Sciences et Techniques - Fès	Directeur de thèse

Laboratoire d'accueil : Techniques Industrielles

Etablissement : Faculté des Sciences et Techniques – Fès.

Remerciements

Je tiens en tout premier lieu à remercier profondément mon directeur de thèse **FOUAD BELMAJDOUB.** Cette thèse n'aurait jamais vu le jour sans sa patience, ses nombreux conseils et le temps qu'il m'a consacré. J'aimerais exprimer mon admiration pour sa passion et son enthousiasme qui m'ont influencé depuis mes études en Master. Je le remercie du fond du cœur tout aussi pour avoir dirigé mes travaux avec talent que pour m'avoir accompagné amicalement dans mon cheminement. Ce fut un réel plaisir de vous avoir comme directeur de thèse. Je vous en suis infiniment reconnaissant.

Je tiens à remercier vivement mes professeurs : **DIMITRI LEFEBVRE** et **MOHAMMED EL HAMMOUMI** pour leurs conseils précieux, pour leur encouragement et leurs remarques, durant nos discussions qui m'ont démontré encore plus leurs qualités scientifiques et humaines. Ils ont largement contribué à l'aboutissement de cette thèse. Qu'ils trouvent ici l'expression de mes profondes gratitudes.

Je voudrais aussi remercier les professeurs **ABDALI Abdelmounaïm, LAGRAT Ismail, MOUHSEN Ahmed, QJIDAA Hassan, SOULHI Aziz, ZOUAK Mohcine** qui m'honorent en participant au jury de cette thèse. Je vous remercie tous mes professeurs pour avoir répondu patiemment à toutes mes questions pendant ma recherche doctorale.

Merci à toute l'équipe de LTI, qui a été ma maison de recherche ces années de thèse je tiens à remercier tout particulièrement **AMAL BOUKILI, MOHAMMED MSAAF, HAJAR ISMAILI ALAOUI, OMAIMA BENATIA**… Je souhaite bon courage aux thésards restants et à ceux qui viennent de commencer.

Je n'oublierai pas de remercier tout le corps enseignant et administratif de la Faculté des Sciences et Techniques de Fès pour leurs accueils et leurs aides. Je remercie toutes les personnes que j'ai peut-être oublié de citer ici et qui ont contribué de près ou de loin au bon déroulement de ma thèse et à sa rédaction.

Je remercie tous mes amis et mes collègues pour leur soutien moral et matériel, et surtout mes amis-frères **RACHID OUCHEKH, MOSTAFA EL MOUMNI, AALAE BENALI, ANASS JAMALI, ILHAM ZEROUK, ANASS ELAROUSSI**…

Je termine enfin par ceux que ne saurai et que je ne pourrai jamais remercier par des mots, je pense à mes parents **AHMED** et **ELGHALIA**, mes frères **MOUHSENE** et **ABDELAH**, mes sœurs, **HOUDA** et **NADA**, et mon oncle **FRI AZIZ**. Vos encouragements et votre soutien ne m'ont jamais fait défaut, je vous dédie ce travail et j'espère être toujours à la hauteur de vos aspirations.

Table des matières

Liste des abréviations

SED	Systèmes à Evénements Discrets
SDED	Systèmes Dynamique à Evénements Discrets
CS	Contrôle de Supervision
API	Automates Programmables Industriels
VBA	Visual Basic for Applications
CPU	Unité centrale de traitement (central processing unit)
GRAFCET	Graphe Fonctionnel de Commande des Étapes et Transitions
SCF	Sequential Chart Function
ARS	Allocation de Ressource Séquentielles
RP	Réseaux de Pétri
RDP	Réseaux de Pétri
TC	Temps Codés
Ti	instant d'arrivé i
Ai	Action i
$d_{min\ i}$	Dates de tir les plus anciennes
$d_{max\ i}$	Dates de tir les plus récentes
C	Contrainte temporelle
TIi	Intervalle de temps
DCy	Début de Cycle
CEI	International Electrotechnical Commission
DMC	Durée Minimale du Cycle
MEAC	Marge d'Erreur Admissible par le Cycle
DEMEi	Durée d'Exécution Minimale d'une Etape i
MEAEi	Marge d'Erreur Admissible par l'une Etape i
t_{di}	Temps de départ i
$t_{fin\ i}$	Temps de fin minimal de l'étape i

$t_{max\ i}$	Temps de fin maximal de l'étape i
$t_{min\ i}$	Temps de départ minimal de l'étape i
θ_i	Temps de départ plus tard de l'étape i
AUTO	Automatique
MANU	Manuelle
MAD	MArocain Dirham
KP	Knapsack Problem
W	Capacité du sac
ωi	le poids de l'objet « i »
Pi	la valeur de l'objet « i »
X	Contenu
AG	Algorithme de Greedy
P	Pareto
PKPGA	Knapsack Problem, Pareto et Algorithme de Greedy
∀i	Quel que soit i
Kg	Kilogramme

Liste des tableaux

Liste des figures

‘Si tu crois, tu auras la force’

A toi…

Introduction Générale

Contexte

Un Système à Evénements Discrets (SED) est un système dynamique dans lequel l'état de l'espace est discret. Ses trajectoires sont composées par des états constants par morceaux. Un tel système fonctionne en fonction de l'apparition d'événements physiques à des intervalles qui sont généralement irréguliers ou inconnus. Il existe de nombreux domaines d'application pour les systèmes à événements discrets : allant de l'ingénierie informatique ou chimique à la fabrication aux systèmes de transport intelligents. Différents aspects du comportement d'un système peuvent être considérés selon l'application. Par conséquent, divers outils pour la modélisation et l'analyse des systèmes à événements discrets ont été développés. L'évolution du système à événements discrets se caractérise par l'apparition des événements. Selon la façon dont un modèle organise des événements, nous pouvons classer les modèles des systèmes à événements discrets en deux catégories : modèles non-temporels et temporels. Les modèles de systèmes à événements discrets désactivés ne tiennent compte que de l'apparition d'événements dans leur ordre séquentiel. Ces modèles ignorent les dates d'occurrence des événements et ils sont utilisés pour l'étude des propriétés qualitatives du système à événements discrets. Dans d'autres applications, les informations sur le temps sont essentielles et doivent être prises en compte par le modèle. Les modèles qui ont cette caractéristique sont des modèles chronométrés où le temps peut être déterministe ou stochastique.

Les systèmes à événements discrets, comme la dynamique des véhicules, les moteurs aéronautiques, les procédés chimiques, le réseau électrique, les machines électriques, les systèmes de fabrication, les systèmes de conversion d'énergie éolienne et les équipements électroniques industriels sont des systèmes critiques pour la sécurité. La fiabilité et la sécurité des systèmes industriels soumirent à des anomalies potentielles des procédés et des défauts de composants deviennent de plus en plus fréquents. En conséquence, il est primordial de détecter et d'identifier tout type d'anomalies et de défauts potentiels le plus tôt possible, ensuite de les corriger, surtout ceux qui sont plus critiques, afin de minimiser la dégradation des performances, d’éviter les situations dangereuses et améliorer la productivité.

Contributions

Dans ce contexte, le travail présenté dans ce mémoire s'inscrit de manière générale dans la problématique de diagnostic et de la maintenance des systèmes à événements discrets.

Notre contribution dans l'éclaircissement de la problématique du diagnostic et maintenance des systèmes à événements discrets peut être présentée en trois points :

- Nous avons mené une étude sur plus de 2000 articles afin de déterminer la tendance et l'actualité de la recherche scientifique dans le domaine des systèmes à événements discrets. Cette étude était divisée en trois articles que nous avons publiés dans la base de données THOMSON REUTERS.
- Nous avons élaboré un diagnostiqueur pour les systèmes à événements discrets. Ce diagnostiqueur était développé sur trois stades. Commençant par un diagnostiqueur pour les systèmes séquentiels passant par un autre pour tous les types de système à événements discrets arrivant au système à événements discrets avec des événements non observables. Dans ce contexte, nous avons publié deux articles un dans la base de données SCOPUS et l'autre dans la base de données THOMSON REUTERS, et nous avons participé à cinq congrès scientifiques dont trois sont indexées IEEE.
- Nous avons contribué dans la prise de décision dans la maintenance industrielle par :

  - La projection de deux méthodes qui sont utilisées dans le domaine de transport et de la logistique dans notre axe de recherche.
  - L'élaboration de deux nouvelles méthodes afin de bien choisir les éléments les plus critiques à maintenir.

Ces contributions étaient approuvées par la publication de trois articles dans la base de données THOMSON REUTERS.

Structure du mémoire

Ce manuscrit est composé de trois parties.

La partie A se compose de deux chapitres : le premier renferme les différents types de systèmes (Système linéaire et non linéaire, Système statique et système dynamique, Système anti-causal et système causal, Système stationnaire et non stationnaire, Système continu et les systèmes à événements discrets). Ensuite nous avons fait recours à leurs définitions, leurs domaines d'application et les problèmes rencontrés dans ces systèmes en général. Dans le deuxième chapitre, nous présentons l'analyse du méga-data des articles publiés dans le domaine des SED.

La partie B se compose de trois chapitres, elle est commencée par la présentation des défauts et ses types ensuite le chapitre entame une étude générale sur les méthodes du diagnostic. Le chapitre trois englobe trois Diagnostiqueurs sous contraintes temporelles. Le premier est destiné au système séquentiel. Le deuxième algorithme est destiné pour tous les types de système : séquentiel, parallèle et au choix. Le troisième diagnostiqueur est dédié pour les SED avec des événements silencieux.

La partie C contient deux chapitres. Le premier chapitre commence par l'explication de la loi de Pareto puis la présentation d'une méthode élaborée appelée Diagramme de CBA et finit par une étude comparative entre les deux méthodes. Le deuxième chapitre commence par l'explication de la méthode de sac à dos et l'algorithme de Greedy avec un exemple explicatif. Ensuite une étude comparative de ces méthodes est appliquée sur une structure de production des boissons gazeuses. Après une méthode qui combine les trois outils : Pareto, problème de Knapsack, KP et Algorithme de Greedy nommé "PKPGA" a été présenté est appliqué sur la même étude de cas.

Notre rapport de thèse est finalisé par une conclusion et des perspectives de nos travaux.

Partie A

Systèmes à Evénements Discrets

Introduction

Les gens observent différents phénomènes de la nature, ils se sont efforcés de les comprendre. La première étape est le reflet des phénomènes par l'imagination et la description. Le processus réflexif est un processus d'abstraction. Dans ce processus, la notion de « système » est d'une importance fondamentale.

Dans le premier chapitre de cette partie, on va présenter les différents types de systèmes et plus spécifiquement les systèmes à événements discrets qui sont présents dans toute industrie, chaîne de fabrication et même dans la vie quotidienne. Ils ont une influence capitale sur le développement des humains. On va exposer leurs définitions, les domaines d'application et les problèmes rencontrés dans ce type des systèmes.

Dans le deuxième chapitre, on va présenter l'analyse du méga-data des articles publiés dans le domaine des systèmes à événements discrets. Nous avons publié cette étude dans des journaux indexés en THOMSON REUTERS. Ces articles sont comme des piliers pour notre étude. L'analyse de plus de 2000 articles et la mise en place de cette étude statistique des publications précédentes permettront aux chercheurs de savoir la tendance de l'orientation scientifique dans le domaine des Systèmes à Evénements Discrets(SED). Elle leur permet d'évaluer le niveau de maturité de chaque outil dédié au SED. Dans notre rapport de thèse, on va présenter des courbes symbolisant le nombre d'articles en fonction de leurs années de publication afin d'éclaircir la tendance de la recherche scientifique dans le domaine des SED. Cette étude nous a permis de bien choisir notre axe de recherche avec exactitude.

Chapitre 1

Les Systèmes à Evénements Discrets

1.1 Système

Le concept d'un système est subtil et il change du domaine à l'autre, dont la compréhension pourrait mieux être laissée à l'intuition plutôt qu'une définition exacte. Néanmoins, nous pouvons nous baser sur la définition de la littérature fournie par : IEEE Standard Dictionary of Electrical and Electronic Terms « Un système est une combinaison de composants qui agissent ensemble pour exécuter une fonction, impossible d'être effectuée avec l'un des composants individuels. » En nous basant sur cette définition, nous déduisons qu'il y a deux principales caractéristiques pour un système. Tout d'abord, un système est constitué d'interagit « composants », il est associé à une « fonction » qui doit vraisemblablement l'effectuer. Ce système reçoit des entrées u (t) et génère des sorties y (t) comme le montre la figure 1.

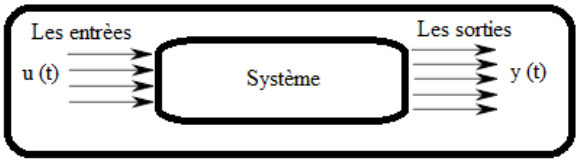


Figure 1 : Représentation d'un système

La figure montre que le système reçoit des entrées u (t) et génère des sorties y(t). La relation entre les entrées et les sorties caractérise le système. Les propriétés importantes (ou les classifications) des systèmes sont présentées dans les paragraphes suivants, et détaillés dans cette référence [1].

1.1.1 Système linéaire et système non linéaire

Un système peut être linéaire ou non linéaire comme il le présente la figure 2. Un système linéaire est un objet du monde matériel qui obéit au principe de superposition : considérons un système ayant deux entrées. Entrée $u_1(t)$ produit en sortie $y_1(t)$ et l'entrée $u_2(t)$ produit en sortie $y_2(t)$. Considérons maintenant deux constantes arbitraires a_1 et a_2, il suffit de multiplier ces constantes avec entrée $u_1(t)$ et $u_2(t)$ respectivement. Par conséquent, $a_1*u_1(t)$ produit la

sortie $a_1*y_1(t)$ et $a_2*x_2(t)$ produit la sortie $a_2*y_2(t)$. Ainsi Le système linéaire a une caractéristique importante : Si l'entrée du système est zéro, il génère une sortie zéro.

Si le système génère une ou quelque sorties non nulles avec une entrée de valeur zéro, alors le système est dit système non linéaire. Les systèmes non linéaires sont plus difficiles à étudier que les systèmes linéaires. Néanmoins, en linéarisant (quand c'est possible) un système non linéaire autour d'un point d'équilibre ou d'une trajectoire, on obtient un système linéaire qui représente correctement le système non linéaire au voisinage de ce point d'équilibre ou de cette trajectoire.

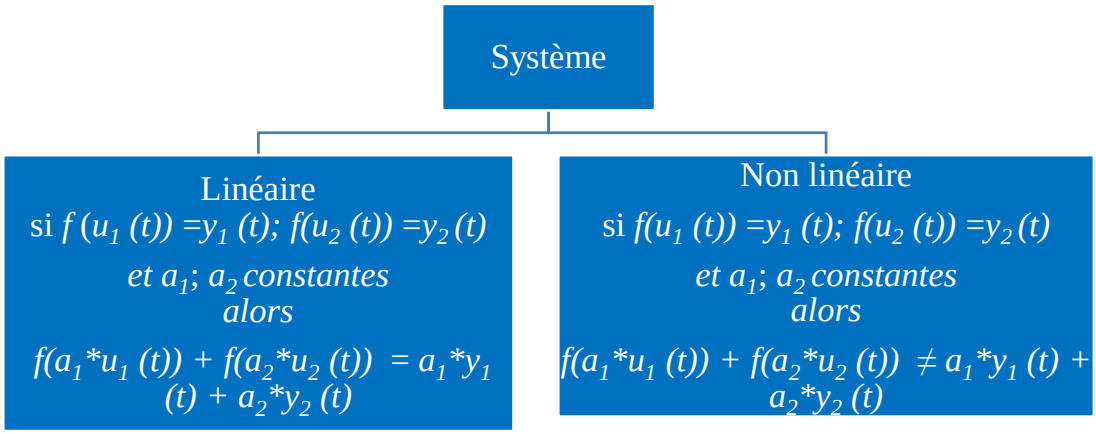


Figure 2 : Système linéaire et système non linéaire

1.1.2 Système statique et système dynamique

Un système peut être statique ou dynamique comme il le présente la figure 3. On dit qu'un système est statique lorsque la sortie $y(t)$ est indépendante des valeurs passées de l'entrée $u(\tau)$, pour toute $\tau < t$. Un système dynamique est celui dans lequel la sortie $y(t)$ dépend de l'entrée $u(t)$ à l'instant présent et d'un autre instant. Ainsi, la détermination de la sortie d'un système statique ne nécessite pas de "mémoire" de l'historique d'entrée, ce qui est le cas pour un système dynamique.

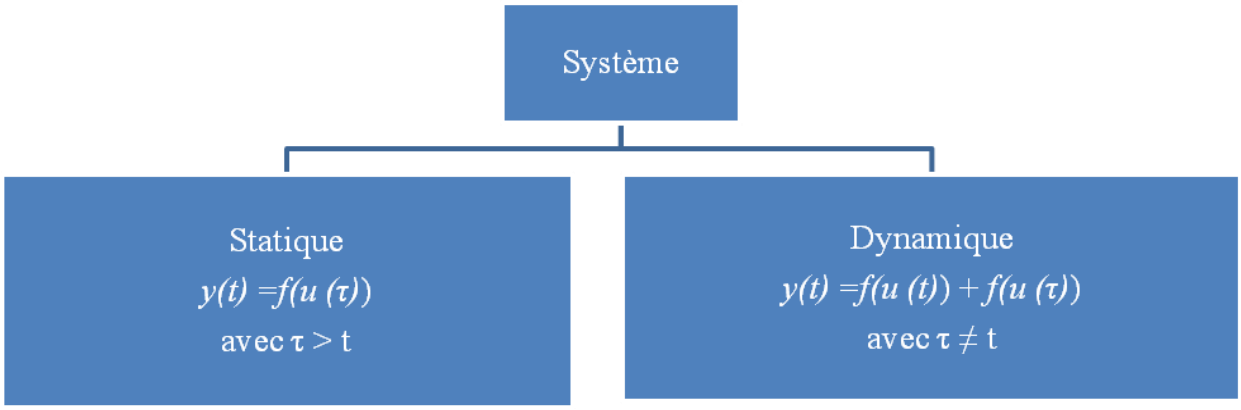


Figure 3 : Système statique et système dynamique

1.1.3 Système anti-causal et système causal

Un système peut être causal ou anti-causal comme il est illustré dans la figure 4. Un système anti-causal est un système dont la sortie dépend des valeurs présentes et futures de l'entrée. Un système anti-causal est pratiquement irréalisable : cela signifie que dans des cas pratiques, il est impossible de mettre en œuvre un système anti-causal, mais si les signaux sont stockés dans la mémoire pour être utilisés après, ils seront traités en tant que signaux avancés ou futurs, car ils étaient déjà présents avant même le fonctionnement du système. Dans tels cas, il est possible de mettre en œuvre un système anti-causal.

Un système est dit causal si sa sortie dépend des entrées passées et présentes et non des entrées futures. Ce type de système est pratiquement réalisable.

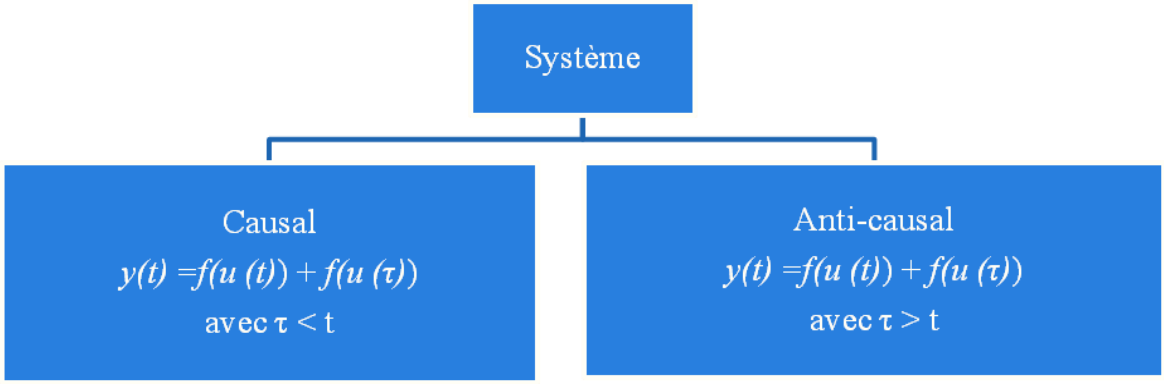


Figure 4 : Système causal et système anti-causal

1.1.4 Système stationnaire et système non stationnaire

Un système peut être stationnaire ou non-stationnaire comme il est illustré dans la figure 5. Un système non stationnaire (varie en temps) est un système qui change sa sortie en fonction du décalage temporel ; par exemple pour une entrée $u_1(t)$ on a comme sortie $y_1(t)$ et pour la même entrée à l'instant $(t+\alpha)$ on a comme sortie $y_2(t) \neq y_1(t+\alpha)$.

Un système est dit stationnaire (invariable dans le temps) si ses caractéristiques de sortie et d'entrée ne changent pas avec le temps. Par exemple : Si une translation du temps est appliquée à l'entrée, elle sera appliquée à la sortie. Un système subissant une lente variation du temps par rapport à ses constantes, peut généralement être considéré comme un système invariant dans le temps : ils sont presque invariants dans le temps sur une petite période. Un exemple de ceci est le vieillissement et l'usure des composants électroniques, qui se passe dans une grande période, ce qui donne lieu à un comportement qualitativement différent de celui observé dans une petite période pour un système invariant.

Les systèmes invariants en temps pour une différence temporelle de quelques jours, peuvent devenir variants si on tient en compte une période large en années, puisque les paramètres peuvent être modifiés.

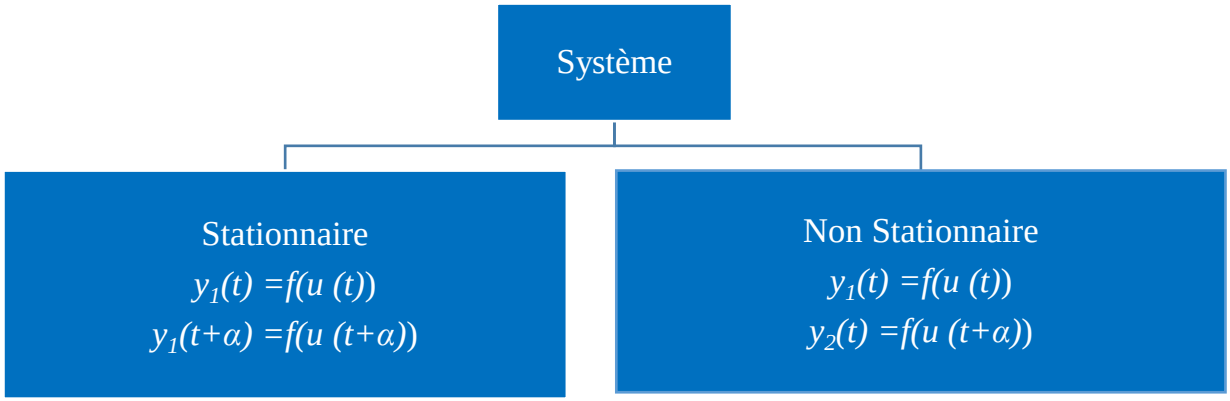


Figure 5 : Système stationnaire et système non stationnaire

1.1.5 Système continu et système discret :

Nous avons supposé jusqu'ici que le temps est une variable continue. Cela correspond certainement à notre notion de base dans le monde physique. Ces systèmes sont généralement décrits par des équations différentielles ou différentielles partielles auxquelles obéissent les phénomènes physiques correspondants. L'avènement des ordinateurs dérive l'évolution de ces systèmes en utilisant des équations dynamiques de temps discret, ce qui ne modifie pas la nature continue intrinsèque de ces systèmes.

Avec les progrès de la technologie, l'homme a commencé à construire des systèmes de plus en plus complexes et complètement artificiels, au moins au niveau conceptuel de leur fonctionnement, qui convient à leur gestion et à leur contrôle. Par exemple : Réseaux de transport, réseaux de communication et d'ordinateurs, unité centrale de traitement des ordinateurs, ateliers de fabrication… Dans ces systèmes, c'est du principal mécanisme dynamique dans la succession

de tâches que découlent les phénomènes suivants : la synchronisation, l'exclusion mutuelle ou la concurrence dans l'utilisation de ressources communes. Celles-ci exigent une politique pour arbitrer les conflits et définir les priorités…

Ce type de dynamique ne peut pas être capturé par des équations différentielles ou par leurs analogues de temps discrets. C'est certainement la raison pour laquelle ces systèmes, qui sont pourtant de véritables systèmes dynamiques, ont longtemps été ignorés par les experts en contrôle automatique. Et ils ont été plutôt considérés par les chercheurs opérationnels, les spécialistes de la fabrication, les informaticiens, etc., selon le domaine d'intérêt de l'application, alors qu'aucune théorie générale du système n'a réussi à émerger. Cependant, on peut mentionner la théorie graphique (réseaux de Petri), la théorie probabiliste (réseaux de files d'attente), etc., qui n'avaient pas de liens étroits avec la théorie des systèmes.

Ce qui était nouveau dans les années 80, c'était le fait que le monde du contrôle automatique prenait en compte ces nouveaux systèmes. On les appelait alors «Système à Evénements Discrets» (SED). Le mot «discret» ne signifie pas que «le temps est discret » ni implique nécessairement que «l'état soit discret» (en effet, comme on peut le voir, les variables d'état peuvent prendre des valeurs continues), mais ce mot se réfère au fait que la dynamique se compose d'événements. Ces événements peuvent éventuellement avoir une évolution continue une fois qu'ils commencent, mais ce n'est pas ce dont on s'intéresse. L'objectif principal est le début et la fin de ces événements, car les fins peuvent provoquer de nouveaux débuts. Ceci est dû au mécanisme de transition d'état qui est normalement basé sur des instructions logiques simples de la forme "si quelque chose se produit et que l'état courant est x_0, alors l'état suivant devient x_1". Il convient de souligner que le terme « Système Dynamique à Evénements Discrets » (SDED) est également et couramment utilisé pour souligner l'importance du comportement dynamique de tels systèmes [1].

1.2 Systèmes à événements discrets

Les Systèmes à Evénements Discrets (SED) sont des systèmes linéaires, dynamiques, causaux, stationnaires et discrets en temps et 'par événement' comme il montre la figure 6. Les SED forment une classe importante de systèmes dynamiques. Le terme a été introduit à la fin des années 1970 pour décrire un système en fonction de son aspect le plus critique : le fait que son comportement est régi par des événements discrets. Ces événements se produisent de façon asynchrone dans le temps et qui sont uniquement responsables de la génération des transitions d'état.

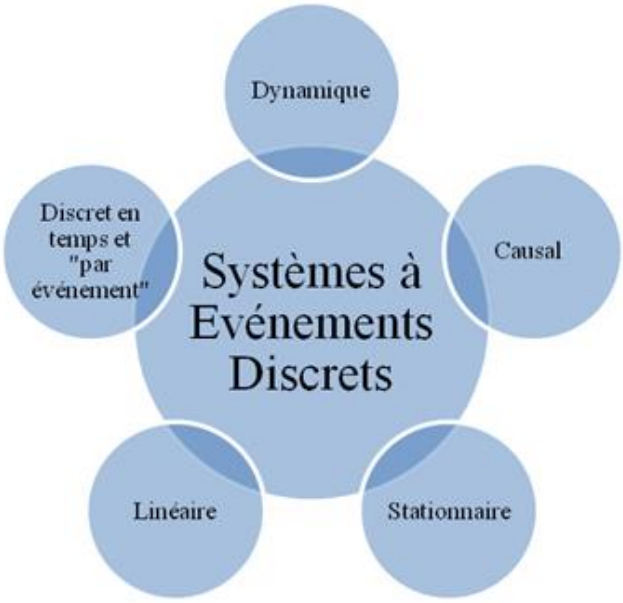


Figure 6 : Caractéristiques des Systèmes à Evénements Discrets

Entre les occurrences d'événements, l'état d'un SED n'est pas affecté. Les exemples de tels comportements abondent dans les environnements technologiques, y compris les réseaux informatiques et de communication, les systèmes de fabrication, les systèmes de transport, la logistique, etc.

Le fonctionnement d'un SED est en grande partie régi par des règles souvent non structurées et souvent faites par l'homme, comme pour lancer ou terminer des activités et programmer l'utilisation de ressources par des événements contrôlés. D'autre part, leur fonctionnement est également soumis à des événements aléatoires non contrôlés (par exemple, une défaillance spontanée de l'équipement) qui peuvent être observés ou non par des capteurs.

Il existe deux aspects d'un SED qui définissent son comportement :

- Les variables impliquées sont à la fois continues et discrètes, parfois purement symboliques, c'est-à-dire non numériques (par exemple, décrivant l'état d'un feu de circulation comme "rouge" ou "vert"). Ceci rend les modèles mathématiques traditionnels basés sur des équations différentielles des méthodes inadéquates et connexes basées sur le calcul d'utilisation limitée.
- En raison de la nature asynchrone des événements qui provoquent des transitions d'état dans un SED, il n'est ni naturel ni efficace d'utiliser le temps comme élément de synchronisation conduisant sa dynamique. C'est pour cette raison que les SED sont souvent appelés événementiels, pour les opposer aux systèmes classiques basés sur les

lois de la physique ; dans ce dernier, au fur et à mesure que le temps évolue, des variables telles que la position, la vitesse, la température, la tension, etc. évoluent aussi continuellement. Cependant, pour capturer la dynamique d'état axée sur les événements, de différents modèles mathématiques sont nécessaires.

En outre, les incertitudes sont inhérentes aux environnements technologiques où les SED sont rencontrés. Par conséquent, les modèles mathématiques associés et les méthodes d'analyse et de contrôle doivent intégrer de telles incertitudes. Enfin, la complexité est également inhérente aux SED d'intérêt pratique, se manifestant habituellement sous la forme d'espaces d'états combinatoires explosifs. Bien que les méthodes purement analytiques pour la conception, l'analyse et le contrôle du SED soient limités, elles ont encore permis des approximations fiables de leur comportement dynamique et de la dérivation de propriétés structurelles utiles et de garanties de performance prouvables. Cependant, une grande partie des progrès réalisés dans ce domaine s'est appuyée sur de nouveaux paradigmes caractérisés par une combinaison de techniques mathématiques, d'outils informatiques et d'un traitement efficace des données expérimentales.

1.2.1 Les applications des systèmes à événements discrets

La théorie des systèmes à événements discrets (SED) est apparue à la fin des années 1970 après les efforts de la communauté des contrôleurs fournis pour répondre aux besoins de contrôle des applications concernant certaines opérations complexes de production et de service. A titre d'exemple, les besoins qui se tiendront à la fabrication et d'autres systèmes de workflow, de télécommunication et de traitement de données, et les systèmes de transport. Ces opérations visaient la capacité de soutenir des niveaux plus élevés d'efficacité et de productivité et des notions plus exigeantes sur la qualité du produit et du service. En même temps, les technologies de pointe de l'informatique de l'époque, et en particulier l'émergence du microprocesseur, cultivent, et dans une mesure significative de support, des visions de plus en plus d'automatisation et d'autonomie pour les opérations mentionnées ci-dessus. La communauté SED s'efforce de fournir une approche systématique et une compréhension rigoureuse de la dynamique. Cela permet de modéliser les opérations précitées et leur complexité, et de développer un modèle de contrôle qui définit et fait respecter les comportements visés pour ces environnements d'une manière efficace et robuste.

Afin de répondre aux objectifs susmentionnés, la communauté de contrôle a dû élargir sa base méthodologique, empruntant des concepts, des modèles et des outils d'autres disciplines.

Parmi ces disciplines, les deux suivantes ont joué un rôle particulièrement important dans le développement de la théorie SED : l'informatique théorique et la recherche opérationnelle. En tant que nouveau domaine de recherche, le SED s'est appuyé sur la force analytique et les synergies résultantes de l'intégration rigoureuse des cadres de modélisation qui ont été empruntées à l'informatique théorique et la recherche opérationnelle. En outre, la communauté SED a considérablement étendu ces contremaitres empruntés, en leur apportant beaucoup de ses perspectives et concepts de la théorie de contrôle.

En général, les approches basées sur le SED sont caractérisées par :

- Leur accent sur une représentation rigoureuse et une représentation formelle des systèmes étudiés et de leurs dynamiques.
- La focalisation porte sur les aspects temporels et les mesures qui définissent les notions traditionnelles/standards de performance pour les systèmes considérés .Cette focalisation porte aussi sur une analyse plus axée sur le comportement qui est nécessaire pour assurer des notions fondamentales de « correction », de « stabilité » et de « sécurité » du fonctionnement du système, en particulier dans le contexte des niveaux d'autonomie aspirés.
- L'interaction entre les deux aspects d'analyses mentionnés au deuxième point et le lien supplémentaire de cette analyse aux attributs structurels du système.
- L'effort pour compléter les caractérisations analytiques et les développements par des procédures et des outils de conception. Ces derniers fournissent des solutions qui sont compatibles avec les spécifications proposées et qui sont effectivement réalisables dans les contraintes de temps et d'autres ressources imposées par la nature « en temps réel » sur l'objectif des applications.

Les systèmes à événements discrets modélisent, diagnostiquent, supervisent et contrôlent les réseaux de communication, les systèmes de base de données, les réseaux de circulation, les circuits numériques, les systèmes de fabrication, les chaînes de fabrication d'automobile et d'aéronautique, et les trains de laminoirs, etc.

1.2.2 Vue générale sur les problèmes et les applications du SED

Il y a une infinité des problèmes rencontrés dans les SED et c'est difficile de les énumérer, mais il est convenable de les présenter suivant trois axes.

1.2.2.1 Modélisation, contrôle et diagnostic

La caractérisation de base du comportement dans le cadre de la théorie des SED est à travers les diverses séquences d'événements qui peuvent être générées par le système. Collectivement, ces séquences sont connues sous le nom de langage formel généré par le système automatisé, l'intention première est de restreindre son comportement dans un sous-ensemble des chaînes d'événements générés. L'étude de ce problème est encore facilitée par l'introduction de certains mécanismes qui agissent comme des représentations formelles des systèmes étudiés, dans le sens qu'ils génèrent les mêmes chaînes d'événements (c'est-à-dire le même langage formel). Comme ces modèles sont concernés par la représentation des séquences d'événements qui sont générées par SED, et non par la synchronisation exacte de ces événements, ils sont fréquemment caractérisés comme des modèles SED non temporisés. Dans les applications pratiques de la théorie des SED, les modèles les plus populaires sont l'automate à états finis [2] et le Réseau de Petri [1] [3].

Dans le contexte des applications des SED, ces cadres de modélisation ont été utilisés pour fournir des caractérisations succinctes de la dynamique axée sur les événements et pour concevoir des contrôleurs, sous la forme de superviseurs. Ces contrôleurs qui restreindront ces dynamiques pour qu'ils respectent la sécurité, la cohérence, l'équité, et d'autres considérations similaires. Donnons un exemple plus concret, dans le contexte de la fabrication contemporaine, le contrôle comportemental basé sur le SED souvent appelé contrôle de supervision a été promu comme une méthodologie systématique pour la synthèse et la vérification de la logique de contrôle. Ceci est nécessaire pour le soutien de la fonction dite (contrôle de supervision et acquisition de données). Cette fonction de contrôle est généralement mise en œuvre par l'intermédiaire des Automates Programmables Industriels (API) qui ont été employés dans les ateliers de la fabrication contemporaine. La théorie du Contrôle de Supervision des SED a pour rôle de :

- Fournir plus de rigueur et de spécificité aux modèles utilisés pour le système et au comportement des processus.
- Offrir la possibilité de synthétiser les politiques de contrôle qui sont vraisemblablement correctes par la construction de système.
- Quelques exemples d'ouvrages qui ont poursuivi l'application de SED dans ce sens peuvent être trouvés dans [4] [5] [6] [7] [8] et [9].

D'autre part, l'activité susmentionnée a également défini un besoin supplémentaire d'interfaces pertinentes qui traduiront :

- La structure du processus et le comportement cible aux modèles SED théoriques nécessaires.
- Les politiques obtenues aux exécutables API.

Ce besoin a conduit à une ligne de recherche, en termes de modèles, de représentation et d'outils de calculs, qui sont complémentaires des développements fondamentaux du SED décrits dans les paragraphes précédents. Nous mentionnons les premières études qui se portaient sur le développement de GRAFCET (Graphe Fonctionnel de Commande des Etapes et Transitions) [10] et des diagrammes de fonctions séquentielles [11], alors que certains efforts plus récents dans ce sens sont rapportés dans [12] et [13].

Outre son emploi dans le domaine de la fabrication, la théorie de contrôle de supervision des SED a également été envisagée pour la coordination des processus de communication qui s'exécutent dans divers systèmes embarqués [14]. La validation systématique des logiciels embarqués qui sont utilisés dans diverses applications de contrôle, allant des systèmes de puissance et des centrales nucléaires aux aéronefs et à l'électronique d'automobile [15]. La synthèse de la logique de commande dans les commutateurs électroniques utilisés dans les réseaux de télécommunication et de données ; et la modélisation, l'analyse et le contrôle des opérations qui ont lieu dans les systèmes de soins dans la santé [16]. [17], donne un aperçu très intéressant des gains, mais aussi des défis considérables rencontrés par une équipe de chercheurs qui ont essayé d'appliquer des méthodes formelles, semblables à celles qui ont été promues par la théorie comportementale du SED, au développement et à la certification du logiciel qui gère certaines opérations critiques pour les centrales nucléaires canadiennes.

Outre le contrôle, des modèles SED non étalés ont également été utilisés pour le diagnostic d'événements critiques, comme certains échecs, qui ne peuvent pas être observés explicitement, mais leur occurrence peut être déduite de certains schémas comportementaux résultants [18]. Plus récemment, la méthodologie pertinente a été étendue avec une capacité de pronostic [19], tandis qu'une variante intéressante traite le problème qui concerne la conception des systèmes où certains événements ou schémas comportementaux doivent rester indétectables par un observateur externe. Ce dernier n'a le droit que d'une observation partielle du comportement du système ; cette exigence a été formellement caractérisée par la notion d'« opacité » dans la littérature pertinente et elle trouve son application dans la conception et le fonctionnement des systèmes sécurisés [20] [21] [22].

1.2.2.2 La complexité computationnelle

Comme le révèle la discussion des paragraphes précédents, un bon nombre des applications de la théorie de contrôle de supervision concernant l'intégration et la coordination du comportement qui est généré par un certain nombre de composants interagissant dans ces cas, les modèles formels qui sont nécessaires pour la description du comportement du système peuvent croître très rapidement, par la suite leurs tailles, les algorithmes impliqués dans l'analyse comportementale et la synthèse de contrôle peuvent devenir pratiquement insolubles. Néanmoins, la base méthodologique rigoureuse qui sous-tend la théorie du SED fournit également un cadre pour aborder ces défis informatiques de manière efficace et structurée.

Plus précisément, la théorie du contrôle de supervision fournit des conditions dans lesquelles les spécifications de contrôle peuvent être décomposables aux composants constitutifs de l'installation tout en maintenant l'intégrité et la correction du comportement global de l'installation [23]. Les travaux susmentionnés de [5] et de [8] fournissent quelques exemples concrets pour l'application de la synthèse de contrôle modulaire. D'autre part, il existe des problèmes fondamentaux abordés par la théorie et la pratique du contrôle de supervision qui nécessitent une vision holistique du système et de son fonctionnement et, par conséquent, ils ne sont pas sujets à des solutions modulaires.

Dans ce cas, la théorie de contrôle de supervision fournit encore des solutions efficaces en identifiant la structure spéciale des processus, d'une pertinence pratique, pour laquelle les superviseurs cibles peuvent être mis en œuvre de manière efficace du point de vue du calcul et fournit systématiquement les spécifications originales pour la traçabilité informatique. Une application particulière qui a bénéficié, et en même temps, a significativement promu cette dernière capacité de la théorie SED CS, est celle concernant le fonctionnement sans blocage de nombreux systèmes concurrents ; où un ensemble de processus qui s'exécutent simultanément et de manière échelonnée sont en compétition, à chacun ses étapes de traitement, pour l'attribution d'un ensemble fini de ressources réutilisables. Dans la théorie SED, ce problème est connu sous le nom d'Allocation de Ressource Séquentielle (ARS) [24] et sous-tend le fonctionnement de nombreuses applications contemporaines : l'allocation de ressources dans les ateliers de fabrication contemporains [25] [26] [27] [28], le chemin de fer automatisé [29] et d'autres systèmes de circulation guidée [30]. Et couvre aussi les systèmes de gestion du flux de travail basé sur Internet comme ceux envisagés pour le commerce électronique et certaines applications de traitement des

demandes bancaires et d'assurance [31]. Ainsi, il inclut l'attribution des sémaphores qui contrôlent l'accessibilité des ressources partagées en exécutant simultanément des threads dans des programmes informatiques parallèles [32]. Dans les articles [33] et [34], les auteurs présentent une introduction systématique à la modélisation de l'ARS basée sur le SED, et certains développements plus récents sont décrits dans [35] [36] [33].

En clôturant la discussion ci-dessus sur la capacité de la théorie SED dans laquelle nous avons abordé efficacement la complexité du problème de Contrôle de Supervision, nous devons souligner que les mêmes mérites de la théorie ont également permis la gestion efficace de cette complexité liée à la modélisation, la performance et le contrôle des différentes applications des SED.

1.2.2.3 Contrôle, Performance et Interaction entre structures

La théorie SED est également intéressée par la modélisation de la performance, l'analyse et le contrôle de ses applications cibles. Les aspects liés au temps comme le débit, l'utilisation des ressources, les latences expérimentées et les modèles de congestion. Pour supporter ce type d'analyse, les modèles comportementaux SED non temporels sont étendus à leurs versions temporelles. Cette extension se fait en dotant les modèles non temporels d'origine d'attributs supplémentaires qui caractérisent les retards technologiques entre l'activation d'un événement et son exécution (à condition qu'il ne soit pas préempté par un autre événement conflictuel). Les modèles temporels sont ensuite classés selon l'étendue et la nature du caractère aléatoire qui est capturé par eux. Une telle catégorisation fondamentale est faite entre les modèles déterministes, où les délais susmentionnés prennent des valeurs fixes pour chaque événement, et les modèles stochastiques qui admettent des distributions plus générales. Du point de vue de l'application, les modèles SED temporisés relient la théorie SED à la multitude des applications qui ont été traitées par la programmation dynamique, le contrôle stochastique et la théorie de l'ordonnancement [37] [38] [39]. De plus, dans leur définition la plus générale, les modèles stochastiques de SED fournissent le fondement théorique de la simulation d'événements discrets [40].

Comme dans le cas de la théorie du comportement, la préoccupation pratique, qui remet en question : l'application de modèles temporels de SED pour la modélisation, l'analyse et le contrôle des performances, est la très grande taille de ces modèles, même pour les systèmes relativement petits. La théorie SED a tenté de contourner ces défis computationnels en développant une méthodologie qui permet d'évaluer la performance du système sur un ensemble de configurations

possibles. Ceci est obtenu à partir de l'observation de son comportement et de la performance résultante à une seule configuration. Les observations requises peuvent être obtenues par simulation et, dans de nombreux cas, elles peuvent être collectées à partir d'une seule réalisation, ou échantillon de chemin, du comportement observé. En revanche, les méthodes considérées peuvent également être appliquées sur le système réel, et donc, elles deviennent un outil pour l'optimisation en temps réel d'adaptation et d'apprentissage.

De façon générale, les méthodes susmentionnées définissent une approche basée sur la « sensibilité » à la modélisation, à l'analyse et au contrôle des performances du SED [1]. Historiquement, l'analyse de sensibilité des SED est née au début des années 1980 dans un effort visant à aborder l'analyse de performance et l'optimisation des systèmes de file d'attente par rapport à certains paramètres structurels. Les paramètres tels que les taux d'arrivée et de traitement [41]. Actuellement, la théorie moderne aborde les modèles stochastiques plus généraux de SED qui la rapprochent des efforts plus larges pour soutenir l'optimisation, l'approximation et l'apprentissage incrémentiels dans le contexte du contrôle optimal stochastique [42]. Dans [43] [44] [45] [46] certaines applications particulières de l'analyse de la sensibilité des SED pour l'optimisation des performances des systèmes de production, de télécommunication et d'informatique [47] [48].

Un autre développement intéressant de la théorie du SED fondée sur le temps est la théorie de l'algèbre (max, +) [49]. Dans ses applications pratiques, cette théorie aborde la dynamique temporisée des systèmes qui impliquent la synchronisation d'un certain nombre de processus concurrents sans conflit entre eux. Elle fournit évidemment des résultats structurels importants sur les facteurs qui déterminent le comportement de ces systèmes en termes de taux d'occurrence de divers événements critiques et les latences expérimentées parmi eux. Les applications de motivation de l'algèbre (max, +) peuvent être tracées dans la conception et le contrôle des réseaux de télécommunication, des données de la fabrication et des systèmes ferroviaires. Plus récemment, la théorie a trouvé une application pratique considérable dans le calcul d'horaires répétitifs pour optimiser le taux de production des cellules robotiques automatisées et des outils de grappes utilisés dans la fabrication des semi-conducteurs [50] [6].

Les deux méthodes basées sur la sensibilité et la théorie de l'algèbre (max, +), qui ont été discutées dans les paragraphes précédents, sont permises par la modélisation explicite et formelle de la structure et du comportement de SED, ainsi que par l'analyse et le contrôle de performance poursuivis. Cette capacité de modélisation intégrative soutenue par la théorie SED permet

également une analyse approfondie de l'impact des politiques de contrôle comportemental imposées sur la performance du système. Les politiques qui sont nécessaires pour toute instanciation SED sont particulières. Il s'agit d'un sujet plutôt nouveau dans la littérature sur le SED, et certains travaux récents dans ce sens peuvent être trouvés dans [51] [52] [53] [54].

Dans ce chapitre, nous avons présenté une définition générale du système ainsi que ses propriétés. Après nous nous sommes penchés sur les systèmes à événements discrets en les présentant et en montrant leur importance. Ensuite, nous avons présenté une vue générale sur les problèmes rencontrés en modélisation, contrôle et complexité computationnelle. D'après cette étude, nous remarquons que les SED ont une importance capitale dans l'industrie et que les chercheurs leur donnent une importance majeure dans le but de les perfectionner et améliorer les processus industriels d'une manière globale. Pour nous focaliser sur un point de recherche bien précis, nous suggérons de se baser sur une analyse d'état d'art.

Chapitre 2

L'Analyse du Méga-Data Des Articles Publiés Dans Le Domaine des Systèmes à Evénements Discrets

D'après le premier chapitre, nous déduisons que les systèmes à événements discrets présentent un axe de recherche évolutif et ils reçoivent beaucoup d'attention, ce qui a créé une littérature immense. Le choix d'un axe de recherche bien précis et qui est intéressant et d'actualité pour les nouveaux chercheurs est très difficile. Pour faire, une étude analytique de l'état d'art est indispensable. Pour cela nous avons effectué une analyse du méga-data des articles publiés dans les SED. Nous avons publié trois articles de classification de l'état de l'art [55] [56] [57]. Cette classification a été faite en fonction de la nature des outils d'analyse utilisés (les outils mathématiques [55] ; les langages [56] ; les outils graphiques [57]. Les lecteurs peuvent évaluer le niveau de maturité de chaque outil pour les axes de recherche (Modélisation/ Contrôle/ Surveillance/ Diagnostic/ Supervision/ Robustesse/ Simulation), structure du système (Centralisé/ Décentralisé) et type des données du système (Déterministe/ Stochastique).

2.1 Axes de recherche :

D'après notre étude de l'état de l'art, nous avons déduit que les chercheurs traitent dans leurs articles les points suivants :

2.1.1 Système à événements discrets déterministe :

Un système déterministe est un système qui suit des lois d'évolution non probabilistes. Une suite d'événements entrants u(t) du système, produisent toujours la même suite d'événements sortants y(t) et selon un ordre déterminé par l'ordre des événements entrants quoi qu'il se soit passé auparavant : à partir du moment où le système arrive dans un état donné, son évolution sera toujours identique.

2.1.2 Système à événements discrets stochastique :

Un système stochastique ou non déterministe est un système qui suit des lois d'évolution probabilistes. Une suite d'événements entrants u(t) du système, produisent une suite d'événements sortants y(t), mais pas toujours la même. C'est-à-dire qu'il y a au moins une de ses variables de sortie qui est aléatoire. En général, l'état d'un système dynamique stochastique définit un processus aléatoire, dont le comportement ne peut être décrit que de manière probabiliste. Dans un système stochastique, l'état au temps t est un vecteur aléatoire, et c'est seulement sa fonction de distribution de probabilité qui peut être évaluée.

2.1.3 Système à événements discrets centralisé :

L'approche centralisée est une approche où il existe un seul module d'analyse ou de décision qui collecte et analyse les informations globales issues du processus.

2.1.4 Système à événements discrets décentralisé :

L'approche décentralisée est une approche où il existe plusieurs structures élémentaires de prise de décision du même type. Chacune de ces structures recueille une partie de l'information globale. La décision globale est obtenue en effectuant un calcul fonctionnel sur toutes les décisions des structures élémentaires.

2.1.5 Contrôle des systèmes à événements discrets :

Le contrôle du SED peut être conçu s'il existe un modèle SED disponible. Un contrôleur effectue les tâches suivantes :

- Formulation et spécification des tâches de contrôle du système donné.
- Détermination des algorithmes de contrôle.
- Conception des moyens techniques nécessaires à la mise en œuvre du contrôle.
- Création et vérification des programmes de contrôle.
- Mise en œuvre, essais et entretien de la fonction du système de contrôle.

La commande automatique est basée sur les manifestations de l'information du système. Autrement dit, un système est décrit par des informations sur l'emplacement spatial des objets, le temps, les paramètres système, les propriétés, les caractéristiques, etc.

Le temps est important pour la dynamique des événements. Comme mentionné précédemment, l'évolution dans le temps des variables système est appelée le processus et dans le contexte de SDED, elle est appelée le Procédé discret.

2.1.6 Supervision des systèmes à événements discrets :

La supervision a pour rôle de contrôler et de surveiller le déroulement et le fonctionnement des opérations d'un procédé. Elle a un rôle décisionnel et opérationnel pour la reprise de la commande. La supervision agit sur les aspects de fonctionnement normal et anormal d'un procédé.

En fonctionnement normal : la supervision contrôle l'exécution d'une opération et réalise l'ordonnancement en temps réel, pour prendre les décisions .En présence d'un défaut : la supervision doit prendre les décisions correctives et assurer le retour du procédé vers un fonctionnement normal.

2.1.7 Surveillance des systèmes à événements discrets :

La surveillance des procédés industriels a pour rôle d'assurer la veille sur l'évolution du comportement du procédé, et de rassembler les informations nécessaires délivrées par les capteurs, afin de générer des alarmes dans le cas d'une défaillance. Elle recueille les données à partir du procédé et de la commande, pour reconstituer l'état réel du système commandé, et faire des déductions nécessaires pour la génération de toutes les informations indispensables, permettant de remettre le procédé en fonctionnement normal. La surveillance est limitée à la collecte et l'inférence des informations, car elle n'agit ni sur le procédé ni sur le système de commande. Une aide à la décision est donc très utile d'être ajoutée à la surveillance à travers le diagnostic.

2.1.8 Robustesse des systèmes à événements discrets :

La robustesse d'un système peut être définie comme sa capacité à conserver les propriétés spécifiées en présence de variations ou d'incertitudes prévues ou imprévues. Pour chaque système et chaque niveau dans le cas d'une conduite hiérarchisée, la robustesse peut être de deux types différents :

- La robustesse interne concerne les changements de valeur de paramètres du modèle du procédé.
- La robustesse externe touche quant à elle les variations, intentionnelles ou non, acceptables à l'entrée du système.

2.1.9 Simulation des systèmes à événements discrets :

La simulation “par événements” sur ordinateur s’agit de reproduire sur un ordinateur, éventuellement avec l’aide de langages de programmation spécialisés pour la simulation, le déroulement de l’“histoire” du système. Le temps courant est égrené et les événements à venir sont stockés dans une pile jusqu’à ce que les conditions soient réunies pour qu’ils “se produisent” et sortent de la pile. Toutes sortes de compteurs et d’outils statistiques peuvent être « branchés » sur une telle simulation. Ce genre d’approche n’a priori de limites que par la complexité du programme à écrire et le temps d’exécution. Cependant, c’est une méthode “aveugle” ou “boîte noire” en ce sens qu’il n’y a pas de compréhension analytique de la relation entre les entrées (les choix faits a priori) et les sorties (les résultats observés), sauf à dépouiller des tonnes de listage permettant de suivre le déroulement des événements pas à pas, ce qui est rarement praticable.

2.1.10 Modélisation des systèmes à événements discrets :

La modélisation d’un système physique est une description de sa structure et une représentation comportementale ou fonctionnelle de chacun de ses composants. Une représentation comportementale est constituée de relations entre diverses variables du système, appelée classiquement relations de causes à effets. Une représentation fonctionnelle est plus abstraite puisqu’elle ne s’adresse qu’aux objectifs présumés que le système physique doit remplir.

2.1.11 Diagnostic des systèmes à événements discrets :

Le diagnostic est une fonction qui établit les liens de causalité, entre les symptômes observés, et les défauts qui sont survenus, leurs causes et leurs conséquences. Il s’intègre plus généralement dans le cadre de la supervision et la surveillance, et considéré comme un système d’aide à la décision. Cette fonction est généralement divisée en trois sous fonctions :

- La localisation : qui détermine le sous-système qui est à l’origine du défaut.
- L’identification : qui caractérise les causes d’une défaillance.
- L’explication : qui élabore les conclusions du diagnostic.

2.2 Représentation des données collectées

Nous avons collecté plus de 2000 articles scientifiques publiés dans des journaux indexés afin de définir la tendance de recherche scientifique en relation avec le domaine des SED. Pour simplifier la représentation de nos résultats, nous avons tracé les allures des courbes de nombre d'articles de chaque structure en fonction des années de publication. Les allures des courbes tracées peuvent nous permettre d'interpréter l'état actuel du point traité. Les résultats de notre étude sont présentés dans les figures suivantes :

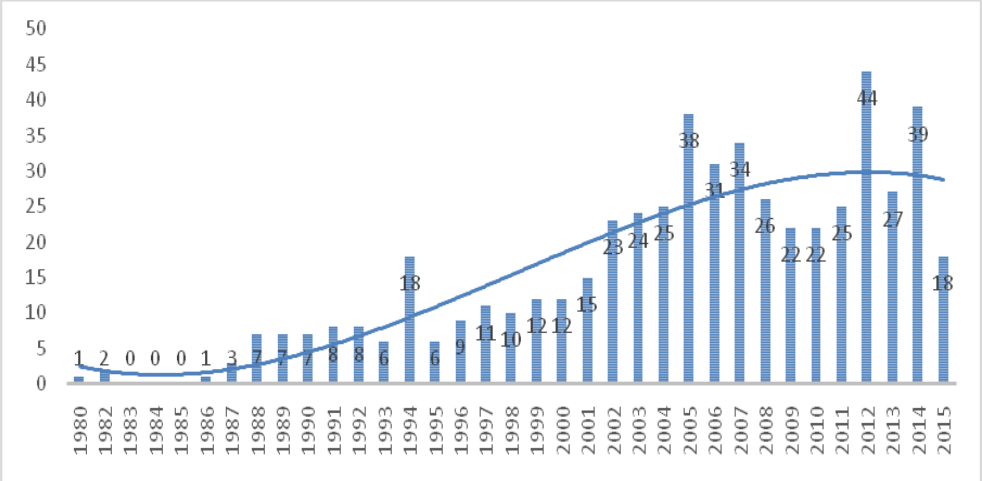


Figure 7 : La courbe d'évolution des articles publiés dans les SED déterministes

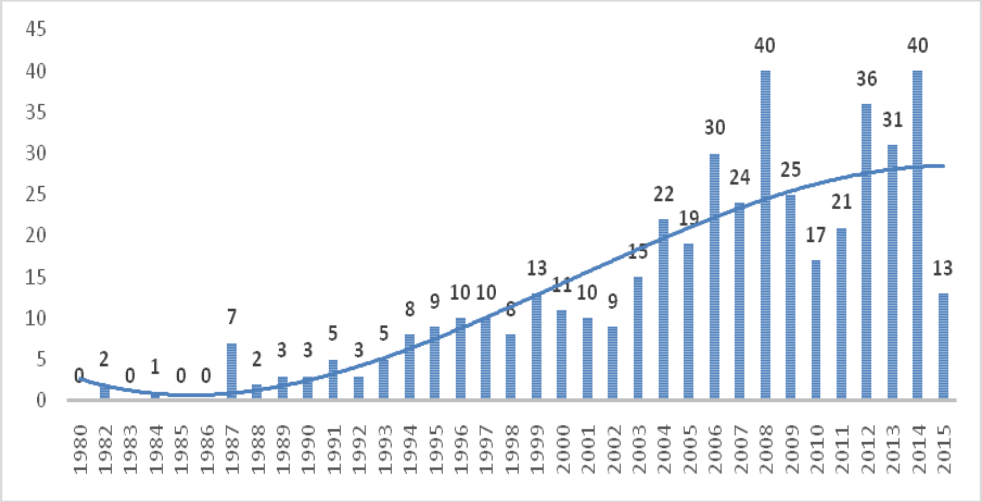


Figure 8 : La courbe d'évolution des articles publiés dans les SED stochastiques

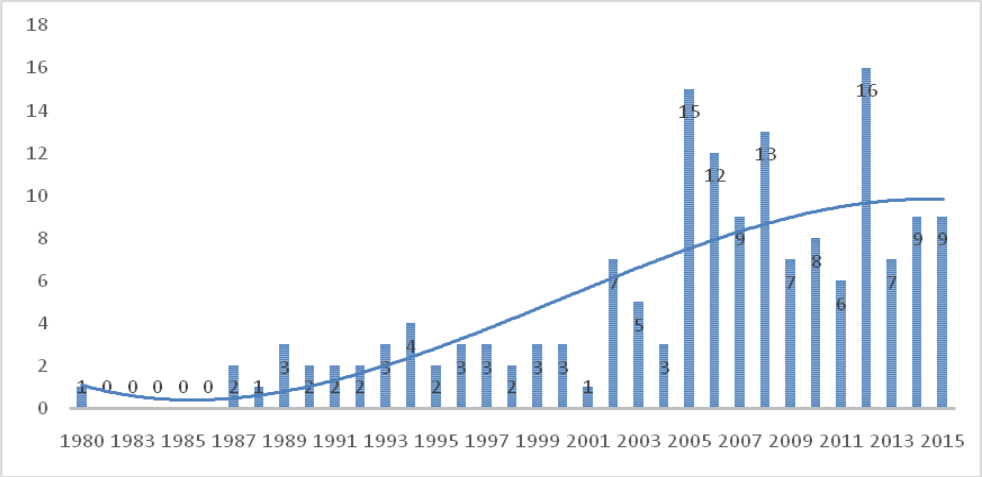


Figure 9 : La courbe d'évolution des articles publiés dans les SED centralisés

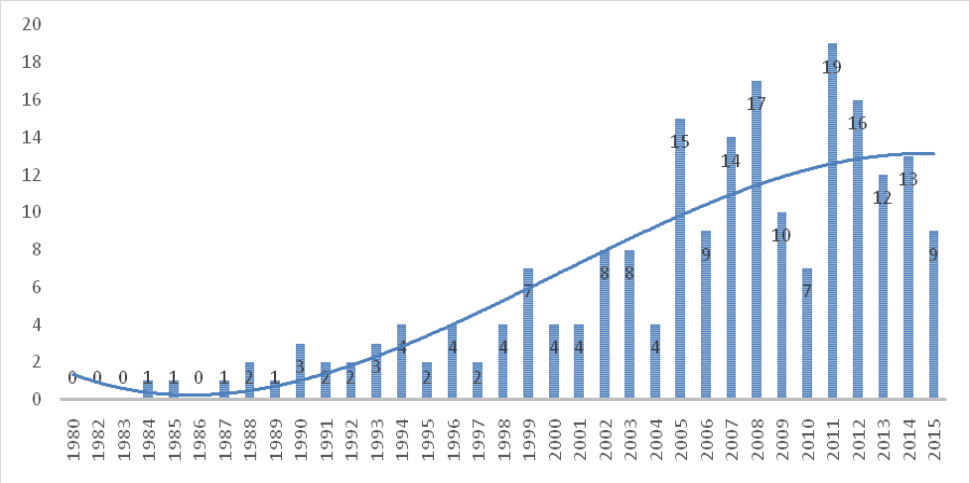


Figure 10 : La courbe d'évolution des articles publiés dans les SED décentralisés

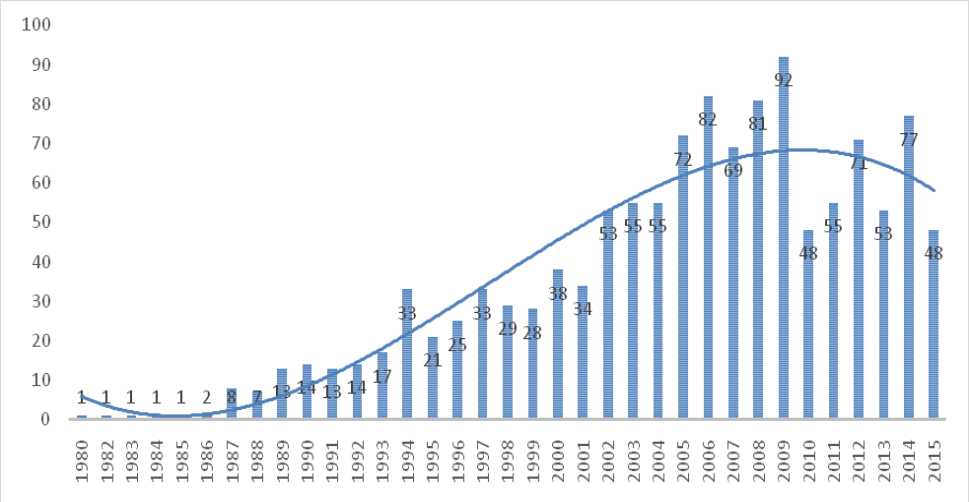


Figure 11 : La courbe d'évolution des articles publiés dans le Contrôle des SED

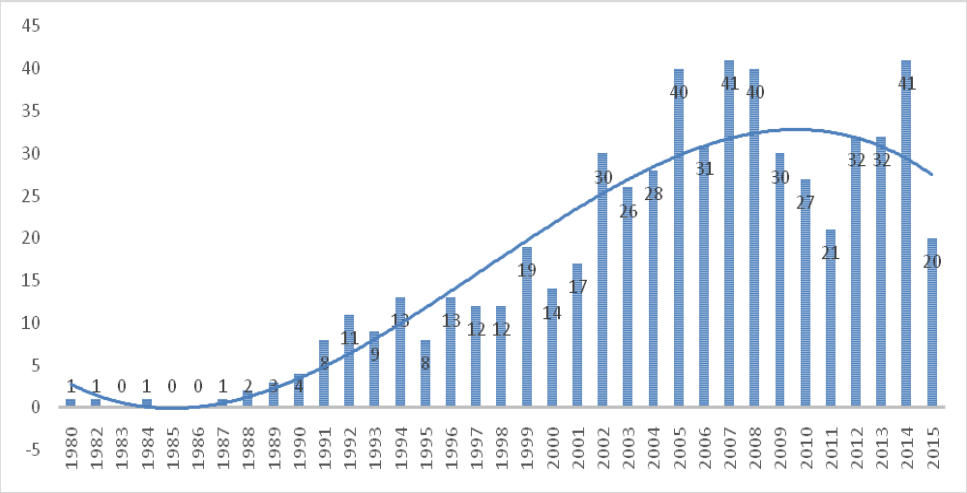


Figure 12 : La courbe d'évolution des articles publiés dans la Supervision des SED

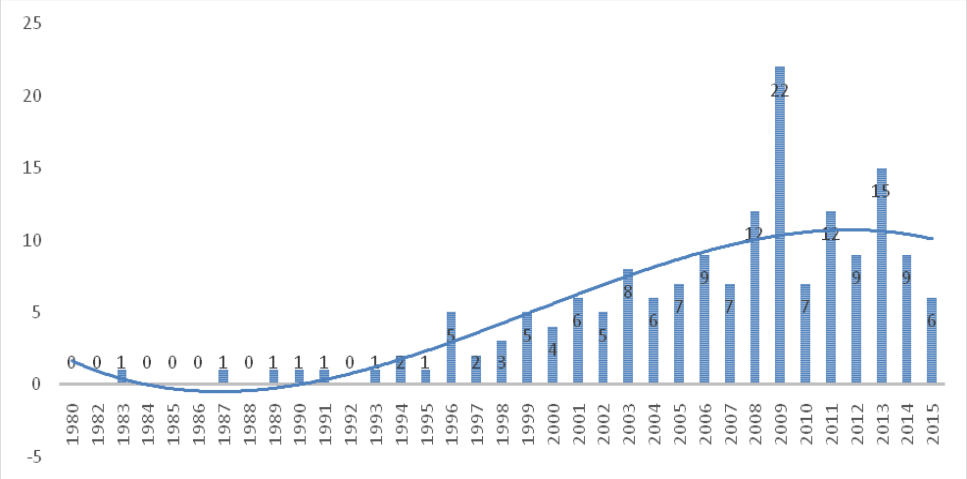


Figure 13 : La courbe d'évolution des articles publiés dans la Surveillance des SED

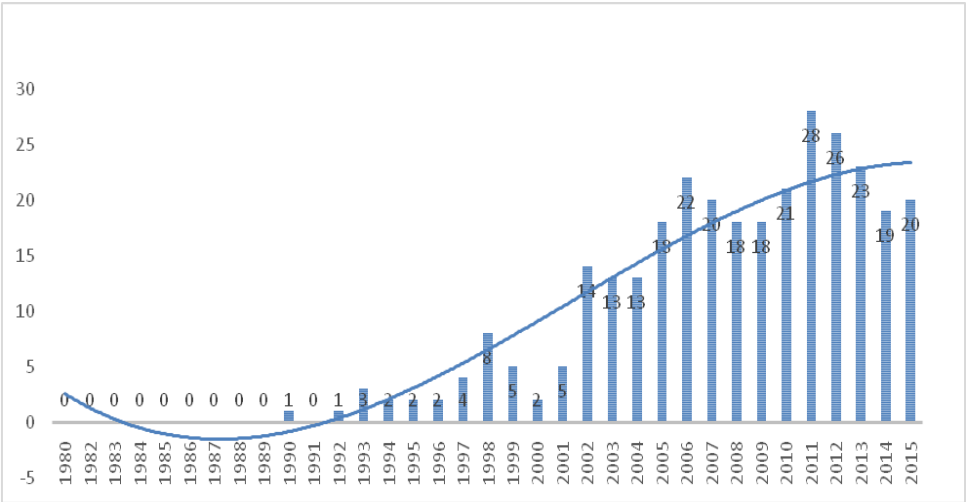


Figure 14 : La courbe d'évolution des articles publiés dans le Diagnostic des SED

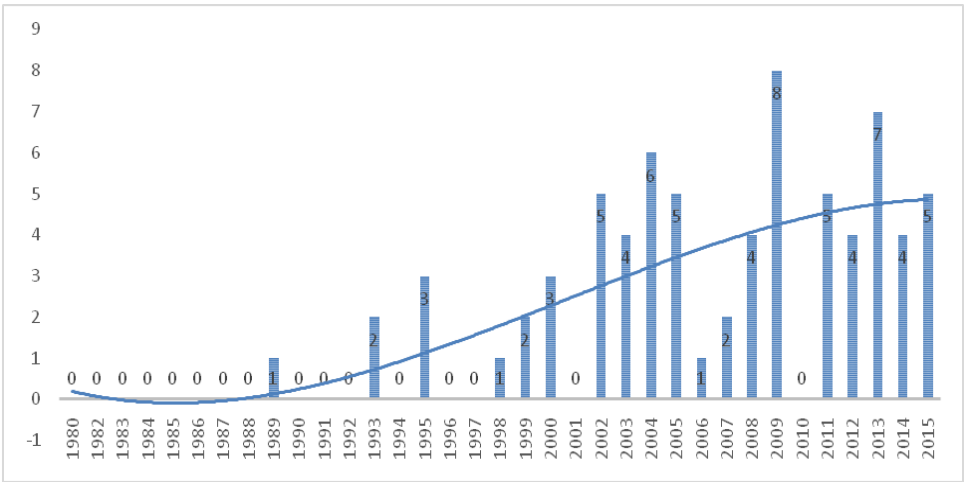


Figure 15 : La courbe d'évolution des articles publiés dans la Robustesse des SED

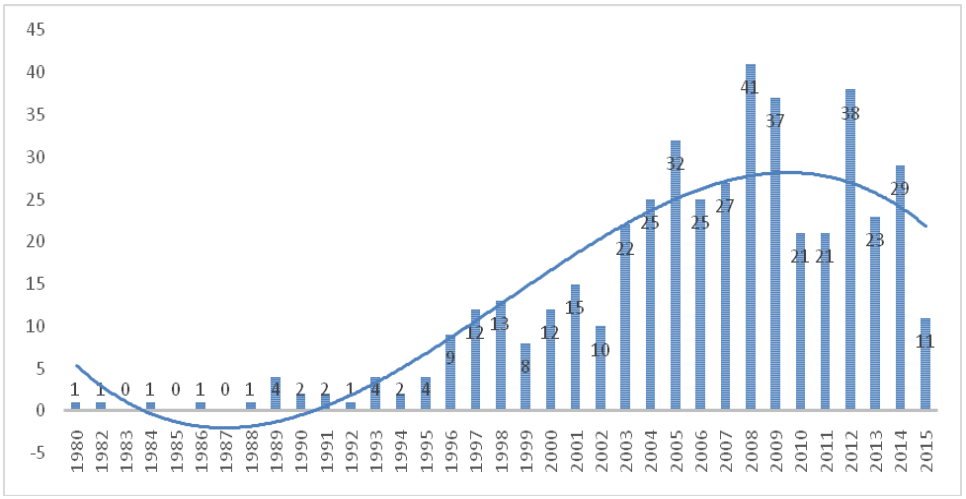


Figure 16 : La courbe d'évolution des articles publiés dans la Simulation des SED

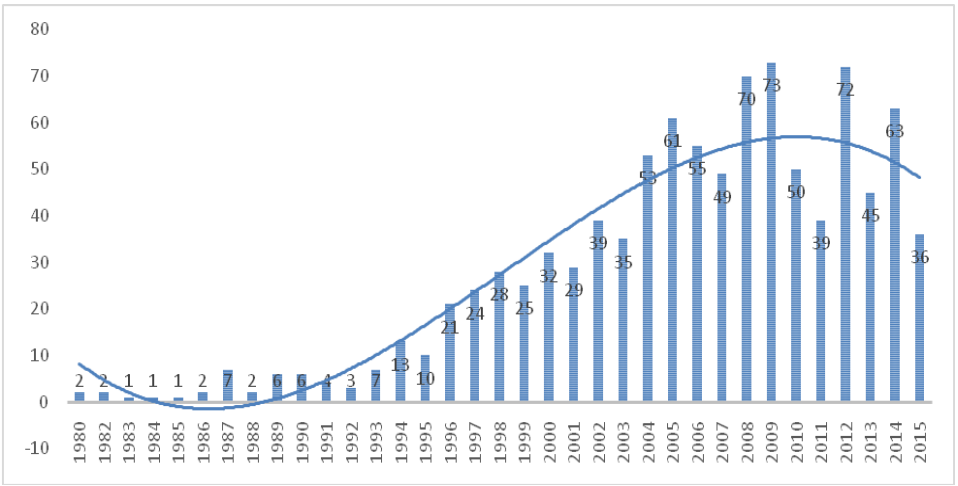


Figure 17 : La courbe d'évolution des articles publiés dans la modélisation des SED

2.3 Analyse du méga-data des articles publiés dans le domaine des systèmes à événements discrets

La représentation des résultats par des courbes qui présente le nombre d'articles publiés par année suivant les différents axes de recherche nous a permis d'avoir une vue générale sur les SED. On remarque que :

- Du 1980 au début des années 1990 le nombre d'articles est très limité et parfois nul comme le cas pour le diagnostic et la robuste des SED. Ça, est dû au fait que le domaine est nouveau et il n'y a pas assez de chercheurs qui travaillent dessus, en plus les articles n'ont pas été informatisés et les publications étaient en format papier.
- Du début des années 1990 jusqu'à la fin des années 2000, les SED sont devenus très connus vu leur applicabilité dans les différents domaines industriels, ce qui a attiré l'attention des chercheurs qui ont commencé à s'en intéresser de plus en plus. En effet, nous remarquons que toutes les courbes sont en évolution continue.
- Pendant la dernière décennie nous remarquons une très grande différence entre les courbes pour cela on va les traiter en fonction des cas :

  - Dans les figures 7 et 8 qui présentent la nature des variables traitées dans le système (déterministe et stochastique), nous remarquons que la pente de la courbe de la figure 8 est un petit peu plus importante que la pente de la courbe de la figure 7. Mais si on calcule le moyen de nombre d'articles publiés ces derniers 6 ans, nous trouverons que le moyen de la courbe de la figure 7 est plus grand que le moyen de la courbe de la figure 8. Ce qui nous permet de dire que : le présent c'est pour les systèmes à événements discrets déterministes et le futur c'est pour les systèmes à événements discrets stochastiques.
  - Dans les figures 9 et 10 qui présentent la nature des structures du système (centralisé, décentralisé) nous remarquons qu'ils ont la même pente. Mais si on calcule le moyen de nombre d'articles publiés ces derniers 6 ans, nous trouverons que le moyen de la courbe de la figure 10 est un petit peu plus grand que le moyen de la courbe de la figure 9. Ce qui nous permet de dire que les deux axes de recherche ont le même degré d'importance pour le présent et le futur.
  - Dans les figures 11, 12, 13, 16 et 17 qui présentent les fonctions de système à événements discrets (contrôle, supervision, surveillance, simulation, modélisation),

nous remarquons que les courbes sont en baisse continue. Ce constat reflet le désintérêt de la communauté scientifique de ces axes de recherche, celui-ci peut être dû au fait que :

- Le domaine est saturé, en attendant des nouvelles technologies par exemple la technologie cellulaire, ou la technologie quantique.
- La limite des outils pour résoudre les problématiques dans ces axes met les chercheurs dans l'attente de dévouement des outils mathématique ou graphique existant ou l'élaboration des nouveaux outils.
- L'industrie a besoin de résoudre d'autres problématiques qui sont urgentes dans d'autres axes de recherche.

– Dans les figures 14 et 15 qui présentent les fonctions de système à événements discrets (diagnostic et robustesse), nous remarquons que les courbes sont en croissance continue, surtout la courbe de diagnostic des SED. On remarque que la pente d'évolution est très importante que celle de robustesse, ainsi que la moyenne de nombre d'articles par année pour le diagnostic est très grand que celle pour la robustesse. Ce qui nous permet de dire que les chercheurs scientifiques se focalisent de plus en plus sur le diagnostic des systèmes à événements discrets.

Conclusion

Dans cette partie, nous avons présenté les différents types du système, après nous avons zoomé sur les systèmes à événements discrets. Dans le dernier chapitre, nous avons présenté les SED ainsi que ces domaines d'applications.

Ceci illustre l'importance capitale de ce type de système. Après nous avons fait un tour général sur les problèmes rencontrés dans ce domaine. Ce tour nous a montré la nécessité de se focaliser sur les SED, mais sur quoi exactement ?

Le troisième chapitre répond à la question avec une méthode scientifique qui est approuvée par trois articles publiés dans des journaux indexés. D'après ces derniers, nous avons décidé de nous focaliser sur le diagnostic des systèmes à événements discret qui présente la tendance de la recherche scientifique et le vif de notre projet de recherche.

Partie B

Diagnostic des Systèmes à Evénements Discrets

Introduction

La dynamique des systèmes à événements discrets est générée par l'apparition des événements qui se produisent instantanément. En absence d'un événement, l'état du système reste inchangé. L'évolution du temps entre les occurrences n'entraîne aucun effet détectable sur le système. Néanmoins, le temps est important pour de nombreux systèmes et la spécification temporelle critique peut-être prise en considération. Ces systèmes se trouvent dans plusieurs domaines d'application tels que les systèmes de production, les ordinateurs, les réseaux de communication, l'industrie aérospatiale, l'automobile, le transport, etc.

Une étude a été réalisée sur l'analyse de la méga base des données des systèmes d'événements discrets publiés dans les travaux suivants [57] [55] [56]. Cette étude est basée sur plus de 2000 articles, indiquant que la communauté scientifique s'intéresse de plus en plus au diagnostic des systèmes à événements discrets.

Le diagnostic est une fonction qui établit les liens de causalité entre les symptômes observés, les défauts qui se sont produits et leurs conséquences. Il est considéré comme un système d'aide à la décision. Cette fonction est généralement divisée en trois sous-fonctions : l'isolation qui sert à déterminer le sous-système qui cause le défaut, l'identification qui sert à caractériser les causes de l'échec, et l'explication qui sert à développer les résultats du diagnostic. Parmi les travaux qui traitent ce sujet, nous mentionnons les références de nos articles publiés et qui étaient pertinents : [58] [59] [60] [61] [62] [63] [64] [65] [66] [67].

Dans cette partie, nous allons présenter une étude générale sur les méthodes du diagnostic afin d'avoir une vue globale sur les différentes études menées dans le diagnostic des SED, ainsi de savoir les avantages et les inconvénients des différents diagnostiqueurs. Après, nous allons déployer ces connaissances afin d'élaborer un diagnostiqueur pour les systèmes à événements discrets.

Chapitre 1

Défauts et Diagnostic

1.1 Défauts

Les systèmes à événements discrets, comme la dynamique des véhicules, les moteurs aéronautiques, les procédés chimiques, le réseau et les machines électriques, les systèmes de fabrication, les systèmes de conversion d'énergie éolienne et les équipements électroniques industriels, sont des systèmes critiques pour la sécurité. La fiabilité et la sécurité des systèmes industriels soumis à des anomalies potentielles des procédés et des défauts de composants sont de plus en plus demandées. En conséquence, il est primordial de détecter et d'identifier tout type d'anomalies et de défauts potentiels le plus tôt possible et de mettre en œuvre un fonctionnement tolérant aux pannes pour minimiser la dégradation des performances et éviter des situations dangereuses.

1.1.1 Définition

Les défauts sont des dysfonctionnements de divers éléments des systèmes techniques. Un défaut peut être toléré ou considéré comme critique. Les cas extrêmes de fautes, appelées échecs, sont des pannes catastrophiques. Un défaut provoque un écartement non souhaité d'un système ou d'un de ses composants par rapport à son comportement normal ou prévu.

Les systèmes à événements discrets dont nous nous occupons vont de systèmes de production complexes (usines chimiques, raffineries de pétrole, centrales électriques) à d'importants équipements de transport (avions, navires) jusqu'aux machines grand public (automobiles, chauffages domestiques, etc.). Les défauts peuvent affecter différentes parties du système technique principal (moteurs, pompes, réservoirs de stockage, pipelines) ou des dispositifs d'interfaçage du système technique principal avec des ordinateurs permettant le contrôle, la surveillance et l'information de l'opérateur. Ces derniers comprennent des capteurs (dispositifs de mesure) et des actionneurs (dispositifs agissant sur le procédé, tels que des vannes).

1.1.2 Classification des défauts

Les genres de défauts qui peuvent se produire dans les systèmes peuvent être classifiés de la façon suivante [68] [67] :

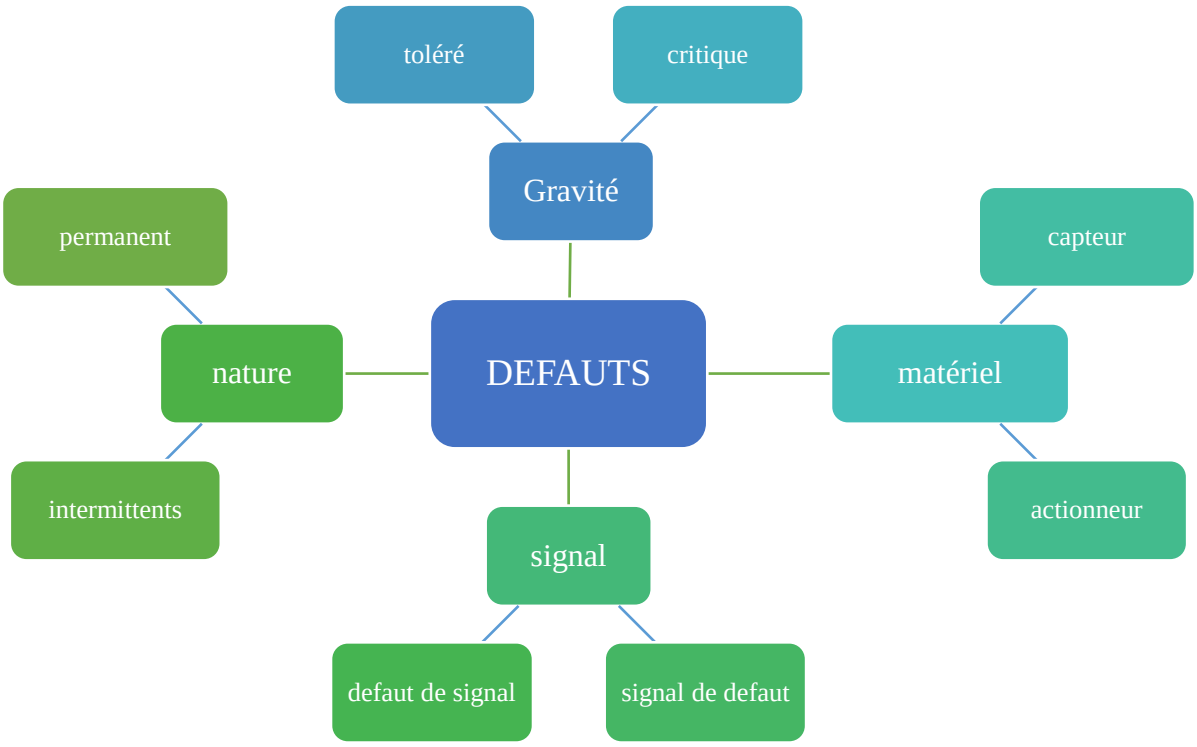


Figure 18 : Classification des défauts.

1.1.2.1 Défauts des capteurs et actionneurs

Typiquement, les actionneurs et les capteurs ont un mécanisme électrique, électromagnétique ou mécanique. En raison d'un défaut dans ces mécanismes, les capteurs et les actionneurs peuvent être verrouillés dans une position particulière. Ainsi, ces défauts sont caractérisés par un blocage de la sortie du capteur au niveau 1 ou 0, ou par un actionneur qui reste verrouillé, actif ou inactif, malgré les commandes envoyées par le système. Dans le système de remplissage représenté à la Figure 19, la soupape peut être soumise à des défauts de blocage au niveau 0 correspondant à l'état fermé de t1. Lorsque ce défaut se produit, le remplissage ne sera pas exécuté malgré que le système de contrôle a ordonné le début de remplissage qui ne sera pas exécuté même si la vanne est remplacée. Il convient de noter qu'il existe d'autres défauts que l'actionneur peut subir. Par exemple, dans le système de transfert. Sur la figure 20, le moteur

d'actionnement M peut être accéléré ou ralenti. Ce défaut peut être détecté par les aspects temporels du système de transfert.

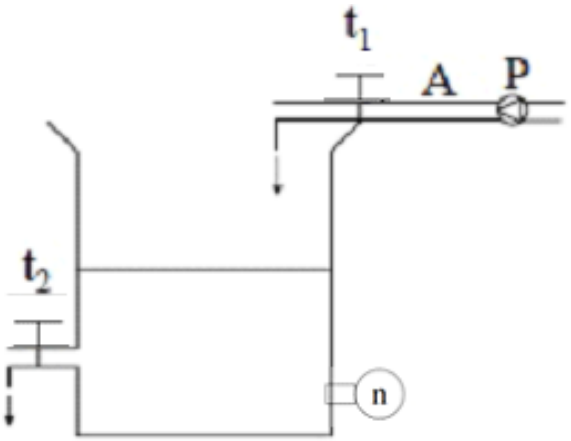


Figure 19 : Système de remplissage

1.1.2.2 Défauts des composants constituant la structure principale du système

Le système et ses composants peuvent également être sujets à des défauts. Certaines anomalies telles que la panne de l'équipement ou la perturbation de la puissance influence sur le fonctionnement de l'ensemble du système.

Dans le système de rangement représenté à la Figure 19, la fuite du réservoir est un exemple de ce type de défaut qui provoque une diminution du niveau du liquide et affecte la performance du ferry du système. Le blocage du tuyau au point ‘A’ représente un autre exemple de ce type de défaut.

1.1.2.3 Défauts permanents et intermittents

Les défauts permanents sont définis comme étant un dysfonctionnement d'un composant qui doit être changé ou réparé. Ainsi, nous disons qu'un défaut est permanent si la récupération ne se produit qu'après la réparation ou le remplacement du composant défectueux. Les défauts intermittents peuvent, à leur tour, permettre un retour du processus, du composant ou de l'actionneur dans son fonctionnement dynamique. Donc, on dit que le défaut est intermittent si la récupération se produit spontanément. Par exemple, dans la Figure 19, une canalisation bloquée peut être débloquée par une pression interne. Il est important de distinguer ces deux types de défauts. Le défaut intermittent provoque l'oscillation d'un système entre deux états : normal et défectueux (Non-offensant). La défaillance permanente peut être associée à des événements de récupération du système, et le commutateur ne peut pas passer spontanément de l'état de fonctionnement à l'état normal.

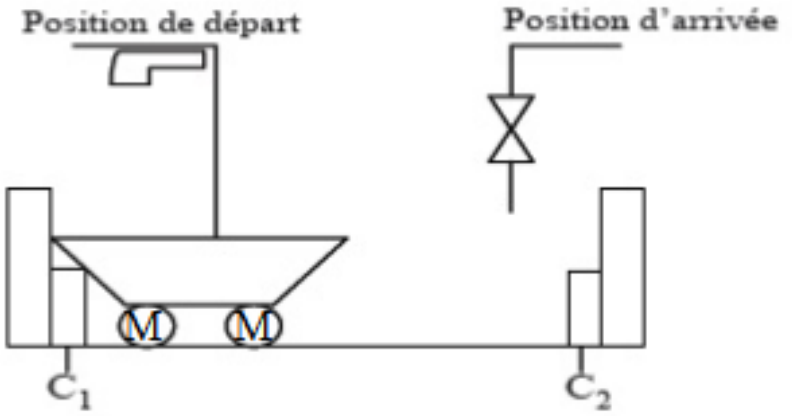


Figure 20 : Système automatisé de transport de matériels

Dans notre travail, nous supposerons que le système de contrôle est exempt de défauts, de sorte que l'origine de défaut de tout système provient des composants de processus qui peuvent être des capteurs ou des actionneurs. Dans les systèmes représentés sur les figures 19 et 20, nous donnons quelques exemples de défauts interruptibles : le verrouillage d'une vanne à l'état fermé, le blocage de la canalisation et le blocage du convoyeur à palettes, dans ce dernier cas le blocage pourrait être dû à un défaut de l'actionneur M ou au convoyeur lui-même, ces défauts interruptibles peuvent être permanents et intermittents.

L'objectif de la détection des défauts est de déterminer et de signaler s'il y a un défaut n'importe où dans le système, le diagnostic est destiné à fournir des informations plus précises sur le défaut, son identification consiste à déterminer, estimer la taille de la défaillance et dans certains cas le temps d'apparition de défaut.

1.2 Diagnostic

La détection des défauts nécessite normalement un modèle du comportement nominal du système, tandis que l'isolation et l'identification des défauts nécessitent également un modèle de comportement du système défectueux sous les défauts considérés.

1.2.1 Définition

Le diagnostic des défauts consiste à réaliser les trois tâches suivantes : La détection, l'isolement et l'identification des défauts.

- La détection des défauts est une fonctionnalité précisant l'état normal ou anormal du système.
- L'isolement des défauts vise à détecter les composants du système qui causent le problème.
- L'identification des défauts consiste à identifier la nature spécifique du défaut (sa taille, sa criticité, son importance, etc.).

Les méthodes de diagnostic des pannes pour les SED ont été développées dans les années 1980 et au début des années 90 par [69] [70] [71] [72]. Ces approches traitent l'estimation de l'état - état actuel ou état initial et du contrôle de supervision. Cependant, ils ne sont pas directement concernés par la partition des défauts et l'identification des types de défauts basés sur des modèles de comportement défectueux. [73] a proposé une approche basée sur l'état pour la diagnosticabilité du SED. Cette approche fournit des algorithmes pour calculer une séquence de commandes de test pour diagnostiquer les défaillances. Les méthodes d'optimisation des capteurs pour le diagnostic ont été étudiées à la même époque par [74]. Plusieurs méthodes de détection des défauts basées sur les modèles de SED de Réseaux de Petri ont également été développées à partir des années 80 par [75] [76] et [77].

1.2.2 Classification des méthodes de diagnostic

Les méthodes de diagnostic des SED peuvent être classifiées en :

1.2.2.1 La compilation des erreurs

Le diagnostic des défauts peut être réalisé en utilisant un diagnostic compilé hors ligne ou calculé en ligne. Dans le cas du hors-ligne, le système à diagnostiquer est considéré comme étant dans un banc d'essai, c'est-à-dire non dans un état de fonctionnement normal. Le diagnostic souhaité est compilé sur la base d'un test d'un ensemble d'entrée (séquences de commandes) et d'observation des sorties résultantes. La compilation hors ligne d'un diagnostic fournit une caractérisation complète du problème. Une solution en ligne efficace en termes de temps de réponse au diagnostic. Ceci est dû au fait que dans chaque état un diagnostic peut être fourni et que sa mise à jour nécessite seulement le déclenchement d'une transition du diagnostiqueur selon les conséquences des événements du système observé. Cependant, comme mentionné dans la section précédente, la compilation hors ligne d'un diagnostic nécessite la disponibilité d'un modèle fautif exhaustif et correct (ce qui est irréaliste dans les systèmes complexes réels) et représente un défi de calcul. Une autre approche consiste à la détection en ligne et l'isolement / l'identification

de l'ensemble des défauts qui peuvent se produire après chaque nouvelle observation acquise du système pendant son fonctionnement.

Des calculs complexes peuvent donc être nécessaires pour réaliser le diagnostic en ligne. Une telle approche est plus exigeante du point de vue du calcul en termes de temps de réponse au diagnostic, mais elle produit un gain substantiel en espace mémoire car il n'est pas nécessaire de stocker le diagnostiqueur complet.

1.2.2.2 Formalisme de modélisation

De nombreux formalismes de modélisation ont été utilisés pour construire des diagnostics, y compris : Automates [78] et leurs extensions chronologiques et probabilistes. Les réseaux de Petri [79] [80] [81] [82] [83] [84] [85] [86], et les diagrammes d'état et les machines d'état hiérarchiques [87]. Une étude comparative a été faite dans ce sens par notre équipe de recherche [88].

Tableau 1 : Etude comparative des différents outils de modélisation

		Avantages	**Inconvénients**
Outils de modélisation	**Automates à états**	-Description précise du comportement des systèmes manufacturiers d'un point de vue matériel -Manipulation facile grâce à la théorie des langages et les outils de composition et de projection	Risque d'explosion combinatoire
	RDP	La gestion des Procédés Concurrentiels nécessitent d'établir une distribution des éléments.	Risque d'explosion combinatoire
	Expressions Logiques/ algorithmes	Modèle abstrait permettant une exploitation et une compréhension souvent plus aisées.	Interprétation difficile Des résultats d'un système complexe

1.2.2.3 La représentation des défauts

Nous discutons d'autres diagnostics de pannes basés sur le comportement nominal et défectueux du système.

1.2.2.3.1 Diagnostic à l'aide de modèles incluant un comportement défectueux

Comme nous l'avons vu précédemment, le diagnostic de panne (y compris la détection et l'isolement / l'identification) nécessite la connaissance du comportement défectueux du système. Cette approche peut fournir de bons résultats de diagnostic dans le cas de défauts prévisibles. Cependant, il n'est pas toujours réaliste de prévoir exhaustivement tous les défauts et, par conséquent, seuls les défauts qui sont explicitement pris en compte dans le modèle de système peuvent être détectés et identifiés. Les modèles défectueux sont basés sur différents types de représentation de défauts, tels que l'exécution d'un événement (diagnostic événementiel), l'atteinte d'un état défectueux (diagnostic basé sur l'état), l'exécution d'un modèle de supervision ou la vérification des contraintes temporelles partielles.

Le diagnostic basé sur l'événement, comme l'approche de [89] examinée dans la section 2, décide si un défaut s'est produit et son type basé uniquement sur l'observation des séquences d'événements. Les méthodes de diagnostic basées sur des événements peuvent être utilisées pour diagnostiquer des défauts intermittents, car ils considèrent un défaut comme l'occurrence d'un événement non observable [90]. Ces approches nécessitent l'initialisation à la fois du diagnostic et du modèle en même temps parce que le diagnostiqueur prend sa décision sur la base des séquences observées des événements. Cette initialisation n'est pas toujours facile à réaliser dans les systèmes réels, ce qui peut nécessiter l'introduction d'autres événements non observables au stade de la modélisation.

Le diagnostic d'état est basé sur le partitionnement de l'espace du système en fonction de sa défaillance. Les approches proposées par [91] et [73] sont liées à des systèmes d’entrées et de sorties binaires. Chaque état est étiqueté avec le vecteur binaire de ses sorties qui sont associées à un diagnostiqueur utilisant la séquence de ces vecteurs pour détecter et isoler les défaillances. Les méthodes de diagnostic basées sur l'état sont bien adaptées pour déterminer les défauts permanents car ils considèrent la faute comme le fait d'atteindre un état défectueux. Cependant, ces méthodes ne conviennent généralement pas pour diagnostiquer des défauts intermittents. Puisqu'un diagnostic basé sur l'état peut déterminer l'apparition d'un mode de défaillance en lien avec la sortie générée, aucune information sur l'état ou l'état de défaillance du système n'est requise avant la mise en pratique d’un diagnostiqueur. Par conséquent, l'avantage du diagnostic basé sur l'état résulte dans l’indépendance d'initialiser le système et le diagnostic simultanément, l’analyse peut être

initialisée à tout moment pendant que le système est en fonctionnement. Dans [92], une approche combinant les avantages du diagnostic étatique et événementiel est proposée.

Un défaut peut également être représenté comme une exécution d'un motif de supervision donné, qui est une propriété temporelle liée à l'apparition d'un ensemble de trajectoires / événements devant être diagnostiqués [93]. La notion de schémas de supervision est assez générale pour couvrir une classe importante d'objectifs de diagnostic, y compris la détection de défauts permanents, mais aussi des défauts transitoires, des défauts multiples, des fautes répétées, ainsi que des séquences d'événements assez complexes.

Les modèles de supervision sont très utiles pour généraliser les propriétés de diagnostic et clarifier la séparation entre les objectifs de diagnostic et les spécifications du système. Les résultats du diagnostic peuvent donc être facilement réutilisés pour des nouveaux problèmes de diagnostic qui sont similaires en vue de leur nature générique. C'est un avantage majeur par rapport aux autres approches de représentation de défaut dont le résultat est assez difficile à réutiliser car elles sont généralement associées à de nombreuses notions différentes de diagnosticabilité et elles utilisent des algorithmes spécifiques pour la construction du diagnostiqueur et pour la vérification de diagnostic.

Tableau 2 : Etude comparative des différents modèles de défauts

		Pannes permanentes	**Pannes intermittentes**	**initialisation**	**Avantages**	**Inconvénients**
Modèles de défauts	**A base d'événements**	Détection	Détection	Problème de détection des défauts à l'initialisation	- Détection des Pannes intermittentes	-Défauts à l'initialisation
	A base d'états	Détection	Problème de détection	détection des défauts à l'initialisation	-Défauts à l'initialisation	-Problème de détection des Pannes intermittentes
	Mixte	Détection	Détection	détection des défauts à l'initialisation	-Détection des pannes intermittentes et des défauts à l'initialisation	

1.2.2.3.2 Diagnostic utilisant des modèles sans défaut

Le diagnostic avec des modèles sans défaut repose sur la comparaison de la sortie du système avec la sortie nominale du modèle. Un défaut est détecté si un comportement observé du système ne peut pas être reproduit par son modèle. Cependant, l'isolement et l'identification des défauts peuvent ne pas être possibles dans ce cas, car le modèle n'inclut pas le comportement défectueux et, par conséquent, la diagnosticabilité d'un défaut donné n'est pas garantie.

L'approche de modélisation sans erreur proposée par [94] utilise des modèles d'état pour déterminer si le système génère des événements dans le bon ordre ou dans les délais indiqués. Un défaut est détecté lorsqu'il y a des réactions manquantes ou erronées dans le processus. Dans ce cas, les événements liés au modèle aident à isoler le défaut. Dans [95], les connaissances spécialisées sont associées à des modèles d'état pour identifier les failles liées à des événements manquants ou inattendus, et un suivi progressif est utilisé pour réduire l'ensemble des candidats à la défaillance après l'apparition de nouveaux événements observables.

Le papier [96] propose une autre approche pratique de modélisation sans faille pour le diagnostic de pannes des systèmes de fabrication. Cette approche commence par identifier le modèle sans défaut du système. Un défaut est détecté pour tout comportement du système qui ne fait pas partie du modèle identifié. La localisation des défauts s'inspire des techniques résiduelles couramment utilisées dans les systèmes continus. Il est basé sur la comparaison des séquences observées et attendues et le calcul d'un petit ensemble de candidats inattendus et ratés. Cet ensemble est encore réduit par l'application d'un algorithme de réduction heuristique candidat-set qui fournit une bonne estimation du défaut qui peut se produire. Malgré sa simplicité et sa praticité, cette approche peut présenter un taux élevé de fausses alertes et ne fournit aucune garantie quant à la diagnosticabilité de certains défauts. Une extension aux modèles chronométrés a été proposée dans [97].

1.2.2.3.3 Classification des méthodes de diagnostic par rapport à la structure décisionnelle

Trois grandes structures de traitement sont utilisées pour calculer la décision de diagnostic de panne : Centralisée, décentralisée [98] [99] [100] [101] [102] et distribuée [103] [104] [105] [106]. Notez que la distinction entre les structures décentralisées et distribuées est parfois floue. D'une manière générale, les approches décentralisées disposent d'un ensemble de diagnostics,

chacun avec des capacités d'observation différentes, mais tout en tenant compte du modèle du système global dans leur inférence basée sur des modèles. Dans les approches distribuées, les diagnostics individuels utilisent uniquement des modèles de systèmes partiels (locaux) par opposition au modèle de système global.

Diagnostic centralisé Dans la structure centralisée, le diagnostic est calculé à l'aide d'un diagnostic global, qui est construit en utilisant le modèle global du système à diagnostiquer.

Le principal avantage des approches de diagnostic centralisées est leur précision de diagnostic et leur simplicité conceptuelle. Cependant, leur inconvénient principal est leur complexité computationnelle prohibitive, puisqu'ils nécessitent un modèle de centrale centralisée pour générer le diagnostic centralisé. Les modèles résultants peuvent devenir trop volumineux pour être stockés physiquement lorsqu'un système à grande échelle est considéré. Même si un diagnostic centralisé existe physiquement, il peut encore souffrir d'une faible robustesse et d'une faible maintenabilité.

Structure décentralisée avec diagnostic coordonné : Dans la structure décentralisée, le système est divisé en plusieurs sites. Chaque site connaît le modèle de système entier, et il fait des observations locales, et il utilise un diagnostiqueur local qui calcule une décision de diagnostic local basée sur son observation partielle de l'ensemble du système. Un coordonnateur fournit la décision finale de diagnostic en fonction des décisions de diagnostic local qui lui sont communiquées. Les diagnostics locaux et le coordinateur sont construits en utilisant un modèle global du système. Les diagnostics locaux peuvent ne pas communiquer directement les uns avec les autres, et seulement la communication limitée entre eux par l'intermédiaire du coordinateur est autorisée, d’une façon générale.

Le problème principal à résoudre dans les architectures décentralisées est de savoir comment les sites peuvent découvrir conjointement l'apparition d'un défaut, sachant que les informations disponibles peuvent être ambiguës, incomplètes, retardées et erronées. Le coordonnateur devrait donc disposer de capacités de mémoire et de traitement pour coordonner les échanges d'informations nécessaires entre les diagnostics locaux afin de résoudre les ambiguïtés des décisions locales. Toutefois, ces capacités devraient être contraintes, sinon, la structure centralisée pourrait être reproduite sur le site du coordinateur en lui communiquant toutes les observations, ce qui irait à l'encontre de l'objectif de la structure décentralisée.

L'objectif principal du diagnostic décentralisé est donc de concevoir un ensemble de protocoles, d'analyser leur compromis de «complexité-performance», de construire des diagnostics et de comparer leurs performances au cas centralisé, et de vérifier leur diagnostic. Par exemple, trois protocoles utilisant un coordinateur ayant des capacités de mémoire et de traitement variables mais limitées ont été proposés dans [100].

Le but de l'introduction des architectures de diagnostic décentralisées est d'effectuer la détection et l'isolement des défauts de manière à se rendre compte de la décentralisation de l'information dans des systèmes complexes (interconnectés) tout en préservant, si possible, la capacité diagnostique d'un diagnostic centralisé. Cependant, une architecture de diagnostic décentralisée exige toujours, en général, un modèle de système centralisé pour construire les diagnostics locaux, ce qui entraînera également des calculs complexes.

Diagnostic réparti : Les approches distribuées [62] [107] [108] [106] permettent d'établir le diagnostic à l'aide d'un ensemble de modèles locaux sans se référer à un modèle de système global. L'objectif est d'améliorer l'évolutivité et la robustesse des méthodologies de diagnostic. Chaque sous-système connaît seulement sa propre partie du modèle global et il accomplit son diagnostic local afin d'effectuer le diagnostic localement. Ce calcul de diagnostic est basé sur le modèle local et sur les informations qui lui sont directement communiquées par les autres diagnosticiens locaux via un protocole de communication. Les informations échangées entre les diagnostics locaux sont utilisées pour mettre à jour leurs propres informations et compense leur observation partielle.

Un protocole de communication doit être défini pour assurer la cohérence entre les diagnostics locaux en cas de conflits entre leurs décisions locales. Si les modèles locaux (sous-systèmes) n'interagissent pas de manière hiérarchique ou arborescente, le protocole de communication peut nécessiter un temps de calcul et un grand espace d'état. Le défi du diagnostic distribué est de savoir comment effectuer un diagnostic local équivalent, si possible, à celui global, en utilisant un protocole de communication évolutif (par rapport au nombre de modules composant) et sans avoir besoin d'utiliser un modèle global. Différents réglages et structures de modèles, modulaires hiérarchiques, ont été proposés pour le diagnostic distribué [109] [110] [111] [112] [113] [114]. L'objectif de cette diversité est de traiter les différentes questions liées aux types de synchronisation, aux retards et aux pertes de communication, à la préservation de l'information, à la structure du modèle et à la complexité. En conséquence des diverses structures de diagnostic décentralisées et distribuées qui ont été étudiées par les chercheurs, de nombreuses notions et

propriétés de diagnostic ont été définies et analysées. Il s'agit notamment du diagnostic local [115], du diagnostic indépendant [102] du D-diagnostics [59], codiagnostic [101], codiagnostic conditionnel [116], D- codiagnosabilité [117], et beaucoup d'autres.

La tâche de la prédiction de la faute est de vérifier l'apparition d'un défaut imminent avant son apparition, en fonction des chaînes d'événements observables [118] [119] [120] [121]. Cela aide à réagir en temps opportun à une panne imminente afin que des actions correctives puissent être initiées avant son apparition. Notez le contraste avec la tâche de diagnostic, qui nécessite la détection et la localisation d'un défaut après son apparition. La notion correspondante de prévisibilité a également été proposée et étudiée par [118] et [120]. Un système est prévisible si chaque trace défectueuse possède un préfixe non défectueux de telle sorte que toute trace indiscernable entraînera inévitablement la faute. De nombreuses contributions ont porté sur le développement d'algorithmes pour la vérification (polynomiale) de la prévisibilité et la construction de prédicateurs hors ligne, en ligne et décentralisés. Des algorithmes pour prédire les occurrences d'un modèle qui décrit des séquences d'événements ont également été proposés [119].

Dans la plupart des cas, le diagnostic de défaut de SED suppose que les capteurs fonctionnent correctement. Cependant, un mauvais fonctionnement du capteur peut empêcher les capteurs de signaler les événements. Cela signifie qu'un événement précédemment observable lié à un capteur peut devenir inobservable lorsque le capteur échoue. Un diagnostic robuste peut être utilisé pour obtenir un système diagnostique malgré les pannes des capteurs. Un diagnostic robuste trouve ses antécédents dans une synthèse de contrôleur robuste dans le cadre du contrôle de supervision de SED [122].

Une approche globale du diagnostic robuste a été proposée dans [123] et [124]. Cette approche déploie la redondance dans les bases de diagnostic, c'est-à-dire le sous-ensemble différent d'événements observables qui garantissent la diagnosticabilité, pour vérifier la diagnosticabilité et concevoir un diagnostic robuste malgré les pannes de capteurs. L'idée est d'exécuter un ensemble de diagnostic partiels en parallèle, chacun conçu pour une base de diagnostic particulière pour fonctionner correctement sous une certaine combinaison de pannes de capteurs et pour garantir qu'au moins un de ces diagnosticiens émettra la décision de diagnostic correcte concernant les défauts non observables . Compte tenu d'un ensemble de combinaisons possibles de pannes de capteurs, la définition et le test d'une diagnosticabilité robuste pour les systèmes centralisés et décentralisés sont également proposés. L'extension du diagnostic et de la diagnosticabilité robustes au cas de perte intermittente d'observations est considérée dans [125].

Dans ce cas, un dysfonctionnement du capteur ou une panne de communication est modélisé en dupliquant sa transition associée avec une transition d'événement non observable.

Le problème de diagnostic introduit par [126] [127] concerne le cas des systèmes avec des configurations multiples, tels que des systèmes de fabrication flexibles, et exige que le système soit décrit par un ensemble de modèles possibles dont chacun a sa propre spécification de non-échec, sur un ensemble d'événements communs. La formulation de diagnostic robuste fournit des conditions pour l'existence d'un seul diagnostic qui détecte les défauts dans n'importe quel modèle possible dans un nombre borné d'étapes. Des algorithmes sont proposés pour vérifier une diagnosticabilité robuste et synthétiser un diagnostic robuste. Une généralisation des deux précédentes notions de diagnostic robustes a été proposée par [128] pour tenir compte des incertitudes à la fois dans le modèle de système et dans l'ensemble d'événements observables.

Tableau 3 : Classification des méthodes de diagnostic par rapport à la structure décisionnelle

		Modèle du procédé	diagnostiqueur	Protocole de communication	Décision	Indécision	Complexité du protocole	Avantages	Inconvénients
Structure de la prise de décision	**Centralisée**	Modèle global	global	aucun	globale	aucune	Aucune	-Pas besoin d'un protocole de communication -Un seul modèle avec un seul diagnostiqueur	-Explosion Combinatoire -faible robustesse -faible maintenabilité -Peu tolérantes aux défaillances -peu évolutives
	Décentralisée Sans coordinateur	Modèle global	Local	aucun	Local	Risque important	Aucune	-Diminution de l'explosion combinatoire	-Risque de conflits Décisionnels -Architecture conditionnelle
	Décentralisée avec coordinateur	Modèle global	Local	coordinateur	Local	Faible risque	Faible	-Diminution de l'explosion et gestion des conflits par le coordinateur	-Nécessité d'un coordinateur
	Distribuée	Modèle local	Local	Entre modules	Local	aucune	importante	-Gestion des conflits directement	-Protocole de communication

Chapitre 2

Problématique et Approche de Résolution

2.1 Question de la recherche

Est-ce que nous pouvons élaborer un diagnostiqueur facile à manipuler, capable de donner une description précise du comportement des systèmes manufacturiers du point de vue matériel, sans risque d'explosion combinatoire, détecter tous les types des pannes, faciles à interpréter, qui n'a pas besoin d'un protocole de communication, ne nécessitant pas un coordinateur et sans risque de conflits décisionnels ?

2.2 Approche de résolution

Un SED est un système dynamique dans lequel l'état de l'espace est discret. Ses trajectoires sont composées avec des états constants par morceaux. Un tel système fonctionne en fonction de l'apparition d'événements physiques dans des intervalles de temps généralement irréguliers ou inconnus.

Il existe de nombreux domaines d'application dans le domaine du SED. Différents aspects du comportement d'un système peuvent être considérés, selon l'application. Par conséquent, divers outils pour la modélisation et l'analyse de SED ont été développés. L'évolution du SED se caractérise par l'apparition d'événements. Selon la façon dont les événements sont organisés dans le modèle, nous pouvons classer les modèles SED en deux catégories : modèles non-temporels et temporels.

Les modèles de SED non-temporels ne tiennent compte que de l'apparition d'événements dans leur ordre séquentiel. Ces modèles ignorent les temps d'occurrence des événements. Ces modèles sont utilisés pour l'étude des propriétés qualitatives du SED. Dans d'autres applications, les informations sur le temps sont essentielles et doivent être prises en compte par le modèle. Les modèles qui ont cette caractéristique sont des modèles chronométrés où le temps peut être

déterministe ou stochastiques. Les modèles où le temps est aléatoire et qui sont caractérisés par des distributions de probabilité associées aux dates des événements.

De façon générale, nous pouvons être intéressés par le calendrier des événements afin de répondre à des questions telles que : «Combien de temps le système dépense-t-il dans un état particulier?» Ou «Cette séquence d'événements peut-elle être complétée d'ici un certain délai ? "Plus généralement, le timing des événements est important dans l'évaluation de la performance d'un SED souvent mesurée par des quantités telles que le débit ou le temps de réponse.

Chaque système industriel a une fonction. Cette fonction est un cycle d'actions et d'événements classés suivant un ordre préétabli. Cet enchainement d'action et de réaction se répète à chaque fois que le système exécute sa fonction, quelles que soient ces types d'actions et de réaction (temporel ou non-temporel). Alors tous les systèmes industriels sont des systèmes cycliques. Ce cycle a une projection dans le temps : il commence à l'instant Ti et se termine à l'instant Tj, cette période est le temps de fonctionnement. Il n'est pas exact, c'est une marge de temps qui dépend de plusieurs variables : la cadence, vieillissement, l'environnement, la charge…

Si nous zoomons sur un événement dans ce cycle, on va trouver qu'elle s'exécute dans une marge de temps après l'instant Ti. En nous basant sur ces informations, nous pouvons dire que les cycles répétitifs caractérisent les comportements des SED à une durée bien définie et chaque événement se produit dans une durée prévue, y compris une marge de tolérance donnée par le fabricant. Si l'une de ces étapes dépasse la période déterminée, il peut être causé par le dysfonctionnement d'un des capteurs ou des actionneurs ou un accident du travail ... etc. Dans ce cas nous devons intervenir. Pour cette raison, nous avons élaboré des diagnostiqueurs afin de vérifier si l'activité satisfait les contraintes temporelles qui sont définies en fonction des intervalles de temps codés dans un ensemble appelé « TC ». Un intervalle de temps TIi = $[d_{min}i:d_{max}\ i]$ est associé à chaque action (Ai). Ai est l'action liée à la contrainte temporelle C. Les dates d_{min} i et d_{max} i se situent respectivement pour les dates de tir les plus anciennes et les plus récentes compatibles avec la contrainte C. Chaque élément $C \in TC$ correspond à une contrainte et il est défini par :

$$C = \{Ai, [d_{min}\ i:\ d_{max}i]$$

2.3 Exemple illustratif

Pour expliquer la problématique et notre proposition, nous allons utiliser un exemple, c'est celui d'un chariot qui se déplace entre le point de charge et le point de décharge comme il est illustré dans la figure 21.

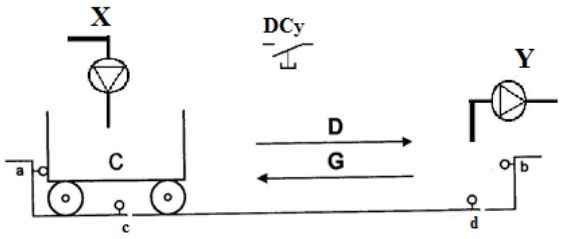


Figure 21 : Chariot.

2.3.1 Cahier de charge

Si on appuie sur le bouton poussoir 'DCy' et le chariot est en position de départ (le capteur de position 'a' est activé). La pompe 'X' se met en marche et remplit le bac du chariot jusqu'à l'activation du capteur 'c'. Ensuite le chariot se déplace vers la droite par l'actionneur 'D' jusqu'à la position de décharge détectée par le capteur 'b'. Après l'activation du capteur 'b', la pompe 'Y' décharge le chariot jusqu'à l'activation du capteur 'd'. une fois le chariot est déchargé il fait un retour à sa position de départ par l'actionneur 'G'.

2.3.2 Modélisation par GRAFCET

Pour commander ce système nous avons proposé une solution par le GRAFCET comme il est illustré dans la figure suivante :

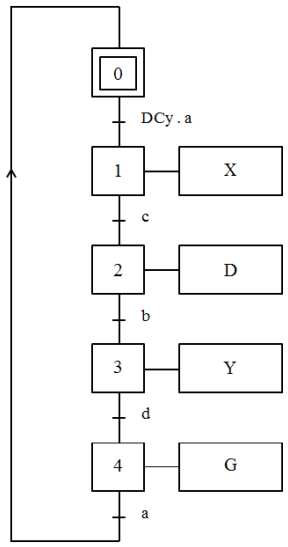


Figure 22 : Modélisation de l'exemple illustratif par le GRAFCET

2.3.3 Exemple des défauts

Ce système peut avoir plusieurs défauts. Le tableau suivant présente les problèmes dérive des capteurs défectueux :

Tableau 4 : Problème dérive des capteurs défectueux

Défaut du capteur	Problèmes dérive
c	la pompe ne s'arrêtera pas lors du remplissage du bac du chariot qui mène à la perte du produit.
a et/ou b	Le chariot ne va pas s'arrêter à la fin de sa course ce qui va causer le blocage du moteur ou le déraillement du chariot cela peut causer des dégâts matériaux ou humains.
d	la pompe ne s'arrêtera pas lors de la décharge du chariot ce qui peut causer la défectuosité de la pompe ou la perte du temps de la productivité.

Pour remédier à ces problèmes, il est indispensable d'élaborer un diagnostiqueur.

2.3.4 Approche de diagnostic

Le système de charge et de décharge est cyclique. Ce cycle s'exécute dans une marge de temps appelée : Durée Minimale du Cycle (DMC) qui varie en fonction de l'environnement (la charge du chariot, la température, la puissance électrique…). Alors il a une Marge d'Erreur Admissible par le Cycle (MEAC).

Dans un cycle séquentiel, il y a plusieurs étapes qui s'exécutent l'une après l'autre.

Cycle =$\sum$étapes(Ai) = $\{A_0, A_1, A_2, \ldots, A_i\}$

Une étape A_n 'par exemple le chargement' a les mêmes caractéristiques de son système, c'est-à-dire qu'elle a une durée d'exécution minimale et une marge d'erreur admissible. Ils sont présentés dans les figures 23 et 24.

DEMEi : Durée d'Exécution Minimale d'une Etape.

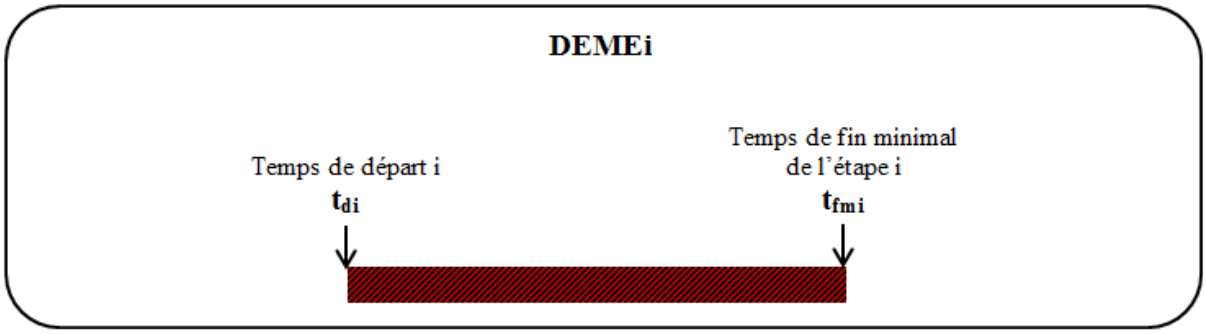


Figure 23 : Durée d'exécution minimale d'une étape.

$DEMEi = t_{fmi} - t_{di}$

MEAEi : Marge d'Erreur Admissible par l'Etape

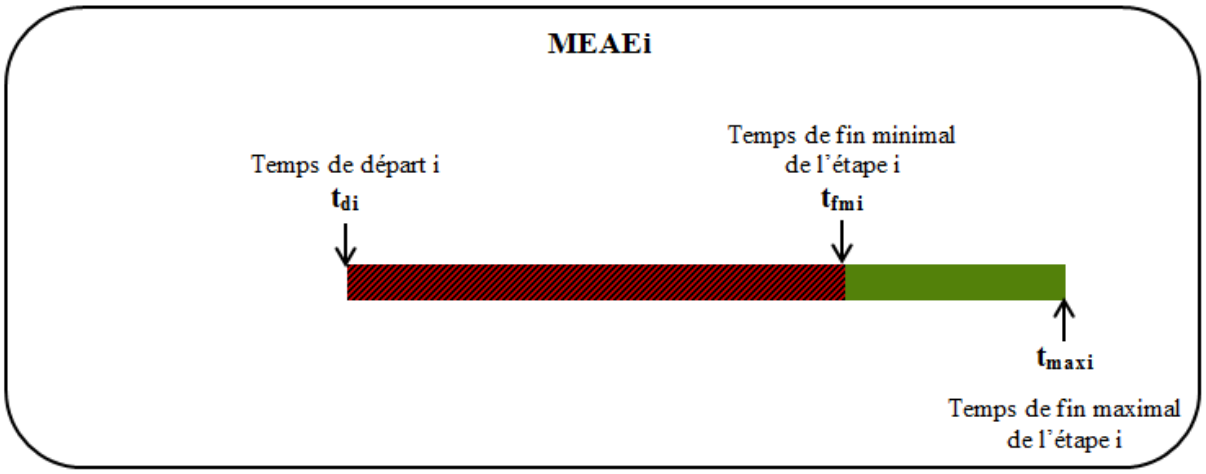


Figure 24 : Durée d'exécution minimale avec marge d'erreur admissible de fin de l'étape

$MEAE = t_{fx\,i} - t_{max\,i}$

Dans notre système l'étape Ai s'exécute après la fin de l'étape Ai-1 qui est définie sur une marge de temps MEAEi-1. Alors l'instant de départ de l'étape Ai est attendu sur une marge de temps comme la présente la figure suivante.

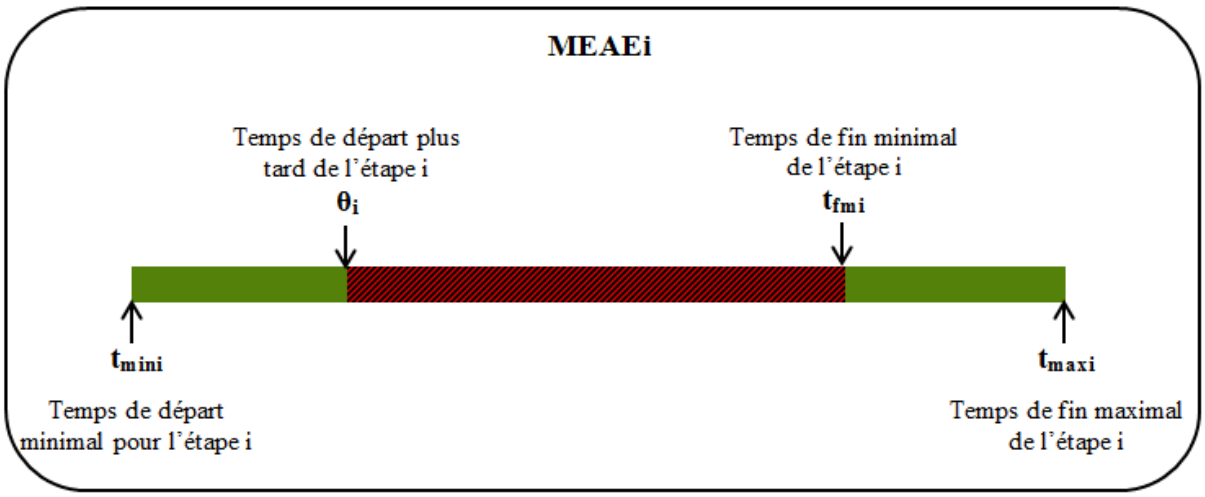


Figure 25 : Durée d'exécution minimale avec marge d'erreur admissible totale par l'étape

2.3.5 Diagnostiqueur

En nous basant sur cette approche nous proposons un diagnostiqueur qui vérifie si chaque étape respecte sa plage de temps, comme il le montre la figure suivante.

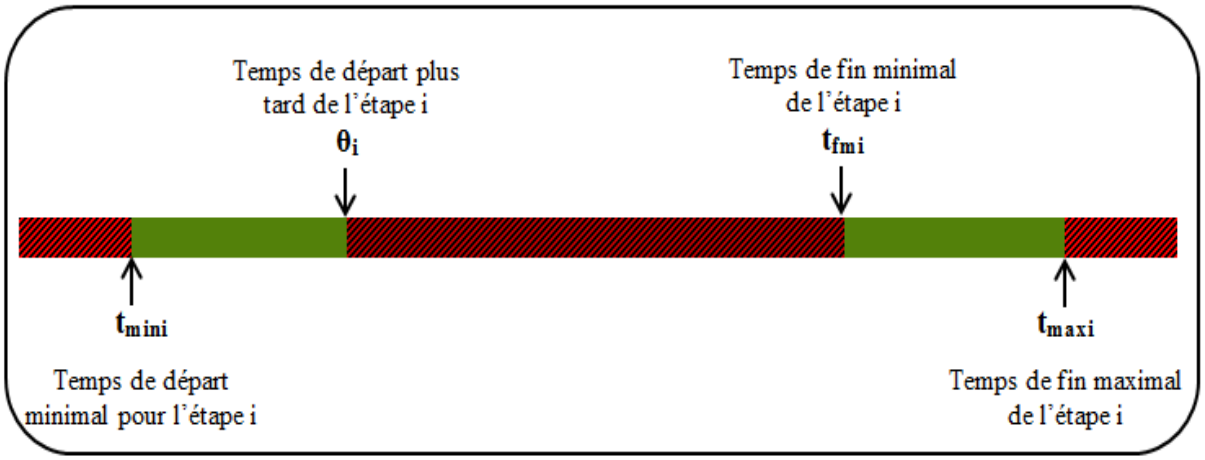


Figure 26 : les plages de temps permis pour une étape

Chapitre 3

Diagnostiqueurs Sous Contraintes Temporelles

Dans notre projet de recherche, nous avons proposé trois diagnostiqueurs pour répondre à notre question de recherche « diagnostiquer les systèmes à événements discrets ». Ces algorithmes sont basés sur notre proposition « La fonction exécutée par chaque système se fait selon un cycle périodique contenant différentes étapes et chaque étape est destinée à un intervalle de temps bien défini ».

Le premier diagnostiqueur est destiné au système séquentiel. Le deuxième algorithme est destiné pour tous types de système séquentiel et parallèle. Le troisième diagnostiqueur est dédié pour les SED avec des événements silencieux.

3.1 Diagnostiqueur pour les systèmes séquentiels

Notre premier diagnostiqueur est un diagnostiqueur en ligne qui est destiné au SED séquentiel. Il est développé sur trois étapes. Nous avons commencé par l'élaboration d'un algorithme qui est publié dans un journal indexé Scopus [67]. Après nous avons passé à son application. Cette étape est devisée en deux :

- **La première étape :** s'agit de développer une application informatique avec une interface qui facilite l'implémentation de l'algorithme et la prise de décision.
- **La deuxième étape :** concerne l'implémentation de l'application informatique sur un prototype industriel réel. L'implémentation de notre algorithme nous a permis d'avoir un feedback qui nous a poussés à améliorer l'algorithme initial.

3.1.1 Algorithme

En se basant sur notre proposition, nous avons formalisé notre premier algorithme suivant les étapes ci-dessous :

Etape 1 : Mesurer les temps d'exécution de chaque action : La mesure se fait dans la phase de test par un chronomètre (clock). La valeur utilisée est la moyenne des valeurs mesurées pour chaque action au cours d'un certain nombre de tests.

Etape 2 : Définir une plage d'erreur pour chaque action en fonction du résultat de diagnostic de sa gravité.

Etape 3 : Vérification du système via notre algorithme [18]. Le cœur de notre algorithme est élaboré comme suit :

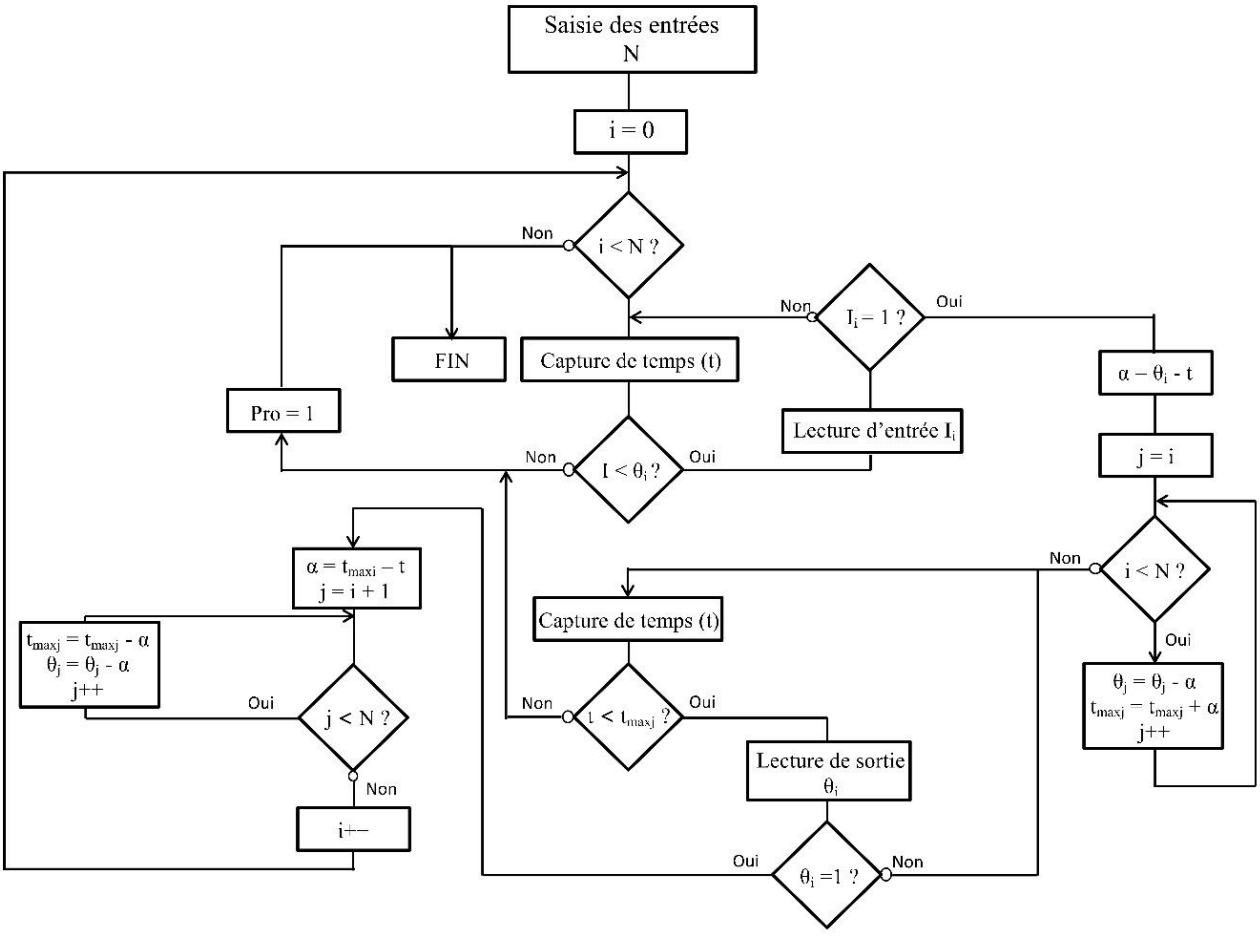


Figure 27 : Organigramme de l'algorithme

```
For ( i = 0; i< N; i + +)
{
        While ( t ≤ θi )
        {
                If ( Ii = 1) go to
        T1;}
```

Dans notre algorithme l'action i devrait commencer avant le temps de départ plus tard θi. L'information de début d'action i est donnée par la variable d'entrée

Si la valeur de l'horloge t dépasse la θi et i n'est pas encore activé, notre algorithme active la sortie " problème" (prob =1). Cela va arrêter le système ou déclencher une alarme en fonction du choix de l'utilisateur.

```
        Pro = 1;

        T1 : α=θi-t;

        For ( j = i, j < N; j + +)
        {
                θ j = θ j - α ;
                tmaxj = tmaxj -
        α; }
```

Après on soustrait la valeur α de tous les temps de départ plus tard des étapes qui ne sont pas encore exécutées et de tous les temps de fin max

Si Ii=1 avant l'instant θi, on déduit une valeur α avec l'équation suivante : α=θi-t.

Après vérification de l'étape, Ai a eu un bon départ dans la plage attendue par la suite, on vérifie l'heure de la fin de cette étape.

```
        While (t ≤ tmax)
        {
                If (Oi = 1)
        go to T2;
        }
        Prob = 1 ;
        T2 : α = t maxi - t
        For (j = i +1, j < N ; j + +)

        {
                tmaxj =tmaxj
        – α ;
                θj = θj - α ;
        }}
```

Si la valeur de l'horloge t dépasse le tmaxi et Oi n'est pas encore activé, notre algorithme active la sortie " problème" (prob =1). Cela va arrêter le système ou déclencher une alarme en fonction du choix de l'utilisateur.

Si Oi=1 avant l'instant tmax, on déduit une valeur α (α=tmax i - t), qui représente la différence entre le temps de fin et le temps réel de fin de l'étape Ai.

Etape 4 : Réagir en cas de problème qui se déclenche si l'action ne s'exécute pas dans la marge du temps permise.

3.1.2 Programmation et implémentation du diagnostiqueur

Pour mettre en œuvre notre solution nous devons l'appliquer sur un système réel, pour cela nous avons décidé de l'appliquer sur une maquette au laboratoire de l'automatisme/automatique. Cette maquette est un prototype d'une machine de traitement de surface. Dans cette partie nous avons commencé par la présentation de notre système, après nous avons passé à l'élaboration d'une application informatique avec une interface en fin nous avons testé l'application informatique qui permet d'implémenter notre algorithme dans une machine réelle afin de l'approuver et l'améliorer.

3.1.2.1 Présentation du prototype

L'équipement complet est constitué de deux sous-ensembles qui se raccordent facilement avec deux cordons équipés de connecteurs comme il illustre la figure suivante.

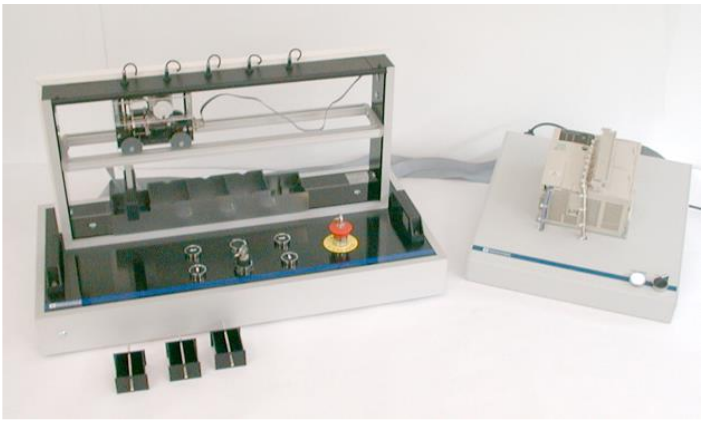


Figure 28 : Le prototype

Une partie opérative : Elle représente l'installation du traitement de surface comme (figures 28, 29, 30 et 31). Elle est composée de cinq postes : trois bacs pour le traitement des pièces et aux extrémités deux postes de chargement / déchargement. Un chariot se déplace sur un rail situé au-dessus de ceux-ci pour manipuler les paniers de pièces à traiter.

- La partie mécanique est constituée d'un portique fixe sur lequel se déplace horizontalement sur rails un chariot muni d'un treuil. Celui-ci se positionne au-dessus de cinq postes de travail :

  - Position 1 : Poste de chargement/déchargement des pièces à traiter (à gauche face aux boutons).
  - Position 2 : Bac de traitement N°1.
  - Position 3 : Bac de traitement N° 2.

- Position 4 : Bac de traitement N° 3.
- Position 5 : Poste de chargement/déchargement des pièces à traiter (à droite face aux boutons).

- Cinq détecteurs de proximité inductifs indiquent et contrôlent la position du chariot par rapport aux postes de travail.
- Deux détecteurs de proximité inductifs contrôlent les positions extrêmes du treuil (positions haute et basse).

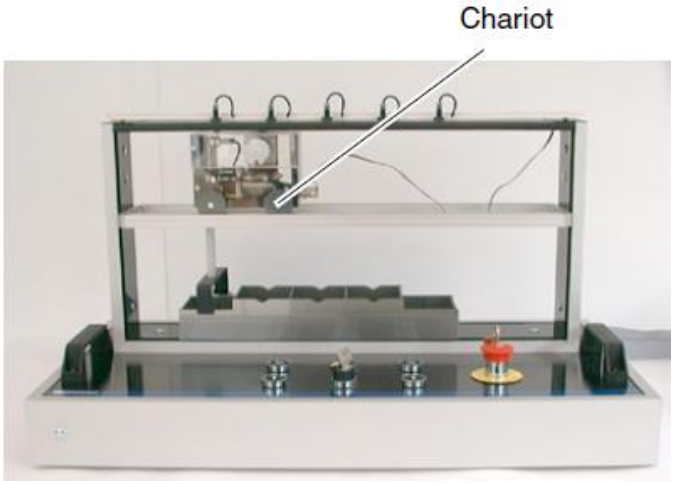


Figure 29 : Partie opérative du prototype

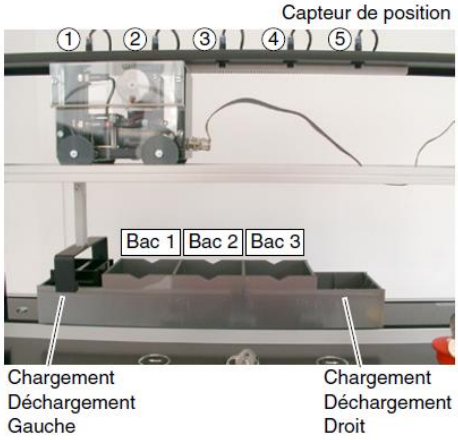


Figure 30 : Les capteurs de détection de la position du chariot

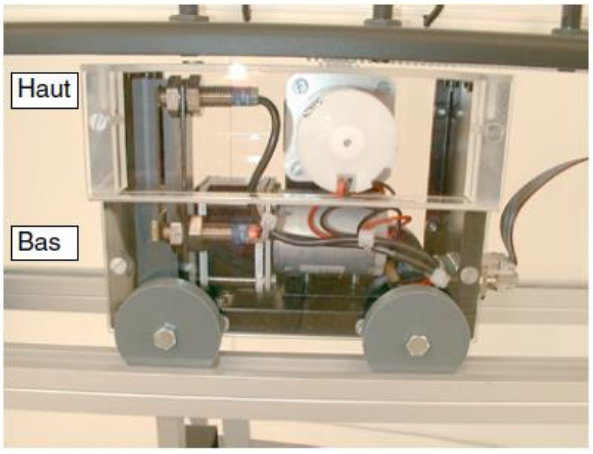


Figure 31 : Les capteurs de détection de la position du treuil

Le pupitre se situe sur l'avant de la partie opérative, il comprend :

- Un bouton : Coup de poing à accrochage mécanique, déverrouillage par clé ;
- Un commutateur à clé « Auto - 0 – Manu » retrait de la clé dans les trois positions ;

- Un bouton poussoir noir à impulsion « Départ cycle » en mode automatique, ou « retour à la position initiale » en mode manuel ;
- Quatre boutons poussoirs noirs à impulsions pour les manipulations du chariot et du treuil en mode manuel (Montée, Descente, Droite et Gauche) figure 31.

Figure 32 : Pupitre du prototype

Figure 33 : La partie commande

Une partie commande :

- Un automate de 16 entrées, 24 Vcc, 12 sorties et un relais, réf : TSX 3705028DR1 alimentés sous une tension de 230 V, transformée en 24 Vcc 0,4A qui est suffisamment puissante pour alimenter les capteurs et les actionneurs de l'équipement figure 32.
- Un commutateur avec deux positions : "Run / Stop".
- Un voyant blanc "Sous tension".
- Deux connecteurs (25 et 37 broches) pour le raccordement à la partie opérative.
- Une prise de format "Europa" avec un fusible incorporé pour le raccordement au réseau par le cordon du secteur fourni.

3.1.2.2 Modélisation du prototype

Dans le cadre de l'ingénierie des systèmes de production automatisés, la conception d'un contrôleur SED est une activité complexe. Elle nécessite souvent l'utilisation des méthodes impliquant un ensemble de langues spécifiques. En outre, nous observons que les graphiques sont de plus en plus préférés par rapport aux représentations littérales et algébriques en raison de leur capacité à servir de moyen de communication entre toutes les parties prenantes. Dans ce travail, nous proposons de faire référence à un outil de modélisation graphique habituel : le GRAFCET. Notre motivation est de préférer un outil largement utilisé dans l'industrie et qui peut être directement utilisé pour accéder à l'historique des mesures collectées avec l'automate.

GRAFCET (Commande Functional Chart Step/Transition) est un graphique qui décrit facilement les fonctionnalités d'une automation séquentielle [129]. Initialement, elle a été normalisée par AFNOR (Association Française de Normes) en 1977, puis par la CEI (International Electrotechnical Commission) en 1989.

Éléments de base :

GRAFCET est un graphe qui consiste à des séquences d'étapes et de transitions interconnectées par des liens directs.

- **Étapes :** L'étape représente un état dans lequel l'automatisation est invariante par rapport à ses entrées. Elle peut être active ou inactive. L'état du GRAFCET est défini à une date donnée par toutes ses étapes actives.
- **Transition :** La transition reflète la possibilité de changer d'un état à l'autre. Cette évolution est la conséquence de la transition. Une transition est prête à être activée si toutes ses étapes d'entrée sont actives.
- **Axé sur la connexion :** Un lien direct relie une étape à une transition et vice versa.

Interprétation du graphique :

Les capteurs et les actionneurs sont connectés à l'automate en fonction des entrées et des sorties. Les activités des actionneurs sont ensuite définies avec des actions et les informations fournies par les capteurs utilisés dans les fonctions de réceptivité.

- **Activité :** L'activité présente l'ensemble des actions qui spécifient ce qui doit être exécuté au cours de chaque étape. Une action peut être interne (compteur, temporisation d'armes) ou externe (sortie API).
- **Réceptivité :** La réceptivité est une expression booléenne qui peut prendre des valeurs vraies ou fausses et qui contrôle les commutateurs d'étape. Une variable peut être interne (état de l'étape de synchronisation) ou externe (entrée API).

Règles d'évolution :

L'aspect dynamique est défini par les règles suivantes :

- La situation initiale correspond aux étapes actives au début de l'opération.
- Une transition est validée lorsque toutes ses étapes précédentes sont activées. Le déclenchement d'une transition est effectif lorsque la transition est activée et lorsque la fonction de réceptivité associée est vraie.

- Le déclenchement d'une transition entraîne immédiatement l'activation des étapes en aval de cette transition et la désactivation de ses étapes en amont.
- Plusieurs transitions peuvent être déclenchées simultanément.
- Pendant le fonctionnement, soit une étape ou plusieurs sont activées ou désactivées simultanément.

Pour piloter notre prototype, nous avons besoin des entrées/sorties du système qui sont présenté dans le tableau suivant.

Tableau 5 : Variables utilisées de l'API

	Symbole	**Adresse**	**Description Désignation**
ENTREES	D1	%I0.7	Capteur de position : Le chariot est en post 1
	D2	%I0.8	Capteur de position : Le chariot est en post 2
	D3	%I0.9	Capteur de position : Le chariot est en post 3
	D4	%I0.10	Capteur de position : Le chariot est en post 4
	D5	%I0.11	Capteur de position : Le chariot est en post 5
	D6	%I1.0	Capteur de position : Le chariot est en bas
	D7	%I1.1	Capteur de position : Le chariot est en haut
	S1	%I1.3	Bouton poussoir : Départ cycle 'DCY'
	S2	%I0.4	Bouton poussoir : Pour déplacer le chariot à gauche
	S3	%I0.3	Bouton poussoir : Pour déplacer le chariot à droite
	S4	%I0.1	Bouton poussoir : Pour faire monter le chariot
	S5	%I0.0	Bouton poussoir : Pour faire descendre le chariot
	S6	%I0.5	Commutateur : Pour le fonctionnement automatique
	S7	%I0.6	Commutateur : Pour le fonctionnement manuel
	S8	%I1.2	Arrêt d'urgence
SORTIES	KM1	%Q0.0	Moteur pour déplacer le chariot à gauche
	KM2	%Q0.1	Moteur pour déplacer le chariot à droite
	KM3	%Q0.2	Moteur pour faire descendre le chariot
	KM4	%Q0.3	Moteur pour faire monter le chariot

Variables internes : Conditions initiales %I1.7.%I1.12

On va réaliser le déplacement du chariot (un aller et un retour) par l'actionnement du bouton poussoir S2 (Départ cycle). Les GRAFCET opérationnels et fonctionnels sont présenté dans la suivante :

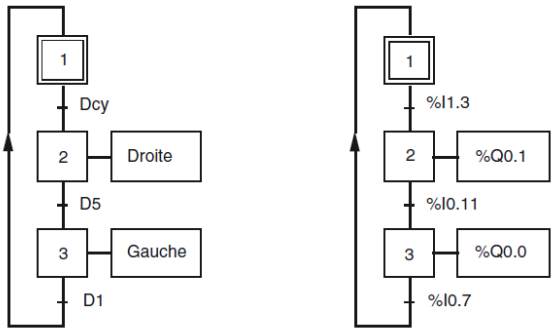


Figure 34 : Les GRAFCET opérationnels et fonctionnels du système

3.1.2.3 Communication avec le système

Pour implémenter notre algorithme, nous devrons utiliser un langage qui est facile à installer sur notre maquette et aussi facile à utiliser. D'après notre recherche nous avons choisi le langage VBA : Le VBA a deux atouts majeurs [130] **:**

- Le VBA est extrêmement utilisé dans les entreprises, notamment l'industrie ;
- Le VBA est beaucoup plus accessible aux yeux d'un utilisateur que ne pourrait l'être un programme C# (C sharp). En effet, toute personne qui utilise Microsoft Office possède aussi les outils nécessaires à la rédaction et au maintien des applications, contrairement au C# qui nécessite l'investissement dans l'IDE Visual Studio. Pour cette raison, le développement d'une application en VBA semble être un investissement beaucoup moins bloquant ou impactant pour des clients qui donnent le feu vert au développement d'une application en VBA qu'ils auraient pu refuser en C#.

Pour l'instant on a :

- Elaboré un algorithme pour diagnostiquer les systèmes à événements discrets ;
- Choisi le prototype où on va appliquer notre algorithme ;
- Développé une application informatique avec le langage VBA pour implémenter notre algorithme dans le système.

On trouve parmi les matériels fournis, un câble de communication multifonction branché à un connecteur mini DIN mâle 8 broches (Port terminal RS485) et dans l'autre côté un connecteur périphérique série (RS232) et en milieux un convertisseur incluant le commutateur rotatif. Quatre positions pour sélectionner les différents modes de fonctionnement. Ce câble prenant en charge la conversion des signaux RS485 en Signaux RS232, il assure la connexion entre un automate (Twido, Nano, Micro, Premium, Neza) et divers Dispositifs de la série RS232 (ordinateur…). Les différentes fonctions du mode série sont sélectionnées selon la position du commutateur rotatif accessible sur l'unité convertisseur.

- Position 0 : TER MULTI -La connexion en mode point à point ou multipoint. Cette position force le port du terminal en mode maître, protocole par défaut ;
- Position 1 : AUTRES MULTI : Connexion en mode multipoint ou autres types de communication ;
- Position 2 : TER DIRECT - -La connexion en mode point à point ou multipoint. Cette position force le port du terminal en mode maître, protocole par défaut ;
- Position 3 : AUTRE DIRECT - Connexion en mode point à point.et autres types de communication sont définies par la configuration de l'automate.

3.1.2.4 Mise en place de l'application informatique

La mise en place de notre diagnostiqueur passe par les étapes suivantes :

3.1.2.4.1 Configuration

On configure l'automate de façon habituelle et on configure le protocole de communication comme l'illustre la figure suivante :

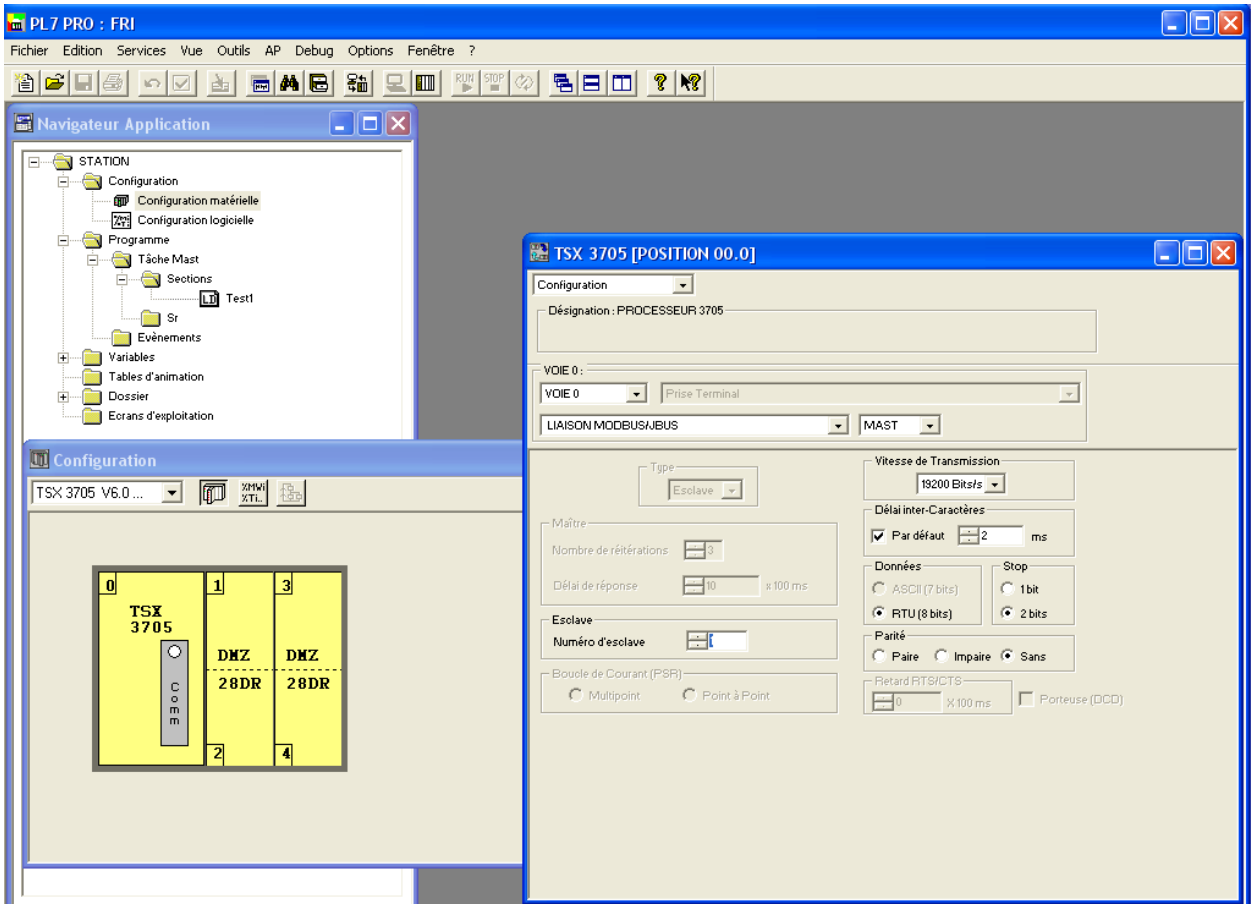


Figure 35 : Configuration du type de communication

Après cette configuration, on met le convertisseur du câble de communication multifonction dans l'état 3 (OTHER DIRECT).

3.1.2.4.2 L'exécution de programme

On place le fichier Excel dans la répartition C du disque dur, après on lance l'application, la fenêtre suivante apparait, figure 35.

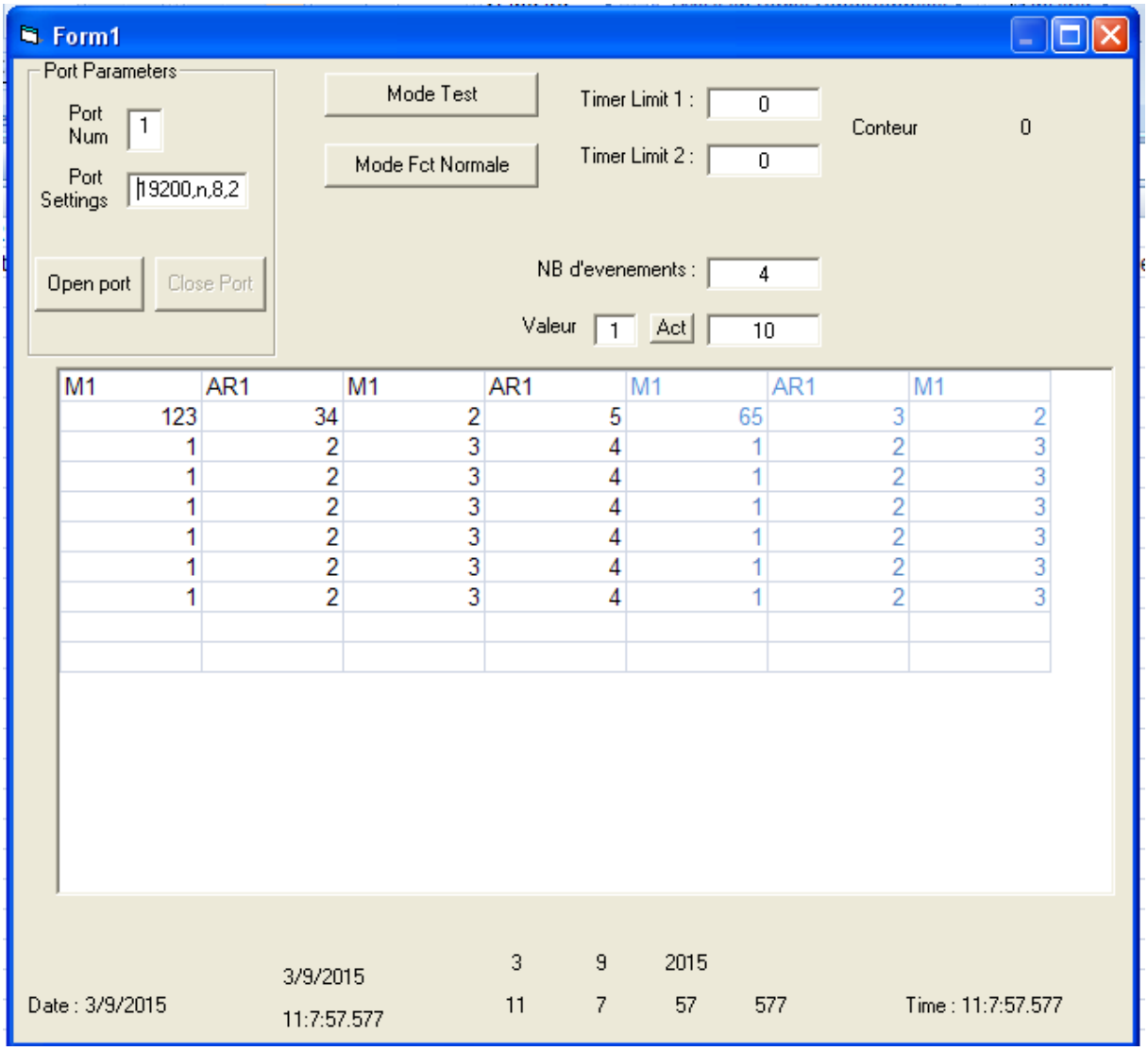
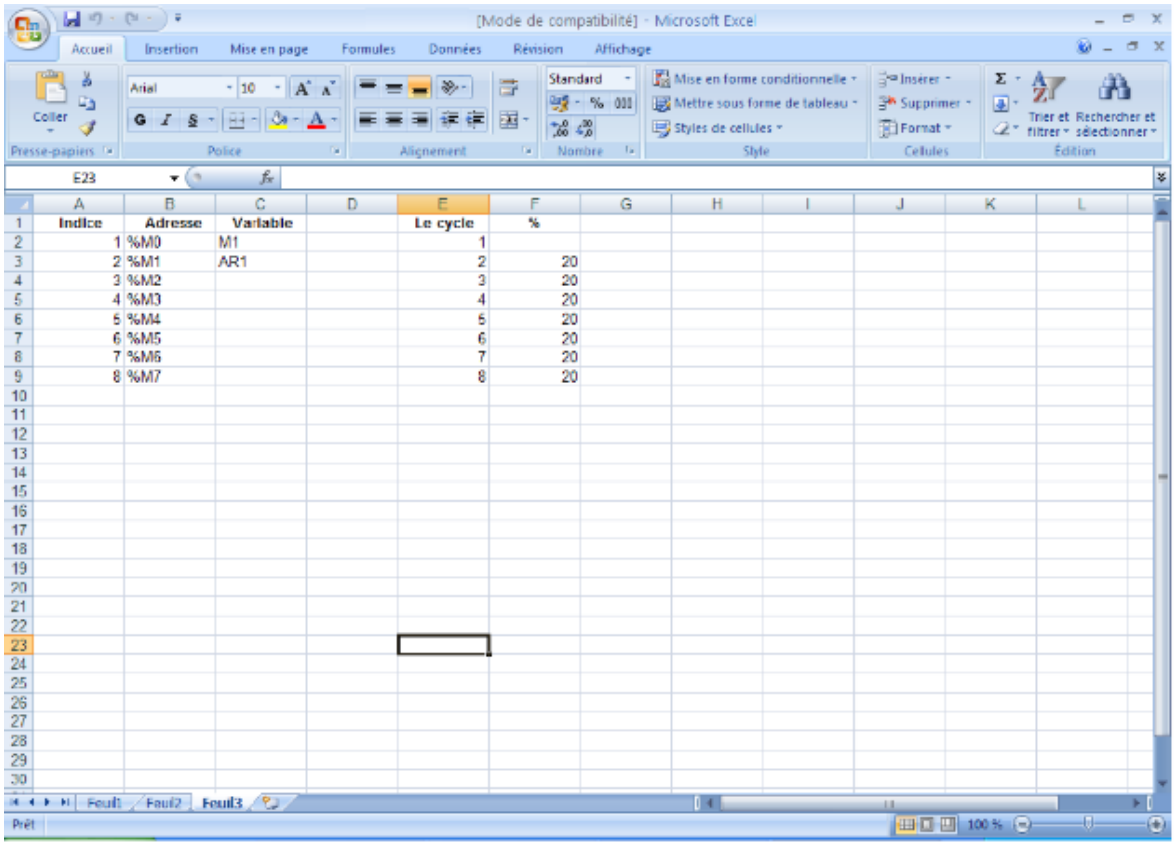


Figure 36 : L'interface de l'application

On détermine le nombre des événements suivant le cycle (dans notre cas, on a 4 événements), après, on appuie sur le bouton « Open Port », après un fichier Excel s'ouvre sur la feuille3, figure 36 :



	A	B	C	D	E	F
1	Indice	Adresse	Variable		Le cycle	%
2	1	%M0	M1		1	
3	2	%M1	AR1		2	20
4	3	%M2			3	20
5	4	%M3			4	20
6	5	%M4			5	20
7	6	%M5			6	20
8	7	%M6			7	20
9	8	%M7			8	20

Figure 37 : Le fichier Excel

La colonne A et la colonne B se remplissent automatiquement. Dans la colonne C, on remplit les variables. Dans la colonne E., on définit l'ordre des séquences. Dans la colonne F on détermine le pourcentage des erreurs permissibles pour chaque séquence, dans notre cas on a fixé 20% comme marge d'erreur pour toutes les séquences.

3.1.2.4.3 Mode Test

On retourne vers l'application, on appuie sur « Mode test », la fenêtre suivante apparaitra, figure 37 :

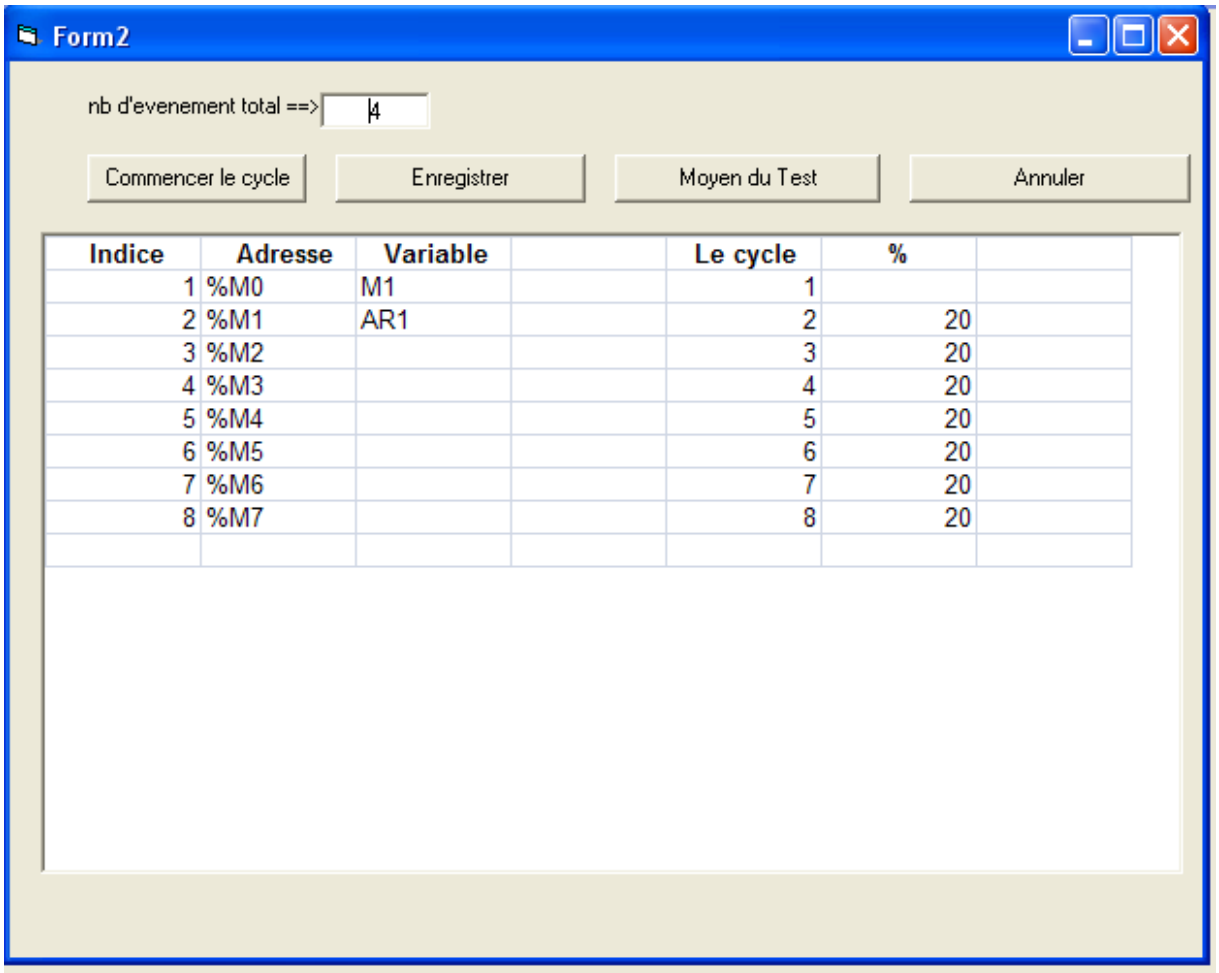


Figure 38 : Interface du mode de teste

On appuie sur le bouton « Commencer le cycle » et le processus commence. A la fin de ce processus, la fenêtre suivante apparait, figure 38 :

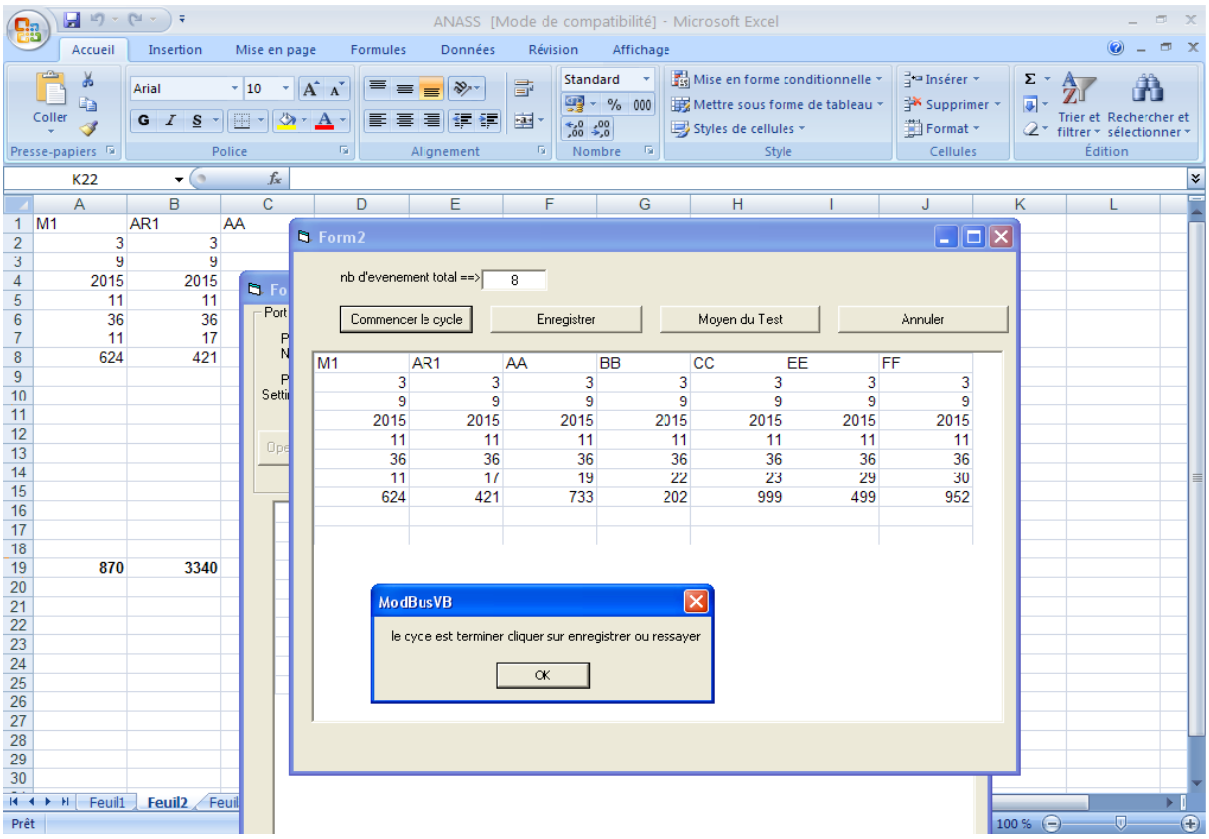


Figure 39 : fenêtre de confirmation de la validation de la fin du teste

Puis, on appuie sur OK pour confirmer la fin du cycle, après on appuie sur « enregistrer », ensuite sur « Moyen de Test ». A ce moment, on a le choix de faire un autre test pour avoir plus de précision (le nombre de tests est illimité selon la mémoire de l'Excel), ou on ferme le mode test et on lance le fonctionnement normal.

Notre analyse est basée sur le concept de base de GRAFCET. Le concept de transition intermédiaire est le même, mais nos méthodes se concentrent sur un seul de ces trois signaux : soit le front descendant de l'action qui était active, soit le front montant de la transition qui a désactivé l'action précédente, soit le front montant de l'action suivante. L'hypothèse suivante est prise en compte : même si le temps de passage d'une transition est infiniment petit, il sera considéré comme nul, car il est instantané [131].

3.1.2.4.4 Mode « fonctionnement normal »

Dans cette phase, on lance le processus, s'il n'y a pas de problème notre diagnostiqueur n'intervient pas, si l'une des séquences ne respecte pas sa marge permise, l'alarme %Q4.0 s'activera, figure 39.

Figure 40 : Décision prise après un défaut temporel

3.1.3 Feedback

Dans cette application, nous définissons l'instant de départ du cycle qu'il soit un point de référence et tous les événements se réfèrent à cet instant exact. À chaque fin d'un événement, l'algorithme recalcule les marges de tous les événements restants. Dans le cas où on a plusieurs événements futurs et l'événement présent est très court, notre algorithme prend plus de temps pour recalculer les nouvelles marges des événements futurs. Quand il finit, il trouve que l'événement présent est déjà écoulé. Il n'arrive pas à le détecter. C'est là où il déclenche un problème même si tout est normal. Pour résoudre ce problème, nous avons amélioré le premier algorithme. Ce nouvel algorithme est publié dans un journal indexé Thomson Reuters. La modification dans le corps du diagnostiqueur est présentée dans la figure suivante :

```
Public class surveillance {
Public static void main (arg[] stag)
{Long θ, t, α;
Date θ[] = θ[N-1];
BooleanI[] =I[N-1];
Date tmax[] =tmax[N-1];
Boolean O[] =O[N-1];
For (int I; i<N; i++)
{ while (t≤ θ[i])
{ if (Ii==1) continue T1; }
pro=1;
T1: α= θ[i] –t;
for (int j=I; j<N;j++)
{ tmax[j] = tmax[i]-α; }
While (t<=tmax)
{if (O[i] ==1) continue T2;}
prob=1;
T2: α= tmax[i] –t;
for (j=i+1; j<N; j++)
{θ[j] = θ[j]-α;  }}}}
```

Figure 41 : Cœur du deuxième algorithme

Cet algorithme ne considère pas l'instant de départ comme un point de référence pour tous les événements. Dans cet algorithme, quand on termine le traitement de l'événement « Ai » on calcule les marges de l'événement « Ai+1 » en prenant l'événement « Ai » comme référence. Comme ça l'algorithme ne prend pas beaucoup de temps pour calculer les marges de l'événement suivant, ce qui le rend plus rapide et efficace.

Nous avons intégré cette modification dans notre application et nous avons eu les résultats souhaités. Mais dans l'industrie il n'y a pas que les systèmes séquentiels, on trouve les systèmes parallèles ou avec choix. Ce qui nous ouvre la pensée à développer notre diagnostiqueur afin de diagnostiquer les différents types des systèmes.

3.2 Diagnostiqueur pour les systèmes séquentiels, parallèles, à choix

Après le succès du notre premier diagnostiqueur pour les systèmes séquentiels. Nous avons élaboré un algorithme pour diagnostiquer tous types de système afin de répondre à la complexité des nouvelles technologies.

Pour simplifier l'illustration de notre nouvel algorithme de diagnostic, nous allons utiliser un exemple général qui présente un système (séquentiel, parallèle, à choix) qui contient les éléments de base comme indiqué dans la figure suivante :

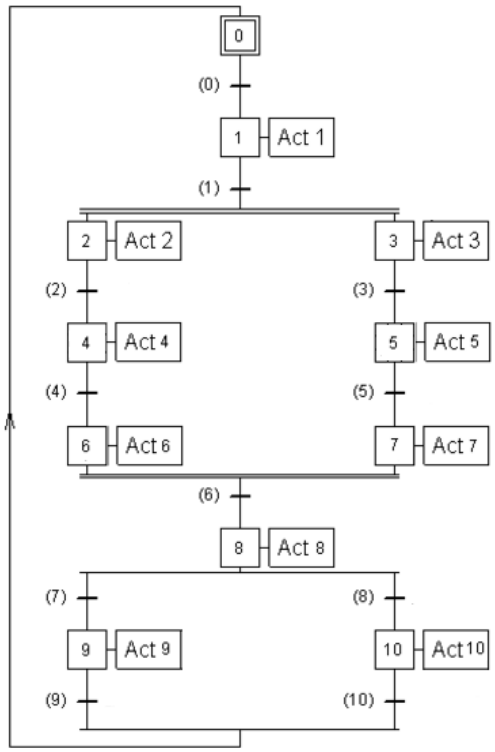


Figure 42 : Exemple d'un processus

3.2.1 Algorithme

Le processus de diagnostic est divisé en trois étapes : la modélisation, le diagnostic et la décision.

- **Modélisation :** Dans cette étape, nous modélisons l'information donnée et les caractéristiques structurelles de notre système.

  – Étape 1 : Déterminer les types de zones (séquentielles, parallèles ou choisies) Le système est divisé en zones simples : pour l'exemple de la Figure 41, ce système peut être divisé en cinq zones comme il montre la figure 42. Les parties séquentielles et la synchronisation sont isolées (zones A et C) ; Le choix de la sous-séquence parallèle est séparé (zone B) ; La sélection des choix de sous-séquences est séparée (zones D et E).

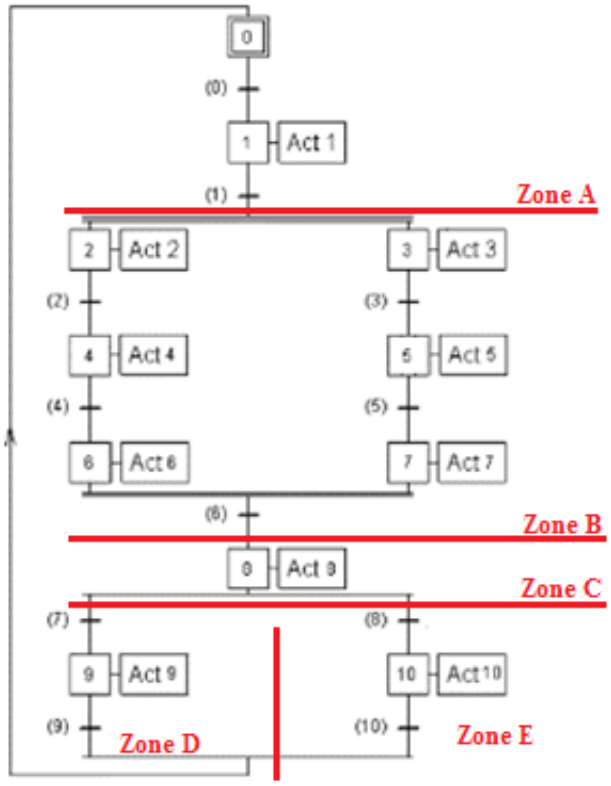


Figure 43 : Exemple d'un processus divisé en plusieurs zones

– Étape 2 : Définir la séquence des étapes individuelles dans chaque zone.

  - Zone A : il y a l'événement (1).
  - Zone B : cette zone affiche des événements qui fonctionnent de façon synchrone : il n'est pas possible de définir un ordre d'exécution précis sans une compréhension très approfondie du système, car il existe une possibilité d'intervalles de chevauchement. Les intervalles de chevauchement indiquent qu'il existe plusieurs modes d'exécution possibles. Avec le manque des exigences claires, nous quittons cette zone sans contraindre l'ordre d'exécution, nous considérons donc que tout ordre d'arrivée des événements est correct.
  - Zone C : il y a l'événement (8).
  - Zone D : il y a l'événement (9).
  - Zone E : il y a l'événement (10).

- **Diagnostic :** Les événements de toutes les sous-séquences sont analysés par l'algorithme de diagnostic proposé. Cet algorithme reçoit des informations contenant l'étiquette et la date de chaque événement en fonction du modèle proposé. Il vérifie si les contraintes temporelles sont satisfaites ou non.

Le programme est présenté dans la figure survente :

```
#include <stdio.h>
main()
{inta,d,b,n,j,t,e,f,TO[1000][1000],TE[1000][4];
printf("donner le nombre d'événements:");
scanf("%d",&a);
for(b=0; b<a; b++)
{printf("nom de la variable");
scanf("%d",&TE[b][0]);
printf("durée min du variable");
scanf("%d",&TE[b][1]);
printf("durée max du variable");
scanf("%d",&TE[b][2]);}
printf("donnée le nombre des événements observés");
scanf("%d",&d);
for(b=0;b<d;b++)
{printf("donner le nom de l'événement observé");
scanf("%d",&TO[b][0]);
printf("donner le temps d'observation");
scanf("%d",&TO[b][1]);}
e=0;
for(b=0;b<a;b++)
```

```
{for(j=0;j<d;j++)
{if(TO[j][0]==TE[b][0])
{f=TO[j][1]-e;
if (f>=TE[b][1])
{printf("l'événement %d respecte le temps min \n",TO[j][0]);}
else
{printf("l'événement %d ne respecte le temps min\n",TO[j][0]);}
if (f<=TE[b][2])
{printf("l'événement %d respecte le temps max \n",TO[j][0]);}
else
{printf("l'événement %d ne respecte le temps max\n",TO[j][0]);}
e=TO[j][1];}}
return 0;}
```

Figure 44 : Algorithme programmé en langage C.

Les étapes de l'algorithme du diagnostic sont :

- Vérification de l'observation de tous les événements.
- Vérification de l'exécution des événements selon l'ordre souhaité (sauf pour le cas du parallélisme, par exemple la zone B dans l'exemple de la figure 42).
- Vérification de l'exécution des événements selon la contrainte temporelle.

- **Décision :** Il existe deux types possibles de cas :

  - Cas 1 : Tous les événements ont été observés, exécutés dans l'ordre souhaité, et la contrainte temporelle a été respectée, aucune décision n'est prise.
  - Cas 2 : Les décisions dépendent de la spécification définie par le concepteur. Dans le cas de l'apparition d'une erreur, le concepteur spécifie s'il veut que le système : s'arrête, déclenche une alarme ou affiche un message d'erreur. Dans notre cas, nous avons codé notre algorithme pour qu'il affiche un message qui donne des informations sur tous les événements du système.

3.2.2 Test du diagnostiqueur

Pour valider l'algorithme proposé, deux exemples de base sont présentés : l'un est un système à choix (sélection de séquences) et l'autre est un système parallèle.

3.2.2.1 Exemple d’un système à choix

3.2.2.1.1 Description du procédé

Un chariot qui délivre les deux produits (P1 et P2) : P1 pour mémoriser M1 et P2 pour mémoriser M2. Le cycle commence par le remplissage du chariot, qui doit être en position STOCK. Le remplissage effectué via une pompe P qui s'active en appuyant sur le bouton "DCY" et s'éteint lors du remplissage du chariot "CH". La livraison du produit P1 au magasin M1 se fait en appuyant sur le bouton-poussoir S1 après le chargement du chariot, puis le chariot effectue un retour de la mémoire M1. La livraison du produit P2 au magasin M2 s'effectue en appuyant sur le bouton-poussoir S2 après le chargement du chariot, puis le chariot effectue un retour du magasin M2. Le choix du magasin active un cylindre qui assure la continuité des lignes du chariot. La figure suivante montre le système et l'opération modélisés avec GRAFCET.

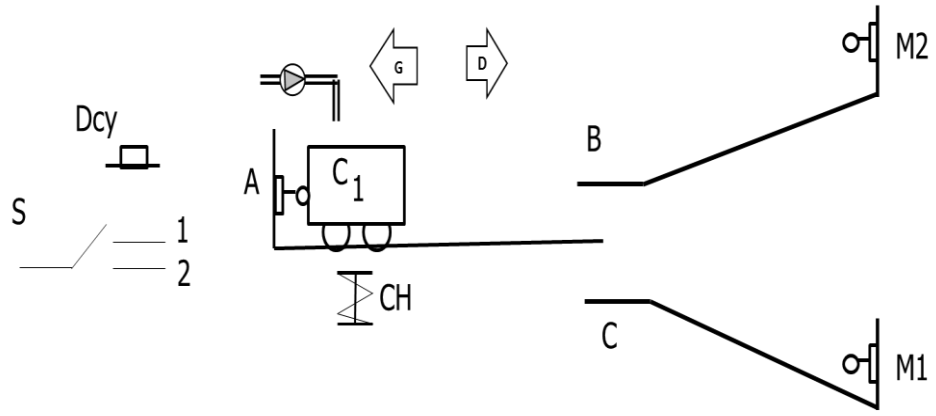


Figure 45 : Système de choix

Pour appliquer notre algorithme, nous devons suivre les trois étapes qui sont : la modélisation, le diagnostic et la décision.

3.2.2.1.2 Modélisation

La modélisation du système présenté dans la figure 44 est présentée dans la figure suivante :

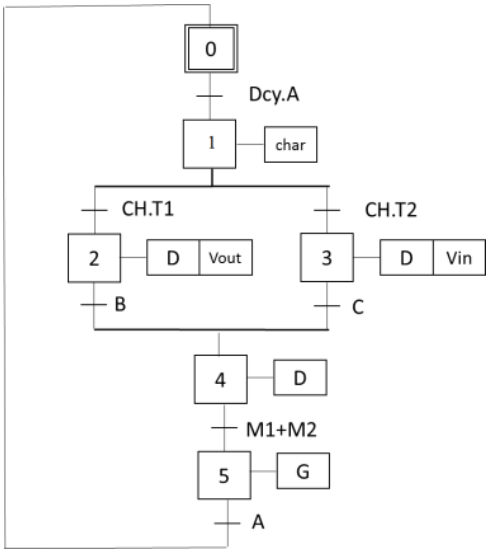


Figure 46 : Fonctionnement du système de choix avec un GRAFCET.

Dans l'exemple de la figure 44, il existe trois zones comme il est présenté dans la figure 46 :

- Zone A : les événements 0 et 1 doivent se produire dans l'ordre suivant : 0,1
- Zone B : les événements 2 et 3 doivent se produire dans l'ordre suivant : 2,3
- Zone C : les événements 4 et 5 doivent se produire dans l'ordre suivant : 4,5

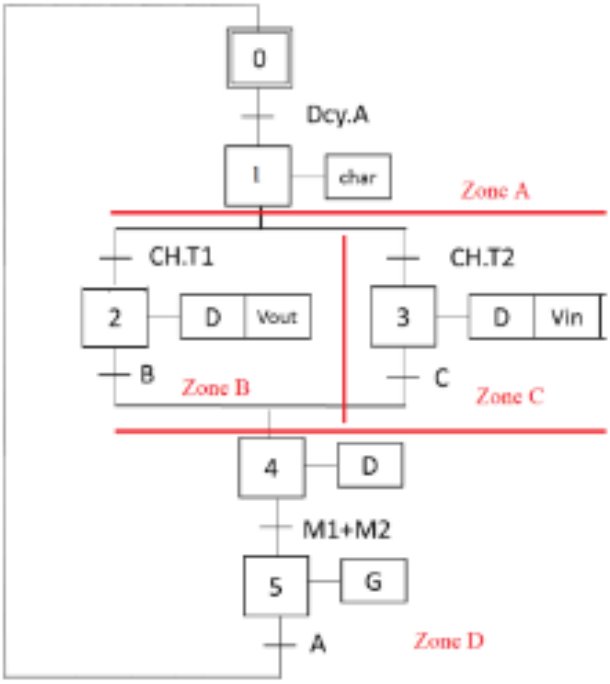


Figure 47 : Exemple d'un processus avec le choix des événements divisés en plusieurs zones

Des spécifications temporelles pour notre système sont présentées dans le tableau suivant :

Tableau 6 : Les spécifications de l'intervalle correct sont détaillées

Action Ai	Durée minimale : dmin i	Durée maximale : dmax i
1	1	2
2	1	3
3	2	5
4	2	4
5	1	2

Deux exemples de séquences chronométrées sont considérés, comme illustre notre approche.

3.2.2.1.3 Premier cas de fonctionnement : Première séquence

Diagnostic

Nous avons proposé la séquence dans le tableau suivant :

Tableau 7 : Première séquence

Actions	1	2	3	4	6	5
Dates	1	2	3	4	8	5

Décision

Le diagnostiqueur fournit les résultats suivants pour S1 :

```
L'événement 1 respecte l'heure minime
L'événement 1 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 6 respecte l'heure minime
L'événement 6 respecte l'heure maximale
L'événement 5 respecte l'heure minime
L'événement 5 respecte l'heure maximale
```

3.2.2.1.4 Deuxième cas de fonctionnement : Deuxième séquence

Diagnostic

Nous avons proposé la séquence dans le tableau suivant :

Tableau 8 : Deuxième séquence

Actions	1	2	3	4	6
Dates	1	2	6	7	10

Décision

Le diagnostiqueur fournit les résultats suivants pour S2 :

```
 L'événement 1 respecte l'heure minime
L'événement 1 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 ne respecte pas l'heure min
L'événement 4 respecte l'heure maximale
L'événement 6 respecte l'heure minime
L'événement 6 respecte l'heure maximale
L'événement 5 est inobservable
```

3.2.2.2 Exemple d'un système parallèle

Nous avons attaqué maintenant un système avec des événements qui s'exécute en parallèle.

3.2.2.2.1 Description du procédé

aEn appuyant sur le bouton "DCY", deux chariots se déplacent simultanément. Le chariot C1 se déplace entre A et B et l'autre chariot C2 se déplace entre C et D. La figure suivante montre le système et l'opération modélisés avec GRAFCET.

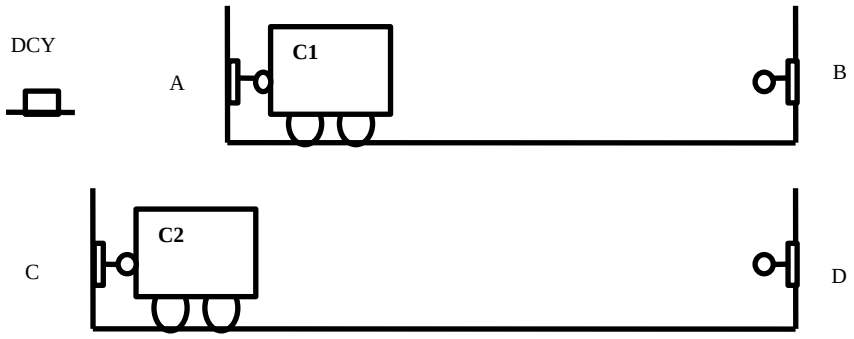


Figure 48 : Système de choix

Pour appliquer notre algorithme, nous devons suivre les trois étapes suivantes : la modélisation, le diagnostic et la décision.

3.2.2.2.2 La modélisation

La modélisation du système présenté dans la figure 49 est présentée dans la figure suivante :

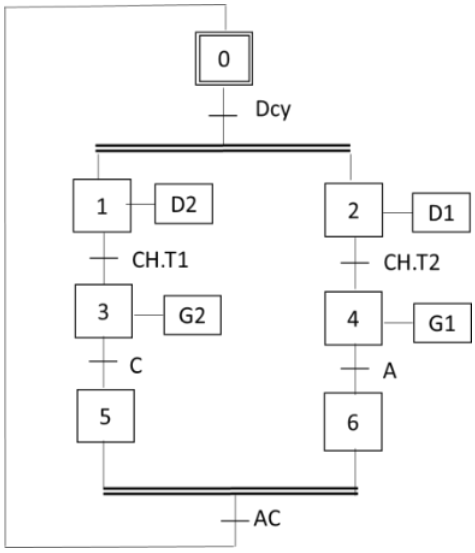


Figure 49 : Fonctionnement avec un GRAFCET pour le système de choix.

Dans l'exemple de la figure 49, il y a deux zones :

- Zone A : événement 0
- Zone B : événements 1,2, 3, 4,5 et 6

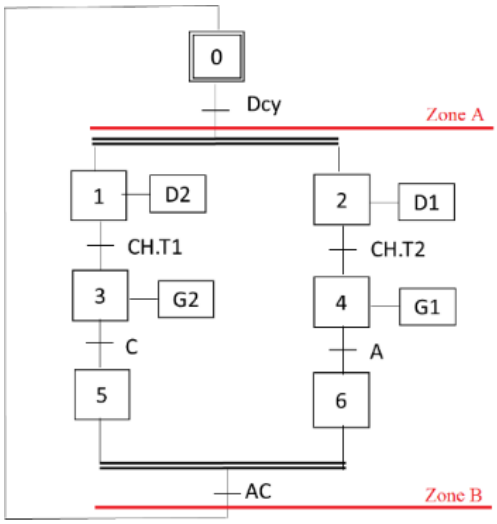


Figure 50 : Exemple d'un processus avec des événements parallèles divisés en plusieurs zones

Des spécifications temporelles pour notre système sont présentées dans le tableau suivant :

Tableau 9 : Paramètres du modèle

Action Ai	Durée minimum : dmin i	Durée maximum : dmax i
1	1	2
2	1	3
3	2	5
4	2	4
5	1	2
6	3	5

Deux exemples de séquences chronométrées sont considérés, comme notre approche l'illustre.

3.2.2.2.3 Premier cas de fonctionnement : Première séquence

Diagnostic

Les spécifications temporelles sont détaillées dans le tableau suivant.

Tableau 10 : Première séquence

Actions	1	2	3	4	6	5
Dates	1	2	3	4	8	5

Décision

Le diagnostic fournit les résultats suivants pour S1 :

```
L'événement 1 respecte l'heure minime
L'événement 1 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 6 respecte l'heure minime
L'événement 6 respecte l'heure maximale
L'événement 5 respecte l'heure minime
L'événement 5 respecte l'heure maximale
```

3.2.2.2.4 Deuxième cas de fonctionnement : Deuxième séquence

Diagnostic

Les spécifications temporelles sont détaillées dans le tableau suivant.

Tableau 11 : Deuxième séquence

Actions	1	2	3	4	6
Dates	1	2	6	7	10

Décision

Le diagnostic fournit les résultats suivants pour S2 :

```
L'événement 1 respecte l'heure minime
L'événement 1 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 ne respecte pas l'heure min
L'événement 4 respecte l'heure maximale
L'événement 6 respecte l'heure minime
L'événement 6 respecte l'heure maximale
L'événement 5 est non observable
```

3.2.3 Feedback

Les résultats obtenus sont très satisfaisants. Dans cet article, nous avons terminé le travail de base qui ne traite que des systèmes séquentiels, et nous avons abordé d'autres types de systèmes : systèmes de choix et systèmes parallèles, qui sont à la base de tous les systèmes industriels. Notre algorithme est prouvé avec des exemples, dans lesquels nous avons détecté tous les types de problèmes : le non-respect de dmin; le non-respect de dmax; et le cas où l'action est inobservable (événements silencieux).

3.3 Diagnostiqueur pour les systèmes à événements non observables

Nos systèmes sont déterministes et finis, ce qui signifie que la structure de notre système est connue. Pour expliquer le principe dans le système de diagnostic proposé, on utilisera un exemple général qui illustre un système avec des éléments de base (séquentiel, parallèle, à choix), comme le montre la figure 54 :

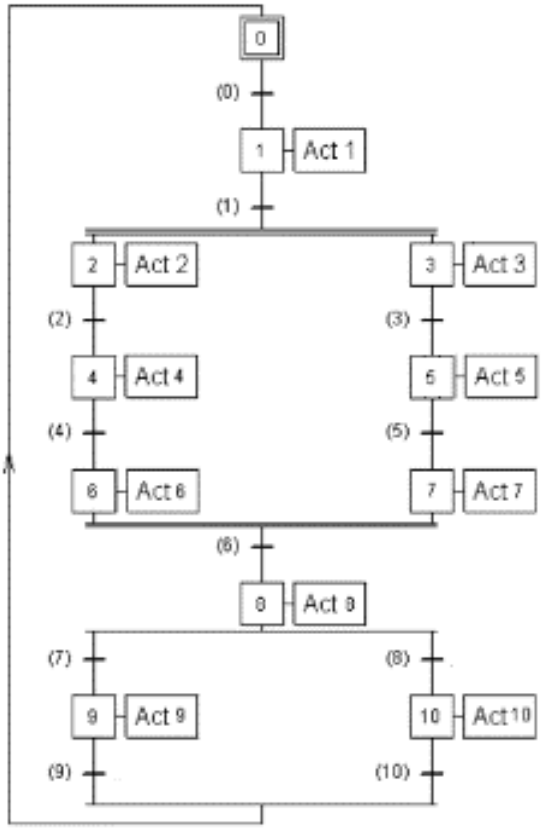


Figure 51 : Exemple de processus général

3.3.1 Algorithme

Le système de diagnostic comporte trois étapes : la modélisation, le diagnostic et la décision.

- **Modélisation :** Dans cette étape, nous saisissons les informations données et les caractéristiques du système. Cette étape est divisée en plusieurs étapes. Le système est divisé en zones plus petites : par exemple, le GRAFCET de la Figure 54 peut être divisé en cinq zones comme indiqué dans la figure suivante :

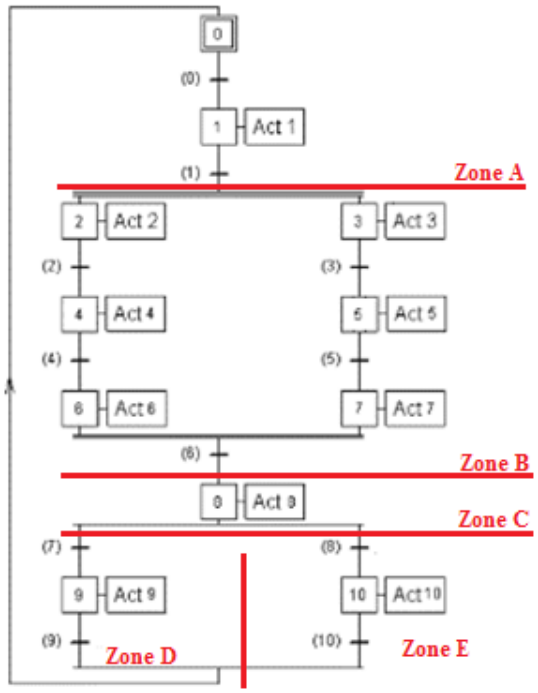


Figure 52 : Exemple d'un processus divisé en plusieurs zones

- Zone A : Il existe deux événements (0, 1) qui doivent être exécutés séquentiellement.
- Zone B : Cette zone affiche des événements qui se déroulent parallèlement les uns aux autres, on ne peut pas définir un ordre d'exécution sans une compréhension très approfondie de notre système car il existe une possibilité de chevauchement des intervalles. Les chevauchements des intervalles indiquent qu'il existe plusieurs modes d'exécution possibles mais qu'ils ne sont pas nécessairement corrects. Avec l'insuffisance des contraintes, nous avons laissé cette zone sans condition sur l'ordre d'exécution. Dans cette zone, nous considérons que plusieurs ordres d'arrivée d'événements sont corrects.
- Zone C : il y a l'événement (8).
- Zone D : il y a l'événement (9).
- Zone E : il y a l'événement (10).

Après avoir défini les zones, l'étape suivante consiste à coder tous les paramètres et les spécifications du système : Spécifications concernant l'ordre des événements : [0 ; 1], [2 ; 3 ; 4 ; 5 ; 6 ; 7] [8 ; 9] [8 ; 10]. Spécifications concernant la durée de chaque événement. Spécifications concernant les événements silencieux. Lorsque le système comprend des événements silencieux, nous devons connaître les événements clés qui donnent la forme générale de notre système. Ces

événements sont ceux qui se produisent au début et à la fin de chaque étape, comme le montre la figure 56 :

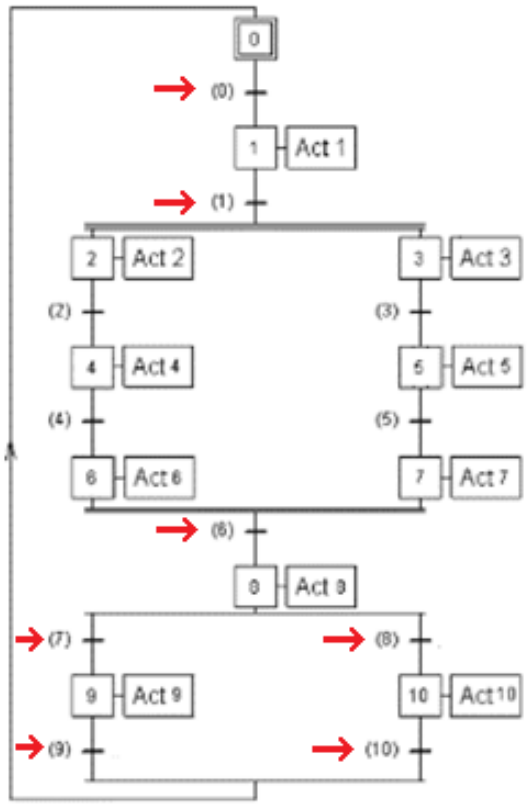


Figure 53 : Présentation de l'emplacement des événements clés

- **Diagnostic :** À ce stade, le système de diagnostic reçoit des informations contenant l'ordre et la date d'occurrence spécifique pour chaque événement observable en fonction des informations reçues dans la première étape. Le traitement comprend les étapes suivantes :

  - Réception de données selon les observations faites avec les dates d'observation.
  - Vérifier que tous les événements non silencieux sont observés.
  - Vérifier que les événements sont exécutés dans l'ordre souhaité, sauf pour le cas du parallélisme (par exemple la zone B dans l'exemple de la Figure 56).
  - Vérifier que chaque étape respecte la marge de temps admissible.

La figure suivante présente le programme qui va nous permettre de diagnostiquer les SED à événements non observable.

```
#include <stdio.h>
main()
{int a,d,b,n,i,j,k,t,e,f,z,TO[1000][1000],TE[1000][1000],TT[100][100];
printf("donner le nombre d'événements:");
scanf("%d",&a);
for(b=0; b<a; b++)
{printf("nom de la variable");
scanf("%d",&TE[b][0]);
printf("durée min du variable");
scanf("%d",&TE[b][1]);
printf("durée max du variable");
scanf("%d",&TE[b][2]);}
printf("donnée le nombre des événements observés");
scanf("%d",&d);
for(b=0;b<d;b++)
{printf("donner le nom de l'événement observé");
scanf("%d",&TO[b][0]);
printf("donner le temps d'observation");
scanf("%d",&TO[b][1]);}
printf("donner le nombre de zones\n");
scanf("%d",&z);
for(i=0; i<z; i++)
{printf("nombre d'événements dans cette zone: %d\n",i);
scanf("%d",&TT[i][0]);}
for(k=0;k<z;k++)
{for(j=1;j<=TT[k][0];j++)
{printf("donner le non d'événement a exécuté le %d:\n",j);
scanf("%d",&TT[k][j]);}}
e=0;
for(b=0;b<a;b++)
{for(j=0;j<d;j++)
{if(TO[j][0]==TE[b][0])
{f=TO[j][1]-e;
if (f>=TE[b][1])
{printf("l'événement %d respecte le temps min \n",TO[j][0]);}
else
{printf("l'événement %d ne respecte le temps min\n",TO[j][0]);}
if (f<=TE[b][2])
{printf("l'événement %d respecte le temps max \n",TO[j][0]);}
else
{printf("l'événement %d ne respecte le temps max\n",TO[j][0]);}
e=TO[j][1];}}}
for(b=0;b<d;b++)
{for(j=0;j<a;j++)
{if(TO[j][0]==TE[b][0])
{gotoAA;}}
printf("l'événement %d n'est pas observable \n",TE[b][0]);
goto BB;
AA:
printf("");
```

```
BB:
printf("");}
for(b=0;b<d;b++)
{for(j=0;j<a;j++)
{if(TE[j][0]==TO[b][0])
{gotoCC;}}
printf("l'événement %d n'est pas attendu \n",TO[b][0]);
CC:
printf("");}
for(i=0;i<z;i++)
{for(j=0;j<d;j++)
{if(TT[i][1]==TO[j][0])
{for(k=2;k<=TT[i][0];k++)
{j=j+1;
if(TT[i][k]==TO[j][0])
{printf("");}
else
{printf("il y a un problème d'ordonnancement");}}
goto DD;}}
DD:
printf("");}
return 0;}
```

Figure 54 : Programme en langage C

- **Sortie :** Les résultats dépendent de la spécification transmise par l'utilisateur final. Cet utilisateur doit spécifier, dans le cas d'une erreur, s'il veut que le système s'arrête, déclencher une alarme ou afficher un message d'erreur qui donne des informations sur l'état de chaque événement dans le système. Cette partie de l'algorithme peut être mieux développée en définissant chaque type d'erreur et en spécifiant la réaction à chaque erreur.

3.3.2 Implémentation du diagnostiqueur

Pour valider notre algorithme, nous considérons le même prototype (traitement de surface). Pour appliquer notre algorithme, nous devons passer par trois étapes : les données, le traitement et les résultats. Si des événements sont inobservables, nous devons connaître au moins les événements clés.

3.3.2.1 Description du procédé

Initialement, le chariot est en position n° 1. L'action Départ du Cycle provoque l'exécution du cycle représenté dans la Figure 58. En utilisant une séquence de récupération pour traiter les opérations de trempage, le drainage au deuxième bac (action 4 : trempage pour 4s, action 6 : drainage de 3s) et au quatrième bac (action 9 : trempage de 4s, action 11 : drainage de 3 s) ainsi

que pour la simulation de la station de déchargement dans le 5ème bac (action 14 : simulation du déchargement pendant 6 s).

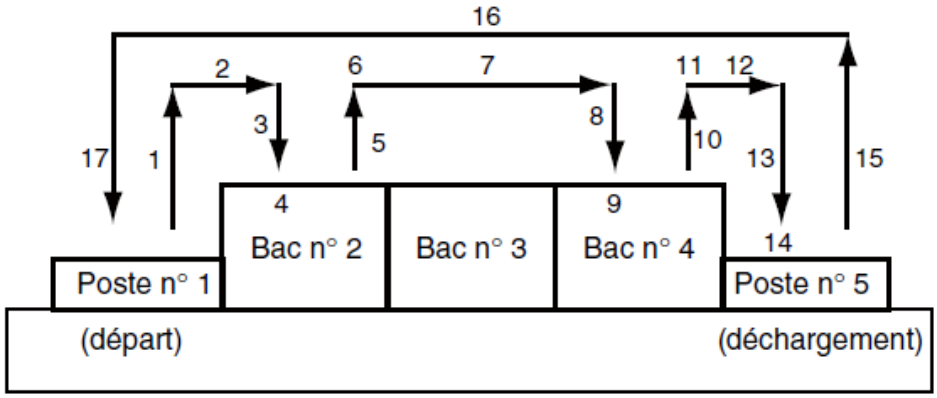


Figure 55 : Déplacements du chariot et du treuil.

Action 4: Trempage bac n° 2 : 4 s

Action 6: Egouttage bac n° 2 : 3 s

Action 9: Trempage bac n° 4 : 4 s

Action 11: Egouttage bac n° 4 : 3 s

Action 14 : Simulation déchargement : 6 s

Conditions initiales : Au départ, le chariot doit être :

- En position basse (capteur D6).
- Au poste n° 1 (capteur D1) l'interrupteur AUTO/MANU doit être sur la position AUTO (S5 = 1).
- Une action sur Départ Cycle (S2) provoque l'exécution du cycle.

Utiliser une reprise de séquence pour traiter les opérations de trempage, égouttage au niveau des bacs n° 2 et n° 4 ainsi que pour la simulation du déchargement au poste n° 5.

3.3.2.2 Modélisation du prototype

Nous appliquerons notre algorithme sur le même traitement de surface avec un modèle GRAFCET simplifié (les actions associées aux étapes 2 à 6, 7 à 11 et 12 à 15 sont identiques) figure suivante :

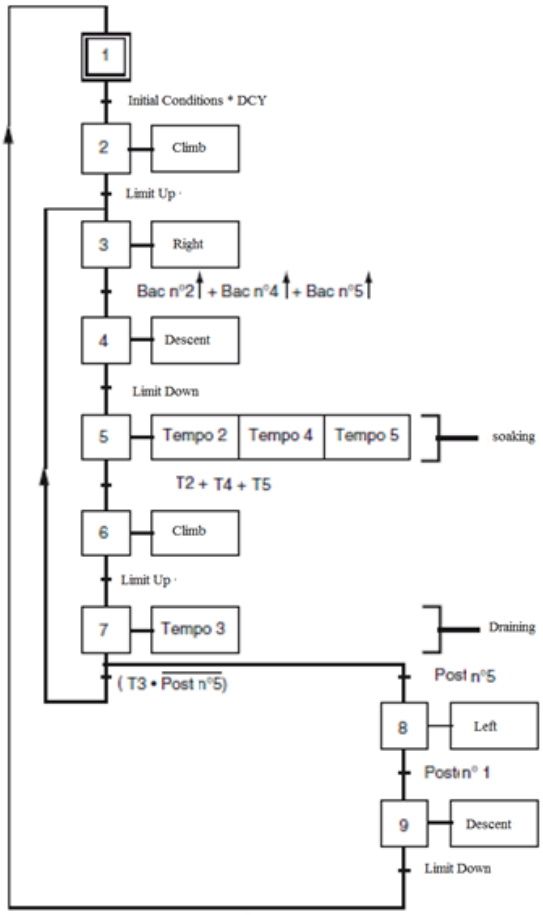


Figure 56 : GRAFCET de notre système

Pour appliquer notre algorithme, nous devons suivre les trois étapes suivantes : modélisation, diagnostic et décision.

Les spécifications temporelles sont détaillées dans le tableau 12 :

Tableau 12 : Les données sur les durées des événements

Actions	1	2	3	4	5	6	7	8
Minimum duration	4	4	4	4	4	3	18	4
Maximum duration	6	10	6	6	6	3	22	6

3.3.2.3 Câblage

Si l'événement est silencieux, ça veut dire que : le système est bien commandé avec tous ces capteurs, mais la partie commande n'est par reliée avec le système de supervision pour n'importe quelle raison. Par exemple : des nouveaux capteurs implantés récemment dans le système et reliés avec la partie commande et non pas avec la partie supervision… Pour illustrer ce cas-là, nous avons utilisé deux automates : Un pour commander API1 et l'autre pour superviser API2. Nous branchons l'API2 à un ordinateur par un connecteur (RS232). Nous fermerons certains interrupteurs reliant entre l'API1 et l'API2 et nous laisserons les autres interrupteurs ouverts comme il présente la figure suivante. Les entrées non branchées de l'API2 seront considérées comme des événements non observables. Par exemple, si nous fermons les interrupteurs et nous laissons S6 et S10 ouverts, ils seront comme des événements non observables.

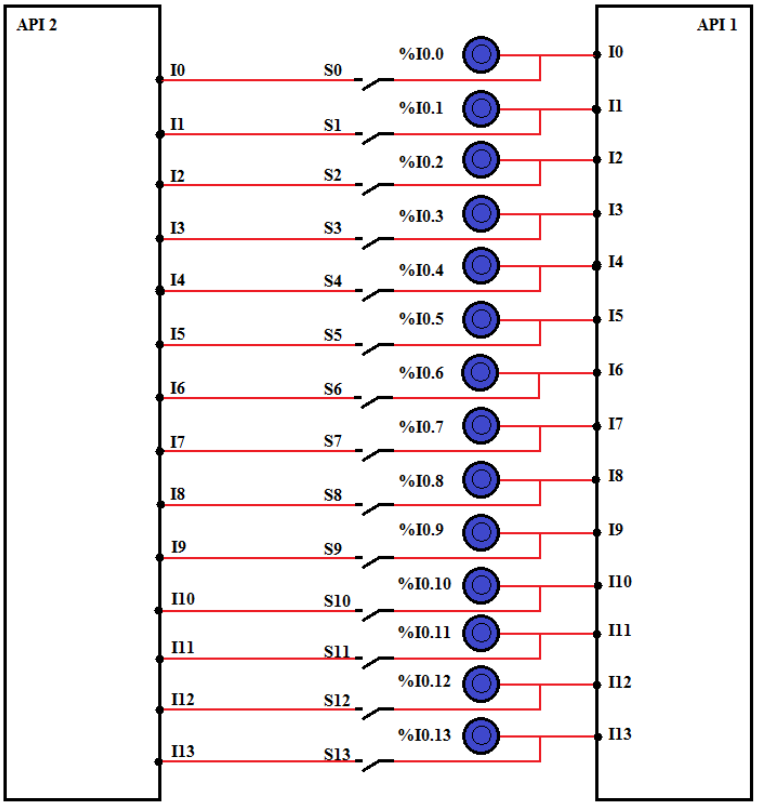


Figure 57 : Schema de câblage de l'API1 avec l'API2

Nous avons câblé le premier pour commander notre système. Pour le deuxième, nous fermerons tous les interrupteurs sauf les commutateurs S7 et S11, ce qui rend les tâches 12 (action 2) et 16 (action 7) silencieux.

3.3.2.3.1 Premier cas de fonctionnement : Première séquence

Diagnostic

Des spécifications temporelles sont détaillées dans le tableau suivant.

Tableau 13 : Première séquence

Actions	1	2	3	4	5	6	2	3	4	5	6	2	3	4	5	7	8
Dates	4	10	15	19	25	28	38	42	46	52	55	-	70	76	81	-	108

Décision

Le diagnostiqueur génère le résultat suivant :

```
L'événement 1 respecte l'heure minime
L'événement 1 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 5 respecte l'heure min
L'événement 5 respecte l'heure maximale
L'événement 6 respecte l'heure minime
L'événement 6 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 5 respecte l'heure min
L'événement 5 respecte l'heure maximale
L'événement 6 respecte l'heure min
L'événement 6 respecte l'heure maximale
L'événement 2 est non observable
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 5 respecte l'heure min
L'événement 5 respecte l'heure maximale
L'événement 7 est non observable
L'événement 8 respecte l'heure minime
L'événement 8 respecte l'heure maximale
```

3.3.2.3.2 Deuxième cas de fonctionnement : Deuxième séquence

Diagnostic

Des spécifications temporelles sont détaillées dans le tableau suivant.

Tableau 14 : Deuxième séquence

Actions	1	2	3	4	5	6	2	3	4	5	6	2	3	4	5	7	8
Dates	4	10	15	19	25	28	30	34	38	44	47	-	62	68	73	-	100

Décision

Le diagnostiqueur génère le résultat suivant :

```
L'événement 1 respecte l'heure minime
L'événement 1 respecte l'heure maximale
L'événement 2 respecte l'heure min
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 5 respecte l'heure min
L'événement 5 respecte l'heure maximale
L'événement 6 respecte l'heure minime
L'événement 6 respecte l'heure maximale
L'événement 2 ne respecte pas l'heure minime
L'événement 2 respecte l'heure maximale
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 5 respecte l'heure min
L'événement 5 respecte l'heure maximale
L'événement 6 respecte l'heure min
L'événement 6 respecte l'heure maximale
L'événement 2 est non observable
L'événement 3 respecte l'heure minime
L'événement 3 respecte l'heure maximale
L'événement 4 respecte l'heure min
L'événement 4 respecte l'heure maximale
L'événement 5 respecte l'heure min
L'événement 5 respecte l'heure maximale
L'événement 7 est non observable
L'événement 8 respecte l'heure minime
L'événement 8 respecte l'heure maximale
```

3.3.3 Synthèse

Nous avons développé un algorithme pour le diagnostic des systèmes d'événements discrets. Ce diagnostiqueur vérifie si les mesures obtenues à partir d'un système à événements discrets temporisé avec une configuration de capteur incomplète sont cohérentes ou non avec un ensemble de contraintes temporelles. Ces contraintes définissent les intervalles de tolérance pour les opérations du système. Notre méthode est basée sur chaque système se produisant dans un intervalle de temps répété et ayant le cycle de fonctionnement normal. Donc, si chaque événement atteint son éventail de temps admissible, notre système répond au temps de fonctionnement normal, ce qui entraîne un bon fonctionnement de l'ensemble du système. À la fin de cet article, nous avons testé notre algorithme. En outre, nous avons obtenu les résultats souhaités puisque nous avons pu détecter tous les types de défauts liés au temps d'exécution. En ce qui concerne la perspective de notre travail, nous considérerons les fonctions de densité de probabilité pendant l'intervalle de temps pour retarder la décision en termes de probabilité.

Conclusion

Dans notre projet, nous avons mis la lumière sur une partie de l'immense littérature dans le diagnostic des SED, en présentant des recherches qui étaient faites dans les méthodes de diagnostic les plus connues. En parallèle, nous avons fait une comparaison entre ces méthodes en nous basant sur les études antérieures dans ce domaine et le travail entamé par notre équipe de recherche.

Cette étude nous a motivés et orientés vers l'axe de recherche qui est le diagnostic des SED en nous basant sur les cycles répétitifs des systèmes qui caractérisent les comportements des SED.

Notre approche, dans laquelle nous avons élaboré un diagnostiqueur qui vérifie que chaque événement se déroule dans la marge de temps acceptable repose sur le déclenchement d'une alarme en cas du surgissement d'un défaut pour mieux le déterminer et le corriger. Cette intervention contribuera à la réduction de la gravité des pannes et leurs conséquences qui sont très coûteuses et critiques sur le rendement au sein des entreprises.

Notre recherche scientifique et technique a été certifiée par l'élaboration d'un algorithme qui est dédié au système séquentiel, il est facile à mettre en œuvre et à fonctionner. Nous l'avons implémenté dans une application VBA qui était appliquée à un prototype. Le feedback de cette expérimentation nous a permis de développer notre algorithme.

L'élargissement et l'approfondissement de ma recherche étaient honorés par un séjour scientifique d'une durée de dix mois au sein du laboratoire GREAH à l'université du Havre en France, suite à l'obtention d'une bourse d'excellence dans le cadre de la mobilité internationale des chercheurs… Au cours de cette période, nous avons développé notre algorithme de base afin de diagnostiquer tous les types de système. Cette amélioration était confirmée par une communication présentée dans un congrès IEEE à Riga. Après nous avons développé notre algorithme afin qu'il puisse prendre en compte les événements silencieux.

Après que nous diagnostiquons un système (machine, robot, chaine de production…), nous devons intervenir, c'est-à-dire le maintenir. Dans la plupart des cas, on n'a pas la possibilité de maintenir tous les éléments du système. D'où la naissance de la discipline de prise de la décision dans la maintenance industrielle. Pour cela, nous avons considéré cette discipline comme une partie de notre axe de recherche. Nos travaux dans la maintenance industrielle sont présentés dans la partie suivante.

Partie C

Maintenance des Systèmes

Introduction

Actuellement, la maintenance des équipements constitue une tâche très importante pour assurer le bon fonctionnement des installations. Des études qui ont été menées récemment sur l'efficacité de la gestion de la maintenance montrent bien que plus d'un tiers des dépenses de l'entreprise proviennent des manœuvres inutiles ou mal exécutées ; Cette inefficacité est la raison principale du manque d'informations réelles qui identifient le besoin immédiat de réparation ou de maintenance.

Les coûts d'entretien représentent souvent la majeure partie des coûts d'exploitation dans un certain nombre d'unités de production. Ces coûts peuvent être considérablement réduits en reconnaissant les décisions les plus appropriées à prendre dans une certaine situation. Le choix de la méthode de gestion de la maintenance influence directement sur le taux de rentabilité et d'efficacité, et il est donc très important de préparer les méthodes et les outils assurant la gestion. Pour cette raison, les décideurs doivent choisir entre plusieurs outils d'aide à la prise de décision pour fournir une solution réalisable et optimale. Dans ce sens, le Pareto [132] [133] [134], est l'un des outils les plus efficaces pour la prise de décision. En fait, il ne se base que sur des calculs mathématiques. Cet outil est en effet une méthode très utile de la classification des équipements les plus critiques.

Dans notre projet de recherche, on s'est amené à élaborer des méthodes de prise de décision afin d'obtenir un meilleur résultat que celui fourni par l'analyse de Pareto. Pour y arriver, on a passé par trois étapes. Premièrement nous avons commencé par l'élaboration d'une nouvelle méthode qui se base sur le même principe de Pareto, mais le tri est fait dans le sens inverse. Deuxièmement nous avons projeté deux méthodes du domaine du transport et logistique : le problème de sac à dos [135] [136] [137] [138] [139] et l'algorithme de Greedy [140] [141] [142] [143] sur notre axe de recherche. Après nous les avons comparés avec la méthode de Pareto. Dans ce contexte, une étude a été faite au sein de l'une des entreprises qui produit des boissons gazeuses, qui nous a permis d'appliquer ces trois méthodes en s'appuyant sur deux variables : les nombres d'heures de pannes et le coût de maintenance pour chaque équipement, dans le but d'améliorer la maintenance industrielle. Après l'identification de différents équipements stratégiques, il sera facile de déceler les différents types d'anomalies et de se concentrer sur les plus critiques. Enfin, en se basant sur ces trois principes, nous avons proposé une nouvelle méthode de prise de décision. Afin d'améliorer la productivité dans l'industrie.

Chapitre 1

Méthodes à un Seul Critère

1.1 Loi de Pareto

Sans hiérarchisation, toute action d'organisation peut s'avérer longue et fastidieuse. En utilisant la loi de Pareto, nous pouvons mettre en évidence les éléments les plus importants criticités afin d'orienter les actions de maintenance. De ce fait, les éléments ayant peu d'influence sur le critère étudié seront éliminés.

La méthode ABC [132] [133] [134] est un outil d'aide à la prise de décision permettant de définir les actions prioritaires. Autrement dit, le diagramme de Pareto fait apparaître les causes les plus importantes qui sont à l'origine du plus grand nombre d'effets.

Les éléments seront classés par ordre d'importance en indiquant les pourcentages pour un critère déterminé. Cette étude nécessite une démarche en trois étapes :

- Définir la nature des éléments à classer : Ces éléments dépendent du critère étudié. Ils peuvent être : des matériels, des causes de pannes, des natures de pannes, des articles en stock, etc ;
- Choisir le critère de classement : Les critères les plus fréquents sont les coûts et les temps, selon le caractère étudié, d'autres critères peuvent être retenus tels que : Le nombre d'accidents, le nombre d'incidents, le nombre de rebuts, le nombre d'heures d'utilisation, le nombre de kilomètres parcourus, la valeur consommée annuellement, souvent nécessaire pour la gestion des stocks etc ;
- Définir les limites de l'étude et classer les éléments.

Le diagramme de Pareto est un graphique à colonnes qui présente les informations par ordre décroissant et fait ainsi ressortir le ou les éléments les plus importants qui expliquent un phénomène ou une situation. Généralement, 20 % du nombre des éléments représentent 80 % du critère étudié : c'est la classe A, les 30 % suivant du nombre des éléments représentent 15% du critère étudié : c'est la classe B et les 50 % restants du nombre des éléments représentent seulement 5 % du critère étudié : c'est la classe C. En cumulant les valeurs décroissantes du critère étudié, la courbe ABC fait apparaître trois zones d'où l'appellation de "courbe ABC". Voir figure 61.

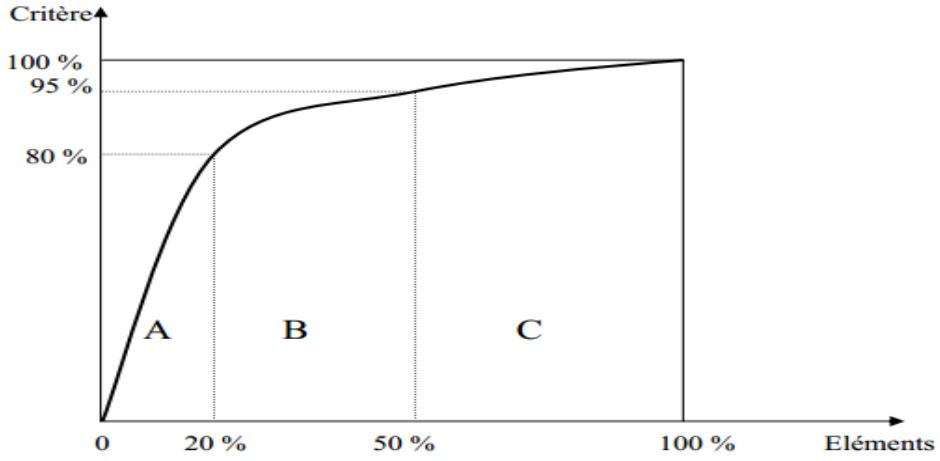


Figure 58 : Diagramme de Pareto (ABC).

Après avoir appliqué cette méthode dans plusieurs études de cas, nous avons remarqué que nous n'avons pas trouvé de résultats cohérents par rapport à la loi 80/20, représentée à la figure 61. En conséquence, nous devons soit modifier la méthode de Pareto, soit utiliser une autre méthode qui nous aidera à faire les décisions qui donnent les meilleurs résultats et les plus cohérents. Une nouvelle méthode de classification est présentée dans la section suivante.

1.2 Diagramme de CBA

Notre objectif de recherche est de prendre la meilleure décision afin d'améliorer la productivité d'une entreprise. La méthode de Pareto est parmi les méthodes les plus connues dans la prise de décision, mais cette méthode ne donne pas toujours le résultat 80/20 attendu ; Par conséquent, il est nécessaire de l'améliorer ou de trouver une nouvelle façon d'avoir une décision plus optimale. Pour le faire nous avons proposé un algorithme afin de réaliser notre objectif qu'on va l'appeler diagramme CBA [144].

Pour la mise en œuvre de ce digramme de CBA, nous devrons suivre les étapes suivantes :

1) Déterminer le problème à résoudre ;
2) Collecter des données ou utiliser des données déjà existantes ;
3) Classer les données en catégories et prévoir une catégorie "Divers" pour les catégories à peu d'éléments ;
4) Calculer le total des données de chaque catégorie et déterminer les pourcentages par rapport au total ;
5) Classer ces pourcentages par ordre croissant ;
6) Calculer le pourcentage cumulé ;

7) Déterminer une échelle adaptée pour tracer le graphique ;
8) Placer les colonnes (les barres) sur le graphique, en commençant par la plus petite à gauche ;
9) Tracer la courbe des pourcentages cumulés ;

Pour évaluer l'efficacité de notre méthode, nous allons l'appliquer et le comparer avec la méthode de Pareto.

1.3 Etude comparative

On va considérer qu'une entreprise qui a 15 machines qui tombent en panne, chacune de ces machines a une durée d'arrêt et un montant nécessaire pour l'entretien. Le tableau suivant illustre les informations en détail pour chaque machine.

Tableau 15 : Le coût de la maintenance et des temps d'arrêt pour chaque machine

Référence	Temps d'arrêt	Coûts d'entretien en MAD
1	5	4000
2	15	15000
3	2	2000
4	20	22000
5	4	2100
6	4	1300
7	4	2500
8	1 ,5	850
9	4	1800
10	2	850
11	2	2500
12	10	10000
13	2	1500
14	4	2000
15	1	700

1.3.1 Résolution par la méthode de Pareto

La méthode de Pareto nous permet de classer les machines par ordre décroissant selon la gravité de leurs problèmes, elle peut être calculée en utilisant la formule suivante : (temps d'arrêt de la machine / temps d'arrêt total) * 100.

Le tableau ci-dessous présente le pourcentage du temps d'arrêt pour chaque machine :

Tableau 16 : Pourcentage de la répartition de chaque machine de la ligne 2

Référence	Coûts de réparation	Pourcentage Temps d'arrêt	Coût cumulé	Temps d'arrêt cumulé %
4	22000	24,84	22000	24,84
2	15000	18,63	37000	43,47
12	10000	12,42	47000	55,90
1	4000	6,21	51000	62,11
5	2100	4,96	53100	67,08
6	1300	4,96	54400	72,04
7	2500	4,96	56900	77,01
9	1800	4,96	58700	81,98
14	2000	4,96	60700	86,95
3	2000	2,48	62700	89,44
10	850	2,48	63550	91,92
11	2500	2,48	66050	94,40
13	1500	2,48	67550	96,89
8	850	1,86	68400	98,75
15	700	1,24	69100	100

La figure ci-dessous présente le pourcentage cumul du temps d'arrêt pour chaque machine :

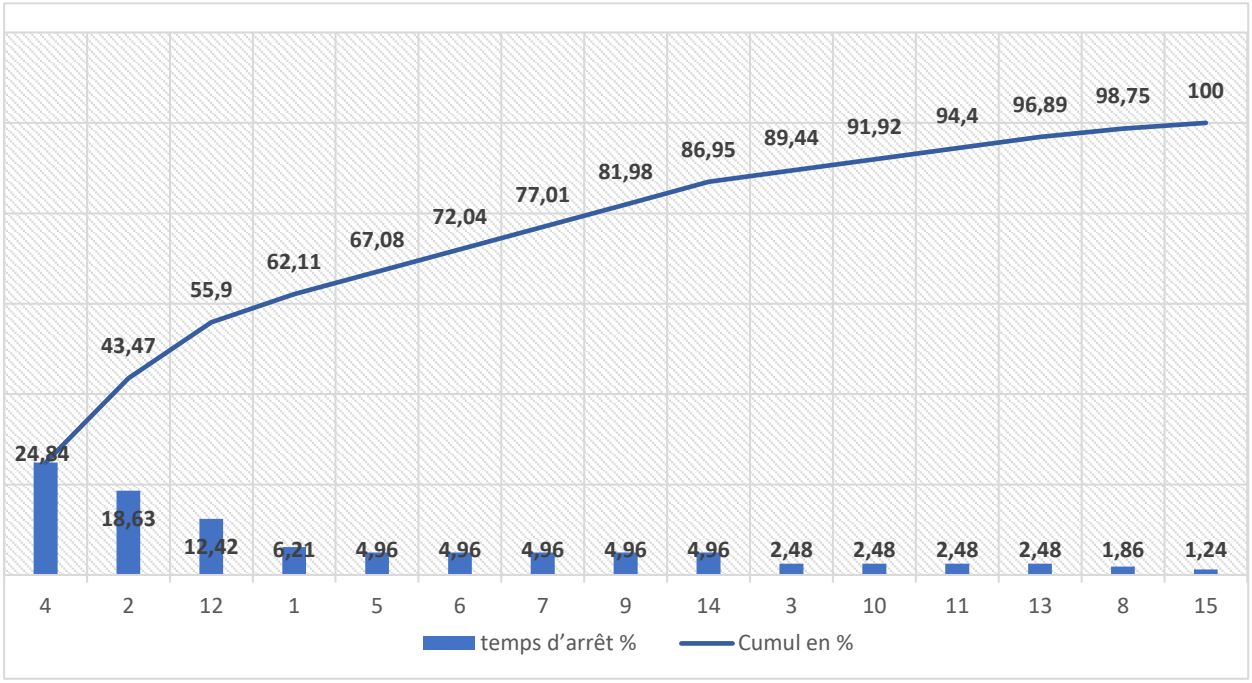


Figure 59 : Graphe du pourcentage de la répartition de chaque machine de la ligne 2

Selon le graphique de Pareto, nous constatons que 81,98% des problèmes d'arrêt sont dus à l'arrêt des machines : 4, 2, 12, 1, 5, 6, 7 et 9. Les durées de panne de ces machines occupent une partie importante du temps de travail et donc contribuent au ralentissement et/ou à l'arrêt de la production.

Dans la plupart du temps, on n'a pas un budget suffisant pour entretenir toutes les machines, pour cela nous avons besoin d'une méthode qui nous permet de classer les machines selon l'ordre de priorité comme la méthode de Pareto.

Dans notre cas, on va considérer que l'administration nous a fourni un budget de 40 000,00 MAD. Avec le classement fourni par la méthode de Pareto et sous la contrainte du budget, nous pouvons résoudre les problèmes concernant les machines 4 et 2.

La maintenance des machines 4 et 2 coute 37 000,00 MAD. Après cette intervention le temps d'arrêt sera minimisé par 43,47%.

1.3.2 Résolution par la méthode CBA

La méthode CBA nous permet de classer les machines dans l'ordre décroissant en raison de la gravité de leurs problèmes ; la gravité peut être calculée par la même formule citée avant : (temps d'arrêt de la machine / temps d'arrêt total) * 100.

Les résultats trouvés après l'application de la méthode CBA sont illustrés dans le tableau et la figure suivante :

Tableau 17 : Les calculs avec la méthode CBA

Référence	Coûts de réparation	Pourcentage du temps d'arrêt	Coût cumulé	Temps d'arrêt cumulé %
15	700	1,24	700	1,24
8	850	1,86	1550	3,10
3	2000	2,48	3550	5,59
10	850	2,48	4400	8,07
11	2500	2,48	6900	10,55
13	1500	2,48	8400	13,04
5	2100	4,96	10500	18,01
6	1300	4,96	11800	22,98
7	2500	4,96	14300	27,95
9	1800	4,96	16100	32,91
14	2000	4,96	18100	37,88
1	4000	6,21	22100	44,09
12	10000	12,42	32100	56,52
2	15000	18,63	47100	75,15
4	22000	24,84	69100	100

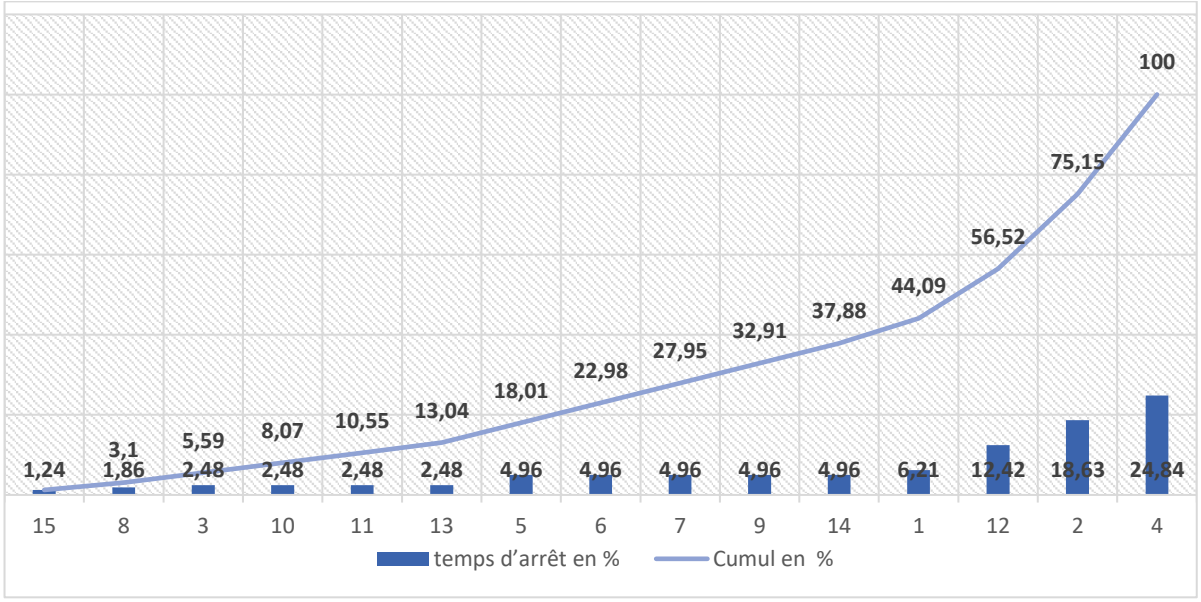


Figure 60 : Graphe du pourcentage de la répartition de chaque machine de la ligne 2

Pour comparer les deux méthodes, nous devons travailler sous la même contrainte. Il s'agit du budget qui est limité à 40 000,00 MAD. Avec notre méthode, nous pouvons résoudre les problèmes concernant les machines 15, 8, 3, 10, 11, 13, 5, 6, 7, 9, 14, 1 et 12.

La maintenance des machines 15, 8, 3, 10, 11, 13, 5, 6, 7, 9, 14, 1 et 12 coute 32 100,00 MAD. Après cette intervention le temps d'arrêt sera minimisé par 56, 52%.

1.3.3 Synthèse

Les résultats de l'application de l'algorithme ABC et l'algorithme de CBA sur l'exemple précédent sont résumés dans le tableau suivant :

Tableau 18 : Le pourcentage du temps d'arrêt remis en état et les coûts de réparation en MAD pour les deux méthodes

	% de temps d'arrêt minimisés	**Coût de réparation en MAD**
Méthode ABC	43,47	37 000
Méthode CBA	56.52	32 100

D'après ce tableau, nous remarquons que la méthode CBA nous a permis d'économiser 13,05% dans le temps d'arrêt par rapport à la méthode ABC et 4 900,00 MAD.

On ne peut pas se baser sur une seule étude pour dire que la méthode CBA est mieux que la méthode ABC. Pour cela nous les avons testés sur plusieurs exemples et nous avons trouvé les trois cas suivants :

Cas 1 : Le résultat de la méthode CBA est mieux que le résultat de la méthode ABC

Cas 2 : Le résultat de la méthode ABC est meilleur que le résultat de la méthode CBA.

Cas 3 : Les résultats de la méthode ABC et les résultats de la méthode CBA sont identiques.

Ces trois cas présentent toutes les possibilités susceptibles, pour cela nous avons proposé d'appliquer les deux méthodes pour les comparer et de choisir celle qui donne le meilleur résultat. Cette proposition est schématisée dans l'algorithme présenté dans la figure suivante :

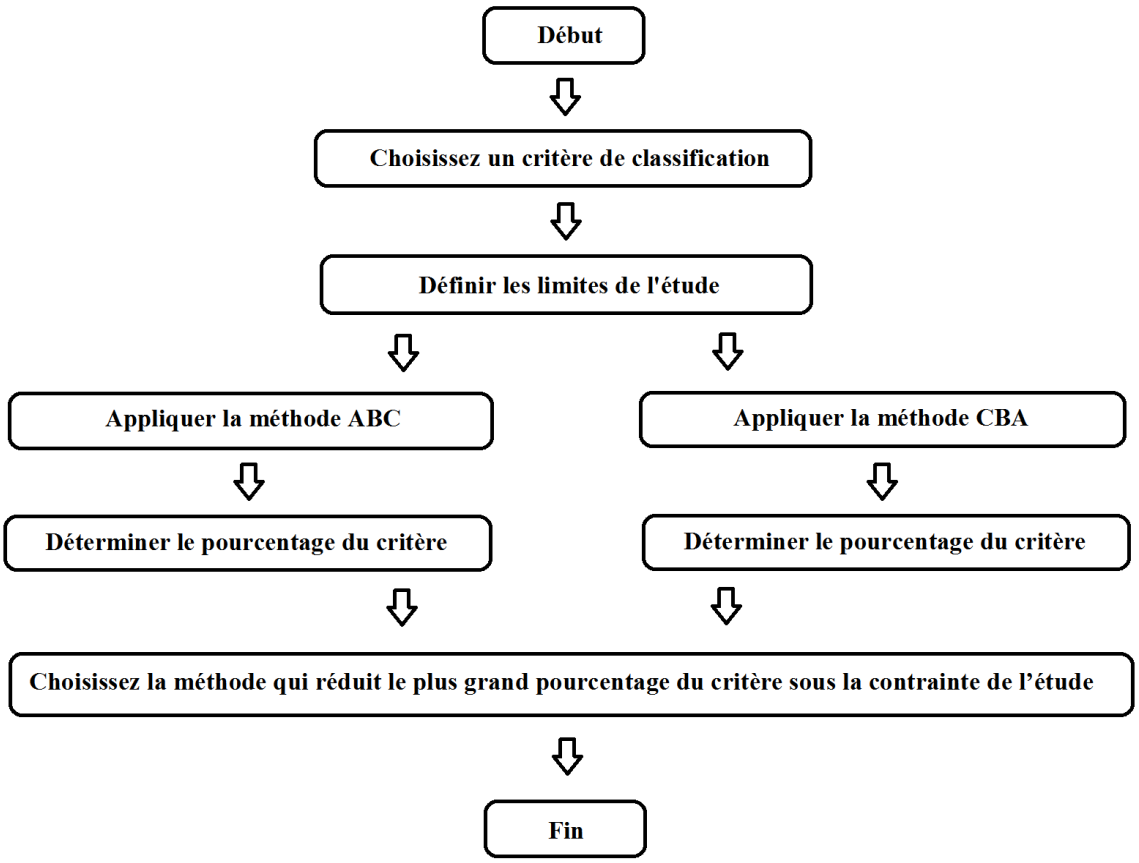


Figure 61 : Algorithme pour choisir celui qui donne le meilleur résultat

Cet algorithme fait la comparaison entre le résultat de la méthode ABC et le résultat de notre méthode CBA sous la contrainte exigée par cette étude afin de choisir la méthode qui donne la solution la plus optimale pour réduire le taux de défaillance et augmenter la productivité.

Il existe de nombreuses méthodes et des algorithmes de prise de décision. Dans notre étude, nous nous sommes intéressés à la méthode de Pareto et nous avons élaboré la méthode CBA. Le choix entre ces deux méthodes se fait sous une deuxième contrainte. Là nous avons pensé à chercher des méthodes qui ne se basent que sur des calculs mathématiques et prendre en considération deux contraintes.

Chapitre 2

Méthodes à Deux Critères

Après avoir parcouru plusieurs méthodes dans la science de la prise de décision nous avons trouvé deux méthodes : problème de sac-à-dos (knapsack Problem) et algorithme de Greedy. Elles sont destinées au domaine du transport et logistique et nous les avons appliquées au domaine de la maintenance industrielle.

2.1 La méthode de sac à dos

Le problème du sac à dos, noté également KP (Knapsack Problem) est un problème d'optimisation combinatoire [135] [136] [137] [138] [139]. Il modélise une situation analogue au remplissage d'un sac à dos, ne pouvant supporter plus d'un certain poids, avec tout ou partie d'un ensemble donné d'objets ayant chacun un poids et une valeur. Les objets mis dans le sac à dos doivent maximiser la valeur totale, sans dépasser le poids maximum. Les données du problème peuvent être exprimées en termes mathématiques. Les objets sont numérotés par l'indice « i » variant de 1 à n. Les nombres « ω_i » et « P_i » représentent respectivement le poids et la valeur de l'objet « i ». La capacité du sac sera notée W.

Il existe multiples façons de remplir le sac à dos. Pour décrire l'une d'elles, il faut indiquer pour chaque élément s'il est pris ou non. On peut utiliser un codage binaire : l'état de l'i-ème élément vaudra $x_i = 1$ si l'élément est mis dans le sac, ou $x_i = 0$ s'il est laissé de côté. Une façon de remplir le sac est donc complètement décrite par un vecteur appelé vecteur contenu, ou simplement contenu : $X=(x_1, x_2, \ldots, x_n)$; et le poids associé ainsi que la valeur associée à ce remplissage peuvent alors être exprimés comme fonction du vecteur contenu.

Pour un contenu X donné, la valeur totale contenue dans le sac est naturellement :

$$Z(X) = \sum_{\{i,xi=1\}} pi = \sum_{i=1}^{n} xipi$$

De même, la somme des poids des objets choisis est :

$$\omega(X) = \sum_{\{i,xi=1\}} \omega i = \sum_{i=1}^{n} xi\omega i$$

Le problème peut alors être reformulé comme la recherche d'un vecteur contenu.

X=(x_1, x_2,…., x_n) (les composantes valant 0 ou 1), réalisant le maximum de la fonction valeur totale Z(X) sous la contrainte suivante :

$$\omega(X) = \sum_{i=1}^{n} xi\omega i \leq W$$

C'est-à-dire que la somme des poids des objets choisis ne dépasse pas la capacité du sac à dos.
En général, on ajoute les contraintes suivantes afin d'éviter les cas singuliers :

- $\sum_{i=1}^{n} xi\omega i > W$: On ne peut pas mettre tous les objets.
- $\omega i \leq W, \forall i \in \{1, \dots, n\}$: aucun objet n'est plus lourd que ce que le sac peut porter.
- $pi > 0, \forall i \in \{1, \dots, n\}$: tout objet a une valeur et apporte un gain.
- $\omega i > 0, \forall i \in \{1, \dots, n\}$: tout objet a un certain poids et consomme des ressources.

Terminologie :
Z(X) : est appelée fonction objectif ;

- Tout vecteur X vérifiant la contrainte (1) est dit réalisable.
- Si la valeur de Z(X) est maximale, alors X est dite optimale.

2.2 Algorithme de Greedy

En ce qui concerne la plupart des problèmes de décision, il peut être suffisant de trouver des solutions professionnelles applicables même si elles ne sont pas optimales. De préférence, l'approximation vient avec une garantie sur la différence entre la valeur de la solution trouvée et la valeur de la solution optimale.

La terminologie adoptée est « Efficacité d'un objet » qui est le rapport de sa valeur à son poids. Si la valeur de l'objet est grande par rapport à ce qu'elle consomme, l'objet est plus efficace. L'idée de l'algorithme Greedy est de choisir les objets en fonction de leurs efficacités [140] [141] [142] [143]. L'algorithme de Greedy est dans la figure suivante :

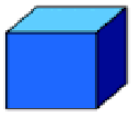
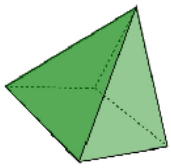
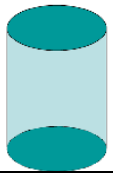
```
trier les objets par ordre décroissant d'efficacité
w_conso := 0
pour i de 1 à n
si w[i] + w_conso ≤ W alors
x[i] := 1
w_conso := w_conso + w[i]
sinon
x[i] := 0
fin si
fin pour
```

Figure 62 : L'algorithme Greedy

2.3 Exemple explicatif

On considère que nous avons cinq objets, chacun a sa valeur et son poids, comme il est illustré dans le tableau suivant :

Tableau 19 : Les valeurs des objets

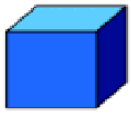
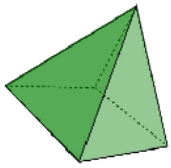
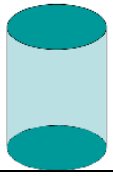
Les objets :	1 	2 	3 	4 	5 
Les poids	4Kg	14Kg	24Kg	10Kg	18Kg
Les valeurs	2MAD	6MAD	14MAD	4MAD	20MAD

Le poids total de ces objets est 70kg, mais nous n'avons le droit de transporter que 30 kg. Pour choisir parmi ces objets, nous allons appliquer les deux méthodes, problème de sac-à-dos et l'algorithme de Greedy, afin de les biens illustrés.

2.3.1 Application du problème de sac-à-dos :

En appliquant la méthode de sac-à-dos nous obtenons les résultats suivants :

Tableau 20 : Application du problème de sac-à-dos

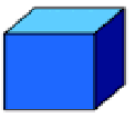
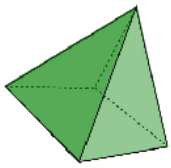
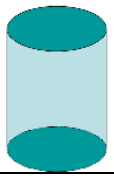
Les objets :	1 	2 	3 	4 	5 
Les poids	4Kg	14Kg	24Kg	10Kg	18Kg
Les valeurs	2MAD	6MAD	14MAD	4MAD	20MAd
Xi	1	1	0	1	0

Selon la méthode nous allons prendre avec nous les objets 1, 2 et 4. Qui ont un poids de 28kg avec une valeur de 12MAD.

2.3.2 Application de l'algorithme de Greedy :

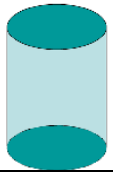
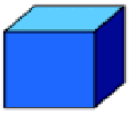
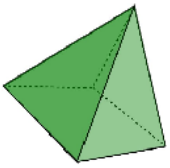
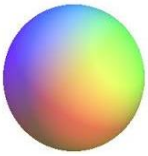
Pour choisir les objets avec l'algorithme de Greedy, nous devons calculer l'efficacité de chaque objet.

Tableau 21 : Application de l'algorithme de Greedy

Les objets :	1 	2 	3 	4 	5 
Les poids	4Kg	14Kg	24Kg	10Kg	18Kg
Les valeurs	2MAD	6MAD	14MAD	4MAD	20MAd
Les efficacités	0,50	0,43	0,58	0,40	1,11

Après, nous classons les objets suivant l'ordre de décroissance de l'efficacité des objets. Comme il est présenté dans ce tableau :

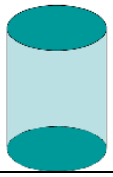
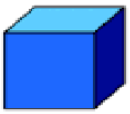
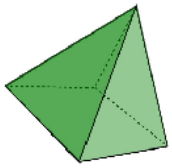
Tableau 22 : l'ordre des objets suivant l'efficacité

Les objets :	5 	3 	1 	2 	4 
Les poids	18Kg	24Kg	4Kg	14Kg	10Kg
Les valeurs	20MAd	14MAD	2MAD	6MAD	4MAD
Les efficacités	1,11	0,58	0,50	0,43	0,40

En appliquant la méthode, nous allons prendre avec nous l'objet 5. Qui a un poids de 18kg et une valeur de 20MAD.

Dans la plupart des études, nous avons trouvé que la méthode du sac-à-dos suit toujours l'algorithme de Greedy. En les combinant, nous trouvons le résultat suivant :

Tableau 23 : résolution avec méthode du sac-à-dos et l'algorithme de Greedy

Les objets :	5 	3 	1 	2 	4 
Les poids	18Kg	24Kg	4Kg	14Kg	10Kg
Les valeurs	20MAd	14MAD	2MAD	6MAD	4MAD
Les efficacités	1,11	0,58	0,50	0,43	0,40
Xi	1	0	1	0	0

Avec les deux méthodes, nous allons amener avec nous l'objet 5 et l'objet 1 qui ont un poids de 22kg avec une valeur de 22MAD.

2.4 Etude de cas

Toutes les entreprises ont un but lucratif, c'est-à-dire « produire plus » et par conséquent le temps de panne doit être minimisé ; pour cela les entreprises réservent des budgets alloués à l'amélioration de leurs productivités. Dans notre projet de recherche, nous nous intéressons à un cas d'étude d'une entreprise de conditionnement de boissons gazeuses. Dans un premier temps, nous commençons par une étude utilisant l'outil d'analyse de problème simple « Pareto », puis on va intégrer le problème de sac à dos à l'outil Pareto et enfin, on applique l'algorithme de Greedy avec le problème de sac à dos.

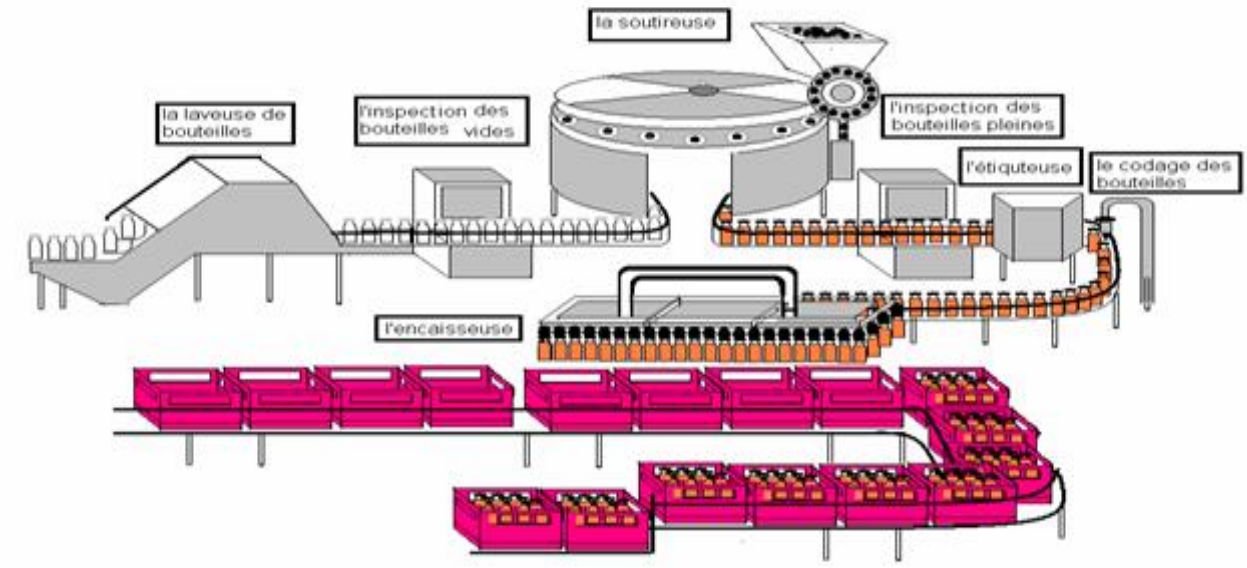


Figure 63 : Schéma illustratif des différentes étapes de la préparation des boissons gazeuses

Pour améliorer le rendement d'une ligne de production (appelé ligne 2) et maintenir la majorité de ses machines en bon état lors de la production figure 62, un budget de 300.000,00MAD est proposé. Pour ce faire on va étudier le temps d'arrêt et le coût de la maintenance pendant 2 mois pour chaque machine de la ligne 2, donné dans le tableau suivant :

Tableau 24 : Le coût de la maintenance et le temps d'arrêt pour chaque Machine de la ligne 2

Les éléments de la ligne 2	Temps d'arrêt (h)	Coût de maintenance en MAD
SOUTIREUSE O+H LV2	7,24	72400
VISSEUSE LV2	6,73	67300
CONVOYEUR BOUTEILLES LV2	6,02	60400
ENCAISSEUSE KETTNER LV2	5,59	56400
CAPSULEUSE LV2	4,47	44300
CONVOYEUR CASIERS LV2	4,14	41900
PALETISEUR LV2	3,73	18000
LAVEUSE BOUTEILLES O+H LV2	3,3	33100
ETIQUETEUSE KRONES LV2	2,41	24200
DECAISSEUSE KETTNER LV2	2,4	24300
DEPALETISEUR LV2	1,51	15500
INSPECTRICE LV2	1,12	11900
DATEUSE LV2	1,03	10900
MIXEUR LV2	0,26	3300
Total	49,95	483900

2.4.1 Résolution selon la méthode de Pareto

La méthode de Pareto consiste à classer les machines par ordre de gravité qui se calcule par (le temps d'arrêt de la machine/ le temps total d'arrêt)*100.

Le tableau ci-dessous représente le pourcentage d'arrêt pour chaque machine de la ligne 2 :

Tableau 25 : Résolution selon la méthode de Pareto

Machine	temps d'arrêt (h)	cumul	% d'arrêt	% cumul
SOUTIREUSE O+H LV2	7,24	7,24	14,49	14,49
VISSEUSE LV2	6,73	13,97	13,48	27,97
CONVOYEUR BOUTEILLES LV2	6,02	19,99	12,05	40,02
ENCAISSEUSE KETTNER LV2	5,59	25,58	11,19	51,21
CAPSULEUSE LV2	4,47	30,05	8,96	60,17
CONVOYEUR CASIERS LV2	4,14	34,19	8,29	68,46
PALETISEUR LV2	3,73	37,92	7,47	75,93
LAVEUSE BOUTEILLES O+H LV2	3,3	41,22	6,61	82,54
ETIQUETEUSE KRONES LV2	2,41	43,63	4,82	87,36
DECAISSEUSE KETTNER LV2	2,4	46,03	4,80	92,16
DEPALETISEUR LV2	1,51	47,54	3,02	95,18
INSPECTRICE LV2	1,12	48,66	2,24	97,42
DATEUSE LV2	1,03	49,69	2,06	99,48
MIXEUR LV2	0,26	49,95	0,52	100,00

D'après le diagramme de Pareto nous constatons que 82,52% des problèmes d'arrêt de la ligne 2 sont dus aux arrêts de la soutireuse, la visseuse, le convoyeur bouteilles, l'encaisseuse, la capsuleuse, le convoyeur casiers, le palettiseur et la laveuse bouteille, provoquant des temps d'arrêts prenant une part assez importante de temps de travail et par conséquent l'arrêt de la production.

Pour le budget alloué (300 000.00MAD), on constate qu'avec la méthode Pareto on peut remédier aux problèmes des machines suivantes :

- SOUTIREUSE O+H LV2
- VISSEUSE LV2
- CONVOYEUR BOUTEILLES LV2
- ENCAISSEUSE KETTNER LV2

Cette solution nous permet de minimiser jusqu'à 51,2% du temps d'arrêt avec un montant de 258 500 .00 MAD. Pour bien exploiter la somme restante, nous allons utiliser la méthode de sac à dos.

La méthode de sac à dos consiste à mettre les objets dans un sac sans dépasser sa capacité, jusqu'à la saturation du sac à dos, si l'objet est mis dans le sac on aura $x_i=1$, sinon $x_i=0$.

L’application de la méthode de sac à dos sur les résultats donnés par la méthode Pareto mène à une nouvelle solution représentée dans le tableau suivant :

Tableau 26 : Résultat de l’application du sac à dos sur les résultats obtenus par Pareto.

Machine	%d’arrêt	Coût de maintenance en MAD	xi
SOUTIREUSE O+H LV2	14,49	72400	1
VISSEUSE LV2	13,48	67300	1
CONVOYEUR BOUTEILLES LV2	12,05	60400	1
ENCAISSEUSE KETTNER LV2	11,19	56400	1
CAPSULEUSE LV2	8,96	44300	0
CONVOYEUR CASIERS LV2	8,29	41900	1
PALETISEUR LV2	7,47	18000	0
LAVEUSE BOUTEILLES O+H LV2	6,61	33100	0
ETIQUETEUSE KRONES LV2	4,82	24200	0
DECAISSEUSE KETTNER LV2	4,80	24300	0
DEPALETISEUR LV2	3,02	15500	0
INSPECTRICE LV2	2,24	11900	0
DATEUSE LV2	2,06	10900	0
MIXEUR LV2	0,52	3300	0

D’après les résultats de sac à dos appliqués au Pareto, on remarque qu’on peut remédier aux problèmes des machines suivantes :

- SOUTIREUSE O+H LV2
- VISSEUSE LV2
- CONVOYEUR BOUTEILLES LV2
- ENCAISSEUSE KETTNER LV2
- CONVOYEUR CASIERS LV2

Cette solution nous permet de minimiser jusqu’à 59,49% du temps d’arrêt avec un montant de 298 400 .00 MAD.

2.4.2 Résolution par l’algorithme de Greedy

L’algorithme de Greedy fait le classement des objets par l’efficacité, ce dernier se calcule par la division du coût de maintenance sur le temps d’arrêt. Le choix des machines à corriger se

fait par la méthode de remplissage du sac à dos ; l'application de cet algorithme sur notre étude de cas donne les résultats qui sont illustrés dans le tableau suivant :

Tableau 27 : Résultat de l'application de l'algorithme de Greedy

Machine	temps d'arrêt (h)	coût de maintenance en MAD	Efficacité (MAD/h)	xi
MIXEUR LV2	0,26	3300	12690	1
INSPECTRICE LV2	1,12	11900	10630	1
DATEUSE LV2	1,03	10900	10580	1
DEPALETISEUR LV2	1,51	15500	10260	1
DECAISSEUSE KETTNER LV2	2,4	24300	10130	1
CONVOYEUR CASIERS LV2	4,14	41900	10120	1
ENCAISSEUSE KETTNER LV2	5,59	56400	10090	1
ETIQUETEUSE KRONES LV2	2,41	24200	10040	1
CONVOYEUR BOUTEILLES LV2	6,02	60400	10030	1
LAVEUSE BOUTEILLES O+H LV2	3,3	33100	10030	1
SOUTIREUSE O+H LV2	7,24	72400	10000	0
VISSEUSE LV2	6,73	67300	10000	0
CAPSULEUSE LV2	4,47	44300	9910	0
PALETISEUR LV2	3,73	18000	4830	1

D'après l'application de l'algorithme de Greedy, on constate qu'on peut remédier aux problèmes des machines suivantes :

- MIXEUR LV2 ;
- INSPECTRICE LV2 ;
- DATEUSE LV2 ;
- DEPALETISEUR LV2 ;
- DECAISSEUSE KETTNER LV2 ;
- CONVOYEUR CASIERS LV2 ;
- ENCAISSEUSE KETTNER LV2 ;
- ETIQUETEUSE KRONES LV2 ;
- CONVOYEUR BOUTEILLES LV2
- LAVEUSE BOUTEILLES O+H LV2
- PALETTISEUR LV2

Cette solution nous permet de minimiser jusqu'à 63,07% du temps d'arrêt avec un montant de 299 900 .00 MAD.

2.4.3 Comparaison

Les résultats des trois outils : Pareto, Pareto & sac à dos et algorithme de Greedy & sac à dos sont représentés dans le tableau 28 [145].

Tableau 28 : Résultats des trois outils : Pareto, Pareto & sac à dos et algorithme de Greedy & sac à dos

	% Temps d'arrêt remédié	**Coût d'intervention en MAD**	**Valeur remédié en 2 mois MAD**
Pareto	51.20	258500	92160
Pareto avec problème de sac à dos	59.49	298400	107082
Algorithme de Greedy avec problème de sac à dos	63.07	299900	113526

Le pourcentage du temps d'arrêt remédié de sac à dos & Pareto par rapport au Pareto ; d'Algorithme de Greedy par rapport au Pareto et sac à dos & Pareto est représenté dans ce tableau.

Tableau 29 : Le pourcentage du temps d'arrêt remédié de chaque outil par rapport aux autres

	Pareto	**Pareto avec problème de sac à dos**	**Algorithme de Greedy avec problème de sac à dos**
Pareto avec problème de sac à dos	8.29%	-	-
Algorithme de Greedy avec problème de sac à dos	11.87%	3.58%	-

2.4.4 Synthèse

Dans cette étude nous avons trouvé que l'Algorithme de Greedy (AG) donne un résultat plus optimal que celle du problème de sac à dos Knapsack Problem (KP) et ce dernier donne un résultat plus optimal que la méthode de Pareto (P).

Résultat de notre étude de cas : AG>KP>P

Mais après plusieurs essais sur les différents cas, on a trouvé les résultats suivants :

Cas 1 : AG>KP>P

Cas 2 : AG=KP>P

Cas 3 : AG>KP=P

Cas 4 : AG=KP=P

Alors ces résultats peuvent être généralisés suivant l'équation suivante :

$$AG \geq KP \geq P$$

Dans notre étude nous avons montré que l'algorithme de Greedy qui est destiné à l'ordonnancement donne des meilleurs résultats dans la maintenance industrielle. L'utilisation de cette méthode va permettre d'améliorer la productivité et la maintenabilité dans l'industrie surtout dans la production, ainsi il va optimiser les dépenses des budgets destinées à l'amélioration des chaines de production.

Il existe de nombreuses méthodes et des algorithmes de prise de décision. Dans notre étude, nous nous sommes intéressés aux méthodes suivantes : Pareto, knapsack et Greedy algorithme. Notre étude de cas montre que la méthode Pareto permet de choisir les éléments les plus critiques basés sur un seul critère. L'algorithme est utilisé pour sélectionner les éléments les plus efficaces basés sur deux variables. Ajouter la méthode du sac à dos pour remédier à plus d'éléments ; donc exploiter les ressources maximales (dans notre étude de cas, exploitant le budget fourni par la direction).

Pour améliorer une ligne de production, nous détectons de critère disponible. Si il égal à 2, nous proposons d'utiliser l'algorithme Greedy et après appliquer la méthode du sac à dos. Sinon, nous proposons d'utiliser la méthode de ABC/CBA et après appliquer la méthode du sac à dos pour obtenir des résultats plus optimaux. Mais la question qui se pose est : est-il possible de combiner les trois outils d'un algorithme pour obtenir des résultats encore meilleurs ?

Pour répondre à cette question, nous allons proposer une méthodologie simple et efficace afin de sélectionner les éléments les plus fiables pour une solution optimale. On se basant sur le Pareto, le Knapsack Problem et l'algorithme Greedy.

2.5 Algorithme PKPGA

Notre méthode se base sur les trois méthodes Knapsack Problem, Pareto et l'Algorithme Greedy, pour cela nous l'avons appelée méthode de PKPGA [146].

2.5.1 Algorithme

L'implémentation de cette méthode se fait selon les étapes suivantes :

- Calculer le pourcentage du cumul de gravité (temps d'arrêt) de chaque machine.
- Calculer la valeur d'efficacité pour chaque élément selon la formule suivante : $\frac{\text{Pourcentage de gravité}}{\text{Valeur d'intervention}}$.
- Choisir et trier les éléments récoltés par ordre décroissant selon leurs efficacités via la méthode de sac à dos.

Pour tester l'efficacité de notre méthode, nous allons appliquer sur la même étude de cas afin de l'approuver.

2.5.2 Application

L'application de notre algorithme sur la chaine de production donne les résultats suivants :

Tableau 30 : Application du problème de sac à dos au problème Pareto

Machine	% d'arrêt	% de coût de maintenance	valeur d'efficacité	xi
PALETISEUR LV2	7,47	3,72	2,01	1
CAPSULEUSE LV2	8,96	9,15	0,98	1
VISSEUSE LV2	13,47	13,91	0,97	1
SOUTIREUSE O+H LV2	14,49	14,96	0,97	1
LAVEUSE BOUTEILLES O+H LV2	6,61	6,84	0,97	1
CONVOYEUR BOUTEILLES LV2	12,05	12,48	0,97	1
ETIQUETEUSE KRONES LV2	4,82	5,00	0,96	0
ENCAISSEUSE KETTNER LV2	11,19	11,66	0,96	0
CONVOYEUR CASIERS LV2	8,29	8,66	0,96	0
DECAISSEUSE KETTNER LV2	4,80	5,02	0,96	0
DEPALETISEUR LV2	3,02	3,20	0,94	0
DATEUSE LV2	2,06	2,25	0,91	0
INSPECTRICE LV2	2,24	2,46	0,91	0
MIXEUR LV2	0,52	0,68	0,76	1

Cette solution nous permet de minimiser jusqu'à 63,56% du temps d'arrêt avec un montant de 298 800 .00 MAD.

2.5.3 Comparaison

Les résultats des quatre outils : Pareto, Pareto & sac à dos, Algorithme de Greedy et notre méthode PKPGA sont représentés dans le tableau suivant :

Tableau 31 : Le pourcentage du temps d'arrêt remédié et le coût d'intervention en MAD pour les quatre outils

	% Temps d'arrêt remédié	**Cout d'intervention en MAD**	**Valeur remédie en MAD**
Pareto	51.20	258500	92160
Pareto & Sac à dos	59.49	298400	107082
Algorithme de Greedy & Sac à dos	63.07	299900	113526
Notre méthodologie PKPGA	63.56	298800	114408

Dans notre étude, nous avons développé une nouvelle méthode de sélection d'éléments dans le but d'améliorer une ligne de production. Cette méthode combine les caractéristiques de trois outils de soutien à la décision : Pareto, Knapsack Problem et Greedy algorithm. Ce nouvel outil de soutien à la décision donne les résultats les plus optimaux par rapport à ceux obtenus en appliquant les trois méthodes mentionnées précédemment.

En outre, cet outil permet aux gestionnaires de la maintenance de prendre la décision la plus optimale pour améliorer l'efficacité de la production. Ceci est obtenu en réduisant la période de pannes de machines concernées et en respectant le coût budgétaire spécifié par la société.

Conclusion

Il existe de nombreuses méthodes et des algorithmes de prise de décision ; dans notre étude, nous nous sommes intéressés aux méthodes suivantes : Pareto, knapsack et Greedy algorithme. Notre étude de cas montre que la méthode Pareto permet de choisir les éléments les plus critiques basés sur un seul critère. Dans notre document, nous avons développé une nouvelle méthode basée sur la méthode Pareto que nous l'avons appelée CBA qui permet de choisir les éléments les moins critiques. Ces deux méthodes ont été appliquées à une étude de cas réel afin d'améliorer la maintenance industrielle dans la structure de production de boissons gazeuses.

Nous avons montré que la méthode CBA produit des résultats plus optimaux que la méthode Pareto, bien que la méthode Pareto soit souvent citée comme le meilleur choix dans le domaine de la maintenance industrielle. Enfin, nous avons développé un algorithme basé sur les deux méthodes choisies : Pareto et CBA. Cet algorithme fait une comparaison entre les résultats individuels des deux méthodes, en termes de réduction du taux d'échec et d'augmentation de la productivité, indique lequel des deux donne le résultat le plus optimal.

Ensuite, nous avons comparé trois méthodes : Pareto, problème de sac à dos (KP) et l'algorithme Greedy à fin de déterminer le meilleur. Pareto qui est la plus utilisée dans la maintenance, problème du sac à dos qui est la plus utilisée dans la gestion du transport des marchandises, et l'algorithme de Greedy qui est utilisé dans l'ordonnancement. Ces trois méthodes sont appliquées sur une étude de cas réel à fin d'améliorer la maintenance industrielle dans une structure de production des boissons gazeuses. Nous avons trouvé que l'algorithme de Greedy qui est destiné à l'ordonnancement donne des résultats plus optimaux que celle de Pareto avec problème de sac à dos (KP) qui est destiné à la gestion de transport de la marchandise. Et cette dernière donne des résultats plus optimaux que de la méthode de Pareto qui est la plus utilisée dans la maintenance industrielle.

Après, nous présentons une méthode qui combine les trois outils : Pareto, problème de Knapsack, KP et Algorithme de Greedy "PKPGA". Cette méthode a été approuvée par une étude de cas via l'utilisation des trois outils du prix de décision. Ce nouvel outil "PKPGA" permet aux responsables de la maintenance industrielle de mieux identifier les anomalies classées en fonction de leur impact négatif sur la production. Cette méthode limite les pertes de production, économisant du temps et de l'argent pour les entreprises.

Cette méthode peut être appliquée seulement si nous avons 2 variables. Mais la question qui se pose : si nous avons le nombre de variables supérieurs à 2 ? Pour cela nous envisageons de chercher une nouvelle méthode qui prendre en considération de trois variables ou plus.

Conclusion Générale

Ce rapport de thèse présente nos résultats de recherche scientifique qui s'inscrit d'une manière générale dans la problématique de diagnostic et maintenance des systèmes à événements discrets. Il est divisé en trois parties.

La première partie a présenté les systèmes à événements discrets qui sont discrets par le temps et par l'événement, le changement d'état d'un système à événements discrets se fait par l'avènement d'un nouvel événement, la partie présente, aussi, leurs applications. Ensuite, il a illustré le résultat d'une étude analytique et statistique qui montre la tendance de la recherche scientifique. En fait, cette étude a montré que la communauté scientifique se focalise sur le diagnostic des systèmes à événements discrets. D'après cette partie, nous nous sommes dirigés vers le diagnostic des systèmes à événements discrets.

La deuxième partie a présenté un état d'art sur le diagnostic des systèmes à événements discrets, il se base principalement sur les études réalisées dans les universités de Sidi Mohamed Ben Abdellah, de Reims et de Michigan. Après il a présenté un diagnostiqueur des systèmes à événements discrets qui se base sur le fait que tous les systèmes à événements discrets fonctionnent de façon cyclique, la durée de ce cycle peut être mesurée ou définie avec une marge d'erreur, ce qui nous a permis de définir les marges pour chaque événement du système. Ce diagnostiqueur vérifie si chaque événement s'exécute dans sa marge de temps ou non. Ce diagnostiqueur permet de diagnostiquer les systèmes avec des événements séquentiels, parallèles, à choix, et les systèmes à des événements non observables. Ce diagnostiqueur a été approuvé par des applications réelles. Le diagnostiqueur passe par les étapes suivantes : détections, isolation et puis identification des défauts, ce qui ramène à la phase de la maintenance. La maintenance de tous les éléments défectueux n'est pas toujours possible, pour cela il faut avoir des méthodes de prise de décision dans la maintenance industrielle.

La troisième partie a commencé par la présentation d'une nouvelle méthode de la prise de décision qui se base sur la méthode de Pareto, qui est suivie par une étude comparative entre ces deux méthodes afin de déterminer quand on utilise l'une ou l'autre, tout ça dans le but d'avoir une décision plus optimale. Ensuite une projection de ces deux

méthodes dans le domaine de la prise de décision en maintenance industrielle est expliquée. A la fin de la partie, il y a une nouvelle méthode qui se base sur les principes de la méthode de Pareto et de ces deux méthodes qui sont dédiées au domaine de transport et logistique. Cette nouvelle méthode donne de bons résultats, mieux que tous les autres. Ainsi, elle est approuvée par une étude de cas.

Ce travail peut être développé par :

- La mise à jour d'étude de la tendance de la recherche scientifique dans le domaine des systèmes à événements discrets.
- L'introduction de la logique floue dans le diagnostiqueur.
- L'élaboration d'une méthode de prise de décision qui prend en compte plus de deux variables.

Bibliographie

[1] C-G. Cassandras & S. Lafortune, "Introduction to discrete event systems," *New York: Springer,* vol. 2nd edn, 2008.

[2] J-E. Hopcroft & J-D Ullman, "Introduction to automata theory, languages and computation," *Addison-Wesley,* p. 1979.

[3] T. Murata, "Petri nets properties, analysis and applications," *Proc IEEE,* no. 77, p. 541–580, 1989.

[4] S. Balemi, G.J Hoffmann, P.G Wong-Toi & G.J Franklin, "Supervisory control of a rapid thermal multiprocessor," *IEEE Trans Autom Control,* no. 38, p. 1040–1059, 1993.

[5] B. Brandin, "The real-time supervisory control of an experimental manufacturing cell," *IEEE Trans Robot Autom,* no. 12, p. 1–14, 1996.

[6] E. Park, D-M. Tilbury & P-P. Khargonekar, "Modular logic controllers for machining systems: formal representations and performance analysis using Petri nets," *IEEE Trans Robot Autom,* no. 115, p. 1046–1061, 1999.

[7] V. Chandra, Z. Huang & R. Kumar, "Automated control synthesis for an assembly line using discrete event system theory," *IEEE Trans Syst Man Cybern Part C,* no. 1 33, p. 284–289, 2003.

[8] E-W. Endsley, E-E. Almeida & D-M. Tilbury, "Modular finite state machines: development and application to reconfigurable manufacturing cell controller generation," *Control Eng Pract,* no. 114, p. 1127–1142, 2006.

[9] K. Andersson, J. Richardsson, B. Lennartson & M. Fabian, "Coordination of operations by relation extraction for manufacturing cell controllers," *IEEE Trans Control Syst Technol,* no. 118, p. 414–429, 2010.

[10] R. David & H. Alla, "Petri nets and Grafcet: tools for modelling discrete event system," *Upper Saddle: Prentice-Hall,* 1992.

[11] R-W. Lewis, "Programming industrial control systems using IEC 1131-3," *The Institution of Electrical Engineers,* no. 998.

[12] N. Wightkin, U. Guy & H. Darabi, "Formal modeling of sequential function charts with time Petri nets," *IEEE Trans Control Syst Technol,* no. 119, p. 455–464, 2011.

[13] T. Alenljung, B. Lennartson & M-N. Hosseini, "Sensor graphs for discrete event modeling applied to formal verification of PLCs," *IEEE Trans Control Syst Technol,* no. 120, p. 1506–1521, 2012.

[14] L. Feng, W-M. Wonham & P-S. Thiagarajan, "Designing communicating transaction processes by supervisory control theory," *Formal Methods Syst Design,* no. 130, p. 117–141, 2007.

[15] M. Li & R. Kumar, "Model-based automatic test generation for Simulink/Stateflow using extended finite automaton," *Proceedings of the CASE, Seoul. IEEE,* 2012.

[16] R. Sampath, H. Darabi U. Buy & J. Liu, "Control reconfiguration of discrete event systems with dynamic control specifications," *IEEE Trans Autom Sci Eng,* no. 15, p. 84–100, 2008.

[17] A. Wassyng, M. Lawford & T. Maibaum, "Software certification experience in the Canadian muclear industry: lessons for the future," *in EMSOFT'11, Taipei,* 2011.

[18] M. Sampath, R. Sengupta, S. Lafortune & K. Sinnamohideen, "Failure diagnosis using disrcete event models," *IEEE Trans Control Syst Technol,* no. 14, p. 105–124, 1996.

[19] R. Kumar & S. Takai, "Decentralized prognosis of failures in discrete event systems," *IEEE Trans Autom Control,* no. 155, p. 48–59, 2010.

[20] J. Dubreil, P. Darondeau & H. Marchand, "Supervisory control for opacity," *IEEE Trans Autom Control,* no. 155, p. 1089–1100, 2010.

[21] A. Saboori & C-N. Hadjicostis, "Current-state opacity formulations in probabilistic finite automata," *IEEE Trans Autom Control,* no. 59, p. 120–133, 2014.

[22] A. Saboori & C-N. Hadjicostis, "Opacity-enforcing supervisory strategies via state estimator constructions," *IEEE Trans Autom Control,* no. 57, p. 1155–1165, 2012.

[23] W-M. Wonham, "Supervisory control of discrete event systems," *Electrical & Computer Eng, University of Toronto,,* 2006.

[24] S-A. Reveliotis, "Real-time management of resource allocation systems a discrete event systems approach," *New York: Springer,* 2005.

[25] J. Ezpeleta, J-M. Colom & J. Martinez, "A Petri net based deadlock prevention policy for flexible manufacturing systems," *IEEE Trans R&A,* no. 111, p. 173–184, 1995.

[26] S-A. Reveliotis & P-M. Ferreira, "Deadlock avoidance policies for automated manufacturing cells," *IEEE Trans Robot Autom,* no. 112, p. 845–857, 1996.

[27] M. Jeng, X. Xie & M-Y Peng, "Process nets with resources for manufacturing modeling and their analysis," *IEEE Trans Robot Autom,* no. 118, p. 875–889, 2002.

[28] S. Reveliotis & E. Roszkowska, "Conflict resolution in free-ranging multi-vehicle systems: a resource allocation paradigm," *IEEE Trans Robot,* no. 127, p. 283–296, 2011.

[29] A. Giua, M-P. Fanti & C. Seatzu, "Monitor design for colored Petri nets: an application to deadlock prevention in railway networks," *Control Eng Pract,* no. 10, p. 1231–1247, 2006.

[30] S-A. Reveliotis, "Conflict resolution in AGV systems," *IIE Trans,* vol. 7, no. 132, p. 647–659, 2000.

[31] D. Van & W. Aalst, "Verification of workflow nets," *Lecture notes in computer science,* vol. 1248, p. 407–426, 1997.

[32] H. Liao, Y. Wang, H-K. Cho, J. Stanley, T. Kelly & Lafortune , "Concurrency bugs in multithreaded software: modeling and analysis using Petri nets," *Discret Event Syst Theory Appl,* no. 123, p. 157–195, 2013.

[33] S. Reveliotis & A. Nazeem, "Deadlock avoidance policies for automated manufacturing systems using finite state automata," *Campos,* 2013.

[34] M. Zhou & M-P. Fanti, "Deadlock resolution in computer-integrated systems," *Marcel Dekker, Singapore,* 2004.

[35] S-A. Reveliotis, "Algebraic deadlock avoidance policies for sequential resource allocation systems," *Lahmar M (ed) Facility logistics: approaches and solutions to next generation challenges, Boca Raton: Auerbach Publications,* p. 235–289, 2007.

[36] Z. Li, M. Zhou & N. Wu, "A survey and comparison of Petri net-based deadlock prevention policies for flexible manufacturing systems," *IEEE Trans Syst Man Cybern,* no. 138, p. 173–188, 2008.

[37] D-P. Bertsekas, "Dynamic programming and optimal control," *Athena Scientific, Belmont,* vol. 1, no. 12, 1995.

[38] S. Meyn, "Control techniques for complex networks," *chez Cambridge University Press, Cambridge,* 2008.

[39] P. M, "Scheduling," *chez Prentice Hall, Upper Saddle River,* 2002.

[40] Y. Wardi & C-G. Cassandras, "Perturbation analysis of discrete event systems," *chez Kluwer Academic, Boston,* 1991.

[41] Y.C Ho & X-R Cao, "Perturbation analysis of discrete event systems," *chez Kluwer Academic, Boston,* 1991.

[42] X-R. Cao , "Stochastic learning and optimization," *sensitivity approach, New York: Springer,* 2007.

[43] C-G. Cassandras & S-G. Strickland, "Perturbation analytic methodologies for design and optimization of communication networks," *IEEE J Sel Areas Commun,* no. 16, p. 158–171, 1988.

[44] C-G. Cassandras, «Perturbation analysis and "rapid learning" in the control of manufacturing systems,» *Leondes CT (ed) Dynamics of discrete event systems,* vol. 51, p. 243–284, 1994.

[45] C-G. Panayiotou & C-G. Cassandras, "Optimization of kanban-based manufacturing systems," *Automatica,* vol. 35, p. 1521–1533, 1999.

[46] D. Homem, T. Mello, A. Shapiro & M-L. Spearman, "Finding optimal material release times using simulation-based optimization," *Manage Sci,* vol. 45, p. 86–102, 1999.

[47] M. Fu & X. Xie, "Derivative estimation for buffer capacity of continuous transfer lines subject to operationdependent failures," *Discret Event Syst Theory,* vol. 12, p. 447–469, 2002.

[48] T. Santoso, S. Ahmed, M. Goetschalckx & A. Shapiro, "A stochastic programming approach for supply chain network design under uncertainty," *Europ J Oper Res,* no. 167, p. 96–115, 2005.

[49] F. Baccelli, G. Cohen, G-J. Olsder & J-P. Quadrat, "Synchronization and linearity: an algebra for discrete event systems," *chez Wiley, New York,* 1992.

[50] J-H. Kim & T-E. Lee, "Feedback control design for cluster tools with wafer residency time constraints," *IEEE conference on systems, man and cybernetics, Seoul. IEEE,* p. 3063–3068, 2012.

[51] X-R. Cao, "Basic ideas for event-based optimization of Markov systems," *Discret Event System Theory,* vol. 15, p. 169–197, 2005.

[52] R. Li & S. Reveliotis, "Performance optimization for a class of generalized stochastic Petri nets," *chez IEEE conference on decision and control, Florence,* 2013.

[53] J. Markovski & R. Su, «Towards optimal supervisory controller synthesis of stochastic nondeterministic discrete event systems,» *chez IEEE conference on decision and control, Florence,* 2013.

[54] X. David-Henriet, L. Hardouin, J. Raisch & B. Cottenceau, "Optimal control for timed event graphs under partial synchronization," *chez IEEE conference on decision and control, Florence,* 2013.

[55] H. Alaoui Ismaili, M. Fri, M. Msaaf, O. Benatia, F. Belmajdoub, "Trend of Scientific research: Analysis of Mega-data of discrete event systems described by mathematical tools," *International Journal of Advanced Iformation Science Technology (IJAIST),* vol. 46, no. 46, 2016.

[56] M. Msaaf, M. Fri, H. Alaoui Ismaili, O. Benatia & F.Belmajdoub, "Analysis of mega database of discrete event systems described by formal language, automata and ladder," *International Journal of Advanced Iformation Science Technology (IJAIST),* vol. 48, no. 48, pp. 1-8, April 2016.

[57] O. Benatia, M. Fri, M. Msaaf, H. Alaoui Ismaili & F.Belmajdoub, "Trend of Scientific research Analysis of Mega-data of discrete event systems described by mathematical tools," *International Journal of Engineering Research,* vol. 5, no. 11, pp. 16-22, 2016.

[58] M. Sampath, R. Sengupta, S. Lafortune & D. Teneketzis, "Failure diagnosis using discrete event models," *IEEE Transactions on Control Systems Technology,* vol. 4, no. 2, pp. 105-124, March, 1996.

[59] R. Sengupta & S. Tripakis, "Decentralized diagnosability of regular languages is undecidable," *Proc. 40th IEEE Conf. on Decision and Control,* pp. 423-428, December, 2002.

[60] R. Debouk, S. Lafortune & D. Teneketzis, "Coordinated decentralized protocols for failure diagnosis of discrete-event systems," *Discrete Event Dynamic Systems : Theory and Applications,* vol. 10, no. 1-2, pp. 33-86, January, 2000.

[61] S. Genc & S. Lafortune, "A distributed algorithm for on-line diagnosis of place-bordered Petri nets," *Proc. of 16th IFAC World Congress,* July, 2005.

[62] A. Benveniste, S. Haar, E. Fabre & C. Jard, "Distributed and asynchronous discrete event systems diagnosis," *Proc. 41st IEEE Conf. on Decision and Control,* pp. 3742-374, December, 2003.

[63] D. Lefebvre & C. Delherm, "Diagnosis of DES with Petri net models," *IEEE Trans. Aut. Science and Eng,* vol. 4, no. 1, p. 114– 118, 2007.

[64] D. Lefebvre, "Firing sequences estimation in vector space over Z3 for ordinary Petri nets," *IEEE Trans. Syst. Man and Cyb. Part A,* vol. 38, no. 6, pp. 1325-1336, 2008.

[65] D. Lefebvre, "On-line fault diagnosis with partially observed Petri nets," *IEEE Trans Aut. Contr,* vol. 59, no. 7, pp. 1919-1924, July 2014.

[66] D. Lefebvre, "Fault diagnosis and prognosis with partially observed Petri nets," *IEEE Trans. Syst. Man and Cyb– Systems,* vol. 44, no. 10, pp. 1413 - 1424, October 2014.

[67] M. Fri & F. Belmajdoub, "Control by Supervision of Discrete Event System," *Journal of Computational Information Systems,* vol. 10, no. 18, pp. 7877-7891, 2014.

[68] Z. Huang, V. Chandra, S. Jiang & R. Kumar, "Modeling discrete event systems with faults using a rules based modeling formalism," *Mathematical Modeling of Systems,* vol. 1, no. 1, 1996.

[69] P. Caines, R. Greiner & S. Wang, "Classical and logic based dynamic observers for finite automata," *IMA Journal of Mathematical Control and Information,* vol. 8, p. 45–80, 1991.

[70] R. Cieslak, D. Desclaux, A. Fawaz & Varaiya, «Supervisory control of discrete-event processes with partial observations,» *IEEE Transactions Automatic Control,* vol. 33, p. 249–260, 1988.

[71] C-M. Ozveren & A-S. Willsky, "Observability of discrete event dynamic systems," *IEEE Transactions Automatic Control,* vol. 35, no. 7, p. 797–806, 1990.

[72] P.J. Ramadge, "Observability of discrete-event systems," *Proc. 25th IEEE conference decision and control,* p. 1108–1112, 1986.

[73] F. Lin, "Diagnosability of discrete event systems and its applications," *Discrete Event Dynamic Systems,* vol. 4, no. 1, p. 197–212, 1994.

[74] S. Bavishi & E. Chong, "Automated fault diagnosis using a discrete event systems framework," *Proc. 9th IEEE int. symp. intelligent contr,* p. 213–218, 1994.

[75] S. Velilla & M. Silva, "The SPY: a mechanism for safe implementation of highly concurrent systems," *Proc. 15th IFAC/IFIP workshop on real-time programming,* p. 75–81, 1988.

[76] J. Prock, "A new technique for fault detection using Petri nets," *Automatica,* vol. 27, p. 239–245, 1991.

[77] V-S. Sreenivas & M-A. Jafari, "Fault detection and monitoring using time Petri nets," *IEEE Transactions Systems, Man and Cybernetics,* vol. 23, p. 1155–1162, 1993.

[78] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohidden & D. Teneketzis, "Diagnosability of discrete event systems," *IEEE Transactions Automatic Control,* vol. 40, p. 1555–1575, 1995.

[79] F. Basile, P. Chiacchio & G-D Tommasi, "An efficient approach for online diagnosis of discrete event systems," *IEEE Transactions Automatic Control,* vol. 54, no. 4, p. 748–759, 2009.

[80] F. Basile, P. Chiacchio & G. De Tommasi, "Sufficient conditions for diagnosability of Petri nets," *Proc. 9th international workshop on discrete event systems (WODES'08),* p. 436–442, 2008.

[81] M-P. Cabasino, A-N. Giua & C. Seatzu, "Fault detection for discrete event systems using Petri nets with unobservable transitions," *Automatica,* vol. 46, no. 9, p. 1531–1539, 2010.

[82] M. Dotoli, M. Fanti, & A. Mangini, "Fault detection of DES by Petri nets and integer linear programming," *Automatica,* vol. 45, no. 11, p. 2665–2672, 2009.

[83] M. Fanti, A-M. Mangini & U. Walter, «Fault detection by labelled Petri nets and time constraints,» *Proc. 3rd international workshop on dependable control of Discrete Systems,* p. 170–175, 2011.

[84] S. Genc & S. Lafortune, "Distributed diagnosis of place-bordered Petri nets," *IEEE Transactions Automation Science and Engineering,* vol. 4, no. 2, p. 206–219, 2007.

[85] D. Lefebvre & C. Delherm, "Diagnosis of DES with Petri net models," *IEEE Transactions Automation Science and Engineering,* vol. 4, no. 1, p. 114–118, 2007.

[86] A. Ramirez-Trevino, E. Ruiz-Beltran, I. Rivera-Rangel & E. Lopez-Mellado, "Online fault diagnosis of discrete event systems. A Petri net-based approach," *IEEE Transactions Automation Science and Engineering,* vol. 4, no. 1, p. 31–39, 2007.

[87] A-M. Idghamishi & S-H. Zad, "Fault diagnosis in hierarchical discrete-event systems," *Proc. 43rd IEEE conference on decision and control,* p. 63–68, 2004.

[88] H. Alaoui-Ismaili & F. Belmajdoub, "Co-diagnosability approach of discrete event systems without," *International Journal Of Engineering Sciences & Research,* vol. 5, no. 10, pp. 113 - 123, 2016.

[89] M. Sampath, A. Godambe, E. Jackson & E. Mallow, "Combining qualitative and quantitative reasoning – A hybrid approach to failure diagnosis of industrial systems," *Proc. IFAC safeprocess conf,* p. 494–501, 2000.

[90] O. Contant, S. Lafortune & T. Teneketzis, "Diagnosis of intermittent faults," *Discrete Event Dynamic Systems,* vol. 14, no. 2, p. 171–202, 2004.

[91] S-H. Zad, R-H. Kwong & W-M. Wonham, "Fault diagnosis in discrete-event systems: Framework and model reduction," *IEEE Transactions Automatic Control,* vol. 48, no. 7, p. 1199–1212, 2003.

[92] M. Sayed-Mouchaweh, A. Philippot & V. Carré-Ménétrier, "Decentralized diagnosis by Boolean discrete event system model: Application on manufacturing systems," *International Journal of Production Research,* vol. 46, no. 19, p. 5469–5490, 2008.

[93] T. Jéron, H. Marchand, S. Pinchinat & M. Cordier, "Supervision patterns in discrete event systems diagnosis," *Proc. 8th international workshop on discrete event systems (WODES 2006),* 2006.

[94] D-N. Pandalai & L-E. Holloway, "Template languages for fault monitoring of timed discrete event processes," *IEEE Transactions Automatic Control,* vol. 45, no. 5, p. 868–882, 2000.

[95] M. Sayed-Mouchaweh, "Decentralized fault free model approach for fault detection and isolation of discrete event systems," *European Journal of Control,* vol. 18, no. 1, p. 82–93, 2012.

[96] M. Roth, J. Lesage & L. Litz, "The concept of residuals for fault localization in discrete event systems," *Control Engineering Practice,* vol. 19, p. 978–988, 2011.

[97] S. Schneider, L. Litz & M. Danancher, "Timed residuals for fault detection and isolation in discrete event systems," *Proc. 3rd international workshop on dependable control of discrete systems,* 2011.

[98] R-K. Boel & J-H. van Schuppen, "Decentralized failure diagnosis for discreteevent systems with costly communication between diagnosers," *Proc. 6th international workshop on discrete event systems (WODES'02),* 2002.

[99] H. Chakib & A. Khoumsi, "Multi-decision diagnosis: Decentralized architectures cooperating for diagnosing the presence of faults in discrete event systems," *Discrete Event Dynamic Systems,* vol. 22, no. 3, p. 333–380, 2012.

[100] R. Debouk, S. Lafortune & D. Teneketzis, «Coordinated decentralized protocols for failure diagnosis of discrete event systems,» *Discrete event dynamic systems,* vol. 10, p. 33–86, 2000.

[101] W. Qiu & R. Kumar, "Decentralized failure diagnosis of discrete event systems," *IEEE Transactions on Systems, Man and Cybernetics: Part A,* vol. 36, no. 2, p. 628–643, 2006.

[102] R. Sengupta, "Diagnosis and communication in distributed systems," *Proc. 4th international workshop on discrete event systems (WODES'98),* p. 144–151, 1998.

[103] E. Fabre, A. Benveniste, S. Haar & C. Jard, "Distributed monitoring of concurrent and asynchronous systems," *Discrete Event Dynamic Systems,* vol. 15, no. 1, p. 33–84, 2005.

[104] S. Genc & S. Lafortune, "Distributed diagnosis of place-bordered Petri nets," *IEEE Transactions Automation Science and Engineering,* vol. 4, no. 2, p. 206–219, 2007.

[105] Y. Pencolé & A. Subias, "A chronicle-based diagnosability approach for discrete timed-event systems: Application to web-services," *Journal of Universal Computer Science,* vol. 15, no. 17, p. 3246–3272, 2009.

[106] R. Su & W-M. Wonham, "A model of component consistency in distributed diagnosis," *Proc. 7th international workshop on discrete event systems (WODES'04),* pp. 427-432, 2004.

[107] D-N. Pandalai & L-E. Holloway, "Template languages for fault monitoring of timed discrete event processes," *IEEE Transactions Automatic Control,* vol. 45, no. 2, p. 868–882, 2000.

[108] Y. Pencolé & A. Subias, "A chronicle-based diagnosability approach for discrete timed-event systems: Application to web-services," *Journal of Universal Computer Science,* vol. 15, no. 17, p. 3246–3272, 2009.

[109] R. Debouk, R. Malik & B. Brandin, "A modular architecture for diagnosis of discrete event systems," *Proc. IEEE conference decision and control,* p. 417– 422, 2002.

[110] E. Fabre, S. Benveniste & C. Jard, "Distributed diagnosis for large discrete event dynamic systems," *Proc. IFAC world congress,* 2002.

[111] Y. Pencolé & M-O. Cordier, "A formal framework for the decentralized diagnosis of large scale discrete event systems and its application to telecommunication networks," *Artificial Intelligence,* vol. 1, no. 2, p. 121–170, 2005.

[112] W. Qiu & R. Kumar, "Distributed Diagnosis under bounded-delay communication of immediately forwarded local observations," *Proc. American control conference,* p. 1027–1032, 2005.

[113] L. Ricker & E. Fabre, "On the construction of modular observers and diagnosers for discrete event systems," *Proc. 39th IEEE conference decision and control,* p. 2240–2244, 2000.

[114] R. Su & W-M. Wonham, "Hierarchical fault diagnosis for discrete-event systems under global consistency," *Discrete Event Dynamic Systems,* vol. 16, p. 39–70, 2006.

[115] Y. Pencolé, "iagnosability analysis of distributed discrete event systems," *Proc. international workshop on principles of diagnosis (DX'04),* p. 173–178, 2004.

[116] Y. Wang, T-S. Yoo & S. Lafortune, "Decentralized diagnosis of discrete event systems using unconditional and conditional decisions," *Proc. 44th conference decision and control,* p. 6298–6304, 2005.

[117] O. Contant, S. Lafortune & D. Teneketzis, "Diagnosability of discrete event systems with modular structure," *Discrete Event Dynamic Systems,* vol. 16, no. 1, p. 9–37, 2006.

[118] S. Genc & S. Lafortune, "Predictability in discrete-event systems under partial observation," *Proc. 6th IFAC safeprocess symposium,* 2006.

[119] T. Jéron, H. Marchand, S. Genc & S. Lafortune, "Predictability of sequence patterns in discrete event systems," *Proc. IFAC world congress,* p. 537–543, 2008.

[120] R. Kumar & S. Takai, "Diagnosis prognosis of failures in discrete event systems," *Proc 9th international workshop on discrete event systems (WODES 2008),* p. 376–381, 2008.

[121] R. Kumar & S. Takai, "Decentralized prognosis of failures in discrete event systems," *IEEE Transactions Automatic Control,* vol. 55, p. 48–59, 2010.

[122] K-R. Rohloff, "Sensor failure tolerant supervisory control," *Proc. 44th Conference on decision and control and European control conference,* p. 3493–3498, 2005.

[123] J-C. Basilio & S. Lafortune, "Robust codiagnosability of discrete event systems," *Proc. American control conference,* p. 2202–2209, 2009.

[124] L-K. Carvalho, S-T. Lima, M-V. Moreira, J-C. Basilio & S. Lafortune , "Robust diagnosis of discrete-event systems against permanent loss of observations," *Automatica,* vol. 49, p. 223–231, 2013.

[125] L-K. Carvalho, J-C. Basilio & M-V. Moreira, "Robust diagnosis of discrete event systems against intermittent loss of observations," *Automatica,* vol. 48, p. 2068–2078, 2012.

[126] S. Takai, "Robust failure diagnosis of partially observed discrete event systems," *Proc. 10th international workshop on discrete event systems (WODES'10),* 2010.

[127] S. Takai, "Verification of robust diagnosability for partially observed discrete event systems," *Automatica,* vol. 48, p. 1913–1919, 2012.

[128] L-K. Carvalho, J-C. Moreira & M-V. Basilio , "Generalized robust diagnosability of discrete event systems," *Proc. IFAC world congress,* p. 8737–8742, 2011.

[129] F. De Lillo, F. Cecconi, G. Lacorata & A. Vulpiani, "Sedimentation speed of inertial particles in laminar and turbulent flows," *EPL,* vol. 84, 2008.

[130] [En ligne]. Available: http://www.supinfo.com/articles/single/809-pourquoi-aprendre-visual-basic.

[131] J-C. Laprie, Dependability basic concepts and terminologies, Springer Verlag edition, 1992.

[132] C. Brooks, "What Is a Pareto Analysis?," *Business News Daily Senior,* vol. 29, March , 2014.

[133] C. Renato, "The Economics of Vilfredo Pareto," *Routledge,* vol. 12, p. 148, 2012.

[134] M. Doostparast & N. Balakrishnan, "Pareto analysis based on records," *Journal of Theoretical and Applied Statistics,* vol. 47, no. 5, pp. 1075-1089, 2013.

[135] H. Kellerer, U. Pferschy & D. Pisinger, Knapsack problems, Spinger, 2004.

[136] M. Hifi, H. Mhallah & S. Sadfi, "Adaptive algorithms for the Knapsack problem," *European Journal of Industrial Engineering,* vol. 2, no. 2, pp. 134-152, 2008.

[137] X. Song, R. Lewis, J. Thompson & Y. Wu, "An incomplete m-exchange algorithm for solving the large-scale multi-scenario knapsack problem," *Computers & Operations Research,* vol. 39, no. 9, p. 1988–2000, 2012.

[138] P. Sharafi, H. Lip The & N. S. Muhammad Hadi, Conceptual design optimization of rectilinear building frames: A knapsack problem approach, Engineering Optimisation, Taylor& Francis, 2014.

[139] F. Furini, M. Iori, S. Martello & M. Yagiura, "Heuristic and exact algorithms for the interval min-max regret knapsack problem," *INFORMS Journal onComputing,* 2015.

[140] B. Diop, D. Diongue & O. Thiaré, "Greedy Algorithms for Target Coverage Lifetime Management Problem in Wireless Sensor Networks," *International Journal of Control and Automation,* vol. 8, no. 2, pp. 232-250, 2015.

[141] Z-Q. Zou, Z-T Li, S. Shen & R-C. Wang, "Energy-Efficient Data Recovery via Greedy Algorithm for Wireless Sensor Networks," *International Journal of Distributed Sensor Networks,* 2015.

[142] L. Turner, M. Ehrgott & H-W. Hamacher, "On the generality of the greedy algorithm for solving matroid base problems," *Discrete Applied Mathematics,* 2014.

[143] F. Glover, "Advanced greedy algorithms and surrogate constraint methods for linear and quadratic knapsack and covering problems," *European Journal of Operational Research,* vol. 230, no. 2, pp. 212-225, 16 October 2013.

[144] A. Boukili, M. Fri, M. El Hammoumi & F. Belmajdoub, "Elaboration of a new method in the science of decision," *International Journal of Scientific and Engineering Research,* vol. 7, no. 4, 2016.

[145] A. Boukili, M. Fri, M. El Hammoumi & F. Belmajdoub, "Comparative study of Pareto, Knapsack and Greedy Algorithm in the field of industrial Maintenance," *International Journal of Scientific and Engineering Research,* vol. 7, no. 4, 2016.

[146] M. Fri, A. Boukili, F. Belmajdoub, M. El Hammoumi, "Developing a new method “PKPGA” by using a combination of the ABC method Knapsack Problem and Greedy algorithm as a tool for decision support," *International Journal of Advanced Engineering, Management and Science (IJAEMS),* vol. 2, no. 7, pp. 1153 - 1159, 2016.

Résumé de la thèse

Le travail présenté dans ce mémoire s'inscrit de manière générale dans la problématique de diagnostic et maintenance des Systèmes à Evénements Discrets (SED). La première phase de recherche consiste à élaborer une étude d'état d'art qui montre que la communauté s'intéresse de plus en plus au diagnostic des SED. Ensuite, une étude comparative entre les méthodes de diagnostic est menée. Ces études nous a motivé à élaborer un diagnostiqueur performant afin d'améliorer la productivité des systèmes. Nous nous sommes basés sur l'approche que chaque système fonctionne de façon cyclique, ce cycle peut être quantifié en temps avec une marge d'erreur donnée par le constructeur. Notre diagnostiqueur vérifie si chaque étape se déroule dans sa marge de temps permise. Si ce n'est pas le cas, nous devons procéder à la maintenance industrielle. En effet, nous avons élaboré des algorithmes de choix des éléments les plus critiques à la fin de la thèse.

Mots clés : Systèmes à Evénements discrets ; Diagnostic ; Algorithme ; Maintenance Industrielle ; Science de Prise de Décision.

Summary of the thesis

The work presented in this thesis is generally enrolled in the diagnosis and maintenance of Discrete Events Systems (DES). The first phase of research consists of developing a state-of-the-art study. In fact, this study shows that the community is increasingly interested in the diagnosis of DES. Then, a comparative study between the diagnosis methods is conducted. These studies motivated us to elaborate a high-performance diagnoser in order to improve the systems productivity. We relied on the approach that any system operates in a cyclical way; this cycle can be quantified in time with an error margin given by the manufacturer. Our Diagnoser checks whether each step takes place within its allowed time range. If this is not the case, then we have to proceed to the industrial maintenance. Indeed, we have developed algorithms for choosing the most critical elements at the end of the thesis. In effect, at the end of the thesis, we have developed algorithms on the choice of the most critical elements.

Keywords: Discrete Event Systems; Diagnostic; Algorithm; Industrial maintenance; Science of Decision Making.